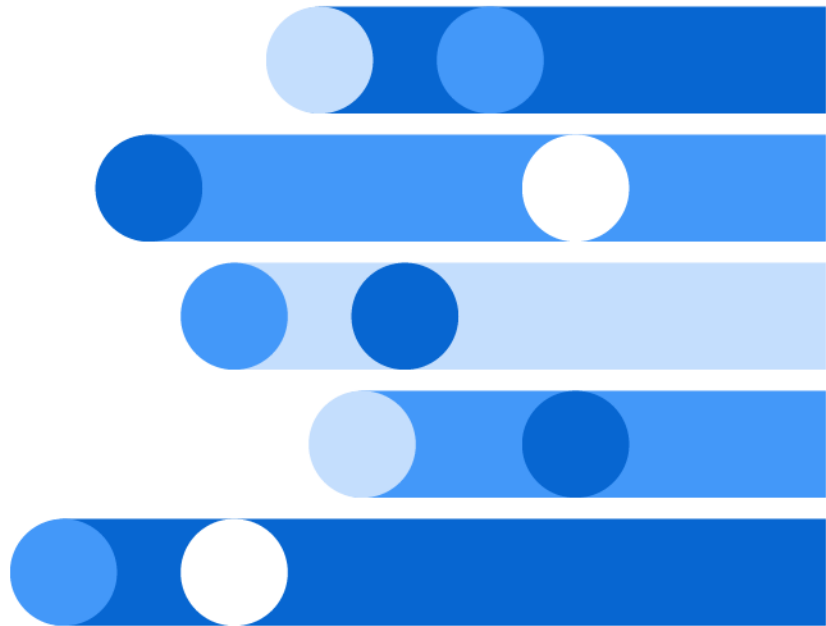




SAS[®] 各国語サポート (NLS): リファレンスガイ ド

2020.1 - 2026.02*



* このドキュメントは、ソフトウェアの追加バージョンに適用される場合があります。このドキュメントを [SAS Help Center](#) で開き、バナーのバージョンをクリックすると、使用できるすべてのバージョンが表示されます。



SAS[®] ドキュメント
2026 年 4 月 14 日

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2020. *SAS®各国語サポート(NLS): リファレンスガイド*. Cary, NC: SAS Institute Inc.

SAS®各国語サポート(NLS): リファレンスガイド

Copyright © 2020, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

For a hard copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

February 2026

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

v_001-P1:nlsref

目次

SAS 言語の構文規則	ix
-------------------	----

1 部 各国語サポートの概念 1

1 章 / 各国語サポート(NLS)	3
各国語サポートの概要	3
地域化と国際化の定義	4
2 章 / 各国語サポート関連のロケール	5
各国語サポート関連のロケール概念の概要	5
ロケールの指定	6
3 章 / 各国語サポート関連のエンコーディング	11
概要: 各国語サポート関連のエンコーディング	12
エンコーディングとトランスコーディングの違い	12
よく用いられるエンコーディング方法	13
各国語サポートエンコーディング関連の標準組織	16
照合順序	17
SAS データセットのエンコーディングの確認	21
デフォルトの SAS セッションエンコーディング	22
SAS セッションエンコーディングの設定	23
SAS セッションのエンコーディングの動作	25
UTF-8 へのデータの移行	27
WLATIN1 から UTF-8 へのデータの移行	27
4 章 / 各国語サポート関連のトランスコーディング	33
トランスコーディングの概要	33
トランスコーディングが必要となる場合	34
トランスコーディングと変換テーブル	34
SAS データのトランスコーディングを指定する SAS オプション	36
トランスコーディングの留意点	37
互換性があるエンコーディングと互換性がないエンコーディング	38
トランスコーディング中の切り捨てとデータ損失の問題のトラブ ルシューティング	43
CVP エンジンの使用による文字データ切り捨ての回避	44
%COPY_TO_NEW_ENCODING マクロを使用した文字の切り捨ての回避	49
KPROPDATA 関数を使用した CEDA トランスコーディング中の文 字データ損失の回避	50

5 章 / 2 バイト文字セット(DBCS)	53
DBCS サポートの概要	53
DBCS でサポートされる言語	54
DBCS の指定	54
DBCS 文字セットの表示要件	54
DBCS の機能	55
汎用機版 SAS での DBCS の利用法	55
DBCS エンコーディングの SAS データ変換	56
DBCS 文字列の取り扱い方	57

6 章 / タイムゾーン	59
SAS タイムゾーンの概要	59
SAS でのタイムゾーンの指定	59

2 部 各国語サポート関連の自動呼び出しマクロ 69

7 章 / 各国語サポート関連の自動呼び出しマクロのディクショナリ	71
カテゴリ別の自動呼び出しマクロエントリ	71
ディクショナリ	72

3 部 各国語サポート関連のデータセットオプション 81

8 章 / 各国語サポート関連のデータセットオプションのディクショナリ	83
カテゴリ別のデータセットオプション	83
ディクショナリ	84

4 部 各国語サポート関連の出力形式 89

9 章 / 各国語サポート関連の出力形式の概要	91
国際化対応の日付、日時出力形式	91
通貨表現	92
日付と時間のデフォルト幅の例外	100
統一パターンを使用する NL 出力形式	102
10 章 / 各国語サポート関連の出力形式のディクショナリ	107
各国語サポート関連の出力形式のカテゴリ	112
ディクショナリ	143

5 部 各国語サポート関連の関数 553

11 章 / SAS 文字列関数の国際化の互換性	555
SAS 文字列関数の国際化の互換性	555
12 章 / 関数での VARCHAR サポート	579
文字列関数の VARCHAR データ型	579
13 章 / 各国語サポート関連の関数のディクショナリ	583
カテゴリ別の関数	584
ディクショナリ	591

6 部 各国語サポート関連の入力形式 755

14 章 / 各国語サポート関連の入力形式のディクショナリ	757
カテゴリ別の入力形式	759
ディクショナリ	773

7 部 各国語サポート関連のマクロ関数 933

15 章 / 各国語サポート関連のマクロ関数のディクショナリ	935
カテゴリ別のマクロ関数	935
ディクショナリ	936

8 部 各国語サポート関連のシステムオプション 955

16 章 / 各国語サポート関連のシステムオプションのディクショナリ	957
カテゴリ別のシステムオプションエントリ	958
ディクショナリ	959

9 部 各国語サポート関連のコマンド、ステートメント、プロシジャオプション 997

17 章 / 各国語サポート関連のコマンド、ステートメントおよびプロシジャオプションのディクショナリ	999
カテゴリ別の各国語サポート関連のコマンド、ステートメント、およびプロシジャオプション	1000
ディクショナリ	1001

10 部 各国語サポート関連のプロシジャ 1041

18 章 / DBCSTAB プロシジャ	1043
概要: DBCSTAB プロシジャ	1043
構文: DBCSTAB プロシジャ	1044
例: DBCSTAB プロシジャ	1045
19 章 / LOCALEDATA プロシジャ	1051
概要: LOCALEDATA プロシジャ	1051
構文: LOCALEDATA プロシジャ	1051
例: LOCALEDATA プロシジャ	1060
20 章 / TRANTAB プロシジャ	1067
概要: TRANTAB プロシジャ	1068
概念: TRANTAB プロシジャ	1069
構文: TRANTAB プロシジャ	1072
例: TRANTAB プロシジャ	1079

11 部 ロケール、エンコーディングおよびトランスコーディングの値
1095

21 章 / LOCALE=システムオプションの値	1097
ロケールからエンコーディングへのマッピング	1097
22 章 / DBCS データ処理に関連する SAS システムオプション	1115
DBCS の SAS セッションで使用されるシステムオプションの概要	1115
SAS セッションの DBCS 値	1115
23 章 / SAS 言語要素中のエンコーディング値	1117
エンコーディング値を使用する SAS 言語要素の概要	1117
データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値	1118
24 章 / SAS セッションのエンコーディング値	1131
LINUX のエンコーディング値	1131

12 部 付録 1133

付録 1 / エンコーディング名と別名	1135
エンコーディングとその別名とエンコーディング文字セットの互換性	1135
付録 2 / タイムゾーン ID とタイムゾーン名	1145
エリア: Africa(アフリカ)	1145
エリア: America(アメリカ-北、中央、および南)	1148
エリア: Antarctica(南極)	1159
エリア: Arctic(北極)	1160

エリア: Asia(アジア)	1160
エリア: Atlantic(大西洋)	1165
エリア: Australia(オーストラリア)	1166
エリア: その他	1168
エリア: Europe(ヨーロッパ)	1174
エリア: Pacific(太平洋)	1179
付録 3 / TK_NLS_REPLACE_ICVEXT	1183

SAS 言語の構文規則

SAS 言語の構文規則の概要

SAS 言語要素の構文のドキュメントでは、標準の規則が使用されています。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
- 書体に関する規則
- 特殊文字
- SAS ライブラリや外部ファイルへの参照

構文の構成要素

ほとんどの言語要素の構文のコンポーネントには、キーワードと引数があります。キーワードのみ必要な言語要素もあります。また、一部の言語要素では、キーワードの後に等号(=)を付加する必要があります。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。

keyword

プログラムを記述するときに使用する SAS 言語要素の名前を指定します。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*)

一部の SAS ステートメントの構文は、引数のない 1 つのキーワードで構成されます。

DO;

... SAS code ...

END;

一部のシステムオプションでは、2 つあるキーワード値から 1 つを指定する必要があります。

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

数値または文字の定数、変数、式のいずれかを指定します。引数は、キーワードまたはキーワードの後の等号記号の後に続きます。SAS では、言語要素を処理するために引数を使用されます。引数は必須の場合と、省略可能な場合があります。構文では、オプションの引数は山かっこ (<>) で囲みます。

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*)

各引数には値があります。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 <options> <argument-2 <options> ...>

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-options)>
 <value-range-set-1 <(picture-1-options)>
 <value-range-set-2 <(picture-2-options)> ...>;

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。

書体に関する規則

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。

UPPERCASE BOLD

関数またはステートメントの名前など、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>;

UPPERCASE

リテラルである引数を表します。

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 |

italic

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、次の特殊文字を使用できます。

=

等号記号は、システムオプションなどの一部の言語要素内のリテラルの値を示します。

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*

<>

山かっこは省略可能な引数を示します。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に排他的です。

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' or "value"

単一引用符や二重引用符付きの引数は、その値にも単一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 ルーチンの終了を示します。

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 ライブラリと外部ファイルへの参照

SAS ステートメントおよびその他の言語要素の多くは、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。SAS ライブラリと外部ファイルを参照する方法は複数あり、一部の方法は動作環境によって異なります。

SAS ドキュメント中の外部ファイルを使用する例では、*file-specification* という語句を斜体で使用しています。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';
```


1 部

各国語サポートの概念

1 章	各国語サポート(NLS)	3
2 章	各国語サポート関連のロケール	5
3 章	各国語サポート関連のエンコーディング	11
4 章	各国語サポート関連のトランスコーディング	33
5 章	2 バイト文字セット(DBCS)	53
6 章	タイムゾーン	59

各国語サポート(NLS)

各国語サポートの概要	3
地域化と国際化の定義	4

各国語サポートの概要

各国語サポートは、ソフトウェアを各国の市場向けに正しく動作させるために必要な機能の集まりです。SAS には、地域の言語規則に準拠した SAS アプリケーションを確実に作成できるようにする、各国語サポート機能が組み込まれています。通常、英語で作成されるソフトウェアは、英語を話し、米国内で使用される規則に従ってフォーマットされたデータを使用するユーザーにとって使いやすくできています。ただし、これらのソフトウェアは、各国語サポートがなければ、世界の他の地域のユーザーにとっては使いにくい製品となります。SAS の各国語サポートでは、アジアやヨーロッパなどの地域のユーザーが、自国の言語および環境の下でデータを正常に処理できます。

各国語サポートは、マシン間を移動するデータに適用されます。たとえば、各国語サポートにより、ターゲットマシンで使用される正しい形式にデータを確実に変換できます。

テキスト文字列の操作は、言語および地域の SAS 設定の影響を受けます。このアクションにより、文字の大文字化と小文字化、文字の分類、データのスキャンなどの操作による結果を修正できます。SAS には、特定の国または国のグループに固有の文字である各国語文字を適正に表示および出力するための機能が組み込まれています。

各国語サポートを組み込んだソフトウェアアプリケーションにより、ソフトウェアの機能に関して、言語固有または文化固有の規則に依存することを回避できます。いくつかの例を示します。

- 文字列の操作
- 文字の分類
- 文字の比較ルール
- エンコーディングの文字セット

- 日時出力形式
- ユーザーインターフェイス
- メッセージテキスト言語
- 数値および通貨の出力形式
- 並べ替え順序

地域化と国際化の定義

国際化は、1つの言語またはロケールに基づくという前提なしで行われる、ソフトウェア製品の設計処理で、地域化を容易にします。国際化により、文字列の並べ替えと日付、時間、数、通貨のフォーマティングのルールを含む、国際的な規則がサポートされます。また、一貫したユーザーエクスペリエンスを製品のどの言語版でも可能にします。国際化の略称はI18Nです。

アプリケーションのロジックが文化的規則(特定の地域の通貨や数値の出力形式など)をサポートするとしても、地域化されたバージョンのソフトウェアのみが現地語のユーザーインターフェイスとシステムメッセージを提供します。

地域化は、特定の対象となる環境や市場の言語、文化、その他の要件に製品を適合させる処理で、これによりユーザーは、その製品の使用時に、現地の言語と規則で作成された結果を参照できます。ユーザーインターフェイス、システムメッセージ、ドキュメントの翻訳も地域化に含まれます。

SAS 各国語サポート(NLS)の機能を使用して、SAS アプリケーションを地域化や国際化することができます。詳細については、“[SAS 文字列関数の国際化の互換性](#)”を参照してください。

各国語サポート関連のロケール

各国語サポート関連のロケール概念の概要	5
ロケールの指定	6
SAS 起動時のロケールの指定法	6
SAS セッション中のロケールの指定法	7
言語切り替え	8

各国語サポート関連のロケール概念の概要

ロケールには、各地域の言語、データのフォーマティングなどの地域規則および文化が反映されています。この地域規則には、その国や地域の日付、時刻、数字のフォーマティングや通貨記号が含まれます。ページ順、用紙サイズ、住所、電話番号などもロケールに含まれます。

日付には、文化圏で受け入れられた地域規則に応じて、多くの表示形式があります。月は、数字または名前で表記される場合があります。名前は、完全に入力するか、略称が使用される場合があります。月、日および年の順序は、ロケールによって異なる可能性があります。

たとえば、"2020 年の 10 月の 3 日目"はロケールごとにさまざまな方法で表示されます。

英語

2020/10/03

ドイツ語

03.Oktober 2020

イタリア語

03 ottobre 2020

簡体字中国語

2020 年 10 月 03 日

言語はロケールの一部ですが、いずれか 1 つのロケールに固有ではありません。たとえばポルトガル語は、ポルトガルだけでなくブラジルでも話されていますが、それぞれの文化は異なります。ブラジルとポルトガルでは、データの表示形式に類似点があります。数字を表示するときに、整数部と小数部を区切るにはカンマ(,)を使用し、基数文字の左側にある桁を区切るにはドット(.)を使用します。ただし、2 つの異なるロケールで使用される通貨記号など、大きな違いもあります。ポルトガルではユーロを使っており、ユーロ記号(€)が必要です。一方、ブラジルではリアルを使っており、2 文字の通貨記号である R\$ で表します。

また、1 つの国で複数の公用語が使用されている場合もあります。カナダでは、英語とフランス語の 2 つが公用語として使用されています。LOCALE=システムオプションには、English_Canada および French_Canada という 2 つの値を指定できます。

通貨などの数値にもさまざまな表示形式があります。たとえば、小数点や基点文字をドット(.)で表す地域もあれば、カンマ(,)で表す地域もあります。3 桁ごとの区切りは、地域によって、ドット、カンマ、さらには空白などさまざまです。貨幣の表記規則もロケールごとに異なります。たとえば、金額にドル記号(\$)や円記号(¥)を付けることもあります。

用紙サイズおよび測定単位もロケールの考慮事項に含まれます。標準の用紙サイズには、レター(8-1/2×11 インチ用紙)および A4 (210×297 ミリメートル用紙)があります。レター用紙サイズは、主に北米の国々で使用されています。A4 はその他のほとんどのロケールで使用されています。ほとんどのロケールではセンチメートルを使用していますが、一部のロケールではインチを使用します。

時間は、一部の国または地域では 12 時間表記を使用して表示されます。別の国では、24 時間表記を使用して時間値が表示されています。

ロケールの指定

SAS 起動時のローケルの指定法

LOCALE=システムオプションを使用すると、SAS 起動時に SAS セッションのロケールを指定できます。LOCALE=によって、次の SAS システムオプションも暗黙的に設定されます。

- DATESTYLE=
- DFLANG=
- ENCODING=
- LOCALEDATA
- MAPEBCDICTOASCII
- ODSLANGCHG
- PAPERSIZE=

- RSASIOTRANSERROR
- TIMEZONE
- URLENCODING

注: ロケールは、POSIX 命名規則を使用しても指定できます。たとえば、POSIX の en_US は、SAS 値の English_UnitedStates に相当します。

LOCALE=オプションのデフォルト値は、各動作環境で同じです。詳細については、[表 21.1](#) を参照してください。

LOCALE=に English_UnitedStates の値を設定すると、SAS 起動時に、次のオプションが指定された Windows デフォルト値に暗黙的に設定されます。

- DATESTYLE=MDY
- DFLANG=English
- ENCODING=WLATIN1
- PAPERSIZE=Letter

起動時に、明示的に設定されたシステムオプションは暗黙的に設定されたオプションより優先されます。

起動時に、明示的に設定された PAPERSIZE=A4 は、LOCALE=オプションを用いて暗黙的に設定された PAPERSIZE=オプションより優先されます。詳細については、["PAGESIZE= システムオプション" \(SAS システムオプション: リファレンス\)](#)を参照してください。

SAS セッション中のロケールの指定法

このコードを使用してロケールオプションを指定できます。この例では、posix 値 da_DK を使用して、デンマーク、フェロー諸島、およびグリーンランドのロケールを指定しています。

```
option locale=da_DK;
data test;
  x=put(-1234.56789,nlmlndkk32.2);
  put x;
run;
```

このコードは、デンマーク、フェロー諸島、およびグリーンランドのロケールを指定し、デンマーク、フェロー諸島、およびグリーンランドのローカル NL 出力形式、nlmlndkk で-1234.56789 を処理します。

```
x=1,234.57 .,|.-
```

LOCALE=システムオプションを使用すると、SAS セッション中に SAS セッションのロケールを指定できます。ただし、変更された LOCALE=の値を反映して暗黙的に変更されるのは、次のシステムオプションの値のみです。

- DFLANG=
- PAPERSIZE=

注: PAPERSIZE オプションが影響を受けるのは、これを LOCALE に設定した場合に限ります。詳細については、“[PAGESIZE= システムオプション](#)” ([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。

ENCODING=システムオプションの値が、変更された LOCALE=の値を反映して暗黙的に変更されることはありません。

LOCALE=オプションに Italian_Italy という値を割り当てると、次のオプションが、変更された LOCALE=システムオプションの値を反映して、SAS セッション中に暗黙的に再設定されます。

- DFLANG=Italian
- PAPERSIZE=A4

言語切り替え

SAS メッセージは、起動中に SAS 構成ファイルの設定で指定された言語で表示されます。言語切り替え機能を使用して別の言語で SAS メッセージを表示できます。言語切り替え機能には、ODSLANGCHG システムオプションを使用してアクセスできます。ODSLANGCHG が有効である場合、LOCALE システムオプションの値によって、プロシジャの出力、ユーザーインターフェイス要素および ODS フォントの言語が決まります。ODSLANGCHG が無効な場合、メッセージは起動中に設定された言語で表示されます。この機能はどのサーバーでも、メッセージを構成する文字がサポート対象のエンコーディングである限り、使用可能です。詳細については、“[ODSLANGCHG システムオプション](#)”を参照してください。SAS 起動時に LOGLANGCHG オプションまたは ODSLANGCHG オプションを設定する際に、LSWLANG システムオプションにより、言語切り替え機能で使用する言語を指定します。LSWLANG システムオプションは、LOCALE システムオプションよりも優先度が高くなっています。LSWLANG オプションの値が有効な SAS 言語に指定された場合、SAS ログ出力は、その値により制御されます。それ以外の場合、SAS ログの言語は LOCALE=オプションにより決定されます。

LOGLANGCHG システムオプションは、SAS ログメッセージに関して言語切り替えを行うかどうかを制御します。このオプションは、SAS ログ出力におけるメッセージ切り替えの言語を制御します。LOGLANGCHG を指定した場合、SAS ログの言語は、LSWLANG オプションまたは LOCALE=により決定されます。詳細については、“[LOGLANGCHG システムオプション](#)”を参照してください。

LSWLANG オプションは、LOGLANGCHG または ODSLANGCHG が有効である場合に、メッセージの言語を指定します。LSWLANG の値が LOCALE に設定されている場合、LOCALE=オプションにより、切り替え用の言語が決定されます。デフォルトは LSWLANG=LOCALE です。詳細については、“[LSWLANG システムオプション](#)”を参照してください。

LOGLANGENG はトグルオプションであり、LOGLANGCHG および LSWLANG をオーバーライドし、これらの値をそれぞれ LOGLANGCHG=ON および LSWLANG=EN に設定します。その結果、LOG 出力は英語になります。NL 出力形式は変わりません。このオプションにより、ODSLANGCHG の設定が変更されます。ODSLANGCHG=OFF であれば、ODS 出力のシステムメッセージ言語は SAS の設定によって決まります。ODSLANGCHG=ON の場合、LSWLANG の値が English に設定されるため、すべての

メッセージが英語になります。詳細については、“[LOGLANGENG システムオプション](#)”を参照してください。

言語切り替えについて詳しくは、[Multilingual Computing with SAS® 9.4](#) を参照してください。

3

各国語サポート関連のエンコーディング

概要: 各国語サポート関連のエンコーディング	12
エンコーディングとトランスコーディングの違い	12
よく用いられるエンコーディング方法	13
各国語サポートエンコーディング関連の標準組織	16
照合順序	17
照合順序の概要	17
要求の代替照合順序	18
変換テーブルの指定	19
エンコーディング値の指定	20
言語照合の指定	20
SAS データセットのエンコーディングの確認	21
PROC CONTENTS を使用して SAS データセットの内容を表示する	21
デフォルトの SAS セッションエンコーディング	22
SAS セッションエンコーディングの設定	23
SAS セッションのエンコーディングの動作	25
SAS リリース別のエンコーディングサポート	25
出力処理	25
入力処理	26
外部ファイルの読み込みと書き込み	26
UTF-8 へのデータの移行	27
WLATIN1 から UTF-8 へのデータの移行	27

概要: 各国語サポート関連のエンコーディング

エンコーディングは、文字セットの各文字を一意的な数値表現にマッピングし、すべてのコードポイントからなるテーブルを構成します。コードページと呼ばれるこのテーブルは、順序に従って並べられた文字セットで、各文字に数値のインデックス(コードポイント値)が関連付けられます。コードページ上の文字の位置によって、2桁の16進数が決まります。

文字セットは、文字と記号のセットで、言語または言語のグループで使用されます。文字セットには、各国語文字(特定の国または国のグループに固有の文字)、特殊文字(句読点など)、アクセント記号のないラテン文字 A から Z、数字 0 から 9、コンピューターに必要なコントロール文字が含まれます。

エンコーディング方法は、数値表現を文字のセットに割り当てる一連のルールです。これらのルールによって、エンコーディングのサイズ(文字の数値表現の保存に使用されるビット数)と文字が表示されるコードページの範囲が決まります。エンコーディング方法は、コンピューター業界で開発された標準に従って定義されています。多くの場合、エンコーディング方法は、コンピューターハードウェアベンダーに固有です。

エンコーディングは、エンコーディング方法を文字セットに適用した結果です。

個々の文字には、使用するコードページによって、コードページの異なる位置が割り当てられます。たとえば、ユーロ記号(€)などがあります。ユーロ記号の Unicode コードポイントは U+20AC です。

各 SAS セッションは、デフォルトのエンコーディングに設定されます。これは、さまざまな SAS 言語要素を使用して指定できます。

エンコーディングとトランスコーディングの違い

エンコーディングは、SAS セッションのデフォルトの作業環境を確立します。たとえば、Windows LATIN1 エンコーディングは、ドイツでのドイツ語の de_DE ロケールなど、西ヨーロッパロケールの Windows 環境での、SAS セッションのエンコーディングです。大文字の Ä の Windows LATIN1 コードポイントは C4 16 進数です。

ただし、国際的な環境で作業していて、別のエンコーディングでエンコードされた SAS データにアクセスする場合、Ä は別のエンコーディングの C4 16 進数にマップ

されない場合があります。エンコーディング間で文字が一致しない場合は、データをトランスコードする必要があります。

注: デフォルトのエンコーディングは、動作環境とロケールにより異なります。UTF-8 が推奨されるエンコーディングになりつつあります。

トランスコーディングとは、1 つのエンコーディングから別のエンコーディングにデータを変換する処理のことです。概念的な情報については、[4 章, “各国語サポート関連のトランスコーディング”](#)を参照してください。

よく用いられるエンコーディング方法

エンコーディングは、エンコーディング方法をコード化された文字セットに適用した結果です。

エンコーディング方法 エンコードされた文字体系を生成するために、制定された業界ルールをコード化された文字に適用すること。そのようなルールでは、特定の文字の数値表現およびエンコーディングでのそのコード位置を保存するために必要とされるビット数が規定されます。ISO 2022 と UTF-8 は、エンコーディング方法の一例です。詳細については、“[各国語サポートエンコーディング関連の標準組織](#)”を参照してください。

エンコーディング方法は、数値表現を文字のセットに割り当てる一連のルールです。これらのルールによって、エンコーディングのサイズ(文字の数値表現の保存に使用されるビット数)と文字が表示されるエンコーディングの範囲が決まります。

次に、一般的なエンコーディング方法の一覧を示します。

Unicode

Unicode は一般的でよく使用されている文字セットです。SAS Viya プラットフォーム 4.0.1 は、Unicode 13.0 文字をサポートします。

最大で 143,859 文字の組み合わせを提供します。基本的に Unicode は、世界中のすべての言語に対応できます。Unicode はコード化された文字セットです。以下に挙げるのは Unicode 文字セットのエンコーディングです。

UTF-8

ラテンスク립ト言語、ギリシャ語、キリル文字、アラビア語、ヘブライ語を含む MBCS エンコーディング。日本語、中国語および韓国語などの東アジアの言語も含まれます。UTF-8 の文字の幅は、1 から 4 バイトまでです。UTF-8 では、1 から 128 のコード位置に ASCII 文字を残すことによって、ASCII との互換性を維持しています。

UTF-8 は最も一般的なエンコーディングです。テキストを UTF-8 エンコーディングとして保存すると、従来のエンコーディングに保存するよりも多くのスペースが必要になる場合があります。拡張量は、言語とテキストによって異なります。いくつかの一般的な従来のエンコーディングのいくつかの可能な拡張は次のとおりです。

表 3.1 UTF-8 でのストレージサイズの増加の例

エンコーディング	言語	UTF-8 でのストレージサイズの増加
ASCII	英語、マレー語	0%
ISO-8859-1	西ヨーロッパ	10%
ISO-8859-7、プレーンテキスト	ギリシャ語	90%
ISO-8859-7、50%マークアップ	ギリシャ語	45%
TIS-620、プレーンテキスト	タイ語	190%
TIS-620、50%マークアップ	タイ語	95%
EUC-KR、プレーンテキスト	韓国語	50%
EUC-KR、50%マークアップ	韓国語	55%

フットノート: W3C 国際化(I18n)アクティビティから 適応

UTF-16

現代のすべての記述体系に含まれる共通文字のほとんどを包含する 16 ビット形式です。ほとんどの文字は一樣に 2 バイトを使用して表現されますが、4 バイトを必要とする追加文字のために、サロゲートスペースと呼ばれる拡張スペースがあります。

UTF-32

各文字が 4 バイトを占める 32 ビット形式です。

ASCII (American Standard Code for Information Interchange、ASCII コード、情報交換用米国標準コード)

128 文字の組み合わせを提供する米国の 7 ビットエンコーディングです。このエンコーディングには、大文字と小文字の英語、アメリカ英語の句読点、基数が 10 の数値、少数のコントロール文字が含まれています。この 128 文字のセットは、他のほとんどのエンコーディングで共通です。ASCII はパーソナルコンピューターと LINUX コンピューターで使用されます。

ISO (国際標準化機構) 646 ファミリ

128 文字の組み合わせを提供する国際標準の 7 ビットエンコーディングです。ISO 646 エンコーディングファミリは、各国バリエーションとして 12 コードポイントを使用する点を除いて ASCII に似ています。12 の各国バリエーションは、特定の言語で必要とされる特殊文字を表します。

ISO 8859 ファミリと Windows ファミリ

すべての ASCII コードポイントをサポートし、12 コードポイントを追加して 256 文字の組み合わせを提供する ASCII の 8 ビット拡張です。公式には ISO-8859-1 という名前を持つ LATIN1 は、ISO 8859 エンコーディングファミリの中で最も頻繁に使用されるメンバーです。LATIN1 には、ASCII 文字に加えて、アクセント記号付き文字、西ヨーロッパの言語で必要とされる他の文字、および

いくつかの特殊文字が含まれています。HTTP および HTML プロトコルは、Unicode に基づいています。

その他のエンコーディング

ISO 8859 ファミリには、他の言語用に設計されたメンバーが他にもあります。次の表に、ISO によって承認された他のエンコーディングの説明を示します。

表 3.2 ISO によって承認されたその他のエンコーディング

ISO 標準	エンコーディング名	説明
ISO 8859-1	LATIN 1	米国および西ヨーロッパの言語
ISO 8859-2	LATIN 2	中央および東ヨーロッパの言語
ISO 8859-3	LATIN 3	南ヨーロッパの言語、マルタ語、エスペラント語
ISO 8859-4	BALTIC	北ヨーロッパ
ISO 8859-5	CYRILLIC	スラブ系言語
ISO 8859-6	ARABIC	アラビア語
ISO 8859-7	GREEK	現代ギリシャ語
ISO 8859-8	HEBREW	ヘブライ語およびイディッシュ語
ISO 8859-9	TURKISH	トルコ語
ISO 8859-10	LATIN 6	北欧(イヌイット語、サーミ語、アイスランド語)
ISO 8859-11	LATIN/THAI	タイ語
ISO 8859-13	LATIN 7	バルト語
ISO 8859-14	LATIN 8	ケルト語
ISO 8859-15	LATIN 9	西ヨーロッパ
ISO-8859-16	LATIN 10	中央および東ヨーロッパの言語

東アジア言語用に多数のエンコーディング標準が開発されています。その一部を次の表に示します。

表 3.3 ISO によって承認された一部の東アジア言語のエンコーディング

標準	エンコーディング名	説明
GB 2312-80	簡体字中国語	中華人民共和国
CNS 11643	繁体字中国語	台湾
BIG-5	繁体字中国語	台湾
KS C 5601	韓国産業標準	韓国
JIS	日本工業規格	日本
SHIFT-JIS	日本工業規格マルチバイトエンコーディング	日本

さまざまな言語とロケールをサポートする他のエンコーディングがあります。

標準エンコーディングとそれらのエイリアスについては、[“エンコーディングとその別名とエンコーディング文字セットの互換性”](#)を参照してください。

各国語サポートエンコーディング関連の標準組織

SAS によってサポートされるエンコーディングは、次の標準組織によって定義されます。

国際標準化機構(ISO)

標準化とその関連活動の発展を促進します。ISO では、エンコーディングの標準も策定します。

米国規格協会(ANSI)

米国の標準規格の調整を計り、それらの標準に承認を与えます。国際規格を作成するために、ISO (国際標準化機構)と協力して活動します。

Unicode コンソーシアム

すべての文字に一意的番号を付与する Unicode 標準を作成および推進します。

照合順序

照合順序の概要

照合順序は、文字を並べ替える順序です。たとえば、SORT プロシジャを実行すると、照合順序により他の文字に対する特定の文字の並べ替え順序(高い、低い、等しい)が決まります。

デフォルトの照合順序はバイナリ照合で、セッションエンコーディングのコードページにおける各文字の場所に従って文字を並べ替えます。(セッションエンコーディングは SAS セッションのデフォルトエンコーディングです。デフォルトエンコーディングは、さまざまな SAS 言語要素を使用して指定できます)。並べ替え順序は、コードページ内のコードポイントの配置に直接対応します。

バイナリ照合はコンピューターにとって最も効率が高く、最速の種類の照合です。ただし、バイナリ照合のレポート内の文字の検索は、この方法がよくわからない場合には多少複雑です。たとえば、バイナリ照合のレポートは、大文字で始まる単語を小文字で始まる単語と区別して表示します。アクセント記号付きの文字で始まる単語はアクセント記号のない文字で始まる単語の後に表示されます。したがって、ASCII ベースのエンコーディングでは、大文字の **Z a. z** は大文字の **A** の前にあります。

バイナリ照合より優先される代替照合順序を要求できます。代替照合順序を要求するには、次の順序のいずれかを指定します。

- 変換テーブル名
- エンコーディング値
- 言語照合

この例は、異なる照合順序を使用して単語のリストを並べ替えた結果を示します。

表 3.4 異なる照合順序の結果

バイナリ	変換テーブル	エンコーディング値	言語
Aaron	aardvark	Aaron	aardvark
Aztec	azimuth	Aztec	Aaron
Zeus	Aaron	Zeus	azimuth
aardvark	Aztec	aardvark	Aztec
azimuth	cote	azimuth	cote

バイナリ	変換テーブル	エンコーディング値	言語
cote	coté	cote	côte
coté	côte	coté	coté
côte	côté	côte	côté
côté	zebra	côté	zebra
zebra	zèbre	zebra	zèbre
zèbre	Zeus	zèbre	Zeus

最初の列は、ASCII ベースのエンコーディングで表示される文字のバイナリ照合の結果です。大文字と小文字で始まる単語は異なるグループになるため、一貫したアルファベット順ではありません。たとえば、単語 Zeus は、ASCII ベースのエンコーディング内で文字に割り当てられたコードポイントでの判断により、aardvark より前に表示されます。

2 番目の列は、小文字と大文字の並べ替えを交互にする変換テーブルを指定した結果です。変換テーブルを使用した場合、単語 aardvark は Zeus より前に表示されます。ただし、変換テーブルにより小文字 **a** は大文字 **A** よりも低い重みの値が割り当てられるため、単語 azimuth は Aaron より前に表示されます。さらに、アクセント記号は左から右の順序で比較して並べ替えられます。たとえば、coté は côte より前に表示されます。

3 番目の列は、ASCII ベースの 2 バイト latin1 エンコーディングを指定した結果です。

最後の列は、セッションロケール fr_CA (French_Canada)の言語照合の結果で、単語をアルファベット順にする照合アルゴリズムを使用します。このアルゴリズムでは、小文字で始まる単語が大文字で始まる単語より前に表示されます。さらに、言語照合によりフランス語のロケール仕様に基づいて、アクセントが右から左の順序で比較して並べ替えられます。

SAS では、言語照合の実装に International Components for Unicode (ICU)を採用しています。ICU とその Unicode 照合アルゴリズム(UCA)の実装は標準となっています。照合順序は、指定したロケールの ICU により提供されたデフォルトが使用されます。

要求の代替照合順序

代替照合順序を要求するには、次の SAS 言語要素を使用します。

- PROC SORT ステートメントの SORTSEQ=オプション。[“照合順序オプション”](#)を参照してください。

どちらの方法も照合順序のすべてをサポートしていないことに注意してください。たとえば、PROC SORT ステートメントの SORTSEQ=オプションが言語照合をサポ

ートします。ただし、PROC SORT ステートメントの SORTSEQ=オプションと SORTSEQ=システムオプションの両方が、変換テーブル照合順序をサポートします。

PROC SQL は、言語照合もサポートしています。[“データの並べ替え” \(SAS SQL プロシジャユーザーガイド\)](#)を参照してください。

SAS/SHARE の BASE (V9) Engine と REMOTE Engine は、すべての代替照合順序をサポートします。V9TAPE 順序エンジンはデータの並べ替えに変換テーブルとエンコーディング値の使用をサポートしますが、V9TAPE Engine は言語照合をサポートしません。

変換テーブルの指定

変換テーブルは、1 バイトエンコーディングから別の 1 バイトエンコーディングにデータをトランスコードする SAS カタログエントリです。また、変換テーブルにより文字が並べ替えられます。ASCII、または DANISH などの標準照合順序のような、SAS が提供する変換テーブルを使用できます。または、ユーザー定義の変換テーブルを使用できます。

代替照合順序に変換テーブルを指定する場合、各文字のコードポイントを 0 から 255 の範囲の整数の重みにマッピングすることで、文字が並べ替えられます。次に、バイナリ照合が実行されます。

照合目的で、小文字と大文字を交互に並べ替える変換テーブルを作成できます。たとえば、ASCII ベースのエンコーディングで **Z** が **a** より前に来る状態を修正する変換テーブルを作成できます。(ただし、変換テーブルの重みの割り当てにかかわらず、大文字、小文字を含めた完全なアルファベット順の並べ替えは困難です)。また、希望する順序で特定の言語の英字を並べ替える変換テーブルも作成できます。

TRANTAB プロシジャは、変換テーブルを作成、編集、表示します。たとえば、文字の重みの値を参照するために変換テーブルを表示できます。SAS で提供される変換テーブルは、SASHELP.HOST カタログに格納されます。作成またはカスタマイズした変換テーブルは、SASUSER.PROFILE カタログに格納されます。変換テーブルには TRANTAB のエントリタイプがあります。変換テーブルについては、[20 章](#), [“TRANTAB プロシジャ”](#)を参照してください。

PROC SORT ステートメントの SORTSEQ=オプションまたは SORTSEQ=システムオプションを使用して、変換テーブルを指定できます。たとえば、動作環境のデフォルトが ASCII ベースの Wlatin1 エンコーディングによる並べ替えで、大文字と小文字を交互にする変換テーブルで並べ替える場合、SAS 変換テーブル FRSOLAT1 を指定するために次のステートメントを発行します。

```
proc sort data=myfiles.test sortseq=FRSOLAT1;
  by name;
run;
```

変換テーブルで並べ替えられた SAS データセットには、指定した変換テーブル名を CONTENTS プロシジャ出力の照合順序として表示するソートインジケーターが含まれます。

エンコーディング値の指定

エンコーディングは、コンピューターが使用するコードポイントと呼ばれる 16 進数値にマッピングされた文字(通常の文字、表語文字、数字、句読点、記号、コントロール文字)のセットです。代替照合順序にエンコーディング値を指定する場合、文字は SAS セッションエンコーディングから指定したエンコーディングにトランスコードされます。次に、バイナリ照合が実行されます。マルチバイトエンコーディングを含め、ENCODING=オプションでサポートされるすべてのエンコーディング値を指定できます。変換テーブルを指定してデータをトランスコードできますが、変換テーブルは 1 バイトエンコーディングのみに制限されることに注意してください。

PROC SORT ステートメントの SORTSEQ=オプションでエンコーディング値を指定できますが、SORTSEQ=システムオプションではエンコーディング値を指定できません。たとえば、SAS データセットを並べ替えて日本語の Windows 環境に移動させるとします。セッションエンコーディングが ASCII ベースでバイナリ照合が実行されている場合、ASCII ベースの 2 バイトエンコーディング SHIFT-JIS を指定するために次のステートメントを発行できます。

```
proc sort data=myfiles.test sortseq='shift-jis';  
  by name;  
run;
```

SAS では、同名の変換テーブルのエンコーディング値があるかどうか確認されることに注意してください。変換テーブル名が存在する場合、その変換テーブルが使用されます。

エンコーディング値で並べ替えられた SAS データセットには、指定したエンコーディング値を CONTENTS プロシジャ出力の照合順序として表示するソートインジケータが含まれます。

言語照合の指定

言語照合は、言語のルールに従って文字を並べ替え、直感的で文化的に受け入れられる結果を作成します。結果は、辞書、電話帳、本の索引などの紙媒体で使用される照合と似ています。言語照合は、レポートやその他のデータ表示の作成と、システム間の互換性の実現に役立ちます。

SAS では International Components for Unicode (ICU)を採用しています。ICU は、Unicode 照合アルゴリズム(UCA)と互換性のある言語照合のルーチンを提供するオープンソースライブラリです。UCA は、Unicode 文字列を比較して並べ替えができる標準です。

SORTSEQ=システムオプションは言語照合をサポートしないため、言語照合を要求するには、PROC SORT ステートメントの SORTSEQ=オプションを使用する必要があります。たとえば、次のステートメントでは、French_Canada のロケールに従って言語的に照合する SORT プロシジャを実行します。

```
options locale=fr_CA;
```

```
proc sort data=myfiles.test sortseq=linguistic;  
  by name;  
run;
```

注: 2021.2.4 以降、ICU ライブラリは 70.1 にアップグレードされました。アップグレードにより、zh_CN や zh_TW などの中国語ロケールの言語照合に非互換性が見られる場合があります。

中国語ロケールの言語照合機能によってソートされた既存のデータセットは、再ソートする必要があります。

言語照合を要求すると、SAS セッションロケール用に ICU で提供されたデフォルトの言語照合アルゴリズムが使用されます。このアルゴリズムには、各地域の言語、データのフォーマティングなどの規則、文化が反映されています。アルゴリズムを変更するには、LINGUISTIC キーワードの後にオプションをカッコで囲んで指定します。たとえば、異なるロケールを指定できます。また、小文字を大文字より前にしたり、その逆の順序にする CASE_FIRST=オプションも指定できます。その他いろいろな指定が可能です。通常は、ICU によりさまざまな言語とロケールがデフォルトで関連付けられるため、オプションを指定する必要はありません。言語オプションの詳細については、“[照合順序オプション](#)”の SORTSEQ=オプション、または *Base SAS プロシジャガイド*内の PROC SORT ステートメントの SORTSEQ=オプションを参照してください。

言語的に並べ替えられた SAS データセットには、CONTENTS プロシジャ出力の照合順序 LINGUISTIC を表示するソートインジケータが含まれます。このソートインジケータとともに、データセットにより言語照合順序の完全な説明がファイルのディスクリプター情報に記録されます。ディスクリプター情報は CONTENTS プロシジャ出力にも表示されます。

SAS データセットのエンコーディングの確認

PROC CONTENTS を使用して SAS データセットの内容を表示する

データセットのエンコーディングを表示するには、PROC CONTENT を使用します。

```
proc contents data=sashelp.class; run;
```

PROC CONTENTS コードからの出力を次に示します。エンコーディングは US-ASCII です。

The CONTENTS Procedure			
Data Set Name	SASHELP.CLASS	Observations	19
Member Type	DATA	Variables	5
Engine	V9	Indexes	0
Created	01/17/2016 20:12:45	Observation Length	40
Last Modified	01/17/2016 20:12:45	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label	Student Data		
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	us-ascii ASCII (ANSI)		

デフォルトの SAS セッションエンコーディング

ENCODING=システムオプションは、SAS セッションのエンコーディングを指定するために使用されます。セッションエンコーディングによって、SAS 構文と SAS データセットの処理、および外部ファイルの読み込みと書き込みを行うための環境が設定されます。LOCALE=オプションと ENCODING=オプションのいずれも設定されていない場合は、デフォルト値が設定されます。

ENCODING システムオプションは、すべての SASFoundation sasv9.cfg 構成ファイルで明示的に設定されています。

BI サーバー構成プロセスはデフォルトの SASFoundation sasv9.cfg ファイル(!SASROOT/sasv9.cfg)を使用します。デフォルトの sasv9.cfg ファイルは選択した LOCALE および ENCODING オプションを含む sasv9.cfg を指します。このアクションは通常、展開時に発生します。さまざまな SAS 構成を使用するサイトで、複数のサーバーコンテキストを展開できます。

表 3.5 デフォルトの SAS セッションエンコーディング値

動作環境	デフォルトの ENCODING=値	説明
LINUX	Latin1	西ヨーロッパ諸語(ISO)

SAS セッションでサポートされるエンコーディング値の一覧については、[24 章, “SAS セッションのエンコーディング値” \(1131 ページ\)](#)を参照してください。

SAS セッションエンコーディングの設定

セッションエンコーディングを設定するには、ENCODING=システムオプション、DBCS オプションまたは LOCALE=システムオプションを使用します。

注: ENCODING=システムオプションの値は、動作環境によって異なります。

エンコーディングは、次の優先順位に従って設定されます。

1 ENCODING=システムオプション

SAS セッションエンコーディングは、DBCS オプションまたは LOCALE=オプションが指定されたかどうかに関係なく、ENCODING=オプションによって決まります。ENCODING=オプションが指定された場合、ユーザーが DBCS オプションを指定したかどうかに関係なく、有効な DBCS オプション一式が設定されます。また、ENCODING=オプションが指定された場合、LOCALE=オプションは、ユーザーによって値が指定されていないければ、適切な値に設定されます。

注: ENCODING=オプションが指定された場合、TRANTAB=オプションは暗黙的に設定されます。TRANTABS は外部ファイルのみに適用されます。

2 DBCS オプション

ほとんどの南北アメリカとヨーロッパ、中東、アフリカおよびオーストラリアのユーザーは SAS SBCS 環境を使用しており、DBCS 環境は使用しません。

ENCODING=オプションが指定されていない場合、SAS セッションエンコーディングは、LOCALE=オプションが指定されたかどうかに関係なく、DBCS オプションによって決まります。LOCALE=オプションは、ユーザーによって値が指定されていないければ、適切な値に設定されます。

日本語、韓国語、簡体字中国語、繁体字中国語などの DBCS 言語の場合、エンコーディングは DBCSLANG および DBCSTYPE オプションの値によって決まります。

DBCS オプションは、DBCS 拡張ディレクトリがパスオプションリストに含まれる場合にのみ有効です。DBCS 拡張 DLL(ダイナミックリンクライブラリ)のパスは、DBCS SAS セッションの起動時に、DBCS 言語用パスオプションのパス名リストの先頭に置く必要があります。デフォルトでは、DBCS 拡張 DLL はディレクトリ **!SASROOT/dbcs/sasexe** にあります。

また、SAS セッションエンコーディングが DBCS 言語でない場合(ポーランド語、ドイツ語、フランス語など)でも、地域化されたリソースを使用するには、resourcesloc.msg および sashelp オプションの指定が必要になることがあります。地域化されたリソースは、**!SASROOT/nls/<language identifier>/<sasmsg, sashelp, sasmacro, resource>**の下にあります。言語識別子の値は、cs、de、en、es、fr、hu、it、ja、ko、pl、ru、sv、zh、zt です。

!SASROOT/nls/<language identifier>などの地域化されたディレクトリにある **sasv9.cfg** ファイルを指定できる場合、**path**、**resourcesloc**、**sasmsg**、および **sashelp** オプションの使用を検討する必要はありません。

DBCS (SAS が DBCS エンコーディングを処理することを指定)が指定された場合、**DBCSLANG**=および **DBCSTYPE**=オプションは暗黙的に設定されます。**DBCSTYPE**=および **DBCSLANG**=のデフォルト値は、ホストの DBCS 環境(たとえば日本語や韓国語や中国語)の対応する値と一致する必要があります。

3 LOCALE=システムオプション

ENCODING=と **DBCS** オプションのいずれも指定されていない場合、SAS セッションエンコーディングは、**LOCALE**=オプションとプラットフォームによって決まります。

次の例では、エンコーディングは、Spanish_Spain ロケールのデフォルトに明示的に設定されます。

```
sas9 -locale Spanish_Spain
```

Spanish_Spain ロケールのデフォルトのエンコーディングは、WLATIN1 エンコーディングです。

次の例では、SAS 起動時に WLATIN2 エンコーディングが明示的に設定されます。

```
sas9 -encoding wlatin2
```

注: DBCS 拡張が使用できない場合、DBCS エンコーディング、DBCS オプションまたは CJK (中国語、日本語、韓国語)ロケールを SAS に設定すると、SAS の起動が失敗します。

注: SAS セッションのエンコーディングを変更しても、SAS キーワードや SAS ログ出力は英語のままで、影響はありません。

表 3.6 (24 ページ)では、CJK ロケールの次の値は、ロケールとプラットフォームに基づきます。

表 3.6 LOCALE=オプションに基づいたデフォルトのエンコーディング値

ロケール	LINUX
zh_TW	X64 の Solaris、SPARC の Solaris、EUC-TW (yeuc) その他: MS-950 (ywin)
zh_HK	
zh_MO	
zh_CN	EUC-CN (zeuc)
zh_SG	
ja_JP	h64、h6i、Power の AIX、SHIFT-JIS (sjis) その他: EUC-JP (jeuc)

ロケール	LINUX
ko_KR	EUC-KR (keuc)

SAS セッションのエンコーディングの動作

SAS リリース別のエンコーディングサポート

SAS ファイルの場合、エンコーディングサポートには 3 つのカテゴリがあります。エンコーディングサポートは、ファイルを作成した SAS のバージョンに基づいています。

- Viya で作成されたデータセットには、ファイルのディスクリプター部分を使用してエンコーディング属性が自動的に指定されます。Viya では、デフォルトで、DBCS が DBCSTYPE 値を認識してエンコーディング値に変換し、その値をフィールドのディスクリプター部分に指定します。
- SAS 7 および SAS 8 で作成されるデータセットには、ファイルで指定されるエンコーディング値がありません。SAS 7 および SAS 8 のデータセットは、動作環境の SAS セッションエンコーディングで作成されたものと見なされます。ただし、ファイルのディスクリプター部分はエンコーディング値をサポートします。SAS 9 セッションで SAS 7 または SAS 8 のファイルを置き換えたり更新したりすると、SAS はデフォルトで、現在のセッションエンコーディングをファイルのディスクリプター部分に指定します。SAS 8 の場合、DBCS には、エンコーディングフィールドではなく DBCSTYPE フィールドがあります。
- SAS 6 で作成されたデータセットには、ファイルに関連付けられたエンコーディング値がありません。エンコーディング値はファイルに指定できません。

出力処理

データセットを作成すると、エンコーディングは次のように決定されます。

- 新しい出力ファイルを作成する場合、データは、現在のセッションエンコーディングを使用してそのファイルに書き込まれます。
- OUTREP=オプションを使用して新しい出力ファイルを作成すると、OUTREP=値によって指定した、オペレーティングシステムのデフォルトセッションエンコーディングを使用してデータはファイルに書き込まれます。OUTREP=オプション

は、現在のセッションとは異なるデータ表現を指定します。詳細については、[“OUTREP= データセットオプション”](#)を参照してください。

- 新しい出力ファイルで既存のファイルを置き換える場合、新しいファイルは既存のファイルのエンコーディングを継承します。別の動作環境にある既存のファイルを置き換える出力処理の場合、または既存のファイルにエンコーディングが指定されていない場合は、現在のセッションエンコーディングが使用されます。

入力処理

入力(読み込み)処理におけるエンコーディング動作を次に示します。

- ほとんどのユーザーは、入力ファイルにエンコーディングを指定しない、デフォルトの動作を選びます。
- セッションエンコーディングとファイルで指定されたエンコーディングの間に互換性がない場合、データはセッションエンコーディングにトランスコードされます。
- ファイルに指定されたエンコーディングが含まれていない場合、SAS は、ファイルのデータ表記がセッションエンコーディングと異なる場合にのみデータをトランスコードします。

外部ファイルの読み込みと書き込み

SAS は、現在のセッションエンコーディングを使用して外部ファイルの読み込みと書き込みを行います。Viya では、外部ファイルのエンコーディングがセッションエンコーディングと同じであると想定します。たとえば、外部ファイルを読み込んで新しいデータセットを作成する場合、Viya は、外部ファイルのエンコーディングと現在のセッションのエンコーディングが同じであると想定します。エンコーディングが同じでない場合、外部データが新しいデータセットに誤って書き込まれる可能性があります。適切な ENCODING オプションを指定する必要があります。次の例では、Shift-JIS エンコーディングを指定します。

```
filename in 'external-file'
encoding='Shift-JIS';
data mylib.contacts;
infile in;
length name $ 30 first $ 30 street $ 60 zip $ 10 city $ 30;
input name first street zip city;
run;
```

入出力処理を実行する SAS ステートメントの構文の詳細については、[“SAS データのトランスコーディングを指定する SAS オプション”](#)を参照してください。

UTF-8 へのデータの移行

UTF-8 は、Unicode でコード化された文字セットの一部です。UTF-8 は、推奨され、最も使用されているエンコーディングであり、Linux などのオペレーティングシステムでの Unicode の使用に推奨されるエンコーディングです。

多言語データをサポートし、Viya をサポートするために、他のエンコーディングを使用するデータを UTF-8 エンコーディングに移行できます。データを移行する方法はいくつかあります: 相互環境データアクセス(CEDA)、文字変数パディング(CVP)エンジン、および%COPY_TO_NEW_ENCODING マクロ。これらの方法の使用方法については、[SAS Viya プラットフォームでの UTF-8 へのデータの移行](#)を参照してください。

WLATIN1 から UTF-8 へのデータの移行

西ヨーロッパの文字データの一般的なエンコーディングは、ISO 8859-1(LATIN1)および Windows cp1252(WLATIN1)です。文字データが LATIN1 または WLATIN1 としてエンコードされている場合には、UTF-8 に移行する際に考慮すべき要因がいくつかあります。このセクションでは、LATIN1 文字のスーパーセットである WLATIN1 文字に焦点を当てます。

すべての WLATIN1 文字は UTF-8 で使用できます。WLATIN1 のコードページの例を次に示します。

図 3.1 WLATIN1 のコードページ

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
2_	(space)	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3_	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4_	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5_	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7_	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
8_	€		,	f	„	...	†	‡	^	‰	Š	‹	Œ		Ž	
9_		‘	’	“	”	•	—	—	~	™	š	›	œ		ž	ÿ
A_	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	(shy)	®	¯
B_	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
C_	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D_	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E_	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F_	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

WLATIN1 の文字には 1 バイトが必要であり、UTF-8 文字には 1~4 バイトが必要です。図 3.2 では、強調表示された 2~7 行目の文字には、UTF-8 で 1 バイトが必要です。これらの文字は、ASCII 文字とも呼ばれます。

図 3.2 WLATIN1 コードページの 2～7 行目

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
2_	(space)	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3_	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4_	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5_	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7_	p	q	r	s	t	u	v	w	x	y	z	{		}	~	<u>DEL</u>
8_	€		,	f	„	...	†	‡	^	%o	Š	<	Œ		Ž	
9_		‘	’	“	”	•	—	—	~	™	š	>	œ		ž	ÿ
A_	<u>NBSP</u>	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	(shy)	®	¯
B_	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
C_	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D_	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E_	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F_	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

図 3.3 では、強調表示された 8～F 行目の文字には、WLATIN1 で 1 バイト、UTF-8 で 2～3 バイトが必要です。

図 3.3 WLATIN1 コードページの 8〜F 行目

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
2_	(space)	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3_	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4_	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5_	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7_	p	q	r	s	t	u	v	w	x	y	z	{		}	~	<u>DEL</u>
8_	€		,	f	„	...	†	‡	^	%o	Š	<	Œ		Ž	
9_		‘	’	“	”	•	—	—	~	™	š	>	œ		ž	ÿ
A_	<u>NBSP</u>	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	(shy)	®	¯
B_	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
C_	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D_	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E_	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F_	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

WLATIN1 では 1 バイト、UTF-8 では 2〜4 バイトの文字を移行した結果として発生するエラーまたは誤ったデータについては、次の推奨事項に従って問題を解決してください。

切り捨て

すべての WLATIN1 文字を UTF-8 にトランスコードできます。トランスコーディングエラーまたは警告は、文字変数がそれらの文字の UTF-8 表現を保持するのに十分な長さでないことを意味します。この問題を解決するには、[CVP](#) を使用して文字列のサイズを拡張してください。

予期しない文字

WLATIN1 データの中のすべての文字が UTF-8 で 1 バイトであると予想している場合に、データを UTF-8 に移行すると、トランスコーディングエラーが表示されて驚くことがあります。WLATIN1 エンコーディングには、いくつかの句読文字の 2 つのバージョンが含まれています。

- 2～7 行目の ASCII バージョン
- 8～F 行目の 8 ビット文字

Microsoft Word などの一部のテキストエディターでは、キーボードから入力された ASCII 文字を、活字デバイスで生成された文字のバージョンに変換する AutoFormat 機能をサポートしています。

たとえば、引用符文字(")(ストレート引用符とも呼ばれます)は、WLATIN1 と UTF-8 の両方で 22 の 16 進表現を持っています。ただし、AutoFormat は、文字を左右の二重引用符(中引用符またはスマート引用符とも呼ばれます)に変換場合があります。WLATIN1 の左右の二重引用符の 16 進値は、93 と 94 です。ただし、UTF-8 では、これらの文字は E2809C および E2809D であり、各文字には 3 バイトが必要です。

[KPROPDATA 関数](#)を PUNC オプションを付けて使用することで、スマート引用符やその他の活版印刷の句読文字を UTF-8 で 1 バイト文字に変換することができます。

WLATIN1 では 1 バイト、UTF-8 では 2～4 バイトの句読文字は次のとおりです。

表 3.7 スマート引用符と句読文字

カテゴリ	句読点				変換		
	句読文字	説明	WLATIN1 コードポ イント (Hex)	UTF-8 コード ポイン ト (Hex)	ASCII 文字	説明	ASCII コ ードポ イント (Hex)
引用符	,	下つき引用 符	82	E2 80 9A	,	カンマ	2C
	„	下つき二重 引用符	84	E2 80 9E	„	引用符	22
	‘	引用符(始)	91	E2 80 98	'	アポストロ フィ	27
	’	引用符(終)	92	E2 80 99	'	アポストロ フィ	27
	“	二重引用符 (始)	93	E2 80 9C	"	引用符	22
	”	二重引用符 (終)	94	E2 80 9D	"	引用符	22
山かっこ	<	山カッコ (始)	8B	E2 80 B9	<	小なり記号	3C
	>	山カッコ(終)	9B	E2 80 BA	>	大なり記号	3E

カテゴリ	句読点				変換		
	句読文字	説明	WLATIN1 コードポ イント (Hex)	UTF-8 コード ポイン ト (Hex)	ASCII 文字	説明	ASCII コ ードポ イント (Hex)
省略記号	...	三点リーダ ー	85	E2 80 A6	...	終止符	2E 2E 2E ¹
行頭文字	•	黒丸	95	E2 80 A2	*	アスタリス ク	2A
ハイフン	-	en ハイフン	96	E2 80 93	-	ハイフンマ イナス	2D
	—	em ハイフ ン	97	E2 80 94	-	ハイフンマ イナス	2D

1 3つのASCIIピリオド文字。

各国語サポート関連のトランスコーディング

トランスコーディングの概要	33
トランスコーディングが必要となる場合	34
トランスコーディングと変換テーブル	34
SAS データのトランスコーディングを指定する SAS オプション	36
トランスコーディングの留意点	37
互換性があるエンコーディングと互換性がないエンコーディング	38
互換性があるエンコーディングと互換性がないエンコーディングの概要	38
セッションエンコーディングとデータセットエンコーディング、注、警告、エラー	42
トランスコーディング中の切り捨てとデータ損失の問題のトラブルシューティング	43
CVP エンジンの使用による文字データ切り捨ての回避	44
%COPY_TO_NEW_ENCODING マクロを使用した文字の切り捨ての回避	49
KPROPDATA 関数を使用した CEDA トランスコーディング中の文字データ損失の回避	50

トランスコーディングの概要

トランスコーディングとは、1つのエンコーディングから別のエンコーディングにデータを変換する処理のことです。トランスコーディングは、SAS セッションエンコーディングとデータのエンコーディングが異なる場合に必要になります。多くの場合、異なるロケールとエンコーディングを使用する動作環境間でデータを移動するときにトランスコーディングが必要になります。

並べ替え順序は、コードページ内のコードポイントの配置に直接対応します。

たとえば、LATIN1 エンコーディングを使用する Linux 動作環境で作成されたファイルを、ドイツ語 EBCDIC エンコーディングを使用する IBM メインフレームに移動する場合などが考えられます。IBM メインフレームでファイルが処理されるとき、デ

ータは、LATIN1 エンコーディングからドイツ語 EBCDIC エンコーディングに再マッピングされます。データに大文字の Ä が含まれる場合、16 進数は C4 から 4A に変換されます。

トランスコーディングでは、言語間の翻訳は行いません。トランスコーディングで行われるのは文字の再マッピングです。

異なるエンコーディングを使用する動作環境間でデータを動的にトランスコードするには、明確なエンコーディング値を指定する必要があります。詳細については、[23 章, "SAS 言語要素中のエンコーディング値"](#)を参照してください。

トランスコーディングが必要となる場合

データがトランスコードされる頻度が高いのは、次のような状況です。

- 異なるロケールまたは異なる動作環境で実行している、2 つの異なる SAS セッション間でデータを共有する場合
- 外部ファイルの読み取りで、ファイルのエンコーディングがセッションのエンコーディングと互換性がない場合

トランスコーディングと変換テーブル

変換テーブルは、LOCALE=システムオプションによって暗黙的に設定されます。これらは外部ファイルのトランスコーディングにのみ使用されます。SAS ファイルに対しては、セッションエンコーディング間の直接トランスコーディングが発生します。

LOCALE=または ENCODING=を間接的に指定すると、TRANTAB=オプションに適切な変換テーブル値が設定されます。変換テーブルは、SBCS エンコーディングを別の SBCS エンコーディングにトランスコードしたり元に戻したりするために使用します。たとえば、Windows LATIN2 を ISO LATIN2 にマッピングする特定の変換テーブルがあります。変換テーブルは外部ファイルのみに適用されます。

次の図に、変換テーブルを示します。Windows LATIN2(wlt2)から ISO LATIN 2(lat2)へのマッピングの変換テーブル領域名は"table 1"で、ISO LATIN2 から Windows LATIN2 へのマッピングの領域名は"table 2"です。

図 4.1 SAS Windows LATIN 2 から ISO LATIN 2 への変換テーブル

WLT2LAT2 table 1:																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	'000	'102	'030	'405	'060	'708	'090	'A0B	'0C0	'D0E	'0F'x					
10	'101	'112	'131	'141	'151	'161	'171	'181	'191	'1A1	'1B1	'1C1	'1D1	'1E1	'1F'x	
20	'202	'122	'232	'242	'252	'262	'272	'282	'292	'2A2	'2B2	'2C2	'2D2	'2E2	'2F'x	
30	'303	'132	'333	'343	'353	'363	'373	'383	'393	'3A3	'3B3	'3C3	'3D3	'3E3	'3F'x	
40	'404	'142	'424	'434	'444	'454	'464	'474	'484	'494	'4A4	'4B4	'4C4	'4D4	'4E4	'4F'x
50	'505	'152	'525	'535	'545	'555	'565	'575	'585	'595	'5A5	'5B5	'5C5	'5D5	'5E5	'5F'x
60	'606	'162	'626	'636	'646	'656	'666	'676	'686	'696	'6A6	'6B6	'6C6	'6D6	'6E6	'6F'x
70	'707	'172	'727	'737	'747	'757	'767	'777	'787	'797	'7A7	'7B7	'7C7	'7D7	'7E7	'7F'x
80	'808	'182	'828	'838	'848	'858	'868	'878	'888	'898	'8A8	'8B8	'8C8	'8D8	'8E8	'8F'x
90	'908	'182	'828	'838	'848	'858	'868	'878	'888	'898	'8A8	'8B8	'8C8	'8D8	'8E8	'8F'x
A0	'A0B	'7A2	'A3A	'4A1	'95A	'7A8	'96A	'A97	'99A	'D9A	'9AF'x					
B0	'B09	'B2B	'2B3	'B49	'C9D	'9EB	'8B1	'BA9	'FA5	'BDB	'5BF'x					
C0	'C0C	'1C2	'C3C	'4C5	'C6C	'7C8	'C9C	'ACB	'CCC	'DCE	'CF'x					
D0	'D0D	'1D2	'D3D	'4D5	'D6D	'7D8	'D9D	'ADB	'DCD	'DDDE	'DF'x					
E0	'E0E	'1E2	'E3E	'4E5	'E6E	'7E8	'E9E	'AEB	'ECE	'DEEE	'F'x					
F0	'F0F	'1F2	'F3F	'4F5	'F6F	'7F8	'F9F	'AFB	'FCF	'DFFF	'F'x					

WLT2LAT2 table 2:																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	'000	'102	'030	'405	'060	'708	'090	'A0B	'0C0	'D0E	'0F'x					
10	'101	'112	'131	'141	'151	'161	'171	'181	'191	'1A1	'1B1	'1C1	'1D1	'1E1	'1F'x	
20	'202	'122	'232	'242	'252	'262	'272	'282	'292	'2A2	'2B2	'2C2	'2D2	'2E2	'2F'x	
30	'303	'132	'333	'343	'353	'363	'373	'383	'393	'3A3	'3B3	'3C3	'3D3	'3E3	'3F'x	
40	'404	'142	'424	'434	'444	'454	'464	'474	'484	'494	'4A4	'4B4	'4C4	'4D4	'4E4	'4F'x
50	'505	'152	'525	'535	'545	'555	'565	'575	'585	'595	'5A5	'5B5	'5C5	'5D5	'5E5	'5F'x
60	'606	'162	'626	'636	'646	'656	'666	'676	'686	'696	'6A6	'6B6	'6C6	'6D6	'6E6	'6F'x
70	'707	'172	'727	'737	'747	'757	'767	'777	'787	'797	'7A7	'7B7	'7C7	'7D7	'7E7	'7F'x
80	'808	'182	'828	'838	'848	'858	'868	'878	'888	'898	'8A8	'8B8	'8C8	'8D8	'8E8	'8F'x
90	'909	'697	'999	'BA6	'A9B	'98A	'CAE	'B1B	'5B6	'B7B	'BB'x					
A0	'A0A	'5A2	'A3A	'4BC	'8CA	'7A8	'8AA	'A8D	'8FA	'D8E	'AF'x					
B0	'B0B	'9B2	'B3B	'4BE	'9CA	'1B8	'9ABA	'9D9	'FBD	'9EB	'F'x					
C0	'C0C	'1C2	'C3C	'4C5	'C6C	'7C8	'C9C	'ACB	'CCC	'DCE	'CF'x					
D0	'D0D	'1D2	'D3D	'4D5	'D6D	'7D8	'D9D	'ADB	'DCD	'DDDE	'DF'x					
E0	'E0E	'1E2	'E3E	'4E5	'E6E	'7E8	'E9E	'AEB	'ECE	'DEEE	'F'x					
F0	'F0F	'1F2	'F3F	'4F5	'F6F	'7F8	'F9F	'AFB	'FCF	'DFFF	'F'x					

LOCALE=または ENCODING=システムオプションとその他のエンコーディングオプション(ステートメント、コマンドまたはプロシジャ)を使用すると、変換テーブルを直接作成または管理する必要がなくなります。

注意

利用方法がよくわからない場合は変換テーブルを変更しないでください。変換テーブルは SAS スーパーバイザーにより各国語サポートを提供するために内部的に利用されます。変換テーブルの利用方法がよくわからない場合や、技術的な支援がない場合は設定を変更しないでください。

TRANTAB=オプションでは、SAS セッションで使用される変換テーブルを指定します。詳細については、“[TRANTAB= システムオプション](#)”を参照してください。TRANTAB プロシジャは、カスタマイズした変換テーブルの作成、編集、表示に使用されます。詳細については、[20 章](#), “[TRANTAB プロシジャ](#)”を参照してください。

SAS データのトランスコーディングを指定する SAS オプション

さまざまな言語要素に次の SAS オプションを使用すると、トランスコーディングを実行したりデフォルトのエンコーディング動作を無効にして別の動作を指定したりできます。これらの要素によって、SAS ファイルまたは SAS アプリケーションに異なるエンコーディングを指定したり、トランスコーディングを抑止したりできます。

表 4.1 SAS データのトランスコーディングを指定する SAS オプション

オプション	使用箇所
CHARSET=	ODS MARKUP ステートメント
CORRECTENCODING=	DATASETS プロシジャの MODIFY ステートメント
ENCODING=	%INCLUDE、FILE、FILENAME、INFILE、ODS の各ステートメント、FILE および INCLUDE コマンド
ENCODING=	DATA ステップ
INENCODING=	LIBNAME ステートメント
ODSCHARSET=	XML の LIBNAME ステートメント
ODSTRANTAB=	XML の LIBNAME ステートメント
OUTENCODING=	LIBNAME ステートメント
XMLENCODING=	XML の LIBNAME ステートメント

これらのオプションに使用するサポート対象エンコーディング値の一覧については、「データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値」を参照してください。

トランスコーディングの留意点

通常、トランスコーディングは問題なく実行されますが、データに影響を与え、不満足な結果をもたらす場合があります。いくつか例を示します。

- エンコーディング同士が競合する可能性があります。つまり、2種類のエンコーディングにおいて、同じ文字に対して異なるコードポイントが使用されていたり、2つの異なる文字に対して同じコードポイントが使用されていたりする場合があります。
- 1つのエンコーディングの文字が、別のエンコーディングに存在しない可能性があります。たとえば、特定のエンコーディングにはドル記号(\$)の文字がない場合があります。ドル記号をサポートしていないエンコーディングにデータをトランスコーディングすると、その文字が出力されなかったり、表示されなかったりします。
- 1つのエンコーディングでの1文字に対応するバイト数が、別のエンコーディングでの同じ文字に対応するバイト数と異なる可能性があります。2バイト文字セット(DBCS)から1バイト文字セット(SBCS)にトランスコーディングする場合などにみられます。したがって、トランスコーディングにより文字値が切り捨てられる可能性があります。
- トランスコーディング中、データを元のエンコーディングにトランスコーディングして戻すことができないなどのエラーが発生した場合は、データが失われる可能性があります。つまり、更新処理のためにデータセットを開いても、オブザベーションが更新されない場合があります。ただし、入力(読み込み)処理のためにデータセットを開き、出力データセットを何も開かない場合、SASは警告を発行して表示します。処理が進行すると、PRINT プロシジャまたは他の読み込み操作を使用して、トランスコーディングされないデータを表示できます。
- CEDA には、いくつかの処理制限があります。たとえば、CEDA は更新処理をサポートしません。[“CEDA による SAS ファイル処理” \(SAS プログラマガイド: エッセンシャル\)](#)を参照してください。
- 不正なエンコーディングについては、SAS®9 セッションでコピーされたり、データの別のセッションエンコーディングと置き換えられたりする場合、SAS 7 または SAS 8 データセットにスタンプできます。不正なエンコーディングスタンプは、PROC DATASETS の MODIFY ステートメントの CORRECTENCODING=オプションによって訂正できます。文字変数にバイナリデータが含まれると、トランスコーディングによってデータが破損する可能性があります。
- データセットと変数のラベルは最大 256 バイト、変数名は最大 32 文字です。データセットに国際的な文字を含む長いラベルが含まれていると、トランスコードされたラベルが最大値を超え、トランスコードエラーが発生する可能性があります。元のデータセットで長いラベルを短くして、トランスコーディングの問題を解決できます。

互換性があるエンコーディングと互換性がないエンコーディング

互換性があるエンコーディングと互換性がないエンコーディングの概要

互換性のないファイルとは、ホストアーキテクチャとは異なるデータ表現か、または SAS セッションエンコーディングとは異なるエンコーディング値を持つファイルです。

表 4.2 はトランスコーディングのために CEDA 処理を必要としないエンコーディングのリストを示します。たとえば、最初の行はエンコーディング方法 SBCS ASCII 用です。セッションエンコーディングが LATIN1 でデータセットエンコーディングが LATIN9 の場合、CEDA 処理は発生しません。別の例として、DBCS ASCII エンコーディング方法があります。その行の最初の行のエンコーディングである SHIFT-JIS、MS-932、IBM-942、MACOS-1 の組み合わせでは、CEDA 処理は起動されません。セッションエンコーディングが IBM-942 でデータセットエンコーディングが MACOS-1 の場合、CEDA 処理は発生しません。

CEDA の詳細については、“[クロス環境データアクセス\(CEDA\)の定義](#)” ([SAS プログラマガイド: エssenシャル](#))を参照してください。

文字データを正しく処理するためには、セッションエンコーディングを使用してデータをエンコードする必要があります。異なるエンコーディングを持つデータセットが SAS で読み取られると、CEDA はデータセットエンコーディングと SAS セッションエンコーディングを比較します。これらのエンコーディングは互換性がない可能性があります。つまり、データセットから読み取られた文字を SAS で安全に処理するには、トランスコーディングが必要です。

ASCII は多くのエンコーディングの基礎で、多くのパーソナルコンピューターおよびワークステーションで使用されます。ただし、IBM メインフレームでは EBCDIC エンコーディングが使用されます。ASCII および EBCDIC マシン、およびデータの保管に使用されるエンコーディングは互換性がありません。そのため、トランスコーディングが必要です。たとえば、SAS ファイルが WLATIN1 で、セッションエンコーディングが OPEN_ED-1047 の場合、データをトランスコードする必要があります。SAS CEDA 処理は、z/OS や USS などの階層ファイルシステムにデータセットがある場合、メインフレーム環境でのみ使用されます。異なる言語の文字をサポートするエンコーディングも、互換性がありません。たとえば、WLATIN1 は西ヨーロッパ言語の文字をサポートする 1 バイト(SBCS)エンコーディングです。WCYRILLIC も SBCS ですが、ロシア語の文字が含まれ、西ヨーロッパの文字はあまり含まれていません。そのため、WLATIN1 と WCYRILLIC は互換性がありません。

CEDA トランスコーディングは無効にすることができます。たとえば、データセットのエンコーディングとセッションエンコーディングの互換性がなく、データセット

内のすべての文字が互換性があることがわかっている場合、トランスコーディングは必要ではないにもかかわらず発生します。CEDA の代わりに KPROPDATA や KCVT などの関数を使用してデータをトランスコーディングすることをお勧めします。ENCODING=データセットオプション、ILIBNAME ステートメントの NENCIDING=オプション/OUTENCODING=オプションのいずれかでそれらのエンコーディング名の 1 つを指定することにより、CEDA のトランスコーディングを回避できます。

注意

これらのエンコーディング値を誤って使用すると、データが破損する可能性があります。ASCIIANY はトランスコーディングを防止します。

- ANY を使用して、EBCDIC エンコーディングと ASCII エンコーディング間でもトランスコーディングを実行しないように指定します。

注: ANY は BINARY の指定と同等です。データはバイナリであるため、実際のエンコーディングには意味がありません。

- ASCIIANY を使用して、すべての ASCII ベースのエンコーディングと互換性があるデータセットを作成できます。ASCIIANY を使用して SAS データセットに設定されるエンコーディング値は US-ASCII です。
- EBCDICANY を使用して、すべての EBCDIC ベースのエンコーディングと互換性があるデータセットを作成できます。

これらのエンコーディングの 1 つを使用してトランスコーディングを無効にする前に、データセット内のデータについて知っておくことが重要です。たとえば、データセットのエンコーディングが WLATIN1 でセッションエンコーディングが WLATIN2 の場合、エンコーディングに互換性がないため、CEDA はデータをトランスコードします。ただし、データセット内のすべての文字が ASCII 文字であることがわかっている場合には、LIBNAME ステートメントの INENCODING オプションとして ASCIIANY を指定して、トランスコーディングを防止してください。

注: UTF-8 セッションエンコーディングで SAS を実行している場合、UTF-8 または US-ASCII 以外のデータセットは互換性がありません。

```
libname myfiles 'path to data sets' inencoding=asciiany;
```

データセットのエンコーディングは変更されませんが、CEDA は、データセットエンコーディングが LIBNAME ステートメントで指定されたエンコーディングであるとみなします。

データセットヘッダーのエンコーディングを変更して、トランスコーディングを防止することもできます。これを行うには、PROC DATASETS MODIFY ステートメントの CORRECTENCODING オプションを使用します。

```
libname myfiles "path to data sets";
proc datasets library=myfiles;
  modify olddata / correctencoding=ASCIIANY;
quit;
```

これらの関数を使用して、エンコーディングに互換性があることを確認できます。

- ENCODCOMPAT 関数は 2 つのエンコーディング間のトランスコーディング互換性を確認します。詳細については、“[ENCODCOMPAT 関数](#)”を参照してください。
- ENCODISVALID 関数は、有効なエンコーディング名を指定します。詳細については、“[ENCODCOMPAT 関数](#)”を参照してください。

動作環境別のエンコーディングの一覧については、[24 章, “SAS セッションのエンコーディング値”](#)を参照してください。

表 4.2 トランスコーディングのための CEDA 処理が必要ないエンコーディングの組み合わせ

エンコーディング方法	エンコーディング
SBCS ASCII	LATIN1、LATIN9
	LATIN1、WLATIN1 ヒント これらの 2 つのエンコーディングの間では、CEDA を使用したトランスコーディングが実行されません。WLATIN1 は、80 から 9F(16 進数)の範囲の制御文字ではなく、表示可能な文字を使用する点で LATIN1 と異なります。この違いは、MS WORD ドキュメントからの切り取り/貼り付けで問題を引き起こす可能性があります。
SBCS EBCDIC	EBCDIC037、OPEN_ED-037
	EBCDIC1025、OPEN_ED-1025
	EBCDIC1026、OPEN_ED-1026
	EBCDIC1047、OPEN_ED-1047
	EBCDIC1112、OPEN_ED-1112
	EBCDIC1122、OPEN_ED-1122
	EBCDIC1130、OPEN_ED-1130
	EBCDIC1140、OPEN_ED-1140
	EBCDIC1141、OPEN_ED-1141
	EBCDIC1142、OPEN_ED-1142
	EBCDIC1144、OPEN_ED-1144
	EBCDIC1145、OPEN_ED-1145

エンコーディング方法	エンコーディング
	EBCDIC1146、OPEN_ED-1146
	EBCDIC1147、OPEN_ED-1147
	EBCDIC1148、OPEN_ED-1148
	EBCDIC1160、OPEN_ED-1160
	EBCDIC1164、OPEN_ED-1164
	EBCDIC037、OPEN_ED-037
	EBCDIC275、OPEN_ED-275
	EBCDIC424、OPEN_ED-424
	EBCDIC425、OPEN_ED-425
	EBCDIC838、OPEN_ED-838
	EBCDIC870、OPEN_ED-870
	EBCDIC875、OPEN_ED-875
	EBCDIC924、OPEN_ED-924
DBCS ASCII	SHIFT-JIS、MS-932、IBM-942、MACOS-1
	EUC-KR、MS-949、MACOS-3
	EUC-CN、MS-936、DEC-CN、MACOS-25
	EUC-TW、DEC-TW、EUC-TW
	BIG5、MACOS-2、MS-950
DBCS EBCDIC	IBM-930、OPEN_ED-930
	IBM-933、OPEN_ED-933
	IBM-935、OPEN_ED-935
	IBM-937、OPEN_ED-937
	IBM-939、OPEN_ED-939

エンコーディング方法	エンコーディング
	IBM-939E、OPEN_ED-939E
	IBM-930E、OPEN_ED930E

セッションエンコーディングとデータセットエンコーディング、注、警告、エラー

SAS データセットの作成に使用されたエンコーディングは、データセットヘッダーに保存されます。たとえば、WLATIN1 のセッションエンコーディングで SAS を実行しているときにデータセットを作成すると、WLATIN1 エンコーディングがデータセットヘッダーに格納されます。SAS は文字データのエンコーディングが SAS セッションのエンコーディングと同じであると判断します。エンコーディングが一致しない場合、CEDA を使用してデータをトランスコードする必要があります。

データセットエンコーディングとセッションエンコーディングが一致せず、メッセージが NOTE の場合、文字データは有効です。たとえば、UTF-8 セッションが WLATIN1 エンコーディングで作成されたデータセットを読み取ると、ログに次のような注が表示されます。

NOTE: Data file MYLIB.CARS.DATA is in a format that is native to another host, or the file encoding does not match the session encoding. Cross Environment Data Access will be used, which might require additional CPU resources and might reduce performance.

データセットエンコーディングとセッションエンコーディングが互換性がなく、トランスコーディングが失敗すると、メッセージが WARNING または ERROR の場合、次のようになります。

WARNING: Some character data was lost during transcoding in the data set "MYLIB.CARS". It contains one or more characters that are not available in the new encoding. To avoid the transcoding error, please refer to "Troubleshooting Truncation and Data loss Issue during Transcoding" in SAS National Language Support (NLS): reference guide.

このエラーは、データセットのオブザベーションバッファの 1 つ以上の文字列に、データをセッションエンコーディングに変換するための十分なスペースがないか、セッションエンコーディングで文字を表現できないことを意味する場合があります。

データセットにセッションエンコーディングでサポートされていない文字が含まれている場合、メッセージはトランスコーディング中に文字が失われたことを意味します。たとえば、セッションのエンコーディングが西ヨーロッパ言語の文字をサポートする WLATIN1 で、SAS が SHIFT-JIS で日本語の文字を含むデータセットを読み取った場合、SAS ログのメッセージは、トランスコーディング中に日本語文字が失われたことを意味します。

トランスコーディング中の切り捨てとデータ損失の問題のトラブルシューティング

通常、トランスコーディングは問題なく実行されます。ただし、データの損失や切り捨てが発生する可能性があります。データの切り捨てが発生する 1 つの例は、あるエンコーディングの文字のバイト数が、別のエンコーディングの同じ文字のバイト数と異なる場合です。シングルバイト文字セット(SBCS)からマルチバイトエンコーディング(UTF-8)へのコード変換では、出力先バッファの長さが十分でない場合、文字値が切り捨てられる可能性があります。

2 つのエンコーディングで異なるバイト数を持つ文字の例を次に示します。この例では、引用符は WLATIN1 で 1 バイト、UTF-8 で 3 バイトです。

注: “[WLATIN1 から UTF-8 へのデータの移行](#)”セクションに、WLATIN1 から UTF-8 へのデータの変換に関する詳細情報があります。

表 4.3 WLATIN1 および UTF-8 の引用符の文字バイト

文字	説明	WLATIN1 コードポイント (16 進表現)	UTF-8 コードポイント(16 進表現)
,	下つき二重引用符	84	E2 80 9E

あるエンコーディングの文字が、別のエンコーディングではサポートされていない可能性があります。たとえば、特定のエンコーディングにはドル記号(\$)の文字が含まれていない場合があります。ドル記号をサポートしていないエンコーディングにデータをトランスコードすると、文字データは失われます。この損失は、互換性のないエンコーディング間のトランスコーディング時に発生します。互換性がなく、文字データの損失を引き起こす可能性があるエンコーディングについては、“[互換性があるエンコーディングと互換性がないエンコーディング](#)”を参照してください。

トランスコードの問題の詳細については、“[トランスコーディングの留意点](#)”を参照してください。

切り捨てやデータ損失の問題を解決するには、いくつかの方法があります。

表 4.4 データの切り捨てを回避する方法

方法	説明
CVP エンジン	CVP エンジンを使用して、文字の損失や切り捨てを回避できます。 詳細と例については、“CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE= オプション”を参照してください。
%COPY_TO_NEW_ENCODING マクロ	%COPY_TO_NEW_ENCODING マクロを使用して、文字の損失や切り捨てを回避できます。 詳細と例については、“%COPY_TO_NEW_ENCODING マクロ”を参照してください。
%COPY_TO_UTF8 マクロ	%COPY_TO_UTF8 マクロを使用して、UTF-8 エンコーディングで新しいバージョンのデータセットを作成できます。
KPROPDATA 関数	KPROPDATA 関数を使用して、文字の損失や切り捨てを回避できます。 詳細と例については、“KPROPDATA 関数”を参照してください。

CVP エンジンの使用による文字データ切り捨ての回避

データセットのエンコーディングが SAS セッションのエンコーディングと一致しない場合、データセット内の文字データのエンコーディングを変更すると、データが占める容量が増えるかも知れません。たとえば、SAS セッションが UTF-8 エンコーディングを使っているとき、Windows cp 1252(WLATIN1)エンコーディングのデータセットを読み込んだ場合、WLATIN1 では 1 バイトしか要しなかった文字が、UTF-8 では 2 バイトまたは 3 バイトを要する可能性があるのです。データの切り捨てを避けるには、文字変数ごとに、元のデータの幅より 1.5 倍広い幅が必要になります。

トランスコーディングを必要とする SAS データファイルを処理する場合は、CVP (文字変数埋め込み)エンジンに文字変数の長さの拡張を要求して、文字データの切り捨てが発生しないようにできます (変数の長さは、各変数の値を保存するために使用するバイト数です)。文字データの切り捨ては、1 つのエンコーディングにおける 1 文

字のバイト数が、別のエンコーディングにおける同じ文字のバイト数と異なる場合に発生する可能性があります。

たとえば、1 バイト文字セット(SBCS)を、2 バイト文字セット(DBCS)やマルチバイト文字セット(MBCS)にトランスコードする場合などです。SBCS では各文字を 1 バイトで表すのに対し、DBCS では各文字を 1 または 2 バイトで表します。MBCS は、1 から 4 バイトのさまざまな長さで文字を表します。たとえば、WLATIN1 から UTF-8 などの Unicode エンコーディングにトランスコーディングする場合、変数の長さ(バイト単位)は値を保持するのに不十分であり、結果として文字データが切り捨てられることとなります。

CVP エンジンを使用して、トランスコーディングの前に変数の長さが拡張されるように拡張量を指定し、それからデータを処理します。CVP エンジンは、トランスコーディングに向けてデータの準備を行うために使用される中間エンジンと考えてください。長さが増えた後は、デフォルトのベースエンジンなどの主エンジンを使用して実際のファイル処理が行われます。

CVP エンジンは、SAS データファイル限定の読み取り専用エンジンです。文字変数の拡張要求(たとえば、LIBNAME ステートメントを使用)は、次のいずれかの方法で実行できます。

- CVP エンジンを明示的に指定し、変数の長さの 1.5 倍のデフォルト拡張を使用します。
- LIBNAME の CVPBYTES=オプションまたは CVPMULTIPLIER=オプションにより、CVP エンジンを暗黙的に指定します。これらのオプションにより、拡張量を指定します。さらに、CVPENGINE=オプションを使用して、SAS ファイルの処理で使用する主エンジンを指定できます。デフォルトは、デフォルトの SAS エンジンです。

たとえば、次の LIBNAME ステートメントでは、CVP エンジンを明示的に割り当てています。文字変数の長さは、長さを 1.5 倍にするデフォルト拡張を使用して増やします。たとえば、長さが 10 の文字変数の新しい長さは 15 になり、長さが 100 の文字変数の新しい長さは 150 になります。

```
libname expand cvp ' SAS data-library';
```

注: 拡張量は、どのような拡張にでも対応できるよう十分大きくする必要があります。そうしない場合は、切り捨てが発生します。エラー情報は SAS ログに出力されます。

注: WHERE 式を使用してオブザベーションのサブセットを条件付きで選択する処理の場合は、CVP エンジンを使用するとパフォーマンスに影響することがあります。CVP エンジンを使用しない場合のファイルの処理は、CVP エンジンを使用する場合のファイルの処理より高速です。たとえば、CVP エンジンを使用する場合は、データセットにインデックスがあっても、WHERE 式を最適化するためにそのインデックスは使用されません。

注: “CASNCHARMULTIPLIER= システムオプション”(SAS システムオプション: リファレンス)は CAS サーバーでの UTF-8 へのトランスコーディングの際にバイト数を増やします。

注: CVP は READONLY のデータコピーを作成します。データの永続的なコピーを保存するには、新規にデータセットを作成する必要があります。

注: データセットと変数のラベルは最大 256 バイト、変数名は最大 32 文字です。データセットに国際的な文字を含む長いラベルが含まれていると、トランスコードされたラベルが最大値を超え、トランスコードエラーが発生する可能性があります。元のデータセットで長いラベルを短くして、トランスコーディングの問題を解決できます。

CVPFORMATWIDTH オプションは、文字出力形式の幅を拡張するかどうかを指定します。CVPFORMATWIDTH のデフォルトでは YES です。

この例は、CVP エンジンを使用した場合の出力形式の幅の PROC CONTENTS 出力の違いを示しています。

CVP エンジンは最初の例では指定されていません。libname in 'C:\temp\datalatin'; 2 番目の例では、CVP エンジンが指定されています。libname in1 cvp 'c:\temp\datalatin'; コードの形式は \$CNTRY24 であることに注意してください。コードを実行すると、CVPFORMATWIDTH がデフォルトで YES に設定されているため、CVP が指定された出力形式の結果 in1 は \$CNTRY36 になります。CVP が指定されていないコードの出力形式 in は、CVP が指定されておらず CVPFORMATWIDTH が使用できないので、\$CNTRY24 となります。

```
options dlcreatedir;
libname in 'C:\temp\datalatin';
proc format;
  value $cntry
    'USA' = 'United States of America'
    'Mexico' = 'United Mexican States';
run;
```

```
data IN.COUNTRY;
  length country $10;
  country = 'USA'; output;
  country = 'Mexico'; output;
  format country $cntry24.;
run;
```

```
libname in1 cvp 'c:\temp\datalatin';
title 'cvp engine';
proc contents data=in1.country;run;
```

```
libname in 'c:\temp\datalatin';
title 'no cvp engine';
proc contents data=in.country;
run;
```

この PROC CONTENTS 出力は、CVP サポートを指定する in1 の結果を示しているため、デフォルトでフォーマット拡張が利用可能です。IN1.COUNTRY の形式は \$CNTRY36 であることに注意してください。

cvp engine**The CONTENTS Procedure**

Data Set Name	IN1.COUNTRY	Observations	2
Member Type	DATA	Variables	1
Engine	CVP	Indexes	0
Created	08/17/2020 10:22:07	Observation Length	16
Last Modified	08/17/2020 10:22:07	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information	
Underlying Engine Accessing Data	V9
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	6470
Obs in First Data Page	2
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\temp\data\latin\country.sas7bdat
Release Created	9.0401M7
Host Created	X64_10PRO
Owner Name	CARYNT\daharr
File Size	128KB
File Size (bytes)	131072

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format
1	country	Char	15	\$CNTRY36.

この出力は、CVP サポートを指定していない *in* の結果を示しているため、出力形式の拡張が利用できません。IN.COUNTRY の形式は\$CNTRY24 であることに注意してください。

no cvp engine**The CONTENTS Procedure**

Data Set Name	IN.COUNTRY	Observations	2
Member Type	DATA	Variables	1
Engine	V9	Indexes	0
Created	08/17/2020 10:22:07	Observation Length	10
Last Modified	08/17/2020 10:22:07	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	6470
Obs in First Data Page	2
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\temp\datalatin\country.sas7bdat
Release Created	9.0401M7
Host Created	X64_10PRO
Owner Name	CARYNT\daharr
File Size	128KB
File Size (bytes)	131072

Alphabetic List of Variables and Attributes				
#	Variable	Type	Len	Format
1	country	Char	10	\$CNTRY24.

詳細と例については、“CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE=オプション” (1010 ページ)を参照してください。

%COPY_TO_NEW_ENCODING マクロを使用した文字の切り捨ての回避

セッションエンコーディングと異なるエンコーディングを使用して CNTLOUT=データセットを作成すると、文字変数の切り捨てのエラーが発生することがあります。このエラーが発生した場合、現在のセッションエンコーディングを使用して CNTLOUT=データセットを書き込み、%COPY_TO_NEW_ENCODING マクロを使用して、新しいエンコーディングで CNTLOUT=データセットを作成します。

エンコーディング間でサポートされていない文字の例を次に示します。ユーロの通貨値を含むデータセットがあり、セッションエンコーディングが WLATIN1 で、CNTLOUT=データセットに UTF-8 エンコーディングを指定する場合、一部の文字は互換性がありません。WLATIN1 の Label 変数の長さは 5 バイトで、値は 1234 ユーロ(16 進数では'8031323334'x)です。この変数を UTF-8 エンコーディングの CNTLOUT=データセットに格納しようとする、文字列の長さは 7 バイト(16 進数では'E282AC31323334')になるため、切り捨てが発生します。

%COPY_TO_NEW_ENCODING マクロを使用すると、このエラーを回避できます。

%COPY_TO_NEW_ENCODING マクロを使用して文字の損失を回避する方法の別の例を次に示します。

LATIN1 SAS セッションから次のコードを実行します。

```
data indata; 1
  price = "8031323334"x;
run;

libname outlib "myutf8lib" INENCODING='UTF-8' OUTENCODING='UTF-8';
2

proc copy out=outlib in=work noclone; 3
  select indata;
run;

proc contents data=work.indata; run; 4

%macro copy_to_new_encoding(indata,outdata,UTF8); 5

proc copy in=work out=outlib noclone; 6
  select outdata;
run;

proc contents data=outlib.outdata; 7
run;

libname out2 "outlib" INENCODING='binary' OUTENCODING='binary';
8

data _null_; 9
  set out2.outdata;
```

```
put price=$HEX10.;
run;
```

- 1 ユーロ文字を含むデータセット INDATA を作成します。
- 2 UTF-8 エンコーディングのライブラリ参照名を割り当てます。
- 3 データセット INDATA(WLATIN1)を OUTLIB(UTF-8)にコピーします。このエラーメッセージは、一部の文字が WLATIN1 と UTF-8 の間で互換性がないために表示されます。

NOTE: Data file OUTLIB.INDATA.DATA is in a format that is native to another host, or the file encoding does not match the session encoding. Cross Environment Data Access will be used, which might require additional CPU resources and might reduce performance.

NOTE: There were 1 observations read from the data set WORK.INDATA.

ERROR: Some character data was lost during transcoding in the dataset OUTLIB.INDATA. Either the data contains characters that are not representable in the new encoding or truncation occurred during transcoding.

ERROR: File OUTLIB.INDATA.DATA has not been saved because copy could not be completed.

- 4 PROC CONTENTS は、エンコーディング値を UTF-8 として表示し、price の可変長を 5 として表示します。
- 5 %macro copy_to_new_encoding は、データを INDATA から OUTDATA にコピーし、データを UTF-8 に変換します。
- 6 PROC COPY は、データセット OUTDATA を OUTLIB ライブラリにコピーします。CEDA は呼び出されず、データ損失は発生しません。
- 7 PROC CONTENTS は、OUTDATA 情報を表示します。エンコーディングは UTF-8 で、price の可変長は 7 です。
- 8 LIBNAME はライブラリ参照名 out2 を作成します。
- 9 現在、price の値は切り捨てられていません。UTF-8 での price の 16 進表現は E282AC31323334 です。

%COPY_TO_NEW_ENCODING マクロの詳細については、
"%COPY_TO_NEW_ENCODING マクロ"を参照してください。

KPROPDATA 関数を使用した CEDA トランスコーディング中の文字データ損失の回避

ソースとセッションのエンコーディングが同じでない場合、CEDA はデータを読み取り、変換します。エンコーディングが異なるデータセットを読み書きする際、互換性がない場合は、トランスコーディングプロセス中に文字の損失が発生する可能性があります。

KPROPDATA 関数を使用して、文字の損失を回避できます。

LATIN1 文字が EUC-JP エンコーディングに変換されるときに、KPROPDATA 関数を使用して文字損失を回避する例を次に示します。

LATIN1 SAS セッションから次のコードを実行します。

```
libname mylib 'my_library'; /* 1

data mylib.indata; /* 2
  a='My Text 2021PBCB$N/A'; /* 3
run;

proc contents data=mylib.indata; /* 4
run;
```

- 1 ライブラリ参照名 mylib を作成します。
- 2 データセット INDATA を作成します。
- 3 文字列 **My Text 2021PBCB\$N/A** を変数 a に割り当てます。
- 4 PROC CONTENTS を使用して、データセット INDATA に関する情報を出力します。エンコーディングは LATIN1 です。

データセットの出力は次のようになります。

The CONTENTS Procedure

Data Set Name	MYLIB.INDATA	Observations	1
Member Type	DATA	Variables	1
Engine	V9	Indexes	0
Created	05/12/2021 16:09:54	Observation Length	20
Last Modified	05/12/2021 16:09:54	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64, LINUX_POWER_64		
Encoding	latin1 Western (ISO)		

データセット INDATA が EUC-JP セッションから印刷される場合、サポートされていない文字があるため、CEDA 警告が発行されます。

```
options msglevel=i;
proc print data=mylib.indata;
run;
```

SAS は次の出力をログに書き込みます。

```
INFO: Data file MYLIB.INDATA.DATA is in a format that is native to another host, or the file encoding does
not match the session encoding.
Cross Environment Data Access will be used, which might require additional CPU resources and might
reduce performance.
WARNING: Some character data was lost during transcoding in the data set "MYLIB.INDATA".
It contains one or more characters that are not available in the new encoding.
To avoid the transcoding error, please refer to "Troubleshooting Truncation and Data loss Issue during
Transcoding" in SAS National Language Support (NLS): reference guide.
NOTE: There were 1 observations read from the data set MYLIB.INDATA.
INFO: One or more characters in the variable "a" at row number 1 in the
encoding "latin1" are not available in the encoding "euc-jp".
```

“KPROPDATA 関数”を使用して、サポートされていない文字を修正できます。

```
data indata(encoding='euc-jp');
set mylib.indata(encoding=binary); /* 1
```

```

a=KPROPDATA(a,"QUESTION","LATIN1","EUC-JP"); /* 2
run;

proc print data=indata; /* 3
run;

```

- 1 データセット INDATA にアクセスします。
- 2 KPROPDATA 関数を使用して、変数 a の文字列の 3 を?に置き換えます。LATIN1 が入力エンコーディングで、EUC-JP が出力エンコーディングです。

KPROPDATA 関数はサポートされていない文字を置き換えるため、ログに CEDA 警告はありません。

SAS は次の出力をログに書き込みます。

```

NOTE: There were 1 observations read from the data set WORK.INDATA.
NOTE: The PROCEDURE PRINT printed page 2.
NOTE: PROCEDURE PRINT used (Total process time):
      real time      0.00 seconds
      cpu time       0.01 seconds

```

サポートされていない文字は?に置き換えられます。 .

```

data _null_;
set indata; /* 1
put a=; /* 2
run;

```

- 1 データセット INDATA にアクセスします。
- 2 a の値をログに印刷します。文字列が KPROPDATA 関数によって処理されたため、サポートされていない文字は?に置き換えられました。

SAS は次の出力をログに書き込みます。

```

a=201743WP?C?$

```

2 バイト文字セット(DBCS)

DBCS サポートの概要	53
DBCS でサポートされる言語	54
DBCS の指定	54
DBCS 文字セットの表示要件	54
DBCS の機能	55
汎用機版 SAS での DBCS の利用法	55
DBCS エンコーディングの SAS データ変換	56
DBCS 文字列の取り扱い方	57

DBCS サポートの概要

東アジア言語には数千の文字があるため、多くの文字を表すには 2 バイトの情報が必要です。

東アジアの各言語では、コンピューターのメーカー間での標準化が行われていないため、通常は複数の DBCS エンコーディングシステムがあります。SAS は、主要な東アジア言語を使用する各メーカーに固有の DBCS エンコーディング情報を処理します。

ソフトウェアを適切に拡張することにより、SAS を使用して次の機能を実行できます。

- 主要な東アジア言語のいずれかを SAS System の DBCS バージョンで表示します。
- 東アジア言語のコンピューターからデータをインポートし、アプリケーションまたは動作環境から別のアプリケーションまたは動作環境にデータを移動します (この場合は、SAS ACCESS や他の SAS 製品が必要になります)。
- 東アジアでの標準的な日付と時間の表記を、SAS 日付値、SAS 時間値および SAS 日時値に変換します。

- 東アジア言語の文字を含むデータセットおよびさまざまなタイプの出力(レポートやグラフなど)を作成します。

DBCS でサポートされる言語

次の言語が DBCS でサポートされます。

- 簡体字中国語(中華人民共和国とシンガポールで使用)
- 繁体字中国語(香港特別行政区、マカオ特別行政区、台湾で使用)
- 日本語
- 韓国語

DBCS の指定

DBCS を指定するには、次の SAS システムオプションを使用します。

ENCODING

デフォルトの文字セットエンコーディングを認識します。

DBCS

DBCS 文字を認識します。

セッションエンコーディングを設定するには、ENCODING=システムオプション、DBCS オプションまたは LOCALE=システムオプションを使用します。ENCODING=システムオプションは優先されます。詳細については、“[SAS セッションエンコーディングの設定](#)”を参照してください。“[ENCODING システムオプション](#)”を参照してください。

DBCS 文字セットの表示要件

DBCS 文字を含むデータセットを表示するには、次のリソースが必要です。

- システムによる複数コードページのサポート
- 使用する言語に対応する DBCS フォント

ユーザー定義文字を作成して SAS ソフトウェアで使用する必要がある場合は、使用するコンピューターが DBCS をサポートしている必要があります。これらのコンピューターは、米国およびヨーロッパでは数が限られています。東アジア言語のコン

ピューターシステムでは、さまざまな方法を使用して文字を作成します。よく使用されている方法として、ラテン語の文字を使用して、文字の音声発音を入力する方法があります。コンピューターは、入力された音声発音に似た音を持つ文字のメニューを表示し、ユーザーに 1 つ選択するよう求めます。

DBCS の機能

特定の DBCS 言語と動作環境を認識する SAS セッションをセットアップした後で、次の一般領域で指定した言語を操作できます。

- DATA ステップおよびバッチ指向のプロシジャ
- ウィンドウ機能と対話型の機能
- システム間の接続性と互換性
- データベースへのアクセス
- グラフィック

DATA ステップおよびバッチ指向のプロシジャでは、引用符内の文字列として許容される場所であればどこでも DBCS を使用できます。変数値、変数ラベルおよびデータセットラベルは、すべて DBCS にできます。DBCS は、入力データとして、また FORMAT プロシジャでの範囲指定とラベル指定で使用することもできます。WHERE 式の処理では、埋め込み DBCS テキストを検索できます。

汎用機版 SAS での DBCS の利用法

汎用機システムで使用される別のタイプの DBCS エンコーディングでは、DBCS サポートと 3270 型データストリームを組み合わせています。それぞれの DBCS 文字列は、シフトアウト/シフトイン(または SO/SI)と呼ばれるエスケープコードで囲まれています。これらのコードは、旧型のプリンターを EBCDIC 文字セットから DBCS 文字セットにシフトアウトする必要があったために採用されました。主要なメーカーでは、さまざまな方法で SO/SI のエンコーディングを行います。一部のメーカーは DBCS コードに 1 バイトのシフトコード情報を埋め込むのに対し、他のメーカーは 2 バイトのシフトコード情報を DBCS コードに埋め込みます。これらの相違により、汎用機に関する DBCS 情報を読み込むときに問題が発生します。

PC、ミニコンピューターおよびワークステーションに SO/SI はありませんが、メーカーによって異なる独自の DBCS エンコーディングが使用されています。SAS では、SO/SI システムで DBCS を読み込める数種類の出力形式と入力形式を備えています。

表 5.1 SO/SI システムでの DBCS をサポートする SAS の出力形式と入力形式

キーワード	言語要素	説明
\$KANJI	入力形式	日本語の漢字 DBCS から SO/SI を削除します
\$KANJIX	入力形式	日本語の漢字 DBCS に SO/SI を追加します
\$KANJI	出力形式	日本語の漢字 DBCS に SO/SI を追加します
\$KANJIX	出力形式	日本語の漢字 DBCS から SO/SI を削除します

DBCS エンコーディングの SAS データ変換

通常、1つのコンピューターシステムで生成された DBCS データは、別のコンピューターシステムで生成されたデータと互換性がありません。SAS には、次の表に示すように、1つの DBCS ソースから別の DBCS ソースへの変換を可能にする機能があります。

表 5.2 DBCS 変換

言語要素	タイプ	使用	参照先
KCVT	関数	1つの動作環境から別の動作環境へ DBCS データを変換します	“KCVT 関数” (p. 628)
CPORT	プロシジャ	1つの環境から別の環境へファイイルを移動します	<i>Base SAS プロシジャガイド</i>
CIMPORT	プロシジャ	CPORT によって作成されたトランスポートファイイルをインポートします	<i>Base SAS プロシジャガイド</i>

DBCS 文字列の取り扱い方

- DBCS 文字を操作する場合は、データを見直して、DATA ステップまたは PROC ステップでデータをインポート、変換または使用するときに SAS が文字列全体を認識することを確認してください。
- シフトアウト/シフトインエスケープコードを使用する汎用機システムでは、動作環境間での変換中に DBCS 文字列が切り捨てられる可能性があります。
- PRINT、REPORT、TABULATE、FREQ プロシジャを使用して処理する場合には、DBCS 文字列が分割される可能性があります。不適切な分割が発生した場合は、DBCS 文字列のどちらかの側にスペースを追加して、強制的に適切な位置で分割されるようにする必要があります。PROC REPORT および PROC PRINT で SPLIT=オプションを使用し、強制的に適切な位置で文字列を分割させることもできます。

タイムゾーン

SAS タイムゾーンの概要	59
SAS でのタイムゾーンの指定	59
定義	59
SAS でのタイムゾーンについて	60
タイムゾーン処理のタスクと SAS 言語要素	62
タイムゾーンの例	67

SAS タイムゾーンの概要

国際的な顧客に対応するためには、ローカルおよび国際の時間値と日付値を処理する必要があります。この章では、タイムゾーンと次の処理方法について説明します。

- タイムゾーン
- 夏時間(DST)
- ローカル時間
- SAS 日時値

SAS でのタイムゾーンの指定

定義

夏時間(DST)

Summer Time と呼ばれる *Daylight Saving Time (DST)* は、夏の日の長い時期に時計を 1 時間繰り上げて夕方の日照を有効に利用し、秋には元に戻すという方法です。北半球の夏時間は 3 月から 4 月までの間に始まり、9 月から 11 月までの

間に終了します。北半球の標準時間は9月から11月の間に始まり、3月と4月の間に終了します。南半球の夏時間は9月から11月までの間に始まり、3月から4月までの間に終了します。南半球の標準時間は3月と4月の間に始まり、9月から11月の間に終了します。北半球の多くの国では、夏時間を採用していません。夏時間は、すべての国で採用されているわけではありません。

ローカル時間

TIMEZONE=システムオプションが設定されなかった場合に使用されるタイムゾーンのローカル時間を指定します。

タイムゾーン

同一の標準時間を使用する地球上のエリア。通常はローカル時間と呼ばれます。タイムゾーンには、政治的な境界や地理的な境界があり、地域住民の便宜を考慮して調整される場合があります。インドや中国などのように、地理的に大きな国で1つのタイムゾーンを使用している国もあれば、ロシアやUSなどのように、複数のタイムゾーンを持つ国もあります。

タイムゾーン ID

フォワードスラッシュ(/)で区切られた地域とエリア。Asia/Tokyo は、タイムゾーン ID の1つです。タイムゾーン ID は、Java タイムゾーンと互換性があります。タイムゾーン ID のリストについては、[付録 2, "タイムゾーン ID とタイムゾーン名"](#)を参照してください。

タイムゾーン名

タイムゾーンを示す3文字または4文字を指定します。タイムゾーン名のリストについては、[付録 2, "タイムゾーン ID とタイムゾーン名"](#)を参照してください。

タイムゾーンオフセット

+|-hh:mm または+|-hhmm という形式で示した、協定世界時(UTC)からタイムゾーンまでの時差。

ユーザーのローカル時間

TIMEZONE=システムオプションで指定されたタイムゾーンのローカル時間を指定します。

協定世界時(UTC)

基準子午線(英国のグリニッジ付近)の時間を指定します。UTC は、ISO 8601 基本形式 `yyyymmddThhmmss+|-hhmm`、または ISO 8601 拡張形式 `yyyy-mm-ddThh:mm:ss+|-hh:mm` が使用された日時値です。

SAS でのタイムゾーンについて

SAS 日時値は、ローカル時間の 1960 年 1 月 1 日 00:00:00 から経過秒数です。たとえば、英国における 1960 年 1 月 1 日 00:00:00 の SAS 日時値は 0 です。日本における 1960 年 1 月 1 日 00:00:00 の SAS 日時値も同様に 0 になります。英国と日本の間には 9 時間の時差があります。同じ瞬間の時刻であれば、この日時値は英国と日本の両方で 0 にはできません。絶対時間と日時値に対応するために、SAS は UTC 日付値と、タイムゾーンの時間値および日時値をサポートしています。

タイムゾーンの指定には、タイムゾーン ID またはタイムゾーン名を使用できます。タイムゾーン ID は、`region/area`(地域/エリア)という形式で地域とエリアを指定します。America/New_York は、タイムゾーン ID の1つです。タイムゾーン名は、タイムゾーンを指定します。東部標準時間(Eastern Standard Time)に使用される EST

は、タイムゾーン名の 1 つです。タイムゾーンが夏時間に切り替わる時、夏時間として別のタイムゾーン名が用意されています。PST は太平洋標準時間です。PDT は太平洋夏時間です。EET は東欧時間です。EEST は東欧夏時間です。タイムゾーン ID を指定すると、SAS によって夏時間かどうかが決定されます。地域/エリアが標準時間または夏時間であるかをユーザーが判断する必要はありません。ゾーン ID とタイムゾーン名のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名”](#)を参照してください。

タイムゾーンを指定すると、SAS はタイムゾーンに合わせて日時値をシフトし、新しい値を保存します。タイムゾーン固有のタイムスタンプが SAS ログに表示されます。SAS タイムスタンプ定数は、ISO 8601 標準のタイムスタンプを指定する際に使用できます。次に、SAS タイムスタンプ定数を使用したタイムスタンプを示します。

```
tstamp='2013-05-17T09:15:30-05:00'dt;
```

SAS タイムスタンプ定数を指定すると、現在のタイムゾーンとタイムゾーンオフセットを考慮して、タイムスタンプがローカル日時値に変換されます。

タイムゾーンの設定は、TIME()関数、TODAY()関数、DATE()関数、DATETIME()関数に影響します。タイムゾーンが設定されると、日付値と日時値はそのタイムゾーンの値になります。

この例は、タイムゾーンに応じて日時の値がどのように変化するかを示しています。

```
options tz='America/New_York'; 1
data NY;
  NY_DT=datetime(); 2
  format NY_DT datetime20.; 3
run;
options tz='Asia/Shanghai'; 4
data SH;
  SH_DT=datetime(); 5
  format SH_DT datetime20.; 6
run;

options tz='America/New_York'; 7
proc contents data = NY nodetails; 8
run;
options tz='Asia/Shanghai'; 9
proc contents data = NY nodetails; 10
run;
```

これらのステートメントは、前の DATA と PROC のステップで番号が付けられた行に対応しています。

- 1 TZ オプションを America/New_York に設定します。
- 2 DATETIME 関数を使用して、日付と時刻を変数 NY_DT に割り当てます。
- 3 DATETIME 形式を使用して NY_DT の値を変更します。
- 4 2 番目の DATA ステップの TZ オプションを Asia/Shanghai に設定します。
- 5 DATETIME 関数を使用して、変数 SH_DT に日付と時刻を割り当てます。
- 6 DATETIME 形式を使用して SH_DT の値を変更します。
- 7 最初の PROC CONTENTS ステップの TZ オプションを America/New_York に設定します。

- 8 データセット NY の情報を表示します。TZ オプションは America/New_York に設定されています。
- 9 2 番目の PROC CONTENTS ステップの TZ オプションを Asia/Shanghai に設定します。
- 10 Asia/Shanghai に設定されている TZ オプションを使用して、データセット NY の情報を表示します。Created 行と LastModified 行のタイムゾーンシフトに注意してください。同じデータセットに関する情報を表示していますが、タイムゾーンが異なります。

これらのステートメントは、次の結果を生成します。最初の出力は、America/New_York タイムゾーンを使用しているデータセットからのものです。2 番目の出力は、Asia/Shanghai のタイムゾーンを使用しているデータセットからのものです。

Data Set Name	WORK.NY	Observations	1
Member Type	DATA	Variables	1
Engine	V9	Indexes	0
Created	27/02/2019 17:17:42	Observation Length	8
Last Modified	27/02/2019 17:17:42	Deleted Observations	0

Data Set Name	WORK.NY	Observations	1
Member Type	DATA	Variables	1
Engine	V9	Indexes	0
Created	28/02/2019 06:17:42	Observation Length	8
Last Modified	28/02/2019 06:17:42	Deleted Observations	0

タイムゾーン処理のタスクと SAS 言語要素

タイムゾーンの設定

タイムゾーンの設定には、TIMEZONE=システムオプションを使用します。

```
options timezone='asia/tokyo';
```

このオプション値には、タイムゾーン名またはタイムゾーン ID を使用します。値は引用符で囲みます。タイムゾーン ID とタイムゾーン名については、[付録 2, “タイムゾーン ID とタイムゾーン名”](#)を参照してください。詳細については、“[TIMEZONE=システムオプション](#)”を参照してください。

タイムゾーン ID オフセットの特定

TZONEOFF()関数を使用して、タイムゾーン名またはタイムゾーン ID のオフセットを特定します。

- TZONEOFF()関数は、現在のタイムゾーンのタイムゾーンオフセットを返します。
- TZONEOFF('time-zone-id') 関数は、'time-zone-id'のタイムゾーンオフセットを返します。

このプログラムは、現在のタイムゾーン(EST)と東京のタイムゾーンのオフセットを返します。SAS ログの出力は次のようになります。

```
data _null_;
  o1=tzoneoff();
  o2=tzoneoff('asia/tokyo');
  put o1 time.;
  put o2 time.;
run;
```

```
-5:00:00
9:00:00
```

2つのタイムゾーン間の時差を特定するために、ABS()関数を使用できます。

```
diff=abs(tzoneoff('america/new_york') - tzoneoff('asia/tokyo'));
```

詳細については、“[TZONEOFF 関数](#)”を参照してください。

タイムゾーン ID またはタイムゾーン名の特定

次の関数を使用して、タイムゾーン名またはタイムゾーン ID を特定します。

- TZONEID()関数は現在のタイムゾーン ID を返します。
- TZONENAME()関数は、タイムゾーン ID と SAS 日時値の両方、または SAS 日時値のみに基づいて、現在のタイムゾーン名を返します。

次に、関数の使用例を示します。2013 年 3 月 10 日は夏時間の開始日です。

```
options timezone='America/Chicago';
data _null_;
  tzid=tzoneid();
  put 'Current time zone is ' tzid;
  tzn=tzonename('america/los_angeles');
  put 'Time zone for Los Angeles: ' tzn;
  tznST=tzonename('america/los_angeles','10mar2013:01:00:00'dt);
  put 'Time zone for Los Angeles standard time: ' tznST;
  tznDT=tzonename('america/los_angeles','10mar2013:02:00:00'dt);
  put 'Time zone for Los Angeles daylight time: ' tznDT;
  tznSDT=tzonename('10mar2013:02:00:00'dt);
  put 'Time zone name for this SAS datetime: ' tznSDT;
run;
```

出力は次のようになります。

```
Current time zone is AMERICA/CHICAGO
Time zone for Los Angeles: PST
Time zone for Los Angeles standard time: PST
Time zone for Los Angeles daylight time: PDT
Time zone name for this SAS datetime: CDT
```

詳細については、“[TZONED 関数](#)”および“[TZONENAME 関数](#)” (732 ページ)を参照してください。

SAS と UTC 間での日時値の変換

次の関数を使用して、SAS 日時値を UTC 日時値へ、UTC 日時値を SAS 日時値へ変換します。

- TZONES2U()は、SAS 日時値を UTC 日時値へ変換します。
- TZONEU2S()関数は、UTC 日時値を SAS 日時値へ変換します。

次に、関数の使用例を示します。

```
options timezone='est';
data _null_;
  put ' The time zone is EST';
  diff=abs(tzoneoff('america/new_york') - tzoneoff('europe/london'));
  put ' New York-London difference:' diff time.;
  diff=abs(tzoneoff('america/new_york') - tzoneoff('asia/tokyo'));
  put ' New York-Tokyo difference:' diff time.;
  put ' The SAS datetime is 2013-03-15T09:15:00+00:00 ';
  put '';
  put ' Change a SAS datetime to a UTC value ';
  put '';
  put ' The time zone offset +00:00 is for London ';
  put ' Subtract the 5 hours for the EST time zone offset';
  stu1=tzones2u('2013-03-15T09:15:00+00:00'dt);
  put ' STU1 Using E8601DX:' stu1 e8601dx.;
  put '';
  put ' 2013-03-15 9:15 AM in Tokyo is 2013-03-14 7:15 PM in New York';
  put ' Subtract the 5 hours for the EST time zone offset';
  stu2=tzones2u('2013-03-15T09:15:00+00:00'dt, 'Asia/Tokyo');
  put ' STU2 Using E8601DX:' stu2 e8601dx.;
  put '';
  put ' Change a UTC to a SAS datetime value. ';
  put '';
  put ' +00:00 is the time zone offset for London.';
  put ' Subtract the 5 hours for the EST time zone offset';
  uts1=tzoneu2s('2013-03-15T09:15:00+00:00'dt);
  put ' UTS1 Using DATETIME:' uts1 datetime.;
  put '';
  put ' 9:15:00+00:00 is 18:15:00 in Tokyo. ';
  put ' Subtract the 5 hours for the EST time zone offset';
  uts2=tzoneu2s('2013-03-15T09:15:00+00:00'dt, 'Asia/Tokyo');
  put ' UTS2 Using DATETIME:' uts2 datetime.;
run;
```

出力は次のようになります。

```
The time zone is EST
New York-London difference: 5:00:00
New York-Tokyo difference:14:00:00
The SAS datetime is 2013-03-15T09:15:00+00:00

Change a SAS datetime to a UTC value

The time zone offset +00:00 is for London
Subtract the 5 hours for the EST time zone offset
STU1 Using E8601DX: 2013-03-15T04:15:00-05:00

2013-03-15 9:15 AM in Tokyo is 2013-03-14 7:15 PM in New York
Subtract the 5 hours for the EST time zone offset
STU2 Using E8601DX: 2013-03-14T14:15:00-05:00

Change a UTC to a SAS datetime value.

+00:00 is the time zone offset for London.
Subtract the 5 hours for the EST time zone offset
UTS1 Using DATETIME:14MAR13:23:15:00

9:15:00+00:00 is 18:15:00 in Tokyo.
Subtract the 5 hours for the EST time zone offset
UTS2 Using DATETIME:15MAR13:13:15:00
```

詳細については、“[TZONES2U 関数](#)” (736 ページ)を参照してください。

タイムゾーンに基づいた ISO 8601 表記の時間値と日時値の書き込み

これらの出力形式では、ISO 8601 基本表記と拡張表記の時間値と日時値を書き込みます。時間、UTC オフセット、またはその両方が、TIMEZONE=システムオプションの値に基づきます。

- B8601TX.および E8601TX.出力形式では、UTC 時間値をユーザーのローカル時間に変換し、その時間値をタイムゾーンオフセット付きの形式で書き込みます。
- B8601DX.および E8601DX.出力形式では、UTC 日時値をユーザーのローカル時間に変換し、その日時値をタイムゾーンオフセットを使用して書き込みます。
- B8601LX.および E8601LX.出力形式では、ユーザーのローカルタイムゾーンの UTC オフセットを使用して、ローカル日時値を書き込みます。

このプログラムは、タイムゾーン ID の Australia/Sydney に基づき、時間値と日時値をフォーマットします。

```
options timezone='Australia/Sydney';
data _null_;
  st='18:33:40't;
  sdt='2013-03-17T14:30:22+00:00'dt;
  put 'Time B8601TX:' st b8601tx.;
  put 'Time E8601TX:' st e8601tx.;
  put 'UTC B8601DX:' sdt b8601dx.;
  put 'UTC E8601DX:' sdt e8601dx.;
  put 'UTC B8601LX:' sdt b8601lx.;
  put 'UTC E8601LX:' sdt e8601lx.;
```

```
run;
```

出力は次のようになります。

```
Time B8601TX:      043340+1000
Time E8601TX:04:33:40+10:00
UTC B8601DX:       20130318T123022+1100
UTC E8601DX: 2013-03-18T12:30:22+11:00
UTC B8601LX:       20130318T013022+1100
UTC E8601LX: 2013-03-18T01:30:22+11:00
```

詳細については、“[B8601DXw. 出力形式](#)”、“[B8601LXw. 出力形式](#)”、“[B8601TXw. 出力形式](#)”、“[E8601DXw. 出力形式](#)”、“[E8601LXw. 出力形式](#)”、と“[E8601TXw. 出力形式](#)”を参照してください。

タイムゾーンオフセットを使用した SAS 日時値の書き込み

次の出力形式では、TIMEZONE=システムオプションの値に基づき、UTC オフセットを使用して SAS 日時値を書き込みます。

- NLDATMZ.出力形式では、SAS 日時値を *ddmmmyyyy:hh:mm:ss +|-hhmm* 形式で書き込みます。
- NLDATMTZ.出力形式では、SAS 時間値を *hh:mm:ss +|-hhmm* 形式で書き込みます。
- NLDATMWZ.出力形式では、SAS 日時値から週の曜日、月、日、年、午前|午後の部分を、*day-of-week, month-name dd, yyyy AM|PM +hhmm* 形式で書き込みます。

このプログラムは、タイムゾーンオフセットを使用して、SAS 時間と日時値をフォーマットします。

```
options timezone='Indian/Maldives';
data _null_;
  st='18:33:40't;
  sdt='2013-03-17T14:30:22+00:00'dt;
  put 'Time NLDATMTZ:' st nldatmtz.;
  put 'SAS datetime NLDATMZ:' sdt nldatmz.;
  put 'SAS datetime NLDATMWZ:' sdt nldatmwz.;
run;
```

出力は次のようになります。

```
Time NLDATMTZ:18:33:40 +0500
SAS datetime NLDATMZ:17Mar2013:19:30:22 +0500
SAS datetime NLDATMWZ:Sunday, March 17, 2013 07:30:22 PM +0500
```

詳細については、“[NLDATMTZw. 出力形式](#)”、“[NLDATMZw. 出力形式](#)”、および“[NLDATMWZw. 出力形式](#)” (245 ページ)を参照してください。

タイムゾーンの例

この例では、ロサンジェルスから東京までのフライトの到着時間を使用します。

```
/* Set the time zone */
options timezone='America/Los_Angeles';
data depart;
/* Set the departure time */
  depart='2013-05-17T09:15:00-08:00'dt;
  put 'Depart Los Angeles: ' depart nldatmwz.;
/* Set the flight time */
  ftime='13:00't;
  put 'Flight time=' ftime time.;
  utc=depart+ftime;
  put 'Arrive PST=' utc nldatmwz.;
  put 'Arrive UTC=' utc nldatmwz.;
run;
/* Set the time zone for Tokyo */
options timezone='Asia/Tokyo';
data arrive;
  set depart;
  put 'Arrive in Tokyo ' utc nldatmwz.;
run;
```

出力は次のようになります。

```

39  /* Set the time zone */
40  options timezone='America/Los_Angeles';
41  data depart;
42  /* Set the departure time */
43  depart='2013-05-17T09:15:00-08:00'dt;
44  put 'Depart Los Angeles: ' depart nldatmwz.;
45  /* Set the flight time */
46  ftime='13:00't;
47  put 'Flight time=' ftime time.;
48  utc=depart+ftime;
49  put 'Arrive PST=' utc nldatmwz.;
50  put 'Arrive UTC=' utc nldatmwz.;
51  run;

Depart Los Angeles: Friday, May 17, 2013 10:15:00 AM -0700
Flight time=13:00:00
Arrive PST=Friday, May 17, 2013 11:15:00 PM -0700
Arrive UTC=Friday, May 17, 2013 11:15:00 PM -0700
NOTE: The data set WORK.DEPART has 1 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds

52  /* Set the time zone for Tokyo */
53  options timezone='Asia/Tokyo';
54  data arrive;
55  set depart;
56  put 'Arrive in Tokyo ' utc nldatmwz.;
57  run;

Arrive in Tokyo Friday, May 17, 2013 11:15:00 PM +0900
NOTE: There were 1 observations read from the data set WORK.DEPART.
NOTE: The data set WORK.ARRIVE has 1 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds

```

2 部

各国語サポート関連の自動呼び出しマクロ

7 章

各国語サポート関連の自動呼び出しマクロのディクショナリ	71
-----------------------------------	----

各国語サポート関連の自動呼び出しマクロのディクショナリ

カテゴリ別の自動呼び出しマクロエントリ	71
ディクショナリ	72
%COPY_TO_NEW_ENCODING マクロ	72
%COPY_TO_UTF8 マクロ	75
%KLOWCASE 自動呼び出しマクロ	76
%QKLOWCAS 自動呼び出しマクロ	77
%KTRIM 自動呼び出しマクロ	77
%QKTRIM 自動呼び出しマクロ	78
%KVERIFY 自動呼び出しマクロ	79

カテゴリ別の自動呼び出しマクロエントリ

次の表に、SAS 各国語サポート自動呼び出しマクロの概要を示します。詳細については、各マクロの各国語サポートエントリを参照してください。

カテゴリ	言語要素	説明
DBCS	%KLOWCASE 自動呼び出しマクロ (p. 76)	大文字を小文字に変更します。
	%QKLOWCAS 自動呼び出しマクロ (p. 77)	大文字を小文字に変更します。
	%KTRIM 自動呼び出しマクロ (p. 77)	末尾の空白を取り除きます。

カテゴリ	言語要素	説明
	%QKTRIM 自動呼び出しマクロ (p. 78)	末尾の空白を取り除きます。
	%KVERIFY 自動呼び出しマクロ (p. 79)	式に固有の最初の文字の位置を返します。
エンコーディング	%COPY_TO_NEW_ENCODING マクロ (p. 72)	指定したエンコーディングでデータセットの新しいバージョンを作成します。
	%COPY_TO_UTF8 マクロ (p. 75)	UTF-8 エンコーディングでデータセットの新しいバージョンを作成します。

ディクショナリ

%COPY_TO_NEW_ENCODING マクロ

指定したエンコーディングでデータセットの新しいバージョンを作成します。

カテゴリ: エンコーディング

注: %COPY_TO_NEW_ENCODING は、ソースデータエンコーディングと一致するシステムで実行する必要があります。

2022.1.2 以降、%COPY_TO_NEW_ENCODING マクロは AUTOCALL ライブラリにあります。以前のバージョンでは、[SAS® 9.4 各国語サポート\(NLS\): リファレンスガイド、第 5 版](#)のソース コードにアクセスできます。

%COPY_TO_UTF8 は、transcode プロパティが NO に設定されている変数をトランスコードしません。

構文

%COPY_TO_NEW_ENCODING(*from_dsname*, *to_dsname*, *new_encoding*)

引数

from_dsname
元のデータセットの名前を指定します。

to_dsname
新しいデータセットの名前を指定します。

new_encoding
新しいデータセットのエンコーディングを指定します。

注: 一般的なエンコーディングは UTF-8 と LATIN1 です。エンコーディングの一覧については、“[エンコーディングとその別名とエンコーディング文字セットの互換性](#)”を参照してください。

詳細

基本

%COPY_TO_NEW_ENCODING マクロは、指定したエンコーディングでデータセットの新しいバージョンを作成します。指定したエンコーディングにトランスコードする際、長さを拡張する必要がある値がデータセットに含まれている場合、DATA ステップは適切な長さでデータセットを作成します。%COPY_TO_NEW_ENCODING マクロが使用されない場合、空白以外の丸めは許容されないため、コピーが失敗する可能性があります。たとえば、wlatin1 エンコーディングが指定された次のコードは、トランスコードされた UTF-8 表現を保持するために変数 x で 3 バイトが必要なため失敗します。

注: %COPY_TO_NEW_ENCODING は、ソースデータエンコーディングと一致するシステムで実行する必要があります

たとえば、wlatin1 エンコーディングが指定された次のコードは、トランスコードされた UTF-8 表現を保持するために変数 x で 3 バイトが必要なため失敗します。

```
data temp;
  length x $1;
  x='€';
run;
data temp(encoding=utf8);
  set temp;
run;
```

%COPY_TO_NEW_ENCODING マクロは、次の種類のコードを生成します。

```
data temp(encoding=utf8);
  length x $3;
  set temp;
run;
```

例

この例では、SAS データセット SASHELP.CLASS を使用し、そのデータセットの変換されたコピーを作成して、データセット CLASS として WORK ライブラリーに保管します。データセット CLASS には UTF8 エンコーディングがあります。

例 1: LATIN9 データを UTF-8 へ

```
%COPY_TO_NEW_ENCODING(sashelp.class, class, utf8);
```

この例は、データが LATIN9 から UTF-8 に変換される方法を示しています。

```
data indata;
  cost = 'A4392E3939'x;          /* 1 */
run;

%copy_to_new_encoding(indata, outdata, utf8); /* 2 */
```

- 1 cost=A4392E3939 の値でデータセット indata を作成します。この値は、LATIN9 での€9.99 の 16 進表現です。
LATIN9 のユーロ(€)文字は 1 バイトの'A4'x を占め、入力データセットの変数 cost の長さは 5 です。UTF-8 エンコーディングでは、同じユーロ文字が 3 バイトの'E282AC'x を占めるため、変数の長さが 7 に変更されます。
- 2 %copy_to_new_encoding マクロは、入力データセット indata のデータを UTF-8 エンコーディングに変換し、データを出力データセット outdata に出力します。

例 2: %COPY_TO_UTF8 マクロ機能

この例は、%COPY_TO_UTF8 マクロの使用例を示しています。

```
data mydsU8;          /* 1 */
  set wlatin1.myds;
run;                  /* 2 */

%COPY_TO_UTF8(wlatin1,tempU8); /* 3 */

libname INU8CVP CVP "path_to_wlatin1_library" CVPMULTIPLIER=4; /* 4 */
data tempU8;
length variable_name $ new_length;
set INU8CVP.wlatin1;
run;

libname INU8CVP clear ;
```

- 1 このコードは、UTF-8 SAS セッションから WLATIN1 でエンコードされたデータセットを読み取ります。
- 2 変数の長さがトランスコードされた値を保持するのに十分な大きさでない場合、SAS はエラーメッセージを表示します。

注: ERROR: データセット"WLATIN1.MYDS"のトランスコーディング中に一部の文字データが失われました。新しいエンコーディングでは使用できない 1 つ以上の文字が含まれています。トランスコードエラーを回避するには、SAS National Language Support (NLS): Reference Guide の"トランスコーディング中の切り捨てとデータ損失の問題のトラブルシューティング"を参照してください。

- 3 %COPY_TO_UTF8 は、実際のデータサイズに基づいて変数に新しい長さを割り当て、データの切り捨てを回避することにより、データセットを生成します。
- 4 %COPY_TO_UTF8 によって実行される生成コード。

%COPY_TO_UTF8 マクロ

UTF-8 エンコーディングでデータセットの新しいバージョンを作成します。

カテゴリ: エンコーディング

注: %COPY_TO_UTF8 は、UTF-8 の SAS セッションで実行する必要があります。
2022.1.2 以降、%COPY_TO_UTF8 マクロは AUTOCALL ライブラリにあります。以前のバージョンでは、[SAS® 9.4 各国語サポート\(NLS\): リファレンスガイド、第 5 版](#)のソース コードにアクセスできます。

%COPY_TO_UTF8 は、CVP エンジンを使用してデータセットを読み取ります。
%COPY_TO_UTF8 マクロの実行中に、切り捨ての警告が表示される場合があります。これらの警告は正常であり、予期されたものです。
%COPY_TO_UTF8 は、transcode プロパティが NO に設定されている変数をトランスコードしません。

参照項目: 詳細については、“[%COPY_TO_NEW_ENCODING マクロ](#)”を参照してください。

構文

%COPY_TO_UTF8(*from_dsname*, *to_dsname*);

必須引数

from_dsname

元のデータセットの名前を指定します。

to_dsname

新しいデータセットの名前を指定します。

.....
注: %COPY_TO_UTF8 は、UTF-8 セッションから実行する必要があります。
.....

詳細

%COPY_TO_UTF8 マクロは UTF-8 エンコーディングでデータセットの新しいバージョンを作成します。

データセットに、UTF-8 にトランスコードするときに値を長くする必要のある文字変数が含まれている場合、マクロは、適切な長さでデータセットを再作成する SAS プログラムを生成します。UTF-8 にトランスコードするときにデータセットに長い長さを必要としない変数がある場合、データセットは同じままです。

%COPY_TO_UTF8 は、CVPMULTIPLIER=4 の CVP エンジンを使用して、指定されたデータセットを読み取り、切り捨ての問題を回避します。次に、マクロは各文字変数をトランスコードして、変数に必要な最大長(バイト単位)を決定します。

%COPY_TO_UTF8 マクロを使用すると、'ライブラリ参照名_INU8CVP は IN と同じ物理ライブラリを参照しています'という警告が表示されることがあります。'IN'は、入力データセット用に定義されたライブラリ参照名です。

たとえば、WLATIN1 エンコーディングが指定された次のコードは、トランスコードされた UTF-8 表現を保持するために変数 x で 3 バイトが必要なため失敗します。

```
data temp(encoding=wlatin1);
  length x $1;
  x='€';
run;
data temp(encoding=utf8);
  set temp;
run;
```

%COPY_TO_UTF8 は、次の種類のコードを生成します。

```
libname INU8CVP CVP path to work library;
data tempU8;
  length x $3;
  set INU8CVP.temp;
run;
```

%KLOWCASE 自動呼び出しマクロ

大文字を小文字に変更します。

カテゴリ:	DBCS
要件	MAUTOSOURCE システムオプション
参照項目:	"%QKLOWCAS 自動呼び出しマクロ"

構文

%KLOWCASE (text | text expression)

詳細

%KLOWCASE と %QKLOWCAS マクロは、大文字の英字を小文字に変更します。次のリストにある特殊文字またはニーモニック演算子が引数に含まれている可能性がある場合は、%QKLOWCAS を使用します。

%KLOWCASE は、引数に引用符が含まれる場合でも、引用符なしの結果を返します。%%QKLOWCAS は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します:

& % ' " () + - * / < > = ~ ^ ~ ; , blank AND OR NOT EQ NE LE LT GE GT IN

自動呼び出しマクロは SAS ライブラリに含まれます。このライブラリはサイトにインストールされていないか、またはサイト固有のバージョンの可能性があります。このマクロにアクセスできない場合、またはライブラリがサイト固有のバージョンかどうか確認する場合は、オンサイトの SAS サポート担当者にお問い合わせください。

%QKLOWCAS 自動呼び出しマクロ

大文字を小文字に変更します。

カテゴリ: DBCS
要件 MAUTOSOURCE システムオプション
参照項目: ["%KLOWCASE 自動呼び出しマクロ"](#)

構文

%QKLOWCAS (text | text expression)

詳細

%KLOWCASE と %QKLOWCAS マクロは、大文字の英字を小文字に変更します。次のリストにある特殊文字またはニーモニック演算子が引数に含まれている可能性がある場合は、%QKLOWCAS を使用します。

%KLOWCASE は、引数に引用符が含まれる場合でも、引用符なしの結果を返します。%%QKLOWCAS は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します:

& % ' " () + - * / < > = ~ ^ ~ ; , blank AND OR NOT EQ NE LE LT GE GT IN

注: 自動呼び出しマクロは SAS ライブラリに含まれます。このライブラリはサイトにインストールされていないか、またはサイト固有のバージョンの可能性があります。このマクロにアクセスできない場合、またはライブラリがサイト固有のバージョンかどうか確認する場合は、オンサイトの SAS サポート担当者にお問い合わせください。

%KTRIM 自動呼び出しマクロ

末尾の空白を取り除きます。

カテゴリ: DBCS
要件 MAUTOSOURCE システムオプション
参照項目: “%QKTRIM 自動呼び出しマクロ”

構文

%KTRIM (*text* | *text expression*)

必須引数

text
テキストまたはテキスト式。

詳細

KTRIM マクロと QKTRIM マクロは、末尾の空白を取り除きます。引数が特殊文字やニーモニック演算子を含む場合は、%QKTRIM を使用します。

QKTRIMS は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します。

& % ' " () + - * / < > = ~ ? ~ ; , # blank AND OR NOT EQ NE LE LT GE GT IN

自動呼び出しマクロは SAS ライブラリに含まれます。このライブラリはサイトにインストールされていないか、またはサイト固有のバージョンの可能性があります。このマクロにアクセスできない場合、またはライブラリがサイト固有のバージョンかどうか確認する場合は、オンサイトの SAS サポート担当者にお問い合わせください。

%QKTRIM 自動呼び出しマクロ

末尾の空白を取り除きます。

カテゴリ: DBCS
要件 MAUTOSOURCE システムオプション
参照項目: “%KTRIM 自動呼び出しマクロ”

構文

%QKTRIM (*text* | *text expression*)

必須引数

text

テキストまたはテキスト式。

詳細

KTRIM マクロと QKTRIM マクロは、末尾の空白を取り除きます。引数が特殊文字やニーモニック演算子を含む場合は、%QKTRIM を使用します。

QKTRIMS は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します。

& % ' " () + - * / < > = ~ ? ~ ; , # blank AND OR NOT EQ NE LE LT GE GT IN

自動呼び出しマクロは SAS ライブラリに含まれます。このライブラリはサイトにインストールされていないか、またはサイト固有のバージョンの可能性があります。このマクロにアクセスできない場合、またはライブラリがサイト固有のバージョンかどうか確認する場合は、オンサイトの SAS サポート担当者にお問い合わせください。

%KVERIFY 自動呼び出しマクロ

式に固有の最初の文字の位置を返します。

カテゴリ: DBCS

要件 MAUTOSOURCE システムオプション

構文

%KVERIFY (source, excerpt)

構文

source

excerpt に存在しない文字を確認する、テキストまたはテキスト式です。

excerpt

%KVERIFY で source の確認に使用される文字のセットを定義する、テキストまたはテキスト式です。

詳細

%KVERIFY は、excerpt に存在しない source 内の最初の文字の位置を返します。source 内のすべての文字が excerpt に存在する場合、%KVERIFY は 0 を返します。

自動呼び出しマクロは SAS ライブラリに含まれます。このライブラリはサイトにインストールされていないか、またはサイト固有のバージョンの可能性があります。このマクロにアクセスできない場合、またはライブラリがサイト固有のバージョンかどうか確認する場合は、オンサイトの SAS サポート担当者にお問い合わせください。

3 部

各国語サポート関連のデータセットオプション

8 章

各国語サポート関連のデータセットオプションのディクショナリ 83

8

各国語サポート関連のデータセットオプションのディクショナリ

カテゴリ別のデータセットオプション	83
ディクショナリ	84
ENCODING= データセットオプション	84
OUTREP= データセットオプション	87

カテゴリ別のデータセットオプション

各国語サポートは、選択したデータセットオプションのデータセットコントロールカテゴリに影響を与えます。次の表に、各データセットオプションの概要を示します。詳細については、各データセットオプションのディクショナリエントリを参照してください。

カテゴリ	言語要素	説明
データセットコントロール	ENCODING= データセットオプション (p. 84)	SAS データセットの読み取りまたは書き込みに使用するエンコーディングを手動で指定します。
	OUTREP= データセットオプション (p. 87)	出力 SAS データセットのデータ表記を指定します。

ディクショナリ

ENCODING= データセットオプション

SAS データセットの読み取りまたは書き込みに使用するエンコーディングを手動で指定します。

該当要素: DATA ステップおよび PROC ステップ

カテゴリ: データセットコントロール

構文

ENCODING= ANY | ASCIIANY | EBCDICANY | SESSION | *encoding-value*

構文の説明

ANY

トランスコーディングを行わないように指定します。

注 ANY は BINARY の指定と同等です。データはバイナリであるため、実際のエンコーディングには意味がありません。

ASCIIANY

混在したエンコーディングが ASCII エンコーディングの場合にトランスコーディングを行わないように指定します。

EBCDICANY

混在したエンコーディングが EBCDIC エンコーディングの場合にトランスコーディングを行わないように指定します。

SESSION

現在の計算サーバーセッションのロケールを指定します。

セッション値は、COPY プロシジャまたは COPY ステートメントの **OVERRIDE=** オプションでサポートされています。詳細については、[OVERRIDE=](#)を参照してください

encoding-value

エンコーディング値を指定します。

参照項目: [3 章, “各国語サポート関連のエンコーディング”](#) (11 ページ)

詳細

ENCODING=の値は、SAS データセットに現在のセッションエンコーディングとは異なるエンコーディングがあることを示します。データは、データセットから読み取られるときに、指定したエンコーディングからセッションエンコーディングにトランスコードされます。また、データセットに書き込まれるときに、セッションエンコーディングから指定したエンコーディングにトランスコードされます。

入力処理

デフォルトでは、入力処理のエンコーディングは次のように決まります。

- セッションエンコーディングとファイルに指定されたエンコーディングが異なる場合、データはセッションエンコーディングにトランスコードされます。
- ファイルにエンコーディングが指定されておらず、ファイルのデータ表記が現在のセッションのエンコーディングとは異なる場合、データは現在のセッションのエンコーディングにトランスコードされます。

出力処理

デフォルトでは、出力処理のエンコーディングは次のように決まります。

- データは現在のセッションのエンコーディングを使用してファイルに書き込まれます。ただし、OUTREP=データセットオプション、LIBNAME ステートメントの OUTENCODING=オプション、または ENCODING=データセットオプションを使用して別の出力表記が指定された場合を除きます。
- 既存のファイルが新しいファイルで置き換えられる場合、新しいファイルは既存のファイルのエンコーディングを継承します。
- 既存のファイルが、別の動作環境で作成されたか、エンコーディングが指定されていない新しいファイルで置き換えられる場合、新しいファイルは現在のセッションのエンコーディングを使用します。

注: 文字メタデータとデータ出力は、データセットの作成時に使用されたエンコーディングと異なるエンコーディングを指定すると文字化けします。この例では、出力するデータセットは内部的に ASCII としてエンコードされます。ただし、データセットオプションでは EBCDIC エンコーディングが指定されています。SAS は、データを EBCDIC から ASCII にトランスコードしようと試みますが、データはすでに ASCII です。その結果、データが文字化けします。

```
data a;
  x=1;
  abc='abc';
run;
proc print data=a (encoding="ebcdic");
run;
```

注: 次の ENCODING=の値は無効です。

- UCS2
- UCS4
- UTF16

■ UTF32

比較

- セッションエンコーディングは、ENCODING=システムオプションまたは LOCALE=システムオプションを使用して指定されます。それぞれの動作環境にはデフォルトのエンコーディングがあります。
- LIBNAME ステートメントの INENCODING=オプション(入力ファイル用)および OUTENCODING=オプション(出力ファイル用)を使用して SAS ライブラリ用のエンコーディングを指定できます。LIBNAME ステートメントオプションと ENCODING=データセットオプションを両方とも指定すると、データセットオプションが使用されます。

例

例 1: エンコーディングが混在し、トランスコーディングが抑制された SAS データセットの作成

データセットオプション ENCODING=ANY を指定すると、混在したエンコーディングが含まれる SAS データセットを作成し、入力処理または出力処理でトランスコーディングを抑制できます。

この例では、新しいデータセット MYFILES.MIXED に、Latin1 エンコーディングを使用するデータと Latin2 エンコーディングを使用するデータが含まれます。データセットの処理時に、トランスコーディングは行われません。たとえば、Latin1 セッションエンコーディングの正しい Latin1 文字と Latin2 セッションエンコーディングの正しい Latin2 文字が表示されます。

```
libname myfiles 'SAS data-library';
data myfiles.mixed (encoding=any);
  set work.latin1;
  set work.latin2;
run;
```

例 2: 特定のエンコーディングを使用した SAS データセットの作成

出力処理では、現在のセッションエンコーディングを手動で指定できます。このアクションは、通常のファイルアクセスに別のセッションエンコーディングを使用する場合などに必要になることがあります。

たとえば、現在のセッションエンコーディングが Wlatin1 の場合、ENCODING=WLATIN2 を指定してエンコーディング Wlatin2 を使用するデータセットを作成できます。次のステートメントでは、セッションエンコーディングのかわりに Wlatin2 エンコーディングを使用して新しいデータセットにデータを書き込むように指示しています。エンコーディングはまた、ファイルのディスクリプター部分にも指定されます。

```
libname myfiles 'SAS data-library';
data myfiles.difencoding (encoding=wlatin2);
.
.
.
run;
```

例 3: 入力処理のエンコーディングのオーバーライド

入力処理では、ファイル内に指定されたエンコーディングをオーバーライドして別のエンコーディングを指定できます。

この例では、現在のセッションエンコーディングは EBCDIC-870 ですが、ファイルのディスクリプター情報にはエンコーディング値 EBCDIC-1047 が含まれています。ENCODING=EBCDIC-870 と指定すると、データのトランスコーディングは行われずに、データは EBCDIC-870 エンコーディングを使用して表示されます。

```
proc print data=myfiles.mixed (encoding=ebcdic870);
run;
```

関連項目

- 概念の説明: 3 章, “各国語サポート関連のエンコーディング” (11 ページ)

ステートメントとコマンドのオプション:

- “ENCODING=オプション” (1021 ページ)
- “INENCODING=, OUTENCODING=オプション” (1024 ページ)

システムオプション:

- “ENCODING システムオプション” (968 ページ)
- “LOCALE システムオプション” (973 ページ)

OUTREP= データセットオプション

出力 SAS データセットのデータ表記を指定します。

該当要素: DATA ステップおよび PROC ステップ

カテゴリ: データセットコントロール

参照項目: “OUTREP= データセットオプション” (SAS データセットオプション: リファレンス).

構文

OUTREP= *format*

4 部

各国語サポート関連の出力形式

9 章	各国語サポート関連の出力形式の概要	91
10 章	各国語サポート関連の出力形式のディクショナリ	107

各国語サポート関連の出力形式の概要

国際化対応の日付、日時出力形式	91
通貨表現	92
通貨の概要	92
米国ドル	92
ユーロ	93
カスタマイズした通貨表現	93
各国語の通貨記号/国際標準の通貨コードを使用した、地域化された通貨表現	94
各国語の通貨記号/国際標準の通貨コードを使用した、独自の通貨表現	96
例: 各国語の通貨記号および国際標準の通貨コードを使用した出力形式での通貨の表現	99
日付と時間のデフォルト幅の例外	100
統一パターンを使用する NL 出力形式	102

国際化対応の日付、日時出力形式

SAS では、最も一般的に使用される英語の日付出力形式に相当する、国際化対応出力形式がサポートされています。いずれの場合でも、出力形式は対応する英語の出力形式と同様に機能します。最大幅、最小幅、デフォルトの幅のみが異なります。

表 9.1 国際化対応の日付、日時出力形式

英語出力形式	国際化対応出力形式	最小	最大	デフォルト
DATE.	NLDATE.	10	200	20
DATETIME.	NLDATM.	10	200	30
DOWNAME.	NLDATEWN.	4	200	9

英語出力形式	国際化対応出力形式	最小	最大	デフォルト
MONNAME.	NLDATEMN.	4	200	9
MONYY.	NLDATEYM.	6	200	10
WEEKDATX.	NLDATEW.	10	200	29
WEEKDAY.	NLDATEWN.	4	200	9
WORDDATX.	NLDATE.	10	200	20

通貨表現

通貨の概要

通貨は、その国特有の交換手段です。SAS では、通貨の書き込みと読み込みに使用する出力形式と入力形式を提供します。

米国ドル

DOLLARw.d 出力形式と入力形式が、最初に、米国通貨の読み込みと書き込みを行うために導入されました。DOLLARw.d では次の記号を使用します。

- 米国通貨の先頭に通貨記号としてドル記号(\$)を使用します。
- カンマ(,)を桁区切り文字として、ピリオド(.)を小数点区切り文字として使用します。

例:

\$12,345.00

DOLLARXw.d はまた、通貨の先頭にドル記号(\$)を付け、桁区切り文字としてピリオド(.)、小数点区切り文字としてカンマ(,)を使用して書き込みます。ピリオドとカンマが逆の通貨表現は、ヨーロッパの多くの国で使用される規則です。

例:

\$12.345,00

DOLLAR 出力形式および入力形式には次の制限があります。

- すべての通貨記号をサポートしません。

- ヨーロッパ諸国には、ピリオドとカンマが逆の通貨表現を使用しない国があります。
- 通貨記号の見え方はコンピューターによって異なります。

ユーロ

EUROW.d 出力形式および入力形式は、1999 年に設立された欧州通貨同盟(EMU)によって制定されたユーロ通貨をサポートするために導入されました。EUROW.d

例:

```
options locale=English_UnitedKingdom;
data _null_;
x=12345;
put x euro10.2;
run;
```

出力:

E12.345,00

EURO 出力形式はロケール依存ではありません。カンマ(,)を桁区切り文字として、ピリオド(.)を小数点区切り文字として使用します。

カスタマイズした通貨表現

カスタマイズした通貨表現を作成するには、FORMAT プロシジャが使用できます。次の例では、オーストラリアドル、スイスフランおよびイギリスポンドに固有の出力形式を作成します。FORMAT プロシジャの詳細については、*Base SAS プロシジャガイド*を参照してください。

例のコード 9.1 通貨表現をカスタマイズする SAS コード

```
proc format;

  picture aud low-<0='0,000,000,009.00'
    (prefix='-AU$' mult=100)
    0-high='0,000,00,009.00 '
    (prefix='AU$' mult=100);

  picture sfr low-<0='0,000,000,009.00'
    (prefix='-SFr.' mult=100)
    0-high='0,000,00,009.00 '
    (prefix='-SFr.' mult=100);

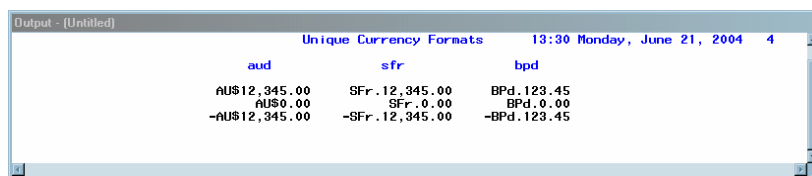
  picture bpd low-<0='0,000,000,009.00'
    (prefix='-BPd.' mult=100)
    0-high='0,000,00,009.00 '
    (prefix='BPd.' mult=100);
```

```

run;
data currency;
input aud sfr bpd 12.2;
datalines;
12345 12345 12345
0 0 0
-12345 -12345 -12345
;

proc print data=currency noobs;
var aud sfr bpd;
format aud aud. sfr sfr. bpd bpd.;
title 'Unique Currency Formats';
run;

```



Unique Currency Formats		
aud	sfr	bpd
AUS\$12,345.00	SFr.12,345.00	BPd.123.45
AUS\$0.00	SFr.0.00	BPd.0.00
-AUS\$12,345.00	-SFr.12,345.00	-BPd.123.45

通貨表現をカスタマイズすると柔軟性は向上しますが、プログラムでの対応が必要です。

各国語の通貨記号/国際標準の通貨コードを使用した、地域化された通貨表現

NLMNYw.d および NLMNYIw.d の出力形式と入力形式は、地域化された通貨を次の2つの形式で表現するために導入されました。

地域化された各国語通貨表現

ロケールの地域規則と慣習が反映されます。各国語のフォーマットは、NLMNYw.d 出力形式および入力形式を使用して指定します。NLMNYw.d 出力形式および入力形式を使用する場合、さらに LOCALE=オプションも使用してロケールを指定する必要があります。

例:

```

options locale=english_UnitedStates;
data _null_;
x=12345;
put x nlmny15.2;
run;

```

出力:

\$12,345.00

次に、一部の通貨の各国語表現を示します。

表 9.2 地域化された各国語通貨表現

LOCALE=	通貨	各国語表現
English_UnitedStates	米国ドル	\$12,345.00
French_Canada	カナダドル	12 345,00 \$
French_France	ユーロ	12 € 345,00
French_Switzerland	スイスフラン	12 345,00 CHF
German_Germany	ユーロ	12.345,00 €
German_Luxembourg	ユーロ	12.345,00 €
Spanish_Spain	ユーロ	12.345,00 €
Spanish_Venezuela	ベネズエラボリバル	Bs.S12.345,00

地域化された表示には、通貨表示に関する各国固有の地域規則が反映されます。たとえば、これらの EMU 諸国の中には、同じユーロ通貨を使用していても通貨の表記が異なる場合があります。French_France は、桁区切り文字は使用せず、小数点区切り文字としてカンマを使用しますが、German_Germany および Spanish_Spain は桁区切り文字としてピリオドを、小数点区切り文字としてカンマを使用します。

国際標準の通貨コードを使用した、地域化された通貨表現

ISO 標準 4217 に準拠します。国際標準の通貨コードを使用した形式は、NLMNYIw.d 出力形式および入力形式を使用して指定します。国際通貨は、世界の通貨の比較を示すために一般的に使用されます。例としては、航空券、貿易、株式市場の価格設定があります。NLMNYIw.d 出力形式および入力形式を使用する場合、さらに LOCALE=オプションも使用してロケールを指定する必要があります。"International"を示す文字"I"が、出力形式名および入力形式名に付加されています。

例:

```
options locale=english_UnitedStates;
data _null_;
x=12345; put x nlmnyi15.2;
run;
```

Output:
USD12,345.00

国際標準の通貨コードを使用した通貨表現の一部を次に示します。

表 9.3 国際標準の通貨コードを使用した、ロケール別の通貨表現(ISO 標準 4217)

LOCALE=	通貨	国際標準の通貨コードを使用した表現
English_UnitedStates	米国ドル	USD12,345.00
French_Canada	カナダドル	12 345,00 CAD
French_France	ユーロ	12 345,00 EUR
French_Luxembourg	ユーロ	12.345,00 EUR
German_Germany	ユーロ	12.345,00 EUR
German_Switzerland	スイスフラン	CHF 12'345.00
Spanish_Spain	ユーロ	12.345,00 EUR
Spanish_Venezuela	ベネズエラボリバル	VES12.345,00

国際標準の通貨コードを使用した表示には、通貨表示に関する各国固有の地域規則も反映されます。たとえば、すべてのロケールで桁区切り文字にカンマ、小数点区切り文字にピリオドが使用されるわけではありません。ISO 通貨コードが使用される場合、その位置はロケールにより異なります。EMU 諸国が通貨の後に通貨コードを付けるのに対し、English_UnitedStates、German_Switzerland および Spanish_Venezuela では通貨の前に ISO コードが付けられます。

ISO 標準 4217 通貨コードの全リストについては、<http://www.xe.com/iso4217.php> を参照してください。

各国語の通貨記号/国際標準の通貨コードを使用した地域化された通貨表現を使用する場合、LOCALE=システムオプションの値に依存することが最大の制限となります。

各国語の通貨記号/国際標準の通貨コードを使用した、独自の通貨表現

NLMNYISOW.d および NLMNYIISOW.d の出力形式と入力形式は、LOCALE=オプションを併用しなくても各通貨を独自に表現できるように導入されました。各通貨は、ISO 標準 4217 通貨コードで示されます。

NLMNY および NLMNYI 形式は、データ内で見つかった数値を、現在の SAS LOCALE オプションと一致する通貨記号を使用して表示する結果を返します。SAS は、値のある通貨から別の通貨の換算しません。LOCALE が変更されると、同じ番号が異なる通貨記号で表示されます。変更された通貨記号は、値が変更されたことを保証するものではありません。

独自の各国語通貨表現

固有の ISO 通貨コードで指定します。各国語のフォーマットは、NLNMLISOw.d 出力形式および入力形式を使用して指定します。次の例では、USD が米国ドルの ISO 通貨コードです。

注: NLNMLISOw.d 出力形式および入力形式を使用する場合は、LOCALE=オプションは省略可能です。

例:

```
data _null_;
  put x nlmnlusd15.2;
run;
```

出力:

US\$12,345.00

次に、一部の通貨の各国語表現を示します。

この例では、ロケールは fr_FR として指定されます。

```
data _null_;
  x=12345;
  put x nlmnleur15.2;
run;
```

出力:

€12,345.00

表 9.4 ISO 通貨コードを使用した、独自の通貨表現

ISO 通貨コード	通貨	各国語表現
USD	米国ドル	US\$12,345.00
CAD	カナダドル	CA\$12,345.00
EUR	ユーロ	€12,345.00
CHF	スイスフラン	SFr.12,345.00
EUR	ユーロ	€12,345.00
EUR	ユーロ	€12,345.00
EUR	ユーロ	€12,345.00

通貨記号または通貨コードは、多くの通貨で先頭に付けられます。また、桁区切り文字にはカンマ、小数点区切り文字にはピリオドが使用されます。ローカル通貨の通貨記号が現在の SAS セッションのエンコーディングでサポートされていない場合、NLNMLxxxw.d 出力形式では、3 文字の ISO 通貨コードを使用して値をフォーマットします。

国際標準の通貨コードを使用した、独自の通貨表現

固有の ISO 通貨コードで指定します。国際化対応出力形式は、NLMNISOw.d 出力形式および入力形式を使用して指定します。国際標準の通貨コードを使用した形式は一般に、世界各国の通貨の対比を示すために使用されます。たとえば、航空券、貿易、株価などで使用されます。たとえば、航空券、貿易、株価などで使用されます。"International"を示す文字"I"が、出力形式名および入力形式名に付加されています。次の例では、USD が米国ドルの ISO 通貨コードです。

注: NLMNISOw.d 出力形式および入力形式を使用する場合、LOCALE=オプションを使用してロケールを指定しません。

```
data _null_;
  x=12345;
  put x nlmniusd15.2;
run;
```

出力:

USD12,345.00

国際標準の通貨コードを使用した通貨表現の一部を次に示します。

表 9.5 国際標準の通貨コードを使用した、ISO 通貨コード別の通貨表現

ISO 通貨コード	通貨	国際標準の通貨コードを使用した表現
USD	米国ドル	USD12,345.00
CAD	カナダドル	CAD12,345.00
EUR	ユーロ	EUR12,345.00
CHF	スイスフラン	CHF12,345.00
EUR	ユーロ	EUR12,345.00
EUR	ユーロ	EUR12,345.00
EUR	ユーロ	EUR12,345.00

国際標準の通貨コードを使用した表示では、該当する ISO コードが通貨の先頭に付けられます。また、桁区切り文字にはカンマ、小数点区切り文字にはピリオドが使用されます。

例: 各国語の通貨記号および国際標準の通貨コードを使用した出力形式での通貨の表現

この SAS プログラムでは、選択したアジア太平洋諸国について米国ドルに対する換算レートを使用します。出力では、各国の通貨が各国語の通貨記号および国際標準の通貨コードを使用した出力形式を使用して表記されます。

例のコード 9.2 各国語の通貨記号および国際標準の通貨コードを使用した出力形式で通貨を表記する SAS コード

```
data curr;
input ex_date mmddyy. usd aud hkd jpy sgd 12.2;
datalines;
061704 1.00000 1.45349 7.79930 110.110 1.71900 1
;
proc print data=curr noobs label;
var ex_date usd aud hkd jpy sgd;
format ex_date mmddyy. usd nlmnlusd15.2 aud nlmnlaud15.2 hkd nlmnlhkd15.2
jpy nlmnljpy15.2 sgd nlmnlsd15.2; 2
label ex_date='Date' usd="US" aud='Australia' hkd='Hong Kong' jpy='Japan'
sgd='Singapore';
title 'Exchange Rates for Selected Asian-Pacific Countries (Localized Currency Codes)';
proc print data=curr noobs label;
var ex_date usd aud hkd jpy sgd;
format ex_date mmddyy. usd nlmniusd15.2 aud nlmniaud15.2 hkd
nlmnhkd15.2 jpy nlmnijpy15.2 sgd nlmnisgd15.2; 3
label ex_date='Date' usd="US" aud='Australia' hkd='Hong Kong' jpy='Japan'
sgd='Singapore';
title 'Exchange Rates for Selected Asian-Pacific Countries (International Currency Codes)';
run;
```

- 1 2004 年 6 月 17 日の時点の各換算レートが、データとして SAS プログラムに指定されます。
- 2 これらの NLMNLISO 出力形式は、INPUT ステートメントで指定された数値データ項目にそれぞれ適用されます。これらの出力形式は、該当する各国語の出力形式で通貨を表示します。
- 3 これらの NLMNIISO 出力形式は、INPUT ステートメントで指定された数値データ項目にそれぞれ適用されます。これらの出力形式は、該当する国際標準の通貨コードを使用した出力形式で通貨を表示します。

図 9.1 各国語の通貨記号および国際標準の通貨コードを使用した出力形式の出力

Exchange Rates for Selected Asian-Pacific Countries (Localized Currency Codes)					
Date	US	Australia	Hong Kong	Japan	Singapore
06/17/04	US\$1.00	AUS\$1.45	HK\$7.80	JPY110.11	SG\$1.72

Exchange Rates for Selected Asian-Pacific Countries (International Currency Codes)					
Date	US	Australia	Hong Kong	Japan	Singapore
06/17/04	USD1.00	AUD1.45	HKD7.80	JPY110.11	SGD1.72

日付と時間のデフォルト幅の例外

ロケールとエンコーディングの組み合わせによっては、*月名*と*週名*の長さが、一部の出力形式のデフォルト幅を超える場合があります。これらの組み合わせに対してはデフォルト幅を使用しないようにします。影響を受けるロケール、エンコーディングおよび出力形式については、次の表を参照してください。正確な出力のためには推奨される幅を使用する必要があります。

次の表に、地域化された出力がデフォルトの幅を超える可能性がある、エンコーディングとロケールの組み合わせを示します。

エンコーディング	ロケール
WLATIN1	fr_FR, it_IT, de_DE, es_ES, en_US, en_GB
WLATIN2	cs_CS, hu_HU, pl_PL
WCYRILLIC	RU_RU
EUC_CN	en_US & zh_CN
SHIFT-JIS	en_US ja_JP4-6
EUC-KR	en_US ko_KR
BIG5	en_us zt_TW zt_HK
UTF-8	示されたすべてのロケール

次の表に、地域化された出力がデフォルトの幅を超える可能性があるエンコーディングとロケールの組み合わせに対する推奨される幅を示します。ここに指名された出力形式には、推奨される幅を使用するようにします。

エンコーディング	ロケール	出力形式	デフォルトの幅	推奨される幅(>=)
WLATIN1	EN_GB	nldatmw	30	32
		nltimap	10	16
	EN_US	nltimap	10	11
WLATIN2	CS_CS	nltimap	10	11
EUC-CN	EN_US	nltimap	10	11

エンコーディング	ロケール	出力形式	デフォルトの幅	推奨される幅(>=)
	ZH_CN	nldatmw	30	38
		nldatmwz	40	44
		nltimap	10	16
SHIFT-JIS	EN_US	nltimap	10	11
	JA_JP	nldatmw	30	34
		nltimap	10	16
EUC-KR	EN_US	nltimap	10	11
	KO_KR	nldatmw	30	37
		nldatmwz	40	43
		nltimap	10	17
BIG5	EN_US	nltimap	10	11
	ZT_HK	nltimap	10	11
	ZT_TW	nltimap	10	11
UTF-8	CS_CS	nltimap	10	11
	EN_GB	nldatmw	30	32
		nltimap	10	16
	EN_US	nltimap	10	11
	JA_JP	nldatmw	30	43
		nldateyq	16	20
		nldatmyq	16	20
		nldatmwz	40	49
		nltimap	10	21
	KO_KR	nldatew	20	21
		nldatmw	30	46
		nldatmwz	40	52
		nltimap	10	22

エンコーディング	ロケール	出力形式	デフォルトの幅	推奨される幅(>=)
	RU_RU	nldatew	20	21
		nldatmw	30	32
	ZH_CN	nldatew	20	25
		nldatmw	30	49
		nldatmwz	40	55
		nltimap	10	21
	ZT_HK	nldatew	20	25
		nldatmw	30	49
		nldatmwz	40	55
		nltimap	10	21
	ZT_TW	nldatew	20	25
		nldatmw	30	49
		nldatmwz	40	55
		nltimap	10	21

統一パターンを使用する NL 出力形式

統一パターンを使用する NL 形式は、顧客がさまざまなロケールでデータを出力するのに役立ちます。出力には、標準、長、中、短のパターンを指定できます。フォーマットされた値が NL 出力形式に指定された長さに収まらない場合は、より短いパターンが使用され、その結果、表示が異なります。いくつかの NL 出力形式には統一したシリーズが提供されており、指定された幅内ですべてのデータが同じスタイルで表示されるように適切なパターンが選択されます。統一パターンを使用する NL 出力形式のリストについては、[表 9.7](#) を参照してください。たとえば、次のように指定します。

NLDATE

現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。

NLDATEL

SAS 日付値を、現在の SAS ロケールに適した月、日付、年の形式を長い統一パターンの日付として書き込みます。

NLDATEM

SAS 日付値を、現在の SAS ロケールに適した中程度の統一パターンで日付として書き込みます。

NLDATES

SAS 日付値を、現在の SAS ロケールに適した短い統一パターンで日付として書き込みます。

この例では、NLDATE 出力形式により、DATA LINES で指定された日付の出力が書き込まれます。NLDATE 出力形式では幅 17 が指定されます。

```
options locale = English_UnitedStates; 1
data dval; 2
infile datalines;
input date mmddyy8.;
datalines;
01/25/02
02/25/02
03/24/02
04/25/02
05/25/02
06/24/02
07/25/02
08/25/02
09/24/02
10/25/02
11/25/02
12/24/02
;
run;
proc print data = dval; 3
format date nldate17.;
run;
```

- 1 English_United States ロケールを指定します。
- 2 データセット dval の場合、DATA LINES の内容は、幅 8 の入力形式 MMDDYY を使用して変数 *date* として入力されます。
- 3 幅 17 の NLDATE 出力形式を使用して dval の内容を出力します。

これらのステートメントは、次の結果を生成します。9 番目の月は 9 月なので、9 番目の Obs は 2023 年 9 月 24 日になります。固定幅 17 では、9 番目のオブザベーション値の日付値の出力形式は他のオブザベーション値とは異なります。これは、出力形式の長さが月名、9 月、および日付値のその他の要素を表示するのに十分な長さではないために発生します。NLDATE 出力形式では、すべての要素を表示できるように短いパターンが選択されます。

The SAS System	
Obs	date
1	January 25, 2023
2	February 25, 2023
3	March 24, 2023
4	April 25, 2023
5	May 25, 2023
6	June 24, 2023
7	July 25, 2023
8	August 25, 2023
9	09/24/2023
10	October 25, 2023
11	November 25, 2023
12	December 24, 2023

NL 統一パターンシリーズの出力形式は、これらのシナリオを支援するように設計されています。統一パターンの出力形式名は、長いパターンを使用する L、中程度の長さのパターンを使用する M、または短いパターンを使用する S で終わります。統一された出力形式により、一貫したレイアウトが保証されます。最も適切なパターンを決定する前に、考えられるすべての出力長を事前計算して予測します。

表 9.6 は、NLDATE が NLDATEL、NLDATEM、および NLDATES に置き換えられる前のコードの結果を示しています。次の PROC PRINT コードは、NLDATE17 の代わりに NLDATEL 出力形式を使用します。NLDATEL を使用するため、幅(width)は必要ありません。9 番目のオブザベーションの出力は 2023 年 9 月 24 日であり、幅は指定されていません。

```
proc print data = dval;
  format date nldatel;
run;
```

表 9.6 NLDATEL、NLDATEM、NLDATES 出力

NLDATEL	NLDATEM	NLDATES
---------	---------	---------

The SAS System		The SAS System		The SAS System	
Obs	date	Obs	date	Obs	date
1	January 25, 2023	1	Jan 25, 2023	1	01/25/2023
2	February 25, 2023	2	Feb 25, 2023	2	02/25/2023
3	March 24, 2023	3	Mar 24, 2023	3	03/24/2023
4	April 25, 2023	4	Apr 25, 2023	4	04/25/2023
5	May 25, 2023	5	May 25, 2023	5	05/25/2023
6	June 24, 2023	6	Jun 24, 2023	6	06/24/2023
7	July 25, 2023	7	Jul 25, 2023	7	07/25/2023
8	August 25, 2023	8	Aug 25, 2023	8	08/25/2023
9	September 24, 2023	9	Sep 24, 2023	9	09/24/2023
10	October 25, 2023	10	Oct 25, 2023	10	10/25/2023
11	November 25, 2023	11	Nov 25, 2023	11	11/25/2023
12	December 24, 2023	12	Dec 24, 2023	12	12/24/2023

統一パターンのバージョンがある NL 出力形式のリストを次に示します。

表 9.7 統一パターンのバージョンを持つ出力形式

出力形式	説明	L(長幅出力形式)	M(中幅出力形式)	S(短幅出力形式)
NLDATE	現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。	NLDATEL	NLDATEM	NLDATES
NLDATEMD	SAS の日付値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。	NLDATEMDL	NLDATEMDM	NLDATEMDS
NLDATEYM	SAS の日付値を、現在の SAS ロケールに適した年	NLDATEYML	NLDATEYMM	NLDATEYMS

出力形式	説明	L(長幅出力形式)	M(中幅出力形式)	S(短幅出力形式)
	と月の名前として書き込みます。			
NLDATEYQ	現在の SAS ロケールに適した年と四半期として SAS 日付値を書き込みます。	NLDATEYQL	NLDATEYQM	NLDATEYQS
NLDATM	現在の SAS ロケールに適した日時として SAS 日時値を書き込みます。	NLDATML (230 ページ)	NLDATMM	NLDATMS
NLDATMMD	SAS 日時値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。	NLDATMMDL	NLDATMMDM	NLDATMMDS
NLDATMYM	現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。	NLDATMYML	NLDATMYMM	NLDATMYMS
NLDATMYQ	現在の SAS ロケールに適した四半期と年として SAS 日時値を書き込みます。	NLDATMYQL	NLDATMYQM	NLDATMYQS

10

各国語サポート関連の出力形式の ディクショナリ

各国語サポート関連の出力形式のカテゴリ	112
ディクショナリ	143
B8601DXw. 出力形式	143
B8601LXw. 出力形式	145
B8601TXw. 出力形式	147
BESTDOTXw. 出力形式	149
\$BIDIw. 出力形式	150
\$CPTDWw. 出力形式	152
\$CPTWDw. 出力形式	153
DTWEEKVw. 出力形式	154
\$EBCDICw. 出力形式	156
E8601DXw. 出力形式	157
E8601LXw. 出力形式	159
E8601TXw. 出力形式	161
EUROW.d 出力形式	163
EUROW.d 出力形式	166
HDATEw. 出力形式	168
HEBDATEw. 出力形式	170
JNENGOW.d 出力形式	172
JNENGOTw. 出力形式	173
JNENGOTW w. 出力形式	174
JNENGOWw.d 出力形式	175
\$KANJIw. 出力形式	177
\$KANJIIXw. 出力形式	178
\$LOGVSw. 出力形式	179
\$LOGVSRw. 出力形式	181
MINGUOW. 出力形式	182
NENGOW. 出力形式	184
NLBESTw. 出力形式	186
NLDw.p 出力形式	188
NLDATEw. 出力形式	190
NLDATECPw.p 出力形式	192
NLDATELw. 出力形式	195

NLDATEMw. 出力形式	196
NLDATEMDw. 出力形式	198
NLDATEMDLw. 出力形式	199
NLDATEMDMw. 出力形式	200
NLDATEMDSw. 出力形式	201
NLDATEMNw. 出力形式	202
NLDATESw. 出力形式	204
NLDATEWw. 出力形式	205
NLDATEWNw. 出力形式	207
NLDATEYw. 出力形式	209
NLDATEYMLw. 出力形式	211
NLDATEYMMw. 出力形式	212
NLDATEYMSw. 出力形式	213
NLDATEYQw. 出力形式	214
NLDATEYQLw. 出力形式	216
NLDATEYQMw. 出力形式	217
NLDATEYQSw. 出力形式	218
NLDATEYRw. 出力形式	219
NLDATEYWw. 出力形式	220
NLDATMw. 出力形式	222
NLDATMAPw. 出力形式	223
NLDATMCPw.p 出力形式	225
NLDATMDTw. 出力形式	229
NLDATMLw. 出力形式	230
NLDATMMw. 出力形式	231
NLDATMMDw. 出力形式	232
NLDATMMDLw. 出力形式	233
NLDATMMDMw. 出力形式	235
NLDATMMDSw. 出力形式	236
NLDATMMNw. 出力形式	237
NLDATMSw. 出力形式	238
NLDATMTMw. 出力形式	239
NLDATMTZw. 出力形式	241
NLDATMWw. 出力形式	242
NLDATMWNw. 出力形式	243
NLDATMWZw. 出力形式	245
NLDATMYMw. 出力形式	246
NLDATMYMLw. 出力形式	247
NLDATMYMMw. 出力形式	248
NLDATMYMSw. 出力形式	249
NLDATMYQw. 出力形式	250
NLDATMYQLw. 出力形式	252
NLDATMYQMw. 出力形式	253
NLDATMYQSw. 出力形式	254
NLDATMYRw. 出力形式	255
NLDATMYWw. 出力形式	256
NLDATMZw. 出力形式	257
NLMNACIAED w.d 出力形式	258
NLMNACIAUD w.d 出力形式	260
NLMNACIBGN w.d 出力形式	261
NLMNACIBRL w.d 出力形式	263
NLMNACICAD w.d 出力形式	264
NLMNACICHF w.d 出力形式	266
NLMNACICNY w.d 出力形式	267

NLMNACICZK w.d 出力形式	269
NLMNACIDKK w.d 出力形式	270
NLMNACIEEK w.d 出力形式	272
NLMNACIEGPw.d 出力形式	273
NLMNACIEURw.d 出力形式	275
NLMNACIGBPw.d 出力形式	276
NLMNACIHKDw.d 出力形式	278
NLMNACIHRKw.d 出力形式	279
NLMNACIHUFw.d 出力形式	281
NLMNACIIDRw.d 出力形式	282
NLMNACIILSw.d 出力形式	284
NLMNACIINRw.d 出力形式	285
NLMNACIJPYw.d 出力形式	287
NLMNACIKRWw.d 出力形式	288
NLMNACILTLw.d 出力形式	290
NLMNACILVLw.d 出力形式	291
NLMNACIMOPw.d 出力形式	293
NLMNACIMXNw.d 出力形式	294
NLMNACIMYRw.d 出力形式	296
NLMNACINOKw.d 出力形式	297
NLMNACINZDw.d 出力形式	299
NLMNACIPLNw.d 出力形式	300
NLMNACIRUBw.d 出力形式	302
NLMNACISEKw.d 出力形式	303
NLMNACISGDw.d 出力形式	305
NLMNACITHBw.d 出力形式	306
NLMNACITRYw.d 出力形式	308
NLMNACITWDw.d 出力形式	309
NLMNACIUSDw.d 出力形式	311
NLMNACIZARw.d 出力形式	312
NLMNACLAEDx.d 出力形式	314
NLMNACLAUDw.d 出力形式	315
NLMNACLBGNw.d 出力形式	317
NLMNACLBRLw.d 出力形式	318
NLMNACLCADw.d 出力形式	320
NLMNACLCFw.d 出力形式	321
NLMNACLCNYw.d 出力形式	323
NLMNACLCZKw.d 出力形式	324
NLMNACLDKKw.d 出力形式	326
NLMNACLEEKw.d 出力形式	327
NLMNACLEGPw.d 出力形式	329
NLMNACLEURw.d 出力形式	330
NLMNACLGBPw.d 出力形式	332
NLMNACLHKDw.d 出力形式	333
NLMNACLHRKw.d 出力形式	335
NLMNACLHUFw.d 出力形式	336
NLMNACLIDRw.d 出力形式	338
NLMNACLILSw.d 出力形式	339
NLMNACLINRw.d 出力形式	341
NLMNACLJPYw.d 出力形式	342
NLMNACLKRWw.d 出力形式	344
NLMNACLLTLw.d 出力形式	345
NLMNACLLVLw.d 出力形式	347
NLMNACLMOPw.d 出力形式	348

NLMNACLMXNw.d 出力形式	350
NLMNACLMYRw.d 出力形式	351
NLMNACLNOKw.d 出力形式	353
NLMNACLNZDw.d 出力形式	354
NLMNACLPLNw.d 出力形式	356
NLMNACLRUBw.d 出力形式	357
NLMNACLSEKw.d 出力形式	359
NLMNACLSGDw.d 出力形式	360
NLMNACLTHBw.d 出力形式	362
NLMNACLTRYw.d 出力形式	363
NLMNACLTDw.d 出力形式	365
NLMNACUSDw.d 出力形式	366
NLMNACLZARw.d 出力形式	368
NLMNIAEDw.d 出力形式	369
NLMNIAUDw.d 出力形式	371
NLMNIBGNw.d 出力形式	372
NLMNIBRLw.d 出力形式	373
NLMNICADw.d 出力形式	375
NLMNICHFW.d 出力形式	376
NLMNICNYw.d 出力形式	378
NLMNICZKw.d 出力形式	379
NLMNIDKKw.d 出力形式	380
NLMNIEEKw.d 出力形式	382
NLMNIEGPw.d 出力形式	383
NLMNIEURw.d 出力形式	385
NLMNIGBPw.d 出力形式	386
NLMNIHKDw.d 出力形式	388
NLMNIHRKw.d 出力形式	389
NLMNIHUFw.d 出力形式	390
NLMNIIDRw.d 出力形式	392
NLMNIILSw.d 出力形式	393
NLMNIINRw.d 出力形式	395
NLMNIJPYw.d 出力形式	396
NLMNIKRWw.d 出力形式	397
NLMNILTlw.d 出力形式	399
NLMNILLVw.d 出力形式	400
NLMNIMOPw.d 出力形式	402
NLMNIMXNw.d 出力形式	403
NLMNIMYRw.d 出力形式	405
NLMNINOKw.d 出力形式	406
NLMNINZDw.d 出力形式	407
NLMNIPLNw.d 出力形式	409
NLMNIRUBw.d 出力形式	410
NLMNISEKw.d 出力形式	412
NLMNISGDw.d 出力形式	413
NLMNITHBw.d 出力形式	414
NLMNITRYw.d 出力形式	416
NLMNITWDw.d 出力形式	417
NLMNIUSDw.d 出力形式	419
NLMNIZARw.d 出力形式	420
NLMNLAEDx.d 出力形式	422
NLMNLAUDw.d 出力形式	423
NLMNLBGNw.d 出力形式	424
NLMNLBRLw.d 出力形式	426

NLMNLCADw.d 出力形式	427
NLMNLCHFw.d 出力形式	429
NLMNLCNYw.d 出力形式	430
NLMNLCZKw.d 出力形式	431
NLMNLDKKw.d 出力形式	433
NLMNLEEKw.d 出力形式	434
NLMNLEGPw.d 出力形式	436
NLMNLEURw.d 出力形式	437
NLMNLGBPw.d 出力形式	439
NLMNLHKDw.d 出力形式	440
NLMNLHRKw.d 出力形式	442
NLMNLHUFw.d 出力形式	443
NLMNLIDRw.d 出力形式	444
NLMNLILSw.d 出力形式	446
NLMNLINRw.d 出力形式	447
NLMNLJPYw.d 出力形式	449
NLMNLKRWw.d 出力形式	450
NLMNLLTLw.d 出力形式	451
NLMNLLVLw.d 出力形式	453
NLMNLMOPw.d 出力形式	454
NLMNLMXNw.d 出力形式	456
NLMNLMYRw.d 出力形式	457
NLMNLNOKw.d 出力形式	459
NLMNLNZDw.d 出力形式	460
NLMNLPLNw.d 出力形式	462
NLMNLRUBw.d 出力形式	463
NLMNLSEKw.d 出力形式	464
NLMNLSGDw.d 出力形式	466
NLMNLTHBw.d 出力形式	467
NLMNLTRYw.d 出力形式	469
NLMNLTWDw.d 出力形式	470
NLMNLUSDw.d 出力形式	471
NLMNLZARw.d 出力形式	473
NLMNYw.d 出力形式	474
NLMNYACw.d 出力形式	476
NLMNYIw.d 出力形式	478
NLMNYIACw.d 出力形式	480
NLNUMw.d 出力形式	482
NLNUMIw.d 出力形式	484
NLPCTw.d 出力形式	486
NLPCTIw.d 出力形式	488
NLPCTNw.d 出力形式	489
NLPCTPw.d 出力形式	490
NLPVALUEw.d 出力形式	492
NLSTRMONw.d 出力形式	493
NLSTRQTRw.d 出力形式	495
NLSTRWKw.d 出力形式	496
NLTIMAPw. 出力形式	498
NLTIMEw. 出力形式	499
NLTIMELw. 出力形式	501
NLTIMEMw. 出力形式	503
NLTIMESw. 出力形式	504
\$UCS2Bw. 出力形式	506
\$UCS2BEw. 出力形式	507

\$UCS2Lw. 出力形式	509
\$UCS2LEw. 出力形式	510
\$UCS2Xw. 出力形式	512
\$UCS2XEw. 出力形式	513
\$UCS4Bw. 出力形式	515
\$UCS4BEw. 出力形式	516
\$UCS4Lw. 出力形式	518
\$UCS4LEw. 出力形式	520
\$UCS4Xw. 出力形式	521
\$UCS4XEw. 出力形式	523
\$UESCw. 出力形式	524
\$UESCEw. 出力形式	526
\$UNCRw. 出力形式	527
\$UNCREw. 出力形式	528
\$UPARENw. 出力形式	530
\$UPARENEw. 出力形式	531
\$UTF8Xw. 出力形式	532
\$UTF8XEw. 出力形式	533
\$VSLOGw. 出力形式	535
\$VSLOGRw. 出力形式	536
WEEKUw. 出力形式	538
WEEKVw. 出力形式	540
WEEKWw. 出力形式	542
YENw.d 出力形式	544
YYWEEKUw. 出力形式	545
YYWEEKVw. 出力形式	547
YYWEEKWw. 出力形式	549

各国語サポート関連の出力形式のカテゴリ

各国語サポートに関するカテゴリ:

カテゴリ	説明
BIDI テキストの操作	データ変数から双方向のデータ値を書き出すように指示します。 出力形式のリストについては、 BIDI テキスト処理 を参照してください。
CAS	この出力形式を CAS サーバー上で実行するように指示します。 出力形式のリストについては、 CAS を参照してください。
文字	文字変数から文字データ値を書き出すように指示します。 出力形式のリストについては、 文字 を参照してください。

カテゴリ	説明
DBCS	アジア言語で使用する 2 バイト文字セットを変換するように指示します。 出力形式のリストについては、 DBCS を参照してください。
ヘブライ語 テキスト処理	データ変数からヘブライ語のデータを読み込むように指示します。 出力形式のリストについては、 ヘブライ語テキスト処理 を参照してください。
数値	数値変数から数値データ値を書き出すように指示します。 出力形式のリストについては、 数値 を参照してください。

カテゴリ	言語要素	説明
BIDI テキストの 操作	\$BIDIw. 出力形式 (p. 150)	ラテン語の単語と数字の順序を維持しつつ、ヘブライ語とアラビア語の文字の順序を逆にするにより、論理的な順序の文字列と視覚的な順序の文字列を双方向に変換します。
	\$LOGVSw. 出力形式 (p. 179)	左から右に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。
	\$LOGVSRw. 出力形式 (p. 181)	右から左に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。
	\$VSLOGw. 出力形式 (p. 535)	視覚的順序で並ぶ文字列を処理し、その文字列を左から右の論理的順序で書き込みます。
	\$VSLOGRw. 出力形式 (p. 536)	視覚的順序で並ぶ文字列を処理し、その文字列を右から左の論理的順序で書き込みます。
CAS	DTWEEKVw. 出力形式 (p. 154)	V アルゴリズムを使用して、10 進数の週番号と日時番号を書き込みます。
	EUROW.d 出力形式 (p. 163)	先頭にユーロ(E)記号を付け、3 桁ごとにカンマを入れ、小数部分との区切りにはピリオドを使用して数値を出力します。
	EUROXw.d 出力形式 (p. 166)	先頭にユーロ(E)記号を付け、3 桁ごとにピリオドを入れ、小数部分との区切りにはカンマを使用して数値を出力します。
	\$LOGVSw. 出力形式 (p. 179)	左から右に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。
	\$LOGVSRw. 出力形式 (p. 181)	右から左に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。
	NENGOW. 出力形式 (p. 184)	SAS 日付値を日本の日付として e.yyymmdd 形式で書き込みます。
	NLBESTw. 出力形式 (p. 186)	ロケールに基づいて最適な数値表記を書き込みます。

カテゴリ	言語要素	説明
	NLDw.p 出力形式 (p. 188)	値の桁数をできる限り最大にし、同じような大きさの数値に対しては小数点以下の桁数を揃えて出力します。
	NLDATEw. 出力形式 (p. 190)	現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。
	NLDATECPw.p 出力形式 (p. 192)	SAS 日付値をロケール依存のコンパクト間隔日付形式に変換します。
	NLDATELw. 出力形式 (p. 195)	SAS の日付値を、現在の SAS ロケールに適した月、日付、年の形式で日付として書き込みます。
	NLDATEMw. 出力形式 (p. 196)	SAS 日付値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で日付として書き込みます。
	NLDATEMDw. 出力形式 (p. 198)	SAS の日付値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。
	NLDATEMDLw. 出力形式 (p. 199)	現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。
	NLDATEMDMw. 出力形式 (p. 200)	現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。
	NLDATEMDSw. 出力形式 (p. 201)	SAS 日付値を、現在の SAS ロケールに適した短い形式で日付として書き込みます。
	NLDATEMNw. 出力形式 (p. 202)	SAS 日付値を現在の SAS ロケールに適した月の名前として書き込みます。
	NLDATESw. 出力形式 (p. 204)	SAS 日付値を、現在の SAS ロケールに適した日付文字列として書き込みます。
	NLDATEWw. 出力形式 (p. 205)	SAS 日付値を、現在の SAS ロケールに適した日付と曜日として書き込みます。
	NLDATEWNw. 出力形式 (p. 207)	現在の SAS ロケールに適した曜日で SAS 日付値を書き込みます。
	NLDATEYMw. 出力形式 (p. 209)	SAS の日付値を、現在の SAS ロケールに適した年と月の名前として書き込みます。
	NLDATEYMLw. 出力形式 (p. 211)	現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。
	NLDATEYMMw. 出力形式 (p. 212)	短縮形を使用して、SAS 日付値を現在の SAS ロケールに適した月と年として書き込みます。
	NLDATEYMSw. 出力形式 (p. 213)	現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。
	NLDATEYQw. 出力形式 (p. 214)	現在の SAS ロケールに適した年と四半期として SAS 日付値を書き込みます。

カテゴリ	言語要素	説明
	NLDATEYQLw. 出力形式 (p. 216)	短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1～Q4)として、SAS 日付値を書き込みます。
	NLDATEYQMw. 出力形式 (p. 217)	短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1～Q4)として、SAS 日付値を書き込みます。
	NLDATEYQSw. 出力形式 (p. 218)	数字と区切り文字を使用して、現在の SAS ロケールに適した年とその年の四半期値(1～4)として、SAS 日付値を書き込みます。
	NLDATEYRw. 出力形式 (p. 219)	現在の SAS ロケールに適した年として SAS 日付値を書き込みます。
	NLDATEYWw. 出力形式 (p. 220)	現在の SAS ロケールに適した年と週として SAS 日付値を書き込みます。
	NLDATMw. 出力形式 (p. 222)	現在の SAS ロケールに適した日時として SAS 日時値を書き込みます。
	NLDATMAPw. 出力形式 (p. 223)	現在の SAS ロケールに適した午前または午後付きの日時として SAS 日時値を書き込みます。
	NLDATMCPw.p 出力形式 (p. 225)	SAS 日時値をロケール依存のコンパクト間隔日時形式に変換します。
	NLDATMDTw. 出力形式 (p. 229)	SAS 日時値を現在の SAS ロケールに適した月の名前、月の何日目、および年として書き込みます。
	NLDATMLw. 出力形式 (p. 230)	SAS 日時値を現在の SAS ロケールに適した長い形式の日付として書き込みます。
	NLDATMMw. 出力形式 (p. 231)	現在の SAS ロケールに適した中間の形式の日付で SAS 日時値を書き込みます。
	NLDATMMDw. 出力形式 (p. 232)	SAS 日時値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。
	NLDATMMDLw. 出力形式 (p. 233)	SAS 日時値を現在の SAS ロケールに適した長い形式の月と月の何日目かとして書き込みます。
	NLDATMMDMw. 出力形式 (p. 235)	短縮形を使って、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。
	NLDATMMDSw. 出力形式 (p. 236)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。
	NLDATMMNw. 出力形式 (p. 237)	SAS 日時値を現在の SAS ロケールに適した月の名前として書き込みます。
	NLDATMSw. 出力形式 (p. 238)	現在の SAS ロケールに適した短い形式の日付で SAS 日時値を書き込みます。
	NLDATMTMw. 出力形式 (p. 239)	現在の SAS ロケールに適した時刻で SAS 日時値の時刻部分を書き込みます。

カテゴリ	言語要素	説明
	NLDATMTZw. 出力形式 (p. 241)	SAS 日時値の時刻部分を現在の SAS ロケールに適した時刻とタイムゾーンとして書き込みます。
	NLDATMWw. 出力形式 (p. 242)	SAS 日時値を現在の SAS ロケールに適した曜日と日時として書き込みます。
	NLDATMWNw. 出力形式 (p. 243)	現在の SAS ロケールに適した曜日の形式で SAS 日時値を書き込みます。
	NLDATMWZw. 出力形式 (p. 245)	SAS 日時値を現在の SAS ロケールに適した曜日、日時、およびタイムゾーン値として書き込みます。
	NLDATMYMw. 出力形式 (p. 246)	現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。
	NLDATMYMLw. 出力形式 (p. 247)	現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。
	NLDATMYMMw. 出力形式 (p. 248)	現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。
	NLDATMYMSw. 出力形式 (p. 249)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と年として、書き込みます。
	NLDATMYQw. 出力形式 (p. 250)	現在の SAS ロケールに適した四半期と年として SAS 日時値を書き込みます。
	NLDATMYQLw. 出力形式 (p. 252)	SAS 日時値を現在の SAS ロケールに適したその年の四半期値(1~4)と年として書き込みます。
	NLDATMYQMw. 出力形式 (p. 253)	SAS 日時値を現在の SAS ロケールに適した年の四半期(1~4)と年として書き込みます。
	NLDATMYQSw. 出力形式 (p. 254)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した年と四半期(1~4)として、書き込みます。
	NLDATMYRw. 出力形式 (p. 255)	現在の SAS ロケールに適した年として SAS 日時値を書き込みます。
	NLDATMYWw. 出力形式 (p. 256)	SAS 日時値を現在の SAS ロケールに適した週番号と年として書き込みます。
	NLDATMZw. 出力形式 (p. 257)	SAS 日時値を現在の SAS ロケールに適した日時とタイムゾーンとして書き込みます。
	NLMNACIAED w.d 出力形式 (p. 258)	アラブ首長国連邦の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIAUD w.d 出力形式 (p. 260)	オーストラリアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIBGN w.d 出力形式 (p. 261)	ブルガリアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACIBRL w.d 出力形式 (p. 263)	ブラジルの国際会計表現の通貨出力形式を書き込みます。
	NLMNACICAD w.d 出力形式 (p. 264)	カナダの国際会計表現の通貨出力形式を書き込みます。
	NLMNACICHF w.d 出力形式 (p. 266)	リヒテンシュタインとスイスの国際会計表現の通貨出力形式を書き込みます。
	NLMNACICNY w.d 出力形式 (p. 267)	中国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACICZK w.d 出力形式 (p. 269)	チェコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIDKK w.d 出力形式 (p. 270)	デンマーク、フェロー諸島およびグリーンランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEEK w.d 出力形式 (p. 272)	エストニアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEGPw.d 出力形式 (p. 273)	エジプトの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEURw.d 出力形式 (p. 275)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIGBPw.d 出力形式 (p. 276)	英国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHKDw.d 出力形式 (p. 278)	香港の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHRKw.d 出力形式 (p. 279)	クロアチアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHUFw.d 出力形式 (p. 281)	ハンガリーの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIIDRw.d 出力形式 (p. 282)	インドネシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIILSw.d 出力形式 (p. 284)	イスラエルの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIINRw.d 出力形式 (p. 285)	インドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIJPYw.d 出力形式 (p. 287)	日本の国際会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACIKRWw.d 出力形式 (p. 288)	韓国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACILTLw.d 出力形式 (p. 290)	リトアニアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACILVLw.d 出力形式 (p. 291)	ラトビアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIMOPw.d 出力形式 (p. 293)	マカオの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIMXNw.d 出力形式 (p. 294)	メキシコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIMYRw.d 出力形式 (p. 296)	マレーシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACINOKw.d 出力形式 (p. 297)	ノルウェーの国際会計表現の通貨出力形式を書き込みます。
	NLMNACINZDw.d 出力形式 (p. 299)	ニュージーランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIPLNw.d 出力形式 (p. 300)	ポーランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIRUBw.d 出力形式 (p. 302)	ロシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACISEKw.d 出力形式 (p. 303)	スウェーデンの国際会計表現の通貨出力形式を書き込みます。
	NLMNACISGDw.d 出力形式 (p. 305)	シンガポールの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITHBw.d 出力形式 (p. 306)	タイの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITRYw.d 出力形式 (p. 308)	トルコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITWDw.d 出力形式 (p. 309)	台湾の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIUSDw.d 出力形式 (p. 311)	プエルトリコと米国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIZARw.d 出力形式 (p. 312)	南アフリカの国際会計表現の通貨出力形式を書き込みます。
	NLMNACLAEDx.d 出力形式 (p. 314)	アラブ首長国連邦の現地会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACLAUDw.d 出力形式 (p. 315)	オーストラリアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLBGNw.d 出力形式 (p. 317)	ブルガリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNACLBRLw.d 出力形式 (p. 318)	ブラジルの現地標準表現の通貨出力形式を書き込みます。
	NLMNACLCADw.d 出力形式 (p. 320)	カナダの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCHFw.d 出力形式 (p. 321)	リヒテンシュタインとスイスの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCNYw.d 出力形式 (p. 323)	中国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCZKw.d 出力形式 (p. 324)	チェコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLDKKw.d 出力形式 (p. 326)	デンマーク、フェロー諸島およびグリーンランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEEKw.d 出力形式 (p. 327)	エストニアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEGPw.d 出力形式 (p. 329)	エジプトの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEURw.d 出力形式 (p. 330)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLGBPw.d 出力形式 (p. 332)	英国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHKDw.d 出力形式 (p. 333)	香港の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHRKw.d 出力形式 (p. 335)	クロアチアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHUFw.d 出力形式 (p. 336)	ハンガリーの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLIDRw.d 出力形式 (p. 338)	インドネシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLILSw.d 出力形式 (p. 339)	イスラエルの現地会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACLINRw.d 出力形式 (p. 341)	インドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLJPYw.d 出力形式 (p. 342)	日本の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLKRWw.d 出力形式 (p. 344)	韓国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTLw.d 出力形式 (p. 345)	リトアニアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLVLw.d 出力形式 (p. 347)	ラトビアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMOPw.d 出力形式 (p. 348)	マカオの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMXNw.d 出力形式 (p. 350)	メキシコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMYRw.d 出力形式 (p. 351)	マレーシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLNOKw.d 出力形式 (p. 353)	ノルウェーの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLNZDw.d 出力形式 (p. 354)	ニュージーランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLPLNw.d 出力形式 (p. 356)	ポーランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLRUBw.d 出力形式 (p. 357)	ロシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLSEKw.d 出力形式 (p. 359)	スウェーデンの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLSGDw.d 出力形式 (p. 360)	シンガポールの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTHBw.d 出力形式 (p. 362)	タイの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTRYw.d 出力形式 (p. 363)	トルコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLUSDw.d 出力形式 (p. 365)	台湾の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLUSDw.d 出力形式 (p. 366)	プエルトリコと米国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACLZARw.d 出力形式 (p. 368)	南アフリカの現地会計表現の通貨出力形式を書き込みます。
	NLMNIAEDw.d 出力形式 (p. 369)	アラブ首長国連邦の国際標準表現の通貨出力形式を書き込みます。
	NLMNIAUDw.d 出力形式 (p. 371)	オーストラリアの国際表現の通貨出力形式を書き込みます。
	NLMNIBGNw.d 出力形式 (p. 372)	ブルガリアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIBRLw.d 出力形式 (p. 373)	ブラジルの国際標準表現の通貨出力形式を書き込みます。
	NLMNICADw.d 出力形式 (p. 375)	カナダの国際標準表現の通貨出力形式を書き込みます。
	NLMNICHFW.d 出力形式 (p. 376)	リヒテンシュタインとスイスの国際標準表現の通貨出力形式を書き込みます。
	NLMNICNYw.d 出力形式 (p. 378)	中国の国際標準表現の通貨出力形式を書き込みます。
	NLMNICZKw.d 出力形式 (p. 379)	チェコの国際標準表現の通貨出力形式を書き込みます。
	NLMNIDKKw.d 出力形式 (p. 380)	デンマーク、フェロー諸島およびグリーンランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEEKw.d 出力形式 (p. 382)	エストニアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEGPw.d 出力形式 (p. 383)	エジプトの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEURw.d 出力形式 (p. 385)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際標準表現の通貨出力形式を書き込みます。
	NLMNIGBPw.d 出力形式 (p. 386)	英国の国際標準表現の通貨出力形式を書き込みます。
	NLMNIHKDw.d 出力形式 (p. 388)	香港の国際標準表現の通貨出力形式を書き込みます。
	NLMNIHRKw.d 出力形式 (p. 389)	クロアチアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIHUFw.d 出力形式 (p. 390)	ハンガリーの国際標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNIIDRw.d 出力形式 (p. 392)	インドネシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIILSw.d 出力形式 (p. 393)	イスラエルの国際標準表現の通貨出力形式を書き込みます。
	NLMNIINRw.d 出力形式 (p. 395)	インドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIJPYw.d 出力形式 (p. 396)	日本の国際標準表現の通貨出力形式を書き込みます。
	NLMNIKRWw.d 出力形式 (p. 397)	韓国の国際標準表現の通貨出力形式を書き込みます。
	NLMNITLw.d 出力形式 (p. 399)	リトアニアの国際標準表現の通貨出力形式を書き込みます。
	NLMNILLVw.d 出力形式 (p. 400)	ラトビアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMOPw.d 出力形式 (p. 402)	マカオの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMXNw.d 出力形式 (p. 403)	メキシコの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMYRw.d 出力形式 (p. 405)	マレーシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNINOKw.d 出力形式 (p. 406)	ノルウェーの国際標準表現の通貨出力形式を書き込みます。
	NLMNINZDw.d 出力形式 (p. 407)	ニュージーランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIPLNw.d 出力形式 (p. 409)	ポーランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIRUBw.d 出力形式 (p. 410)	ロシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNISEKw.d 出力形式 (p. 412)	スウェーデンの国際標準表現の通貨出力形式を書き込みます。
	NLMNISGDw.d 出力形式 (p. 413)	シンガポールの国際標準表現の通貨出力形式を書き込みます。
	NLMNITHBw.d 出力形式 (p. 414)	タイの国際標準表現の通貨出力形式を書き込みます。
	NLMNITRYw.d 出力形式 (p. 416)	トルコの国際標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNITWDw.d 出力形式 (p. 417)	台湾の国際標準表現の通貨出力形式を書き込みます。
	NLMNIUSDw.d 出力形式 (p. 419)	プエルトリコと米国の国際標準表現の通貨出力形式を書き込みます。
	NLMNIZARw.d 出力形式 (p. 420)	南アフリカの国際標準表現の通貨出力形式を書き込みます。
	NLMNLAEDx.d 出力形式 (p. 422)	アラブ首長国連邦の現地標準表現の通貨出力形式を書き込みます。
	NLMNLAUDw.d 出力形式 (p. 423)	オーストラリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLBGNw.d 出力形式 (p. 424)	ブルガリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLBRLw.d 出力形式 (p. 426)	ブラジルの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCADw.d 出力形式 (p. 427)	カナダの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCHFw.d 出力形式 (p. 429)	リヒテンシュタインとスイスの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCNYw.d 出力形式 (p. 430)	中国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLCZKw.d 出力形式 (p. 431)	チェコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLDKKw.d 出力形式 (p. 433)	デンマーク、フェロー諸島およびグリーンランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEEKw.d 出力形式 (p. 434)	エストニアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEGPw.d 出力形式 (p. 436)	エジプトの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEURw.d 出力形式 (p. 437)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地標準表現の通貨出力形式を書き込みます。
	NLMNLGBPw.d 出力形式 (p. 439)	英国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLHKDw.d 出力形式 (p. 440)	香港の現地標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNLHRKw.d 出力形式 (p. 442)	クロアチアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLHUFw.d 出力形式 (p. 443)	ハンガリーの現地標準表現の通貨出力形式を書き込みます。
	NLMNLIDRw.d 出力形式 (p. 444)	インドネシアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLILSw.d 出力形式 (p. 446)	イスラエルの現地標準表現の通貨出力形式を書き込みます。
	NLMNLINRw.d 出力形式 (p. 447)	インドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLJPYw.d 出力形式 (p. 449)	日本の現地標準表現の通貨出力形式を書き込みます。
	NLMNLKRWw.d 出力形式 (p. 450)	韓国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLLTLw.d 出力形式 (p. 451)	リトアニアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLLVLw.d 出力形式 (p. 453)	ラトビアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMOPw.d 出力形式 (p. 454)	マカオの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMXNw.d 出力形式 (p. 456)	メキシコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMYRw.d 出力形式 (p. 457)	マレーシアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLNOKw.d 出力形式 (p. 459)	ノルウェーの現地標準表現の通貨出力形式を書き込みます。
	NLMNLNZDw.d 出力形式 (p. 460)	ニュージーランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLPLNw.d 出力形式 (p. 462)	ポーランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLRUBw.d 出力形式 (p. 463)	ロシアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLSEKw.d 出力形式 (p. 464)	スウェーデンの現地標準表現の通貨出力形式を書き込みます。
	NLMNLSGDw.d 出力形式 (p. 466)	シンガポールの現地標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNLTHBw.d 出力形式 (p. 467)	タイの現地標準表現の通貨出力形式を書き込みます。
	NLMNLTRYw.d 出力形式 (p. 469)	トルコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLTDWw.d 出力形式 (p. 470)	台湾の現地標準表現の通貨出力形式を書き込みます。
	NLMNLUSDw.d 出力形式 (p. 471)	プエルトリコと米国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLZARw.d 出力形式 (p. 473)	南アフリカの現地標準表現の通貨出力形式を書き込みます。
	NLMNYw.d 出力形式 (p. 474)	現在の SAS ロケールに適した現地通貨を使用して、現地標準表現の通貨出力形式を書き込みます。
	NLMNYACw.d 出力形式 (p. 476)	現在の SAS ロケールに適した現地通貨を使用して、現地会計表現の通貨出力形式を書き込みます。
	NLMNYIw.d 出力形式 (p. 478)	現在の SAS ロケールに適応するように、国際標準表現の通貨出力形式を書き込みます。
	NLMNYIACw.d 出力形式 (p. 480)	現在の SAS ロケールに適応するように、国際会計表現の通貨出力形式を書き込みます。
	NLNUMw.d 出力形式 (p. 482)	現在の SAS ロケールに適応するように、数値をローカル表示形式で書き込みます。
	NLNUMIw.d 出力形式 (p. 484)	現在の SAS ロケールに適応するように、数値を国際表示形式で書き込みます。
	NLPCTw.d 出力形式 (p. 486)	現在の SAS ロケールに適応するように、百分率のデータを現地表現で書き込みます。
	NLPCTIw.d 出力形式 (p. 488)	現在の SAS ロケールに適応するように、百分率のデータを国際表現で書き込みます。
	NLPCTNw.d 出力形式 (p. 489)	現在の SAS ロケールに適応するように、負の値にはマイナス記号を使用して、百分率を書き込みます。
	NLPCTPw.d 出力形式 (p. 490)	ロケール固有の数値を、現在の SAS ロケールに適したパーセントとして書き込みます。
	NLPVALUEw.d 出力形式 (p. 492)	現在の SAS ロケールに適応するように、p 値をローカル表示形式で書き込みます。
	NLSTRMONw.d 出力形式 (p. 493)	現在の SAS ロケールに適応するように、月名を書き込みます。
	NLSTRQTRw.d 出力形式 (p. 495)	現在の SAS ロケールに適した四半期の形式で数値を書き込みます。

カテゴリ	言語要素	説明
	NLSTRWKw.d 出力形式 (p. 496)	現在の SAS ロケールに適した曜日の形式で数値を書き込みます。
	NLTIMAPw. 出力形式 (p. 498)	SAS 時刻値を現在の SAS ロケールに適した a.m.または p.m.付きの時刻値として書き込みます。
	NLTIMew. 出力形式 (p. 499)	SAS 時刻値を現在の SAS ロケールに適した時刻値として書き込みます。NLTIME は、SAS 日時値も書き込みます。
	NLTIMELw. 出力形式 (p. 501)	SAS 時間値を、現在の SAS ロケールに適した長い形式で時間値として書き込みます。
	NLTIMEMw. 出力形式 (p. 503)	SAS 時間値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で時間値として書き込みます。
	NLTIMESw. 出力形式 (p. 504)	SAS 時間値を、現在の SAS ロケールに適した短い形式で時間値として書き込みます。
	\$VSLOGw. 出力形式 (p. 535)	視覚的順序で並ぶ文字列を処理し、その文字列を左から右の論理的順序で書き込みます。
	\$VSLOGRw. 出力形式 (p. 536)	視覚的順序で並ぶ文字列を処理し、その文字列を右から左の論理的順序で書き込みます。
	WEEKUw. 出力形式 (p. 538)	U アルゴリズムを使用して、10 進数の週番号を書き込みます。
	WEEKVw. 出力形式 (p. 540)	V アルゴリズムを使用して、10 進数の週番号を書き込みます。
	WEEKWw. 出力形式 (p. 542)	W アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YENw.d 出力形式 (p. 544)	円記号、桁区切り、小数点付きの数値を出力します。
	YYWEEKUw. 出力形式 (p. 545)	曜日情報を除外し、U アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YYWEEKVw. 出力形式 (p. 547)	曜日情報を除外し、V アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YYWEEKWw. 出力形式 (p. 549)	曜日情報を除外し、W アルゴリズムを使用して、10 進数の週番号を書き込みます。
DBCS	\$KANJIw. 出力形式 (p. 177)	シフトコードデータを DBCS データに追加します。
	\$KANJIXw. 出力形式 (p. 178)	DBCS データからシフトコードデータを削除します。
ISO 8601	B8601DXw. 出力形式 (p. 143)	協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの基本表記 <code>yyyymmddThhmmss+hhmm</code> を使用して書き出されます。

カテゴリ	言語要素	説明
	B8601LXw. 出力形式 (p. 145)	ISO 8601 規格の基本表記 <code>yyyymmddThhmmss+ -hhmm</code> を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。
	B8601TXw. 出力形式 (p. 147)	協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ISO 8601 規格の基本時間表記 <code>hhmmss+ -hhmm</code> を使用して、ローカル時間を書き出します。
	E8601DXw. 出力形式 (p. 157)	協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 <code>yyyy-mm-ddThh:mm:ss+hh:mm</code> を使用して書き出されます。
	E8601LXw. 出力形式 (p. 159)	ISO 8601 規格の拡張表記 <code>yyyy-mm-ddThh:mm:ss+ -hh:mm</code> を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。
	E8601TXw. 出力形式 (p. 161)	協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ローカルの時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 <code>hh:mm:ss+ -hh:mm</code> を使用して読み込まれます。
数値	BESTDOTXw. 出力形式 (p. 149)	最適な表記を SAS が選択し、ピリオド(.)を小数点として使用することを指定します。
	EUROW.d 出力形式 (p. 163)	先頭にユーロ(E)記号を付け、3 桁ごとにカンマを入れ、小数部分との区切りにはピリオドを使用して数値を出力します。
	EUROXw.d 出力形式 (p. 166)	先頭にユーロ(E)記号を付け、3 桁ごとにピリオドを入れ、小数部分との区切りにはカンマを使用して数値を出力します。
	NLBESTw. 出力形式 (p. 186)	ロケールに基づいて最適な数値表記を書き込みます。
	NLDw.p 出力形式 (p. 188)	値の桁数をできる限り最大にし、同じような大きさの数値に対しては小数点以下の桁数を揃えて出力します。
	NLMNACIAED w.d 出力形式 (p. 258)	アラブ首長国連邦の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIAUD w.d 出力形式 (p. 260)	オーストラリアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIBGN w.d 出力形式 (p. 261)	ブルガリアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIBRL w.d 出力形式 (p. 263)	ブラジルの国際会計表現の通貨出力形式を書き込みます。
	NLMNACICAD w.d 出力形式 (p. 264)	カナダの国際会計表現の通貨出力形式を書き込みます。
	NLMNACICHF w.d 出力形式 (p. 266)	リヒテンシュタインとスイスの国際会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACICNY w.d 出力形式 (p. 267)	中国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACICZK w.d 出力形式 (p. 269)	チェコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIDKK w.d 出力形式 (p. 270)	デンマーク、フェロー諸島およびグリーンランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEEK w.d 出力形式 (p. 272)	エストニアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEGPw.d 出力形式 (p. 273)	エジプトの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIEURw.d 出力形式 (p. 275)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIGBPw.d 出力形式 (p. 276)	英国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHKDw.d 出力形式 (p. 278)	香港の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHRKw.d 出力形式 (p. 279)	クロアチアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIHUFw.d 出力形式 (p. 281)	ハンガリーの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIIDRw.d 出力形式 (p. 282)	インドネシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIILSw.d 出力形式 (p. 284)	イスラエルの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIINRw.d 出力形式 (p. 285)	インドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIJPYw.d 出力形式 (p. 287)	日本の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIKRWw.d 出力形式 (p. 288)	韓国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACILTLw.d 出力形式 (p. 290)	リトアニアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACILVLw.d 出力形式 (p. 291)	ラトビアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACIMOPw.d 出力形式 (p. 293)	マカオの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIMXNw.d 出力形式 (p. 294)	メキシコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIMYRw.d 出力形式 (p. 296)	マレーシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACINOKw.d 出力形式 (p. 297)	ノルウェーの国際会計表現の通貨出力形式を書き込みます。
	NLMNACINZDw.d 出力形式 (p. 299)	ニュージーランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIPLNw.d 出力形式 (p. 300)	ポーランドの国際会計表現の通貨出力形式を書き込みます。
	NLMNACIRUBw.d 出力形式 (p. 302)	ロシアの国際会計表現の通貨出力形式を書き込みます。
	NLMNACISEKw.d 出力形式 (p. 303)	スウェーデンの国際会計表現の通貨出力形式を書き込みます。
	NLMNACISGDw.d 出力形式 (p. 305)	シンガポールの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITHBw.d 出力形式 (p. 306)	タイの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITRYw.d 出力形式 (p. 308)	トルコの国際会計表現の通貨出力形式を書き込みます。
	NLMNACITWDw.d 出力形式 (p. 309)	台湾の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIUSDw.d 出力形式 (p. 311)	プエルトリコと米国の国際会計表現の通貨出力形式を書き込みます。
	NLMNACIZARw.d 出力形式 (p. 312)	南アフリカの国際会計表現の通貨出力形式を書き込みます。
	NLMNACLAEDx.d 出力形式 (p. 314)	アラブ首長国連邦の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLAUDw.d 出力形式 (p. 315)	オーストラリアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLBGNw.d 出力形式 (p. 317)	ブルガリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNACLBRLw.d 出力形式 (p. 318)	ブラジルの現地標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACLCADw.d 出力形式 (p. 320)	カナダの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCHFw.d 出力形式 (p. 321)	リヒテンシュタインとスイスの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCNYw.d 出力形式 (p. 323)	中国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLCZKw.d 出力形式 (p. 324)	チェコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLDKKw.d 出力形式 (p. 326)	デンマーク、フェロー諸島およびグリーンランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEEKw.d 出力形式 (p. 327)	エストニアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEGPw.d 出力形式 (p. 329)	エジプトの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLEURw.d 出力形式 (p. 330)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLGBPw.d 出力形式 (p. 332)	英国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHKDw.d 出力形式 (p. 333)	香港の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHRKw.d 出力形式 (p. 335)	クロアチアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLHUFw.d 出力形式 (p. 336)	ハンガリーの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLIDRw.d 出力形式 (p. 338)	インドネシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLILSw.d 出力形式 (p. 339)	イスラエルの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLINRw.d 出力形式 (p. 341)	インドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLJPYw.d 出力形式 (p. 342)	日本の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLKRWw.d 出力形式 (p. 344)	韓国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNACLLTLw.d 出力形式 (p. 345)	リトアニアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLLVLw.d 出力形式 (p. 347)	ラトビアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMOPw.d 出力形式 (p. 348)	マカオの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMXNw.d 出力形式 (p. 350)	メキシコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLMYRw.d 出力形式 (p. 351)	マレーシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLNOKw.d 出力形式 (p. 353)	ノルウェーの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLNZDw.d 出力形式 (p. 354)	ニュージーランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLPLNw.d 出力形式 (p. 356)	ポーランドの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLRUBw.d 出力形式 (p. 357)	ロシアの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLSEKw.d 出力形式 (p. 359)	スウェーデンの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLSGDw.d 出力形式 (p. 360)	シンガポールの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTHBw.d 出力形式 (p. 362)	タイの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTRYw.d 出力形式 (p. 363)	トルコの現地会計表現の通貨出力形式を書き込みます。
	NLMNACLTDWw.d 出力形式 (p. 365)	台湾の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLUSDw.d 出力形式 (p. 366)	プエルトリコと米国の現地会計表現の通貨出力形式を書き込みます。
	NLMNACLZARw.d 出力形式 (p. 368)	南アフリカの現地会計表現の通貨出力形式を書き込みます。
	NLMNIAEDw.d 出力形式 (p. 369)	アラブ首長国連邦の国際標準表現の通貨出力形式を書き込みます。
	NLMNIAUDw.d 出力形式 (p. 371)	オーストラリアの国際表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNIBGNw.d 出力形式 (p. 372)	ブルガリアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIBRLw.d 出力形式 (p. 373)	ブラジルの国際標準表現の通貨出力形式を書き込みます。
	NLMNICADw.d 出力形式 (p. 375)	カナダの国際標準表現の通貨出力形式を書き込みます。
	NLMNICHFw.d 出力形式 (p. 376)	リヒテンシュタインとスイスの国際標準表現の通貨出力形式を書き込みます。
	NLMNICNYw.d 出力形式 (p. 378)	中国の国際標準表現の通貨出力形式を書き込みます。
	NLMNICZKw.d 出力形式 (p. 379)	チェコの国際標準表現の通貨出力形式を書き込みます。
	NLMNIDKKw.d 出力形式 (p. 380)	デンマーク、フェロー諸島およびグリーンランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEEKw.d 出力形式 (p. 382)	エストニアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEGPw.d 出力形式 (p. 383)	エジプトの国際標準表現の通貨出力形式を書き込みます。
	NLMNIEURw.d 出力形式 (p. 385)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際標準表現の通貨出力形式を書き込みます。
	NLMNIGBPw.d 出力形式 (p. 386)	英国の国際標準表現の通貨出力形式を書き込みます。
	NLMNIHKDw.d 出力形式 (p. 388)	香港の国際標準表現の通貨出力形式を書き込みます。
	NLMNIHRKw.d 出力形式 (p. 389)	クロアチアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIHUFw.d 出力形式 (p. 390)	ハンガリーの国際標準表現の通貨出力形式を書き込みます。
	NLMNIIDRw.d 出力形式 (p. 392)	インドネシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIILSw.d 出力形式 (p. 393)	イスラエルの国際標準表現の通貨出力形式を書き込みます。
	NLMNIINRw.d 出力形式 (p. 395)	インドの国際標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNIJPYw.d 出力形式 (p. 396)	日本の国際標準表現の通貨出力形式を書き込みます。
	NLMNIKRWw.d 出力形式 (p. 397)	韓国の国際標準表現の通貨出力形式を書き込みます。
	NLMNILTlw.d 出力形式 (p. 399)	リトアニアの国際標準表現の通貨出力形式を書き込みます。
	NLMNILVLw.d 出力形式 (p. 400)	ラトビアの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMOPw.d 出力形式 (p. 402)	マカオの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMXNw.d 出力形式 (p. 403)	メキシコの国際標準表現の通貨出力形式を書き込みます。
	NLMNIMYRw.d 出力形式 (p. 405)	マレーシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNINOKw.d 出力形式 (p. 406)	ノルウェーの国際標準表現の通貨出力形式を書き込みます。
	NLMNINZDw.d 出力形式 (p. 407)	ニュージーランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIPLNw.d 出力形式 (p. 409)	ポーランドの国際標準表現の通貨出力形式を書き込みます。
	NLMNIRUBw.d 出力形式 (p. 410)	ロシアの国際標準表現の通貨出力形式を書き込みます。
	NLMNISEKw.d 出力形式 (p. 412)	スウェーデンの国際標準表現の通貨出力形式を書き込みます。
	NLMNISGDw.d 出力形式 (p. 413)	シンガポールの国際標準表現の通貨出力形式を書き込みます。
	NLMNITHBw.d 出力形式 (p. 414)	タイの国際標準表現の通貨出力形式を書き込みます。
	NLMNITRYw.d 出力形式 (p. 416)	トルコの国際標準表現の通貨出力形式を書き込みます。
	NLMNITWDw.d 出力形式 (p. 417)	台湾の国際標準表現の通貨出力形式を書き込みます。
	NLMNIUSDw.d 出力形式 (p. 419)	プエルトリコと米国の国際標準表現の通貨出力形式を書き込みます。
	NLMNIZARw.d 出力形式 (p. 420)	南アフリカの国際標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNLAEDx.d 出力形式 (p. 422)	アラブ首長国連邦の現地標準表現の通貨出力形式を書き込みます。
	NLMNLAUDw.d 出力形式 (p. 423)	オーストラリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLBGNw.d 出力形式 (p. 424)	ブルガリアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLBRLw.d 出力形式 (p. 426)	ブラジルの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCADw.d 出力形式 (p. 427)	カナダの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCHFw.d 出力形式 (p. 429)	リヒテンシュタインとスイスの現地標準表現の通貨出力形式を書き込みます。
	NLMNLCNYw.d 出力形式 (p. 430)	中国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLCZKw.d 出力形式 (p. 431)	チェコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLDDKw.d 出力形式 (p. 433)	デンマーク、フェロー諸島およびグリーンランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEEKw.d 出力形式 (p. 434)	エストニアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEGPw.d 出力形式 (p. 436)	エジプトの現地標準表現の通貨出力形式を書き込みます。
	NLMNLEURw.d 出力形式 (p. 437)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地標準表現の通貨出力形式を書き込みます。
	NLMNLGBPw.d 出力形式 (p. 439)	英国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLHKDw.d 出力形式 (p. 440)	香港の現地標準表現の通貨出力形式を書き込みます。
	NLMNLHRKw.d 出力形式 (p. 442)	クロアチアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLHUFw.d 出力形式 (p. 443)	ハンガリーの現地標準表現の通貨出力形式を書き込みます。
	NLMNLIDRw.d 出力形式 (p. 444)	インドネシアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNLIISw.d 出力形式 (p. 446)	イスラエルの現地標準表現の通貨出力形式を書き込みます。
	NLMNLINRw.d 出力形式 (p. 447)	インドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLJPYw.d 出力形式 (p. 449)	日本の現地標準表現の通貨出力形式を書き込みます。
	NLMNLKRWw.d 出力形式 (p. 450)	韓国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLLTLw.d 出力形式 (p. 451)	リトアニアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLLVLw.d 出力形式 (p. 453)	ラトビアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMOPw.d 出力形式 (p. 454)	マカオの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMXNw.d 出力形式 (p. 456)	メキシコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLMYRw.d 出力形式 (p. 457)	マレーシアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLNOKw.d 出力形式 (p. 459)	ノルウェーの現地標準表現の通貨出力形式を書き込みます。
	NLMNLNZDw.d 出力形式 (p. 460)	ニュージーランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLPLNw.d 出力形式 (p. 462)	ポーランドの現地標準表現の通貨出力形式を書き込みます。
	NLMNLRUBw.d 出力形式 (p. 463)	ロシアの現地標準表現の通貨出力形式を書き込みます。
	NLMNLSEKw.d 出力形式 (p. 464)	スウェーデンの現地標準表現の通貨出力形式を書き込みます。
	NLMNLSGDw.d 出力形式 (p. 466)	シンガポールの現地標準表現の通貨出力形式を書き込みます。
	NLMNLTHBw.d 出力形式 (p. 467)	タイの現地標準表現の通貨出力形式を書き込みます。
	NLMNLTRYw.d 出力形式 (p. 469)	トルコの現地標準表現の通貨出力形式を書き込みます。
	NLMNLTWDw.d 出力形式 (p. 470)	台湾の現地標準表現の通貨出力形式を書き込みます。

カテゴリ	言語要素	説明
	NLMNLUSDw.d 出力形式 (p. 471)	プエルトリコと米国の現地標準表現の通貨出力形式を書き込みます。
	NLMNLZARw.d 出力形式 (p. 473)	南アフリカの現地標準表現の通貨出力形式を書き込みます。
	NLMNYw.d 出力形式 (p. 474)	現在の SAS ロケールに適した現地通貨を使用して、現地標準表現の通貨出力形式を書き込みます。
	NLMNYACw.d 出力形式 (p. 476)	現在の SAS ロケールに適した現地通貨を使用して、現地会計表現の通貨出力形式を書き込みます。
	NLMNYIw.d 出力形式 (p. 478)	現在の SAS ロケールに適応するように、国際標準表現の通貨出力形式を書き込みます。
	NLMNYIACw.d 出力形式 (p. 480)	現在の SAS ロケールに適応するように、国際会計表現の通貨出力形式を書き込みます。
	NLNUMw.d 出力形式 (p. 482)	現在の SAS ロケールに適応するように、数値をローカル表示形式で書き込みます。
	NLNUMIw.d 出力形式 (p. 484)	現在の SAS ロケールに適応するように、数値を国際表示形式で書き込みます。
	NLPCTw.d 出力形式 (p. 486)	現在の SAS ロケールに適応するように、百分率のデータを現地表現で書き込みます。
	NLPCTIw.d 出力形式 (p. 488)	現在の SAS ロケールに適応するように、百分率のデータを国際表現で書き込みます。
	NLPCTNw.d 出力形式 (p. 489)	現在の SAS ロケールに適応するように、負の値にはマイナス記号を使用して、百分率を書き込みます。
	NLPCTPw.d 出力形式 (p. 490)	ロケール固有の数値を、現在の SAS ロケールに適したパーセントとして書き込みます。
	NLPVALUEw.d 出力形式 (p. 492)	現在の SAS ロケールに適応するように、p 値をローカル表示形式で書き込みます。
	NLSTRMONw.d 出力形式 (p. 493)	現在の SAS ロケールに適応するように、月名を書き込みます。
	NLSTRQTRw.d 出力形式 (p. 495)	現在の SAS ロケールに適した四半期の形式で数値を書き込みます。
	NLSTRWKw.d 出力形式 (p. 496)	現在の SAS ロケールに適した曜日の形式で数値を書き込みます。
	YENw.d 出力形式 (p. 544)	円記号、桁区切り、小数点付きの数値を出力します。
日付と時間	B8601DXw. 出力形式 (p. 143)	協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの基本表記 <code>yyyymmddThhmmss+hhmm</code> を使用して書き出されます。

カテゴリ	言語要素	説明
	B8601LXw. 出力形式 (p. 145)	ISO 8601 規格の基本表記 <code>yyyymmddThhmmss+ -hhmm</code> を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。
	B8601TXw. 出力形式 (p. 147)	協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ISO 8601 規格の基本時間表記 <code>hhmmss+ -hhmm</code> を使用して、ローカル時間を書き出します。
	DTWEEKVw. 出力形式 (p. 154)	V アルゴリズムを使用して、10 進数の週番号と日時番号を書き込みます。
	E8601DXw. 出力形式 (p. 157)	協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 <code>yyyy-mm-ddThh:mm:ss+hh:mm</code> を使用して書き出されます。
	E8601LXw. 出力形式 (p. 159)	ISO 8601 規格の拡張表記 <code>yyyy-mm-ddThh:mm:ss+ -hh:mm</code> を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。
	E8601TXw. 出力形式 (p. 161)	協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ローカルの時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 <code>hh:mm:ss+ -hh:mm</code> を使用して読み込まれます。
	HDATEw. 出力形式 (p. 168)	<code>yyyy mmmmm dd</code> 形式で日付値を書き込みます。dd は月の日を、mmmmmm はヘブライ語の月名を、yyyy は年を表します。
	HEBDATEw. 出力形式 (p. 170)	日付値をユダヤ暦で書き込みます。
	JNENGOW.d 出力形式 (p. 172)	SAS の日付値を、元号、年、月、および日の日本の日付として書き込みます。
	JNENGOTw. 出力形式 (p. 173)	SAS 日時値を、元号、年、月、日、時、分などの日本語の日時として、数字に半角文字を使用して書き込みます。
	JNENGOTW w. 出力形式 (p. 174)	SAS 日時値を、元号、年、月、日、時、分などの日本語の日時として、数字に全角文字を使用して書き込みます。
	JNENGOWw.d 出力形式 (p. 175)	SAS 日付値を、元号、年、月、日の日本語の日付として、数字に全角文字を使用して書き込みます。
	MINGUOW. 出力形式 (p. 182)	SAS 日付値を台湾の日付として <code>yyyymmdd</code> 形式で書き込みます。
	NENGOW. 出力形式 (p. 184)	SAS 日付値を日本の日付として <code>e.yymmdd</code> 形式で書き込みます。
	NLDATEw. 出力形式 (p. 190)	現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。

カテゴリ	言語要素	説明
	NLDATECPw.p 出力形式 (p. 192)	SAS 日付値をロケール依存のコンパクト間隔日付形式に変換します。
	NLDATELw. 出力形式 (p. 195)	SAS の日付値を、現在の SAS ロケールに適した月、日付、年の形式で日付として書き込みます。
	NLDATEMw. 出力形式 (p. 196)	SAS 日付値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で日付として書き込みます。
	NLDATEMDw. 出力形式 (p. 198)	SAS の日付値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。
	NLDATEMDLw. 出力形式 (p. 199)	現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。
	NLDATEMDMw. 出力形式 (p. 200)	現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。
	NLDATEMDSw. 出力形式 (p. 201)	SAS 日付値を、現在の SAS ロケールに適した短い形式で日付として書き込みます。
	NLDATEMNw. 出力形式 (p. 202)	SAS 日付値を現在の SAS ロケールに適した月の名前として書き込みます。
	NLDATESw. 出力形式 (p. 204)	SAS 日付値を、現在の SAS ロケールに適した日付文字列として書き込みます。
	NLDATEWw. 出力形式 (p. 205)	SAS 日付値を、現在の SAS ロケールに適した日付と曜日として書き込みます。
	NLDATEWNw. 出力形式 (p. 207)	現在の SAS ロケールに適した曜日で SAS 日付値を書き込みます。
	NLDATEYMw. 出力形式 (p. 209)	SAS の日付値を、現在の SAS ロケールに適した年と月の名前として書き込みます。
	NLDATEYMLw. 出力形式 (p. 211)	現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。
	NLDATEYMMw. 出力形式 (p. 212)	短縮形を使用して、SAS 日付値を現在の SAS ロケールに適した月と年として書き込みます。
	NLDATEYMSw. 出力形式 (p. 213)	現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。
	NLDATEYQw. 出力形式 (p. 214)	現在の SAS ロケールに適した年と四半期として SAS 日付値を書き込みます。
	NLDATEYQLw. 出力形式 (p. 216)	短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1～Q4)として、SAS 日付値を書き込みます。
	NLDATEYQMw. 出力形式 (p. 217)	短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1～Q4)として、SAS 日付値を書き込みます。

カテゴリ	言語要素	説明
	NLDATEYQSw. 出力形式 (p. 218)	数字と区切り文字を使用して、現在の SAS ロケールに適した年とその年の四半期値(1~4)として、SAS 日付値を書き込みます。
	NLDATEYRw. 出力形式 (p. 219)	現在の SAS ロケールに適した年として SAS 日付値を書き込みます。
	NLDATEYWw. 出力形式 (p. 220)	現在の SAS ロケールに適した年と週として SAS 日付値を書き込みます。
	NLDATMw. 出力形式 (p. 222)	現在の SAS ロケールに適した日時として SAS 日時値を書き込みます。
	NLDATMAPw. 出力形式 (p. 223)	現在の SAS ロケールに適した午前または午後付きの日時として SAS 日時値を書き込みます。
	NLDATMCPw.p 出力形式 (p. 225)	SAS 日時値をロケール依存のコンパクト間隔日時形式に変換します。
	NLDATMDTw. 出力形式 (p. 229)	SAS 日時値を現在の SAS ロケールに適した月の名前、月の何日目、および年として書き込みます。
	NLDATMLw. 出力形式 (p. 230)	SAS 日時値を現在の SAS ロケールに適した長い形式の日付として書き込みます。
	NLDATMMw. 出力形式 (p. 231)	現在の SAS ロケールに適した中間の形式の日付で SAS 日時値を書き込みます。
	NLDATMMDw. 出力形式 (p. 232)	SAS 日時値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。
	NLDATMMDLw. 出力形式 (p. 233)	SAS 日時値を現在の SAS ロケールに適した長い形式の月と月の何日目かとして書き込みます。
	NLDATMMDMw. 出力形式 (p. 235)	短縮形を使って、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。
	NLDATMMDSw. 出力形式 (p. 236)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。
	NLDATMMNw. 出力形式 (p. 237)	SAS 日時値を現在の SAS ロケールに適した月の名前として書き込みます。
	NLDATMSw. 出力形式 (p. 238)	現在の SAS ロケールに適した短い形式の日付で SAS 日時値を書き込みます。
	NLDATMTMw. 出力形式 (p. 239)	現在の SAS ロケールに適した時刻で SAS 日時値の時刻部分を書き込みます。
	NLDATMTZw. 出力形式 (p. 241)	SAS 日時値の時刻部分を現在の SAS ロケールに適した時刻とタイムゾーンとして書き込みます。
	NLDATMWw. 出力形式 (p. 242)	SAS 日時値を現在の SAS ロケールに適した曜日と日時として書き込みます。

カテゴリ	言語要素	説明
	NLDATMWNw. 出力形式 (p. 243)	現在の SAS ロケールに適した曜日の形式で SAS 日時値を書き込みます。
	NLDATMWZw. 出力形式 (p. 245)	SAS 日時値を現在の SAS ロケールに適した曜日、日時、およびタイムゾーン値として書き込みます。
	NLDATMYMw. 出力形式 (p. 246)	現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。
	NLDATMYMLw. 出力形式 (p. 247)	現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。
	NLDATMYMMw. 出力形式 (p. 248)	現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。
	NLDATMYMSw. 出力形式 (p. 249)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と年として、書き込みます。
	NLDATMYQw. 出力形式 (p. 250)	現在の SAS ロケールに適した四半期と年として SAS 日時値を書き込みます。
	NLDATMYQLw. 出力形式 (p. 252)	SAS 日時値を現在の SAS ロケールに適したその年の四半期値(1~4)と年として書き込みます。
	NLDATMYQMw. 出力形式 (p. 253)	SAS 日時値を現在の SAS ロケールに適した年の四半期(1~4)と年として書き込みます。
	NLDATMYQSw. 出力形式 (p. 254)	数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した年と四半期(1~4)として、書き込みます。
	NLDATMYRw. 出力形式 (p. 255)	現在の SAS ロケールに適した年として SAS 日時値を書き込みます。
	NLDATMYWw. 出力形式 (p. 256)	SAS 日時値を現在の SAS ロケールに適した週番号と年として書き込みます。
	NLDATMZw. 出力形式 (p. 257)	SAS 日時値を現在の SAS ロケールに適した日時とタイムゾーンとして書き込みます。
	NLTIMAPw. 出力形式 (p. 498)	SAS 時刻値を現在の SAS ロケールに適した a.m.または p.m.付きの時刻値として書き込みます。
	NLTIMEw. 出力形式 (p. 499)	SAS 時刻値を現在の SAS ロケールに適した時刻値として書き込みます。NLTIME は、SAS 日時値も書き込みます。
	NLTIMELw. 出力形式 (p. 501)	SAS 時間値を、現在の SAS ロケールに適した長い形式で時間値として書き込みます。
	NLTIMEMw. 出力形式 (p. 503)	SAS 時間値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で時間値として書き込みます。
	NLTIMESw. 出力形式 (p. 504)	SAS 時間値を、現在の SAS ロケールに適した短い形式で時間値として書き込みます。

カテゴリ	言語要素	説明
	WEEKUw. 出力形式 (p. 538)	U アルゴリズムを使用して、10 進数の週番号を書き込みます。
	WEEKVw. 出力形式 (p. 540)	V アルゴリズムを使用して、10 進数の週番号を書き込みます。
	WEEKWw. 出力形式 (p. 542)	W アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YYWEEKUw. 出力形式 (p. 545)	曜日情報を除外し、U アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YYWEEKVw. 出力形式 (p. 547)	曜日情報を除外し、V アルゴリズムを使用して、10 進数の週番号を書き込みます。
	YYWEEKWw. 出力形式 (p. 549)	曜日情報を除外し、W アルゴリズムを使用して、10 進数の週番号を書き込みます。
ヘブライ語のテキストの操作	\$CPTDWw. 出力形式 (p. 152)	IBM-PC (cp862)でエンコードされたヘブライ語テキスト中の文字列を処理し、その文字列を Windows のヘブライ語エンコーディング(cp 1255)で書き込みます。
	\$CPTWDw. 出力形式 (p. 153)	Windows でエンコード(cp1255)された文字列を処理し、その文字列をヘブライ語 DOS (cp862)エンコーディングで書き込みます。
文字	\$EBCDICw. 出力形式 (p. 156)	ネイティブな形式の文字データを EBCDIC 表現に変換します。
	\$UCS2Bw. 出力形式 (p. 506)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、ビッグエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS2BEw. 出力形式 (p. 507)	ビッグエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UCS2Lw. 出力形式 (p. 509)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、リトルエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS2LEw. 出力形式 (p. 510)	リトルエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UCS2Xw. 出力形式 (p. 512)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、ネイティブエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS2XEw. 出力形式 (p. 513)	ネイティブエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ	言語要素	説明
	\$UCS4Bw. 出力形式 (p. 515)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、ビッグエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS4BEw. 出力形式 (p. 516)	ビッグエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UCS4Lw. 出力形式 (p. 518)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、リトルエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS4LEw. 出力形式 (p. 520)	リトルエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UCS4Xw. 出力形式 (p. 521)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、ネイティブエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。
	\$UCS4XEw. 出力形式 (p. 523)	ネイティブエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UESCw. 出力形式 (p. 524)	現在の SAS セッションでエンコードされる文字列を処理し、その文字列を Unicode エスケープ (UESC) 表記で書き込みます。
	\$UESCEw. 出力形式 (p. 526)	Unicode エスケープ (UESC) 表記の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UNCRw. 出力形式 (p. 527)	現在の SAS セッションでエンコードされる文字列を処理し、その文字列を数値文字表現 (NCR) で書き込みます。
	\$UNCREw. 出力形式 (p. 528)	数値文字表現 (NCR) の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UPARENw. 出力形式 (p. 530)	現在の SAS セッションでエンコードされる文字列を処理し、その文字列を Unicode かつこ (UPAREN) 表記で書き込みます。
	\$UPARENEw. 出力形式 (p. 531)	Unicode かつこ (UPAREN) 内の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。
	\$UTF8Xw. 出力形式 (p. 532)	現在の SAS セッションのエンコーディングを使用する文字列を処理し、UTF-8 (Universal Transformation Format) エンコーディングの文字列を書き込みます。
	\$UTF8XEw. 出力形式 (p. 533)	UTF-8 (Universal Transformation Format) の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

ディクショナリ

B8601DXw. 出力形式

協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの基本表記 `yyyymmddThhmmss+hhmm` を使用して書き出されます。

カテゴリ:	日付と時間 ISO 8601
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。
サポート:	ISO 8601 Element 5.4.1, complete representation
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

B8601DX[w](#).

構文の説明

w	出力フィールドの幅を指定します。
デフォルト	26
範囲	20-35

詳細

UTC 値は、イギリスのグリニッジの基準子午線に基づく日付と時間を示します。SAS は、この出力形式を使用して、日時値を UTC 値に変換したり、TIMEZONE=システムオプションの値を使用することで、ユーザーローカルの日付と時間を決定したりします。TIMEZONE=オプションが設定されていない場合、ユーザーローカルの

日付と時間は、ローカルの日付と時間に基づきます。B8601DX 出力形式は、SAS 日時値を次の ISO 8601 規格の基本日時表記を使用して書き出します。

■ *yyyymmddThhmmss+hhmm*

yyyy

4 桁の年です。

mm

01 から 12 までの 2 桁の月です(ゼロ埋め込み)。

dd

01 から 31 までの 2 桁の日です(ゼロ埋め込み)。

hh

00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

mm

00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

ss

00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

+|-*hhmm*

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは+|-*hhmm* (つまり、+または-を伴う 4 文字)である必要があります。

+は基準子午線の東部タイムゾーンに、-は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、+0200 は基準子午線の東部での 2 時間差を表し、-0600 は基準子午線の西部での 6 時間差を表します。

制約: 短い形式+|-*hh* はサポートされていません。

例

最初の例では、時間とタイムゾーンオフセットの決定にローカル時間を使用します。2 番目の例では、タイムゾーンを America/Adak に変更します。それはハワイ-アリューシャン標準時です。

```
data _null_;
  t='01Feb2021T12:34:56'dt;
  put t b8601dx.;
run;
```

SAS は次の出力をログに書き込みます。

```
20210201T083456-0400
```

この例では、タイムゾーンを America/Adak に変更します。それはハワイ-アリューシャン標準時です。

```
options timezone='America/Adak';
data _null_;
  t='01Feb2021T12:34:56'dt;
  put t b8601dx.;
run;
```

SAS は次の出力をログに書き込みます。

20210201T023456-1000

関連項目

“ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理” ([SAS 出力形式と入力形式: リファレンス](#))

B8601LXw. 出力形式

ISO 8601 規格の基本表記 `yyyymmddThhmmss+|-hhmm` を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。

カテゴリ:	日付と時間 ISO 8601
配置:	右
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。
サポート:	ISO 8601 Elements 5.3.3 and 5.3.4.2
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

B8601LX*w*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 26

範囲 20-35

詳細

B8601LX 出力形式は、ISO 8601 規格の基本日時表記を使用し、時間値を調整せずに書き出してローカル SAS セッションに対応した UTC タイムゾーンオフセットを追加します。

■ `yyyymmddThhmmss+|-hhmm``yyyy`

4 桁の年です。

`mm`

01 から 12 までの 2 桁の月です(ゼロ埋め込み)。

`dd`

01 から 31 までの 2 桁の日です(ゼロ埋め込み)。

`hh`

00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

`mm`

00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

`ss`

00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

`+|-hhmm`

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは`+|-hhmm` (つまり、`+`または`-`を伴う 4 文字)である必要があります。

`+`は基準子午線の東部タイムゾーンに、`-`は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、`+0200` は基準子午線の東部での 2 時間差を表し、`-0600` は基準子午線の西部での 6 時間差を表します。

制約: 短い形式`+|-hh` はサポートされていません。

例

この PUT ステートメントは東部標準タイムゾーンに対応した時間を書き出します。

```
data a;
  blx='01Feb2021T12:34:56'dt;
  put blx=;
  put blx b8601lx.;
run;
```

SAS は次の出力をログに書き込みます。

```
blx=1927802096
20210201T123456-0500
```

関連項目

[“ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理” \(SAS 出力形式と入力形式: リファレンス\)](#)

B8601TXw. 出力形式

協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ISO 8601 規格の基本時間表記 *hhmmss+|-hhmm* を使用して、ローカル時間を書き出します。

カテゴリ:	日付と時間 ISO 8601
配置:	右
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。
サポート:	ISO 8601 Elements 5.3.3 and 5.3.4
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

B8601TX*w*.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 14

範囲 9-20

詳細

UTC 値は、イギリスのグリニッジの基準子午線に基づく時間を示します。SAS は、この出力形式を使用して、時間値を UTC 値に変換したり、TIMEZONE=システムオプションを使用することで、ユーザーのローカル時間を決定したりします。TIMEZONE=オプションが設定されていない場合、ユーザーのローカル時間は、ローカル時間に基づきます。B8601TX 出力形式は、SAS 日時値を次の ISO 8601 規格の基本時間表記を使用して書き出します。

■ *hhmmss+|-hhmm*

hh

00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

mm

00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

ss

00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

+|-hhmm

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは+|-
hhmm (つまり、+または-を伴う 4 文字)である必要があります。

+は基準子午線の東部タイムゾーンに、-は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、+0200 は基準子午線の東部での 2 時間差を表し、-0600 は基準子午線の西部での 6 時間差を表します。

制約: 短い形式+|-*hh* はサポートされていません。

SAS では UTC 時間を B8601TZ 入力形式を使用して読み込む際に、オフセット調整後の時間が 24 時間を超えていたり 00 時間未満である場合、オフセット調整後の時間が 000000 と 240000 の間になるように値を調整します。B8601TX 出力形式をこの時間範囲外の時間に適用する場合、時間はアスタリスクを使用してフォーマットされ、その値が範囲外であることが示されます。

例

最初の例では、時間とタイムゾーンオフセットの決定にローカル時間を使用します。2 番目の例では、タイムゾーンを America/Adak に変更します。それはハワイ-アリューシャン標準時です。

ステートメント	結果
<pre>data _null_; t='12:34:56't; put t b8601tx; run;</pre>	073456-0500
<pre>options timezone='America/Adak'; data _null_; t='12:34:56't; put t b8601tx; run;</pre>	023456-1000

関連項目

[“ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理” \(SAS 出力形式と入力形式: リファレンス\)](#)

BESTDOTXw. 出力形式

最適な表記を SAS が選択し、ピリオド(.)を小数点として使用することを指定します。

カテゴリ: 数値
 配置: 右
 制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

BESTDOTX_w.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 1-32

ヒント 0 から.01 の数字(端点を除く)を出力する場合、過度に丸めないように 7 以上のフィールド幅を使用します。0 から-01 の数字(端点を除く)を出力する場合は、8 以上のフィールド幅を使用します。

詳細

NLDECSEPARATOR システムオプションを無効にすると、BEST_w と BESTDOTX_w の出力形式のデータの処理方法は同じになります。NLDECSEPARATOR システムオプションを有効にすると、BEST と BESTDOTX の出力形式の結果は異なります。この違いについて、次の表に示します。

LOCALE オプション	ロケールのデフォルトの小数点 区切り文字	NLDECSEPARATOR OR オプション	BEST _w .で使用 される区切り文字	BESTDOTX _w .の 場合の小数点 文字
en_US	ピリオド	無効(デフォルト)	ピリオド	ドット(.)
		有効	ピリオド	ドット(.)
fr_FR	カンマ	無効(デフォルト)	ピリオド	ドット(.)
		有効	カンマ	ドット(.)

例

次のコードでは、NLSDECSEPARATOR オプションを有効にし、ロケールを fr_FR に設定して、BESTDOTX 出力形式を使用することで、結果を生成しています。

ステートメント	結果
<pre>options nld locale=fr_FR; data _null_; x=1.2; call symput('macx', put(x, BESTDOTX.)); put x; /* Result is printed as "1,2" */ y=put(x, BESTDOTX.); /*Result is printed as "1.2" */ run;</pre>	<pre>1,2 1.2</pre>
<pre>&put &macx;</pre>	<pre>1.2</pre>
<pre>data _null_; bx=&macx; /* Succeeds because the value of &macx is 1.2 */ put bx; run;</pre>	<pre>1.2</pre>

関連項目

- [“BESTw. 出力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)
- [“NLDECSEPARATOR システムオプション” \(980 ページ\)](#)

\$BIDIw. 出力形式

ラテン語の単語と数字の順序を維持しつつ、ヘブライ語とアラビア語の文字の順序を逆にするにより、論理的な順序の文字列と視覚的な順序の文字列を双方向に変換します。

カテゴリ:	BIDI テキストの操作
配置:	左
制限事項:	CAS

構文

\$BIDIw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト **w** が指定されていない場合は 1。

範囲 1-32767

詳細

Windows の動作環境では、ヘブライ語とアラビア語のテキストが論理的な順序で保存されます。テキストは書き込まれるときの順序で保存され、表示されるときに順序で保存されるとは限りません。ただし、その他の動作環境でのヘブライ語のテキストは、表示されるときと同じ順序で保存されます。SAS ユーザーは、逆順のヘブライ語とアラビア語のテキストを検出できます。このような状況は、SAS/CONNECT または他のソフトウェアを使用して、ヘブライ語およびアラビア語のテキストを含む SAS データセットまたはレポートを表示動作環境から論理動作環境に転送するときに発生する可能性があります。\$BIDI 出力形式は、数字および Latin-1 単語の順序を維持しながらヘブライ語とアラビア語のテキストを逆順にする出力形式です。

動作環境の情報: 汎用機動作環境でのこの出力形式は、NewCode Hebrew および Arabic を使用できるように設計されています。一部の汎用機動作環境では、OldCode Hebrew または Arabic エンコーディングを使用するため、不満足な結果になる場合があります。このエンコーディングの Hotfix は、[SAS Institute の Web サイト](#)にあります。

比較

\$BIDIw.出力形式では、文字データを逆順に書き込んで空白を保持する\$REVERJw.出力形式と似た逆転処理が実行されます。\$BIDIw.は、次のように動作します。

- \$BIDIw.により、空白を保持したまま、指定した文字列の単語と数字の順序が逆になります。Latin-1 の単語と数字自体は逆順にならず、文字列での順序のみが逆になります。
- テキスト文字列にヘブライ語またはアラビア語で構成される単語が登場すると、\$BIDI により、ヘブライ語またはアラビア語の単語の文字が逆順になります。文字列中のヘブライ語またはアラビア語の単語の位置は逆順です。

例

この例は、\$BIDIw.によってヘブライ語文字が逆順になる方法を示しています。文字列内のヘブライ語は逆順になります。単語内のヘブライ語文字も逆になります。

```
data;
  a= 'שלוש תיבות א abc 123';
  b1 = put (a,$bidi20.);
  put b=;
  b2 = put (b,$bidi20.);
  put b=;
run;
```

次の行が SAS ログに書き出されます。

```
b1=123 abc א תיבות שלוש
b2=שלוש תיבות א abc 123
```

\$CPTDWw. 出力形式

IBM-PC (cp862)でエンコードされたヘブライ語テキスト中の文字列を処理し、その文字列を Windows のヘブライ語エンコーディング(cp 1255)で書き込みます。

カテゴリ: ヘブライ語のテキストの操作

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$CPTDWw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 200

範囲 1-32767

比較

\$CPTDWw.出力形式では、\$CPTWDw.出力形式とは逆の処理を行います。

例

次の例では、入力値として"808182x"を使用します。

ステートメント	結果
	----+----1----+
put text \$cptdw3,;	אבא

関連項目

出力形式:

- ["\\$CPTWDw. 出力形式" \(153 ページ\)](#)

入力形式:

- ["\\$CPTDWw. 入力形式" \(773 ページ\)](#)
- ["\\$CPTWDw. 入力形式" \(774 ページ\)](#)

\$CPTWDw. 出力形式

Windows でエンコード(cp1255)された文字列を処理し、その文字列をヘブライ語 DOS (cp862)エンコーディングで書き込みます。

カテゴリ:	ヘブライ語のテキストの操作
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$CPTWDw.

構文の説明

w	出力フィールドの幅を指定します。
デフォルト	200
範囲	1-32767

比較

\$CPTWDw.出力形式では、\$CPTDWw.出力形式とは逆の処理を行います。

例

次の例では、入力値として『1111』を使用します。

ステートメント	結果
	----+----1----+----2----+
put text \$cptwd3,;	€□,

関連項目

出力形式:

- ["\\$CPTDWw. 出力形式" \(152 ページ\)](#)

入力形式:

- ["\\$CPTDWw. 入力形式" \(773 ページ\)](#)
- ["\\$CPTWDw. 入力形式" \(774 ページ\)](#)

DTWEEKVw. 出力形式

V アルゴリズムを使用して、10 進数の週番号と日時番号を書き込みます。

カテゴリ: CAS
日付と時間
配置: 左

構文

DTWEEKVw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 18

範囲 3-200

詳細

この表には幅、出力形式および例の情報が含まれています。

幅	出力形式	例
3-4	<i>Www</i>	w01
5-6	<i>yyWww</i>	03w01
7-8	<i>yyWwwdd</i>	03w0101
9-11	<i>yyyyWwwdd</i>	2003w0101
12-14	<i>yyyyWwwdd:hh</i>	2003w0101:09
15-17	<i>yyyyWwwdd:hh:mm</i>	2003w0101:09:10
18-19	<i>yyyyWwwdd:hh:mm:ss</i>	2003w0101:09:10:56
20-200	<i>yyyy-Www-dd:hh:mm:ss</i>	2003-w01-01:09:10:56

例

DTWEEKVw.出力形式の例を次に示します。

ステートメント	結果
DATA _NULL_; t='01Feb2017:1:2:3'dt; put t= DTWEEKV30.;	t=2017-W05-03:01:02:03
RUN;	

\$EBCDICw. 出力形式

ネイティブな形式の文字データを EBCDIC 表現に変換します。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

注: サポートされているセッションエンコーディングは UTF-8 のみです。\$EBCDIC 出力形式は 7 ビット ASCII 文字でのみ動作します。

構文

\$EBCDICw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 1

範囲 1-32767

詳細

EBCDIC がネイティブな形式の場合、変換は実行されません。

ASCII システムでは、\$EBCDICw.出力形式は、SAS 起動時に指定された LOCALE=オプションのデフォルトエンコーディング値に基づいています。たとえば、ロケールが en_US ロケールに設定されていた場合、\$EBCDICw.出力形式で使用するデフォルトのエンコーディングは Open_ed-1047 です。ロケールが de_DE (German_Germany)である場合、\$EBCDICw.出力形式で使用するデフォルトのエンコーディングは Open_ed-1141 です。ロケールおよびエンコーディング値のリストについては、[表 21.1 \(1097 ページ\)](#)を参照してください。

MAPEBCDIC2ASCII システムオプションを使用して、EBCDIC と ASCII 間での文字のマッピングに使用される変換テーブルを指定できます。詳細については、["MAPEBCDIC2ASCII= システムオプション" \(979 ページ\)](#)を参照してください。

比較

- ASCII システムで\$EBCDICw.を使用すると、ASCII 文字データは EBCDIC に変換されます。
- その他のすべてのシステムでは、\$EBCDICw は\$CHARw.出力形式と同じ結果になります。

例

```
data one;
  input x $;
  datalines;
ABC
{ä}
[@]
;
run;

data two;
  set one;
  y=put(x,$ebcdic3.);
  put y $HEX6.;
run;
```

表 10.1 アメリカ英語、フランス語、ドイツ語の出力ロケール値

name の値	ロケール値が en_US の場合	ロケール値が fr_FR の場合	ロケール値が de_DE の場合
ABC	C1C2C3	C1C2C3	C1C2C3
{ä}	C043D0	514354	43C0DC
[@]	AD7CBD	9044B5	63B5FC

上記の結果は、EBCDIC 形式のコード値を 16 進表現で出力したものです。16 進数の各ペアがバイナリデータの 1 バイトに相当し、各バイトが 1 文字に相当します。

E8601DXw. 出力形式

協定世界時(UTC)日時値をユーザーのローカル日付と時間に調整します。その後、ローカルの日付と時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 `yyyy-mm-ddThh:mm:ss+hh:mm` を使用して書き出されます。

カテゴリ: 日付と時間
ISO 8601

配置:	左
サポート:	ISO 8601 Element 5.4.1, complete representation
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

E8601DX^w.

構文の説明

^w
出力フィールドの幅を指定します。

デフォルト 26

範囲 20-35

詳細

UTC 値は、イギリスのグリニッジの基準子午線に基づく日付と時間を示します。SAS は、この出力形式を使用して、日時値を UTC 値に変換したり、TIMEZONE=システムオプションの値を使用することで、ユーザーローカルの日付と時間を決定したりします。TIMEZONE=オプションが設定されていない場合、ユーザーローカルの日付と時間は、ローカルの日付と時間に基づきます。E8601DX 出力形式は、SAS 日時値を次の ISO 8601 規格の基本日時表記を使用して書き出します。

■ *yyyy-mm-ddThh:mm:ss+hh:mm*

yyyy
4 桁の年です。

mm
01 から 12 までの 2 桁の月です(ゼロ埋め込み)。

dd
01 から 31 までの 2 桁の日です(ゼロ埋め込み)。

hh
00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

mm
00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

ss
00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

+|-hh:mm

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは+|-hh:mm (つまり、+または-を伴う 5 文字)である必要があります。

+は基準子午線の東部タイムゾーンに、-は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、+02:00 は基準子午線の東部での 2 時間差を表し、-06:00 は基準子午線の西部での 6 時間差を表します。

制約: 短い形式+|-hh はサポートされていません。

例

最初の例では、時間の決定にローカル時間を使用します。2 番目の例では、タイムゾーンを America/Adak に変更します。それはハワイ-アリューシャン標準時です。

ステートメント	結果
data _null_; t='15Sep2018T12:34:56'dt; put t e8601dx; run;	2018-09-15T08:34:56-04:00
options timezone='America/Adak'; data _null_; t='15Sep2018T12:34:56'dt; put t e8601dx; run;	2018-09-15T03:34:56-09:00

関連項目

[“ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理” \(SAS 出力形式と入力形式: リファレンス\)](#)

E8601LXw. 出力形式

ISO 8601 規格の拡張表記 yyyy-mm-ddThh:mm:ss+|-hh:mm を使用して、ローカル時間と UTC 間のタイムゾーンオフセット差を追加することにより、日時値をローカル時間として書き出します。

カテゴリ: 日付と時間

ISO 8601

配置: 右

サポート: ISO 8601 Elements 5.3.3 and 5.3.4.2

注: **2025.12** 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

E8601LX^w

構文の説明

^w

出力フィールドの幅を指定します。

デフォルト 26

範囲 20-35

詳細

E8601LX 出力形式は、ISO 8601 規格の基本日時表記を使用し、時間値を調整せずに書き出してローカル SAS セッションに対応した UTC タイムゾーンオフセットを追加します。

■ `yyyy-mm-ddThh:mm:ss+hh:mm`

yyyy

4 桁の年です。

mm

01 から 12 までの 2 桁の月です(ゼロ埋め込み)。

dd

01 から 31 までの 2 桁の日です(ゼロ埋め込み)。

hh

00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

mm

00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

ss

00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

+|-*hh:mm*

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは+|-*hh:mm* (つまり、+または-を伴う 5 文字)である必要があります。

+は基準子午線の東部タイムゾーンに、-は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、+02:00 は基準子午線の東部での 2 時間差を表し、-06:00 は基準子午線の西部での 6 時間差を表します。

制約: 短い形式+|-*hh* はサポートされていません。

例

この例は東部タイムゾーンに対応した時間を書き出します。

```
data one;
  mydt='15sep2018 T12:34:56'dt;
  put mydt e8601lx.
run;
```

mydt の値	結果
1852634096	2018-09-15T12:34:56-05:00

関連項目

["ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理" \(SAS 出力形式と入力形式: リファレンス\)](#)

E8601TXw. 出力形式

協定世界時(UTC)値をユーザーのローカル時間に調整します。その後、ローカルの時間は、ISO 8601 規格の日時とタイムゾーンの拡張表記 `hh:mm:ss+|-hh:mm` を使用して読み込まれます。

カテゴリ: 日付と時間
ISO 8601

配置: 右

サポート: ISO 8601 Elements 5.3.3 and 5.3.4

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

E8601TXw.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 14

範囲 9-20

詳細

UTC 値は、イギリスのグリニッジの基準子午線に基づく時間を示します。SAS は、この出力形式を使用して、時間値を UTC 値に変換したり、TIMEZONE=システムオプションの値を使用することで、ユーザーのローカル時間を決定したりします。TIMEZONE=オプションが設定されていない場合、ユーザーのローカル時間は、ローカル時間に基づきます。E8601TX 出力形式は、SAS 日時値を次の ISO 8601 規格の基本時間表記を使用して書き出します。

■ *hh:mm:ss+|-hh:mm*

hh

00 から 23 までの 2 桁の時間です(ゼロ埋め込み)。

mm

00 から 59 までの 2 桁の分です(ゼロ埋め込み)。

ss

00 から 59 までの 2 桁の秒です(ゼロ埋め込み)。

+|-*hh:mm*

基準子午線時間からの時間と分の符号付きオフセットです。オフセットは+|-*hhmm* (つまり、+または-を伴う 4 文字)である必要があります。

+は基準子午線の東部タイムゾーンに、-は基準子午線の西部タイムゾーンにそれぞれ使用します。たとえば、+0200 は基準子午線の東部での 2 時間差を表し、-0600 は基準子午線の西部での 6 時間差を表します。

制約: 短い形式+|-*hh* はサポートされていません。

SAS では UTC 時間を E8601TZ 入力形式を使用して読み込む際に、オフセット調整後の時間が 24 時間を超えていたり 00 時間未満である場合、オフセット調整後の時間が 000000 と 240000 の間になるように値を調整します。E8601TX 出力形式をこの時間範囲外の時間に適用する場合、時間はアスタリスクを使用してフォーマットされ、その値が範囲外であることが示されます。

例

最初の例では、時間とタイムゾーンオフセットの決定にローカル時間を使用します。2 番目の例では、タイムゾーンを America/Adak に変更します。それはハワイ-アリューシャン標準時です。

ステートメント

結果

```
data _null_;
t='12:34:56't;
put t e8601tx;
```

07:34:56-05:00

ステートメント	結果
run;	
options timezone='America/Adak';	02:34:56-10:00
data _null_;	
t='12:34:56't;	
put t e8601tx.;	
run;	

関連項目

“ISO 8601 規格の基本表記と拡張表記を使用した日付と時間の処理” (SAS 出力形式と入力形式: リファレンス)

EUROW.d 出力形式

先頭にユーロ(E)記号を付け、3 桁ごとにカンマを入れ、小数部分との区切りにはピリオドを使用して数値を出力します。

カテゴリ: CAS
数値

配置: 右

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

EUROW.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 1-32

ヒント 出力にユーロ記号を含める場合、適切な幅を選択していることを確認してください。

<i>d</i>	数値の小数点以下の桁数を指定します。
デフォルト	0
範囲	0-31
要件	<i>w</i> より小さい数にします。

比較

- EUROW.*d* 出力形式は EUROX*w.d* 出力形式に似ていますが、EUROX*w.d* 出力形式では小数点とカンマの役割が逆になっています。この規則はヨーロッパの国で共通です。
- EUROW.*d* 出力形式は DOLLAR*w.d* 出力形式に似ていますが、DOLLAR*w.d* 出力形式は先頭にユーロ記号のかわりにドル記号を付けます。

例

この例では、金額の値に 1254.71 を使用します。

ステートメント	結果
	-----1-----2-----3
put amount euro10.2;	E1,254.71
put amount euro5.;	1,255
put amount euro9.2;	E1,254.71
put amount euro15.3;	E1,254.710

```
data _null_;
  input x;
  put x euro10.2;
  put x euro5.;;
  put x euro9.2;
  put x euro15.3;
  datalines;
1254.71
;
run;
SAS Log:
E1,254.71
1,255
E1,254.71
E1,254.710
```

```
/* This code determines the default length. */
```

```
data _null_;
  input x;
  put x euro.;
  datalines;
```

```
1
22
333
4444
55555
666666
7777777
88888888
999999999
1234561234
```

```
;run;
```

```
SAS Log:
```

```
  datalines;
```

```
    E1
```

```
    E22
```

```
    E333
```

```
    E4,444
```

```
    55,555
```

```
    666666
```

```
    7.78E6
```

```
    8.89E7
```

```
    1E9
```

```
    1.23E9
```

NOTE: At least one W.D format was too small for the number to be printed.

The decimal may be shifted by the "BEST" format.

```
/* This code determines the range. */
```

```
data _null_;
```

```
  input x;
```

```
  put x euro5.;
```

```
  put x euro6.;
```

```
  put x euro7.;
```

```
  put x euro8.;
```

```
  put x euro9.;
```

```
  put x euro9.2;
```

```
  put x euro10.;
```

```
  put x euro10.2;
```

```
  put x euro10.4;
```

```
  put x euro11.;
```

```
  put x euro11.3;
```

```
  put x euro12.;
```

```
  put x euro12.2;
```

```
  put x euro13.;
```

```
  put x euro13.2;
```

```
  datalines;
```

```
333
```

```
4444
```

```
55555
```

```
666666
```

```
7777777
```

```
88888888
```

```
999999999
```

1234561234
;run;

関連項目

出力形式:

- [“EUROXw.d 出力形式” \(166 ページ\)](#)

入力形式:

- [“EUROW.d 入力形式” \(775 ページ\)](#)
- [“EUROXw.d 入力形式” \(778 ページ\)](#)

EUROXw.d 出力形式

先頭にユーロ(E)記号を付け、3 桁ごとにピリオドを入れ、小数部分との区切りにはカンマを使用して数値を出力します。

カテゴリ: CAS
数値

配置: 右

注: [2025.12](#) 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

EUROXw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 1-32

ヒント 出力にユーロ記号を含める場合、適切な幅を選択していることを確認してください。

- d*
- 数値の小数点以下の桁数を指定します。
- デフォルト 0
- 範囲 0-31
- 要件 *w* より小さい数にします。

比較

- EUROX*w.d* 出力形式は EUROW*d* 出力形式に似ていますが、EUROW*d* 出力形式では小数点とカンマの役割が逆になっています。この規則は英語圏の国で共通です。
- EUROX*w.d* 出力形式は DOLLARX*w.d* 出力形式に似ていますが、DOLLARX*w.d* 出力形式は先頭にユーロ記号のかわりにドル記号を付けます。

例

この例では、金額の値に 1254.71 を使用します。

ステートメント	結果
	-----1-----2-----3
put amount eurox10.2;	E1.254,71
put amount eurox5.;	1.255
put amount eurox9.2;	E1.254,71
put amount eurox15.3;	E1.254,710

```
data _null_;
  input x;
  put x eurox10.2;
  put x eurox5.;;
  put x eurox9.2;
  put x eurox15.3;
  datalines;
1254.71
; run;
SAS Log:
E1.254,71
1.255
E1.254,71
E1.254,710
```

```

/* This code determines the default length. */
data _null_;
  input x;
  put x eurox.;
  datalines;
1
22
333
4444
55555
666666
7777777
88888888
999999999
1234561234
;run;
SAS Log:
  E1
  E22
  E333
E4.444
55.555
666666
7.78E6
8.89E7
  1E9
1.23E9

```

注: 1 つ以上の W.D 出力形式において、数値を出力するには幅の長さが十分ではありませんでした。"BEST"出力形式が使用されて、10 進数のシフトが行われた可能性があります。

関連項目

出力形式:

- ["EUROw.d 出力形式" \(163 ページ\)](#)

入力形式:

- ["EUROw.d 入力形式" \(775 ページ\)](#)
- ["EUROXw.d 入力形式" \(778 ページ\)](#)

HDATEw. 出力形式

yyyy mmmmm dd 形式で日付値を書き込みます。dd は月の日を、mmmmm はヘブライ語の月名を、yyyy は年を表します。

カテゴリ:	日付と時間
配置:	右
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

HDATEw.

構文の説明

w
出力フィールドの幅を指定します。

.....
注: 最善の表示状態にするため、幅には 9、11、15、17 を使用してください。
.....

デフォルト 17

範囲 9-17

詳細

HDATEw.出力形式では、SAS 日付値を *yyyy mmmmm dd* 形式で書き込みます。

yyyy
年です。

mmmmm
ヘブライ文字で書かれた英語の月名です。

dd
月の日です。

例

次の例では、15780 を入力値として使用します。この値は 2003 年 3 月 16 日に相当する SAS 日付値です。

ステートメント	結果
	----+----1----+----2----+
put day hdate9.;	03 יָרֵךְ 16
put day hdate11.;	2003 יָרֵךְ 16
put day hdate17.;	2003 יָרֵךְ 16

関連項目

出力形式:

- [“HEBDATEw. 出力形式” \(170 ページ\)](#)

HEBDATEw. 出力形式

日付値をユダヤ暦で書き込みます。

カテゴリ: 日付と時間

配置: 右

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

HEBDATEw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 7-24

ヒント 非ヘブライエンコーディングを使用する場合、アスタリスクはヘブライ文字の代わりに表示されます。

詳細

ユダヤ暦は、太陽暦と太陰暦を組み合わせた暦です。年は天地創造以来の年数として計算されます。ユダヤの歴史によれば、天地創造はキリスト教の始まりより 3760 年と 3 か月前です。ユダヤ暦の年数を計算するには、グレゴリオ暦の指定された年の秋を起点として 3761 年を追加する必要があります。

HEBDATEw.出力形式では、SAS 日付値をユダヤ暦に従って書き込みます。日付は次のいずれかの出力形式で書き出されます。

long

ראשון י' אדר ה'תשס"ג

デフォルト

י' אדר תשס"ג

short

י' / 1 / תשס"ג

例

次の例では、15780 を入力値として使用します。この値は 2003 年 3 月 16 日に相当する SAS 日付値です。

ステートメント	結果
	----+----1----+
put day hebdate13.;	י' ב' / 21 ' תשס"ג
put day hebdate16.;	י' ב' אדר ב' - תשס"ג
put day hebdate24.;	ראשון י' ב' אדר ב' - ה' תשס"ג

関連項目

入力形式:

■ [“HDATEw. 出力形式” \(168 ページ\)](#)

JNENGOW.d 出力形式

SAS の日付値を、元号、年、月、および日の日本の日付として書き込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。
この形式は、日本語環境でのみサポートされています。

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

JNENGOW.d

オプション引数

w

入力幅を指定します。

デフォルト 21

範囲 6-200

d

次の値を指定します。

- .0 すべての情報を書き込みます。
- .1 元号、年、月の情報を書き込みます。
- .2 元号と年の情報を書き込みます。
- .3 元号の情報のみを書き込みます。

デフォルト 0

範囲 0-3

比較

JNENGOW.d は JNENGOWw.d に似ていますが、JNENGOW.d は数字に半角文字を使用します。

例

この例では、Japanese_Japan セッションエンコーディングを使用します。ここでは、15760 を入力値として使用します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。

```
Data _null;  
  x = 15760;  
  put x JNENG0.3;  
  put x JNENG0.2;  
  put x JNENG0.1;  
  put x JNENG0.0;  
run;
```

```
平成  
平成 15 年  
平成 15 年 2 月  
平成 15 年 2 月 24 日
```

JNENGOTw. 出力形式

SAS 日時値を、元号、年、月、日、時、分などの日本語の日時として、数字に半角文字を使用して書き込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。
この形式は、日本語環境でのみサポートされています。

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

JNENGOTw.

オプション引数

w

入力幅を指定します。

デフォルト 32

範囲 27-200

比較

JNENGOTw. は JNENGOTWw. に似ていますが、JNENGOTw. は数字に半角文字を使用します。

例

この例では、Japanese_Japan セッションエンコーディングを使用します。ここでは、1361709583 を入力値として使用します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。

```
data _null_;
  x = 1361709583;
  put x JNENGOT32.;
  put x JNENGOT40.;
run;
```

```
平成 15 年 2 月 24 日 12:39
平成 15 年 2 月 24 日 12 時 39 分
```

JNENGOTW w. 出力形式

SAS 日時値を、元号、年、月、日、時、分などの日本語の日時として、数字に全角文字を使用して書き込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この形式は、日本語環境でのみサポートされています。
この出力形式は CAS で実行する DATA ステップではサポートされていません。

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

JNENGOTWw.

オプション引数

w

入力幅を指定します。

デフォルト 52

範囲 49-200

比較

JNENGOTWw. は JNENGOTw. に似ていますが、JNENGOTWw. は数字に半角文字を使用します。

例

この例では、Japanese_Japan セッションエンコーディングを使用します。ここでは、1361709583 を入力値として使用します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。

```
Data _null_;
  x = 1361709583;
  put x JNENGOTW49.;
  put x JNENGOTW80.;
run;
```

```
平成 15 年  2 月 24 日  12 : 39
平成 15 年  2 月 24 日  12 時 39 分
```

JNENGOWw.d 出力形式

SAS 日時値を、元号、年、月、日の日本語の日付として、数字に全角文字を使用して書き込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。
この形式は、日本語環境でのみ実行できます。

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

JNENGOWw.d

オプション引数

w

入力幅を指定します。

デフォルト 33

範囲 6-200

d

次の値を指定します。

- .0 すべての情報を書き込みます。
- .1 元号、年、月の情報を書き込みます。
- .2 元号と年の情報を書き込みます。
- .3 元号の情報のみを書き込みます。

デフォルト 0

範囲 0-3

比較

JNENGOWw.d は JNENGOW.d に似ていますが、JNENGOWw.d は数字に半角文字を使用します。

例

この例では、Japanese_Japan セッションエンコーディングを使用します。ここでは、15760 を入力値として使用します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。

```
data _null;
  x=15760 ;
  put x JNENGOW.3;
  put x JNENGOW.2;
  put x JNENGOW.1;
  put x JNENGOW.;
```

```
run;
```

```
平成
平成 15 年
平成 15 年 2 月
平成 15 年 2 月 24 日
```

\$KANJIw. 出力形式

シフトコードデータを DBCS データに追加します。

カテゴリ: DBCS

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$KANJIw.

構文の説明

w

出力フィールドの幅を指定します。

範囲 出力形式の最小幅は、 $2 + (\text{現在の DBCSTYPE} = \text{設定で使用するシフトコードの長さ}) * 2$ です。

制限事項 幅は偶数幅(2 バイト単位)で指定します。奇数幅を指定すると幅が切り捨てられます。この幅は、シフトコードデータの長さ以上にする必要があります。

詳細

\$KANJI 出力形式では、シフトコードデータが、シフトコードデータのない DBCS データに追加されます。入力データが空白の場合、シフトコードデータは追加されません。

\$KANJI 出力形式で処理されるのはホストメインフレームデータですが、\$KANJI は他のプラットフォームで使用できます。\$KANJI 出力形式を非 EBCDIC (モーダルエンコーディングではない)ホストで使用する場合、データは変化しません。

関連項目

出力形式:

- ["\\$KANJIw. 出力形式" \(178 ページ\)](#)

入力形式:

- ["\\$KANJIw. 入力形式" \(781 ページ\)](#)
- ["\\$KANJIw. 入力形式" \(782 ページ\)](#)

システムオプション:

- ["DBCSTYPE システムオプション" \(964 ページ\)](#)

\$KANJIw. 出力形式

DBCS データからシフトコードデータを削除します。

カテゴリ: DBCS

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$KANJIw.

構文の説明

w

出力フィールドの幅を指定します。

範囲 出力形式の最小幅は 2 です。

制限事項 幅は偶数幅(2 バイト単位)で指定します。奇数幅を指定すると幅が切り捨てられます。この幅は、シフトコードデータの長さ以上にする必要があります。

詳細

\$KANJI 出力形式では、DBCS データからシフトコードデータが削除されます。入力データ長は、 $2 + (\text{SO/SI length}) * 2$ である必要があります。SBCS データが返された場合を除き、データは SO で開始し、SI で終了する必要があります。

\$KANJIIX 出力形式で処理されるのはホストメインフレームデータですが、\$KANJIIX は他のプラットフォームで使用できます。\$KANJIIX 出力形式を非 EBCDIC (モデルエンコーディングではない)ホストで使用する場合、データは変化しません。

関連項目

出力形式:

- ["\\$KANJIw. 出力形式" \(177 ページ\)](#)

入力形式:

- ["\\$KANJIw. 入力形式" \(781 ページ\)](#)
- ["\\$KANJIIXw. 入力形式" \(782 ページ\)](#)

システムオプション:

- ["DBCSTYPE システムオプション" \(964 ページ\)](#)

\$LOGVSw. 出力形式

左から右に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。

カテゴリ: BIDI テキストの操作
CAS
配置: 左

構文

\$LOGVSw.

構文の説明

w
出力フィールドの幅を指定します。
デフォルト 200
範囲 1-32767

詳細

\$LOGVSw.出力形式は、論理的順序のテキストを視覚サーバーに保存するときに使用します。

注: \$LOGVSw.出力形式を使用できない場合、データのヘブライ語またはアラビア語の部分は逆順になります。

比較

\$LOGVSw.出力形式では、\$LOGVSRw.出力形式とは逆の処理を行います。

例

次の例では、"flight"のヘブライ語の入力値を使用します。 טיסה

ステートメント	結果
	----+----1----+----2----+
put text \$logvs12.;	טיסה flight

次の例では、computer のアラビア語の入力値"ثابت"を使用します。

ステートメント	結果
	----+----1----+----2----+
put text \$logvs12.;	ثابت コンピューター

関連項目

出力形式:

- ["\\$LOGVSRw. 出力形式" \(181 ページ\)](#)

入力形式:

- ["\\$LOGVSRw. 入力形式" \(785 ページ\)](#)

\$LOGVSRw. 出力形式

右から左に論理的順序で並ぶ文字列を処理し、視覚的順序でその文字列を書き込みます。

カテゴリ: BIDI テキストの操作
CAS
配置: 左

構文

\$LOGVSRw.

構文の説明

w
出力フィールドの幅を指定します。
デフォルト 200
範囲 1-32767

詳細

\$LOGVSRw.出力形式は、論理的順序のテキストを視覚サーバーに保存するときに使用します。テキストのヘブライ語とアラビア語の部分は、\$LOGVSw.出力形式がサーバーにない場合には、逆順になります。

比較

\$LOGVSRw.出力形式では、\$LOGVSw.出力形式とは逆の処理を行います。

例

次の例では、ヘブライ語の入力値"החלפה flight"を使用します。

ステートメント	結果
---------	----

ステートメント	結果
	----+----1----+
put text \$logvsr12.;	flight ﻭ ﺋ ﻭ ﻭ ﻭ

次の例では、computer のアラビア語の入力“**ثابت**”を使用します。

ステートメント	結果
	----+----1----+
put text \$logvsr12.;	ثابت コンピューター

関連項目

出力形式:

- [“\\$LOGVSw. 出力形式” \(179 ページ\)](#)

入力形式:

- [“\\$LOGVSw. 入力形式” \(783 ページ\)](#)
- [“\\$LOGVSRw. 入力形式” \(785 ページ\)](#)

MINGUOW. 出力形式

SAS 日付値を台湾の日付として `yyyymmdd` 形式で書き込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

注: [2025.12](#) 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

MINGUOw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 8

範囲 1-10

詳細

MINGUOw.出力形式では、SAS 日付値を *yyyymmdd* 形式で書き込みます。

yyyy

年を表す整数です。

mm

月を表す整数です。

dd

月の日を表す整数です。

台湾のカレンダーでは、1912 年を基準年(1912 年 1 月 1 日が 01/01/01)として使用します。1912 年より前の日付は、一連のアスタリスクで表示されます。年の値は 100 年周期で繰り返されることはありません。年の値は、継続して増加していきます。

例

例の表では、次の入力値を使用します。

- 12054 は、1993 年 1 月 1 日に対応する SAS 日付値です。
- 18993 は、2012 年 1 月 1 日に対応する SAS 日付値です。
- -20088 は、1905 年 1 月 1 日に対応する SAS 日付値です。

ステートメント	結果
x=put(12054,minguo7.); put x=;	x=820101
x=put(12054,minguo9.); put x=;	x=82/01/01
x=put(12054,minguo10.); put x=;	x=0082/01/01

ステートメント	結果
x=put(18993,minguo7.); put x=;	x=1000101
x=put(18993,minguo9.); put x=;	x=100/01/01
x=put(18993,minguo10.); put x=;	x=0101/01/01
x=put(-20088,minguo7.); put x=;	*****
x=put(-20088,minguo9.); put x=;	*****
x=put(-20088,minguo10.); put x=;	*****

関連項目

入力形式:

- [“MINGUOw. 入力形式” \(786 ページ\)](#)

NENGOW. 出力形式

SAS 日付値を日本の日付として *e.yymmdd* 形式で書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NENGOW.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 10

範囲 2-10

詳細

NENGOW.出力形式では、SAS 日付値を *e.yymmdd* 形式で書き込みます。

e

元号(明治、大正、昭和、平成、または令和)の先頭の文字(M、T、S、H、R)です。

yy

年を表す整数です。

mm

月を表す整数です。

dd

月の日を表す整数です。

幅が小さすぎる場合は、ピリオドが省略されます。

例

例の表では、15342 を入力値として使用します。この値は 2002 年 1 月 02 日に相当する SAS 日付値です。

```
data _null_;
  date=15342;
  put date nengo3.;
  put date nengo6.;
  put date nengo8.;
  put date nengo9.;
  put date nengo10.;
run;
```

```
H14
H14/01
H.140102
H14/01/02
H.14/01/02
```

関連項目

入力形式:

■ “NENGOW. 入力形式” (788 ページ)

NLBESTw. 出力形式

ロケールに基づいて最適な数値表記を書き込みます。

カテゴリ: CAS

数値

配置: 右

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLBESTw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 1-32

ヒント 0 から.01 の数字(端点を除く)を出力する場合、過度に丸めないように 7 以上のフィールド幅を使用します。0 から-.01 の数字(端点を除く)を出力する場合は、8 以上のフィールド幅を使用します。

詳細

NLBEST 出力形式は、ロケールの小数点と記号の位置に基づいて、最も適切な数値を書き込みます。NLBEST は BEST 出力形式に似ています。詳細については、*SAS 出力形式と入力形式* リファレンスの BEST 出力形式を参照してください。

例

次に、コードのロケールに基づいた結果を示します。


```

options locale=English_UnitedStates;
data test;
  x=-1257000;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=-0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=1257000;
  put x nlbest6.;
  put x nlbest3.;
run;

```

```

-126E4
***
=====
-0.1
-.1
=====
0.1
0.1
=====
1.26E6
1E6

```

```

options locale=German_Germany;
data test;
  x=-1257000;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=-0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=1257000;
  put x nlbest6.;
  put x nlbest3.;
run;

```

```
-126E4
***
=====
-0,1
-,1
=====
0,1
0,1
=====
1,26E6
1E6
```

```
options locale=ar_BH;
data test;
  x=-1257000;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=-0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=0.1;
  put x nlbest6.;
  put x nlbest3.;
  put "=====";
  x=1257000;
  put x nlbest6.;
  put x nlbest3.;
run;
```

```
-126E4
***
=====
-0.1
-.1
=====
0.1
0.1
=====
1.26E6
1E6
```

NLDw.p 出力形式

値の桁数をできる限り最大にし、同じような大きさの数値に対しては小数点以下の桁数を揃えて出力します。

カテゴリ: CAS
数値

配置: 右

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDw.p

オプション引数

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 1-32

p

有効桁数を指定します。

デフォルト 3

範囲 0-16

要件 p は w より小さい数でなければなりません。

ヒント p が省略される、または 0 が指定されると、 p は 3 に設定されます。

詳細

NLDw.p 出力形式は、ロケールの小数点が同様の大きさの値のグループに揃うように数値を書き込みます。 p の値が大きくなると、出力されるデータ値の有効桁数が増え、ロケールと小数点揃えのシフトが増す可能性があります。 p の値が小さくなると、出力されるデータ値の有効桁数が減り、ロケールと小数点揃えの可能性が高くなります。

例

例 1

これらの例は、NLD 出力形式と D 出力形式の機能を示しています。最初の例では、English_United States ロケールを使用しています。2 番目の例は、German_Germany ロケールを使用しています。

```
options locale=en_US;
data _null_;
  x=.75;
  put 'NLD=<' x nld4.1 '>';
  put 'D= <' x d4.1 '>';
run;
```

```
NLD=< 0.8>
D= < 0.8>
```

例 2

```
options locale=de_DE;
data _null_;
  x=.75;
  put 'NLD=<' x nld4.1 '>';
  put 'D= <' x d4.1 '>';
run;
```

```
NLD=< 0.8>
D= < 0.8>
```

NLDATEw. 出力形式

現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 20

範囲 10-200

比較

NLDATEw.は DATEw.および WORDDATEw.と似ていますが、NLDATEw.がロケール固有である点が違います。

例

これらの例では、15760 を入力値として使用します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。最初の例では、english_United States ロケールを指定しています。2 番目の例は、german_Germany ロケールを指定します。

```
options locale=English_UnitedStates;  
data test;  
  day=15760;  
  put day nldate.;  
run;
```

```
February 24, 2003
```

```
options locale=German_Germany;  
data test;  
  day=15760;  
  put day nldate.;  
run;
```

```
24. Februar 2003
```

関連項目

出力形式:

- “NLDATEMNw. 出力形式” (202 ページ)
- “NLDATEWw. 出力形式” (205 ページ)
- “NLDATEWNw. 出力形式” (207 ページ)

NLDATECPw.p 出力形式

SAS 日付値をロケール依存のコンパクト間隔日付形式に変換します。

カテゴリ: CAS

日付と時間

配置: 左

注: NLDATECP 形式は 2023.03 からサポートされています。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATECP_{w,p}

オプション引数

w

出力フィールドの幅を指定します。この引数はオプションで、デフォルト値の使用をお勧めします。w の値が出力を収容できるほど大きくない場合は、デジタルパターンが使用され、出力にはアスタリスク(*)が含まれます。

デフォルト 20

範囲 2-200

p

コンパクト間隔の種類を指定します。

デフォルト 9

範囲 1-9

詳細

NLDATECPw.p 出力形式は、コンパクトな形式の間隔パターンで日付を書き込みます。

例

例 1

この例では、18972 を入力値として指定します。この値は、2011 年 12 月 11 日に相当する SAS 日付値です。最初の例では、en_US ロケールを指定します。2 番目の例では、fr_FR ロケールを指定します。3 番目の例では、ja_JP ロケールを指定します。4 番目の例は、デジタルパターン出力を示します。

```
option locale=en_US;
data _null_;
  dt = 18972;
  put dt nldatecp.1;
  put dt nldatecp.2;
  put dt nldatecp.3;
  put dt nldatecp.4;
  put dt nldatecp.5;
  put dt nldatecp.6;
  put dt nldatecp.7;
  put dt nldatecp.8;
  put dt nldatecp.9;
run;
```

```
11
Dec
11
2011
11Dec
Dec11
Dec2011
11Dec11
11Dec2011
```

例 2

```
option locale=fr_FR;
data _null_;
  dt = 18972;
  put dt nldatecp.1;
  put dt nldatecp.2;
  put dt nldatecp.3;
  put dt nldatecp.4;
  put dt nldatecp.5;
```

```

put dt nldatecp.6;
put dt nldatecp.7;
put dt nldatecp.8;
put dt nldatecp.9;
run;

```

```

11
déc.
11
2011
11déc.
déc.11
déc.2011
11déc.11
11déc.2011

```

例 3

```

option locale=ja_JP;
data _null_;
  dt = 18972;
  put dt nldatecp.1;
  put dt nldatecp.2;
  put dt nldatecp.3;
  put dt nldatecp.4;
  put dt nldatecp.5;
  put dt nldatecp.6;
  put dt nldatecp.7;
  put dt nldatecp.8;
  put dt nldatecp.9;
run;

```

```

11 日
w12 月
11 年
2011 年
12 月 11 日
11 年 12 月
2011 年 12 月
11 年 12 月 11 日
2011 年 12 月 11 日

```

例 4

この例は、*w* の値が出力を収容できるほど大きくない場合に、デジタルパターンが使用されることを示しています。なお、一部の出力行にはアスタリスク(*)が出力されます。

```

option locale=en_US;
data _null_;
  dt = 18972;
  put dt nldatecp2.9;

```



```

put dt nldatecp3.9;
put dt nldatecp4.9;
put dt nldatecp5.9;
put dt nldatecp6.9;
put dt nldatecp7.9;
put dt nldatecp8.9;
put dt nldatecp9.9;
run;

```

これらのステートメントは、次の結果を生成します。出力の最初の 6 行にはアスタリスク(*)が表示されることに注意してください。これらの出力行は、値 *w* が出力を収容できるほど大きくない場合の結果です。

```

**
***
****
*****
*****
*****
*****
11122011
11Dec2011

```

関連項目

[“NLDATMCPw.p 出力形式”](#)

NLDATELw. 出力形式

SAS の日付値を、現在の SAS ロケールに適した月、日付、年の形式で日付として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATELw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 18

範囲 2-200

詳細

NLDATEL では、長い形式で月と週の名前を含む日付が出力されます。

例

この例では、日付として 2012 年 11 月 19 日を使用します。

```
data _null_;
  dt = dt='19Nov2012:00:00:00'dt;
  dy='19Nov2012'd;
  put "+--- NLDATEL min=2 default=18
max=200 ---+";
  put dy nldatel.;
  put dy nldatel10.;
  put dy nldatel12.;
  put dy nldatel18.;
  put dy nldatel200.;
run;
```

```
+--- NLDATEL min=2 default=18max=200 ---+
November 19, 2012
11/19/2012
Nov 19, 2012
November 19, 2012
November 19, 2012
```

NLDATEMw. 出力形式

SAS 日付値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で日付として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEM_w

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 14

範囲 2-200

詳細

NLDATEM では、Nov 19, 2012 など、簡略形式で日付が出力されます。

例

この例では、日付として Nov 19, 2012 を指定します。

```
data _null_;
  dt = dt='19Nov2012:00:00:00'dt;
  dy='19Nov2012'd;
  put '+--- NLDATEM min=2 default=14
    max=200 ---+';
  put dy nldatem.;
  put dy nldatem8.;
  put dy nldatem14.;
  put dy nldatem200.;
run;
```

```
+--- NLDATEM min=2 default=14 max=200 ---+
Nov 19, 2012
11/19/12
Nov 19, 2012
Nov 19, 2012
```

NLDATEMDw. 出力形式

SAS の日付値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEMDw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 6-200

例

この例では、english_United States ロケールオプションを使用します。

```
option locale=en_US;  
data _null_;  
  dy = date();  
  put dy nldatemd.;  
run;
```

```
April 03
```

関連項目

出力形式:

NLDATMDLw. 出力形式

現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMDLw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 12

範囲 5-200

詳細

NLDATMDL では、November 19 など、長い形式で月と日を含む日付が出力されません。

例

次の例では、日付として 2012 年 11 月 19 日を指定します。

```
data _null_;  
  dt = dt='19Nov2012:00:00:00'dt;  
  dy='19Nov2012'd;  
  put '+--- NLDATMDL min=5 default=12
```

```

max=200 ---+';
put dy nldatemd1.;
put dy nldatemd15.;
put dy nldatemd19.;
put dy nldatemd12.;
put dy nldatemd1200.;
run;

```

```

+--- NLDATEMDL min=5 default=12max=200 ---+
November 19
11/19
Nov 19
November 19
November 19

```

NLDATEMDM_w. 出力形式

現在の SAS ロケールに適した月と日として SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEMDM_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 9

範囲 5-200

詳細

NLDATEMDM では、Nov 19 など、短縮形を使用した簡略形式で月と日を含む日付が出力されます。月名が数字に短縮される場合は、区切り文字が使用されます。

例

次の例では、日付として 2012 年 11 月 19 日を指定します。

```
data _null_;  
  dt = dt='19Nov2012:00:00:00'dt;  
  dy='19Nov2012'd;  
  put '+--- NLDATEMDL min=5 default=9  
    max=200 ---+';  
  put dy nldatemdmdm.;  
  put dy nldatemdmdm5.;  
  put dy nldatemdmdm9.;  
  put dy nldatemdmdm200.;  
run;
```

```
+--- NLDATEMDL min=5 default=9max=200 ---+  
Nov 19  
11/19  
Nov 19  
Nov 19
```

NLDATEMDSw. 出力形式

SAS 日付値を、現在の SAS ロケールに適した短い形式で日付として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEMDSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 5

範囲 5-200

詳細

NLDATEMDS では、MM/dd など、数字と区切り文字を使用した短い形式で月と日を含む日付が出力されます。

例

次の例では、日付として 2012 年 11 月 19 日を指定します。

```
data _null_;
  dt = dt='19Nov2012:00:00:00'dt;
  dy='19Nov2012'd;
  put '+--- NLDATEMDL min=5 default=5
max=200 ---+';
  put dy nldatemds.;
  put dy nldatemds5.;
  put dy nldatemds5.;
  put dy nldatemds200.;
run;
```

```
+--- NLDATEMDL min=5 default=5max=200 ---+
11/19
11/19
11/19
11/19
```

NLDATEMNw. 出力形式

SAS 日付値を現在の SAS ロケールに適した月の名前として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEMN_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように月名が短縮されます。

デフォルト 9

範囲 4-200

詳細

NLDATEMN は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで値を書き込みます。

この例は、均一なパターンを示しています。この例では UTF-8 セッションを使用しており、幅の値は両方のステートメントで 6 です。

```
option locale = zh_cn;
data test;
  x = '08jun91'd;
  put x nldatmn6.;
  = '08sep91'd;
  put x nldatmn6.;
run;
```

6
12

比較

NLDATEMN_w.は MONNAMEMONNAME_w と似ていますが、NLDATEMN_w.がローケル固有である点が違います。

例

これらの例では、15760 を入力値として指定します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。最初の例では、en_US ロケールを指定します。2 番目の例は、german_Germany ロケールを指定します。

```
option locale=en_US;
data _null_;
  month=15760;
  put month nldatmn.;
run;
```

February

```
option locale=german_germany;
data _null_;
  month=15760;
  put month nldatmn.;
run;
```

Februar

関連項目

出力形式:

- [“NLDATEw. 出力形式” \(190 ページ\)](#)
- [“NLDATEWw. 出力形式” \(205 ページ\)](#)
- [“NLDATEWNw. 出力形式” \(207 ページ\)](#)

NLDATEsw. 出力形式

SAS 日付値を、現在の SAS ロケールに適した日付文字列として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: [2025.12](#) 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATES_w.

構文の説明

_w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 10

範囲 2-200

詳細

NLDATES では、mm/dd/yyyy など、数字と区切り文字のみ使用した短い形式で日付が出力されます。

例

この例では、日付として 2012 年 11 月 19 日を指定します。この例では、en_US ロケールを指定します。

```
option locale=en_US;
data _null_;
  dt = dt='19Nov2012:00:00:00'dt;
  dy='19Nov2012'd;
  put '+--- NLDATEMDL min=2 default=10
max=200 ---+';
  put dy nldates.;
  put dy nldates8.;
  put dy nldates10.;
  put dy nldates200.;;
run;
```

```
+--- NLDATEMDL min=2 default=10max=200 ---+
11/19/2012
11/19/12
11/19/2012
11/19/2012
```

NLDATEWw. 出力形式

SAS 日付値を、現在の SAS ロケールに適した日付と曜日として書き込みます。

カテゴリ:	CAS 日付と時間
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEW_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付と曜日が短縮されます。

デフォルト 29

範囲 10-200

詳細

エンコーディングとロケールの組み合わせによっては日時名が長くなるため、デフォルトの幅を使用した NLDATEW 出力形式では正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。推奨される幅を使用するようにします。

NLDATEW は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで日付値を書き込みます。

比較

NLDATEW_w は WEEKDATE_w に似ていますが、NLDATEW_w がロケール固有である点が違います。

例

これらの例では、15760 を入力値として使用します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。最初の例では、en_US ロケールを指定します。2 番目の例は、de_DE ロケールを指定します。

```
options locale=en_US;
data _null_;
dy=15760;
put dy nldatew.;
put dy nldatew20.;
put dy nldatew200.;
run;
```

```
Monday, February 24, 2003
Mon, Feb 24, 2003
Monday, February 24, 2003
```

```
options locale=de_DE;
data _null_;
dy=15760;
put dy nldatew.;
put dy nldatew20.;
put dy nldatew200.;
run;
```

```
Mo., 24. Feb 2003
Mo., 24. Feb 2003
Montag, 24. Februar 2003
```

関連項目

出力形式:

- [“NLDATEw. 出力形式” \(190 ページ\)](#)
- [“NLDATEMNw. 出力形式” \(202 ページ\)](#)
- [“NLDATEWNw. 出力形式” \(207 ページ\)](#)

NLDATEWNw. 出力形式

現在の SAS ロケールに適した曜日で SAS 日付値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEWN_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように曜日が短縮されます。

デフォルト 9

範囲 4-200

詳細

NLDATEWN は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで日付値を書き込みます。

比較

NLDATEWN_w. は DOWNAME_w. と似ていますが、NLDATEWN_w. がロケール固有である点が違います。

例

これらの例では、15760 を入力値として使用します。この値は、2003 年 2 月 24 日に相当する SAS 日付値です。最初の例では、en_US ロケールを指定します。2 番目の例は、de_DE ロケールを指定します。

```
options locale=en_US;
data _null_;
  dy=15760;
  put dy nldatewn.;
run;
```

Monday

```
options locale=de_DE;
data _null_;
  dy=15760;
  put dy nldatewn10.;
run;
```

Montag

関連項目

出力形式:

- “NLDATEw. 出力形式” (190 ページ)
- “NLDATEMNw. 出力形式” (202 ページ)
- “NLDATEWw. 出力形式” (205 ページ)

NLDATEYMw. 出力形式

SAS の日付値を、現在の SAS ロケールに適した年と月の名前として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYM*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 6-200

詳細

データが指定した幅の 6 より大きい場合、出力には次のようにアスタリスクが含まれます: d=*****。アスタリスクを削除するには PROC LOCALEDATA を使用します。次の例では PROC LOCALEDATA を使用してアスタリスクなしで日付を出力しています。

```
PROC LOCALEDATA;  
  LOAD SASLOCALE;  
  MODIFY key=DATE_YYMM_SHORT_FORMAT value='%b %y';  
  SAVE REGISTRY / _ALL_ syntax=SAS;
```

```
data _null_;  
  format d nldateym6.;  
  d = '17OCT14'd;  
  put d=;  
run;
```

```
d=Oct 14
```

例

この例では、spanish_Spain ロケールオプションを指定します。この例では、日付として 2019 年 04 月 04 日を指定します。

```
options locale=spanish_Spain;  
data _null_;  
  dy=today();  
  put dy nldateym.;  
  put dy nldateym12.;  
  put dy nldateym200.;  
run;
```

```
abril de 2019  
abr de 2019  
abril de 2019
```

関連項目

出力形式:

- [“NLDATEMDw. 出力形式” \(198 ページ\)](#)

NLDATEYMLw. 出力形式

現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYMLw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 14

範囲 5-200

詳細

NLDATEYML では、April 2019 など、長い形式で年と月を含む日付が出力されます。出力形式の幅に合わせて短縮形が使用されることもあります。

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、en_US ロケールを指定します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put "+--- NLDATEYML min=5 default=14 max=200 ---+";
```

```
put dy nldateyml;
put dy nldateyml7;
put dy nldateyml11;
put dy nldateyml14;
put dy nldateyml200;
run;
```

```
+--- NLDATEYML min=5 default=14 max=200 ---+
April 2019
04/2019
Apr 2019
April 2019
April 2019
```

NLDATEYMMw. 出力形式

短縮形を使用して、SAS 日付値を現在の SAS ロケールに適した月と年として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYMMw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 11

範囲 5-200

詳細

NLDATEYMS では、Apr 2019 など、短縮形を使用した簡略形式で月と年を含む日付が出力されます。

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、en_US ロケールを指定します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put "+++ NLDATEYML min=5 default=14 max=200 ---+";
  put dy nldatymm.;
  put dy nldatymm7.;
  put dy nldatymm11.;
  put dy nldatymm200.;
run;
```

```
+++ NLDATEYML min=5 default=14 max=200 ---+
Apr 2019
04/2019
Apr 2019
Apr 2019
```

NLDATEYMSw. 出力形式

現在の SAS ロケールに適した日付として SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYMSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 7

範囲 5-200

詳細

NLDATEYMS では、mm/yyy など、数字と区切り文字を使用した短い形式で年と月を含む日付が出力されます。

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、en_US ロケールを指定します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put "+--- NLDATEYMS min=5 default=14 max=200 ---+";
  put dy nldateyms.;
  put dy nldateyms5.;
  put dy nldateyms7.;
  put dy nldateyms200.;
run;
```

```
+--- NLDATEYMS min=5 default=14 max=200 ---+
04/2019
04/19
04/2019
04/2019
```

NLDATEYQw. 出力形式

現在の SAS ロケールに適した年と四半期として SAS 日付値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注:

2025.12以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYQ_w.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 4-200

詳細

エンコーディングとロケールの組み合わせによっては日時名が長くなるため、デフォルトの幅を使用した NLDATEYQ 出力形式では正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。推奨される幅を使用するようにします。

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、fr_FR ロケールオプションを指定します。

```
options locale=fr_FR;
data _null_;
  dy=today();
  dt=datetime();
  put "+++ NLDATEYQ min=4 default=16 max=200 +++";
  put '16' +5 dy nldateyq.;
  put '4' +5 dy nldateyq4.;
  put '14' +5 dy nldateyq14.;
  put '32' +5 dy nldateyq32.;
  put '200' +5 dy nldateyq200.;
run;
```

```

+--- NDATEYQ min=4 default=16 max=200 ---+
16      T2 2019
4       ****
14      T2 2019
32      2e trimestre 2019
200
2e trimestre 2019

```

NLDATEYQLw. 出力形式

短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1～Q4)として、SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYQLw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 18

範囲 4-200

詳細

NLDATEYQL では、2nd quarter 2019 など、長い形式で年と四半期を含む日付が出力されます。

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、en_US ロケールオプションを使用します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATEYQL min=4 default=18 max=200 ---+';
  put dy nldateyql;
  put dy nldateyql6;
  put dy nldateyql7;
  put dy nldateyql18;
  put dy nldateyql200;
run;
```

```
+--- NLDATEYQL min=4 default=18 max=200 ---+
2nd quarter 2019
2019/2
Q2 2019
2nd quarter 2019
2nd quarter 2019
```

NLDATEYQMw. 出力形式

短縮形を使用して、現在の SAS ロケールに適した年とその年の四半期値(Q1~Q4)として、SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYQMw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト	7
範囲	4-200

詳細

NLDATEYQM では、Q2 2019 など、短縮形を使用した簡略形式で年と四半期を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 04 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATEYQM min=4 default=7 max=200 ---+';
  put dy nldateyqm.;
  put dy nldateyqm6.;
  put dy nldateyqm7.;
  put dy nldateyqm200.;
run;
```

```
+--- NLDATEYQM min=4 default=7 max=200 ---+
Q2 2019
2019/2
Q2 2019
Q2 2019
```

NLDATEYQSw. 出力形式

数字と区切り文字を使用して、現在の SAS ロケールに適した年とその年の四半期値(1~4)として、SAS 日付値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYQS_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 6

範囲 4-200

詳細

NLDATEYQS では、2019/2 など、数字と区切り文字を使用した短い形式で年と四半期を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 04 日を指定します。

```
data _null_;  
  dt = datetime();  
  dy = date();  
  put '+--- NLDATEYQS min=4 default=6 max=200 ---+';  
  put dy nldateyqs;  
  put dy nldateyqs4;  
  put dy nldateyqs6;  
  put dy nldateyqs200;  
run;
```

```
+--- NLDATEYQS min=4 default=6 max=200 ---+  
2019/2  
19/2  
2019/2  
2019/2
```

NLDATEYRw. 出力形式

現在の SAS ロケールに適した年として SAS 日付値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYR_w.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 2-200

例

この例では、fr_FR ロケールオプションを指定します。この例では、日付として 2019 年 04 月 04 日を指定します。

```
options locale=fr_FR;
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATEYR min=4 default=6 max=200 ---+';
  put dy nldateyr.;
  put dy nldateyr2.;
  put dy nldateyr8.;
  put dy nldateyr200.;
run;
```

```
+--- NLDATEYR min=4 default=6 max=200 ---+
2019
19
2019
2019
```

NLDATEYW_w. 出力形式

現在の SAS ロケールに適した年と週として SAS 日付値を書き込みます。

カテゴリ:	CAS
	日付と時間
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEYWw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 5-200

例

この例では、日付として 2019 年 04 月 04 日を指定します。この例では、fr_FR ロケールオプションを指定します。

```
options locale=fr_FR;
data _null_;
  dt = datetime();
  dy = date();
  put "+--- NLDATEYW min=5 default=16 max=200 ---+";
  put ' 16' +5 dy nldateyw.;
  put ' 5' +5 dy nldateyw5.;
  put ' 8' +5 dy nldateyw8.;
  put ' 32' +5 dy nldateyw32.;
  put '200' +5 dy nldateyw200.;
run;
```

```
+--- NLDATEYW min=5 default=16 max=200 ---+
16      Semaine 14 2019
5       *****
8       S14 19
32      Semaine 14 2019
200
Semaine 14 2019
```

NLDATEMw. 出力形式

現在の SAS ロケールに適した日時として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEM_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日時値が短縮されます。

デフォルト 30

範囲 10-200

比較

NLDATEMw.出力形式は DATETIMEw.出力形式と似ていますが NLDATEMw.出力形式がロケール固有である点が違います。

例

これらの例では、1361709583 を入力値として指定します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。最初の例では、en_US ロケールを指定します。2 番目の例は、de_DE ロケールを指定します。

```
options locale=en_US;  
data _null_;  
  dy=1361709583;  
  put dy nldatm.;
```

```
run;
```

```
24Feb2003:12:39:43
```

```
options locale=de_DE;
data _null_;
  dy=1361709583;
  put dy nldatm.;
run;
```

```
24. Februar 2003 12.39 Uhr
```

関連項目

出力形式:

- [“NLDATMAPw. 出力形式” \(223 ページ\)](#)
- [“NLDATMTMw. 出力形式” \(239 ページ\)](#)
- [“NLDATMWw. 出力形式” \(242 ページ\)](#)

NLDATMAPw. 出力形式

現在の SAS ロケールに適した午前または午後付きの日時として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMAPw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日時値が短縮されます。

デフォルト	32
範囲	16-200

比較

NLDATMAPw.出力形式は DATEAMPMw.と似ていますが、NLDATMAPw.出力形式がロケール固有である点が違います。

例

これらの例では、1361709583 を入力値として指定します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。最初の例では、en_US ロケールを指定します。2 番目の例は、es_MX ロケールを指定します。

```
options locale=en_US;
data _null_;
dy=1361709583;
put dy nldatmap.;
run;
```

```
February 24, 2003 12:39:43 PM
```

```
options locale=es_MX;
data _null_;
dy=1361709583;
put dy nldatmap200.;
run;
```

```
24 de febrero de 2003 12:39:43 p.m.
```

関連項目

出力形式:

- [“NLDATMw. 出力形式” \(222 ページ\)](#)
- [“NLDATMTMw. 出力形式” \(239 ページ\)](#)
- [“NLDATMWw. 出力形式” \(242 ページ\)](#)

NLDATMCPw.p 出力形式

SAS 日時値をロケール依存のコンパクト間隔日時形式に変換します。

カテゴリ: CAS
日付と時間

配置: 左

注: NLDATMCP 形式は 2023.03 からサポートされています。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMCPw.p

オプション引数

w
出力フィールドの幅を指定します。この引数はオプションで、デフォルト値の使用をお勧めします。w の値が出力を収容できるほど大きくない場合は、デジタルパターンが使用され、出力にはアスタリスク(*)が含まれます。

デフォルト 33

範囲 2-200

p
コンパクト間隔の種類を指定します。

デフォルト 12

範囲 1-12

詳細

NLDATMCPw.p 出力形式は、コンパクトな間隔パターンで日時値を書き込みます。

例

これらの例では、1361709583 を入力値として指定します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。最初の例では、en_US ロケールを指定します。2 番目の例では、fr_FR ロケールを指定します。3 番目の例では、ja_JP ロケールを指定します。4 番目の例は、デジタルパターン出力を示します。

例 1

```
option locale=en_US;
data _null_;
  dt = 1361709583;
  put dt nldatmcp.1;
  put dt nldatmcp.2;
  put dt nldatmcp.3;
  put dt nldatmcp.4;
  put dt nldatmcp.5;
  put dt nldatmcp.6;
  put dt nldatmcp.7;
  put dt nldatmcp.8;
  put dt nldatmcp.9;
  put dt nldatmcp.10;
  put dt nldatmcp.11;
  put dt nldatmcp.12;
run;
```

```
24
Feb
03
2003
24Feb
Feb03
Feb2003
24Feb03
24Feb2003
24Feb2003 12
24Feb2003 12:39
24Feb2003 12:39:43
```

例 2

```
option locale=fr_FR;
data _null_;
  dt = 1361709583;
  put dt nldatmcp.1;
  put dt nldatmcp.2;
  put dt nldatmcp.3;
  put dt nldatmcp.4;
  put dt nldatmcp.5;
  put dt nldatmcp.6;
  put dt nldatmcp.7;
```



```

put dt nldatmcp.8;
put dt nldatmcp.9;
put dt nldatmcp.10;
put dt nldatmcp.11;
put dt nldatmcp.12;
run;

```

```

24
févr.
03
2003
24févr.
févr.03
févr.2003
24févr.03
24févr.2003
24févr.2003 12h
24févr.2003 12:39
24févr.2003 12:39:43

```

例 3

```

option locale=ja_JP;
data _null_;
  dt = 1361709583;
  put dt nldatmcp.1;
  put dt nldatmcp.2;
  put dt nldatmcp.3;
  put dt nldatmcp.4;
  put dt nldatmcp.5;
  put dt nldatmcp.6;
  put dt nldatmcp.7;
  put dt nldatmcp.8;
  put dt nldatmcp.9;
  put dt nldatmcp.10;
  put dt nldatmcp.11;
  put dt nldatmcp.12;
run;

```

```

24 日
2 月
03 年
2003 年
2 月 24 日
03 年 2 月
2003 年 2 月
03 年 2 月 24 日
2003 年 2 月 24 日
2003 年 2 月 24 日 12 時
2003 年 2 月 24 日 12:39
2003 年 2 月 24 日 12:39:43

```

NLDATMDTw. 出力形式

SAS 日時値を現在の SAS ロケールに適した月の名前、月の何日目、および年として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMDTw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 18

範囲 10-200

詳細

NLDATMDT は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで値を書き込みます。

例

この例では、1960 年 1 月 2 日に対応する値 86400 を指定します。最初の例では、english_United States ロケールを指定しています。2 番目の例は、turkish_Turkey ロケールを指定します。

```
options locale=en_US;  
data _null_;  
  x=86400;  
  put x nldatmdt.;  
run;
```

January 2, 1960

```
options locale=tr_TR;
data _null_;
x=86400;
put x nldatmdt200.;
run;
```

2 Ocak 1960

関連項目

出力形式:

- [“NLDATMMDw. 出力形式” \(232 ページ\)](#)

NLDATMLw. 出力形式

SAS 日時値を現在の SAS ロケールに適した長い形式の日付として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATML*w*.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 30

範囲 9-200

詳細

NLDTML では、April 10, 2019 03:13:27 PM など、長い形式で月、日、年、時間を含む日付が出力されます。

例

```
data _null_;  
  dt = datetime();  
  dy = date();  
  put "+--- NLDTML min=9 default=30  
max=200 ---+";  
  put dt nldatml.;  
  put dt nldatml19.;  
  put dt nldatml24.;  
  put dt nldatml30.;  
  put dt nldatml200.;  
run;
```

```
+--- NLDTML min=9 default=30max=200 ---+  
April 10, 2019 03:13:27 PM  
04/10/2019 15:13:27  
Apr 10, 2019 03:13:27 PM  
April 10, 2019 03:13:27 PM  
April 10, 2019 03:13:27 PM
```

NLDTMMw. 出力形式

現在の SAS ロケールに適した中間の形式の日付で SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDTMMw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 24

範囲 9-200

詳細

NLDATMM では、Apr 10, 2019 03:23:34 PM など、短縮形を使用した簡略形式で月、日、年、時間を含む日時が出力されます。

例

```
data _null_;
  dt = datetime();
  dy = date();
  put '---- NLDATMM min=9 default=30
max=200 ---+';
  put dt NLDATMM.;
  put dt NLDATMM19.;
  put dt NLDATMM24.;
  put dt NLDATMM30.;
  put dt NLDATMM200.;
run;
```

```
+--- NLDATMM min=9 default=30max=200 ---+
Apr 10, 2019 03:23:34 PM
04/10/2019 15:23:34
Apr 10, 2019 03:23:34 PM
Apr 10, 2019 03:23:34 PM
Apr 10, 2019 03:23:34 PM
```

NLDATMMDw. 出力形式

SAS 日時値を現在の SAS ロケールに適した月の名前と月の何日目かとして書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMMD*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 6-200

例

この例では、en_US ロケールオプションを使用します。

```
options locale=en_US;
data _null_;
  x=put(86400,nldatmmd.);
  put x=;
run;
```

```
x=January 02
```

関連項目

出力形式:

- [“NLDATMYMw. 出力形式” \(246 ページ\)](#)

NLDATMMDLw. 出力形式

SAS 日時値を現在の SAS ロケールに適した長い形式の月と月の何日目かとして書き込みます。

カテゴリ: CAS
日付と時間
配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMMDL_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 12

範囲 5-200

詳細

NLDATMMDL では、November 19 など、長い形式で月と日を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 12 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '---- NLDATMMDL min=5 default=12
max=200 ---+';
  put dt nldatmmdl;
  put dt nldatmmdl5;
  put dt nldatmmdl9;
  put dt nldatmmdl12;
  put dt nldatmmdl200;
run;
```

```
+--- NLDATMMDL min=5 default=12max=200 ---+
April 12
04/12
Apr 12
April 12
April 12
```

NLDATMMDMw. 出力形式

短縮形を使って、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMMDMw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 9

範囲 5-200

詳細

NLDATMMDM では、Nov 19 など、短縮形を使用した簡略形式で月と日を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 12 日を使用します。

```
data _null_;  
  dt = datetime();  
  dy = date();  
  put '---- NLDATMMDM min=5 default=9  
max=200 ---+';  
  put dt nldatmmdm.;
```

```
put dt nldatmmdm5.;
put dt nldatmmdm9.;
put dt nldatmmdm200.;
run;
```

```
+--- NLDATMMDM min=5 default=9max=200 ---+
Apr 12
04/12
Apr 12
Apr 12
```

NLDATMMDSw. 出力形式

数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と月の何日目かとして書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMMDSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 5

範囲 5-200

詳細

NLDATMMDS では、4/12 など、数字と区切り文字を使用した短い形式で月と日を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 12 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '---- NLDATMMDS min=5 default=5 max=200 ----';
  put dt nldatmmds.;
  put dt nldatmmds5.;
  put dt nldatmmds5.;
  put dt nldatmmds200.;
run;
```

```
---- NLDATMMDS min=5 default=5max=200 ----
04/12
04/12
04/12
04/12
```

NLDATMMNw. 出力形式

SAS 日時値を現在の SAS ロケールに適した月の名前として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMMN*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 4-200

詳細

NLDATMMN は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで値を書き込みます。

例

この例では、en_US ロケールオプションを使用します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put '---- NLDATMMN min=5 default=10 max=200 ----';
  put dt nldatmmn.;
  put dt nldatmmn4.;
  put dt nldatmmn10.;
  put dt nldatmmn200.;
run;
```

```
---- NLDATMMN min=5 default=10max=200 ----+
April
Apr
April
April
```

NLDATMSw. 出力形式

現在の SAS ロケールに適した短い形式の日付で SAS 日時値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 19

範囲 9-200

詳細

NLDATMS では、MM/dd/yyyy hh:mm:ss など、数字と区切り文字を使用した短い形式で月、日、年、時間を含む日付が出力されます。

例

この例では、日付として 2019 年 04 月 12 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '---- NLDATMS min=9 default=19 max=200---+';
  put dt nldatms;
  put dt nldatms10;
  put dt nldatms19;
  put dt nldatms200;
run;
```

```
+--- NLDATMS min=9 default=19 max=200---+
04/12/2019 11:50:53
041219 11
04/12/2019 11:50:53
04/12/2019 11:50:53
```

NLDATMTMw. 出力形式

現在の SAS ロケールに適した時刻で SAS 日時値の時刻部分を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMTM_w.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 16-200

比較

NLDATMTM_w.出力形式は TOD_w.出力形式と似ていますが NLDATMTM_w.出力形式がロケール固有である点が違います。

例

これらの例では、1361709583 を入力値として使用します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。

この例では、English_United_States ロケールを指定します。

```
options locale=en_US;
data one;
  event=1361709583;
  put event nldatmtm.;
run;
```

```
12:39:43
```

この例では、German_Germany ロケールを指定します。

```
options locale=de_DE;
data one;
  event=1361709583;
  put event nldatmtm.;
run;
```

```
12.39 Uhr
```

関連項目

出力形式:

- “NLDATMw. 出力形式” (222 ページ)
- “NLDATMAPw. 出力形式” (223 ページ)
- “NLDATMWw. 出力形式” (242 ページ)

NLDATMTZw. 出力形式

SAS 日時値の時刻部分を現在の SAS ロケールに適した時刻とタイムゾーンとして書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMTZw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 32

範囲 16-200

例

この例は現在の日時値を使用します。この例では、French_France ロケールを指定します。

```
options locale=fr_FR;
data test;
  x=datetime();
  put x=nldatmtz.;
```

```
run;
```

```
x=10 h 08 -0400
```

NLDATMWw. 出力形式

SAS 日時値を現在の SAS ロケール適した曜日と日時として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMWw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように曜日と日時が短縮されます。

デフォルト 41

範囲 16-200

詳細

エンコーディングとロケールの組み合わせによっては日時名が長くなるため、デフォルトの幅を使用した NLDATMW 出力形式では正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。NLDATMW 出力形式の推奨される幅を使用するようにします。

NLDATMW は、同じフォーマット幅で異なる入力データを使用する場合に出力が同じパターンになるように、均一なパターンで値を書き込みます。

比較

NLDTMWNw.出力形式は TWMDYw.出力形式と似ていますが NLDTMWNw.出力形式がロケール固有である点が違います。

例

この例では、1361709583 を入力値として使用します。この値は、2003 年 2 月 24 日の午後 12:39:43 に相当する SAS 日時値です。この例では、English_United States ロケールを使用します。

```
options locale=English_UnitedStates;
data one;
  x=put(1361709583,nldatmw.);
  y=put(1361709583,nldatmw30.);
  z=put(1361709583,nldatmw200.);
  put x=;
  put y=;
  put z=;
run;
```

```
x=Monday, February 24, 2003 12:39:43 PM
y=Mon, Feb 24, 2003 12:39:43 PM
z=Monday, February 24, 2003 12:39:43 PM
```

関連項目

出力形式:

- [“NLDTMw. 出力形式” \(222 ページ\)](#)
- [“NLDTMAPw. 出力形式” \(223 ページ\)](#)
- [“NLDTMTMw. 出力形式” \(239 ページ\)](#)

NLDTMWNw. 出力形式

現在の SAS ロケールに適した曜日の形式で SAS 日時値を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLDATMWN*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 4-200

詳細

NLDATMWN は、同じフォーマット幅で異なる入力データを使用する場合に出力が
同じパターンになるように、均一なパターンで値を書き込みます。

例

SAS 日時値から曜日を書き込みます。この例では、デフォルトの SAS セッションエ
ンコーディングである English_United States ロケールを指定します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATMWN min=4 default=9 max=200 ---+';
  put dt nldatmwn;
  put dt nldatmwn4.;
  put dt nldatmwn9.;
  put dt nldatmwn200.;
run;
```

```
+--- NLDATMWN min=4 default=9 max=200 ---+
Tuesday
Tue
Tuesday
Tuesday
```

NLDATMWZw. 出力形式

SAS 日時値を現在の SAS ロケールに適した曜日、日時、およびタイムゾーン値として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMWZ*w*.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように曜日と日時が短縮されます。

デフォルト 40

範囲 16-200

詳細

エンコーディングとロケールの組み合わせによっては日時名が長くなるため、デフォルトの幅を使用した NLDATMWZ 出力形式では正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。推奨される幅を使用するようにします。

例

この例は現在の日時値を使用します。この例では、French_France ロケールを指定します。

```
options locale=fr_FR;  
data test;
```

```
x=datetime();
put x=nldatmwz.;
run;
```

```
x=mardi 16 avril 2019 10:35:06 -0400
```

NLDATMYMw. 出力形式

現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYMw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 6-200

例

この例では、en_US ロケールオプションを使用します。

```
options locale=en_US;
data _null_;
  x=put(86400,nldatmym.);
  y=put(86400,nldatmym12.);
  put x=;
  put y=;
run;
```

```
x=January 1960  
y=January 1960
```

関連項目

出力形式:

- [“NLDATMMDw. 出力形式” \(232 ページ\)](#)

NLDATMYMLw. 出力形式

現在の SAS ロケールに適した月と年として SAS 日付値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYMLw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 14

範囲 5-200

詳細

NLDATMYML では、November 2012 など、長い形式で月と年を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。

```
data _null_;
  dt = datetime();
  put '+--- NLDATMYML min=5 default=14 max=200 ---+';
  put dt nldatmyml;
  put dt nldatmyml7.;
  put dt nldatmyml11.;
  put dt nldatmyml14.;
  put dt nldatmyml200.;
run;
```

```
+--- NLDATMYML min=5 default=14max=200 ---+
April 2019
04/2019
Apr 2019
April 2019
April 2019
```

NLDATMYMMw. 出力形式

現在の SAS ロケールに適した月と年として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYMMw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 11

範囲 5-200

詳細

NLDATMYMM では、Nov 2012 など、簡略形式で月と年を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。

```
data _null_;  
  dt = datetime();  
  dy = date();  
  put '+--- NLDATMYMM min=5 default=11 max=200 ---+';  
  put dt nldatmymm.;  
  put dt nldatmymm7.;  
  put dt nldatmymm11.;  
  put dt nldatmymm200.;  
run;
```

```
+--- NLDATMYMM min=5 default=11 max=200 ---+  
Apr 2019  
04/2019  
Apr 2019  
Apr 2019
```

NLDATMYMSw. 出力形式

数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した月と年として、書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYMSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 7

範囲 5-200

詳細

NLDATMYMS では、11/2012 など、数字と区切り文字を使用した短い形式で月と年を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATMYMS min=5 default=7 max=200 ---+';
  put dt nldatmyms.;
  put dt nldatmyms5.;
  put dt nldatmyms7.;
  put dt nldatmyms200.;
run;
```

```
+--- NLDATMYMS min=5 default=7 max=200 ---+
04/2019
04/19
04/2019
04/2019
```

NLDATMYQw. 出力形式

現在の SAS ロケールに適した四半期と年として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYQ*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 4-200

詳細

エンコーディングとロケールの組み合わせによっては日時名が長くなるため、デフォルトの幅を使用した NLDATMYQ 出力形式では正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。推奨される幅を使用するようにします。

例

この例では、en_US ロケールオプションを使用します。

```
options locale=en_US;
data _null_;
  dy=today();
  dt=datetime();
  put "+--- NLDATMYQ min=4 default=16 max=200 ---+";
  put ' 16' +5 dt nldatmyq.;
  put ' 4' +5 dt nldatmyq4.;
  put ' 14' +5 dt nldatmyq14.;
  put ' 32' +5 dt nldatmyq32.;
  put '200' +5 dt nldatmyq200.;
run;
```

```
+--- NLDATMYQ min=4 default=16 max=200 ---+
 16      2nd quarter 2019
 4       ****
 14      Q2 2019
 32      2nd quarter 2019
200
2nd quarter 2019
```

NLDATMYQLw. 出力形式

SAS 日時値を現在の SAS ロケールに適したその年の四半期値(1~4)と年として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYQLw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 18

範囲 4-200

詳細

NLDATMYQL では、4th quarter 2012 など、長い形式で四半期、次に年を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。この例では、en_US ロケールを指定します。

```
options locale=en_US;
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATMYQL min=4 default=18 max=200 ---+';
```

```
put dt nldatmyql;
put dt nldatmyql6;
put dt nldatmyql7;
put dt nldatmyql18;
put dt nldatmyql200;
run;
```

```
+--- NLDATMYQL min=4 default=18 max=200 ---+
2nd quarter 2019
2019/2
Q2 2019
2nd quarter 2019
2nd quarter 2019
```

NLDATMYQMw. 出力形式

SAS 日時値を現在の SAS ロケールに適した年の四半期(1~4)と年として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYQM_w.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 7

範囲 4-200

詳細

NLDATMYQM では、Q2 2019 など、短縮形を使用した簡略形式で四半期、次に年を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。

```
data _null_;  
  dt = datetime();  
  dy = date();  
  put '+++ NLDATMYQM min=4 default=7 max=200 ---+';  
  put dt nldatmyqm.;  
  put dt nldatmyqm6.;  
  put dt nldatmyqm7.;  
  put dt nldatmyqm200.;  
run;
```

```
+++ NLDATMYQM min=4 default=7 max=200 ---+  
Q2 2019  
2019/2  
Q2 2019  
Q2 2019
```

NLDATMYQSw. 出力形式

数字と区切り文字を使用して、SAS 日時値を現在の SAS ロケールに適した年と四半期(1~4)として、書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYQSw.

構文の説明

w

出力フィールドの幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 6

範囲 4-200

詳細

NLDATMYQS では、2012/4 など、数字と区切り文字を使用した短い形式で年と四半期を含む日付が出力されます。

例

次の例では、日付として 2019 年 04 月 16 日を使用します。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATMYQS min=4 default=6 max=200 ---+';
  put dt nldatmyqs;
  put dt nldatmyqs4;
  put dt nldatmyqs6;
  put dt nldatmyqs200;
run;
```

```
+--- NLDATMYQS min=4 default=6 max=200 ---+
2019/2
19/2
2019/2
2019/2
```

NLDATMYRw. 出力形式

現在の SAS ロケールに適した年として SAS 日時値を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYR*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 2-200

例

この例では、デフォルトの SAS セッションエンコーディングである en_US ロケール オプションを使用しています。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+--- NLDATMYR min=2 default=16 max=200 ---+';
  put dt nldatmyr.;
  put dt nldatmyr2.;
  put dt nldatmyr32.;
  put dt nldatmyr200.;
run;
```

```
+--- NLDATMYR min=2 default=16 max=200 ---+
2019
19
2019
2019
```

NLDATMYW*w*. 出力形式

SAS 日時値を現在の SAS ロケールに適した週番号と年として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMYW*w*.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 16

範囲 5-200

例

この例では、デフォルトの SAS セッションエンコーディングである en_US ロケールオプションを使用しています。

```
data _null_;
  dt = datetime();
  dy = date();
  put '+++ NLDATMYW min=5 default=16 max=200 ---+';
  put ' 16' +5 dt nldatmyw.;
  put ' 5' +5 dt nldatmyw5.;
  put ' 8' +5 dt nldatmyw8.;
  put ' 32' +5 dt nldatmyw32.;
  put ' 200' +5 dt nldatmyw200.;
run;
```

```
+++ NLDATMYW min=5 default=16 max=200 ---+
16      Week 16 2019
5       *****
8       W16 19
32      Week 16 2019
200
Week 16 2019
```

NLDATMZw. 出力形式

SAS 日時値を現在の SAS ロケールに適した日時とタイムゾーンとして書き込みます。

カテゴリ:	CAS 日付と時間
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMZ*w*.

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	40
範囲	16-200

例

この例は現在の日時値を使用します。この例では、fr_FR ロケールを指定します。

```
options locale=fr_FR;
data test;
  x=datetime();
  put x=nldatmz.;
run;
```

```
x=16 avril 2019 11 h 40 -0400
```

NLMNACIAED w.d 出力形式

アラブ首長国連邦の国際会計表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIAED w.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_AE;  
x=put(-1234.56789,nlmnaciaed32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.2 NLMNACIAED 出力

ロケール	フォーマットされた値
ar_AE	(1,234.57AED)
en_US	(AED1,234.57)
es_ES	-1.234,57 AED

関連項目

出力形式:

- “NLMNLAEDx.d 出力形式” (422 ページ)
- “NLMNACLAEDx.d 出力形式”
- “NLMNIAEDw.d 出力形式”

NLMNACIAUD w.d 出力形式

オーストラリアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIAUD w.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_AU;
x=put(-1234.56789,nlmnaciaud32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.3 NLMNACIAUD 出力

ロケール	フォーマットされた値
en_AU	(AUD1,234.57)
en_US	(AUD1,234.57)
es_ES	-1.234,57 AUD

関連項目

出力形式:

- [“NLMNLAEDx.d 出力形式” \(422 ページ\)](#)
- [“NLMNIAUDw.d 出力形式”](#)
- [“NLMNACIAUDw.d 出力形式”](#)

NLMNACIBGN w.d 出力形式

ブルガリアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIBGN *w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=bg_BG;  
x=put(-1234.56789,nlmnacibgn32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.4 NLMNACIBGN 出力

ロケール	フォーマットされた 値
bg_BG	(1 234,57 BGN)
en_US	(BGN1,234.57)
es_ES	-1.234,57 BGN

関連項目

出力形式:

- “NLMNLBGNw.d 出力形式” (424 ページ)
- “NLMNIBGNw.d 出力形式”
- “NLMNACLBGNw.d 出力形式”

NLMNACIBRL w.d 出力形式

ブラジルの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIBRL w.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=pt_BR;  
x=put(-1234.56789,nlmcacibrl32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.5 NLMNACIBRL 出力

ロケール	フォーマットされた 値
pt_BR	-BRL 1.234,57
en_US	(BRL1,234.57)
es_ES	-1.234,57 BRL

関連項目

出力形式:

- [“NLMNLBRLw.d 出力形式” \(426 ページ\)](#)
- [“NLMNIBRLw.d 出力形式”](#)
- [“NLMNACLBRLw.d 出力形式”](#)

NLMNACICAD w.d 出力形式

カナダの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACICAD *w.d*

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=en_CA;  
x=put(-1234.56789,nlmnacicad32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.6 NLMNACICAD 出力

ロケール	フォーマットされた 値
en_CA	(CAD1,234.57)
en_US	(CAD1,234.57)
es_ES	-1.234,57 CAD

関連項目

出力形式:

- “NLMNLCADw.d 出力形式” (427 ページ)
- “NLMNICADw.d 出力形式”
- “NLMNACLCADw.d 出力形式”

NLMNACICHF w.d 出力形式

リヒテンシュタインとスイスの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACICHF w.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=fr_CH;
x=put(-1234.56789,nlmcnchf32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.7 NLMNACICHF 出力

ロケール	フォーマットされた値
fr_CH	(1 234,57 CHF)
en_US	(CHF1,234.57)
es_ES	-1.234,57 CHF

関連項目

出力形式:

- “NLMNLCHFW.d 出力形式” (429 ページ)
- “NLMNICHFW.d 出力形式”
- “NLMNACLCHFW.d 出力形式”

NLMNACICNY w.d 出力形式

中国の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACICNY *w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 02

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_CN;  
x=put(-1234.56789,nlmnacicny32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.8 NLMNACICNY 出力

ロケール	フォーマットされた 値
zh_CN	(CNY1,234.57)
en_US	(CNY1,234.57)
es_ES	-1.234,57 CNY

関連項目

出力形式:

- “NLMNLCNYw.d 出力形式” (430 ページ)
- “NLMNICNYw.d 出力形式”
- “NLMNACLCNYw.d 出力形式”

NLMNACICZK w.d 出力形式

チェコの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACICZK w.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=cs_CZ;  
x=put(-1234.56789,nlmnacicz32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.9 NLMNACICZK 出力

ロケール	フォーマットされた 値
cs_CZ	-1 234,57 CZK
en_US	(CZK1,234.57)
es_ES	-1.234,57 CZK

関連項目

出力形式:

- [“NLMNLCZKw.d 出力形式” \(431 ページ\)](#)
- [“NLMNICZKw.d 出力形式”](#)
- [“NLMNACLCZKw.d 出力形式”](#)

NLMNACIDKK w.d 出力形式

デンマーク、フェロー諸島およびグリーンランドの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになる

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIDKK *w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=da_dk;  
x=put(-1234.56789,nlmnacidkk32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.10 NLMNACIDKK 出力

ロケール	フォーマットされた 値
da_DK	-1.234,57 DKK
en_US	(DKK1,234.57)
es_ES	-1.234,57 DKK

関連項目

出力形式:

- “NLMNLDKKw.d 出力形式” (433 ページ)
- “NLMNIDKKw.d 出力形式”
- “NLMNACLDKKw.d 出力形式”

NLMNACIEEK w.d 出力形式

エストニアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: エストニアは通貨にユーロを使用しています。NLMNACIEEK は、歴史的な目的のためにドキュメント化されています。

この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIEEK w.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=et_EE;
x=put(-1234.56789,nlmnacieek32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.11 NLMNACIEEK 出力

ロケール	フォーマットされた値
et_EE	(1 234,57 EEK)
en_US	(EEK1,234.57)
es_ES	-1.234,57 EEK

関連項目

出力形式:

- [“NLMNLEEKw.d 出力形式” \(434 ページ\)](#)
- [“NLMNIEEKw.d 出力形式”](#)
- [“NLMNACLEEKw.d 出力形式”](#)

NLMNACIEGPw.d 出力形式

エジプトの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIEGP*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
option locale=ar_EG;  
x=put(-1234.56789,nlmnaciegp32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.12 NLMNACIEGP 出力

ロケール	フォーマットされた 値
ar_EG	(1,234.57EGP)
en_US	(EGP1,234.57)
es_ES	-1.234,57 EGP

関連項目

出力形式:

- “NLMNLEGPw.d 出力形式” (436 ページ)
- “NLMNIEGPw.d 出力形式”
- “NLMNACLEGPw.d 出力形式”

NLMNACIEURw.d 出力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIEURw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。次の例では、LOCALE=システムオプションが de_DE に設定されています。

```
option locale=de_DE;  
x=put(-1234.56789,nlmcacur32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.13 NLMNACIEUR 出力

ロケール	フォーマットされた 値
de_DE	-1.234,57 EUR
en_US	(EUR1,234.57)
es_ES	-1.234,57 EUR

関連項目

出力形式:

- [“NLMNLEURw.d 出力形式” \(437 ページ\)](#)
- [“NLMNIEURw.d 出力形式”](#)
- [“NLMNACLEURw.d 出力形式”](#)

NLMNACIGBPw.d 出力形式

英国の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIGBPw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
option locale=en_GB;  
x=put(-1234.56789,nlmcacigbp32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.14 NLMNACIGBP 出力

ロケール	フォーマットされた 値
en_AU	(GBP1,234.57)
en_US	(GBP1,234.57)
es_ES	-1.234,57 GBP

関連項目

出力形式:

- “NLMNLGBPw.d 出力形式” (439 ページ)
- “NLMNIGBPw.d 出力形式”
- “NLMNACLGBPw.d 出力形式”

NLMNACIHKDw.d 出力形式

香港の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIHKDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=en_HK;  
x=put(-1234.56789,nlmnacihrkd32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.15 NLMNACIHKD 出力

ロケール	フォーマットされた値
en_AU	(HKD1,234.57)
en_US	(HKD1,234.57)
es_ES	-1.234,57 HKD

関連項目

- 出力形式:
- “NLMNLHKDw.d 出力形式” (440 ページ)
 - “NLMNIHKDw.d 出力形式”
 - “NLMNACLHKDw.d 出力形式”

NLMNACIHRKw.d 出力形式

クロアチアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIHRK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
option locale=hr_HR;  
x=put(-1234.56789,nlmnacihrk32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.16 NLMNACIHRK 出力

ロケール	フォーマットされた 値
hr_HR	-1.234,57 HRK
en_US	(HRK1,234.57)
es_ES	-1.234,57 HRK

関連項目

出力形式:

- “NLMNLHRKw.d 出力形式” (442 ページ)
- “NLMNIHRKw.d 出力形式”
- “NLMNACLHRKw.d 出力形式”

NLMNACIHUFw.d 出力形式

ハンガリーの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIHUFw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=hu_HU  
x=put(-1234.56789,nlmcacihuf32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.17 NLMNACIHUF 出力

ロケール	フォーマットされた 値
hu_HU	-1.234,57 HRK
en_US	(HUF1,234.57)
es_ES	-1.234,57 HUF

関連項目

出力形式:

- “NLMNLHUFw.d 出力形式” (443 ページ)
- “NLMNIHUFw.d 出力形式”
- “NLMNACLHUFw.d 出力形式”

NLMNACIIDRw.d 出力形式

インドネシアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIIDR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=id_ID;  
x=put(-1234.56789,nlmnaciidr32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.18 NLMNACIIDR 出力

ロケール	フォーマットされた 値
id_ID	-IDR1.234,57
en_US	(IDR1,234.57)
es_ES	-1.234,57 IDR

関連項目

出力形式:

- “NLMNLIDRw.d 出力形式” (444 ページ)
- “NLMNIIDRw.d 出力形式”
- “NLMNACLIDRw.d 出力形式”

NLMNACIILSw.d 出力形式

イスラエルの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIILSw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=he_IL;
x=put(-1234.56789,nlmcnaciils32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.19 NLMNACIILS 出力

ロケール	フォーマットされた値
he_IL	-1.234,57 ILS
en_US	(IL\$1,234.57)
es_ES	-1.234,57 ILS

関連項目

出力形式:

- [“NLMNLILSw.d 出力形式” \(446 ページ\)](#)
- [“NLMNIILSw.d 出力形式”](#)
- [“NLMNACLILSw.d 出力形式”](#)

NLMNACIINRw.d 出力形式

インドの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIINR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=hi_IN;  
x=put(-1234.56789,nlmnaciinr32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.20 NLMNACIINR 出力

ロケール	フォーマットされた 値
hi_IN	-INR1,234.57
en_US	(INR1,234.57)
es_ES	-1.234,57 INR

関連項目

出力形式:

- “NLMNLINRw.d 出力形式” (447 ページ)
- “NLMNIINRw.d 出力形式”
- “NLMNACLINRw.d 出力形式”

NLMNACIJPYw.d 出力形式

日本の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIJPYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=ja_JP;  
x=put(-1234.56789,nlmcijpy32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.21 NLMNACIJPY 出力

ロケール	フォーマットされた 値
ja_JP	(JPY1,234.57)
en_US	(JPY1,234.57)
es_ES	-1.234,57 JPY

関連項目

出力形式:

- [“NLMNLCJPYw.d 出力形式” \(449 ページ\)](#)
- [“NLMNICJPYw.d 出力形式”](#)
- [“NLMNACLCJPYw.d 出力形式”](#)

NLMNACIKRWw.d 出力形式

韓国の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになる

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIKRW*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=ko_KR;  
x=put(-1234.56789,nlmacikrw32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.22 NLMNACIKRW 出力

ロケール	フォーマットされた 値
ko_KR	(KRW1,234.57)
en_US	(KRW1,234.57)
es_ES	-1.234,57 KRW

関連項目

出力形式:

- “NLMNLKRWw.d 出力形式” (450 ページ)
- “NLMNIKRWw.d 出力形式”
- “NLMNACLKRWw.d 出力形式”

NLMNACILTLw.d 出力形式

リトアニアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACILTLw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=lt_LT;  
x=put(-1234.56789,nlmcnclt32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.23 NLMNACILTL 出力

ロケール	フォーマットされた値
lt_LT	-1 234,57 LTL
en_US	(LTL1,234.57)
es_ES	-1.234,57 LTL

関連項目

出力形式:

- [“NLMNLLTLw.d 出力形式” \(451 ページ\)](#)
- [“NLMNITLw.d 出力形式”](#)
- [“NLMNACLLTLw.d 出力形式”](#)

NLMNACILVLw.d 出力形式

ラトビアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACILVL*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=lv_LV;  
x=put(-1234.56789,nlmcacilvl32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.24 NLMNACILVL 出力

ロケール	フォーマットされた 値
lv_LV	-1 234,57 LVL
en_US	(LVL1,234.57)
es_ES	-1.234,57 LVL

関連項目

出力形式:

- “NLMNLLVLw.d 出力形式” (453 ページ)
- “NLMNILVLw.d 出力形式”
- “NLMNACLLVLw.d 出力形式”

NLMNACIMOPw.d 出力形式

マカオの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIMOPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=zh_MO;  
x=put(-1234.56789,nlmcimop32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.25 NLMNACIMOP 出力

ロケール	フォーマットされた値
zh_MO	(MOP1,234.57)
en_US	(MOP1,234.57)
es_ES	-1.234,57 MOP

関連項目

出力形式:

- [“NLMNLMOPw.d 出力形式” \(454 ページ\)](#)
- [“NLMNIMOPw.d 出力形式”](#)
- [“NLMNACLMOPw.d 出力形式”](#)

NLMNACIMXNw.d 出力形式

メキシコの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIMXN*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
option locale=es_MX;  
x=put(-1234.56789,nlmcimxn32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.26 NLMNACIMXN 出力

ロケール	フォーマットされた 値
es_MX	-MXN1,234.57
en_US	(MXN1,234.57)
es_ES	-1.234,57 MXN

関連項目

出力形式:

- “NLMNLMXNw.d 出力形式” (456 ページ)
- “NLMNIMXNw.d 出力形式”
- “NLMNACLMXNw.d 出力形式”

NLMNACIMYRw.d 出力形式

マレーシアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIMYRw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=ms_MY;  
x=put(-1234.56789,nlmnacimyr32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.27 NLMNACIMYR 出力

ロケール	フォーマットされた値
ms_MY	(MYR1,234.57)
en_US	(MYR1,234.57)
es_ES	-1.234,57 MYR

関連項目

- 出力形式:
- “NLMNLMYRw.d 出力形式” (457 ページ)
 - “NLMNIMYRw.d 出力形式”
 - “NLMNACLMYRw.d 出力形式”

NLMNACINOKw.d 出力形式

ノルウェーの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACINOK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
options locale=no_NO;  
x=put(-1234.56789,nlmnacinok32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.28 NLMNACINOK 出力

ロケール	フォーマットされた 値
no_NO	(NOK 1 234,57)
en_US	(NOK1,234.57)
es_ES	-1.234,57 NOK

関連項目

出力形式:

- “NLMNLNOKw.d 出力形式” (459 ページ)
- “NLMNINOKw.d 出力形式”
- “NLMNACLNOKw.d 出力形式”

NLMNACINZDw.d 出力形式

ニュージーランドの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACINZDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_NZ;  
x=put(-1234.56789,nlmcnacd32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.29 NLMNACINZD 出力

ロケール	フォーマットされた値
en_NZ	(NZD1,234.57)
en_US	(NZD1,234.57)
es_ES	-1.234,57 NZD

関連項目

- 出力形式:
- [“NLMNLNZDw.d 出力形式” \(460 ページ\)](#)
 - [“NLMNINZDw.d 出力形式”](#)
 - [“NLMNACLNZDw.d 出力形式”](#)

NLMNACIPLNw.d 出力形式

ポーランドの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIPLNw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=pl_PL;  
x=put(-1234.56789,nlmnacipln32.2;
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.30 NLMNACIPLN 出力

ロケール	フォーマットされた 値
pl_PL	(1 234,57 PLN)
en_US	(PLN1,234.57)
es_ES	-1.234,57 PLN

関連項目

出力形式:

- “NLMNLPLNw.d 出力形式” (462 ページ)
- “NLMNIPLNw.d 出力形式”
- “NLMNACLPLNw.d 出力形式”

NLMNACIRUBw.d 出力形式

ロシアの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIRUBw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ru_RU;
x=put(-1234.56789,nlmcacirub32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.31 NLMNACIRUB 出力

ロケール	フォーマットされた値
ru_RU	-1 234,57 RUB
en_US	(RUB1,234.57)
es_ES	-1.234,57 RUB

関連項目

出力形式:

- [“NLMNLRUBw.d 出力形式” \(463 ページ\)](#)
- [“NLMNIRUBw.d 出力形式”](#)
- [“NLMNACLRUBw.d 出力形式”](#)

NLMNACISEKw.d 出力形式

スウェーデンの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACISEK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=sv_SE;  
x=put(-1234.56789,nlmnacisek32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.32 NLMNACISEK 出力

ロケール	フォーマットされた 値
sv_SE	-1 234,57 SEK
en_US	(SEK1,234.57)
es_ES	-1.234,57 SEK

関連項目

出力形式:

- “NLMNLSEKw.d 出力形式” (464 ページ)
- “NLMNISEKw.d 出力形式”
- “NLMNACLSEKw.d 出力形式”

NLMNACISGDw.d 出力形式

シンガポールの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACISGD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_SG;  
x=put(-1234.56789,nlmcisgd32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.33 NLMNACISGD 出力

ロケール	フォーマットされた値
zh_SG	(SGD1,234.57)
en_US	(SGD1,234.57)
es_ES	-1.234,57 SGD

関連項目

出力形式:

- [“NLMNLSGDw.d 出力形式” \(466 ページ\)](#)
- [“NLMNISGDw.d 出力形式”](#)
- [“NLMNACLSGDw.d 出力形式”](#)

NLMNACITHBw.d 出力形式

タイの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACITHBw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=th_TH;  
x=put(-1234.56789,nlmcacithb32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.34 NLMNACITHB 出力

ロケール	フォーマットされた 値
th_TH	(THB1,234.57)
en_US	(THB1,234.57)
es_ES	-1.234,57 THB

関連項目

出力形式:

- “NLMNLTHBw.d 出力形式” (467 ページ)
- “NLMNITHBw.d 出力形式”
- “NLMNACLTHBw.d 出力形式”

NLMNACITRYw.d 出力形式

トルコの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACITRYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=tr_TR;  
x=put(-1234.56789,nlmcnctry32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.35 NLMNACITRY 出力

ロケール	フォーマットされた値
tr_TR	(TRY1.234,57)
en_US	(TRY1,234.57)
es_ES	-1.234,57 TRY

関連項目

- 出力形式:
- [“NLMNLTRYw.d 出力形式” \(469 ページ\)](#)
 - [“NLMNITRYw.d 出力形式”](#)
 - [“NLMNACLTRYw.d 出力形式”](#)

NLMNACITWDw.d 出力形式

台湾の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになる

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACITWD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_TW;  
x=put(-1234.56789,nlmcacitwd32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.36 NLMNACITWD 出力

ロケール	フォーマットされた 値
zh_TW	(TWD1,234.57)
en_US	(TWD1,234.57)
es_ES	-1.234,57 TWD

関連項目

出力形式:

- “NLMNLTWDw.d 出力形式” (470 ページ)
- “NLMNITWDw.d 出力形式”
- “NLMNACLTDw.d 出力形式”

NLMNACIUSDw.d 出力形式

プエルトリコと米国の国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACIUSDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=es_US;  
x=put(-1234.56789,nlmcnaciUSD32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.37 NLMNACIUSD 出力

ロケール	フォーマットされた値
es_US	-USD1,234.57
en_US	(USD1,234.57)
es_ES	-1.234,57 USD

関連項目

- 出力形式:
- [“NLMNLUSDw.d 出力形式” \(471 ページ\)](#)
 - [“NLMNIUSDw.d 出力形式”](#)
 - [“NLMNACIUSDw.d 出力形式”](#)

NLMNACIZARw.d 出力形式

南アフリカの国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACIZAR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=af_ZA;  
x=put(-1234.56789,nlmcizar32.2;
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.38 NLMNACIZAR 出力

ロケール	フォーマットされた 値
af_ZA	(ZAR1 234,57)
en_US	(ZAR1,234.57)
es_ES	-1.234,57 ZAR

関連項目

出力形式:

- “NLMNLZARw.d 出力形式” (473 ページ)
- “NLMNIZARw.d 出力形式”
- “NLMNACLZARw.d 出力形式”

NLMNACLAEDx.d 出力形式

アラブ首長国連邦の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLAEDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_AE
x=put(-1234.56789,nlmnaclaed32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.39 NLMNACLAED 出力

ロケール	フォーマットされた値
ar_AE	(1,234.57)ل.ا.
en_US	(1,234.57)ل.ا.
es_ES	-1.234,57 ل.ا.

関連項目

- 出力形式:**
- [“NLMNIAEDw.d 出力形式” \(369 ページ\)](#)
 - [“NLMNLAEDx.d 出力形式”](#)
 - [“NLMNACIAED w.d 出力形式”](#)

NLMNACLAUDw.d 出力形式

オーストラリアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLAUD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=en_AU;  
x=put(-1234.56789,nlmnaclaud32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.40 NLMNACLAUD 出力

ロケール	フォーマットされた 値
en_AU	(AU\$1,234.57)
en_US	(AU\$1,234.57)
es_ES	-1.234,57 AU\$

関連項目

出力形式:

- “NLMNIAUDw.d 出力形式” (371 ページ)
- “NLMNLAUDw.d 出力形式”
- “NLMNACIAUD w.d 出力形式”

NLMNACLBGNw.d 出力形式

ブルガリアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLBGNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=bg_BG;  
x=put(-1234.56789,nlmlaclbgn32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.41 NLMNACLBGN 出力

ロケール	フォーマットされた値
bg_BG	(1 234,57 лв)
en_US	-лв1,234.57
es_ES	-1.234,57 лв

関連項目

- 出力形式:
- [“NLMNIBGNw.d 出力形式” \(372 ページ\)](#)
 - [“NLMNLBGNw.d 出力形式”](#)
 - [“NLMNACIBGN w.d 出力形式”](#)

NLMNACLBRLw.d 出力形式

ブラジルの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLBRL*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=pt_BR;  
x=put(-1234.56789,nlmmaclbrl32.2);
```

表 10.42 NLMNACLBRL 出力

ロケール	フォーマットされた 値
pt_BR	-R\$ 1.234,57
en_US	(R\$1,234.57)
es_ES	-1.234,57 R\$

関連項目

出力形式:

- “NLMNIBRLw.d 出力形式” (373 ページ)
- “NLMNLBRLw.d 出力形式”
- “NLMNACIBRL w.d 出力形式”

NLMNACLCADw.d 出力形式

カナダの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLCADw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_CA;  
x=put(-1234.56789,nlmcad32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.43 NLMNACLCAD 出力

ロケール	フォーマットされた値
en_CA	(CA\$1,234.57)
en_US	(CA\$1,234.57)
es_ES	-1.234,57 CA\$

関連項目

- 出力形式:
- [“NLMNICADw.d 出力形式” \(375 ページ\)](#)
 - [“NLMNLCADw.d 出力形式”](#)
 - [“NLMNACICAD w.d 出力形式”](#)

NLMNACLCHFW.d 出力形式

リヒテンシュタインとスイスの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLCHF*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=fr_CH;  
x=put(-1234.56789,nlmnaclchf32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.44 NLMNACLCHF 出力

ロケール	フォーマットされた 値
fr_CH	(1 234.57 SFr.)
en_US	(SFr.1,234.57)
es_ES	-1.234,57 SFr.

関連項目

出力形式:

- “NLMNICHw.d 出力形式” (376 ページ)
- “NLMNLCHw.d 出力形式”

NLMNACLCNYw.d 出力形式

中国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLCNYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_CN;  
x=put(-1234.56789,nlмнаclcnу32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.45 NLMNACLCNY 出力

ロケール	フォーマットされた値
zh_CN	(RMB1,234.57)
en_US	(RMB1,234.57)
es_ES	-1.234,57 RMB

関連項目

- 出力形式:**
- [“NLMNICNYw.d 出力形式” \(378 ページ\)](#)
 - [“NLMNLCNYw.d 出力形式”](#)
 - [“NLMNACICNY w.d 出力形式”](#)

NLMNACLCZKw.d 出力形式

チェコの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLCZK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=cs_CZ;  
x=put(-1234.56789,nlmmaclczk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.46 NLMNACLCZK 出力

ロケール	フォーマットされた 値
cs_CZ	-1 234,57 Kč
en_US	(Kč1,234.57)
es_ES	-1.234,57 Kč

関連項目

出力形式:

- “NLMNICZKw.d 出力形式” (379 ページ)
- “NLMNLCZKw.d 出力形式”
- “NLMNACICZK w.d 出力形式”

NLMNACLDKKw.d 出力形式

デンマーク、フェロー諸島およびグリーンランドの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLDKKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=da_dk;  
x=put(-1234.56789,nlmcldkk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.47 NLMNACLDKK 出力

ロケール	フォーマットされた値
da_dk	-1.234,57 kr
en_US	(kr1,234.57)
es_ES	-1.234,57 kr

関連項目

出力形式:

- [“NLMNIDKKw.d 出力形式” \(380 ページ\)](#)
- [“NLMNLDDKKw.d 出力形式”](#)
- [“NLMNACIDKK w.d 出力形式”](#)

NLMNACLEEKw.d 出力形式

エストニアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
エストニアは通貨にユーロを使用しています。NLMNACLEEK は、歴史的な目的のためにドキュメント化されています。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YESをお勧めします。

構文

NLMNACLEEK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=et_EE;  
x=put(-1234.56789,nlmnacleek32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.48 NLMNACLEEK 出力

ロケール	フォーマットされた 値
et_EE	(1 234,57 kr)
en_US	(kr1,234.57)
es_ES	-1.234,57 kr

関連項目

出力形式:

- “NLMNIEEKw.d 出力形式” (382 ページ)
- “NLMNLEEKw.d 出力形式”
- “NLMNACIEEK w.d 出力形式”

NLMNACLEGPw.d 出力形式

エジプトの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLEGPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_EG;  
x=put(-1234.56789,nlmcnclgep32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.49 NLMNACLEGP 出力

ロケール	フォーマットされた 値
ar_EG	(1,234.57م.ع.)
en_US	(1,234.57,م.ع.)
es_ES	-1.234,57 م.ع.

関連項目

出力形式:

- [“NLMNIEGPw.d 出力形式” \(383 ページ\)](#)
- [“NLMNLEGPw.d 出力形式”](#)
- [“NLMNACIEGPw.d 出力形式”](#)

NLMNACLEURw.d 出力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YESをお勧めします。

構文

NLMNACLEUR $w.d$

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=de_DE;  
y=put(-1234.56789,nlmnacleur32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.50 NLMNACLEUR 出力

ロケール	フォーマットされた 値
de_DE	-1.234,57 €
en_US	(€1,234.57)
es_ES	-1.234,57 €

関連項目

出力形式:

- “NLMNIEURw.d 出力形式” (385 ページ)
- “NLMNLEURw.d 出力形式”
- “NLMNACIEURw.d 出力形式”

NLMNACLGBPw.d 出力形式

英国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLGBPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_GB;  
x=put(-1234.56789,nlmcnclgbp32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.51 NLMNACLGBP 出力

ロケール	フォーマットされた値
en_GB	(£1,234.57)
en_US	(£1,234.57)
es_ES	£ -1.234,57

関連項目

- 出力形式:
- [“NLMNIGBPw.d 出力形式” \(386 ページ\)](#)
 - [“NLMNLGBPw.d 出力形式”](#)
 - [“NLMNACIGBPw.d 出力形式”](#)

NLMNACLHKDw.d 出力形式

香港の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLHKD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=en_HK;  
x=put(-1234.56789,nlmnaclhkd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.52 NLMNACLHKD 出力

ロケール	フォーマットされた 値
en_HK	(HK\$1,234.57)
en_US	(HK\$1,234.57)
es_ES	-1.234,57 HK\$

関連項目

出力形式:

- “NLMNIHKDw.d 出力形式” (388 ページ)
- “NLMNLHKDw.d 出力形式”
- “NLMNACIHKDw.d 出力形式”

NLMNACLHRKw.d 出力形式

クロアチアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLHRKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=hr_HR;  
x=put(-1234.56789,nlmlaclhrk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.53 NLMNACLHRK 出力

ロケール	フォーマットされた 値
hr_HR	-1.234,57 Kn
en_US	(Kn1,234.57)
es_ES	-1.234,57 Kn

関連項目

出力形式:

- [“NLMNIHRKw.d 出力形式” \(389 ページ\)](#)
- [“NLMNLHRKw.d 出力形式”](#)
- [“NLMNACIHRKw.d 出力形式”](#)

NLMNACLHUFw.d 出力形式

ハンガリーの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLHUFw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=hu_HU;  
x=put(-1234.56789,nlmlaclhuf32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.54 NLMNACLHUF 出力

ロケール	フォーマットされた 値
hu_HU	-1 234,57 Ft
en_US	(Ft1,234.57)
es_ES	-1.234,57 Ft

関連項目

出力形式:

- “NLMNIHUFw.d 出力形式” (390 ページ)
- “NLMNLHUFw.d 出力形式”
- “NLMNACIHUFw.d 出力形式”

NLMNACLIDRw.d 出力形式

インドネシアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLIDRw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=id_ID;  
x=put(-1234.56789,nlmnaclidr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.55 NLMNACLIDR 出力

ロケール	フォーマットされた値
id_ID	-Rp1.234,57
en_US	(Rp1,234.57)
es_ES	-1.234,57 Rp

関連項目

- 出力形式:
- [“NLMNIIDRw.d 出力形式” \(392 ページ\)](#)
 - [“NLMNLIDRw.d 出力形式”](#)
 - [“NLMNACIIDRw.d 出力形式”](#)

NLMNACLILSw.d 出力形式

イスラエルの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLILS*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=he_IL;  
x=put(-1234.56789,nlmnaclils32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.56 NLMNACLILS 出力

ロケール	フォーマットされた 値
he_IL	-1,234.57 n"ש
en_US	(1,234.57n"ש)
es_ES	-1,234.57 n"ש

関連項目

出力形式:

- “NLMNIILSw.d 出力形式” (393 ページ)
- “NLMNLILSw.d 出力形式”
- “NLMNACIILSw.d 出力形式”

NLMNACLINRw.d 出力形式

インドの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLINR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=hi_IN;  
x=put(-1234.56789,nlмнаclnr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.57 NLMNACLINR 出力

ロケール	フォーマットされた値
hi_IN	-Rs1,234.57
en_US	(Rs1,234.57)
es_ES	-1.234,57 Rs

関連項目

- 出力形式:
- [“NLMNIINRw.d 出力形式” \(395 ページ\)](#)
 - [“NLMNLINRw.d 出力形式”](#)
 - [“NLMNACIINRw.d 出力形式”](#)

NLMNACLJPYw.d 出力形式

日本の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLJPY*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=ja_JP;  
x=put(-1234.56789,nlmlacljpy32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.58 NLMNACLJPY 出力

ロケール	フォーマットされた 値
ja_JP	(¥ 1,234.57)
en_US	(¥ 1,234.57)
es_ES	-1.234,57 ¥

関連項目

出力形式:

- “NLMNIJPYw.d 出力形式” (396 ページ)
- “NLMNLJPYw.d 出力形式”
- “NLMNACJPYw.d 出力形式”

NLMNACLKRWw.d 出力形式

韓国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLKRWw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ko_KR;
x=put(-1234.56789,nlmlnclkrw32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.59 NLMNACLKRW 出力

ロケール	フォーマットされた値
ko_KR	(₩1,234.57)
en_US	(¥1,234.57)
es_ES	-1.234,57 ₩

関連項目

出力形式:

- [“NLMNIKRWw.d 出力形式” \(397 ページ\)](#)
- [“NLMNLKRWw.d 出力形式”](#)
- [“NLMNACIKRWw.d 出力形式”](#)

NLMNACLLTLw.d 出力形式

リトアニアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLLTL*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

Options locale=lt_LT;
x=put(-1234.56789,nlmmaclltl32.2);

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.60 NLMNACLLTL 出力

ロケール	フォーマットされた 値
lt_LT	-1 234,57 Lt
en_US	(Lt1,234.57)
es_ES	-1.234,57 Lt

関連項目

出力形式:

- “NLMNITLw.d 出力形式” (399 ページ)
- “NLMNLLTLw.d 出力形式”
- “NLMNACILTLw.d 出力形式”

NLMNACLLVLw.d 出力形式

ラトビアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLLVLw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=lv_LV;  
x=put(-1234.56789,nlmlncllv132.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.61 NLMNACLLVL 出力

ロケール	フォーマットされた値
lv_LV	-1 234,57 Ls
en_US	(Ls1,234.57)
es_ES	-1.234,57 Ls

関連項目

- 出力形式:
- [“NLMNILVLw.d 出力形式” \(400 ページ\)](#)
 - [“NLMNLLVLw.d 出力形式”](#)
 - [“NLMNACILVLw.d 出力形式”](#)

NLMNACLMOPw.d 出力形式

マカオの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLMOPw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_MO;  
x=put(-1234.56789,nlmmaclmop32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.62 NLMNACLMOP 出力

ロケール	フォーマットされた 値
zh_MO	(P1,234.57)
en_US	(P1,234.57)
es_ES	-1.234,57 P

関連項目

出力形式:

- “NLMNIMOPw.d 出力形式” (402 ページ)
- “NLMNLMOPw.d 出力形式”
- “NLMNACIMOPw.d 出力形式”

NLMNACLMXNw.d 出力形式

メキシコの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLMXNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=es_MX;
x=put(-1234.56789,nlmlnclmxn32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.63 NLMNACLMXN 出力

ロケール	フォーマットされた値
es_MX	-MX\$1,234.57
en_US	(MX\$1,234.57)
es_ES	-1.234,57 MX\$

関連項目

出力形式:

- [“NLMNIMXNw.d 出力形式” \(403 ページ\)](#)
- [“NLMNLMXNw.d 出力形式”](#)
- [“NLMNACIMXNw.d 出力形式”](#)

NLMNACLMYRw.d 出力形式

マレーシアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLMYR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=ms_MY;  
x=put(-1234.56789,nlmnaclmyr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.64 NLMNACLMYR 出力

ロケール	フォーマットされた 値
ms_MY	(R1,234.57)
en_US	(R1,234.57)
es_ES	-1.234,57 R

関連項目

出力形式:

- “NLMNIMYRw.d 出力形式” (405 ページ)
- “NLMNLMYRw.d 出力形式”
- “NLMNACIMYRw.d 出力形式”

NLMNACLNOKw.d 出力形式

ノルウェーの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLNOKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=no_NO;  
x=put(-1234.56789,nlmlnclnok32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.65 NLMNACLNOK 出力

ロケール	フォーマットされた 値
no_NO	(kr 1 234,57)
en_US	(kr1,234.57)
es_ES	-1.234,57 kr

関連項目

出力形式:

- [“NLMNINOKw.d 出力形式” \(406 ページ\)](#)
- [“NLMNLNOKw.d 出力形式”](#)
- [“NLMNACINOKw.d 出力形式”](#)

NLMNACLNZDw.d 出力形式

ニュージーランドの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLNZD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=en_NZ;  
x=put(-1234.56789,nlmmaclnzd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.66 NLMNACLNZD 出力

ロケール	フォーマットされた 値
en_NZ	(NZ\$1,234.57)
en_US	(NZ\$1,234.57)
es_ES	-1.234,57 NZ\$

関連項目

出力形式:

- “NLMNINZDw.d 出力形式” (407 ページ)
- “NLMNLNZDw.d 出力形式”
- “NLMNACINZDw.d 出力形式”

NLMNACLPLNw.d 出力形式

ポーランドの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLPLNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=pl_PL;  
x=put(-1234.56789,nlmln32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.67 NLMNACL PLN 出力

ロケール	フォーマットされた値
pl_PL	(1 234,57 zł)
en_US	(zł1,234.57)
es_ES	-1.234,57 zł

関連項目

- 出力形式:
- [“NLMNIPLNw.d 出力形式” \(409 ページ\)](#)
 - [“NLMNLPLNw.d 出力形式”](#)
 - [“NLMNACIPLNw.d 出力形式”](#)

NLMNACL RUBw.d 出力形式

ロシアの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLRUB*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=ru_RU;  
x=put(-1234.56789,nlmmaclrub32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.68 NLMNACLRUB 出力

ロケール	フォーマットされた 値
ru_RU	-1 234,57 py6
en_US	(py61,234.57)
es_ES	-1.234,57 py6

関連項目

出力形式:

- “NLMNIRUBw.d 出力形式” (410 ページ)
- “NLMNLRUBw.d 出力形式”
- “NLMNACIRUBw.d 出力形式”

NLMNACLSEKw.d 出力形式

スウェーデンの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLSEKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=sv_SE;  
x=put(-1234.56789,nlmnacsek32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.69 NLMNACLSEK 出力

ロケール	フォーマットされた値
sv_SE	-1 234,57 kr
en_US	(kr1,234.57)
es_ES	-1.234,57 kr

関連項目

- 出力形式:
- [“NLMNISEKw.d 出力形式” \(412 ページ\)](#)
 - [“NLMNLSEKw.d 出力形式”](#)
 - [“NLMNACISEKw.d 出力形式”](#)

NLMNACLSGDw.d 出力形式

シンガポールの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLSGDw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_SG;  
x=put(-1234.56789,nlmlnclsgd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.70 NLMNACLSGD 出力

ロケール	フォーマットされた 値
zh_SG	(SG\$1,234.57)
en_US	(SG\$1,234.57)
es_ES	-1.234,57 SG\$

関連項目

出力形式:

- “NLMNISGDw.d 出力形式” (413 ページ)
- “NLMNLSGDw.d 出力形式”
- “NLMNACISGDw.d 出力形式”

NLMNACLTHBw.d 出力形式

タイの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLTHBw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=th_TH;  
x=put(-1234.56789,nlmlaclthb32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.71 NLMNACLTHB 出力

ロケール	フォーマットされた値
th_TH	(฿1,234.57)
en_US	(฿1,234.57)
es_ES	-1.234,57 ฿

関連項目

- 出力形式:
- [“NLMNITHBw.d 出力形式” \(414 ページ\)](#)
 - [“NLMNLTHBw.d 出力形式”](#)
 - [“NLMNACITHBw.d 出力形式”](#)

NLMNACLTRYw.d 出力形式

トルコの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになる

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLTRY*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。
出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=tr_TR;  
x=put(-1234.56789,nlmnacstry32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.72 NLMNACLTRY 出力

ロケール	フォーマットされた 値
tr_TR	(YTL1.234,57)
en_US	(YTL1,234.57)
es_ES	-1.234,57 YTL

関連項目

出力形式:

- “NLMNITRYw.d 出力形式” (416 ページ)
- “NLMNLTRYw.d 出力形式”
- “NLMNACITRYw.d 出力形式”

NLMNACLTWDw.d 出力形式

台湾の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLTWDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_TW;  
x=put(-1234.56789,nlmlnactwd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.73 NLMNACLWTD 出力

ロケール	フォーマットされた 値
zh_TW	(NT\$1,234.57)
en_US	(NT\$1,234.57)
es_ES	-1.234,57 NT\$

関連項目

出力形式:

- [“NLMNITWDw.d 出力形式” \(417 ページ\)](#)
- [“NLMNLTWDw.d 出力形式”](#)
- [“NLMNACITWDw.d 出力形式”](#)

NLMNACLUSDw.d 出力形式

プエルトリコと米国の現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNACLUSDw.d

構文の説明

w 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d 数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=es_US;  
x=put(-1234.56789,nlmnaclUSD32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.74 NLMNACLUSD 出力

ロケール	フォーマットされた 値
es_US	-US\$1,234.57
en_US	(US\$1,234.57)
es_ES	-1.234,57 US\$

関連項目

出力形式:

- “NLMNIUSDw.d 出力形式” (419 ページ)
- “NLMNLUSDw.d 出力形式”
- “NLMNACIUSDw.d 出力形式”

NLMNACLZARw.d 出力形式

南アフリカの現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNACLZAR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=af_ZA;  
x=put(-1234.56789,nlmlaclzar32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.75 NLMNACLZAR 出力

ロケール	フォーマットされた値
af_ZA	(R1 234,57)
en_US	(R1,234.57)
es_ES	-1.234,57 R

関連項目

出力形式:

- [“NLMNIZARw.d 出力形式” \(420 ページ\)](#)
- [“NLMNLZARw.d 出力形式”](#)
- [“NLMNACIZARw.d 出力形式”](#)

NLMNIAEDw.d 出力形式

アラブ首長国連邦の国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIAED*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_AE;  
x=put(-1234.56789,nlmniaed32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.76 NLMNIAED 出力

ロケール	フォーマットされた 値
ar_AE	-1.234,57 AED
en_US	-AED1,234.57
es_ES	-1.234,57 AED

関連項目

出力形式:

NLMNIAUDw.d 出力形式

オーストラリアの国際表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIAUDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_AU;  
x=put(-1234.56789,nlmniaud32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

Locale	Formatted value
en_AU	(AUD1,234.57)
en_US	(AUD1,234.57)
es_ES	-1.234,57 AUD

関連項目

出力形式:

- [“NLMNLAUDw.d 出力形式” \(423 ページ\)](#)

NLMNIBGNw.d 出力形式

ブルガリアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIBGNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2
範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=bg_BG;  
x=put(-1234.56789,nlmnibgn32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.77 NLMNIBGN 出力

ロケール	フォーマットされた値
bg_BG	-1 234,57 BGN
en_US	-BGN1,234.57
es_ES	-1.234,57 BGN

関連項目

出力形式:
■ [“NLMNLBGNw.d 出力形式” \(424 ページ\)](#)

NLMNIBRLw.d 出力形式

ブラジルの国際標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNIBRL*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=pt_BR;  
x=put(-1234.56789,nlmnibr132.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.78 NLMNIBRL 出力

ロケール	フォーマットされた 値
pt_BR	-BRL 1.234,57
en_US	-BRL1,234.57
es_ES	-1.234,57 BRL

関連項目

出力形式:

- “NLMNLBRLw.d 出力形式” (426 ページ)

NLMNICADw.d 出力形式

カナダの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICADw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_CA;  
x=put(-1234.56789,nlmcad32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.79 NLMNICAD 出力

ロケール	フォーマットされた 値
en_CA	-CAD1,234.57
en_US	-CAD1,234.57
es_ES	-1.234,57 CAD

関連項目

出力形式:

- [“NLMNLCADw.d 出力形式” \(427 ページ\)](#)

NLMNICHFw.d 出力形式

リヒテンシュタインとスイスの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICHFw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=fr_CH;  
x=put(-1234.56789,nlmnichf32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.80 NLMNICHF 出力

ロケール	フォーマットされた値
fr_CH	-1 234,57 CHF
en_US	-CHF1,234.57
es_ES	-1.234,57 CHF

関連項目

出力形式:

- [“NLMNLCHFw.d 出力形式” \(429 ページ\)](#)

NLMNICNYw.d 出力形式

中国の国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICNYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 02

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_CN;  
x=put(-1234.56789,nlmicny32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.81 NLMNICNY 出力

ロケール	フォーマットされた値
zh_CN	-CNY1,234.57
en_US	-CNY1,234.57
es_ES	-1.234,57 CNY

関連項目

出力形式:

- [“NLMNLCNYw.d 出力形式” \(430 ページ\)](#)

NLMNICZKw.d 出力形式

チェコの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICZKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=cs_CZ;  
x=put(-1234.56789,nlmcnzk32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.82 NLMNICZK 出力

ロケール	フォーマットされた値
cs_CZ	-1 234,57 CZK)
en_US	-CZK1,234.57
es_ES	-1.234,57 CZK

関連項目

出力形式:

- [“NLMNLCZKw.d 出力形式” \(431 ページ\)](#)

NLMNIDKKw.d 出力形式

デンマーク、フェロー諸島およびグリーンランドの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIDKKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=da_dk;  
x=put(-1234.56789,nlmsidkk32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.83 NLMNIDKK 出力

ロケール	フォーマットされた 値
da_dk	-1.234,57 DKK
en_US	-DKK1,234.57
es_ES	-1.234,57 DKK

関連項目

出力形式:

- [“NLMNLDKKw.d 出力形式” \(433 ページ\)](#)

NLMNIEEKw.d 出力形式

エストニアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: エストニアは通貨にユーロを使用しています。NLMNIEEK は、歴史的な目的のためにドキュメント化されています。

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEEKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=et_EE;  
x=put(-1234.56789,nlmmieek32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.84 NLMNIEEK 出力

ロケール	フォーマットされた値
et_EE	-1 234,57 EEK
en_US	-EEK1,234.57
es_ES	-1.234,57 EEK

関連項目

- 出力形式:
- [“NLMNLEEKw.d 出力形式” \(434 ページ\)](#)

NLMNIEGPw.d 出力形式

エジプトの国際標準表現の通貨出力形式を書き込みます。

カテゴリ:

CAS
数値

配置:

左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEGP*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=ar_EG;  
x=put(-1234.56789,nlmniegp32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.85 NLMNIEGP 出力

ロケール	フォーマットされた 値
ar_EG	-1.234,57 EGP
en_US	-EGP1,234.57
es_ES	-1.234,57 EGP

関連項目

出力形式:

■ “NLMNLEGPw.d 出力形式” (436 ページ)

NLMNIEURw.d 出力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEUR*w.d*

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。次の例では、LOCALE=システムオプションが de_DE に設定されています。

```
option locale=de_DE;  
x=put(-1234.56789,nlmnieur32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.86 NLMNIEUR 出力

ロケール	フォーマットされた 値
de_DE	-1.234,57 EUR
en_US	-EUR1,234.57
es_ES	-1.234,57 EUR

関連項目

出力形式:

- [“NLMNLEURw.d 出力形式” \(437 ページ\)](#)

NLMNIGBPw.d 出力形式

英国の国標標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIGBPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=en_GB;  
x=put(-1234.56789,nlmgbp32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.87 NLMNIGBP 出力

ロケール	フォーマットされた値
en_GB	-GBP1,234.57
en_US	-GBP1,234.57
es_ES	-1.234,57 GBP

関連項目

出力形式:

■ [“NLMNLGBPw.d 出力形式” \(439 ページ\)](#)

NLMNIHKDw.d 出力形式

香港の国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHKDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=en_HK;
x=put(-1234.56789,nlmnihkd32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.88 NLMNIHKD 出力

ロケール	フォーマットされた 値
en_HK	-HKD1,234.57
en_US	-HKD1,234.57
es_ES	-1.234,57 HKD

関連項目

- 出力形式:
- [“NLMNLHKDw.d 出力形式” \(440 ページ\)](#)

NLMNIHRKw.d 出力形式

クロアチアの国際標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHRKw.d

構文の説明

- w
- 出力フィールドの幅を指定します。
- デフォルト 12
- 範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=hr_HR;  
x=put(-1234.56789,nlmonihrk32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.89 NLMNIHRK 出力

ロケール	フォーマットされた値
hr_HR	-1.234,57 HRK
en_US	-HRK1,234.57
es_ES	-1.234,57 HRK

関連項目

出力形式:

- [“NLMNLHRKw.d 出力形式” \(442 ページ\)](#)

NLMNIHUFw.d 出力形式

ハンガリーの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
 数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHUFw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=hu_HU
x=put(-1234.56789,nlmnihuf32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.90 NLMNIHUF 出力

ロケール	フォーマットされた値
hu_HU	-1 234,57 HUF
en_US	-HUF1,234.57
es_ES	-1.234,57 HUF

関連項目

出力形式:

- [“NLMNLHUFw.d 出力形式” \(443 ページ\)](#)

NLMNIIDRw.d 出力形式

インドネシアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIIDR w . d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=id_ID;
x=put(-1234.56789,nlmniidr32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.91 NLMNIIDR 出力

ロケール	フォーマットされた値
id_ID	-IDR1.234,57
en_US	-IDR1,234.57
es_ES	-1.234,57 IDR

関連項目

出力形式:

- [“NLMNLIDRw.d 出力形式” \(444 ページ\)](#)

NLMNIILSw.d 出力形式

イスラエルの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIILS*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=he_IL;  
x=put(-1234.56789,nlmniils32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.92 NLMNIILS 出力

ロケール	フォーマットされた値
he_IL	-1.234,57 ILS
en_US	-IL\$1,234.57
es_ES	-1.234,57 ILS

関連項目

出力形式:

NLMNIINRw.d 出力形式

インドの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIINR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=hi_IN;  
x=put(-1234.56789,nlmniinr32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.93 NLMNIINR 出力

ロケール	フォーマットされた 値
hi_IN	-INR1,234.57
en_US	-INR1,234.57
es_ES	-1.234,57 INR

関連項目

出力形式:

- [“NLMNLINRw.d 出力形式” \(447 ページ\)](#)

NLMNIJPYw.d 出力形式

日本の国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIJPYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=ja_JP;  
x=put(-1234.56789,nlmnijpy32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.94 NLMNIJPY 出力

ロケール	フォーマットされた値
ar_AE	-JPY1,234.57
en_US	-JPY1,234.57
es_ES	-1.234,57 JPY

関連項目

出力形式:

■ [“NLMNLJPYw.d 出力形式” \(449 ページ\)](#)

NLMNIKRWw.d 出力形式

韓国の国際標準表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIKRW*w.d*

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	12
範囲	8-32
<i>d</i>	数値の小数点以下の桁数を指定します。
デフォルト	0
範囲	0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=ko_KR;  
x=put(-1234.56789,nlmmikrw32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.95 NLMNIKRW 出力

ロケール	フォーマットされた 値
ko_KR	-KRW1,234.57

ロケール	フォーマットされた 値
en_US	-KRW1,234.57
es_ES	-1.234,57 KRW

関連項目

出力形式:

- “NLMNLKRW_w.d 出力形式” (450 ページ)

NLMNILT_w.d 出力形式

リトアニアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNILT_w.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=lt_LT;  
x=put(-1234.56789,nlmltl32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.96 NLMNILTL 出力

ロケール	フォーマットされた値
lt_LT	-1 234,57 LTL
en_US	-LTL1,234.57
es_ES	-1.234,57 LTL

関連項目

出力形式:

- [“NLMNLLTLw.d 出力形式” \(451 ページ\)](#)

NLMNILVLw.d 出力形式

ラトビアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNILVLw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=lv_LV;
x=put(-1234.56789,nlmlvl32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.97 NLMNILVL 出力

ロケール	フォーマットされた値
lv_LV	-1 234,57 LVL
en_US	-LVL1,234.57
es_ES	-1.234,57 LVL

関連項目

出力形式:

- [“NLMNLLVLw.d 出力形式” \(453 ページ\)](#)

NLMNIMOPw.d 出力形式

マカオの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIMOPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=zh_MO;  
x=put(-1234.56789,nlmnimop32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.98 NLMNIMOP 出力

ロケール	フォーマットされた 値
zh_MO	-MOP1,234.57
en_US	-MOP1,234.57
es_ES	-1.234,57 MOP

関連項目

- 出力形式:
- [“NLMNLMOPw.d 出力形式” \(454 ページ\)](#)

NLMNIMXNw.d 出力形式

メキシコの国際標準表現の通貨出力形式を書き込みます。

カテゴリ:

CAS
数値

配置:

左

注:

2025.12以降、この形式はDS2言語で実行する場合にBIGINTデータ型をサポートします。BIGINTPROCESSING=YESを設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく19桁までの完全な精度で表示されるようになります。15桁または16桁を超える大きな値の場合は、BIGINTPROCESSING=YESをお勧めします。

構文

NLMNIMXNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
option locale=es_MX;  
x=put(-1234.56789,nlmmnxn32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.99 NLMNIMXN 出力

ロケール	フォーマットされた 値
es_MX	-MXN1,234.57
en_US	-MXN1,234.57
es_ES	-1.234,57 MXN

関連項目

出力形式:

- [“NLMNLMXNw.d 出力形式” \(456 ページ\)](#)

NLMNIMYRw.d 出力形式

マレーシアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIMYR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=ms_MY;  
x=put(-1234.56789,nlmnimyr32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.100 NLMNIMYR 出力

ロケール	フォーマットされた 値
ms_MY	-MYR1,234.57
en_US	-MYR1,234.57
es_ES	-1.234,57 MYR

関連項目

出力形式:

- [“NLMNLMYRw.d 出力形式” \(457 ページ\)](#)

NLMNINOKw.d 出力形式

ノルウェーの国際標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNINOK w .d

構文の説明

- w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=no_NO;  
x=put(-1234.56789,nlminok32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.101 NLMNINOK 出力

ロケール	フォーマットされた値
no_NO	NOK -1 234,57
en_US	-NOK1,234.57
es_ES	-1.234,57 NOK

関連項目

出力形式:

- [“NLMNLNOKw.d 出力形式” \(459 ページ\)](#)

NLMNINZDw.d 出力形式

ニュージーランドの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNINZD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_NZ;  
x=put(-1234.56789,nlminz32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.102 NLMNINZD 出力

ロケール	フォーマットされた 値
en_NZ	-NZD1,234.57
en_US	-NZD1,234.57
es_ES	-1.234,57 NZD

関連項目

出力形式:

- [“NLMNLNZDw.d 出力形式” \(460 ページ\)](#)

NLMNIPLNw.d 出力形式

ポーランドの国際標準表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIPLNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=pl_PL;  
x=put(-1234.56789,nlmipln32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.103 NLMNIPLN 出力

ロケール	フォーマットされた値
pl_PL	-1 234,57 PLN
en_US	-PLN1,234.57
es_ES	-1.234,57 PLN

関連項目

- 出力形式:
- [“NLMNLPLNw.d 出力形式” \(462 ページ\)](#)

NLMNIRUBw.d 出力形式

ロシアの国際標準表現の通貨出力形式を書き込みます。

カテゴリ:

CAS
数値

配置:

左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIRUBw.d

構文の説明

w
出力フィールドの幅を指定します。
デフォルト 12
範囲 8-32

d
数値の小数点以下の桁数を指定します。
デフォルト 2
範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ru_RU;  
x=put(-1234.56789,nlmmnirub32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.104 NLMNIRUB 出力

ロケール	フォーマットされた値
ru_RU	-1 234,57 RUB
en_US	-RUB1,234.57
es_ES	-1.234,57 RUB

関連項目

出力形式:

■ “NLMNLRUBw.d 出力形式” (463 ページ)

NLMNISEKw.d 出力形式

スウェーデンの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNISEK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=sv_SE;
x=put(-1234.56789,nlmnisek32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.105 NLMNISEK 出力

ロケール	フォーマットされた値
sv_SE	-1 234,57 SEK
en_US	-SEK1,234.57
es_ES	-1.234,57 SEK

関連項目

出力形式:

- [“NLMNLSEkw.d 出力形式” \(464 ページ\)](#)

NLMNISGDw.d 出力形式

シンガポールの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNISGDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_SG;  
x=put(-1234.56789,nlmnisgd32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.106 NLMNISGD 出力

ロケール	フォーマットされた 値
zh_SG	-SGD1,234.57
en_US	-SGD1,234.57
es_ES	-1.234,57 SGD

関連項目

出力形式:

- [“NLMNLSGDw.d 出力形式” \(466 ページ\)](#)

NLMNITHBw.d 出力形式

タイの国際標準表現の通貨出力形式を書き込みます。

- カテゴリ:

CAS
数値
- 配置:

左
- 注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNITHBw.d

構文の説明

- w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32
- d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=th_TH;  
x=put(-1234.56789,nlmnithb32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.107 NLMNITHB 出力

ロケール	フォーマットされた 値
th_TH	-THB1,234.57

ロケール	フォーマットされた 値
en_US	-THB1,234.57
es_ES	-1.234,57 THB

関連項目

出力形式:

- [“NLMNLTHBw.d 出力形式” \(467 ページ\)](#)

NLMNITRYw.d 出力形式

トルコの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNITRY*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=tr_TR;  
x=put(-1234.56789,nlmnitry32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.108 NLMNITRY 出力

ロケール	フォーマットされた値
tr_TR	-TRY1.234,57
en_US	-TRY1,234.57
es_ES	-1.234,57 TRY

関連項目

出力形式:

- [“NLMNLTRYw.d 出力形式” \(469 ページ\)](#)

NLMNITWDw.d 出力形式

台湾の国際標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになる

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNITWD*w.d*

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_TW;  
x=put(-1234.56789,nlmitwd32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されていま
す。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.109 NLMNITWD 出力

ロケール	フォーマットされた 値
zh-TW	-TWD1,234.57
en_US	-TWD1,234.57
es_ES	-1.234,57 TWD

関連項目

出力形式:

- [“NLMNLTWDw.d 出力形式” \(470 ページ\)](#)

NLMNIUSDw.d 出力形式

プエルトリコと米国の国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS

数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIUSD*w.d*

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=es_US;  
x=put(-1234.56789,nlmniusd32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.110 NLMNIUSD 出力

ロケール	フォーマットされた 値
es_US	-USD1,234.57
en_US	-USD1,234.57
es_ES	-1.234,57 USD

関連項目

出力形式:

- [“NLMNLUSDw.d 出力形式” \(471 ページ\)](#)

NLMNIZARw.d 出力形式

南アフリカの国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIZARw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=af_ZA;
x=put(-1234.56789,nlmnizar32.2;
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.111 NLMNIZAR 出力

ロケール	フォーマットされた値
af_ZA	-ZAR1 234,57
en_US	-ZAR1,234.57
es_ES	-1.234,57 ZAR

関連項目

出力形式:

■ [“NLMNLZARw.d 出力形式” \(473 ページ\)](#)

NLMNLAEDx.d 出力形式

アラブ首長国連邦の現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLAED*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_AE
x=put(-1234.56789,nlmnlaed32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.112 NLMNIAED 出力

ロケール	フォーマットされた値
ar_AE	-1,234.57 ل.د.
en_US	-1,234.57 ل.د.
es_ES	-1.234,57 ل.د.

関連項目

出力形式:

- [“NLMNIAEDw.d 出力形式” \(369 ページ\)](#)

NLMNLAUDw.d 出力形式

オーストラリアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLAUDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_AU;  
x=put(-1234.56789,nlmlaud32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.113 NLMNIAUD 出力

ロケール	フォーマットされた値
en_AU	-AU\$1,234.57
en_US	-AU\$1,234.57
es_ES	-1.234,57 AU\$

関連項目

出力形式:

- [“NLMNIAUDw.d 出力形式” \(371 ページ\)](#)

NLMNLBGNw.d 出力形式

ブルガリアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLBGNw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=bg_BG;
x=put(-1234.56789,nlmnlbgn32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.114 NLMNLBGN 出力

ロケール	フォーマットされた値
bg_BG	-1 234,57 лв
en_US	-лв1,234.57
es_ES	-1.234,57 лв

関連項目

出力形式:

- [“NLMNIBGNw.d 出力形式” \(372 ページ\)](#)

NLMNLBRLw.d 出力形式

ブラジルの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLBRLw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=pt_BR;
x=put(-1234.56789,nlmlbri32.2);
```

表 10.115 NLMNLBRL 出力

ロケール	フォーマットされた値
pt_BR	-R\$ 1.234,57
en_US	-R\$1,234.57
es_ES	-1.234,57 R\$

関連項目

出力形式:

- [“NLMNIBRLw.d 出力形式” \(373 ページ\)](#)

NLMNLCADw.d 出力形式

カナダの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCAD*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_CA;  
x=put(-1234.56789,nlmnlnad32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.116 NLMNLCAD 出力

ロケール	フォーマットされた 値
en_CA	-CA\$1,234.57
en_US	-CA\$1,234.57
es_ES	-1.234,57 CA\$

関連項目

出力形式:

■ “NLMNICADw.d 出力形式” (375 ページ)

NLMNLCHFw.d 出力形式

リヒテンシュタインとスイスの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCHFw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=fr_CH;  
x=put(-1234.56789,nlmnlchf32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.117 NLMNLCHF 出力

ロケール	フォーマットされた値
fr_CH	-1 234.57 SFr.
en_US	-SFr.1,234.57
es_ES	-1.234,57 SFr.

関連項目

出力形式:

- [“NLMNICHw.d 出力形式” \(376 ページ\)](#)

NLMNLCNYw.d 出力形式

中国の現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCNY w .d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_CN;  
x=put(-1234.56789,nlmlncy32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.118 NLMNLCNY 出力

ロケール	フォーマットされた 値
zh_CN	-RMB1,234.57
en_US	-RMB1,234.57
es_ES	-1.234,57 RMB

関連項目

出力形式:

■ [“NLMNICNYw.d 出力形式” \(378 ページ\)](#)

NLMNLCZKw.d 出力形式

チェコの現地標準表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCZK*w.d*

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	12
範囲	8-32
<i>d</i>	数値の小数点以下の桁数を指定します。
デフォルト	4
範囲	0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=cs_CZ;  
x=put(-1234.56789,nlmlczk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.119 NLMNLCZK 出力

ロケール	フォーマットされた 値
cs_CZ	-1 234,57 Kč

ロケール	フォーマットされた 値
en_US	-Kč1,234.57
es_ES	-1.234,57 Kč

関連項目

出力形式:

- [“NLMNICZKw.d 出力形式” \(379 ページ\)](#)

NLMNLDKKw.d 出力形式

デンマーク、フェロー諸島およびグリーンランドの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLDKKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
options locale=da_dk;  
x=put(-1234.56789,nlmlndkk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.120 NLMNLDDK 出力

ロケール	フォーマットされた値
da_dk	-1.234,57 kr
en_US	-kr1,234.57
es_ES	-1.234,57 kr

Locale	Formatted value
da_DK	-1.234,57 kr
en_US	(kr1,234.57)
es_ES	-1.234,57 kr

関連項目

- 出力形式:
- [“NLMNIDKKw.d 出力形式” \(380 ページ\)](#)

NLMNLEEKw.d 出力形式

エストニアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

- 配置: 左
- 注: エストニアは通貨にユーロを使用しています。NLMNIEEK は、歴史的な目的のためにドキュメント化されています。
2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEEKw.d

構文の説明

w
出力フィールドの幅を指定します。
デフォルト 12
範囲 8-32

d
数値の小数点以下の桁数を指定します。
デフォルト 4
範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=et_EE;  
x=put(-1234.56789,nlmnleek32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.121 NLMNLEEK 出力

ロケール	フォーマットされた 値
et_EE	-1 234,57 kr

ロケール	フォーマットされた 値
en_US	-kr1,234.57
es_ES	-1.234,57 kr

関連項目

出力形式:

- [“NLMNIEEKw.d 出力形式” \(382 ページ\)](#)

NLMNLEGPw.d 出力形式

エジプトの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEGPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 3

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ar_EG;  
x=put(-1234.56789,nlmlnlegr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.122 NLMNLEGP 出力

ロケール	フォーマットされた値
ar_EG	-1,234.57 م.ع.
en_US	-1,234.57 م.ع.
es_ES	-1.234,57 م.ع.

関連項目

出力形式:

- [“NLMNIEGPw.d 出力形式” \(383 ページ\)](#)

NLMNLEURw.d 出力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保

存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEUR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=de_DE;  
y=put(-1234.56789,nlmnleur32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.123 NLMNLEUR 出力

ロケール	フォーマットされた 値
de_DE	-1.234,57 €
en_US	-€1,234.57
es_ES	-1.234,57 €

関連項目

出力形式:

- [“NLMNIEURw.d 出力形式” \(385 ページ\)](#)

NLMNLGBPw.d 出力形式

英国の現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLGBPw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_GB;  
x=put(-1234.56789,nlmlnlgbp32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.124 NLMNLGBP 出力

ロケール	フォーマットされた値
en_GB	-£1,234.57
en_US	-\$1,234.57
es_ES	€ -1.234,57

関連項目

- 出力形式:
- [“NLMNIGBPw.d 出力形式” \(386 ページ\)](#)

NLMNLHKDw.d 出力形式

香港の現地標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHKDw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_HK;
x=put(-1234.56789,nlmnlhkd32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.125 NLMNLHKD 出力

ロケール	フォーマットされた値
en_HK	-HK\$1,234.57
en_US	-HK\$1,234.57
es_ES	-1.234,57 HK\$

関連項目

出力形式:

■ [“NLMNIHKDw.d 出力形式” \(388 ページ\)](#)

NLMNLHRKw.d 出力形式

クロアチアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHRKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=hr_HR;
x=put(-1234.56789,nlmnlhrk32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.126 NLMNLHRK 出力

ロケール	フォーマットされた 値
hr_HR	-1.234,57 Kn
en_US	-Kn1,234.57
es_ES	-1.234,57 Kn

関連項目

出力形式:

- [“NLMNIHRKw.d 出力形式” \(389 ページ\)](#)

NLMNLHUFw.d 出力形式

ハンガリーの現地標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHUFw.d

構文の説明

- w*
出力フィールドの幅を指定します。
- デフォルト 12
- 範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=hu_HU;  
x=put(-1234.56789,nlmlnhuf32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.127 NLMNLHUF 出力

ロケール	フォーマットされた値
hu_HU	-1 234,57 Ft
en_US	-Ft1,234.57
es_ES	-1.234,57 Ft

関連項目

出力形式:

- [“NLMNIHUFw.d 出力形式” \(390 ページ\)](#)

NLMNLIDRw.d 出力形式

インドネシアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLIDRw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=id_ID;  
x=put(-1234.56789,nlmnlidr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.128 NLMNLIDR 出力

ロケール	フォーマットされた値
af_ZA	-Rp1.234,57
en_US	-Rp1,234.57
es_ES	-1.234,57 Rp

関連項目

出力形式:

- [“NLMNIIDRw.d 出力形式” \(392 ページ\)](#)

NLMNLILSw.d 出力形式

イスラエルの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLILSw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=he_IL;  
x=put(-1234.56789,nlmlnls32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.129 NLMNLILS 出力

ロケール	フォーマットされた値
he_IL	-1,234.57n"ש
en_US	-1,234.57n"ש
es_ES	-1,234.57n"ש

関連項目

- 出力形式:
- [“NLMNIILSw.d 出力形式” \(393 ページ\)](#)

NLMNLINRw.d 出力形式

インドの現地標準表現の通貨出力形式を書き込みます。

カテゴリ:

CAS
数値

配置:

左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLINR*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=hi_IN;  
x=put(-1234.56789,nlmnlr32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.130 NLMNLINR 出力

ロケール	フォーマットされた値
hi_IN	-Rs1,234.57
en_US	-Rs1,234.57
es_ES	-1.234,57 Rs

関連項目

出力形式:

NLMNLJPYw.d 出力形式

日本の現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLJPYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ja_JP;  
x=put(-1234.56789,nlmnljpy32.2);
```

この表は、国際値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.131 NLMNLJPY 出力

ロケール	フォーマットされた 値
ja_JP	-¥ 1,234.57
en_US	-¥ 1,234.57
es_ES	-1.234,57 ¥

関連項目

出力形式:

- [“NLMNLJPYw.d 出力形式” \(396 ページ\)](#)

NLMNLKRWw.d 出力形式

韓国の現地標準表現の通貨出力形式を書き込みます。

- カテゴリ:
- CAS
数値
- 配置:
- 左
- 注:
- 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLKRWw.d

構文の説明

- w
- 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d

数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ko_KR;
x=put(-1234.56789,nlmlnkrw32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.132 NLMNLKRW 出力

ロケール	フォーマットされた値
ko_KR	-₩1,234.57
en_US	-\$1,234.57
es_ES	-1.234,57 ₩

関連項目

出力形式:

- [“NLMNLKRWw.d 出力形式” \(397 ページ\)](#)

NLMNLLTLw.d 出力形式

リトアニアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLLTL*w.d*

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	12
範囲	8-32
<i>d</i>	数値の小数点以下の桁数を指定します。
デフォルト	4
範囲	0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=lt_LT;  
x=put(-1234.56789,nlmlnltl32.2);
```

この表は、国際値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.133 NLMNLLTL 出力

ロケール	フォーマットされた 値
lt_LT	-1 234,57 Lt

ロケール	フォーマットされた値
en_US	-Lt1,234.57
es_ES	-1.234,57 Lt

関連項目

出力形式:

- [“NLMNILLw.d 出力形式” \(399 ページ\)](#)

NLMNLLVLw.d 出力形式

ラトビアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLLVLw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=lv_LV;  
x=put(-1234.56789,nlmlnlvl32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.134 NLMNLLVL 出力

ロケール	フォーマットされた値
lv_LV	-1 234,57 Ls
en_US	-Ls1,234.57
es_ES	-1.234,57 Ls

関連項目

出力形式:

- [“NLMNILVLw.d 出力形式” \(400 ページ\)](#)

NLMNLMOPw.d 出力形式

マカオの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようにな

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES
をお勧めします。

構文

NLMNLMOP*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプション
を設定できます。

```
Options locale=zh_MO;  
x=put(-1234.56789,nlmnlmop32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されて
います。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.135 NLMNLMOP 出力

ロケール	フォーマットされた 値
zh_MO	-P1,234.57
en_US	-P1,234.57
es_ES	-1.234,57 P

関連項目

出力形式:

- [“NLMNIMOPw.d 出力形式” \(402 ページ\)](#)

NLMNLMXNw.d 出力形式

メキシコの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLMXNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=es_MX;
x=put(-1234.56789,nlmlmxn32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.136 NLMNLMXN 出力

ロケール	フォーマットされた値
es_MX	-MX\$1,234.57
en_US	-MX\$1,234.57
es_ES	-1.234,57 MX\$

関連項目

出力形式:

- [“NLMNIMXNw.d 出力形式” \(403 ページ\)](#)

NLMNLMYRw.d 出力形式

マレーシアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLMYRw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ms_MY;  
x=put(-1234.56789,nlmlnmyr32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.137 NLMNLMYR 出力

ロケール	フォーマットされた 値
ms_MY	-R1,234.57
en_US	-R1,234.57
es_ES	-1.234,57 R

関連項目

出力形式:

- [“NLMNIMYRw.d 出力形式” \(405 ページ\)](#)

NLMNLNOKw.d 出力形式

ノルウェーの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLNOKw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=no_NO;  
x=put(-1234.56789,nlmnl nok32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.138 NLMNLNOK 出力

ロケール	フォーマットされた 値
no_NO	kr -1 234,57
en_US	-kr1,234.57
es_ES	-1.234,57 kr

関連項目

出力形式:

- [“NLMNINOKw.d 出力形式” \(406 ページ\)](#)

NLMNLNZDw.d 出力形式

ニュージーランドの現地標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLNZDw.d

構文の説明

- w
出力フィールドの幅を指定します。
- デフォルト 12
- 範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=en_NZ;  
x=put(-1234.56789,nlmlnlnzd32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.139 NLMNZNZD 出力

ロケール	フォーマットされた値
en_NZ	-NZ\$1,234.57
en_US	-NZ\$1,234.57
es_ES	-1.234,57 NZ\$

Locale	Formatted value
en_NZ	(NZ\$1,234.57)
en_US	(NZ\$1,234.57)
es_ES	-1.234,57 NZ\$

関連項目

出力形式:

■ [“NLMNZNZDw.d 出力形式” \(407 ページ\)](#)

NLMNLPLNw.d 出力形式

ポーランドの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLPLNw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=pl_PL;  
x=put(-1234.56789,nlmnlpln32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.140 NLMNLPLN 出力

ロケール	フォーマットされた 値
pl_PL	-1 234,57 zł
en_US	-zł1,234.57
es_ES	-1.234,57 zł

関連項目

出力形式:

- [“NLMNIPLNw.d 出力形式” \(409 ページ\)](#)

NLMNLRUBw.d 出力形式

ロシアの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLRUBw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=ru_RU;  
x=put(-1234.56789,nlmlnrub32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.141 NLMNLRUB 出力

ロケール	フォーマットされた 値
ru_RU	-1 234,57 py6
en_US	-py61,234.57
es_ES	-1.234,57 py6

関連項目

出力形式:

■ [“NLMNIRUBw.d 出力形式” \(410 ページ\)](#)

NLMNLSEKw.d 出力形式

スウェーデンの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
 数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLSEK*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=sv_SE;  
x=put(-1234.56789,nlmnlsek32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.142 NLMNLSEK 出力

ロケール	フォーマットされた 値
sv_SE	-1 234,57 kr
en_US	-kr1,234.57
es_ES	-1.234,57 kr

関連項目

出力形式:

- [“NLMNISEKw.d 出力形式” \(412 ページ\)](#)

NLMNLSGDw.d 出力形式

シンガポールの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINTとして保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLSGDw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_SG;  
x=put(-1234.56789,nlmlsgd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.143 NLMNLSGD 出力

ロケール	フォーマットされた値
zh_SG	-SG\$1,234.57
en_US	-SG\$1,234.57
es_ES	-1.234,57 SG\$

関連項目

- 出力形式:
- [“NLMNISGDw.d 出力形式” \(413 ページ\)](#)

NLMNLTHBw.d 出力形式

タイの現地標準表現の通貨出力形式を書き込みます。

カテゴリ:

CAS
数値

配置:

左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTHB*w.d*

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=th_TH;  
x=put(-1234.56789,nlmnlthb32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.144 NLMNLTHB 出力

ロケール	フォーマットされた値
th_TH	-฿1,234.57
en_US	-\$1,234.57
es_ES	-1.234,57 ¢

関連項目

出力形式:

NLMNLTRYw.d 出力形式

トルコの現地標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTRYw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 4

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=tr_TR;  
x=put(-1234.56789,nlmlntry32.2);
```

この表は、ローカル値に対する出力を示しています。3つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.145 NLMNLTRY 出力

ロケール	フォーマットされた 値
tr_TR	-YTL1.234,57
en_US	-YTL1,234.57
es_ES	-1.234,57 YTL

関連項目

出力形式:

- [“NLMNITRYw.d 出力形式” \(416 ページ\)](#)

NLMNLTWDw.d 出力形式

台湾の現地標準表現の通貨出力形式を書き込みます。

- カテゴリ:
- CAS
数値
- 配置:
- 左
- 注:
- 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTWD*w.d*

構文の説明

- w*
- 出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32

d
数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=zh_TW;  
x=put(-1234.56789,nlmlntwd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.146 NLMNLTWD 出力

ロケール	フォーマットされた 値
zh_TW	-NT\$1,234.57
en_US	-NT\$1,234.57
es_ES	-1.234,57 NT\$

関連項目

出力形式:

- [“NLMNITWDw.d 出力形式” \(417 ページ\)](#)

NLMNLUSDw.d 出力形式

プエルトリコと米国の現地標準表現の通貨出力形式を書き込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLUSD_{w,d}

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	12
範囲	8-32
<i>d</i>	数値の小数点以下の桁数を指定します。
デフォルト	2
範囲	0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=es_US;  
x=put(-1234.56789,nlmnlusd32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.147 NLMNLUSD 出力

ロケール	フォーマットされた 値
es_US	-US\$1,234.57

ロケール	フォーマットされた 値
en_US	-US\$1,234.57
es_ES	-1.234,57 US\$

関連項目

出力形式:

- [“NLMNIUSDw.d 出力形式” \(419 ページ\)](#)

NLMNLZARw.d 出力形式

南アフリカの現地標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLZARw.d

構文の説明

- w

出力フィールドの幅を指定します。

デフォルト 12

範囲 8-32
- d

数値の小数点以下の桁数を指定します。

デフォルト 2

範囲 0-28

例

この例は UTF-8 SAS セッションで実行されます。

出力値はロケールによって異なります。このコードを使用してロケールオプションを設定できます。

```
Options locale=af_ZA;  
x=put(-1234.56789,nlmlzar32.2);
```

この表は、ローカル値に対する出力を示しています。3 つのロケールが指定されています。出力形式の通貨/ロケール、en_US、および es_ES。

表 10.148 NLMLNZAR 出力

ロケール	フォーマットされた値
af_ZA	-R1 234,57
en_US	-R1,234.57
es_ES	-1.234,57 R

関連項目

- 出力形式:
- [“NLMNIZARw.d 出力形式” \(420 ページ\)](#)

NLMNYw.d 出力形式

現在の SAS ロケールに適した現地通貨を使用して、現地標準表現の通貨出力形式を書き込みます。

- カテゴリ: CAS
数値
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。

ります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNYw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 1-32

d

数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-31

詳細

NLMNYw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLMNYw.d 出力形式では、そのロケールでの通貨記号、桁区切り文字および小数点区切り文字を使用して、標準パターンで数値を書き込みます。

注: NLMNYw.d 出力形式では、通貨出力形式は変換されません。したがって、フォーマットされた数値は、現在のロケールの通貨が使用された値と等しくなります。

比較

The NLMNYw.d 出力形式、NLMNYACw.d 出力形式、NLMNYIw.d 出力形式、および NLMNYIACw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して通貨出力形式を書き込みます。ただし、NLMNYIACw.d 出力形式と NLMNYACw.d 出力形式は会計形式を使用するのにに対し、NLMNYw.d 出力形式と NLMNYIw.d 出力形式は標準形式を使用します。NLMNYI w.d 出力形式と NLMNYIAC w.d 出力形式では USD などの 3 文字の国際通貨コードを使用するのにに対し、NLMNYw.d 出力形式と NLMNYACw.d 出力形式では\$などの現地通貨記号を使用します。

NLMNYw.d 出力形式は DOLLARw.d 出力形式と似ていますが NLMNYw.d 出力形式がロケール固有である点が違います。

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put(-1234.56789,nlmnyi32.2);
y=put(-1234.56789,nlmny32.2);
x2=put(-1234.56789,nlmnyiac32.2);
y2=put(-1234.56789,nlmnyac32.2);
z=put(-1234.56789,dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-USD1,234.57
put y=;	\$-1,234.57
put x2=;	(USD1,234.57)
put y2=;	(\$1,234.57)
put z=;	\$-1,234.57

関連項目

出力形式:

- [“NLMNYIw.d 出力形式” \(478 ページ\)](#)

入力形式:

- [“NLMNYw.d 入力形式” \(888 ページ\)](#)
- [“NLMNYIw.d 入力形式” \(890 ページ\)](#)

NLMNYACw.d 出力形式

現在の SAS ロケールに適した現地通貨を使用して、現地会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

構文

NLMNYACw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 1-32

d

数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-31

詳細

NLMNYACw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLMNYACw.d 出力形式では、そのロケールでの通貨記号、桁区切り文字および小数点区切り文字を使用して、会計形式で数値を書き込みます。

注: NLMNYACw.d 出力形式では、通貨出力形式は変換されません。したがって、フォーマットされた数値は、現在のロケールの通貨が使用された値と等しくなります。

比較

The NLMNYw.d 出力形式、NLMNYACw.d 出力形式、NLMNYIw.d 出力形式、および NLMNYIACw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して通貨出力形式を書き込みます。ただし、NLMNYIACw.d 出力形式と NLMNYACw.d 出力形式は会計形式を使用するのに対し、NLMNYw.d 出力形式と NLMNYIw.d 出力形式は標準形式を使用します。NLMNYI w.d 出力形式と NLMNYIAC w.d 出力形式では USD などの 3 文字の国際通貨コードを使用するのに対し、NLMNYw.d 出力形式と NLMNYACw.d 出力形式では\$などの現地通貨記号を使用します。

NLMNYACw.d 出力形式は DOLLARw.d 出力形式と似ていますが NLMNYACw.d 出力形式がロケール固有で会計形式を使用する点が違います。

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put(-1234.56789,nlmnyi32.2);
y=put(-1234.56789,nlmny32.2);
x2=put(-1234.56789,nlmnyiac32.2);
y2=put(-1234.56789,nlmnyac32.2);
z=put(-1234.56789,dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-USD1,234.57
put y=;	-\$1,234.57
put x2=;	(USD1,234.57)
put y2=;	(\$1,234.57)
put z=;	\$-1,234.57

関連項目

出力形式:

- [“NLMNYIw.d 出力形式” \(478 ページ\)](#)
- [“NLMNYw.d 出力形式”](#)
- [“NLMNYIACw.d 出力形式”](#)

入力形式:

- [“NLMNYw.d 入力形式” \(888 ページ\)](#)
- [“NLMNYIw.d 入力形式” \(890 ページ\)](#)

NLMNYIw.d 出力形式

現在の SAS ロケールに適応するように、国際標準表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値
配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNYIw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 1-32

d

数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-31

詳細

NLMNYIw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLMNYI *w* . *d* 出力形式は、国際通貨コード、およびロケールに依存する桁区切り文字と小数点区切り文字を使用して、標準形式で数値を書き込みます。国際通貨コードの位置もロケールに応じて決まります。

注: NLMNYIw.d 出力形式では、通貨出力形式は変換されません。したがって、フォーマットされた数値は、現在のロケールの通貨が使用された値と等しくなります。

比較

NLMNYw.d 出力形式、NLMNYIACw.d 出力形式、および NLMNYIw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して通貨出力形式を書き込みます。ただし、NLMNYIACw.d 出力形式と NLMNYACw.d 出力形式は会計形式を使用するのに対し、NLMNYw.d 出力形式と NLMNYIw.d 出力形式は標準形式を使用します。NLMNYI *w* . *d* 出力形式と NLMNYIAC *w* . *d* 出力形式では USD などの 3 文字の国際通貨コードを使用するのに対し、NLMNYw.d 出力形式と NLMNYACw.d 出力形式では \$ などの現地通貨記号を使用します。

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put(-1234.56789,nlmnyi32.2);
y=put(-1234.56789,nlmny32.2);
x2=put(-1234.56789,nlmnyiac32.2);
y2=put(-1234.56789,nlmnyac32.2);
z=put(-1234.56789,dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-USD1,234.57
put y=;	-\$1,234.57
put x2=;	(USD1,234.57)
put y2=;	(\$1,234.57)
put z=;	\$-1,234.57

関連項目

出力形式:

- [“NLMNYw.d 出力形式” \(474 ページ\)](#)

入力形式:

- [“NLMNYw.d 入力形式” \(888 ページ\)](#)
- [“NLMNYIw.d 入力形式” \(890 ページ\)](#)

NLMNYIACw.d 出力形式

現在の SAS ロケールに適応するように、国際会計表現の通貨出力形式を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: この出力形式は、2023.11 以降で使用できます。

構文

NLMNYIACw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 9

範囲 1-32

d

数値の小数点以下の桁数を指定します。

デフォルト 0

範囲 0-31

詳細

NLMNYIACw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLMNYIACw.d 出力形式では、国際通貨コード、およびロケールに依存する桁区切り文字と小数点区切り文字を使用して数値を書き込みます。国際通貨コードの位置もロケールに応じて決まります。

注: NLMNYIACw.d 出力形式では、通貨出力形式は変換されません。したがって、フォーマットされた数値は、現在のロケールの通貨が使用された値と等しくなります。

比較

NLMNYw.d 出力形式、NLMNYACw d 出力形式、NLMNYIACw d 出力形式、および NLMNYIw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して通貨出力形式を書き込みます。ただし、NLMNYIAC w.d および NLMNYAC w.d 出力形式は会計形式を使用するのに対し、NLMNY w.d および NLMNYI w.d 出力形式は標準形式を使用します。NLMNYI w.d および NLMNYIAC w.d 出力形式では USD などの 3 文字の国際通貨コードを使用するのに対し、NLMNY w.d および NLMNYAC w.d 出力形式では、\$などの現地通貨記号を使用します。

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put(-1234.56789,nlmnyi32.2);
y=put(-1234.56789,nlmny32.2);
x2=put(-1234.56789,nlmnyiac32.2);
y2=put(-1234.56789,nlmnyac32.2);
z=put(-1234.56789,dollar32.2);
```

ステートメント	結果
put x=;	-USD1,234.57
put y=;	-\$1,234.57
put x2=;	(USD1,234.57)
put y2=;	(\$1,234.57)
put z=;	\$-1,234.57

関連項目

出力形式:

- [“NLMNYw.d 出力形式” \(474 ページ\)](#)
- [“NLMNYACw.d 出力形式”](#)
- [“NLMNYIw.d 出力形式”](#)

入力形式:

- [“NLMNYw.d 入力形式” \(888 ページ\)](#)
- [“NLMNYIw.d 入力形式” \(890 ページ\)](#)

NLNUMw.d 出力形式

現在の SAS ロケールに適応するように、数値をローカル表示形式で書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLNUMw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 1-32

d

10^d で数値を除算するよう指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLNUMw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLNUMw.d 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。

比較

NLNUMw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して数値を書き込みます。NLNUMiw.d 出力形式では、桁区切り文字としてカンマ(,)を、小数点区切り文字としてピリオド(.)を使用して数値を書き込みます。

w 値または d 値が、フォーマットされた数値を生成するほど十分に大きくない場合、NLNUMw.d 出力形式では、小数精度が若干失われるとしても、可能な限り桁区切り文字を出力するアルゴリズムを使用します。

例

```
x=put(-1234356.7891,nlNum32.2);
```

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates;	-1,234,356.79

ステートメント	結果
put x=;	
options LOCALE=German_Germany; put x=;	-1.234.356,79

関連項目

出力形式:

- [“NLNUMIw.d 出力形式” \(484 ページ\)](#)

入力形式:

- [“NLNUMw.d 入力形式” \(892 ページ\)](#)
- [“NLNUMIw.d 入力形式” \(893 ページ\)](#)

NLNUMIw.d 出力形式

現在の SAS ロケールに適応するように、数値を国際表示形式で書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLNUMIw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 6

範囲 1-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLNUMIw.*d* 入力形式では、2 の補数で表される負の値を含む、バイナリ整数(固定小数点)値を読み込みます。NLNUMIw.*d* 出力形式では、すべてのロケールに対し、桁区切り文字としてカンマ(,)を、また小数点区切り文字としてピリオド(.)を使用して数値を書き込みます。

比較

NLNUMIw.*d* 出力形式では、指定されたロケールで、数値データを国際表示形式で書き込みます。NLNUMIw.*d* 出力形式では、桁区切り文字としてカンマ(,)を、小数点区切り文字としてピリオド(.)を使用して数値を書き込みます。

w 値または *d* 値が、フォーマットされた数値を生成するほど十分に大きくない場合、NLNUMw.*d* 出力形式では、小数精度が若干失われるとしても、可能な限り桁区切り文字を出力するアルゴリズムを使用します。

例

x=put(-1234356.7891,nlnumi32.2);

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates; put x=;	-1,234,356.79
options LOCALE=German_Germany; put x=;	-1,234,356.79

関連項目

出力形式:

- “NLNUMw.d 出力形式” (482 ページ)

入力形式:

- “NLNUMw.d 入力形式” (892 ページ)
- “NLNUMIw.d 入力形式” (893 ページ)

NLPCTw.d 出力形式

現在の SAS ロケールに適応するように、百分率のデータを現地表現で書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 4-32

d

10^d で数値を除算するよう指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

比較

NLPCTw.d 出力形式では、指定されたロケールで、百分率のデータを現地表現で書き込みます。NLPCTw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して百分率値を書き込みます。NLPCTIw.d 出力形式では、桁区切り文字としてカンマ(,)を、また小数点区切り文字としてピリオド(.)を使用して百分率値を書き込みます。

NLPCTw.d 出力形式は PERCENTw.d 出力形式に似ていますが、NLPCTw.d 出力形式がロケール固有である点が違います。

例

```
x=put(-12.3456789,nlpct32.2);
y=put(-12.3456789,nlpcti32.2);
z=put(-12.3456789,percent32.2);
```

ステートメント	結果
	----+-----1
options LOCALE=English_UnitedStates;	-1,234.57%
put x=;	-1,234.57%
put y=;	(1234.57%)
put z=;	
options LOCALE=German_Germany;	-1.234,57%
put x=;	-1,234.57%
put y=;	(1234.57%)
put z=;	

関連項目

出力形式:

- [“NLPCTIw.d 出力形式” \(488 ページ\)](#)

入力形式:

- [“NLPCTw.d 入力形式” \(895 ページ\)](#)
- [“NLPCTIw.d 入力形式” \(897 ページ\)](#)

NLPCTIw.d 出力形式

現在の SAS ロケールに適応するように、百分率のデータを国際表現で書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTIw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 4-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

比較

NLPCTIw.d 出力形式では、指定されたロケールで、百分率のデータを国際表現で書き込みます。NLPCTw.d 出力形式では、ロケールに依存する桁区切り文字と小数点区切り文字を使用して百分率値を書き込みます。NLPCTIw.d 出力形式では、桁区切り文字としてカンマ(,)を、また小数点区切り文字としてピリオド(.)を使用して百分率値を書き込みます。

NLPCTw.d 出力形式は PERCENTw.d 出力形式に似ていますが、NLPCTw.d 出力形式がロケール固有である点が違います。

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put(-12.3456789,nlpcti32.2);
y=put(-12.3456789,percent32.2);
```

ステートメント	結果
	----+-----1
put x=;	-1,234.57%
put y=;	(1234.57)

関連項目

- 出力形式:
- [“NLPCTw.d 出力形式” \(486 ページ\)](#)
- 入力形式:
- [“NLPCTw.d 入力形式” \(895 ページ\)](#)
 - [“NLPCTIw.d 入力形式” \(897 ページ\)](#)

NLPCTNw.d 出力形式

現在の SAS ロケールに適應するように、負の値にはマイナス記号を使用して、百分率を書き込みます。

カテゴリ: CAS
 数値

配置: 右

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTN*w.d*

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 4-32

ヒント 出力フィールドの幅数には、数値が負か正かにかかわらず、マイナス記号(-)、パーセント記号(%)、末尾の空白分を含める必要があります。

d

数値の小数点以下の桁数を指定します。この引数はオプションです。

デフォルト 0

範囲 0-31

要件 *w* より小さい数にします。

詳細

NLPCTN*w.d* 出力形式は、100 を乗算した負の値の先頭にマイナス記号を追加し、フォーマットした値の最後にパーセント記号(%)を追加します。

例

x=-0.02;

ステートメント	結果
put x nlpctn6.;	x=-2%
put x percentn6.;	x=-2%

NLPCTP*w.d* 出力形式

ロケール固有の数値を、現在の SAS ロケールに適したパーセントとして書き込みます。

カテゴリ:	CAS 数値
配置:	右
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTP $w.d$

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 6

範囲 4-32

ヒント 出力フィールドの幅数には、パーセント記号(%)分を含める必要があります。

d

数値の小数点以下の桁数を指定します。この引数はオプションです。NLPCTP 出力形式の桁区切り文字と小数点の記号は、ロケールに固有です。

デフォルト 0

範囲 0-31

要件 w より小さい数にします。

詳細

NLPCTP $w.d$ 出力形式は、100 を乗算してフォーマットした値の最後にパーセント記号(%)を追加します。NLPCTP $w.d$ 出力形式は PERCENT $w.d$ 出力形式に似ていますが NLPCTP $w.d$ 出力形式の桁区切り文字と小数点の記号がロケール固有である点が違います。

例

$x=-0.02;$

ステートメント	結果
put x nlpctp6.;	-2%
put x percent6.;	(2%)

NLPVALUEw.d 出力形式

現在の SAS ロケールに適応するように、p 値をローカル表示形式で書き込みます。

カテゴリ:	CAS
	数値
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPVALUEw.d

構文の説明

<i>w</i>	出力フィールドの幅を指定します。
デフォルト	6
範囲	3-32
<i>d</i>	10 ^{<i>d</i>} で数値を除算するよう指定します。データに小数点区切り文字が含まれる場合、 <i>d</i> 値は無視されます。
デフォルト	4
範囲	1-30

例

この例では、german_Germany ロケールオプションを使用します。

ステートメント:

```
options locale=german_germany;
data _null_;
  put "+--- nlpvalue min=3 default=6 max=32 ---+";
  x=0.1248;
  put x= +5 x pvalue. +5 x nlpvalue.;
  put x= +5 x pvalue3.1 +5 x nlpvalue3.1;
  put x= +5 x pvalue20.2 +5 x nlpvalue20.2;
  put x= +5 x pvalue32.3 +5 x nlpvalue32.3;
run;
```

結果:

```
+--- nlpvalue min=3 default=6 max=32 ---+
x=0.1248 0.1248 0,1248
x=0.1248 0.1 0,1
x=0.1248 0.12 0,12
x=0.1248 0.125 0,125
```

関連項目

出力形式:

- [“PVALUEw.d 出力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)

NLSTRMONw.d 出力形式

現在の SAS ロケールに適応するように、月名を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: [2025.12](#) 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLSTRMON*w.d*

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 20

範囲 1-200

d

次の項目を指定します。

- 00000001:短縮形式で書き込みます。
- 00000010:大文字形式で書き込みます。

デフォルト 0

範囲 0-3

詳細

NLSTRMON*w.d* 出力形式では、指定されたロケールの月名として 1 から 12 の SAS 値を書き込みます。次の例では、English_UnitedStates ロケールを使用します。

- 1 =第 1 月(January)
- 2 =第 2 月(February)
- 3 =第 3 月(March)
- 4 =第 4 月(April)
- 5 =第 5 月(May)
- 6 =第 6 月(June)
- 7 =第 7 月(July)
- 8 =第 8 月(August)
- 9 =第 9 月(September)
- 10 =第 10 月(October)
- 11 =第 11 月(November)
- 12 =第 12 月(December)

例

この例では、English_UnitedSates セッションエンコーディングを使用します。

```
data _null_ ;
  monnum = 1; /* January=1, December=12 */
  put monnum NLSTRMON20.;
  put monnum NLSTRMON20.1; /* decimal .1 specified use abbreviation. */
  put monnum NLSTRMON20.2;
  put monnum NLSTRMON20.3;
run;
```

```
January
Jan
JANUARY
JAN
```

NLSTRQTRw.d 出力形式

現在の SAS ロケールに適した四半期の形式で数値を書き込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLSTRQTRw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 20

範囲 1-200

d

次の項目を指定します。

- 00000001:短縮形式で書き込みます。

- 00000010:大文字形式で書き込みます。

デフォルト 0

範囲 0-3

詳細

NLSTRQTRw.d 出力形式では、指定されたロケールの四半期名として 1 から 4 の SAS 値を書き込みます。次の例では、English_UnitedStates ロケールを使用します。

- 1 = 1st quarter
- 2 = 2nd quarter
- 3 = 3rd quarter
- 4 = 4th quarter

例

この例では、English_UnitedSates セッションエンコーディングを使用します。

ステートメント	結果
Data _null_ ;	
qtrnum = 1 ; /* January=1,	1st quarter
December=12 */	Q1
put qtrnum NLSTRQTR20.;	1ST QUARTER
put qtrnum NLSTRQTR20.1; /*	Q1
decimal .1 specified use abbreviation.*/	
put qtrnum NLSTRQTR20.2;	
put qtrnum NLSTRQTR20.3; run;	

NLSTRWKw.d 出力形式

現在の SAS ロケールに適した曜日の形式で数値を書き込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLSTRWKw.d

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 20

範囲 1-200

d

次の項目を指定します。

- 00000001:短縮形式で書き込みます。
- 00000010:大文字形式で書き込みます。

デフォルト 0

範囲 0-3

詳細

NLSTRWKw.d 出力形式では、指定されたロケールの曜日名として 1 から 7 までの SAS 値を書き込みます。次の例では、English_UnitedStates ロケールを使用します。

- 1 =週の 1 日目(Monday)
- 2 =週の 2 日目(Tuesday)
- 3 =週の 3 日目(Wednesday)
- 4 =週の 4 日目(Thursday)
- 5 =週の 5 日目(Friday)
- 6 =週の 6 日目(Saturday)
- 7 =週の 7 日目(Sunday)

例

この例では、English_UnitedSates セッションエンコーディングを使用します。

ステートメント	結果
Data _null_ ;	
wknum = 1 ; /* Sunday=1, Saturday=7	Sunday
*/	Sun
put wknum NLSTRWK20.;	SUNDAY
put wknum NLSTRWK20.1; /*	SUN
decimal .1 specified use abbreviation.*/	
put wknum NLSTRWK20.2;	
put wknum NLSTRWK20.3;	
run;	

NLTIMAPw. 出力形式

SAS 時刻値を現在の SAS ロケールに適した a.m.または p.m.付きの時刻値として書き込みます。

- カテゴリ: CAS
日付と時間
- 配置: 左
- 注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMAPw.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 10

範囲 4-200

詳細

デフォルトの幅を使用した NLTIMAP 出力形式では、エンコーディングとロケールの組み合わせによっては日時名が長くなるため、正確でない地域化された出力が生成される場合があります。ロケールとエンコーディングの組み合わせに対する推奨される幅については、[日付と時間のデフォルト幅の例外 \(100 ページ\)](#) を参照してください。推奨される幅を使用するようにします。

NLTIMAP は日時値も書き込みます。

比較

NLTIMAPw.出力形式は TIMEAMPMw.出力形式と似ていますが NLTIMAPw.出力形式がロケール固有である点が違います。

例

これらの例では、59083 を入力値として使用します。この値は、午後 04:24:43 に相当する SAS 日時値です。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; put time nltimap.;	4:24:43 PM
options locale=German_Germany; put time nltimap14.;	4:24:43 nachm

関連項目

出力形式:

- [“NLTIMEw. 出力形式” \(499 ページ\)](#)

NLTIMEw. 出力形式

SAS 時刻値を現在の SAS ロケールに適した時刻値として書き込みます。NLTIME は、SAS 日時値も書き込みます。

カテゴリ:	CAS 日付と時間
配置:	左
注:	2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIME_w.

構文の説明

w	入力幅を指定します。
デフォルト	20
範囲	10-200

比較

NLTIME_w.出力形式は TIME_w.出力形式と似ていますが NLTIME_w.出力形式がロケール固有である点が違います。

例

これらの例では、59083 を入力値として使用します。この値は、午後 04:24:43 に相当する SAS 日時値です。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; put time ntime.;	4:24:43
options locale=German_Germany; put time ntime.;	16.24

関連項目

出力形式:

- [“NLTIMAPw. 出力形式” \(498 ページ\)](#)

NLTIMELw. 出力形式

SAS 時間値を、現在の SAS ロケールに適した長い形式で時間値として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMELw.

オプション引数

w

入力幅を指定します。必要に応じて、出力形式の幅に合うように時間が短縮されます。

デフォルト 30

範囲 10-200

詳細

NLTIMELw.出力形式は、時間を長い形式で書き込みます。

例

例 1

これらの例では、59083 を入力値として使用します。この値は、午後 04:24:43 に相当する SAS 日時値です。

```
data a;
options locale=English_UnitedStates;
options timezone='America/Chicago';
time=59083;
put time nltimel.;
put time nltimel10.;
put time nltimel200.;
run;
```

```
04:24:43 PM CDT
04:24 PM
04:24:43 PM CDT
```

例 2

```
data b;
options locale=German_Germany;
options timezone='Europe/Berlin';
time=59083;
put time nltimel.;
put time nltimel10.;
put time nltimel12.;
put time nltimel18.;
put time nltimel200.;
run;
```

```
16:24:43 CEST
16:24:43
16:24:43
16:24:43 CEST
16:24:43 CEST
```

例 3

```
data c;
options locale=English_UnitedStates;
options timezone='Etc/GMT-6';
time=59083;
put time nltimel.;
put time nltimel10.;
```

```
put time nltimel200.;  
run;
```

```
04:24:43 PM GMT+6  
04:24 PM  
04:24:43 PM GMT+6
```

NLTIMEMw. 出力形式

SAS 時間値を、現在の SAS ロケールに適した短縮形を使用した簡略形式で時間値として書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMEMw.

オプション引数

w

入力幅を指定します。必要に応じて、出力形式の幅に合うように時間が短縮されます。

デフォルト 14

範囲 10-200

詳細

NLTIMEMw.出力形式は、時間を短縮形を使用した簡略形式で書き込みます。

例

これらの例では、59083 を入力値として使用します。この値は、午後 04:24:43 に相当する SAS 日時値です。

```
data a;
options locale=English_UnitedStates;
time=59083;
put time nltimem.;
put time nltimem10.;
put time nltimem200.;
run;
```

```
04:24:43 PM
04:24 PM
04:24:43 PM
```

```
data b;
options locale=German_Germany;
time=59083;
put time nltimem.;
put time nltimem10.;
put time nltimem12.;
put time nltimem18.;
put time nltimem200.;
run;
```

```
16:24:43
16:24:43
16:24:43
16:24:43
16:24:43
```

NLTIMESw. 出力形式

SAS 時間値を、現在の SAS ロケールに適した短い形式で時間値として書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMESw.

オプション引数

w.

入力幅を指定します。必要に応じて、出力形式の幅に合うように時間が短縮されます。

デフォルト 11

範囲 10–200

詳細

NLTIMESw.出力形式は、時間を短い形式で書き込みます。

例

これらの例では、59083 を入力値として使用します。この値は、午後 04:24:43 に相当する SAS 日時値です。

```
data a;
options locale=English_UnitedStates;
time=59083;
put time nltimes;
put time nltimes10.;
put time nltimes200.;
run;
```

```
04:24 PM
04:24 PM
04:24 PM
```

```
data b;
options locale=German_Germany;
time=59083;
put time nltimes;
put time nltimes10.;
put time nltimes12.;
put time nltimes18.;
put time nltimes200.;
run;
```

```
16:24
16:24
16:24
16:24
16:24
```

\$UCS2Bw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、ビッグエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2Bw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2Bw.出力形式では、ビッグエンディアン、16 ビット、UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を書き込みます。現在の SAS セッションのエンコーディングを使用する文字列を処理します。

比較

\$UCS2Bw.出力形式では、\$UCS2BEw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
data _null_;	5927
x = '大';	
y=put(x,\$ucs2b2.);	
put y \$hex.;	
run;	

関連項目

出力形式:

- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)
- ["\\$UCS2BEw. 出力形式" \(507 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2BEw. 入力形式" \(906 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS2BEw. 出力形式

ビッグエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2BEw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2BEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。ビッグエンディアン、16 ビット、UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS2BEw.出力形式では、\$UCS2Bw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
x='592700410042'x; put x \$ucs2be.;	大 AB

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)

入力形式:

- "\$UCS2Bw. 入力形式" (904 ページ)
- "\$UCS2BEw. 入力形式" (906 ページ)

\$UCS2Lw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、リトルエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ:	文字
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2Lw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2Lw.出力形式では、リトルエンディアン、16 ビット、UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を書き込みます。現在の SAS セッションのエンコーディングを使用する文字列を処理します。

比較

\$UCS2Lw.出力形式では、\$UCS2LEw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
data _null_;	2759
x = '大';	
y=put(x,\$ucs2l2.);	
put y \$hex.;	
run;	

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2LEw. 出力形式" \(510 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2LEw. 入力形式" \(908 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS2LEw. 出力形式

リトルエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2LEw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2LEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。リトルエンディアン、16 ビット、UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS2LEw.出力形式では、\$UCS2Lw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
x='275941004200'x; put x \$ucs2le.;	大 AB

関連項目

出力形式:

- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)

入力形式:

- “\$UCS2Lw. 入力形式” (907 ページ)
- “\$UCS2LEw. 入力形式” (908 ページ)

\$UCS2Xw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、ネイティブエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ:	文字
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2Xw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2Xw.出力形式では、動作環境のバイトオーダーで、16 ビットの UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を書き込みます。

比較

\$UCS2Xw.出力形式では、\$UCS2XEw.出力形式とは逆の処理を行います。同一の動作環境内のデータを変換する場合は、\$UCS2Xw 出力形式を使用します。異なる動作環境でデータを変換する場合は、\$UCS2Bw.出力形式または\$UCS2Lw.出力形式を使用します。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x = '大'; put x \$ucs2x2.;</pre>	'5927'x (binary) or '2759'x (little endian)

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2XEw. 出力形式" \(513 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UCS2XEw. 入力形式" \(911 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS2XEw. 出力形式

ネイティブエンディアン、16 ビット、UCS2 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS2XEw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

詳細

\$UCS2XEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。ネイティブエンディアン、16 ビット、UCS2 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS2XEw.出力形式では、\$UCS2Xw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x='e5a4a7'x; /* Japanese '大' in UTF8 */; put x \$utf8xe10.;</pre>	大

関連項目

出力形式:

- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)

入力形式:

- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UCS2xEw. 入力形式" \(911 ページ\)](#)

\$UCS4Bw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、ビッグエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ:	文字
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4Bw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4Bw.出力形式では、ビッグエンディアン、32 ビット、UCS4 (universal character set code in four octets)形式の UNICODE (ユニコード)文字列を書き込みます。現在の SAS セッションのエンコーディングを使用する文字列を処理します。

比較

\$UCS4Bw.出力形式では、\$UCS4BEw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
x = '大'; put x \$ucs4b4.;	'00005927'x (binary)

関連項目

出力形式:

- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UCS4BEw. 出力形式" \(516 ページ\)](#)
- ["\\$UCS4Lw. 出力形式" \(518 ページ\)](#)
- ["\\$UCS4Xw. 出力形式" \(521 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UCS4Bw. 入力形式" \(912 ページ\)](#)
- ["\\$UCS4Lw. 入力形式" \(913 ページ\)](#)
- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS4BEw. 出力形式

ビッグエンディアン、32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項:

この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4BEw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4BEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。ビッグエンディアン、32 ビット、UCS4 (universal character set code in four octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS4BEw.出力形式では、\$UCS4Bw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
x='000059270000004100000042'x; put x \$ucs4be.;	大 AB

関連項目

出力形式:

- ["\\$UCS4Bw. 出力形式" \(515 ページ\)](#)

入力形式:

- ["\\$UCS4Bw. 入力形式" \(912 ページ\)](#)

\$UCS4Lw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、リトルエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4Lw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4Lw.出力形式では、リトルエンディアン、32 ビット、UCS4 (universal character set code in four octets)形式の UNICODE (ユニコード)文字列を書き込みます。現在の SAS セッションのエンコーディングを使用する文字列を処理します。

比較

\$UCS4Lw.出力形式では、\$UCS4LEw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
data _null_;	2759
x = '太';	
y=put(x,\$ucs4l4.);	
put y \$hex.;	
run;	

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UCS4Bw. 出力形式" \(515 ページ\)](#)
- ["\\$UCS4LEw. 出力形式" \(520 ページ\)](#)
- ["\\$UCS4Xw. 出力形式" \(521 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UCS4Bw. 入力形式" \(912 ページ\)](#)
- ["\\$UCS4Lw. 入力形式" \(913 ページ\)](#)
- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS4LEw. 出力形式

リトルエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4LEw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4LEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。リトルエンディアン、32 ビット、UCS4 (universal character set code in four octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS4LEw.出力形式では、\$UCS4Lw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
x='275900004100000042000000'x; put x \$ucs4le.;	大 AB

関連項目

出力形式:

- ["\\$UCS4Lw. 出力形式" \(518 ページ\)](#)

入力形式:

- ["\\$UCS4Lw. 入力形式" \(913 ページ\)](#)

\$UCS4Xw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、ネイティブエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を書き込みます。

カテゴリ:	文字
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4Xw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4Xw.出力形式では、動作環境のバイトオーダーで、32 ビットの UCS4 (universal character set code in two octets)形式の UNICODE (ユニコード)文字列を書き込みます。

比較

\$UCS4Xw.出力形式では、\$UCS4XEw.出力形式とは逆の処理を行います。同一の動作環境内のデータを変換する場合は、\$UCS4Xw.出力形式を使用します。異なる動作環境でデータを変換する場合は、\$UCS4Bw.出力形式または\$UCS4Lw.出力形式を使用します。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x = '大'; put x \$ucs4x4;</pre>	'00005927'x (binary) or '27590000'x (little endian)

関連項目

出力形式:

- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS4XEw. 出力形式" \(523 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UCS4Bw. 出力形式" \(515 ページ\)](#)
- ["\\$UCS4Lw. 出力形式" \(518 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)

- “\$UCS4Bw. 入力形式” (912 ページ)
- “\$UCS4Bw. 出力形式” (515 ページ)
- “\$UCS4Lw. 入力形式” (913 ページ)
- “\$UCS4Xw. 入力形式” (915 ページ)
- “\$UTF8Xw. 入力形式” (927 ページ)

\$UCS4XEw. 出力形式

ネイティブエンディアン、32 ビット、UCS4 形式の UNICODE (ユニコード)文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UCS4XEw.

構文の説明

w

出力フィールドの幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

詳細

\$UCS4XEw.出力形式では、現在の SAS セッションのエンコーディングで文字列を書き込みます。ネイティブエンディアン、32 ビット、UCS4 (universal character set code in four octets)形式の UNICODE (ユニコード)文字列を処理します。

比較

\$UCS4XEw.出力形式では、\$UCS4Xw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x='275900004100000042000000'x; put x \$ucs4be4.;</pre>	大 AB (little endian)

関連項目

出力形式:

- ["\\$UCS4Xw. 出力形式" \(521 ページ\)](#)

入力形式:

- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)

\$UESCW. 出力形式

現在の SAS セッションでエンコードされる文字列を処理し、その文字列を Unicode エスケープ (UESC) 表記で書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UESCW.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1-32767

詳細

0 から 9、a から z、A から Z など、すべての動作環境で使用可能でない文字は、UESC で表す必要があります。\$UESCw. はネストできます。

比較

\$UESCw.出力形式では、\$UESCEw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+----2
x='大';	¥u5927
y='u5927'	¥uu5927
z='uu5927';	¥uuu5927
put x = \$uesc10.;	
put y = \$uesc10.;	
put z = \$uesc10.;	

関連項目

出力形式:

- ["\\$UESCEw. 出力形式" \(526 ページ\)](#)

入力形式:

- ["\\$UESCw. 入力形式" \(917 ページ\)](#)
- ["\\$UESCEw. 入力形式" \(919 ページ\)](#)

\$UESCEw. 出力形式

Unicode エスケープ(UESC)表記の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UESCEw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32767

詳細

データが現在の SAS セッションのエンコーディングによってサポートされていない場合、データは UESC のまま残ります。

比較

\$UESCEw.出力形式では、\$UESCw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1----+----2
x=put('¥u5927',\$uesce10.);	x= 太
x=put('¥uu5927',\$uesce10.);	x=¥u5927
x=put('¥uuu5927',\$uesce10.);	x=¥uu5927

関連項目

出力形式:

- ["\\$UESCw. 出力形式" \(524 ページ\)](#)

入力形式:

- ["\\$UESCw. 入力形式" \(917 ページ\)](#)
- ["\\$UESCEw. 入力形式" \(919 ページ\)](#)

\$UNCRw. 出力形式

現在の SAS セッションでエンコードされる文字列を処理し、その文字列を数値文字表現(NCR)で書き込みます。

カテゴリ:	文字
配置:	左
制限事項:	この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UNCRw.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32767

比較

\$UNCRw.出力形式では、\$UNCREw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1----+----2
<pre>x='91E5'x; /* Japanese '木' in Shift-JIS */ y='abc'; put x \$uncr10.; put y \$uncr10.;</pre>	<pre>&#22823 abc</pre>

関連項目

出力形式:

- [“\\$UNCREw. 出力形式” \(528 ページ\)](#)

入力形式:

- [“\\$UNCRw. 入力形式” \(920 ページ\)](#)
- [“\\$UNCREw. 入力形式” \(921 ページ\)](#)

\$UNCREw. 出力形式

数値文字表現(NCR)の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UNCREW.

構文の説明

W

出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32767

詳細

各国語の文字は NCR で表記する必要があります。

比較

\$UNCREW.出力形式では、\$UNCRW.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1
x='大abc'; put x \$uncre10.;	大 abc

関連項目

出力形式:

- ["\\$UNCRW. 出力形式" \(527 ページ\)](#)

入力形式:

- ["\\$UNCRW. 入力形式" \(920 ページ\)](#)

■ “\$UNCREw. 入力形式” (921 ページ)

\$UPARENw. 出力形式

現在の SAS セッションでエンコードされる文字列を処理し、その文字列を Unicode かっこ (UPAREN) 表記で書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UPARENw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 8

範囲 7-32767

詳細

文字列は、かっこおよび Unicode の 16 進表現によってエンコードされます。

比較

\$UPARENw.出力形式では、\$UPARENw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語 (Shift_JIS) セッションエンコーディングを使用します。

ステートメント	結果
	-----1-----2-----3-----+
x='大';	<u5927>
y='abc3';	<u0061> <u0062> <u0063> <u0033>
put x \$uparen7.;	
put y \$uparen28.;	

関連項目

出力形式:

- [“\\$UPARENw. 出力形式” \(531 ページ\)](#)

入力形式:

- [“\\$UPARENw. 入力形式” \(923 ページ\)](#)
- [“\\$UPARENw. 入力形式” \(924 ページ\)](#)

\$UPARENw. 出力形式

Unicode カッコ (UPAREN) 内の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UPARENw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32767

比較

\$UPARENEw.出力形式では、\$UPARENw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+
<pre>x='<u0061><u0062><u0063><u0033>'; put x \$uparene4.;</pre>	abc3

関連項目

出力形式:

- ["\\$UPARENw. 出力形式" \(530 ページ\)](#)

入力形式:

- ["\\$UPARENw. 入力形式" \(923 ページ\)](#)
- ["\\$UPARENEw. 入力形式" \(924 ページ\)](#)

\$UTF8Xw. 出力形式

現在の SAS セッションのエンコーディングを使用する文字列を処理し、UTF-8 (Universal Transformation Format)エンコーディングの文字列を書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UTF8Xw.

構文の説明

w

出力フィールドの幅を指定します。変数が入る十分な文字幅を指定します。文字の幅は、個々の文字のコードポイント値によって異なります。

デフォルト 8

範囲 1-32767

比較

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x='91E5'x; /* Japanese '太' in Shift-JIS */ put x \$utf8x10.;</pre>	x='e5a4a7'x

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)

\$UTF8XEw. 出力形式

UTF-8 (Universal Transformation Format)の文字列を処理し、その文字列を現在の SAS セッションのエンコーディングで書き込みます。

カテゴリ: 文字

配置: 左

制限事項: この出力形式は CAS で実行する DATA ステップではサポートされていません。

構文

\$UTF8XEw.

構文の説明

w

出力フィールドの幅を指定します。変数が入る十分な文字幅を指定します。文字の幅は、個々の文字のコードポイント値によって異なります。

デフォルト 8

範囲 1-32767

比較

\$UTF8XEw.出力形式では、\$UTF8Xw.出力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
	----+----1
<pre>x = unicode('u5927'); put x \$utf8xe10;</pre>	大

関連項目

出力形式:

- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$VSLOGw. 出力形式

視覚的順序で並ぶ文字列を処理し、その文字列を左から右の論理的順序で書き込みます。

カテゴリ: BIDI テキストの操作
CAS
配置: 左

構文

\$VSLOGw.

構文の説明

w
出力フィールドの幅を指定します。
デフォルト 200
範囲 1-32767

詳細

\$VSLOGw.出力形式は、視覚的順序で保存されるデータを転送するときに使用します。

.....
注: \$VSLOGw.出力形式では、データ文字列の一部の組み合わせが正しく処理されません。
.....

比較

\$VSLOGw.出力形式では、\$VSLOGrw.出力形式とは逆の処理を行います。

例

次の例では、ヘブライ語の入力値"טוֹר פֿליגט"を使用します。

ステートメント	結果
	----+----1----+----2----+
put text \$vslog12.;	ﻟﯩﺘﻪ flight

次の例では、computer のアラビア語の入力“**ﻛﻮﻣﭙﻮﺗﻪﺭ**”を使用します。

ステートメント	結果
	----+----1----+----2----+
put text \$vslog12.;	ﻛﻮﻣﭙﻮﺗﻪﺭ ﻛﻮﻧﭙﻮﺗﻪﺭ

関連項目

出力形式:

- [“\\$VSLOGRw. 出力形式” \(536 ページ\)](#)

入力形式:

- [“\\$VSLOGw. 入力形式” \(928 ページ\)](#)
- [“\\$VSLOGRw. 入力形式” \(930 ページ\)](#)

\$VSLOGRw. 出力形式

視覚的順序で並ぶ文字列を処理し、その文字列を右から左の論理的順序で書き込みます。

カテゴリ: BIDI テキストの操作
CAS
配置: 左

構文

\$VSLOGRw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 200

範囲 1-32767

詳細

\$VSLOGRw.出力形式は、視覚的順序で保存されるデータを転送するときに使用します。

注: \$VSLOGRw.出力形式では、データ文字列の一部の組み合わせが正しく処理されません。

比較

\$VSLOGRw.出力形式では、\$VSLOGw.出力形式とは逆の処理を行います。

例

次の例では、ヘブライ語の入力値"טוֹר flight"を使用します。

ステートメント	結果
	----+----1----+
put text \$logvs12;	flight טוֹר

次の例では、computer のアラビア語の入力"ت اذ"を使用します。

ステートメント	結果
	----+----1----+
put text \$logvs12;	ت اذ コンピューター

関連項目

入力形式:

- [“\\$VSLOGw. 入力形式” \(928 ページ\)](#)
- [“\\$VSLOGRw. 入力形式” \(930 ページ\)](#)

WEEKUw. 出力形式

U アルゴリズムを使用して、10 進数の週番号を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

WEEKUw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 11

範囲 2-200

詳細

WEEKUw.出力形式では、週番号形式で書き込みます。WEEKUw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。U アルゴリズムでは、年内の週番号を使用して SAS 日付値を計算します(日曜日を週の最初の日と見なします)。週番号値は、先頭に 0 を付けた 0 から 53 の範囲の 10 進数として表され、最大値は 53 になります。たとえば、年の 5 週目は 05 として表されます。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	Www	w01
5-6	yyWww	03W01
7-8	yyWwwdd	03W0101
9-10	yyyyWwwdd	2003W0101
11-200	yyyy-Www-dd	2003-W01-01

比較

WEEKVw.出力形式では、01 から 53 の範囲の 10 進数として週番号を書き込みます。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。WEEKWw.出力形式では、00 から 53 の範囲の 10 進数として年の週番号を書き込みます。第 1 週の最初の日は月曜日になります。WEEKUw.出力形式では、先頭に 0 を付けた 0 から 53 の範囲の 10 進数として年の週番号を書き込みます(週の最初の日は日曜日になります)。

例

```
sasdate = '01JAN2003'd;
```

ステートメント	結果
	----+----1----+
v=put(sasdate,weeku3.);	W00
w=put(sasdate,weeku5.);	03W00
x=put(sasdate,weeku7.);	03W0004
y=put(sasdate,weeku9.);	2003W0004
z=put(sasdate,weeku11.);	2003-W00-04
put v;	
put w;	
put x;	
put y;	
put z;	

関連項目

出力形式:

- “WEEKVw. 出力形式” (540 ページ)
- “WEEKWw. 出力形式” (542 ページ)

WEEKVw. 出力形式

V アルゴリズムを使用して、10 進数の週番号を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

WEEKVw.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 11

範囲 2-200

詳細

WEEKVw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。V アルゴリズムでは、SAS 日付値を計算します。週番号値は、先頭に 0 を付けた 01 から 53 の範囲の 10 進数として表され、最大値が 53 になります。各週は月曜日から始まり、年の第 1 週は 1 月 4 日と年の最初の木曜日の両方を含む週です。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。たとえば、年の 5 週目は 06 として表されます。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	Www	w01
5-6	yyWww	03W01
7-8	yyWwwdd	03W0101
9-10	yyyyWwwdd	2003W0101
11-200	yyyy-Www-dd	2003-W01-01

比較

WEEKVw.出力形式では、01 から 53 の範囲の 10 進数として週番号を書き込みます。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。WEEKWw.出力形式では、00 から 53 の範囲の 10 進数として年の週番号を書き込みます。第 1 週の最初の日は月曜日になります。WEEKUw.出力形式では、先頭に 0 を付けた 0 から 53 の範囲の 10 進数として年の週番号を書き込みます(週の最初の日は日曜日になります)。

例

```
sasdate='01JAN2003'd;
```

ステートメント	結果
	----+----1----+
v=put(sasdate,weekv3.);	W01
w=put(sasdate,weekv5.);	03W01
x=put(sasdate,weekv7.);	03W0103
y=put(sasdate,weekv9.);	2003W0103
z=put(sasdate,weekv11.);	2003-W01-03
put v;	
put w;	
put x;	
put y;	
put z;	

関連項目

出力形式:

- “WEEKUw. 出力形式” (538 ページ)
- “WEEKWw. 出力形式” (542 ページ)

WEEKWw. 出力形式

W アルゴリズムを使用して、10 進数の週番号を書き込みます。

カテゴリ: CAS
日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

WEEKWw.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 11

範囲 2-200

詳細

WEEKWw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。W アルゴリズムでは、年内の週番号を使用して SAS 日付値を計算します(月曜日を週の最初の日と見なします)。週番号値は、先頭に 0 を付けた 0 から 53 の範囲の 10 進数として表され、最大値は 53 になります。たとえば、年の 5 週目は 05 として表されます。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	Www	w01

幅	出力形式	例
5-6	yyWww	03W01
7-8	yyWwwdd	03W0101
9-10	yyyyWwwdd	2003W0101
11-200	yyyy-Www-dd	2003-W01-01

比較

WEEKVw.出力形式では、01 から 53 の範囲の 10 進数として週番号を書き込みます。各週は月曜日から始まり、年の第 1 週には 1 月 4 日と年の最初の木曜日の両方が含まれます。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。WEEKWw.出力形式では、00 から 53 の範囲の 10 進数として年の週番号を書き込みます。第 1 週の最初の日は月曜日になります。WEEKUw.出力形式では、先頭に 0 を付けた 0 から 53 の範囲の 10 進数として年の週番号を書き込みます(週の最初の日は日曜日になります)。

例

```
sasdate = '01JAN2003'd;
```

ステートメント	結果
	----+----1----+
v=put(sasdate,weekw3.);	W03
w=put(sasdate,weekw5.);	03W03
x=put(sasdate,weekw7.);	03W0003
y=put(sasdate,weekw9.);	2003W0003
z=put(sasdate,weekw11.);	2003-W00-03
put v;	
put w;	
put x;	
put y;	
put z;	

関連項目

出力形式:

- [“WEEKUw. 出力形式” \(538 ページ\)](#)

■ “WEEKVw. 出力形式” (540 ページ)

YENw.d 出力形式

円記号、桁区切り、小数点付きの数値を出力します。

カテゴリ: CAS
数値

配置: 右

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

YENw.d

構文の説明

w
出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32

d
数値の小数点以下の桁数を指定します。

範囲 0-31

詳細

YENw.d 出力形式は、先頭に円記号を付け、各値を 3 桁ごとにカンマで区切って数値を書き込みます。

円記号文字のコードの 16 進表現は、EBCDIC のシステムでは 5B、ASCII のシステムでは 5C です。これらのコードが表す通貨記号は、他の国では異なる場合があります。

例

```
put cost yen10.2;
data _null_;
  value=1254.71;
  put value yen10.2;
run;
```

Cost	結果
	----+----1
1254.71	¥1,254.71

関連項目

入力形式:

- [“YENw.d 入力形式” \(931 ページ\)](#)

YYWEEKUw. 出力形式

曜日情報を除外し、U アルゴリズムを使用して、10 進数の週番号を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

YYWEEKU*w*.

構文の説明

w
出力フィールドの幅を指定します。

デフォルト	7
範囲	2-8

詳細

YYWEEKUw.出力形式では、週番号形式で書き込みます。YYWEEKUw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。U アルゴリズムでは、年内の週番号を使用して SAS 日付値を計算します(日曜日を週の最初の日と見なします)。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	Www	W01
5-6	yyWww	07W01
7	yyyyWww	2007W01
8	yyyy-Www	2007-W01
9 以上	無効	無効

比較

YYWEEKUw.出力形式は WEEKUw.出力形式と似ていますが、YYWEEKUw.出力形式では曜日情報を示しません。また、YYWEEKUw.出力形式では、8 より大きい幅は受け付けません。

例

sasdate = '01JAN2007'd;

ステートメント	結果
	----+----1----+
u=put(sasdate,yyweeku3.);	W00
v=put(sasdate,yyweeku4.);	W00
w=put(sasdate,yyweeku5.);	07W00
x=put(sasdate,yyweeku6.);	07W00

ステートメント	結果
y=put(sasdate,yyweeku7.); z=put(sasdate,yyweeku8.); put u; put v; put w; put x; put y; put z;	2007W00 2007-W00

関連項目

出力形式:

- [“WEEKUw. 出力形式” \(538 ページ\)](#)

YYWEEKVw. 出力形式

曜日情報を除外し、V アルゴリズムを使用して、10 進数の週番号を書き込みます。

- カテゴリ:
- CAS
日付と時間
- 配置:
- 左
- 注:
- 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

YYWEEKVw.

構文の説明

- w
- 出力フィールドの幅を指定します。
- デフォルト 7
- 範囲 2-8

詳細

YYWEEKVw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。V アルゴリズムでは、SAS 日付値を計算します。週番号値は、先頭に 0 を付けた 01 から 53 の範囲の 10 進数として表され、最大値が 53 になります。各週は月曜日から始まり、年の第 1 週は 1 月 4 日と年の最初の木曜日の両方を含む週です。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。たとえば、年の 5 週目は 06 として表されます。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	www	w01
5-6	yyWww	07W01
7	yyyyWww	2007W01
8	yyyy-Www	2007-W01
9-above	無効	無効

比較

YYWEEKVw.出力形式は WEEKVw.出力形式と似ていますが、YYWEEKVw.出力形式では曜日情報を示しません。また、YYWEEKVw.出力形式では、8 より大きい幅は受け付けません。

例

sasdate = '01JAN2007'd;

ステートメント	結果
	----+----1----+
u=put(sasdate,yyweekv3.);	W01
v=put(sasdate,yyweekv4.);	W01
w=put(sasdate,yyweekv5.);	07W01
x=put(sasdate,yyweekv6.);	07W01
y=put(sasdate,yyweekv7.);	2007W01
z=put(sasdate,yyweekv8.);	2007-W01
put u;	
put v;	
put w;	
put x;	

ステートメント

結果

```
put y;
put z;
```

関連項目

出力形式:

- [“WEEKVw. 出力形式” \(540 ページ\)](#)

YYWEEKWw. 出力形式

曜日情報を除外し、W アルゴリズムを使用して、10 進数の週番号を書き込みます。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この形式は DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

YYWEEKWw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 7

範囲 2-8

詳細

YYWEEKWw.出力形式では、指定された幅に応じてさまざまな形式で書き込みます。W アルゴリズムでは、年内の週番号を使用して SAS 日付値を計算します。

幅、出力形式および例の詳細については、次の表を参照してください。

幅	出力形式	例
3-4	Www	W01
5-6	yyWww	07W01
7	yyyyWww	2007W01
8	yyyy-Www	2007-W01
9-above	無効	無効

比較

YYWEEKWw.出力形式は WEEKWw.出力形式と似ていますが、YYWEEKWw.出力形式では曜日情報を示しません。また、YYWEEKWw.出力形式では、8 より大きい幅は受け付けません。

例

sasdate = '01JAN2007'd

ステートメント	結果
	----+----1----+
u=put(sasdate,yyweekw3.);	W01
v=put(sasdate,yyweekw4.);	W01
w=put(sasdate,yyweekw5.);	07W01
x=put(sasdate,yyweekw6.);	07W01
y=put(sasdate,yyweekw7.);	2007W01
z=put(sasdate,yyweekw8.);	2007-W01
put u;	
put v;	
put w;	
put x;	
put y;	
put z;	

関連項目

出力形式:

- [“WEEKWw. 出力形式” \(542 ページ\)](#)

5 部

各国語サポート関連の関数

11 章	SAS 文字列関数の国際化の互換性	555
12 章	関数での VARCHAR サポート	579
13 章	各国語サポート関連の関数のディクショナリ	583

SAS 文字列関数の国際化の互換性

SAS 文字列関数の国際化の互換性 555

SAS 文字列関数の国際化の互換性

SAS では、文字データを簡単に操作できる文字列関数と CALL ルーチンを提供します。従来の SAS 文字列関数の多くは、1 文字のサイズが常に 1 バイトであることを前提としています。この処理は、1 バイト文字セット(SBCS)のデータでは正常に動作します。ただし、これらの関数と CALL ルーチンの一部を 2 バイト文字セット(DBCS)または UTF-8 のようなマルチバイト文字セット(MBCS)のデータに使用すると、多くの場合データは適切に処理されず、文字列関数から正しい結果が得られません。

この問題を解決するために、DBCS と MBCS データを慎重に処理する文字列操作が可能な、*K 関数*と呼ばれる文字列関数と CALL ルーチンが導入されました。K 関数には文字列内の文字のサイズに関して何の前提もありません。[表 11.4 \(557 ページ\)](#)では、各 SAS 文字列関数の I18N 互換性のレベルを説明します。I18N とは、internationalization (国際化)の略称です。互換性とは、特定の文字列関数を使用するプログラムが、プログラムを変更することなく異なる言語とロケールに適合できるかどうかを意味します。

ユーザーが K 関数を適切に使用するには、バイトベースのオフセット長と文字ベースのオフセット長の違いを理解することが重要です。

バイトベースのオフセットでは、文字に対して指定された開始位置が、文字列中の当該文字のバイト位置であると想定します。SBCS データの場合、1 文字は常に 1 バイト長なので、文字列中の 2 番目の文字は 2 バイト目から始まると考えてかまいません。しかし、文字列中のデータがマルチバイトデータであれば、2 バイト目のデータは、データやそのエンコーディングに応じて、次のいずれかになります。

- 文字列中、2 番目の文字
- 2 バイト文字の第 2 バイト
- 文字列中、1 番目の文字の第 1 バイト

バイトベースの長さは、文字列を構成するバイトの数を表します。

文字ベースのオフセットでは、指定された位置が、文字列中の当該文字の位置であると想定します。どのエンコーディングでも、文字ベースの位置 2 にあるのは、文字列中の 2 番目の文字です。文字列中の文字のサイズが分かっていると想定はできません。

文字ベースの長さは、文字列を構成する文字の数を表します。

K 関数は文字ベースのオフセットや長さを使います。文字列中の各文字のバイト位置は考慮されません。

SAS 文字列関数および対応する K 関数の機能を表す例を以下に示します。

UTF8 エンコーディングの場合、STR='E282AC313233'x という文字列は 6 バイトからなり、4 つの文字を表します。

表 11.1 UTF-8 での文字列表現

16 進表現	E282AC	31	32	33
	€	1	2	3

次の例では、この同じ 16 進文字列を SUBSTR 関数に渡していますが、戻り値は UTF-8 の予期しない値です。予期される戻り値は、1 文字目に当たるユーロ記号です。しかし、結果は無効な文字になっています。LENGTH 関数はバイト指向であるため、データの個々のバイトが読み取られます。SUBSTR 関数からは先頭文字の先頭バイトが返されますが、これは不完全な値です。なぜならば先頭文字は 3 バイトに基づいてエンコードされているからです。

表 11.2 正規関数を使用したコード例

ステートメント	結果
<pre>/* SAS program to submit in a UTF-8 SAS session data _null_; str= "€123"; s=substr(str,1,1); sl=length(s); l=length(str); put str= \$hex16. /s= sl= / s= \$hex. /l=; run;</pre>	<pre>str=E282AC313233 s= sl=1 s=E22020202020 l=6</pre>

表 11.3 (557 ページ)のコードはこれを解決するため、1 バイトデータしか正常に扱えない SUBSTR()関数の代わりに、1 バイトデータでもマルチバイトデータでも動作する、KSUBSTR()関数を使っています。他の文字列関数についても直接的な置き換えが可能で、LENGTH()または INDEX()関数を変更して、KLENGTH()または KINDEX()関数を使用します。2 つの関数はデータの読み込み方が異なるため、マルチバイト環境で実行した場合、異なる結果が得られます。

表 11.3 K 関数を使用したコード例

ステートメント	結果
<pre>/* SAS program to submit in a UTF-8 SAS session */ data _null_; str= "€123"; s=ksubstr(str,1,1); sl=length(s); l=length(str); put str= \$hex16. /s= sl= / s= \$hex. /l=; run;</pre>	<pre>str=E282AC313233 s=€ l=4</pre>

SAS 文字列処理関数をすべて K 関数に置換する際には、あらかじめ SAS プログラムを検査しておいてください。文字列関数が、単一バイト文字のみを含んでいるデータを処理する場合、K 関数は必要ありません。たとえば、XML タグを含んでいる文字列に対しては、K 関数を使用する必要はありません。SAS プログラムに含まれているデータの内容と、それがどのように処理されるかを把握することにより、SAS コードの不要な更新を回避できます。バイナリデータ処理は文字列処理用の K 関数ではサポートされていません。K 関数は、処理対象となる文字列が、現在のセッションのエンコーディングに一致するものと予想します。

文字列関数は、関数が DBCS、MBCS、または SBCS のデータを処理できるかどうかに応じて I18N レベルが割り当てられます。レベルの説明を次に示します。

I18N レベル 0

この関数は SBCS データ向けに設計されています。この関数を MBCS データの処理に使用しないでください。

I18N レベル 1

MBCS データを処理する場合、可能な限りこの関数の使用は避けてください。I18N レベル 1 の関数は、特定の環境下では DBCS または MBCS エンコーディングを使用すると正常に動作しない場合があります。

I18N レベル 2

この関数は SBCS、DBCS、および MBCS(UTF-8)データに使用できます。

表 11.4 SAS 文字列関数

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
“ANORM420 関数” (p. 591)	EBCDIC420 でエンコードされた入力文字列を基に正規化された文字列を返します。		X	
“ANYALNUM 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から英数字を検索し、最初に検索された文字の位置を返します。		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
“ANYALPHA 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から英字を検索し、最初に検索された文字の位置を返します。		X	
“ANYCNTRL 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から制御文字を検索し、最初に検索された文字の位置を返します。			X
“ANYDIGIT 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から数字を検索し、最初に検索された数字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。		X	
“ANYFIRST 関数” (SAS 関数と CALL ルーチン: リファレンス)	VALIDVARNAME=V7 の SAS 変数名の開始文字として有効な文字を文字列から検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。こ		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。			
“ANYGRAPH 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列からグラフィカル文字を検索し、最初に検索された文字の位置を返します。		X	
“ANYLOWER 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から小文字を検索し、最初に検索された文字の位置を返します。		X	
“ANYNAME 関数” (SAS 関数と CALL ルーチン: リファレンス)	VALIDVARNAME=V7 において、SAS 変数名として有効な文字を文字列から検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数が使用されていない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。		X	
“ANYPRINT 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から印刷可能な文字を検索し、最初に検索された文字の位置を返します。		X	
“ANYPUNCT 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から句読文字を検索し、最初に		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	検索された文字の位置を返します。			
“ANYSPACE 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から空白文字(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)を検索します。最初に検索された文字の位置を返します。		X	
“ANYUPPER 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から大文字を検索し、最初に検索された文字の位置を返します。		X	
“ANYXDIGIT 関数” (SAS 関数と CALL ルーチン: リファレンス)	数字を表す 16 進法の文字を文字列から検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。		X	
“BASECHAR 関数” (p. 601)	文字を基本文字に変換します。			X
“BYTE 関数” (SAS 関数と CALL ルーチン: リファレンス)	ASCII 照合順序または EBCDIC 照合順序の 1 文字を返します。	X		

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"CAT 関数" (SAS 関数と CALL ルーチン: リファレンス)	先頭または末尾の空白を削除せずに、連結文字列を返します。			X
"CATQ 関数" (SAS 関数と CALL ルーチン: リファレンス)	項目を区切る区切り文字を使用したり、区切り文字を含んでいる文字列に引用符を追加したりして、文字値または数値を連結します。			X
"CATS 関数" (SAS 関数と CALL ルーチン: リファレンス)	先頭と末尾の空白を削除して、連結文字列を返します。			X
"CATT 関数" (SAS 関数と CALL ルーチン: リファレンス)	末尾の空白を削除して、連結文字列を返します。			X
"CATX 関数" (SAS 関数と CALL ルーチン: リファレンス)	先頭と末尾の空白を削除し、区切り文字を挿入して、文字列を返します。			X
"CHAR 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列の指定した位置の 1 文字を返します。			X
"CHOOSEC 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数のリストからの選択結果を表す文字値を返します。			X
"CHOOSEN 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数のリストからの選択結果を表す数値を返します。			X
"COALESCEC 関数" (SAS 関数と CALL ルーチン: リファレンス)	数値の引数のリストからの最初の非欠損値を返します。			X
"COLLATE 関数" (SAS 関数と CALL ルーチン: リファレンス)	ASCII 照合順序または EBCDIC 照合順序の文字列を返します。	X		

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"COMPARE 関数" (SAS 関数と CALL ルーチン: リファレンス)	2 つの文字列を比較し、異なる文字が検出された場合には最も左にある文字の位置を返し、異なる文字が検出されない場合には 0 を返します。	X		
"COMPBL 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列内のワード間にある複数の空白を取り除きます。			X
"COMPGED 関数" (SAS 関数と CALL ルーチン: リファレンス)	2 つの文字列間の一般化編集距離を返します。	X		
"COMPLEV 関数" (SAS 関数と CALL ルーチン: リファレンス)	2 つの文字列間のレーベンシュタインの編集距離を返します。	X		
"COMPRESS 関数" (SAS 関数と CALL ルーチン: リファレンス)	元の文字から指定した文字を削除した文字列を返します。	X		
"COUNT 関数" (SAS 関数と CALL ルーチン: リファレンス)	指定した部分文字列が文字列内に含まれる回数を数えます。	X		
"COUNTC 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字のリストに表示される(または表示されない)文字列内の文字の個数を数えます。	X		
"COUNTW 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列中の単語数を数えます。	X		
"DEQUOTE 関数" (SAS 関数と CALL ルーチン: リファレンス)	引用符で始まる文字列から、対となる引用符を削除し、閉じる側の引用符より右側にあるすべての文字を削除します。		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"FIND 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列内の特定の部分文字列を検索します。	X		
"FINDC 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字のリストにある文字を文字列から検索します。	X		
"FINDW 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列の単語の文字位置または文字列の単語の数を返します。		X	
"FIRST 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列内の先頭文字を返します。			X
"GETLOCENV 関数" (p. 606)	現在のロケール/言語環境を返します。			X
"GETPXLANGUAGE 関数" (p. 607)	現在の 2 文字の言語コードを返します。			X
"GETPXLOCALE 関数" (p. 608)	SAS ロケールの POSIX ロケール値を返します。			X
"GETPXREGION 関数" (p. 610)	現在の 2 文字の地域コードを返します。			X
"HTMLDECODE 関数" (SAS 関数と CALL ルーチン: リファレンス)	HTML 数値文字参照または HTML 文字実体参照を含む文字列をデコードし、デコードされた文字列を返します。			X
"HTMLENCODE 関数" (SAS 関数と CALL ルーチン: リファレンス)	HTML 文字実体参照を使用して文字列をエンコードし、エンコードされた文字列を返します。			X
"TFC 関数" (SAS 関数と CALL ルーチン: リファレンス)	式の true、false、欠損に基づいて文字値を返します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"IFN 関数" (SAS 関数と CALL ルーチン: リファレンス)	式の真、偽、欠損に基づいて数値を返します。			X
"INDEX 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字式から文字列を検索し、最初に検索された文字列の最初の文字の位置を返します。	X		
"INDEXC 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字式から指定した文字すべてを検索し、その文字の位置を返します。	X		
"INDEXW 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字式から単語として指定した文字列を検索し、単語の最初の文字の位置を返します。	X		
"KCOMPARE 関数" (p. 614)	文字式の比較結果を返します。			X
"KCOMPRESS 関数" (p. 616)	指定された文字を文字式から削除します。			X
"KCOUNT 関数" (p. 618)	式の 2 バイト文字の数を返します。			X
"KCOUNTC 関数" (p. 619)	文字列に含まれる個々の文字を数えます。			X
"KCOUNTW 関数" (p. 622)	文字列中の単語数を数えます。			X
"KCOUNTX 関数" (p. 626)	指定した部分文字列が文字列内に含まれる回数を数えます。			X
"KCVT 関数" (p. 628)	データのある種類のエンコーディングデータから別のエンコーディング			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	データに変換します。			
“K FIND 関数” (p. 630)	文字列内の特定の部分文字列を検索します。			X
“K FINDC 関数” (p. 632)	文字のリストにある文字を文字列から検索します。			X
“K FINDW 関数” (p. 636)	文字列内の単語の文字位置か、文字列内の単語の番号を返します。			X
“K INDEX 関数” (p. 641)	文字式から文字列を検索します。			X
“K INDEXB 関数” (p. 643)	文字式から文字列を検索します。			X
“K INDEXCB 関数” (p. 645)	文字式から指定した文字を検索します。			X
“K INDEXC 関数” (p. 647)	文字式から指定した文字を検索します。			X
“K LEFT 関数” (p. 649)	不要な先頭の DBCS の空白と SO-SI を削除し、文字式を左詰めにします。			X
“K LENGTH 関数” (p. 652)	引数の長さを返します。			X
“K LOWCASE 関数” (p. 653)	引数のすべての文字を小文字に変換します。			X
“K PROPCASE 関数” (p. 658)	中国語、日本語、韓国語、台湾語(CJKT) 文字を変換します。			X
“K PROPCHAR 関数” (p. 662)	特殊文字を標準文字に変換します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"KPROPDATA 関数" (p. 663)	印刷不可文字を削除または変換します。			X
"KREVERSE 関数" (p. 668)	文字式を逆にします。			X
"KRIGHT 関数" (p. 669)	末尾の DBCS の空白と SO-SI を削除し、文字式を右詰めにします。			X
"KSCAN 関数" (p. 670)	指定された単語を文字式から選択します。			X
"KSCANX 関数" (p. 673)	指定された単語を文字式から選択します。			X
"KSTRCAT 関数" (p. 678)	複数の文字式を連結します。			X
"KSTRIP 関数" (p. 680)	文字列から先頭と末尾の空白を削除します。			X
"KSUBSTR 関数" (p. 682)	引数から部分文字列を抽出します。			X
"KSUBSTRB 関数" (p. 684)	引数内の部分文字列のバイト位置に従って、引数から部分文字列を抽出します。			X
"KSUBSTRN 関数" (p. 686)	部分文字列を返します。長さがゼロの結果も返せます。			X
"KTRANSLATE 関数"	文字式内の指定された文字を置換します。			X
"KTRIM 関数" (p. 689)	末尾の DBCS の空白と SO-SI を文字式から削除します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"KTRUNCATE 関数" (p. 691)	マルチバイト文字を壊すことなく、文字列をバイト単位で、指定した長さに切ります。			X
"KUPCASE 関数" (p. 692)	引数内のすべての文字を大文字に変換します。			X
"KUPDATE 関数" (p. 694)	文字値の内容を挿入、削除および置換します。			X
"KUPDATEB 関数" (p. 697)	引数内の文字値のバイト位置に従って、文字値の内容を挿入、削除および置換します。			X
"KUPDATES 関数" (p. 699)	文字値の内容を挿入、削除および置換します。			X
"KVERIFYB 関数" (p. 703)	式に固有の最初の文字の位置を返します。			X
"KVERIFY 関数" (p. 702)	式に固有の最初の文字の位置を返します。			X
"LEFT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列を左詰めにします。			X
"LENGTH 関数" (SAS 関数と CALL ルーチン: リファレンス)	末尾の空白を除いた文字列の長さを返します。文字列が空白の場合には、1 を返します。			X
"LENGTHC 関数" (SAS 関数と CALL ルーチン: リファレンス)	末尾の空白を含めた文字列の長さを返します。			X
"LENGTHM 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列に割り当てられたメモリの量			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	を返します(バイト単位)。			
"LENGTHN 関数" (SAS 関数と CALL ルーチン: リファレンス)	末尾の空白を除いた文字列の長さを返します。			X
"LOWCASE 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数のすべての文字を小文字に変換します。			X
"MD5 関数" (SAS 関数と CALL ルーチン: リファレンス)	指定した文字列のメッセージダイジェストの結果を返します。			X
"MISSING 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数が欠損値を含むかどうかの結果を表す数値を返します。			X
"MVALID 関数" (SAS 関数と CALL ルーチン: リファレンス)	SAS メンバー名として使用する文字列の有効性を確認します。			X
"NLDATE 関数" (p. 705)	SAS 日付値を、日付のディスクリプターを使用して指定したロケールの日付値に変換します。			X
"NLDATM 関数" (p. 708)	SAS 日時値を、日時形式のディスクリプターを使用して指定したロケールの時間値に変換します。			X
"NLTIME 関数" (p. 711)	SAS 時間値または SAS 日時値を、NLTIME ディスクリプターを使用して指定したロケールの時間値に変換します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"NLITERAL 関数" (SAS 関数と CALL ルーチン: リファレンス)	指定した文字列を SAS 名前リテラルに変換します。			X
"NOTALNUM 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から英数字ではない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTALPHA 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から英字でない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTCNTRL 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列からコントロール文字ではない文字を検索し、最初に検索された文字の位置を返します。			X
"NOTDIGIT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から数字ではない文字を検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。		X	
"NOTFIRST 関数" (SAS 関数と CALL ルーチン: リファレンス)	VALIDVARNAME=V7 の場合、SAS 変数名として無効な開始文字を文字列か		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	ら検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。			
"NOTGRAPH 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列からグラフィカル文字ではない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTLOWER 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から小文字ではない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTNAME 関数" (SAS 関数と CALL ルーチン: リファレンス)	VALIDVARNAME=V7 の場合、SAS 変数名として無効な文字を文字列から検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1 ステータスが割り		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。			
"NOTPRINT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から印刷不可文字を検索し、最初に検索された文字の位置を返します。	X		
"NOTPUNCT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から句読文字ではない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTSPACE 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から空白文字(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)でない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTUPPER 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から大文字ではない文字を検索し、最初に検索された文字の位置を返します。		X	
"NOTXDIGIT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から 16 進数文字ではない文字を検索し、最初に検索された文字の位置を返します。 注: この関数には、VARCHAR 変数がない場合、または関数がスレッド化されているか DS2 で実行されている場合を除き、I18N レベル 1		X	

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	ステータスが割り当てられます。これらの例外が発生した場合、この関数には I18N レベル 2 ステータスが割り当てられます。			
"INVALID 関数" (SAS 関数と CALL ルーチン: リファレンス)	SAS 変数名として使用する文字列の有効性を確認します。			X
"PROPCASE 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数のすべての単語を適切に大文字と小文字に変換します。			X
"PRXCHANGE 関数" (SAS 関数と CALL ルーチン: リファレンス)	パターンマッチング置換を実行します。			X
"PRXMATCH 関数" (SAS 関数と CALL ルーチン: リファレンス)	パターンの一致を検索し、見つかったパターンの位置を返します。			X
"PRXPAREN 関数" (SAS 関数と CALL ルーチン: リファレンス)	パターン内に一致が存在する最後のブラケットマッチを返します。			X
"PRXPARE 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列値のパターンマッチングに使用できる Perl 正規表現(PRX)をコンパイルします。			X
"PRXPOSN 関数" (SAS 関数と CALL ルーチン: リファレンス)	キャプチャバッファの値を含んでいる文字列を返します。			X
"PUT 関数" (SAS 関数と CALL ルーチン: リファレンス)	指定された出力形式を使用して値を返します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"PUTC 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列の出力形式を実行時に指定できます。			X
"PUTN 関数" (SAS 関数と CALL ルーチン: リファレンス)	数値の出力形式を実行時に指定できます。			X
"QUOTE 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字値に二重引用符を付加します。			X
"RANK 関数" (SAS 関数と CALL ルーチン: リファレンス)	ASCII 照合順序または EBCDIC 照合順序による文字の位置を返します。	X		
"REPEAT 関数" (SAS 関数と CALL ルーチン: リファレンス)	最初の引数を n+1 回繰り返した文字値を返します。			X
"REVERSE 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列を逆にします。	X		
"RIGHT 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字式を右詰めにします。			X
"SASMSG 関数" (p. 712)	データセットからメッセージを返します。返されるメッセージは現在のロケールと指定されたキーに基づきます。			X
"SASMSG1 関数" (p. 716)	データセットからメッセージを返します。メッセージは、指定されたロケール値と指定されたキー値に基づきます。			X
"SCAN 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列から n 番目の単語を返します。	X		
"SETLOCALE 関数" (p. 718)	現在の SAS ロケールにロケールキーを指定します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
"SHA256 関数" (SAS 関数と CALL ルーチン: リファレンス)	指定した文字列のメッセージダイジェストの結果を返します。			X
"SORTKEY 関数" (p. 726)	言語並べ替えキーを作成します。			X
"SOUNDEX 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列をエンコードして検索しやすくします。	X		
"SPEDIS 関数" (SAS 関数と CALL ルーチン: リファレンス)	2 つの単語の一致尤度を調べて、2 つの単語間のスペルの違いをコストで表します。	X		
"STRIP 関数" (SAS 関数と CALL ルーチン: リファレンス)	先頭と末尾の空白を削除して文字列を返します。	X		
"SUBPAD 関数" (SAS 関数と CALL ルーチン: リファレンス)	必要に応じて空白埋め込みを使用し、指定した長さの部分文字列を返します。			X
"擬似 SUBSTR(割り当ての左辺に用いた場合) 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字値の内容を置換します。	X		
"擬似 SUBSTR(割り当ての右辺に用いた場合) 関数" (SAS 関数と CALL ルーチン: リファレンス)	引数から部分文字列を抽出します。	X		
"SUBSTRN 関数" (SAS 関数と CALL ルーチン: リファレンス)	部分文字列を返します。長さがゼロの結果も返せます。	X		
"TRANSLATE 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列に含まれる特定の文字を置き換えます。	X		
"TRANSTRN 関数" (SAS 関数と CALL ルーチン: リファレンス)	文字列に含まれる特定の部分文字列をすべて置き換えるか取り除きます。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
“TRANTAB 関数” (730 ページ)	指定された変換テーブルを使用してデータをトランスコードします。	X		
“TRANWRD 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列に含まれる特定の部分文字列をすべて置き換えるか取り除きます。			X
“TRIM 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字列から末尾の空白を取り除きます。文字列が欠損値の場合は、1 つの空白を返します。			X
“TRIMN 関数” (SAS 関数と CALL ルーチン: リファレンス)	文字式から末尾の空白を取り除きます。文字式が欠損値の場合は、長さがゼロの文字列を返します。			X
“TZONEDSTNAME 関数” (p. 737)	夏時間名を返します。			X
“TZONEDSTOFF 関数” (p. 738)	指定された夏時間のタイムゾーンオフセット値を返します。			X
“TZONEID 関数” (p. 731)	現在のタイムゾーン ID を返します。			X
“TZONENAME 関数” (p. 732)	現在の標準時間または夏時間のタイムゾーン名を返します。			X
“TZONEOFF 関数” (p. 734)	ユーザーのタイムゾーンオフセットを返します。			X
“TZONES2U 関数” (p. 736)	SAS 日時値を UTC 日時値へ変換します。			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
“TZONESTTNAME 関数” (p. 739)	標準のタイムゾーン名を返します。			X
“TZONESTTOFF 関数” (p. 740)	指定された標準時間のタイムゾーンオフセット値を返します。			X
“TZONEU2S 関数” (p. 741)	UTC 日時値を SAS 日時値へ変換します。			X
“UNICODE 関数” (p. 742)	Unicode 文字を現在の SAS セッションエンコーディングに変換します。			X
“UNICODEC 関数” (p. 745)	現在の SAS セッションエンコーディングの文字を Unicode 文字に変換します。			X
“UNICODELEN 関数” (p. 748)	Unicode データの文字単位の長さを示します。			X
“UNICODEWIDTH 関数” (p. 749)	Unicode データの表示単位の長さを示します。			X
“UPCASE 関数” (SAS 関数と CALL ルーチン: リファレンス)	引数内のすべての文字を大文字に変換します。			X
“URLDECODE 関数” (SAS 関数と CALL ルーチン: リファレンス)	URL のエスケープ構文を使用してデコードされた文字列を返します。		X	
“URLENCODE 関数” (SAS 関数と CALL ルーチン: リファレンス)	URL のエスケープ構文を使用してエンコードされた文字列を返します。		X	
“VERIFY 関数” (SAS 関数と CALL ルーチン: リファレンス)	他の文字列に存在しない文字の最初			X

関数	説明	I18N レベル 0	I18N レベル 1	I18N レベル 2
	の出現位置を返します。			
"WHICHC 関数" (SAS 関数と CALL ルーチン: リファレンス)	1 番目の引数と等しい文字列値を検索し、最初に一致した値のインデックスを返します。			X

関数での VARCHAR サポート

文字列関数の VARCHAR データ型 579

文字列関数の VARCHAR データ型

SAS は DATA ステップおよび PROC ステップでの文字データ処理で VARCHAR データ型をサポートしています。変数の種類を維持するには、CAS エンジンによる VARCHAR データを読み取りおよび書き取りが必要です。VARCHAR データ型の変数は、可変幅であり、文字としてのセマンティックスに従います。SAS 本来の文字データ型と違って、固定幅ではなく、また、バイトとしてのセマンティックスでもありません。VARCHAR 変数を文字列関数(従来からある文字列関数、K 関数を含む)に渡すと、SAS はその長さが文字数を表していると想定します。VARCHAR 変数のオフセットは、文字列中の文字の位置を表します。固定幅文字データ型は、バイナリデータに向いています。

VARCHAR は CAS テーブルでサポートされ、data_NULL_step で使われます。VARCHAR カラムを扱えるプロシジャもあります。VARCHAR カラムに非対応のプロシジャは、カラムを固定幅文字カラムに変換します。データセットで VARCHAR を扱うことはできません。

SUBSTR 関数 substr(X, 10, 2)でこのデータ型を使っている例を示します。

変数 X が固定幅文字変数として定義されている場合、SUBSTR は文字列中の 10 番目と 11 番目のバイトを返します。X が VARCHAR 変数として定義されている場合、SUBSTR は文字列中の 10 番目と 11 番目の文字を返します。

データ型が長さ 10 の VARCHAR である変数は、10 個までの文字を保持できます。

このテーブルに、固定幅データ型と VARCHAR データ型の違いを示します。固定幅文字データ型はバイトとしてのセマンティックス、VARCHAR は文字としてのセマンティックスに従います。

データ型	従来型の文字列関数	K 関数
固定幅の文字	バイト	文字

データ型	従来型の文字列関数	K 関数
VARCHAR	文字	文字

SAS Viya プラットフォームセッションのエンコーディングである UTF-8 は、1 バイトのデータもマルチバイトのデータも扱えます。マルチバイトの UTF-8 文字の例としてユーロ文字(€)があり、これは 3 バイトを占めます。文字列€123 の表記を次の表に示します。文字€は 3 バイト、数字はそれぞれ 1 バイトを占めます。

文字	€			1	2	3
16 進値	E2	82	AC	31	32	33

次の例では、固定幅のデータ型と VARCHAR データ型を比較します。[表 12.1 \(580 ページ\)](#)に、データ€123 の固定幅変数から得た結果を示します。[表 12.2 \(581 ページ\)](#)に、VARCHAR を使用した場合の結果を示します。

K 関数は固定幅文字データ(マルチバイト文字が含まれていても可)を対象とします。

- 例の 1 行目では、固定幅文字カラムに文字列を代入しています。PUT ステートメントは文字を表示し、さらに対応する 16 進値も表示します。
- 例の 2 行目では、SUBSTR 関数に位置として 2 を渡しています。変数 x は固定幅文字カラムなので、この位置は x の文字列中 2 番目のバイトと解釈されます。すなわち、文字€を構成する 2 番目のバイトです。したがって、xsub に代入される先頭 2 バイトは、UTF-8 データとしては無効です。xsub の先頭 2 バイトとしてここに表示される結果はごみ(あるいは Unicode の代替文字)です。
- 例の 3 行目では、INDEX 関数を使って文字列中の'1'を検索しています。変数 x は固定幅文字カラムなので、INDEX 関数の戻り値は、文字列中の該当する文字のバイト位置です。結果は **xidx=4** となります。x 中、'1'は 4 番目のバイトにあるからです。

K 関数は固定幅文字データ(マルチバイト文字が含まれていても可)を対象とします。

表 12.1 固定幅文字の例

ステートメント	結果
data _null; x='€123'; put x= x= \$hex6; xsub=substr(x,2); put xsub= xsub= \$hex; xidx=index(x,'1'); put xidx=; run;	x=€123 x=E282AC xsub=~123 xsub=82AC31323320 xidx=4

- 例の 1 行目では、VARCHAR カラムに文字列を代入しています。PUT ステートメントは文字を表示し、さらに対応する 16 進値も表示します。

各国語サポート関連の関数のディクショナリ

カテゴリ別の関数	584
ディクショナリ	591
ANORM420 関数	591
CALL KSCANX ルーチン	596
BASECHAR 関数	601
ENCODCOMPAT 関数	603
ENCODISVALID 関数	605
GETLOCENV 関数	606
GETPXLANGUAGE 関数	607
GETPXLOCALE 関数	608
GETPXREGION 関数	610
KCHARLIST 関数	611
KCOMPARE 関数	614
KCOMPRESS 関数	616
KCOUNT 関数	618
KCOUNTC 関数	619
KCOUNTW 関数	622
KCOUNTX 関数	626
KCVT 関数	628
KFIND 関数	630
KFINDC 関数	632
KFINDW 関数	636
KINDEX 関数	641
KINDEXB 関数	643
KINDEXCB 関数	645
KINDEXC 関数	647
KLEFT 関数	649
KLENGTH 関数	652
KLOWCASE 関数	653
KNEXTCHAR 関数	655
KOFFSET 関数	657
KPROPCASE 関数	658
KPROPCHAR 関数	662

KPROPDATA 関数	663
KREVERSE 関数	668
KRIGHT 関数	669
KSCAN 関数	670
KSCANX 関数	673
KSTRCAT 関数	678
KSTRIP 関数	680
KSUBSTR 関数	682
KSUBSTRB 関数	684
KSUBSTRN 関数	686
KTRANSLATE 関数	687
KTRIM 関数	689
KTRUNCATE 関数	691
KUPCASE 関数	692
KUPDATE 関数	694
KUPDATEB 関数	697
KUPDATES 関数	699
KVERIFY 関数	702
KVERIFYB 関数	703
NLDATE 関数	705
NLDATM 関数	708
NLTIME 関数	711
SASMSG 関数	712
SASMSG1 関数	716
SETLOCALE 関数	718
SORTKEY 関数	726
TRANTAB 関数	730
TZONEID 関数	731
TZONENAME 関数	732
TZONEOFF 関数	734
TZONES2U 関数	736
TZONEDSTNAME 関数	737
TZONEDSTOFF 関数	738
TZONESTTNAME 関数	739
TZONESTTOFF 関数	740
TZONEU2S 関数	741
UNICODE 関数	742
UNICODEC 関数	745
UNICODELEN 関数	748
UNICODEWIDTH 関数	749
VARTRANSCODE 関数	750
VTRANSCODE 関数	752
VTRANSCODEX 関数	753

カテゴリ別の関数

一部の関数は SAS 内でのみ実行され、一部の関数は SAS 内および CAS エンジン上で実行されます。関数カテゴリとして CAS を指定した場合、関数は SAS 内および CAS サーバー上で実行されます。関数カテゴリとして CAS を指定しない場合、関数は

SAS 内でのみ実行されます。たとえば、BASECHAR 関数は SAS 内および CAS サーバー上で実行されるため、カテゴリとして CAS を指定します。KCVT 関数は SAS でしか実行されないため、カテゴリとして CAS を指定しません。

各国語サポートに関するカテゴリ:

表 13.1 各国語サポート関連の関数のカテゴリ

カテゴリ	説明
CAS	CAS サーバー上で実行される関数。 関数のリストについては、 CAS を参照してください。
文字	文字データを処理します。 関数のリストについては、 文字 を参照してください。
文字操作	任意のエンコーディングで文字データを処理し、マルチバイト文字を処理します。 関数のリストについては、 文字操作 を参照してください。
DBCS	2 バイト文字セットを処理します。 関数のリストについては、 DBCS を参照してください。
日付と時間	日時データを処理します。 関数のリストについては、 日付と時間 を参照してください。
ロケール	指定されたロケールに基づいてデータを処理します。 関数のリストについては、 ロケール を参照してください。
変数情報	変数情報を処理します。 関数のリストについては、 変数情報 を参照してください。

カテゴリ	言語要素	説明
CAS	CALL KSCANX ルーチン (p. 596)	文字列から n 番目の単語の位置と長さを返します。
	BASECHAR 関数 (p. 601)	文字を基本文字に変換します。
	KCHARLIST 関数 (p. 611)	修飾子に従って文字リストを返します。

カテゴリ	言語要素	説明
	KCOMPARE 関数 (p. 614)	文字式の比較結果を返します。
	KCOMPRESS 関数 (p. 616)	指定された文字を文字式から削除します。
	KCOUNT 関数 (p. 618)	式の 2 バイト文字の数を返します。
	KCOUNTC 関数 (p. 619)	文字列に含まれる個々の文字を数えます。
	KCOUNTW 関数 (p. 622)	文字列内の単語の数を数えます。
	KCOUNTX 関数 (p. 626)	指定した部分文字列が文字列内に含まれる回数を数えます。
	KCVT 関数 (p. 628)	データがある種類のエンコーディングデータから別の種類のエンコーディングデータに変換します。
	KFIND 関数 (p. 630)	文字列内の特定の部分文字列を検索します。
	KFINDC 関数 (p. 632)	文字のリストにある各文字を文字列から検索します。
	KFINDW 関数 (p. 636)	文字列内の単語の文字位置か、文字列内の単語の番号を返します。
	KINDEX 関数 (p. 641)	文字式から文字列を検索します。
	KINDEXC 関数 (p. 647)	文字式で指定された文字を検索し、文字ベースの値を返します。
	KLEFT 関数 (p. 649)	不要な先頭の DBCS の空白と SO/SI を削除し、文字式を左詰めにします。
	KLENGTH 関数 (p. 652)	引数の長さを文字数で返します。
	KLOWCASE 関数 (p. 653)	大文字のアルファベットを小文字に変換します。
	KNEXTCHAR 関数 (p. 655)	指定されたバイトオフセットで文字列から文字を返します。
	KOFFSET 関数 (p. 657)	character position からのバイトオフセットを返します。
	KREVERSE 関数 (p. 668)	文字式を逆にします。
	KRIGHT 関数 (p. 669)	末尾の DBCS の空白と SO/SI を削除し、文字式を右詰めにします。
	KSCAN 関数 (p. 670)	指定された単語を文字式から選択します。
	KSCANX 関数 (p. 673)	関数のアクションを処理する修飾子を使用して、文字式から指定された単語を選択します。
	KSTRCAT 関数 (p. 678)	複数の文字式を連結します。
	KSTRIP 関数 (p. 680)	文字列から先頭と末尾の空白を削除します。
	KSUBSTR 関数 (p. 682)	引数から部分文字列を抽出します。

カテゴリ	言語要素	説明
	KTRANSLATE 関数 (p. 687)	文字式内の指定された文字を置換します。
	KTRIM 関数 (p. 689)	末尾の DBCS の空白と SO/SI を文字式から削除します。
	KUPCASE 関数 (p. 692)	小文字のアルファベットを大文字に変換します。
	KUPDATE 関数 (p. 694)	文字値の内容を置換します。
	KUPDATES 関数 (p. 699)	文字値の内容を置換します。
	KVERIFY 関数 (p. 702)	式に固有の最初の文字（文字ベースの値）の位置を返します。
	UNICODE 関数 (p. 742)	Unicode 文字を現在の SAS セッションエンコーディングに変換します。
	UNICODEC 関数 (p. 745)	現在の SAS セッションエンコーディングの文字を Unicode 文字に変換します。
	UNICODELEN 関数 (p. 748)	Unicode データの文字単位の長さを示します。
	UNICODEWIDTH 関数 (p. 749)	Unicode データの表示単位の長さを示します。
	VTRANSCODE 関数 (p. 752)	指定された文字変数でトランスコーディングが有効かどうかを示す値を返します。
	VTRANSCODEX 関数 (p. 753)	指定された引数でトランスコーディングが有効かどうかを示す値を返します。
DBCS	CALL KSCANX ルーチン (p. 596)	文字列から n 番目の単語の位置と長さを返します。
	KLEFT 関数 (p. 649)	不要な先頭の DBCS の空白と SO/SI を削除し、文字式を左詰めにします。
	KTRIM 関数 (p. 689)	末尾の DBCS の空白と SO/SI を文字式から削除します。
エンコーディング	ENCODCOMPAT 関数 (p. 603)	2 つのエンコーディング間のトランスコーディング互換性を確認します。
	ENCODISVALID 関数 (p. 605)	有効なエンコーディング名であるかどうかを検証します。
日付と時間	NLDATE 関数 (p. 705)	SAS 日付値を、日付のディスクリプターを使用して指定したロケールの日付値に変換します。
	NLDATM 関数 (p. 708)	SAS 日時値を、日時形式のディスクリプターを使用して指定したロケールの時間値に変換します。
	NLTIME 関数 (p. 711)	SAS 時間値または SAS 日時値を、NLTIME ディスクリプターを使用して指定したロケールの時間値に変換します。
	TZONEID 関数 (p. 731)	現在のタイムゾーン ID を返します。

カテゴリ	言語要素	説明
	TZONENAME 関数 (p. 732)	現在の標準時間または夏時間のタイムゾーン名を返します。
	TZONEOFF 関数 (p. 734)	ユーザーのタイムゾーンオフセットを返します。
	TZONES2U 関数 (p. 736)	SAS 日時値を UTC 日時値へ変換します。
	TZONEDSTNAME 関数 (p. 737)	夏時間名を返します。
	TZONEDSTOFF 関数 (p. 738)	指定された夏時間のタイムゾーンオフセット値を返します。
	TZONESTTNAME 関数 (p. 739)	標準のタイムゾーン名を返します。
	TZONESTTOFF 関数 (p. 740)	指定された標準時間のタイムゾーンオフセット値を返します。
変数情報	TZONEU2S 関数 (p. 741)	UTC 日時値を SAS 日時値へ変換します。
	VARTRANSCODE 関数 (p. 750)	SAS データセット変数のトランスコーディング属性を返します。
	VTRANSCODE 関数 (p. 752)	指定された文字変数でトランスコーディングが有効かどうかを示す値を返します。
文字	VTRANSCODEX 関数 (p. 753)	指定された引数でトランスコーディングが有効かどうかを示す値を返します。
	ANORM420 関数 (p. 591)	EBCDIC420 でエンコードされた入力文字列から正規化された文字列を返します。
	BASECHAR 関数 (p. 601)	文字を基本文字に変換します。
	KCVT 関数 (p. 628)	データのある種類のエンコーディングデータから別の種類のエンコーディングデータに変換します。
	KSTRIP 関数 (p. 680)	文字列から先頭と末尾の空白を削除します。
	TRANTAB 関数 (p. 730)	指定された変換テーブルを使用してデータをトランスコードします。
	UNICODE 関数 (p. 742)	Unicode 文字を現在の SAS セッションエンコーディングに変換します。
	UNICODEC 関数 (p. 745)	現在の SAS セッションエンコーディングの文字を Unicode 文字に変換します。
	UNICODELEN 関数 (p. 748)	Unicode データの文字単位の長さを示します。
文字操作	UNICODEWIDTH 関数 (p. 749)	Unicode データの表示単位の長さを示します。
	KCHARLIST 関数 (p. 611)	修飾子に従って文字リストを返します。

カテゴリ	言語要素	説明
	KCOMPARE 関数 (p. 614)	文字式の比較結果を返します。
	KCOMPRESS 関数 (p. 616)	指定された文字を文字式から削除します。
	KCOUNT 関数 (p. 618)	式の 2 バイト文字の数を返します。
	KCOUNTC 関数 (p. 619)	文字列に含まれる個々の文字を数えます。
	KCOUNTW 関数 (p. 622)	文字列内の単語の数を数えます。
	KCOUNTX 関数 (p. 626)	指定した部分文字列が文字列内に含まれる回数を数えます。
	KFIND 関数 (p. 630)	文字列内の特定の部分文字列を検索します。
	KFINDC 関数 (p. 632)	文字のリストにある各文字を文字列から検索します。
	KFINDW 関数 (p. 636)	文字列内の単語の文字位置か、文字列内の単語の番号を返します。
	KINDEX 関数 (p. 641)	文字式から文字列を検索します。
	KINDEXB 関数 (p. 643)	文字式で指定された文字を検索し、バイトベースの値を返します。
	KINDEXCB 関数 (p. 645)	文字式で指定された文字を検索し、バイトベースの値を返します。
	KINDEXC 関数 (p. 647)	文字式で指定された文字を検索し、文字ベースの値を返します。
	KLENGTH 関数 (p. 652)	引数の長さを文字数で返します。
	KLOWCASE 関数 (p. 653)	大文字のアルファベットを小文字に変換します。
	KNEXTCHAR 関数 (p. 655)	指定されたバイトオフセットで文字列から文字を返します。
	KOFFSET 関数 (p. 657)	character position からのバイトオフセットを返します。
	KPROPCASE 関数 (p. 658)	中国語、日本語、韓国語、台湾語(CJKT)文字を変換します。
	KPROPCHAR 関数 (p. 662)	Unicode を、括弧で囲まれた対応する文字に変換します。
	KPROPDATA 関数 (p. 663)	印刷不可文字を削除または変換します。
	KREVERSE 関数 (p. 668)	文字式を逆にします。
	KRIGHT 関数 (p. 669)	末尾の DBCS の空白と SO/SI を削除し、文字式を右詰めにします。
	KSCAN 関数 (p. 670)	指定された単語を文字式から選択します。
	KSCANX 関数 (p. 673)	関数のアクションを処理する修飾子を使用して、文字式から指定された単語を選択します。

カテゴリ	言語要素	説明
	KSTRCAT 関数 (p. 678)	複数の文字式を連結します。
	KSUBSTR 関数 (p. 682)	引数から部分文字列を抽出します。
	KSUBSTRB 関数 (p. 684)	引数内の部分文字列のバイト位置に従って、引数から部分文字列を抽出します。
	KSUBSTRN 関数 (p. 686)	部分文字列を返します。長さがゼロの結果も返せます。
	KTRANSLATE 関数 (p. 687)	文字式内の指定された文字を置換します。
	KTRUNCATE 関数 (p. 691)	マルチバイト文字を壊すことなく、文字列をバイト単位で、指定した長さに切ります。
	KUPCASE 関数 (p. 692)	小文字のアルファベットを大文字に変換します。
	KUPDATE 関数 (p. 694)	文字値の内容を置換します。
	KUPDATEB 関数 (p. 697)	引数内の文字値のバイト位置に従って、文字値の内容を置換します。
	KUPDATES 関数 (p. 699)	文字値の内容を置換します。
	KVERIFY 関数 (p. 702)	式に固有の最初の文字（文字ベースの値）の位置を返します。
	KVERIFYB 関数 (p. 703)	式に固有の最初の文字（バイトベースの値）の位置を返します。
ロケール	GETLOCALENV 関数 (p. 606)	現在のロケール/言語環境を返します。
	GETPXLANGUAGE 関数 (p. 607)	現在の 2 文字の言語コードを返します。
	GETPXLOCALE 関数 (p. 608)	SAS ロケールの POSIX ロケール値を返します。
	GETPXREGION 関数 (p. 610)	現在の 2 文字の地域コードを返します。
	SASMSG 関数 (p. 712)	データセットからメッセージを返します。返されるメッセージは現在のロケールと指定されたキーに基づきます。
	SASMSG1 関数 (p. 716)	データセットからメッセージを返します。メッセージは、指定されたロケール値と指定されたキー値に基づきます。
	SETLOCALE 関数 (p. 718)	現在の SAS ロケールにロケールキーを指定します。
	SORTKEY 関数 (p. 726)	言語並べ替えキーを作成します。

ディクショナリ

ANORM420 関数

EBCDIC420 でエンコードされた入力文字列から正規化された文字列を返します。

カテゴリ: 文字

制限事項: この関数には I18N レベル 1 のステータスが付与されています。英語以外の言語を使用する場合、可能な限り I18N レベル 1 の関数の使用は避けてください。特定の環境下では、I18N レベル 1 の関数は、2 バイト文字セット(DBCS)または複数バイト文字セット(MBCS)エンコーディングを使用すると正常に動作しない場合があります。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

ANORM420 (*string*,<*modifiers*>)

必須引数

string

入力文字列が EBCDIC420 でエンコードされるように指定します。

オプション引数

modifiers

1 つ以上の修飾子を指定する文字定数、変数、または式。次の修飾子は大文字でも小文字でも指定できます。

- g 合字を無視します
- i アラビアインド数字を無視します
- p 形状文字のマッピングを無視します
- s 空白の追加を無視します
- t トランスコーティングを無視します
- z ゼロ幅空白文字のマッピングを無視します

詳細

ANORM420 関数は、EBCDIC420 でエンコードされたデータを処理します。

修飾子の変数については、次の表を参照してください。

ANORM420 関数は、修飾子 *g* が指定されていない場合は、次の文字とコードポイントを適切な合字に置き換えます。

表 13.2 修飾子 *g*: アラビア語合字

文字	ユニコード値	開始	終了	説明
ﻻ	(U+FEF5)	0xB2	0xB1 + 0x47	アラビア合字 Lam-Aleph(上部 Madda)独立形
ﻻ	(U+FEF6)	0xB3	0xB1 + 0x47	アラビア合字 Lam-Aleph(上部 Madda)語末形
ﻻ	(U+FEF7)	0xB4	0xB1 + 0x49	アラビア合字 Lam-Aleph(上部 Hamza)独立形
ﻻ	(U+FEF8)	0xB5	0xB1 + 0x49	アラビア合字 Lam-Aleph(上部 Hamza)語末形
ﻻ	(U+FEFC)	0xB8	0xB1 + 0x56	アラビア合字 Lam-Aleph 独立形
ﻻ	(U+FEFD)	0xB9	0xB1 + 0x56	アラビア合字 Lam-Aleph 語末形

修飾子 *i* を指定せず、トランスコーディングが発生しない場合、ANORM420 はアラビアインド数字を数字に再マッピングします。

表 13.3 修飾子 *i*: アラビアインド数字

開始	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
	U+0660	U+0661	U+0662	U+0663	U+0664	U+0665	U+0666	U+0667	U+0668	U+0669
16 進数	0xDF	0xEA	0xEB	0xED	0xEE	0xEF	0xFB	0xFC	0xFD	0xFE
終了	0	1	2	3	4	5	6	7	8	9
16 進数	0xF0	0xF1	0xF2	0xF3	0xF4	0xF5	0xF6	0xF7	0xF8	0xF9

修飾子 p を指定しない場合、ANORM420 関数は形状文字に対応する非形状文字に再マッピングします。

表 13.4 修飾子 p: 形状文字のマッピング

開始		終了	
0x43	U+FE7D	0x42	U+0651
0x48	FE82	0x47	U+0622
0x51	U+FE84	0x49	U+0623
0x57	U+FE8E	0x56	U+0627
0x59	U+FE91	0x58	U+0628
0x64	U+FE97	0x63	U+062A
0x66	U+FE9B	0x65	U+062B
0x68	U+FE9F	0x67	U+062C
0x70	U+FEA3	0x69	U+062D
0x72	U+FEA7	0x71	U+062E
0x78	U+FEB3	0x77	U+0633
0x8A	U+FEB7	0x80	U+0634
0x8C	U+FEBB	0x8B	U+0635
0x8E	U+FEBF	0x8D	U+0636
0x9B	U+FECA	0x9A	U+0639
0x9C	U+FECB	0x9A	U+0639
0x9D	U+FECC	0x9A	U+0639
0x9F	U+FECE	0x9E	U+063A
0xA0	U+FECF	0x9E	U+063A
0xAA	U+FED0	0x9E	U+063A
0xAC	U+FED3	0xAB	U+0641

開始		終了	
0xAE	U+FED7	0xAD	U+0642
0xB0	U+FEDB	0xAF	U+0643
0xBA	U+FEDF	0xB1	U+0644
0xBC	U+FEE3	0xBB	U+0645
0xBE	U+FEE7	0xBD	U+0646
0xCB	U+FEED	0xBF	U+0647
0xCD	U+FEED	0xBF	U+0647
0xDB	U+FEF0	0xDA	U+061F
0xDD	U+FEF2	0xDC	U+064A
0xDE	U+FEF3	0xDC	U+064A

修飾子 *s* を指定していない場合、ANORM420 関数は次の文字の後に空白を追加します。

表 13.5 修飾子 *s*: 空白の追加を無視

ب	0x58	(U+0628)	アラビア文字 BEH
ة	0x62	(U+0629)	アラビア文字 TEH MARBUTA
ت	0x63	(U+062A)	アラビア文字 TEH
ث	0x65	(U+062B)	アラビア文字 THEH
ج	0x67	(U+062C)	アラビア文字 JEEM
ح	0x69	(U+062D)	アラビア文字 HAH
خ	0x71	(U+062E)	アラビア文字 KHAH
س	0x77	(U+0633)	アラビア文字 SEEN
ش	0x80	(U+0634)	アラビア文字 SHEEN

ص	0x8B	(U+0635)	アラビア文字 SAD
ض	0x8D	(U+0636)	アラビア文字 DAD
ع	0x9A	(U+0639)	アラビア文字 AIN
ع	0x9B	(U+FECA)	アラビア文字 AIN 語末形
غ	0x9E	(U+063A)	アラビア文字 GHAIN
غ	0x9F	(U+FECE)	アラビア文字 GHAIN 語末形
ف	0xAB	(U+0641)	アラビア文字 FEH
ق	0xAD	(U+0642)	アラビア文字 QAF
ك	0xAF	(U+0643)	アラビア文字 KAF
ل	0xB1	(U+0644)	アラビア文字 LAM
م	0xBB	(U+0645)	アラビア文字 MEEM
ن	0xBD	(U+0646)	アラビア文字 NOON
ه	0xBF	(U+0647)	アラビア文字 HEH
ى	0xDA	(U+0649)	アラビア文字 ALEF MAKSURA
ي	0xDC	(U+064A)	アラビア文字 YEH
ى	0xDB	(U+FEF0)	アラビア文字 ALEF MAKSURA 語末形
ي	0xDD	(U+FEF2)	アラビア文字 YEH 語末形

修飾子 t を指定していない場合、ANORM420 関数は正規化された文字列をセッションエンコーディングにトランスコーディングします。

修飾子 z を指定しない場合は、ANORM420 関数は ZWSP と呼ばれるゼロ幅空白(U+200B)を再マッピングします。ゼロ幅文字は、コード位置 0x45 に配置されています。

例

```
data _null_ ;
  a = '59CD57BC577745'x ;
  s1 = anorm420(a) ;
  /* Turn off addition of space and mapping of
  Arabic-Indic numbers */
  s2 = anorm420(a,"si") ;
  /* Turn off transcoding */
  s3 = anorm420(a,'t') ;
  put s1= $hex20. / s2= $hex20. / s3=$hex20. ;
run;
```

これらのステートメントは、次の結果を生成します。

```
s1=D8A8D987D8A7D985D8A7
s2=D8A8D987D8A7D985D8A7
s3=58BF56BB567740404040
```

CALL KSCANX ルーチン

文字列から n 番目の単語の位置と長さを返します。

カテゴリ: CAS
DBCS

注: CALL KSCANX は、マルチバイトデータを処理するように設計されています。ただし、最初の引数 *string* が 1 バイト文字の場合、CALL KSCANX ルーチンは SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ) を参照してください。

更新される引数の種類は、CALL ルーチン内で一致する必要があります。すべての引数の種類は、CHAR、VARCHAR または NUMERIC である必要があります。引数の種類が一致しない場合、ログに警告が発行されます。

構文

CALL KSCANX(*<string>*, *count*, *position*, *length*,*<character-list>*,*<modifier(s)>*);

必須引数

string

文字定数、変数または式を指定します。

count

整数値を使用するゼロ以外の数値の定数、変数または式を指定します。CALL KSCANX で選択する文字列中の単語数を整数値で指定します。たとえば、値 1 は

1 番目の単語を示し、値 2 は 2 番目の単語を示します。次のルールが適用されます。

- *count* が正の場合、CALL KSCANX は文字列の単語を左から右へ数えます。
- *count* が負の場合、CALL KSCANX は文字列の単語を右から左へ数えます。

position

単語の位置を返す数値変数を指定します。*count* が文字列内の単語数を超えると、*position* で返される値は 0 になります。*count* が 0 または欠損している場合、*position* で返される値は欠損します。

length

単語の長さを返す数値変数を指定します。*count* が文字列内の単語数を超えると、*length* で返される値は 0 になります。*count* が 0 または欠損している場合、*length* で返される値は欠損します。

オプション引数

character-list

文字のリストを初期化する文字式を指定します(省略可能)。このリストは、単語を区切る区切り文字として使用する文字を決定します。次のルールが適用されます。

- デフォルトでは、*character-list* のすべての文字が区切り文字として使用されます。
- *modifier* 引数で K 修飾子を指定すると、*character-list* にないすべての文字が区切り文字として使用されます。

ヒント その他の修飾子を使用して *character-list* にさらに文字を追加できます。

modifier(s)

空白以外の各文字によって CALL KSCANX のアクションが変更される、文字定数、変数または式を指定します。空白は無視されます。次の文字を修飾子として使用します。

- a または A 文字のリストにアルファベット文字を追加します。
- b または B *count* 引数の符号に関係なく、左から右ではなく、右から左へ逆方向にスキャンします。
- c または C 文字のリストに制御文字を追加します。
- d または D 文字のリストに数字を追加します。
- f または F 文字のリストにアンダースコアと英文字(VALIDVARNAME=V7 を使用した SAS 変数名内の有効な最初の文字)を追加します。
- g または G 文字のリストにグラフィカル文字を追加します。グラフィカル文字は、紙面に印刷するとイメージになる文字です。

h また は H	文字のリストに水平タブを追加します。
i また は I	大文字か小文字かは無視します。
k また は K	文字のリストにないすべての文字が区切り文字として扱われます。つまり、K を指定すると、文字のリストにある文字は、区切り文字であるため省略されるのではなく、戻り値に保持されます。K を指定しなければ、文字のリストにあるすべての文字が区切り文字として扱われます。
l また は L	文字のリストに小文字が追加されます。
m ま たは M	複数の連続する区切り文字、および <i>string</i> 引数の先頭または末尾の区切り文字が、長さが 0 の単語を参照するように指定します。M 修飾子を指定しない場合、複数の連続する区切り文字は 1 つの区切り文字として扱われ、 <i>string</i> 引数の先頭または末尾の区切り文字は無視されます。
n また は N	文字のリストに数字、アンダースコアおよび英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内に表示可能な文字) を追加します。
o また は O	<i>character-list</i> および <i>modifier</i> 引数が、CALL KSCANX ルーチン呼び出すたびにではなく、1 回だけ処理されます。DATA ステップで O 修飾子を使用すると、 <i>character-list</i> および <i>modifier</i> 引数が変更されないため、CALL KSCANX をループで呼び出す際により速く実行させられます。O 修飾子は、SAS コードの CALL KSCANX ルーチンの各インスタンスに個別に適用されます。この場合、CALL KSCANX ルーチンのすべてのインスタンスで同じ区切り文字と修飾子が使用されるわけではありません。
p また は P	文字のリストに句読点を追加します。
q また は Q	引用符で囲まれた部分文字列内の区切り文字は無視します。 <i>string</i> 引数の値に一致しない引用符が含まれている場合は、左から右へのスキャンによって、右から左へのスキャンとは異なる単語が生成されます。
s また は S	文字のリストに空白文字(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)を追加します。
t また は T	<i>string</i> 引数と <i>character-list</i> 引数から末尾の空白を取り除きます。両方の文字引数ではなく 1 つの文字引数から末尾の空白を削除する場合は、T 修飾子付きの CALL KSCANX ルーチンのかわりに TRIM 関数を使用します。
u また は U	文字のリストに大文字が追加されます。
w また は W	文字のリストに印刷可能(書き込み可能)文字が追加されます。
x また は X	文字のリストに 16 進文字を追加します。

ヒント *modifier* 引数が文字定数の場合は引用符で囲みます。一組の引用符で複数の修飾子を指定します。*modifier* 引数は、文字変数または文字式としても表すことができます。

詳細

区切り文字と単語の定義

単語を区切るために使用されるいくつかの文字はいずれも *区切り文字* と呼ばれます。*character-list* および *modifier* 引数に区切り文字を指定できます。

CALL KSCANX ルーチンでは、次の条件をすべて満たす部分文字列を単語とします。

- 左境界が、区切り文字または文字列の先頭である。
- 右境界が、区切り文字または文字列の末尾である。
- 区切り文字を含まない。

単語は、その文字列の先頭または末尾に区切り文字がある場合、またはその文字列に 2 つ以上の連続した区切り文字が含まれる場合には長さが 0 となります。ただし、M 修飾子を指定しない限り、CALL KSCANX ルーチンでは長さが 0 の単語は無視されます。

ASCII 環境でデフォルトの区切り文字を使用する

コンピュータで ASCII 文字が使われている場合、デフォルトの区切り文字は次のとおりです。blank!\$%&()*+,-./;<^.^文字が含まれない ASCII 環境では、CALL KSCANX ルーチンはかわりに~文字を使用します。

区切り文字とする文字を指定せずに *modifier* 引数を使用すると、使用される区切り文字は *modifier* 引数で定義される文字のみになります。この場合、ASCII 環境のデフォルトの区切り文字のリストは使用されません。つまり、修飾子は *character-list* 引数で明示的に指定された区切り文字のリストに追加します。修飾子は、デフォルトの修飾子のリストには追加しません。

M 修飾子を指定して CALL KSCANX ルーチンを使用

M 修飾子を指定すると、文字列内の単語の数は、文字列内の区切り文字の数に 1 を加えたものとして定義されます。ただし、Q 修飾子を指定すると、引用符内にある区切り文字は無視されます。

M 修飾子を指定した場合、次のいずれかの条件が真のとき、CALL KSCANX ルーチンは正の位置と長さ 0 を返します。

- 文字列の先頭が区切り文字であり、ユーザーが最初の単語を要求した場合
- 文字列の末尾が区切り文字であり、ユーザーが最後の単語を要求した場合
- 文字列が 2 つの連続する区切り文字を含んでおり、ユーザーがこれら 2 つの区切り文字間にある単語を要求した場合

文字列内の単語数よりも絶対値が大きい数を指定した場合、CALL KSCANX ルーチンは位置と長さを 0 として返します。

M 修飾子なしで CALL KSCANX ルーチンを使用

M 修飾子を指定しなければ、文字列内の単語の数は、連続する区切り文字ではない文字の最大部分文字列の数として定義されます。ただし、Q 修飾子を指定すると、引用符内にある区切り文字は無視されます。

M 修飾子を指定しない場合、CALL KSCANX ルーチンは次の処理を行います。

- 文字列の先頭または末尾の区切り文字を無視する。
- 2 つ以上の連続する区切り文字を単一の区切り文字として扱う。

文字列に区切り文字以外の文字が含まれていない場合や、文字列内の単語数よりも絶対値が大きい数を指定した場合、CALL KSCANX ルーチンは位置と長さを 0 として返します。

単語を文字列として検索する

CALL KSCANX ルーチンを呼び出した後に指定した単語を文字列として検索するには、*string* 引数、*position* 引数および *length* 引数を指定した SUBSTRN 関数を使用します。

```
ksubstr(string, position, length);
```

null 引数の使用

CALL KSCANX ルーチンでは、文字引数に null を指定できます。null 引数は、長さが 0 の文字列として処理されます。数値引数は null にできません。

例

この例では、漢字を使用します。

```
data test;
  length word word_r $30;
  string='这个是字符函数的测试。';
  do until(position <= 0);
    count + 1;
    call kscanx(string, count, position, length, '是的。', 'oq');
    word=ksubstrn(string, position, length);
    if not position then leave;
    output;
  end;
run;
proc print data=test noobs;
  var count position length word;
run;
```

SAS は次の出力をログに書き込みます。

The SAS System

count	position	length	word
1	1	2	这个
2	4	4	字符函数
3	9	2	测试

BASECHAR 関数

文字を基本文字に変換します。

カテゴリ: CAS
文字

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

STR=BASECHAR(*<instr>*(,*<Unicode type>*))

必須引数

str
変換されたデータ文字列。

instr
入力データ文字列。

Unicode type
次のいずれかの Unicode 文字形式を指定した場合、各国語文字が指定した形式で表記されます。

- ESC Unicode エスケープ(\u00cd)。
- NCR 数値文字表現(Å)。
- PAREN Unicode かっこエスケープ(<u00c5>).

詳細

BASECHAR 関数は、文字を読み込み、基本文字に変換します。基本文字ならびに 1 つ以上のアクセントまたは結合文字から成る文字もあります。BASECHAR 関数は、文字列の文字を読み込み、Unicode 指定に基づいてアクセントなしの基本文字形式に変換します。Unicode タイプを指定した場合、各国語文字が指定した形式で表記されます。たとえば、データ文字列 Mühlenfließ は Muhlenfließに変換されます。

例

この例の SAS セッションエンコーディングは wlatin1 です。

注: BASECHAR 関数は、Unicode 文字分解マッピングを使用するので、シャープ s (ß)には Unicode 文字分解が含まれません。 http://www.unicode.org/reports/tr44/#Character_Decomposition_Mappings

```
data cities;
  length name$24 name_ascii$24;
  input name;
  name_ascii = basechar(name);
  put name_ascii=;
  datalines;
Mühlenfließ
LaUña
ZielonaGóra
;
```

これらのステートメントは、次の結果を生成します。

```
name_ascii=Muhlenfließ
name_ascii=LaUna
name_ascii=ZielonaGora
```

この例のセッションエンコーディングは wlatin1 で、ESC オプションを指定します。

```
data cities;
  length name$24 name_ascii$80;
  input name;
  name_ascii = basechar(name, "ESC");
  put name_ascii=;
  datalines;
Mühlenfließ
LaUña
ZielonaGóra
;
```

これらのステートメントは、次の結果を生成します。


```
name_ascii=Muhlenflie\u00df
name_ascii=LaUna
name_ascii=ZielonaGora
```

この例のセッションエンコーディングは `wlatin1` で、NCR オプションを指定します。

```
data cities;
  length name$24 name_ascii$80;
  input name;
  name_ascii = basechar(name, "NCR");
  put name_ascii=;
  datalines;
Muhlenfließ
LaUña
ZielonaGóra
;
```

これらのステートメントは、次の結果を生成します。

```
name_ascii=Muhlenfließ
name_ascii=LaUna
name_ascii=ZielonaGora
```

ENCODCOMPAT 関数

2 つのエンコーディング間のトランスコーディング互換性を確認します。

カテゴリ: エンコーディング

構文

ENCODCOMPAT(*source1*,<*source2*>)

必須引数

source1

エンコーディングを表す文字列です。

source2

エンコーディングを表す文字列です。この引数はオプションです。

詳細

エンコーディングを 1 つ指定した場合、この関数は現在の SAS セッションのエンコーディングとの互換性を確認します。

エンコーディングを 2 つ指定した場合、この関数は 2 つのエンコーディング間の互換性を確認します。

この関数は 2 つのエンコーディング識別子を比較し、データをトランスコードする必要があるかどうかを評価します。 *Source1* はソースのエンコーディングです。 *Source2* は出力先のエンコーディングです。7 ビット ASCII を他のタイプの ASCII にトランスコーディングする場合の互換性がありますが、ASCII を 7 ビット ASCII にトランスコーディングする場合は、互換性がないことがあります。

ENCODCOMPAT 関数は次の値を示します。

- 1 *Source1* は有効なエンコーディング名ではありません。
- 2 *Source2* は有効なエンコーディング名ではありません。
- 0 エンコーディング間に互換性はありません。トランスコーディングが必要です。
- 1 エンコーディング間の互換性があります。トランスコーディングは不要です。
- 2 改行文字が検出されました。

例

これらの例のセッションエンコーディングは UTF-8 です。

```
data encoding;
  isCompat= EncodCompat('xyz');
  put isCompat=;
run;
```

これらのステートメントは、次の結果を生成します。

```
isCompat=-1
```

```
data encoding;
  isCompat= EncodCompat ('ascii');
  put isCompat=;
run;
```

これらのステートメントは、次の結果を生成します。

```
isCompat=1
```

```
data encoding;
  isCompat= EncodCompat
    ("ebcdic1149","open_ed-1149");
  put isCompat;
run;
```

これらのステートメントは、次の結果を生成します。

```
isCompat=2
```

```
data encoding;  
isCompat= EncodCompat ('cp1251','ebcdic1149');  
put isCompat=;  
run;
```

これらのステートメントは、次の結果を生成します。

```
isCompat=0;
```

このコードは、WLATIN2 としても知られる Windows エンコーディング cp1251 を、open_ed-1149、Iceland OpenEdition EBCDIC と比較します。

```
data encoding;  
isCompat= EncodCompat ('ebcdic1149','open_ed-1149');  
put isCompat=;  
run;
```

これらのステートメントは、次の結果を生成します。

```
isCompat=2;
```

ENCODISVALID 関数

有効なエンコーディング名であるかどうかを検証します。

カテゴリ: エンコーディング

構文

ENCODISVALID(*source*)

必須引数

source

エンコーディング名を表す文字列です。

詳細

ENCODISVALID 関数は次の値を返します。

- 0 文字列は有効なエンコーディング名ではありません。
- 1 文字列は有効な短いエンコーディング名です。
- 2 文字列は有効な長いエンコーディング名です。
- 3 文字列は有効な別名エンコーディング名です。

例

```
data valid;
  isValid1=EncodIsValid('xyz');
  isValid2=EncodIsValid('wlt2');
  isValid3=EncodIsValid('wlatin2');
  isValid4=EncodIsValid('cp1250');
  put isValid1=;
  put isValid2=;
  put isValid3=;
  put isValid4=;
run;
```

これらのステートメントは、次の結果を生成します。

```
isValid1=0
isValid2=1
isValid3=2
isValid4=3
```

GETLOCENV 関数

現在のロケール/言語環境を返します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

GETLOCENV()

詳細

GETLOCENV 関数は、有効な SAS ロケールのロケール/言語環境値を返します。次の環境値が表示されます。

SBCS

SAS セッションエンコーディングは SBCS (1 バイト文字セット)です。文字列の操作のため、SASWZSD が読み込まれます。

DBCS

SAS セッションエンコーディングは DBCS (2 バイト文字セット)です。文字列の操作のため、SASWZSD が読み込まれます。

MBCS

SAS セッションエンコーディングは Unicode (UTF8)です。文字列の操作のため、SASWZSU が読み込まれます。

空白値が返された場合は、WZSS サブシステムが使用できません。このアクションは、構成またはインストールでエラーが発生したことを示します。

例

```
Data test;  
option locale=french_france;  
  environ=getlocenv();  
  put environ=;  
run;
```

これらのステートメントは、次の結果を生成します。

```
environ=MBCS
```

GETPXLANGUAGE 関数

現在の 2 文字の言語コードを返します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

GETPXLANGUAGE()

詳細

GETPXLANGUAGE 関数は、LOCALE=SAS システムオプションの現在の値に基づいて、2 文字の言語コードを返します。言語名の長さは 2 文字です。値を受け取る変数のサイズが 2 文字より小さい場合、値は切り捨てられます。

例

最初の例では、LOCALE=システムオプションが French_France に設定されています。2 番目の例では German に設定されています。3 番目の例では English_United States に設定されています。

ステートメント	結果
<pre>option locale=french_france; lang=getpxLanguage(); put lang;</pre>	fr
<pre>option locale=German; lang=getpxLanguage(); put lang;</pre>	de
<pre>option locale=en_US; lang=getpxLanguage(); put lang;</pre>	ja

関連項目

システムオプション:

- [“LOCALE システムオプション” \(973 ページ\)](#)

関数:

- [“GETPXREGION 関数” \(610 ページ\)](#)
- [“GETPXLOCALE 関数” \(608 ページ\)](#)

GETPXLOCALE 関数

SAS ロケールの POSIX ロケール値を返します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

GETPXLOCALE(*<source>*)

必須引数

<source>

ロケール名を指定する省略可能な引数です。

詳細

GETPXLOCALE 関数は、有効な SAS ロケール名に対する POSIX ロケール値を返します。無効なロケール名を指定すると、null 文字列が返されます。*<source>* 引数に値を指定しない場合、関数は現在の SAS セッションの POSIX 名を返します。POSIX ロケール名の長さは 5 文字です。値を受け取る変数のサイズが 5 文字より小さい場合、値は切り捨てられます。

例

最初の例では、LOCALE=システムオプションが French_France に設定されています。2 番目の例では、*<source>* 引数が German_Germany に設定されています。3 番目の例では、*<source>* 引数が English_United States に設定されています。

ステートメント	結果
option locale=french_france; locale=getpxLocale(); put locale;	fr_FR
locale=getpxLocale("german_germany"); put locale;	de_DE
locale=getpxLocale("english_unitedstates"); put locale;	en_US

関連項目

システムオプション:

- ["LOCALE システムオプション" \(973 ページ\)](#)

関数:

- [“GETPXLANGUAGE 関数” \(607 ページ\)](#)
- [“GETPXREGION 関数” \(610 ページ\)](#)

GETPXREGION 関数

現在の 2 文字の地域コードを返します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

GETPXREGION()

詳細

GETPXREGION 関数は、現在の LOCALE=SAS システムオプションに基づいて、2 文字の地域コードを返します。地域名の長さは 2 文字です。値を受け取る変数のサイズが 2 文字より小さい場合、値は切り捨てられます。

例

この例では、LOCALE システムオプションが French_France に設定されています。

```
data test;
option locale=french_france;
  region=getpxRegion();
put region;
run;
```

これらのステートメントは、次の結果を生成します。

FR

この例では、LOCALE システムオプションが German に設定されています。

```
data test;
option locale=german;
  region=getpxRegion();
put region;
```



```
run;
```

これらのステートメントは、次の結果を生成します。

```
DE
```

この例では、LOCALE システムオプションが English_United States に設定されています。

```
data test;
option locale=en_US;
      region=getpxRegion();
put region;
run;
```

これらのステートメントは、次の結果を生成します。

```
US
```

関連項目

システムオプション:

- [“LOCALE システムオプション” \(973 ページ\)](#)

関数:

- [“GETPXLOCALE 関数” \(608 ページ\)](#)
- [“GETPXLANGUAGE 関数” \(607 ページ\)](#)

KCHARLIST 関数

修飾子に従って文字リストを返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

注: 返される文字リストまたは文字列は、変数に指定されます。 str=KCHARLIST(*modifier*)

構文

KCHARLIST(*modifier*)

必須引数

modifier

文字定数、変数または式を指定します。空白でない文字はそれぞれ KCHARLIST 関数のアクションを変更します。次の文字を修飾子として使用します。

- A または a 英字を返します。
- C または c 制御文字を返します。
- D または d デジタル文字を返します。
- F または f SAS 名の最初の文字を返します。
- G または g グラフィック文字を返します。
- L または l 小文字の文字を返します。
- P または p 句読文字を返します。
- S または s すべてのスペース文字を返します。
- U または u 大文字の文字を返します。
- X または x 16 進文字を返します。

詳細

KCHARLIST は、他の K 関数で使えるエンコーディング固有の文字を返します。KCHARLIST を使用して区切り文字を決定すると、SAS 文字列関数の区切り文字のあいまいさを回避するのに役立ちます。

KCHARLIST は、DBCS スペースおよび句読文字リストを返すことができます。

例

例 1

この例では、KCHARLIST を使用して、修飾子を使用してアルファベット文字を返します。

```
data _null_;
  str=kcharlist('a');
  put str;
run;
```

上記のステートメントは、次の結果を生成します。

```
str=ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
```

例 2

この例は、区切り文字使用の機能を示しています。

```
data _null_;
  str = 'Circumference = 2 × π × radius';
  result1 = kscan(str, 5);          /* 1 */
  put result1=;
  result2 = kscan(str, 2, kcharlist('p')); /* 2 */
  put result2=;
run;
```

- 1 KSCAN は区切り文字である SBCS スペースを使用して 5 番目の単語を選択するため、 π が選択されます。
- 2 KCHARLIST は、句読点区切り文字 p が指定されているため、 $2 \times \pi \times$ 半径を選択します。=文字は文字列の最初の句読点区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result1=π
result2=2 × π × radius
```

例 3

この例では、KCHARLIST および KSCAN を使用して、DBCS スペースと SBCS スペースを持つストリングを処理します。

```
data _null_;
  string = "北京 赛仕 软件";          /* 1 */
  result1 = KSCAN(string, 2);          /* 2 */
  put result1=;                        /* 3 */
  result2 = KSCAN(string, 2, KCHARLIST('S')); /* 4 */
  put result2=;                        /* 5 */
run;
```

- 1 DBCS スペースと SBCS スペースを含む値を変数 *string* に割り当てます。入力文字列には 2 つのスペースが含まれています。最初のスペースは DBCS スペースであり、2 番目のスペースは SBCS スペースです。
- 2 KSCAN は 2 番目の単語を選択しますが、KSCAN は DBCS スペースを無視するため、「软件」が選択されます。
- 3 値「软件」が出力に書き込まれます。
- 4 KCHARLIST は DBCS スペースを処理し、スペースリストを区切り文字として KSCAN に渡します。正しい値「赛仕」が選択されています。
- 5 値「赛仕」が出力に書き込まれます。

```
result1=软件
result2=赛仕
```

関連項目

関数

- “SCAN 関数” (SAS 関数と CALL ルーチン: リファレンス)
- “KSCAN 関数” (670 ページ)

KCOMPARE 関数

文字式の比較結果を返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、“COMPARE 関数” (SAS 関数と CALL ルーチン: リファレンス)です。

構文

KCOMPARE(*source*,<*pos*, *count*,> *findstr*,<*modifiers*>)

必須引数

source

比較される文字式を指定します。

findstr

source と比較する文字式を指定します。

オプション引数

pos

比較を開始する *source* の開始位置を指定します。*pos* を省略した場合、*source* 全体が比較されます。*pos* が 0 よりも小さい場合、*source* は SO/SI 文字を含まない拡張 DBCS データとして判断されます。

modifiers

i または I *source* および *findstr* の大文字と小文字を区別しません。

l または L 値を比較する前に *source* および *findstr* の先頭の空白を削除します。

- n または N リテラル名の引数から引用符を削除し、*source* および *findstr* の大文字と小文字を区別しません。リテラル名は、引用符内の文字列として表される名前トークンで、大文字または小文字の n が後に続きます。リテラル名により、SAS データセットまたは変数名に使用できない特殊文字(空白文字を含む)を使用できるようになります。KCOMPARE で文字列がリテラル名として認識されるように、最初の文字を引用符にする必要があります。
- :(コロン) truncates the longer string in *source* または *findstr* の短い方の文字列の長さになるように長い方の文字列を切り捨てます。‘:’修飾子が指定されていない場合は、長い文字列の長さを使用します。短い文字列には、文字列の最後に空白が埋め込まれます。

count

比較するバイト数を指定します。*count* を省略した場合、末尾の空白を除き、*source* の後ろのすべての *pos* が比較されます。

詳細

COMPARE 関数では、修飾子の出現順序に意味があります。

- LN は各文字列から先頭の空白を削除した後、名前リテラルから引用符を削除します。
- NL は名前リテラルから引用符を削除した後、各文字列から先頭の空白を削除します。

KCOMPARE では、値を次のように返します。

- *source* が *findstr* より小さい場合は負の値
- *source* が *findstr* と等しい場合は 0
- *source* が *findstr* より大きい場合は正の値

例

次の例では、漢字を使用します。

```
options pageno=1 nodate ls=80 ps=60;
data test;
  rc1=kcompare('ABCDEF','ABC');
  rc2=kcompare('déclaré','declared');
  rc3=kcompare('déclaré',2,'declared');
  rc4=kcompare('漢字のテスト','漢字');
  rc5=kcompare('漢字のテスト',10,'テスト');
  rc6=kcompare('漢字のテスト',1,9,'漢字');
run;
```

```
proc print data=test; run;
```

上記のステートメントは、次の結果を生成します。

The SAS System						
Obs	rc1	rc2	rc3	rc4	rc5	rc6
1	4	2	1	3	0	3

KCOMPRESS 関数

指定された文字を文字式から削除します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、SAS 関数と CALL ルーチン: リファレンスの [COMPRESS](#) です。

構文

KCOMPRESS(*source*, <*characters-to-remove*>, <*modifier(s)*>)

必須引数

source

削除する文字を含む文字式を指定します。KCOMPRESS は、*source* のみが指定されると、すべての 1 バイトおよび 2 バイトの空白を削除してこの式を返します。

オプション引数

characters-to-remove

KCOMPRESS が文字式から削除する文字を指定します。characters-to-remove を省略すると、KCOMPRESS はすべての空白の式を削除します。

ヒント 文字のリテラル文字列を引用符で囲みます。

modifier

文字定数、変数または式を指定します。空白でない文字はそれぞれ KCOMPRESS 関数のアクションを変更します。全角アルファベットと数字がサポートされています。空白は無視されます。修飾子として使用できる文字は次のとおりです。

a または A 文字のリストにアルファベット文字を追加します。英文字と全角アルファベットおよび数字がサポートされています。

c または C 文字のリストに制御文字を追加します。

d または D	文字のリストに数字を追加します。英文字と全角アルファベットの文字および数字がサポートされています。
f または F	アンダースコア文字および英文字を文字リストに追加します。英文字のみがサポートされています。
g または G	文字のリストにグラフィカル文字を追加します。
h または H	文字のリストに水平タブを追加します。
i または I	保持または削除する文字と全幅文字の大文字と小文字を区別しません。
k または K	リスト内の文字を削除するのではなく、保持します。
l または L	小文字を文字リストに追加します。英文字と全角アルファベットの文字および数字がサポートされています。
n または N	数字、アンダースコア文字および英文字を文字リストに追加します。英文字のみがサポートされています。
o または O	DATA ステップ(WHERE 句を除く)または SQL プロシジャで O 修飾子を使用すると、第 2 引数および第 3 引数が変更されないループで KCOMPRESS を呼び出すときに、より迅速に実行できます。
p または P	文字のリストに句読点を追加します。
s または S	Unicode SAS セッション内のすべての Unicode スペース文字を削除します。
t または T	第 1 引数と第 2 引数から末尾の空白を取り除きます。英文字のみがサポートされています。
u または U	大文字を文字リストに追加します。英文字と全角アルファベットの文字および数字がサポートされています。
w または W	文字のリストに印刷可能文字が追加されます。
x または X	文字のリストに 16 進文字を追加します。

ヒント *modifier* が定数の場合、引用符で囲みます。一組の引用符で複数の定数を指定します。*modifier* を変数または式として表すこともできます。

例

次の例では、日本語文字を使用します。

```
data _null;
```

```

result1=kcompress('Renée','é');
result2=kcompress('Renée','é','a');
result3=kcompress('漢字のテスト','の');
put result1=/result2=/result3=;
run;

```

上記のステートメントは、次の結果を生成します。

```

result1=Rene
result2=!
result3=漢字テスト

```

関連項目

関数:

- [“KLEFT 関数” \(649 ページ\)](#)
- [“KTRIM 関数” \(689 ページ\)](#)

KCOUNT 関数

式の 2 バイト文字の数を返します。

カテゴリ: CAS
文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KCOUNT(*source*)

必須引数

source
数える文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

例

これらの例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。次の例では、絵文字を使用します。

```
data _null_;
  text=' abc👉'; /* 1 */
  result=kcount(text); /* 2 */
  put result=;
run;
```

- 1 `text` に文字と絵文字を割り当てます。
- 2 `KCOUNT` は DBCS/MBCS 文字をカウントし、その値を `result` に割り当てます。

上記のステートメントは、次の結果を生成します。

```
result=1
```

KCOUNTC 関数

文字列に含まれる個々の文字を数えます。

カテゴリ:

CAS

文字操作

制限事項:

この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。

構文

KCOUNTC(*string*,*character-list*,<*modifier(s)*>);

必須引数

string

文字定数、変数または式を指定します。この中にある文字を数えます。

ヒント 文字のリテラル文字列を引用符で囲みます。

character-list

文字リストを初期化する文字定数、変数または式を指定します。`KCOUNTC` 関数では、`modifier` 引数に `V` 修飾子を指定しない場合、このリストの文字が数えられます。`V` 修飾子を指定すると、このリストにないすべての文字が数えられます。他の修飾子を使うことでリストに文字をさらに追加できます。

ヒント 文字のリテラル文字列を引用符で囲みます。

修飾子の処理後にリストに文字がない場合、KCOUNTC 関数は 0 を返します。

オプション引数

modifer(s)

文字定数、変数または式を指定します。ここでは、空白以外の文字のそれぞれによって KCOUNTC 関数のアクションが変更されます。空白は無視されます。次の文字を大文字または小文字で修飾子として使用できます。

空白 無視されます。

a または A 文字のリストに英文字が追加されます。

b または B *string* が、左から右へではなく、右から左へスキャンされます。

c または C 文字のリストにコントロール文字が追加されます。

d または D 文字のリストに数字が追加されます。

f または F 文字のリストにアンダースコアと英文字(*VALIDVARNAME=V7* を使用した SAS 変数名内の有効な最初の文字)を追加します。

g または G 文字のリストにグラフィック文字が追加されます。グラフィカル文字は、紙面に印刷するとイメージになる文字です。

h または H 文字のリストに水平タブが追加されます。

i または I 大文字と小文字を区別しません。

l または L 文字のリストに小文字が追加されます。

n または N 数字、アンダースコアおよび英文字(すなわち、*VALIDVARNAME=V7* を使用する SAS 変数名に記述できる文字)が文字のリストに追加されます。

o または O KCOUNTC 関数のこのインスタンスへの最初の呼び出しで、*character-list* および *modifier* 引数が 1 回だけ処理されます。後続の呼び出しで *character-list* または *modifier* の値を変更すると、KCOUNTC 関数によって変更が無視されることがあります。

p または P 文字のリストに句読点が追加されます。

s または S 文字のリストに空白文字が追加されます(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)。

t または T *string* 引数と *character-list* 引数から末尾の空白を取り除きます。両方の文字引数ではなく一方のみから末尾の空白を削除す

る場合は、KCOUNTC ルーチンで T 修飾子を使用するかわりに、TRIM 関数を使用します。

u または U 文字のリストに大文字が追加されます。

v または V 文字のリストに出現しない文字を数えます。この修飾子を指定しない場合、KCOUNTC 関数は文字のリストに出現する文字を数えます。

w または W 文字のリストに印刷可能(書き込み可能)文字が追加されます。

x または X 文字のリストに 16 進数の文字が追加されます。

ヒント *modifier* が定数の場合、引用符で囲みます。一組の引用符で複数の定数を指定します。

詳細

基本

KCOUNTC 関数では、文字引数に NULL を指定できます。NULL 引数は、長さがゼロの文字列として処理されます。数える対象となる文字リストの文字がない場合、KCOUNTC 関数はゼロを返します。

比較

KCOUNTC 関数は文字列内の個々の文字を数えますが、KCOUNTX 関数は文字列内の文字の部分文字列を数えます。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では絵文字を使用しています:

```
data _null_;
text=' abc👉';
result=kcountc(text,'a👉'); /* 1 */
put result =;
```

```
run;
```

- 1 KCOUNTC は、*a* と *b* の出現回数をカウントします。

```
result=2
```

例 2

この例では、漢字を使用します。

```
data _null_ ;
text='SAS 是全球最大的软件公司之一,是全球商业智能和分析软件与服务领袖。';
result=kcountc(text,'软件'); /* 1 */
put result = ;
run;
```

- 1 KCOUNTC は、「软」と「件」の出現回数をカウントします。

上記のステートメントは、次の結果を生成します。

```
result=4
```

KCOUNTW 関数

文字列内の単語の数を数えます。

カテゴリ: CAS
文字操作

制限事項: この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 *string* が 1 バイト文字の場合、KCOUNTW 関数は SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

構文

```
KCOUNTW(<string>,<character-list>,<modifier(s)>);
```

オプション引数

string

文字定数、変数または式を指定します。この中にある単語を数えます。

character-list

文字のリストを初期化する任意の文字定数、変数または式を指定します。このリストの文字は、単語の区切り文字です。ただし、*modifier* 引数に K 修飾子を指定

すると、すべての文字(リストの文字を含む)が区切り文字として扱われます。リストにさらに文字を追加するには、他の修飾子を使用します。単語を区切るために指定できるいくつかの文字はいずれも **区切り文字** と呼ばれます。

ヒント 文字引数には NULL を指定できます。NULL 引数は、長さがゼロの文字列として処理されます。数値引数には NULL を指定できません。

modifier

文字定数、変数または式を指定します。ここでは、空白以外の文字のそれぞれによって KCOUNTW 関数のアクションが変更されます。次の文字を修飾子として使用できます。

空白	無視されます。
a または A	文字のリストに英文字が追加されます。
b または B	左から右へではなく、右から左へ数えていきます。右から左方向へ数えて変化があるのは、Q 修飾子を使っている場合で、かつ一組になっていない引用符が文字列中にある場合のみです。
c または C	文字のリストにコントロール文字が追加されます。
d または D	文字のリストに数字が追加されます。
f または F	文字のリストにアンダースコアと英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内の有効な最初の文字) を追加します。
g または G	文字のリストにグラフィック文字が追加されます。グラフィカル文字は、紙面に印刷するとイメージになる文字です。
h または H	文字のリストに水平タブが追加されます。
i または I	大文字と小文字の区別が無視されます。
k または K	リストにないすべての文字が区切り文字として扱われます。K を指定しなければ、リストにあるすべての文字が区切り文字として扱われます。
l または L	文字のリストに小文字が追加されます。
m または M	複数の連続した区切り文字および <i>string</i> 引数の先頭または末尾にある区切り文字が、長さゼロの単語を参照するように指定されます。M 修飾子を指定しない場合、複数の連続する区切り文字は 1 つの区切り文字として扱われ、 <i>string</i> 引数の先頭または末尾の区切り文字は無視されます。
n または N	文字のリストに数字、アンダースコアおよび英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内に表示可能な文字) を追加します。
o または O	<i>character-list</i> および <i>modifier</i> 引数が、KCOUNTW 関数を呼び出すたびにではなく、1 回だけ処理されます。DATA ステップ(WHERE

句を除く)または SQL プロシジャで O 修飾子を使用すると、*chars* および *modifier* 引数に変更されないため、KCOUNTW 関数をループで呼び出す際により速く実行させられます。

p または P	文字のリストに句読点が追加されます。
q または Q	引用符で囲まれた部分文字列内にある区切り文字が無視されます。 <i>string</i> の値に一致しない引用符が含まれている場合は、左から右へのスキャンによって、右から左へのスキャンとは異なる単語が生成されます。
s または S	文字のリストに空白文字(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)を追加します。
t または T	<i>string</i> および <i>character-list</i> 引数から末尾の空白が削除されます。
u または U	文字のリストに大文字が追加されます。
w または W	文字のリストに印刷可能(書き込み可能)文字が追加されます。
x または X	文字のリストに 16 進数の文字が追加されます。

詳細

単語の定義

KCOUNTW 関数では、次のいずれかの条件を満たす部分文字列を単語とします。

- 左境界が、区切り文字または文字列の先頭である。
- 右境界が、区切り文字または文字列の末尾である。
- Q 修飾子を使用し、かつ区切り文字が引用符を含む部分文字列内にある場合を除き、区切り文字が含まれていない

ASCII 環境での KCOUNTW 関数の使用

引数を 2 つだけ指定して KCOUNTW 関数を使用し、コンピュータで ASCII 文字を使用している場合、次のデフォルトの区切り記号が使用されます。

空白 ! \$ % & () * + , - . / ; < ^ _ { | ; > ^ _ { |

^文字が含まれない ASCII 環境では、SCAN 関数はかわりに~文字を使用します。

M 修飾子の使用

M 修飾子を使用しない場合、単語には少なくとも 1 文字が含まれている必要があります。M 修飾子を使用する場合、単語の長さをゼロとすることができます。後者の場合、単語数は文字列中の区切り文字数に 1 を加算した数です。Q 修飾子を使用するときに引用符に囲まれている文字列中の区切り文字の数ではありません。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では絵文字を使用します。

```
data _null_;
  text='Renée Dupont?';
  result1=kcountw(text); /* 1 */
  result2=kcountw(text, 'p'); /* 2 */
  put result1= /result2=;
run;
```

- 1 KCOUNTW は *text* 内の単語数をカウントします。SBCS スペースは区切り文字です。
- 2 KCOUNTW は *text* 内の単語数をカウントします。SBCS のスペースと ? は区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result1=2
result2=3
```

例 2

これらの例では、UTF-8 エンコーディングを使用します。

この例では、漢字を使用します。

```
data _null_;
  text='SAS 是全球最大的软件公司之一,是全球商业智能和分析软件与服务领袖。';
  result=kcountw(text, ' ', 'p'); /* 1 */
  put result =;
run;
```

- 1 KCOUNTW は *text* 内の単語数をカウントします。句読点は区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=2
```

KCOUNTX 関数

指定した部分文字列が文字列内に含まれる回数を数えます。

カテゴリ: CAS

文字操作

制限事項: この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 *string* が 1 バイト文字の場合、KCOUNTX 関数は SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

構文

KCOUNTX(*string*,*substring*,<*modifier(s)*>);

必須引数

string

部分文字列を数える文字定数、変数または式を指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

substring

string の中で数える対象となる部分文字列を文字定数、変数または式で指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

オプション引数

modifier

1 つ以上の修飾子を文字定数、変数または式で指定します。次の修飾子 (*modifiers*)には大文字と小文字のどちらでも使用できます。

- i 数える際に大文字と小文字の区別が無視されます。この修飾子が指定されていない場合、KCOUNTX 関数は、大文字と小文字が *substring* の文字と同じ部分文字列のみを数えます。
- t *string* および *substring* から末尾の空白が削除されます。

ヒント *modifier* が定数の場合、引用符で囲みます。一組の引用符で複数の定数を指定します。*modifier* は、変数や式として表現することもできます。

詳細

基本

KCOUNTX 関数は、*string* を左から右へ検索し、指定された *substring* の出現回数を取得して、その出現回数を返します。*string* で部分文字列が見つからない場合、KCOUNTX 関数は 0 の値を返します。

注意

文字列で指定された部分文字列の 2 つの出現個所が重複している場合、その結果は未定義です。たとえば、`kcountx('boobooboo', 'booboo')` は、1 または 2 のいずれかを返す可能性があります。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

この例では、アクセント記号付き文字を使用します。

```
data _null_;
  text='Renée Dupont';
  result=kcountx(text,'ée'); /* 1 */
  put result=;
run;
```

1 KCOUNTX はアクセント記号付き文字 `ée` の出現回数をカウントします。

上記のステートメントは、次の結果を生成します。

```
result=1
```

例 2

この例では、漢字を使用します。

```
data _null_;
  text='SAS 是全球最大的软件公司之一,是全球商业智能和分析软件与服务领袖。';
```

```
result=kcountx(text,'软件'); /* 1 */
put result = ;
run;
```

1 KCOUNTX は、「软件」の出現回数をカウントします。

上記のステートメントは、次の結果を生成します。

```
result=2
```

KCVT 関数

データのある種類のエンコーディングデータから別の種類のエンコーディングデータに変換します。

カテゴリ: CAS
文字

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KCVT(*text*, *intype*, *outtype*, <*options*>)

必須引数

text
変換する文字変数を指定します。

intype
データのエンコーディングを指定します。テキストのエンコーディングは、入力データのエンコーディングと一致する必要があります。有効な値については、[“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” \(1118 ページ\)](#)を参照してください。

ASCIIANY および EBCDICANY は、無効なエンコーディング値です。

outtype
文字データに変換するエンコーディングを指定します。有効な値については、[“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” \(1118 ページ\)](#)を参照してください。

ASCIIANY および EBCDICANY は、無効なエンコーディング値です。

options
文字データ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。

NOSOSI | シフトコードと半角文字は含めません。
NOSHIFT

INPLACE	文字データを変換によって置き換えます。INPLACE オプションは、異なるホスト間で文字データの長さが異なる場合に、同じ位置を確保するために指定します。たとえば、INPLACE オプションは、データをシフトコードが必要なホストから、シフトコードが不要な別のホストに変換します。シフトコードに関して <i>outtype</i> に変換される文字データの長さが、 <i>intype</i> に指定された長さよりも長い場合には、切り捨てが行われます。
KANA	文字データの列に、半角カタカナ文字を含めます。
UPCASE	2 バイトの英字を大文字に変換します。
LOWCASE	2 バイトの英字を小文字に変換します。
KATA2HIRA	カタカナデータをひらがなに変換します。
HIRA2KATA	ひらがなデータをカタカナに変換します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ) を参照してください。

KCVT 関数は、SBCS、DBCS および MBCS 文字列をエンコーディングデータに変換します。たとえば、KCVT 関数では、ASCII コードデータから UCS2 エンコーディングデータ、ギリシャ語コードデータから UTF-8、日本語 SJIS コードデータから別の日本語コードデータの変換が可能です。指定可能な *intype* および *outtype* オプションの種類は、UCS2、UCS2L、UCS2B、および UTF8 です。DBCS モードを有効にするには、構成ファイル内またはコマンドラインで、次の DBCS オプションを指定します。

KCVT 関数が事前に長さを割り当てられていない変数に値を返す場合、デフォルトで変数の長さは 200 に割り当てられます。

SQL プロシジャまたはプロシジャの WHERE 句では、KCVT 関数によって返される単語の最大長は 200 文字になります。

例

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  src = '436972636F6E66E9726556E636520'; /* 1 */
  text = kcvf(src, 'latin1', 'utf-8'); /* 2 */
  put text=;
run;
```

- 1 Circonférence の latin1 値を変数 src に割り当てます。
- 2 Circonférence の latin1 値を UTF-8 エンコーディングに変換します。

上記のステートメントは、次の結果を生成します。

```
text=Circonférence
```

関連項目

システムオプション:

- [“DBCS システムオプション” \(962 ページ\)](#)

プロシジャ:

- [18 章, “DBCSTAB プロシジャ,” \(1043 ページ\)](#)

KFIND 関数

文字列内の特定の部分文字列を検索します。

カテゴリ:

CAS

文字操作

制限事項:

この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 *string* がマルチバイト文字の場合、KFIND 関数はマルチバイトデータを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

構文

KFIND(*string*,*substring*,<*start-position*>,<*modifier(s)*>)

KFIND(*string*,*substring*,<*modifier(s)*>,<*start-position*>)

必須引数

string

検索する部分文字列に関する文字定数、変数または式を指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

substring

string で検索する部分文字列を指定する文字定数、変数または式です。

ヒント 文字のリテラル文字列を引用符で囲みます。

オプション引数

modifier(s)

1 つ以上の修飾子を文字定数、変数または式で指定します。有効な修飾子は次のとおりです。

i or I

検索時に大文字と小文字を区別しません。この修飾子を指定しないと、KFIND は *substring* の文字の大文字と小文字に一致する部分文字列のみを検索します。

t or T

string および *substring* から末尾の空白を取り除きます。

注: 両方(またはすべて)の文字引数ではなく 1 つの文字引数のみから末尾の空白を削除する場合は、T 修飾子付きの KFIND 関数のかわりに KTRIM 関数を使用します。

ヒント *modifier* が定数の場合は、引用符で囲みます。1 組の引用符に複数の定数を指定します。*modifier* は、変数や式として表現することもできます。

start-position

検索を開始する位置を指定し、検索の方向を指定する整数値を持つ数値定数、変数または式です。

詳細

基本

KFIND 関数は、*string* で最初に出現する指定 *substring* を検索し、その部分文字列の位置を返します。*string* で部分文字列が見つからない場合、KFIND 関数は 0 の値を返します。

start-position が指定されていない場合、KFIND 関数は、*string* の先頭から検索を開始し、*string* を左から右へ検索していきます。*start-position* が指定されている場合は、*start-position* の絶対値によって、検索を開始する位置が決定されます。検索方向は、*start-position* の符号で決まります。

startpos の値 アクション

- 0 より大きい 位置 *start-position* から検索を開始し、右向きに検索していきます。
start-position が *string* の長さより大きい場合、KFIND は 0 の値を返します。
- 0 より小さい 位置 *start-position* から検索を開始し、左向きに検索していきます。
start-position が *string* の長さより大きい場合、*string* の末尾から検索が開始されます。

startpos の値 アクション

0 に等しい 0 の値を返します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  pos_of=kfind(text,'Dupont'); /* 1 */
  put pos_of=;
run;
```

1 K FIND は、Dupont の最初の文字の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=7
```

例 2

この例では、漢字を使用します。

```
data _null_;
  text='SAS 是全球最大的软件公司之一,是全球商业智能和分析软件与服务领袖。';
  pos_of=kfind(text,'软件'); /* 1 */
  put pos_of=;
run;
```

1 K FIND は、「软件」の最初の文字の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=10
```

KFINDC 関数

文字のリストにある各文字を文字列から検索します。

カテゴリ: CAS

文字操作

制限事項: この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 *string* が 1 バイト文字の場合、KFINDC 関数は SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

注: この関数は varchar 型をサポートしていません。

構文

KFINDC(*string*,<*character-list*>)

KFINDC(*string*,*character-list*,<*modifier(s)*>)

KFINDC(*string*,*character-list*,*modifier(s)*,<*start-position*>)

KFINDC(*string*,*character-list*,<*start-position*>,<*modifier(s)*>)

必須引数

string

検索する文字列を指定する文字定数、変数または式です。

ヒント 文字のリテラル文字列を引用符で囲みます。

character-list

文字リストを初期化する定数、変数または文字式です。KFINDC 関数は、*modifier* 引数に K 修飾子を指定しなければ、このリストの文字を検索します。K 修飾子を指定すると、KFINDC 関数は、この文字リストにないすべての文字を検索します。他の修飾子を使用すると、さらに多くの文字をリストに追加できます。

modifier(s)

各文字が KFINDC 関数のアクションを変更する文字定数、変数または式です。次の文字を大文字または小文字で修飾子として使用できます。

空白 無視されます。

a または A 文字のリストに英文字が追加されます。

b または B *string* が、左から右へではなく、右から左へスキャンされます。

c または C 文字のリストにコントロール文字が追加されます。

d または D 文字のリストに数字が追加されます。

f または F 文字のリストにアンダースコアと英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内の有効な最初の文字)を追加します。

g または G	文字のリストにグラフィック文字が追加されます。グラフィカル文字は、紙面に印刷するとイメージになる文字です。
h または H	文字のリストに水平タブが追加されます。
i または I	大文字と小文字を区別しません。
k または K	文字のリストに出現しない文字をすべて検索します。この修飾子を指定しない場合、KFINDC 関数は、文字のリストに出現する文字をすべて検索します。この修飾子は、v または V 修飾子と同じ機能を持ちます。
l または L	文字のリストに小文字が追加されます。
n または N	数字、アンダースコアおよび英文字(すなわち、VALIDVARNAME=V7 を使用する SAS 変数名に記述できる文字)が文字のリストに追加されます。
o または O	<i>character-list</i> および <i>modifier</i> 引数が、KFINDC 関数を呼び出すたびにではなく、1 回だけ処理されます。DATA ステップ(WHERE 句を除く)または SQL プロシジャで O 修飾子を使用すると、 <i>character-list</i> および <i>modifier</i> 引数に変更されないため、KFINDC 関数をループで呼び出す際により速く実行させられます。
p または P	文字のリストに句読点が追加されます。
s または S	文字のリストに空白文字が追加されます(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)。
t または T	string 引数と <i>character-list</i> 引数から末尾の空白を取り除きます。両方の文字引数ではなく一方のみから末尾の空白を削除する場合は、KFINDC ルーチンで T 修飾子を使用するかわりに、TRIM 関数を使用します。
u または U	文字のリストに大文字が追加されます。
v または V	文字のリストに出現しない文字をすべて検索します。この修飾子を指定しない場合、KFINDC は、文字のリストに出現する文字をすべて検索します。この修飾子は、k または K 修飾子と同じ機能を持ちます。
w または W	文字のリストに印刷可能(書き込み可能)文字が追加されます。
x または X	文字のリストに 16 進数の文字が追加されます。

ヒント *modifier* が定数の場合、引用符で囲みます。一組の引用符で複数の定数を指定します。*modifier* は、変数や式として表現することもできます。

オプション引数

start-position

検索の開始位置と検索方向を指定する整数値を使用する任意の数値定数、変数または式です。

詳細

基本

KFINDC 関数は、string で最初に出現する指定文字を検索し、最初に見つかった文字の位置を返します。string で文字が見つからない場合、KFINDC 関数は 0 の値を返します。

KFINDC 関数では、文字引数に NULL を指定できます。NULL 引数は、長さがゼロの文字列として処理されます。数値引数には NULL を指定できません。

start-position が指定されていない場合、KFINDC 関数は、B 修飾子を使用するときは文字列の末尾から、B 修飾子を使用しないときは文字列の先頭から、検索を開始します。

start-position を指定すると、*start-position* の絶対値で検索の開始位置が決定されます。B 修飾子を使用する場合、検索は常に右から左に進みます。B 修飾子を使用しない場合、*start-position* の符号で検索方向を指定します。次の表に、検索の方向をまとめます。

<i>startpos</i> の値	アクション
0 より大きい	位置 <i>start-position</i> から検索を開始し、右向きに検索していきます。 <i>start-position</i> が文字列の長さより大きい場合、KFINDC 関数は 0 の値を返します。
0 より小さい	位置 <i>start-position</i> から検索を開始し、左向きに検索していきます。 <i>start-position</i> が文字列の負の長さよりも小さい場合、文字列の末尾から検索を開始します。
0 に等しい	0 の値を返します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null;  
text='Renée Dupont';
```

```
pos_of=kfindc(text,'éut'); /* 1 */
put pos_of=;
run;
```

- 1 文字列 éut 内で見つかった最初の文字の位置を指定します。

```
pos_of=4
```

例 2

```
data _null_;
text='Renée, Dupont';
pos_of=kfindc(text,'ut','p'); /* 1 */
put pos_of=;
run;
```

- 1 文字列 ut または句読点の最初の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=6
```

例 3

この例では、漢字を使用します。

```
data _null_;
text='SAS 是全球最大的软件公司之一,是全球商业智能和分析软件与服务领袖。';
pos_of=kfindc(text,'软件的'); /* 1 */
put pos_of=;
run;
```

- 1 文字列软件的の最初の文字の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=9
```

KFINDW 関数

文字列内の単語の文字位置か、文字列内の単語の番号を返します。

カテゴリ: CAS
文字操作

制限事項:

この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 *string* が 1 バイト文字の場合、KFINDW 関数は SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

構文

KFINDW(*string*,*word*,<*character-list*>)

KFINDW(*string*,*word*,*character-list*,*modifier(s)*,<*start-position*>)

KFINDW(*string*,*word*,*character-list*,*start-position*,<*modifier(s)*>)

KFINDW(*string*,*word*,*start-position*,<*character-list*,*modifier(s)*>)

必須引数

string

検索する文字列を指定する文字定数、変数または式です。

word

string で検索する単語を指定する文字定数、変数または式です。

character-list

文字のリストを初期化する文字定数、変数または式です(省略可能)。

リスト中の文字は単語を区切るための区切り文字です(ただし *modifier* 引数に K 修飾子を指定しない場合)。K 修飾子を指定すると、このリストにないすべての文字が区切り文字となります。他の修飾子を使用して、このリストに文字をさらに追加できます。

start-position

検索を開始する位置を指定し、検索の方向を指定する整数値を持つ数値定数、変数または式です(省略可能)。

modifier

文字定数、変数または式を指定します。ここでは、空白以外の文字のそれぞれによって KFINDW 関数のアクションが変更されます。

ヒント *modifier* 引数を使用する場合は、*character-list* 引数の後に置く必要があります。

修飾子として次の文字を使用します。

空白 無視されます。

a または A 文字のリストに英文字が追加されます。

b または B *string* が、左から右へではなく、右から左へスキャンされます。

c または C 文字のリストにコントロール文字が追加されます。

d または D	文字のリストに数字が追加されます。
e または E	文字列内の指定された単語の文字位置を決定するかわりに、指定された単語が見つかるまでスキャンされる単語を数えます。単語の断片は数えられません。
f または F	文字のリストにアンダースコアと英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内の有効な最初の文字) を追加します。
g または G	文字のリストにグラフィック文字が追加されます。グラフィカル文字は、紙面に印刷するとイメージになる文字です。
h または H	文字のリストに水平タブが追加されます。
i または I	大文字と小文字を区別しません。
k または K	文字のリストにないすべての文字が区切り文字として扱われます。K を指定しなければ、文字のリストにあるすべての文字が区切り文字として扱われます。
l または L	文字のリストに小文字が追加されます。
n または N	文字のリストに数字、アンダースコアおよび英文字 (VALIDVARNAME=V7 を使用した SAS 変数名内に表示可能な文字) を追加します。
o または O	<i>character-list</i> および <i>modifier</i> 引数が、KFINDW 関数を呼び出すたびにではなく、1 回だけ処理されます。DATA ステップ (WHERE 句を除く) または SQL プロシジャで O 修飾子を使用すると、 <i>character-list</i> および <i>modifier</i> 引数が変更されないため、KFINDW 関数をループで呼び出す際により速く実行させられます。
p または P	文字のリストに句読点が追加されます。
q または Q	引用符で囲まれた部分文字列内にある区切り文字が無視されます。string 引数の値に一致しない引用符が含まれている場合は、左から右へのスキャンによって、右から左へのスキャンとは異なる単語が生成されます。
r または R	word 引数から先頭および末尾の区切り文字を削除します。
s または S	文字のリストに空白文字 (空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード) を追加します。
t または T	<i>string</i> 、 <i>word</i> および <i>character-list</i> 引数から末尾の空白が削除されます。
u または U	文字のリストに大文字が追加されます。
w または W	文字のリストに印刷可能 (書き込み可能) 文字が追加されます。
x または X	文字のリストに 16 進数の文字が追加されます。

詳細

区切り文字

単語を区切るために使用されるいくつかの文字はいずれも *区切り文字* と呼ばれます。区切り文字は、*character* 引数または *modifier* 引数、あるいはその両方を使用して指定できます。Q 修飾子を指定すると、引用符で囲まれた部分文字列内の文字は、区切り文字として扱われません。

単語

次の条件を両方とも満たす部分文字列を *単語* とします。

- 左境界が、区切り文字または文字列の先頭である
- 右境界が、区切り文字または文字列の末尾である。

注: 単語に区切り文字を含めることができます。この場合、KFINDW 関数は、単語は区切り文字を含まないと定義されている SCAN 関数とは異なります。

文字列の検索

指定された単語と一致し、かつ単語の定義を満たす部分文字列を見つけられなかった場合、KFINDW 関数は 0 の値を返します。

KFINDW 関数が、指定された単語と一致し、かつ単語の定義を満たす部分文字列を見つけた場合、KFINDW 関数によって返される値は、E 修飾子が指定されているかどうかによって異なります。

- E 修飾子を指定すると、KFINDW 関数は、指定された単語を検索している間にスキャンされた完全な単語の数を返します。*start-position* で、単語の途中の位置が指定された場合、その単語は数に入りません。
- E 修飾子を指定しなければ、KFINDW 関数は、見つかった部分文字列の文字位置を返します。

start-position 引数を指定すると、*start-position* の絶対値によって、検索を開始する位置が指定されます。*start-position* の符号によって、検索する方向が指定されます。

<i>startpos</i> の値	アクション
0 より大きい	<i>start-position</i> から検索を開始し、右向きに検索していきます。 <i>start-position</i> が文字列の長さより大きい場合、KFINDW 関数は 0 の値を返します。
0 より小さい	位置 <i>start-position</i> から検索を開始し、左向きに検索していきます。 <i>start-position</i> が文字列の負の長さよりも小さい場合、文字列の末尾から検索を開始します。
0 に等しい	KFINDW 関数は 0 の値を返します。

start-position 引数や B 修飾子を指定しない場合、KFINDW 関数は、文字列の先頭から開始して左から右へ検索します。B 修飾子を指定しても、*start-position* 引数を使用しない場合、KFINDW 関数は、文字列の末尾から開始して右から左へ検索します。

ASCII 環境での KFINDW 関数の使用

引数を 2 つだけ指定して KFINDW 関数を使用し、コンピューターで ASCII 文字を使用している場合、次のデフォルトの区切り記号が使用されます。

空白!\$%&()*+,-./;<^!/_;<^|

^文字が含まれない ASCII 環境では、KFINDW 関数はかわりに~文字を使用します。

null 引数の使用

KFINDW 関数では、文字引数を null に指定できます。null 引数は長さがゼロの文字列として扱われます。数値引数は null にできません。

SBCS および DBCS データの処理

KFINDW 関数は SBCS データを処理するように設計されていますが、特定の条件で DBCS データを処理できます。SBCS および DBCS 処理の基準は次のとおりです。

- *string* が *varchar* として宣言され、かつマルチバイトデータを処理している場合、KFINDW 関数は DBCS を処理します。
- *string* が *varchar* として宣言されていないか、または 1 バイトデータを処理している場合、KFINDW 関数は SBCS を処理します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  pos_of=kfindw(text,'Dupont'); /* 1 */
  put pos_of=;
run;
```

- 1 KFINDW は、SBCS スペースを区切り文字として使用して、テキスト Dupont の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=7
```

例 2

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では絵文字を使用します。

```
data _null_;
  text='Renée Dupont?';
  pos_of=kfindw(text,'□','p'); /* 1 */
  put pos_of=;
run;
```

- 1 KFWINDW は、SBCS スペースと句読点を区切り文字として使用して絵文字の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=14
```

例 3

この例では、漢字を使用します。

```
data _null_;
  text='SAS 是全球最大的软件公司之一,sas 是全球商业智能和分析软件与服务领袖。';
  pos_of=kfindw(text,'sas','sp'); /* 1 */
  put pos_of=;
run;
```

- 1 KFWINDW は、スペースと句読点を区切り文字として使用して、sas の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
pos_of=17
```

KINDEX 関数

文字式から文字列を検索します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、SAS 関数と CALL ルーチン: リファレンスの [INDEX](#) を参照してください。

構文

KINDEX(*source*, *excerpt*)

必須引数

source

検索する文字式を指定します。

excerpt

文字式で検索する文字列を指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KINDEX 関数は、*source* 内を左から右へ、*excerpt* で指定された文字列が最初に現れる個所を検索し、*source* 内でのその文字列の先頭文字の位置を返します。*source* 内に文字列が見つからない場合、KINDEX は値 0 を返します。文字列が複数回現れる場合、KINDEX は、最初の出現個所の位置のみを返します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;  
  text='Renée Dupont';  
  result1=kindex(text,'ée'); /* 1 */  
  result2=kindex(text,'éec'); /* 2 */  
  put result1=/result2=;  
run;
```

1 文字「ée」の位置を指定します。

2 文字「éec」の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=5  
result2=0
```


例 2

この例では、漢字を使用します。

```
data _null_;  
  text='漢字文字列の検索';  
  result1=index(text,'検索'); /* 1 */  
  text2='探す';  
  result2=index(text,text2); /* 2 */  
  put result1=/result2=;  
run;
```

1 KINDEX は文字「検索」の位置を指定します。

2 KINDEX は文字「探す」の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=7  
result2=0
```

関連項目

関数:

■ [“KINDEXC 関数” \(647 ページ\)](#)

KINDEXB 関数

文字式で指定された文字を検索し、バイトベースの値を返します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KINDEXB(*source*, *excerpt*)

必須引数

source

検索する文字式を指定します。

excerpt

文字式で検索する文字列を指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

詳細

KINDEXB 関数は、*source* を左から右へ *excerpt* に指定された文字列の最初の出現個所を検索し、その文字列の先頭文字の *source* 内での位置を返します。*source* 内に文字列が見つからない場合、KINDEXB は値 0 を返します。文字列が複数回現れる場合、KINDEXB は、最初の出現個所の位置のみを返します。

比較

KINDEXB は、バイトベースの値を返します。KINDEX は、文字ベースの値を返します。SBCS 文字列、(たとえば wlatin1 エンコーディング)を処理する場合、KINDEXB と KINDEX は同一の結果を返します。一方、DBCS セッション(たとえば、EUC-JP または SHIFT-JIS エンコーディング)では、ほとんどの CJK 文字は 2 バイト幅です。このため、これらの関数は異なる結果を返します。

KINDEXCB 関数は、文字列中に存在するあらゆる文字の、最初の出現個所を検索します。KINDEXB 関数は、文字列の、1 つのパターンとしての最初の出現個所を検索します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  result1=kindexb(text,'ée'); /* 1 */
  result2=kindexb(text,'éec'); /* 2 */
  put result1=result2=;
run;
```

- 1 KINDEXB は、「ée」のバイト位置を指定します。
- 2 KINDEXB は、「éec」のバイト位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=8
result2=0
```

例 2

この例では、漢字を使用します。

```
data _null_;
text='漢字文字列の検索';
result1=kindexb(text,'検索'); /* 1 */
text2='探す';
result2=kindexb(text,text2); /* 2 */
put result1=/result2=;
run;
```

- 1 KINDEXB は「検索」のバイト位置を指定します。
- 2 KINDEXB は、「探す」のバイト位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=19
result2=0
```

KINDEXCB 関数

文字式で指定された文字を検索し、バイトベースの値を返します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KINDEXCB(*source*, *excerpt-1*, ..., *excerpt-n*)

必須引数

source

検索する文字式を指定します。

excerpt-1, ..., *excerpt-n*

文字式で検索する文字を指定します。

ヒント 複数の *excerpt* を指定する場合は、カンマで区切ります。

文字のリテラル文字列を引用符で囲みます。

詳細

TKINDEXCB 関数は、*source* 内のデータを左から右へ、*excerpt* の引数内のあらゆる文字の、最初の出現個所を検索し、その文字の *source* 内での位置を返します。*excerpt-1* から *excerpt-n* までの文字がいずれも *source* 内で見つからなかった場合、KINDEXC は値 0 を返します。

比較

KINDEXCB は、バイトベースの値を返します。KINDEXC は、文字ベースの値を返します。SBCS 文字列、(たとえば wlatin1 エンコーディング)を処理する場合、両方の関数は同一の結果を返します。一方、DBCS セッション(たとえば、EUC-JP または SHIFT-JIS エンコーディング)では、ほとんどの CJK 文字は 2 バイト幅です。このため、これらの関数は異なる結果を返します。

KINDEXCB 関数は、文字列中に存在するあらゆる文字の、最初の出現個所を検索するのに対し、KINDEXB 関数は、文字列の、1 つのパターンとしての最初の出現個所を検索します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  result1=kindexcb(text,'ée'); /* 1 */
  result2=kindexcb(text,'éc'); /* 2 */
  put result1=result2=;
run;
```

- 1 KINDEXCB は、文字「é」と「e」の最初のバイト位置を指定します。
- 2 KINDEXCB は、文字「é」、「e」、「c」の最初のバイト位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=6
result2=6
```

例 2

この例では、漢字を使用します。

```
data _null_;
  text= '感.def(漢字 123456)';
  result=kindexcb(text,'1234567890','中漢'); /* 1 */
  put result=;
run;
```

1 KINDEXCB は、「中」、「漢」、と 0～9 の最初の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result=9
```

KINDEXC 関数

文字式で指定された文字を検索し、文字ベースの値を返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、“[INDEXC 関数](#)” ([SAS 関数と CALL ルーチン: リファレンス](#))です。

構文

KINDEXC(*source*, *excerpt-1*, ...*excerpt-n*)

必須引数

source

検索する文字式を指定します。

excerpt-1, ..., *excerpt-n*

文字式で検索する文字を指定します。

ヒント 複数の *excerpt* を指定する場合は、カンマで区切ります。

文字のリテラル文字列を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KINDEXC 関数は、*source* を左から右へ、*excerpt* 内に含まれるいずれかの文字が最初に現れる個所を検索し、*source* 内でのその文字の位置を返します。*excerpt-1* から *excerpt-n* までの文字がいずれも *source* 内で見つからなかった場合、KINDEXC は値 0 を返します。

比較

KINDEXC 関数は、文字列に含まれる個々の文字が最初に現れる個所を検索するのに対し、KINDEX 関数は、文字列がパターンとして最初に現れる個所を検索します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  result1=kindexc(text,'ée'); /* 1 */
  result2=kindexc(text,'éec'); /* 2 */
  put result1=result2=;
run;
```

- 1 KINDEXC は、文字「é」と「e」の最初の位置を指定します。
- 2 KINDEXC は、文字「é」、「e」、「c」の最初の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result1=3
result2=3
```

例 2

この例では、漢字を使用します。

```
data _null_;
  text='感.def(漢字 123456)';
  result=kindexc(text,'1234567890','中漢'); /* 1 */
  put result=;
run;
```

1 KINDEXC は、「中」、「漢」、と 0～9 の文字の最初の位置を指定します。

上記のステートメントは、次の結果を生成します。

```
result=7
```

関連項目

関数:

- [“KINDEX 関数” \(641 ページ\)](#)

KLEFT 関数

不要な先頭の DBCS の空白と SO/SI を削除し、文字式を左詰めにします。

カテゴリ: CAS
DBCS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

注: この関数は VARCHAR データ型をサポートしています。

ヒント: 非 DBCS で相当する関数は、*SAS 関数*と *CALL ルーチン: リファレンス*の [LEFT](#) を参照してください。

構文

KLEFT(*argument*)

必須引数

argument

SAS 文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KLEFT は、引数を返して先頭の空白を削除します。

KLEFT、KRIGHT および KTRIM 関数は、次の Unicode 空白文字を削除します。

表 13.6 KLEFT、KRIGHT および KTRIM によって削除される Unicode 空白

ANYSOACE での Unicode 空白定義	KLEFT、KRIGHT および KTRIM によって削除さ れる Unicode 空白	コメント
	U+0009	CHARACTER TABULATION
	U+000A	LINE FEED
	U+000B	LINE TABULATION
	U+000C	FORM FEED
	U+000D	CARRIAGE RETURN
U+0020	U+0020	SINGLE-BYTE SPACE, DEPEND ON COMPILING
	U+0085	NEXT LINE
U+00A0	U+00A0	NO-BREAK SPACE
U+1680	U+1680	OGHAM SPACE MARK
U+2000	U+2000	EN QUAD
U+2001	U+2001	EM QUAD
U+2002	U+2002	EN SPACE
U+2003	U+2003	EM SPACE
U+2004	U+2004	THREE-PER-EM SPACE
U+2005	U+2005	FOUR-PER-EM SPACE
U+2006	U+2006	SIX-PER-EM SPACE
U+2007	U+2007	FIGURE SPACE
U+2008	U+2008	PUNCTUATION SPACE
U+2009	U+2009	THIN SPACE
U+200A	U+200A	HAIR SPACE
U+2028	U+2028	U+2028 LINE SEPARATOR

ANYSIZE での Unicode 空白定義	KLEFT、KRIGHT および KTRIM によって削除さ れる Unicode 空白	コメント
U+2029	U+2029	U+2029 PARAGRAPH SEPARATOR
U+202f	U+202F	NARROW NO-BREAK SPACE
U+205f	U+205F	MEDIUM MATHEMATICAL SPACE
U+3000	U+3000	IDEOGRAPHIC SPACE

例

次の例では、UTF-8 エンコーディングを使用します。

この例では、漢字を使用します。

```
data _null_;
  a=' 漢字のテスト'; /* 1 */
  b=kleft(a);
  put '---- +---- 1 ---- + ---- 2 ---- +';
  put a $25.;
  put b $25.;
run;
```

1 最初の文字は DBCS スペースです。

上記のステートメントは、次の結果を生成します。

```
---- +---- 1 ---- + ---- 2 ---- +
 漢字のテスト
 漢字のテスト
```

関連項目

関数:

- “KCOMPRESS 関数” (616 ページ)
- “KRIGHT 関数” (669 ページ)
- “KTRIM 関数” (689 ページ)

KLENGTH 関数

引数の長さを文字数で返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、*SAS 関数と CALL ルーチン: リファレンス*の [LENGTH](#) を参照してください。

構文

KLENGTH(*argument*)

必須引数

argument

SAS 式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KLENGTH 関数は、引数内で最も右にある空白以外の文字の位置を表す整数を返します。引数の値がない場合、KLENGTH は値 0 を返します。引数が初期化されていない数値変数の場合、KLENGTH は値 12 を返し、SAS ログに数値が文字値に変換されたという注意を出力します。

SBCS EnglishVARCHAR データを使っている場合、処理性能に問題が生じる可能性があります。性能向上のため、K 関数でない関数を用いた、次のようなコードにしてください。

性能が向上し、その結果は KLENGTH 関数と同じになります。

```
len = LENGTHC(TRIMN(x));  
if NOT len then len = 1;
```

例

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  result1=klength('Renée Dupont');
  result2=klength('abc あいう');
  put result1= / result2=;
run;
```

上記のステートメントは、次の結果を生成します。

```
result1=13
result2=6
```

KLOWCASE 関数

大文字のアルファベットを小文字に変換します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、[LOWCASE](#) で、[SAS 関数と CALL ルーチン: リファレンス](#)を参照してください。

構文

KLOWCASE(*argument*)

必須引数

argument

SAS 文字式を指定します。

詳細

KLOWCASE 関数により、文字引数がコピーされ、大文字のアルファベット文字が小文字に変換され、変更された値が返されます。

アルファベットの定義は、SAS セッションのエンコーディングによって異なります。典型的な SAS セッションと、KLOWCASE がさまざまなセッションのアルファベットを処理する方法を次に示します。

UTF-8 (SAS Unicode サーバー)

SAS セッションエンコーディングが UTF-8 の場合、Unicode 文字属性が適用されます。文字が文字として分類され、小文字に相当するものがある場合、KLOWCASE は小文字を返します。それ以外の場合は、同じ文字が返されます。

DBCS (たとえば、中国語、日本語、韓国語)

KLOWCASE は、DBCS SAS セッションで、英語のアルファベットの大文字を小文字に変換します。

SBCS(たとえば、英語またはヨーロッパ言語)

SAS セッションエンコーディングが SBCS の場合、TRANTAB オプションが更新され、大文字とそれに相当する小文字のマッピングテーブルが使用されます。小文字のマッピングテーブルは、*nnnn_lcs* の命名パターンを使用します。ここで、*nnnn* は[短いエンコーディング名](#)と一致します。KLOWCASE 関数は、これらのテーブルを使用して、大文字を、セッションエンコーディングでサポートされている小文字に変換します。

例

例 1

この例では、テキストに英語のアルファベットの大文字と大文字の Ñ が混在しています。SAS セッションエンコーディングは、Windows コードページ 1252 です。これは、WLATIN1 です。KLOWCASE は、ñ を含め、すべての文字を小文字で返すことに注意してください。この例は、SAS エンコーディングが SBCS であり、西ヨーロッパ文字をサポートしている場合の KLOWCASE の結果を示しています。SAS エンコーディングが UTF-8 の場合、UTF-8 テキストに対して同じ結果が返されます。

```
options locale=en_US;
data one;
  x='NIÑO';
  y=klowcase(x);

  put x=;
  put y=;
run;
```

```
x=NIÑO
y=niño
```

例 2

```
data _null_;
  result=klowcase('ABC あいうえお');
  put result =;
run;
```

上記のステートメントは、次の結果を生成します。

```
result=abc あいうえお
```

KNEXTCHAR 関数

指定されたバイトオフセットで文字列から文字を返します。

カテゴリ:	CAS 文字操作
制限事項:	この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、 国際化の互換性 (555 ページ) を参照してください。
注:	指定されたオフセットで返された文字が変数に割り当てられます。 <code>onechar=KNEXTCHAR(text,offset</code>
参照項目:	"KOFFSET 関数"

構文

KNEXTCHAR(*text*,*offset*)

必須引数

text

文字定数、変数または式を指定します。

offset

バイトベースの入力値と出力値を表す数値。*offset* の初期値は 1 でなければなりません。

詳細

KNEXTCHAR 関数は、指定されたバイトオフセットで文字列から文字を返します。この関数は、すべての文字を線形時間で効率的に処理するために使用されます。巨大な文字列を処理する際に、より許容範囲の広いパフォーマンスを発揮します。

KOFFSET 関数を KNEXTCHAR と一緒に使用して、*offset* の初期値を取得できます。両方の関数の機能を利用し、ループでデータを処理することにより、文字列の繰り返し処理が非常に効率的になります。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  result=knextchar(text,5); /* 1 */
  put result =;
run;
```

1 5 はバイトオフセットです。

上記のステートメントは、次の結果を生成します。

```
result=R
```

例 2

```
data _null_;
  length x varchar(18);
  x = 'Renée Dupont';
  n = length(x);
  offset = koffset(x); /* 1 */
  do i = 1 to n;
    c = knextchar(x, offset); /* 2 */
    put i= c= offset=;
  end;
run;
```

1 オフセットを 1 に初期化します。

2 オフセットは、KNEXTCHAR を実行した後の次の文字のバイト位置に設定されず。

上記のステートメントは、次の結果を生成します。

```
i=1 c= offset=5
i=2 c=R offset=6
i=3 c=e offset=7
i=4 c=n offset=8
i=5 c=é offset=10
i=6 c=e offset=11
i=7 c= offset=12
i=8 c=D offset=13
i=9 c=u offset=14
i=10 c=p offset=15
i=11 c=o offset=16
i=12 c=n offset=17
i=13 c=t offset=18
```

KOFFSET 関数

character position からのバイトオフセットを返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

参照項目: “KNEXTCHAR 関数”

構文

byte_offset=**KOFFSET**(*text*<,<*character_position*>);

必須引数

text

文字定数、変数または式を指定します。

オプション引数

character_position

文字列内の位置を指定します。

詳細

KOFFSET 関数は、バイトオフセットを指定します。*character_position* が指定されていない場合、KOFFSET は値 1 を返します。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;  
  text='Renée Dupont';  
  result=koffset(text,2); /* 1 */  
  put result =;
```

```
run;
```

- 1 2 番目の文字は「R」で、バイトオフセットは5 です。

上記のステートメントは、次の結果を生成します。

```
result=5
```

例 2

```
data _null_;
  length x varchar(18);
  x = 'Renée Dupont';
  n = length(x);
  offset = koffset(x); /* 1 */
  do i = 1 to n;
    c = knextchar(x, offset); /* 2 */
    put i= c= offset=;
  end;
run;
```

- 1 オフセットを 1 に初期化します。
- 2 オフセットは、KNEXTCHAR を実行した後の次の文字のバイト位置に設定されます。

上記のステートメントは、次の結果を生成します。

```
i=1 c= offset=5
i=2 c=R offset=6
i=3 c=e offset=7
i=4 c=n offset=8
i=5 c=é offset=10
i=6 c=e offset=11
i=7 c= offset=12
i=8 c=D offset=13
i=9 c=u offset=14
i=10 c=p offset=15
i=11 c=o offset=16
i=12 c=n offset=17
i=13 c=t offset=18
```

KPROPCASE 関数

中国語、日本語、韓国語、台湾語(CJKT)文字を変換します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

```
str=KPROPCASE(<instr> , (<options> ))
```

必須引数

str

変換された、現在の SAS セッションエンコーディングのデータ文字列。

instr

入力データ文字列。

options

指定されたオプションに基づいて、日本語、中国語、韓国語、台湾語文字を変換します。

HALF-KATAKANA, FULL-KATAKANA

半角カタカナを全角カタカナに変換します。このオプションは、日本語エンコーディングでのみ使用します。

制限事項 このオプションは、FULL-KATAKANA, HALF-KATAKANA オプションと同時に使用できません。

FULL-KATAKANA, HALF-KATAKANA

全角カタカナを半角カタカナに変換します。このオプションは、日本語エンコーディングでのみ使用します。

制限事項 このオプションは、HALF-KATAKANA, FULL-KATAKANA オプションと同時に使用できません。

KATAKANA, ROMAJI

カタカナ文字列をローマ字文字列に変換します。このオプションは、日本語エンコーディングでのみ使用します。

制限事項 このオプションは、ROMAJI, KATAKANA オプションと同時に使用できません。

ROMAJI, KATAKANA

ローマ字文字列をカタカナ文字列に変換します。このオプションは、日本語エンコーディングでのみ使用します。

制限事項 このオプションは、KATAKANA, ROMAJI オプションと同時に使用できません。

FULL-ALPHABET, HALF-ALPHABET

全角アルファベット文字を半角アルファベット文字に変換します。このオプションは、日本語、中国語、韓国語、台湾語エンコーディングでのみ使用します。

制限事項 このオプションは、HALF-ALPHABET, FULL-ALPHABET オプションと同時に使用できません。

HALF-ALPHABET, FULL-ALPHABET

半角アルファベット文字を全角アルファベット文字に変換します。このオプションは、日本語、中国語、韓国語、台湾語エンコーディングでのみ使用します。

制限事項 このオプションは、FULL-ALPHABET, HALF-ALPHABET オプションと同時に使用できません。

LOWERCASE, UPPERCASE

小文字アルファベット文字を大文字アルファベット文字に変換します。

制限事項 このオプションは、UPPERCASE, LOWERCASE オプションと同時に使用できません。

UPPERCASE, LOWERCASE

大文字アルファベット文字を小文字アルファベット文字に変換します。

制限事項 このオプションは、LOWERCASE, UPPERCASE オプションと同時に使用できません。

PROPER

エンコーディングに基づいて、次のデフォルトオプションを指定します。

- 日本語エンコーディング:
 - ☐ HALF-KATAKANA, FULL-KATAKANA
 - ☐ FULL-ALPHABET, HALF-ALPHABET
 - ☐ LOWERCASE, UPPERCASE
- 韓国語エンコーディング:
 - ☐ FULL-ALPHABET, HALF-ALPHABET
- 中国語エンコーディング:
 - ☐ FULL-ALPHABET, HALF-ALPHABET
- 台湾語エンコーディング:
 - ☐ FULL-ALPHABET, HALF-ALPHABET

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

指定されたオプションとデフォルトオプションに基づいて、入力文字列を変換します。KPROPCASE 関数は、中国語、日本語、韓国語、台湾語(CJKT)環境をサポートします。

比較

PROPCASE 関数は、ラテンベースの言語用に設計されています。単語はスペースなどの区切り文字で定義され、各単語の最初の文字が大文字になります。

KPROPCASE 関数は、簡体字中国語、繁体字中国語、日本語、韓国語などの 2 バイト言語用に設計されています。これらの言語でも適切な大文字と小文字は必要ですが、ラテン語の場合のように頭文字の形式ではありません。そのかわり、全幅変換や半角変換など、他の種類の変換が伴うこともあります。さらに、2 バイト言語は、ラテン言語のように、文字を単語に区切るのに区切り文字を使用しません。

例

次に、KPROPCASE 関数の使用例を示します。

```
length fullkana halfkana upper lower fullalpha $ 200;
length str1 str2 str3 str4 str5 str7 str8 $ 30 str6 $44;
lower = 'do-naxtutsu'; /* Doughnuts in Japanese Roman word. */
upper = 'DO-NAXTUTSU'; /* Doughnuts in Japanese Roman word. */
fullkana = unicode("\u30C9\u30FC\u30CA\u30C3\u30C4");
halfkana = unicode("\uFF84\uFF9E\uFF70\uFF85\uFF6F\uFF82");
fullalpha = unicode("\uFF24\uFF2F\uFF0D\uFF2E\uFF21\uFF38\uFF34\uFF35\uFF34\uFF33\uFF35");
str1 = kpropcase(fullkana, 'full-katakana,half-katakana');
if (halfkana EQ trim(str1)) then
  put str1= $hex14.;
str2 = kpropcase(halfkana, 'half-katakana, full-katakana');
if (fullkana EQ trim(str2)) then
  put str2= $hex22.;
str3 = kpropcase(fullkana, 'katakana,romaji');
if (trim(str3) EQ upper) then
  put str3= ;
str4 = kpropcase(upper, 'romaji,katakana');
if (trim(str4) EQ fullkana) then
  put str4= $hex22.;
str5 = kpropcase(fullalpha, 'full-alphabet, half-alphabet');
if (trim(upper) EQ str5) then
  put str5=;
str6 = kpropcase(upper, 'half-alphabet, full-alphabet');
if (trim(str6) EQ fullalpha) then
  put str6= $hex46.;
str7 = kpropcase(lower, 'lowercase, uppercase');
if (trim(str7) EQ upper) then
  put str7=;
str8 = kpropcase(upper, 'uppercase, lowercase');
if (trim(str8) EQ lower) then
  put str8=;
```

```

str1=C4DEB0C5AFC220
str2=8368815B83698362836320
str3=DO-NAXTUTSU
str4=8368815B83698362836320
str5=DO-NAXTUTSU
str6=8263826E817C826D826082778273827482738272827420
str7=DO-NAXTUTSU
str8=do-naxtutsu

```

KPROPCCHAR 関数

Unicode を、括弧で囲まれた対応する文字に変換します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

str=KPROPCCHAR(<*instr*>)

必須引数

str

結果文字列。Unicode は文字に変換されます。

instr

入力データ文字列。

詳細

この関数は、①(0x2460)などの Unicode を同等の文字(1)に変換します。KPROPCCHAR 関数では、次の範囲の文字を変換します。

- 囲まれた英数字値: \u2460 から\u24FF。 <http://www.unicode.org/charts/PDF/U2460.pdf> を参照してください。
- Dingbats: \u2776 から\u2793。 <http://www.unicode.org/charts/PDF/U2700.pdf> を参照してください。
- 囲まれた CJK 文字と月: \u3200 から\u32FF。 <http://www.unicode.org/charts/PDF/U3200.pdf> を参照してください。

例

次に、KPROPCHAR 関数の使用例を示します。次の例は UTF8 SAS セッションで処理されます。

```
length in1 out1 $30 ;
in1=unicode('\u2460\u2473\u277F\u325F');
out1=KPROPCHAR(in1);
put out1;
RESULTS:
(1)(20)(-10)(35)
```

ステートメント	結果
data _null_; length in1 out1 \$30 ; in1=unicode('\u2460\u2473\u277F\u325F'); out1=KPROPCHAR(in1); put out1; run;	(1)(20)(-10)(35)
data _null_; length in1 out1 \$30 ; in1=unicode('\u2776\u2793'); out1=KPROPCHAR(in1); put out1; run;	(-1)(-10)
data _null_; length in1 out1 \$30 ; in1=unicode('\u3200\u32FE'); out1=KPROPCHAR(in1); put out1; run;	(ー)(ヲ)

KPROPDATA 関数

印刷不可文字を削除または変換します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

```
str=KPROPDATA(<instr>, <options>, <input encode name>, <output encode name>)
```

必須引数

str

変換された、セッションエンコーディングでのデータ文字列を指定します。

instr

入力データ文字列を指定します。

options

印刷不可文字の処理に関する指示を指定します。

UESC

Unicode エスケープ文字列(たとえば、\u0000\u1234)を使用して印刷不可文字を変換します。

TRIM

印刷不可文字を削除します。置換文字は使用しません。

BLANK or ' '

各印刷不可文字を 1 バイトの空白で置換します。

QUESTION or '?'

印刷不可文字を 1 バイトの '?' で置換します。

HEX

印刷不可文字を 16 進表現(たとえば、0x810x82)で置換します。

TRUNCATE or TRUNC

最初の印刷不可文字が出現したときにデータ文字列を切り捨てます。

REMOVE

いずれかの印刷不可文字が検出された場合にデータ文字列を削除します。

NCR

Unicode にコードが存在する場合は、NCR 表示形式を使用して印刷不可文字をエンコードします。

PUNC

注: PUNC オプションは、SAS 9 ではサポートされていません。

一般的に使用される 8 ビットの句読点を 7 ビット文字に変換します。

WLATIN1 文字は 1 バイトですが、UTF-8 ではこれらの文字の一部は 2~4 バイトです。このオプションは、WLATIN1 では 1 バイトですが、UTF-8 では 2~4 バイトの句読点を変換します。変換できる句読点は次のとおりです。

表 13.7 スマート引用符と句読文字

カテゴリ	句読点				変換		
	句読文字	説明	WLATIN1 コードポ イント (Hex)	UTF-8 コード ポイン ト (Hex)	ASCII 文字	説明	ASCII コ ードポ イント (Hex)
引用符	,	下つき引用 符	82	E2 80 9A	,	カンマ	2C
	„	下つき二重 引用符	84	E2 80 9E	„	引用符	22
	‘	引用符(始)	91	E2 80 98	'	アポストロ フィ	27
	’	引用符(終)	92	E2 80 99	'	アポストロ フィ	27
	“	二重引用符 (始)	93	E2 80 9C	"	引用符	22
	”	二重引用符 (終)	94	E2 80 9D	"	引用符	22
山かっこ	<	山カッコ (始)	8B	E2 80 B9	<	小なり記号	3C
	>	山カッコ(終)	9B	E2 80 BA	>	大なり記号	3E
省略記号	...	三点リーダ ー	85	E2 80 A6	...	終止符	2E 2E 2E ¹
行頭文字	•	黒丸	95	E2 80 A2	*	アスタリス ク	2A
ハイフン	-	en ハイフン	96	E2 80 93	-	ハイフンマ イナス	2D
	—	em ハイフ ン	97	E2 80 94	-	ハイフンマ イナス	2D

1 3つのASCIIピリオド文字。

input encode name

必要に応じて、入力データのエンコーディング名を指定します。入力エンコーディング名を指定しないと、KPROPDATA 関数では、現在の SAS セッションでエンコードされた文字列としてデータを処理します。SAS エンコーディング名の詳細については、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ)を参照してください。

output encode name

出力データのエンコーディング名を指定します。エンコーディング名を指定しないと、KPROPDATA 関数では、現在の SAS セッションエンコーディングとして出力を認識します。SAS エンコーディング名の詳細については、[“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” \(1118 ページ\)](#)を参照してください。

詳細

入力データ文字列を現在の SAS セッションエンコーディングに変換し、オプションに基づいて印刷不可文字を削除または置換します。

例

例 1

次に、KPROPDATA 関数の使用例を示します。この例では、LATIN2 セッションエンコーディングを使用します。この例では、登録された記号(latin1 セッションの'ae'x)は LATIN2 エンコーディングではサポートされていません。この例は、サポートされていない文字を表すのに KPROPDATA がどのように役立つかを示しています。

```
data test;
  length instr $12;
  length str1 str2 str3 str4 str5 str6 str7 str8 $ 50;
  instr = "SAS" || "ae"x || " System";
  put instr;
  str1 = kpropdata(instr, 'UESC', 'wlatin1');
  put str1= +2 str1= $hex34.;
  str2 = kpropdata(instr, 'TRIM', 'wlatin1');
  put str2= +2 str2= $hex26.;
  str3 = kpropdata(instr, 'BLANK', 'wlatin1');
  put str3= +2 str3= $hex26.;
  str4 = kpropdata(instr, '?', 'wlatin1');
  put str4= +2 str4= $hex26.;
  str5 = kpropdata(instr, 'hex', 'wlatin1');
  put str5= +2 str5= $hex26.;
  str6 = kpropdata(instr, 'TRUNC', 'wlatin1');
  put str6= +2 str6= $hex26.;
  str7 = kpropdata(instr, 'REMOVE', 'wlatin1');
  put str7= +2 str7= $hex26.;
  str8 = kpropdata(instr, 'NCR', 'wlatin1');
  put str8= +2 str8= $hex26.;
run;
```

上記のステートメントは、次の結果を生成します。


```

SASŽ System
str1=SAS\u00ae System   str1=5341535C75303061652053797374656D20
str2=SAS System         str2=5341532053797374656D202020
str3=SAS System         str3=534153202053797374656D2020
str4=SAS? System        str4=5341533F2053797374656D2020
str5=SAS\xae System     str5=5341535C786165205379737465
str6=SAS                str6=53415320202020202020202020
str7=                    str7=20202020202020202020202020
str8=SAS&#174; System   str8=53415326233137343B20537973

```

例 2

この例は、PUNC オプションの機能を示しています。*instr* に割り当てられた文字列には、ASCII 文字ではない句読文字がいくつか含まれています。この例は、WLATIN1 セッションエンコーディングを使用して SAS で実行されました。

```

data _null_;
  length instr $48;
  length outstr $14;
  instr = "—' , \" , •...<>"; /* "969791928293948495858B9B"x */
  put instr= /instr=hex.;
  outstr = kpropdata(instr, 'PUNC');
  put outstr= /outstr=hex.;

run;

```

上記のステートメントは、次の結果を生成します。

```

instr =--' , \" , •...<>
instr  =979691928293948495858B9B
outstr=--' , \" , •...<>
outstr=2D2D27272C222222A2E2E2E3C3E
\

```

例 3

この例では、UTF-8 セッションエンコーディングを使用します。*instr* に割り当てられた値に、無効な文字/バイトである *'ae'x* が含まれています。KPROPDATA は、無効な UTF-8 文字を処理します。

```

data test;
  length instr $12;
  length str1 str2 $50;
  instr = "SAS" || 'ae'x || " System"; /* 'ae'x represents here the invalid character/byte */
  put instr;
  str1 = kpropdata(instr, 'hex');
  put str1= +2 str1= $hex26.;
  str2 = kpropdata(instr, '?');
  put str2= +2 str2= $hex26.;

run;

```

これらのステートメントは、次の結果を生成します: KPROPDATA を使用すると、無効な UTF-8 文字'ae'x が処理されることに注意してください。

```
SAS? System
str1=SAS\xae System str1=5341535C786165205379737465
str2=SAS? System str2=5341533F2053797374656D2020
```

KREVERSE 関数

文字式を逆にします。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、[SAS 関数と CALL ルーチン: リファレンス](#)で、[REVERSE](#)を参照してください。

構文

KREVERSE(*argument*)

必須引数

argument

SAS 文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

例

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  result1=kreverse('Renée Dupont');
  result2=kreverse('漢字のテスト');
  put result1= /result2=;
run;
```

上記のステートメントは、次の結果を生成します。

```
result1=tnopuD eéneR
result2=トステの字漢
```

KRIGHT 関数

末尾の DBCS の空白と SO/SI を削除し、文字式を右詰めにします。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: “RIGHT 関数” ([SAS 関数と CALL ルーチン: リファレンス](#))を参照してください。

構文

KRIGHT(*argument*)

必須引数

argument

SAS 文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KRIGHT 関数は、末尾の空白を値の先頭に移動した引数を返します。引数の長さは変わりません。

KRIGHT、KLEFT および KTRIM 関数は、Unicode 空白文字を削除します。Unicode 文字のリストは[表 13.180 \(650 ページ\)](#)を参照してください。

例

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  a='漢字のテスト'; /* 1 */
```

```

b=kright(a);
put '---- +---- 1 ---- + ---- 2 ---- +';
put a $25.;
put b $25.;
run;

```

1 最後の文字は DBCS スペースです。

上記のステートメントは、次の結果を生成します。

```

---- +---- 1 ---- + ---- 2 ---- +
漢字のテスト
漢字のテスト

```

関連項目

関数:

- [“KCOMPRESS 関数” \(616 ページ\)](#)
- [“KLEFT 関数” \(649 ページ\)](#)
- [“KTRIM 関数” \(689 ページ\)](#)

KSCAN 関数

指定された単語を文字式から選択します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、*SAS 関数*と *CALL ルーチン: リファレンスの SCAN* を参照してください。

構文

KSCAN(*argument*,*n*<, *delimiters*>)

必須引数

argument

文字式を指定します。

n

KSCAN が文字式内で選択する単語の番号を算出する数式を指定します。

ヒント *n* が負の場合、KSCAN は、文字式内の単語を文字列の最後から検索して選択します。|*n*| が文字式内の単語数よりも大きい場合、KSCAN は空白値を返します。

delimiters

KSCAN が文字式内で単語区切り文字として使用する文字を指定します。

デフォルト ASCII 環境で *delimiters* を省略すると、SAS は *blank*(空白)を使用します。
 <(+ & !\$ *); ^ - / , % |を使用します。^文字のない ASCII 環境の場合、KSCAN はかわりに ~文字を使用します。

EBCDIC 環境で *delimiters* を省略すると、SAS は *blank*(空白)を使用します。
 <(+ | & !\$ *); ~ - / , % | ¢を使用します。

ヒント *delimiters* を定数として表す場合は、*delimiters* を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

区切り文字が文字列内で先頭の単語より前にあっても、KSCAN に影響はありません。複数の区切り文字が連続する場合、KSCAN はそれらをまとめて 1 つとして扱います。

KSCAN 関数は、*modify* 引数を持ちません。*modify* 機能が必要な場合は、[KSCANX](#) 関数を使用します。KSCANX は、SCAN 関数の完全な NLS 置換です。

例

例 1

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée Dupont';
  result=kscan(text,2); /* 1 */
  put result=;
run;
```

1 KSCAN は 2 番目の単語を指定します。SBCS スペースは区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=Dupont
```

例 2

```
data _null_;
  text='Renée×Dupont';
  result=kscan(text,2,'x'); /* 1 */
  put result=;
run;
```

1 KSCAN は 2 番目の単語を指定します。x は区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=Dupont
```

例 3

この例では、漢字を使用します。

```
data _null_;
  length x $20 y $20;
  text1='これは漢字関数のテストです。';
  x='nonblank';
  i=1;
  do until (x=' ');
    x=kscan(text1,i,'はの');
    put x =;
    i=i+1;
  end;
  y='nonblank';
  i=-1;
  do until (y=' ');
    y=kscan(text1,i,'はの');
    put y =;
    i=i-1;
  end;
run;
```

上記のステートメントは、次の結果を生成します。

```
x=これ
x=漢字関数
x=テストです。
x=
y=テストです。
y=漢字関数
y=これ
y=
```

関連項目

■ [“KSCANX 関数” \(673 ページ\)](#)

■ “SCAN 関数” (SAS 関数と CALL ルーチン: リファレンス)

KSCANX 関数

関数のアクションを処理する修飾子を使用して、文字式から指定された単語を選択します。

カテゴリ: CAS

文字操作

制限事項: この関数には I18N レベル 2 のステータスが割り当てられ、DBCS データ用に設計されています。ただし、最初の引数 `string` が 1 バイト文字の場合、KSCANX 関数は SBCS データを処理します。詳細については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

構文

KSCANX(*string*,*count*,<*character-list*>,<*modifier*>)

必須引数

string

文字定数、変数または式を指定します。

count

整数値を使用するゼロ以外の数値の定数、変数または式です。この整数値によって、KSCANX が文字列内で選択する単語の番号が指定されます。たとえば、値 1 は最初の単語を示し、値 2 は 2 番目の単語を示すなどです。次のルールが適用されます。

- *count* が正の場合、KSCANX は文字列の左から右へ単語を数えます。
- *count* が負の場合、KSCANX は文字列の単語を右から左へ数えます。

オプション引数

character-list

文字のリストを初期化する文字式を指定します(省略可能)。このリストは、単語を区切る区切り文字として使用する文字を決定します。次のルールが適用されます。

- デフォルトでは、*character-list* のすべての文字が区切り文字として使用されます。
- 修飾子を指定すると、区切り文字として使用される文字リストを変更することもできます。たとえば、*modifier* 引数に K 修飾子を指定すると、*character-list* にない文字がすべて区切り文字として使用されます。

注: 詳細については“[ASCII 環境でのデフォルトの区切り文字の使用](#)” (SAS 関数と CALL ルーチン: リファレンス)を参照してください。

ヒント その他の修飾子を使用して `character-list` にさらに文字を追加できます。

modifier

文字定数、変数または式を指定します。ここでは、空白以外の文字のそれぞれによって KSCANX 関数のアクションが変更されます。空白は無視されます。修飾子として次の文字を使用します。

- a または A 文字のリストに英文字が追加されます。
- b または B *string* が、左から右へではなく、右から左へスキャンされます。
- c または C 文字のリストにコントロール文字が追加されます。
- d または D 文字のリストに数字が追加されます。
- f または F 文字のリストにアンダースコアと英文字(`VALIDVARNAME=V7` を使用した SAS 変数名内の有効な最初の文字)を追加します。
- g または G 文字のリストにグラフィック文字が追加されます。グラフィカル文字は、紙面に印刷するとイメージになる文字です。
- h または H 文字のリストに水平タブが追加されます。
- i または I 大文字と小文字の区別が無視されます。
- k または K 文字のリストにないすべての文字が区切り文字として扱われるようになります。つまり、K を指定すると、文字リストに含まれている文字が、区切り文字として省略されるのではなく、戻り値内に保持されるようになります。K を指定しない場合、文字のリストに含まれているすべての文字が区切り文字として扱われます。
- l または L 文字のリストに小文字が追加されます。
- m または M 複数の連続した区切り文字および *string* 引数の先頭または末尾にある区切り文字が、長さゼロの単語を参照するように指定されます。M 修飾子を指定しない場合、複数の連続する区切り文字は 1 つの区切り文字として扱われ、*string* 引数の先頭または末尾の区切り文字は無視されます。
- n または N 文字のリストに数字、アンダースコアおよび英文字(`VALIDVARNAME=V7` を使用した SAS 変数名内に表示可能な文字)を追加します。
- o または O *character-list* および *modifier* 引数が、KSCANX 関数を呼び出すたびにではなく、1 回だけ処理されます。DATA ステップ(WHERE 句を除く)または SQL プロシジャで O 修飾子を使用すると、*character-list* および *modifier* 引数が変更されないため、KSCANX をループで呼び出す際により速く実行させられます。O 修飾子は、SAS コード内の KSCANX 関数の各インスタンスに個別に適用され、

KSCANX 関数のすべてのインスタンスで同じ区切り文字と修飾子を使用することにはなりません。

- p または P 文字のリストに句読点が追加されます。
- q または Q 引用符で囲まれた部分文字列内にある区切り文字が無視されます。*string* 引数の値に一致しない引用符が含まれている場合は、左から右へのスキャンによって、右から左へのスキャンとは異なる単語が生成されます。
- r または R KSCANX が返す単語から先頭と末尾の空白を削除します。Q 修飾子と R 修飾子を指定すると、KSCANX 関数はまず単語の先頭と末尾の空白を削除します。次に、単語が引用符で始まる場合、KSCANX は単語から引用符も 1 階層削除します。
- s または S 文字のリストに空白文字が追加されます(空白、水平タブ、垂直タブ、キャリッジリターン、ラインフィード、フォームフィード)。
- t または T *string* 引数と *character-list* 引数から末尾の空白を取り除きます。両方の文字引数ではなく一方のみから末尾の空白を削除する場合は、KSCANX ルーチンで T 修飾子を使用するかわりに、TRIM 関数を使用します。
- u または U 文字のリストに大文字が追加されます。
- w または W 文字のリストに印刷可能(書き込み可能)文字が追加されます。
- x または X 文字のリストに 16 進数の文字が追加されます。

ヒント *modifier* 引数が文字定数の場合は、引数を引用符で囲みます。1 組の引用符に複数の修飾子を指定します。*modifier* 引数は、文字変数または式として表現することもできます。

詳細

区切り文字と単語

単語を区切るために使用されるいくつかの文字はいずれも区切り文字と呼ばれます。*character-list* および *modifier* 引数に区切り文字を指定できます。

Q 修飾子を指定すると、引用符で囲まれた部分文字列内の区切り文字が無視されます。

KSCANX 関数では、次のすべての条件を満たす部分文字列を単語とします。

- 左境界が、区切り文字または文字列の先頭である
- 右境界が、区切り文字または文字列の末尾である

- **単語**は、その文字列の先頭または末尾に区切り文字がある場合、またはその文字列に 2 つ以上の連続した区切り文字が含まれる場合には長さが 0 となります。ただし、M 修飾子を指定しない限り、KSCANX 関数では長さがゼロの単語は無視されます。

ASCII 環境と EBCDIC 環境でデフォルトの区切り文字を使用する

KSCANX 関数に指定する引数が 2 つのみの場合、デフォルトの区切り文字は、コンピュータの使用文字が ASCII であるか EBCDIC であるかによって異なります。

- コンピューターで ASCII 文字が使われている場合、デフォルトの区切り文字は次のとおりです。

```
blank!$%&()*+,-./;<^_|/;<^|
```

^文字が含まれない ASCII 環境では、KSCANX 関数はかわりに~文字を使用します。

- お使いのコンピューターが EBCDIC 文字を使用している場合、デフォルトの区切り文字は次のようになります。

```
空白!$%&()*+,-./;<-|¢
```

区切り文字として文字を指定せずに *modifier* 引数を使用する場合、使用される区切り文字は *modifier* 引数で定義された区切り文字だけです。この場合、ASCII 環境と EBCDIC 環境のデフォルトの区切り文字のリストは使用されません。つまり、修飾子は *character-list* 引数で明示的に指定された区切り文字のリストに追加します。修飾子は、デフォルトの修飾子のリストには追加しません。

M 修飾子を指定した KSCANX 関数の使用

M 修飾子を指定すると、文字列内の単語の数は、文字列内の区切り文字の数に 1 を加えたものとして定義されます。ただし、Q 修飾子を指定すると、引用符内の区切り文字は無視されます。

M 修飾子を指定した場合、次のいずれかの条件が真のとき、KSCANX 関数は長さゼロの単語を返します。

- 文字列が区切り文字で始まり、最初の単語が要求されている。
- 文字列が区切り文字で終わり、最後の単語が要求されている。
- 文字列に 2 つの連続する区切り文字が含まれており、2 つの区切り文字の間にある単語が要求されている。

M 修飾子を指定しない KSCANX 関数の使用

M 修飾子を指定しなければ、文字列内の単語の数は、連続する区切り文字ではない文字の最大部分文字列の数として定義されます。ただし、Q 修飾子を指定すると、引用符内の区切り文字は無視されます。

M 修飾子を指定しない場合、KSCANX 関数は次のように作動します。

- 文字列の先頭または末尾の区切り文字を無視する
- 2 つ以上の連続する区切り文字を単一の区切り文字のように扱う

文字列に区切り文字以外の文字が含まれていない場合や、文字列内の単語数よりも絶対値が大きい数を指定した場合、KSCANX 関数は、次の項目のいずれかを返します。

- DATA ステップから KSCANX 関数を呼び出す場合は、1 つの空白
- マクロプロセッサから KSCANX 関数を呼び出す場合は、長さ 0 の文字列。

null 引数の使用

この例では、文字列をスキャンして最初と最後の単語を探します。

- 負の数を指定すると、KSCANX 関数が右から左にスキャンするように指示されます。
- 先頭と末尾の区切り文字は、M 修飾子を使用されていないため無視されます。
- 最後のオブザベーションでは、文字列のすべての文字が区切り文字です。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  text='Renée,Dupont';
  result=kscanx(text,2); /* 1 */
  put result=;
run;
```

- 1 KSCANX は 2 番目の単語を指定します。SBCS スペースは区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=Dupont
```

例 2

```
data _null_;
  text='RenéexDupont';
  result=kscan(text,2,'x'); /* 1 */
  put result=;
run;
```

- 1 KSCANX は 2 番目の単語を指定します。x は区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=Dupont
```

例 3

この例では、漢字を使用します。

```
data _null_;
text='SAS 是全球最大的软件公司之一,sas 是全球商业智能和分析软件与服务领袖。';
result=kscanx(text, 2, ", 'sp'); /* 1 */
put result=;
run;
```

1 スペースと句読点は区切り文字です。

上記のステートメントは、次の結果を生成します。

```
result=sas
```

関連項目

- [“SCAN 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)
- [“KSCAN 関数”](#)

KSTRCAT 関数

複数の文字式を連結します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、[SAS 関数と CALL ルーチン: リファレンス](#)で、[CAT](#) を参照してください。

構文

KSTRCAT(*argument-1*, *argument-2*<, ... *argument-n*>)

必須引数

argument

1 バイトまたは 2 バイトの文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KSTRCAT 関数が事前に長さを割り当てられていない変数に値を返す場合、デフォルトで変数の長さは 200 に割り当てられます。

SQL プロシジャ、または任意のプロシジャの WHERE 句で、KSTRCAT 関数が返す単語の最大長は 200 文字です。

例

例 1

次の例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では絵文字を使用します。

```
data _null_;  
  text1='Renée';  
  text2='Dupont';  
  result=kstrcat (text1, '👤', text2);  
  put result=;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result=Renée Dupont
```

例 2

この例では、漢字を使用します。

```
data _null_;  
  text1='漢字';  
  text2='文字列';  
  text3='連結';  
  result=kstrcat (text1, text2,'の', text3);  
  put result=;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result=漢字文字列の連結
```

KSTRIP 関数

文字列から先頭と末尾の空白を削除します。

カテゴリ: CAS
文字

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

注: この関数は VARCHAR データ型をサポートしています。

構文

KSTRIP(*string*)

必須引数

string
文字定数、変数または式を指定します。

詳細

KSTRIP 関数は、先頭と末尾の SBCS の空白をすべて削除した引数を返します。引数が空白の場合、KSTRIP は長さがゼロの文字列を返します。

KSTRIP の結果を変数に割り当てる場合、受け取る変数の長さには影響しません。調整された値が受け取る変数の長さより短い場合、値の末尾に新しく空白が埋め込まれます。

DATA ステップでは、KSTRIP 関数が事前に長さを割り当てられていない変数に値を返す場合、その変数に引数の長さが割り当てられます。

比較

空白の文字列に対して、KSTRIP 関数と STRIP 関数は長さがゼロの文字列を返し、TRIM 関数は単一の空白を返します。

KSTRIP 関数と STRIP 関数は、受け取る変数の長さが入力文字列より短い場合、出力を切り捨てます。STRIP 関数は SBCS 文字列の切り捨てのみ処理します。KSTRIP 関数は SBCS データと DBCS データを処理します。

KSTRIP 関数と STRIP 関数は、不要な空白を削除することによってデータを正規化します。KSTRIP 関数は DBCS 環境で使用する必要があります。KSTRIP 関数と STRIP 関数は同じように動作します。KTRIM 関数と KLEFT 機能は、データの正規化にも使用できます。ただし、KTRIM 関数と KLEFT 関数は、次のコード例のように、すべての種類の DBCS スペースを削除し、パフォーマンスの問題を引き起こす可能性があります。

```
long_char_variable = KTRIM(KLEFT(long_char_variable));
```

KSTRIP 関数では、これらの問題が解決されます。コードを変更し、KSTRIP 関数を使用して DBCS SAS セッションで DBCS データ正規化を正常に実行するには、次のコードを使用します。

```
long_char_variable = KSTRIP(long_char_variable);
```

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  str = ' Renée '; /* 1 */
  result1 = '*' || str || '*';
  result2 = '*' || kstrip(str) || '*';
  put result1= / result2=;
run;
```

1 文字列の先頭と末尾に SBCS 空白が含まれます。

上記のステートメントは、次の結果を生成します。

```
result1=* Renée *
result2=*Renée*
```

例 2

次の例では、KSTRIP 関数による先頭と末尾の空白の削除方法、および DBCS 文字の切り捨て方法について示します。

```
data _null_;
  length a $5;
  v = unicode("\u0020\u0020\u8404\u8404\u8404\u0020\u0020");
  put v = hex;
  a = kstrip(v);
  put a = hex;
```

```
run;
```

上記のステートメントは、次の結果を生成します。

```
93B893B820
```

関連項目

関数

- [“STRIP 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)

KSUBSTR 関数

引数から部分文字列を抽出します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: [“擬似 SUBSTR\(割り当ての右辺に用いた場合\) 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)を参照してください。

構文

KSUBSTR(*argument*, *position*<,*n*>)

必須引数

argument

SAS 文字式を指定します。

position

文字の開始位置となる数値式を指定します。

n

抽出する部分文字列の長さを示す数値式を指定します。

操作 *n* が *argument* 内で *position* より後に残った式の長さよりも長い場合、式の残りの部分が抽出されます。

ヒント *n* を省略すると、式の残りの部分が抽出されます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KSUBSTR 関数は、式内の *argument* で指定された部分を返します。この部分は、*position* で指定された文字から始まり、*n* 文字の長さです。

KSUBSTR で作成される変数の長さは、*argument* の長さから求めることができます。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;  
  text='Renée Dupont';  
  result=ksubstr(text,3,5); /* 1 */  
  put result=;  
run;
```

- 1 KSUBSTR は、3 番目の文字から始まり、5 つの文字を選択して文字列から文字を指定します。

上記のステートメントは、次の結果を生成します。

```
result=née D
```

例 2

この例では、漢字を使用します。

```
data _null_;  
  text='漢字文字列の抽出';  
  result1=ksubstr(text, 1,2);  
  result2=ksubstr(text,3,4);  
  kstart=7;  
  klen=2;  
  result3=ksubstr(text,kstart,klen);  
  put result1=/result2=/result3=;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result1=漢字  
result2=文字列の  
result3=抽出
```

関連項目

関数:

- [“KSUBSTRB 関数” \(684 ページ\)](#)

KSUBSTRB 関数

引数内の部分文字列のバイト位置に従って、引数から部分文字列を抽出します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

KSUBSTRB 関数は、インメモリの CAS テーブルまたは拡張子が.hdat の SAS ファイルには使用できません。

KSUBSTRB 関数は.hdat 拡張子では機能しません。

構文

KSUBSTRB(*argument*, *position*<*n*>)

必須引数

argument

SAS 文字式を指定します。

position

開始文字位置をバイト単位で指定します。

n

抽出する部分文字列の長さをバイト単位で指定します。

操作 *n* が *argument* 内で *position* より後に残った式の長さ(バイト単位)よりも長い場合、式の残りの部分が抽出されます。

ヒント *n* を省略すると、式の残りの部分が抽出されます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性” \(555 ページ\)](#)を参照してください。

KSUBSTR 関数は、式内の *argument* で指定された部分を返します。この部分は、*position* で指定されたバイト単位で始まり、*n* バイトの長さです。

KSUBSTRB で作成される変数の長さは、*argument* の長さから求めることができます。

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

```
data _null_;  
  text='Renée Dupont';  
  result=ksubstrb(text,3,4); /* 1 */  
  put result=;  
run;
```

1 KSUBSTRB は、バイト位置 3 から始まる合計 4 バイトの文字列を指定します。

上記のステートメントは、次の結果を生成します。

```
result=née
```

例 2

この例では、漢字を使用します。

```
data _null_;  
  text='漢字文字列の抽出';  
  result1=ksubstrb(text,1,9);  
  result2=ksubstrb(text,4,6);  
  put result1=/result2=;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result1=漢字文  
result2=字文
```

関連項目

関数:

■ [“KSUBSTR 関数” \(682 ページ\)](#)

KSUBSTRN 関数

部分文字列を返します。長さがゼロの結果も返せます。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。

参照項目: 詳細については、“[SUBSTRN 関数](#)” ([SAS 関数と CALL ルーチン: リファレンス](#))および“[SAS 文字列関数の国際化の互換性](#)” ([555 ページ](#))を参照してください。

構文

KSUBSTRN(*string*,*position*,<*length*>)

必須引数

string

文字または数値の定数、変数または式を指定します。

string が数値の場合、BEST32.形式の文字値に変換されます。先頭と末尾の空白は削除され、メッセージは SAS ログに送信されません。数値引数は PROC SQL ではサポートされていません。

position

部分文字列の最初の文字の位置を指定する整数です。

オプション引数

length

部分文字列の長さを指定する整数です。length の指定がなければ、KSUBSTRN 関数は指定された位置から末尾までの部分文字列を返します。

詳細

KSUBSTRN の機能は SUBSTRN に似ています。違いは、第 1 パラメーターが数値であるとき、SUBSTRN の出力長が 8 であるのに対して、KSUBSTRN の出力長は 200 である点です。

例

KSUBSTR と KSUBSTRN の違いが分かる例を示します。

```
data _null_;
```

```
ksubstr_result = "*" || ksubstr(1234.5678,2,6) || "*";
put ksubstr_result;
ksubstrn_result = "*" || ksubstrn(1234.5678,2,6) || "*";
put ksubstrn_result;
run;
```

上記のステートメントは、次の結果を生成します。

```
ksubstr_result=* 1234*
ksubstrn_result=*234.56*
```

KTRANSLATE 関数

文字式内の指定された文字を置換します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください

ヒント: 非 DBCS で相当する関数は、[SAS 関数と CALL ルーチン: リファレンス](#)で [TRANSLATE](#) を参照してください。

構文

KTRANSLATE(*source,to-1,from-1<,...to-n,from-n>*)

必須引数

source

元の文字値を含む SAS 式を指定します。

to

KTRANSLATE で代替文字として使用する文字を指定します。

from

KTRANSLATE で置換する文字を指定します。

操作 *to* と *from* に指定された値は、1 文字ずつ対応します。KTRANSLATE は、*from* の 1 文字を *to* の 1 文字に変換するという具合に処理を続けます。*to* の文字が *from* の文字より少ない場合、KTRANSLATE は *from* の余った文字を空白に変更します。*to* の文字が *from* の文字より多い場合、KTRANSLATE は *to* の余った文字を無視します。

注 一部の動作環境では、*to* と *from* の引数ペアが必要です。照合順序のセグメントが、NULL である *from* の引数に置き換わる動作環境もあります。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KTRANSLATE を使用して、1 バイト文字式を 2 バイト文字式に変換したり、2 バイト文字式を 1 バイト文字式に変換したりできます。

KTRANSLATE が受け入れる *to* および *from* 文字ペアの最大数は 256 以下にしてください。短い引数のペアを数多く使用する場合と、長い引数のペアを少数使用する場合との機能的な違いはありません。これら 2 つの例では、引数と文字の組み合わせのペアが異なりますが、出力は同じです。

1 番目の例: `x=ktranslate('AABBAABABB','12','AB','3','C');`は、2 つの引数ペアと 3 つの文字ペアがあります。

2 番目の例: `y=ktranslate('AABBAABABB','123','ABC');`1 つの引数ペアと 3 つの文字ペアがあります。

注: CAS は 2 番目の例のみをサポートします。1 つの引数ペアと最大 256 文字のペアです。余分な引数のペアは無視されます。

SAS Viya プラットフォームは、無制限の引数ペアと最大 256 文字のペアをサポートします。

```
data test;
x=ktranslate('AABBAABABB','12','AB','3','C'); /* Two to/from argument pairs, three from/to char pairs. */
y=ktranslate('AABBAABABB','123','ABC'); /* One to/from argument pair, three from/to char pairs. */
put x=;
put y=;
run;
```

```
x=1122112122
y=1122112122
```

KTRANSLATE 関数が事前に長さを割り当てられていない変数に値を返す場合、変数の長さはデフォルトの 200 に割り当てられます。

例

例 1

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  result=ktranslate('Renée Dupont','π','é');
  put result =;
run;
```

上記のステートメントは、次の結果を生成します。

```
result=Renne,Dupont
```

例 2

この例では、漢字を使用します。

```
data _null_;  
  result=ktranslate('仲等','中忠','仲仲');  
  put result =;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result=中等
```

KTRIM 関数

末尾の DBCS の空白と SO/SI を文字式から削除します。

カテゴリ: CAS
DBCS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: 非 DBCS で相当する関数は、“[TRIM 関数](#)” ([SAS 関数と CALL ルーチン: リファレンス](#))です。

構文

KTRIM(*argument*)

必須引数

argument
SAS 文字式を指定します。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KTRIM では、文字の引数をコピーし、すべての末尾の空白を削除して、その結果調整された引数を返します。引数が空白の場合、KTRIM は 1 つの空白を返します。連結では末尾の空白は削除されないため、KTRIM は連結する場合に便利です。

注: 引数が空白の場合、CHAR タイプについては KTRIM から 1 つの空白が返されます。引数が空白の場合、varchar 型については KTRIM から長さ 0 の空白が返されます。varchar タイプは文字セマンティックス、CHAR タイプはバイトとしてのセマンティックスに従います。

KTRIM の結果を変数に割り当てる場合、受け取る変数の長さには影響しません。調整された値が受け取る変数の長さより短い場合、その変数への割り当て時に空白で値が埋め込まれます。

KRIGHT、KLEFT および KTRIM 関数は、Unicode 空白文字を削除します。Unicode 文字のリストは表 13.180 (650 ページ)を参照してください。

例

この例では、UTF-8 エンコーディングを使用します。

この例では、漢字を使用します。

```
data _null_;
  part1='漢字の '; /* 1 */
  part2='テスト';
  hasblank=part1 || part2;
  noblank=ktrim(part1) || part2;
  put hasblank;
  put noblank;
run;
```

1 文字列は DBCS スペースで終わります。

上記のステートメントは、次の結果を生成します。

```
漢字の テスト
漢字のテスト
```

関連項目

関数:

- “KCOMPRESS 関数” (616 ページ)
- “KLEFT 関数” (649 ページ)
- “KRIGHT 関数” (669 ページ)

KTRUNCATE 関数

マルチバイト文字を壊すことなく、文字列をバイト単位で、指定した長さに切ります。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KTRUNCATE(*argument*, *number*, *length*)

必須引数

argument

SAS 文字式を指定します。

number

数値です。

length

整数です。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KTRUNCATE 関数では、完全な長さの *number*(倍精度浮動小数点数として保存)を使用して文字列を指定した長さに切り捨て、*length* で指定したより小さいバイト数にし、切り捨てられたバイトを 0 で埋め込みます。切り捨てとその後の展開は、最初に完全な長さより短い数を保存した結果を複製し、次にそれを読み込みます。

例

次の例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、漢字と絵文字を使用します。

```
data _null_;  
  x1=ktruncate('Renée Dupont',4); /* 1 */  
  x2=ktruncate('Renée Dupont',5);
```

```
x3=ktruncate('Renée Dupont',6);
y1=ktruncate('漢字のテスト',3,6); /* 2 */
y2=ktruncate('漢字のテスト',4,6);
y3=ktruncate('漢字のテスト',5,6);
put x1/x2/x3/y1/y2/y3;
run;
```

- 1 KTRUNCATE は 4 番目のバイトから始まる文字を切り捨てます。
- 2 KTRUNCATE は 3 番目のバイトから始まる 6 バイトを切り捨てます。

上記のステートメントは、次の結果を生成します。

```
Ren
René
Renée
字
字の
の
```

KUPCASE 関数

小文字のアルファベットを大文字に変換します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: “UPCASE 関数” ([SAS 関数と CALL ルーチン: リファレンス](#))を参照してください。

構文

KUPCASE(*argument*)

必須引数

argument

SAS 文字式を指定します。

詳細

KUPCASE 関数により、文字引数がコピーされ、小文字のアルファベットが大文字に変換され、変更された値が返されます。

アルファベットの定義は、SAS セッションのエンコーディングによって異なります。典型的な SAS セッションと、KUPCASE がさまざまなセッションのアルファベットを処理する方法を次に示します。

UTF-8 (SAS Unicode サーバー)

SAS セッションエンコーディングが UTF-8 の場合、Unicode 文字属性が適用されます。文字が文字として分類され、大文字に相当するものがある場合、KUPCASE は大文字を返します。それ以外の場合は、同じ文字が返されます。

DBCS (たとえば、中国語、日本語、韓国語)

KUPCASE は、DBCS SAS セッションで、英語のアルファベットの小文字を大文字に変換します。

SBCS(たとえば、英語またはヨーロッパ言語)

SAS セッションエンコーディングが SBCS の場合、TRANTAB オプションが更新され、大文字とそれに相当する小文字のマッピングテーブルが使用されます。大文字のマッピングテーブルは、*nnnn_ucs* の命名パターンを使用します。ここで、*nnnn* は、[SAS 各国語サポート\(NLS\): リファレンスガイド](#)に記載されている[短いエンコーディング名](#)と一致します。KUPCASE 関数は、これらのテーブルを使用して、小文字を、セッションエンコーディングでサポートされている大文字に変換します。

例

例 1

この例では、テキストに英語のアルファベットの小文字と小文字の ñ が混在しています。SAS セッションエンコーディングは、Windows コードページ 1252 です。これは、WLATIN1 です。KUPCASE は、Ñ を含むすべての文字を大文字で返すことに注意してください。この例は、SAS エンコーディングが SBCS であり、西ヨーロッパ文字をサポートしている場合の KUPCASE の結果を示しています。SAS エンコーディングが UTF-8 の場合、UTF-8 テキストに対して同じ結果が返されます。

```
options locale=en_US;
data one;
  x='niño';
  y=kupcase(x);

  put x=;
  put y=;
run;
```

上記のステートメントは、次の結果を生成します。

```
x=niño
y=NIÑO
```

例 2

この例では、漢字を使用します。

```
data _null_;  
result=kupcase('abc あいうえお');  
put result=;  
run;
```

上記のステートメントは、次の結果を生成します。

```
result=ABC あいうえお
```

KUPDATE 関数

文字値の内容を置換します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KUPDATE(*argument,position,n*<, *characters-to-replace*>)

KUPDATE(*argument,position*,<*n*>, *characters-to-replace*)

必須引数

argument

文字変数または文字列を指定します。

position

文字の開始位置となる数値式を指定します。

n

置き換える部分文字列の長さを示す数値式を指定します。

制限事項 *n* は、*argument* 内で、*position* より後に残った式の長さ以下である必要があります。

n の指定はオプションですが、この関数で *n* と *characters-to-replace* を両方とも省略することはできません。

ヒント *n* を省略すると、*characters-to-replace* のすべての文字を使用して *argument* の値を置き換えます。

characters-to-replace

argument の内容を置き換える文字式を指定します。

制限事項 *characters-to-replace* の指定はオプションですが、この関数で *characters-to-replace* と *n* を両方とも省略することはできません。

ヒント 文字のリテラル文字列を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ)を参照してください。

KUPDATE 関数は、*argument* の値を、*position* で指定された開始文字以降で、*characters-to-replace* の式で置き換えます。

KUPDATE 関数が事前に長さを割り当てられていない変数に値を返す場合、デフォルトで変数の長さは 200 に割り当てられます。

SQL プロシジャまたはプロシジャの WHERE 句では、KUPDATE 関数によって返される単語の最大長は 200 文字になります。

比較

KUPDATES 関数は、第 3 引数 *n* に基づいて対象文字列 *argument* を更新します。KUPDATE 関数は、第 3 引数 *n* 値に関係なく、最後の引数 *characters-to-replace* の長さを使用して対象文字列 *argument* を更新します。

KUPDATE と KUPDATES の例を次に示します。

```
data _null_;
  x = 'abcde';
  y = kupdate(x, 1, 4, '12');
  z = kupdates(x, 1, 4, '12');
  put y;
  put z;
run;
```

KUPDATE は abcd を 12 と置き換えます。KUPDATES は、abcd を 12 と空白埋め込みで置き換えます。

```
y=12e
z=12 e
```

例

例 1

次の例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、日本語文字と絵文字を使用します。

```
data _null_;
  x1=kupdate('Renée□Dupont',1,2,'Yè'); /* 1 */
  x2=kupdate(x1,1,2,'abcde');
  x3=kupdate(x1,1,3);
  x4=kupdate(x1,3,'π!€');
  y1=kupdate('感じのテスト',1,2,'漢字');
  y2=kupdate(y1,1,2,'kanji');
  y3=kupdate(y1,1,3);
  y4=kupdate(y1,3,'かんじ');
  put x1/x2/x3/x4/y1/y2/y3/y4;
run;
```

- 1 KUPDATE は最初の 2 文字を「Yè」に置き換えます。つまり、「Re」は「Yè」に置き換えられます。

上記のステートメントは、次の結果を生成します。

```
Yènée Dupont
abcdeenée Dupont
ée Dupont
Yèπ!€
漢字のテスト
kanji のテスト
テスト
漢字かんじ
```

例 2

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、日本語文字と絵文字を使用します。

この例では、KUPDATE と KUPDATES の違いを示します。

```
data _null_;
  x1='Renée□Dupont';
  x2=kupdate(x1,1,2,'abcde');
  x3=kupdates(x1,1,2,'abcde');
  y1='感じのテスト';
  y2=kupdate(y1,1,2,'kanji');
  y3=kupdates(y1,1,2,'ka');
  put x2/x3/y2/y3;
run;
```

上記のステートメントは、次の結果を生成します。

```

abcdenée Dupont
abnée Dupont
kanji のテスト
ka のテスト

```

関連項目

関数:

- [“KUPDATES 関数” \(699 ページ\)](#)
- [“KUPDATEB 関数” \(697 ページ\)](#)

KUPDATEB 関数

引数内の文字値のバイト位置に従って、文字値の内容を置換します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KUPDATEB(*argument*, *position*, *n*<, *characters-to-replace*>)

KUPDATEB(*argument*, *position* <, *n*> , *characters-to-replace*)

必須引数

argument

文字変数を指定します。

position

開始文字位置をバイト単位で指定します。

n

バイト単位に置き換えられる部分文字列の長さを指定します。

制限事項 *n* は、*argument* 内で、*position* より後に残った式の長さ(バイト)以下である必要があります。

n の指定はオプションですが、この関数で *n* と *characters-to-replace* を両方とも省略することはできません。

ヒント *n* を省略すると、*characters-to-replace* のすべての文字を使用して *argument* の値を置き換えます。

characters-to-replace

argument の内容を置き換える文字式を指定します。

制限事項 *characters-to-replace* の指定はオプションですが、この関数で *characters-to-replace* と *n* を両方とも省略することはできません。

ヒント 文字のリテラル文字列を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ) を参照してください。

KUPDATEB 関数は、*argument* の値を *characters-to-replace* の式で置き換えます。KUPDATEB は、*position* で指定したバイト単位で開始する *n* バイト単位を置き換えます。

KUPDATEB 関数が事前に長さを割り当てられていない変数に値を返す場合、デフォルトで変数の長さは 200 に割り当てられます。

SQL プロシジャまたはプロシジャの WHERE 句では、KUPDATEB 関数によって返される単語の最大長は 200 文字になります。

例

この例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、漢字と絵文字を使用します。

```
data _null_;
  x1=kupdateb('Renée Dupont',1,2,'Yr'); /* 1 */
  x2=kupdateb(x1,1,4,'abcde');
  x3=kupdateb(x1,1,5);
  x4=kupdateb(x1,3,'π!€');
  y1=kupdateb('感じのテスト',1,6,'漢字');
  y2=kupdateb(y1,1,6,'kanji');
  y3=kupdateb(y1,1,6);
  y4=kupdateb(y1,6,'かんじ');
  put x1/x2/x3/x4/y1/y2/y3/y4;
run;
```

- 1 KUPDATEB は文字列の最初の 2 バイトを置き換えます。「Re」は「Yr」に置き換えられます。

上記のステートメントは、次の結果を生成します。


```
Yrnée Dupont
abcd e Dupont
e Dupont
Yrn!e
漢字のテスト
kanji のテスト
のテスト
漢 かんじ
```

関連項目

関数:

- [“KUPDATE 関数” \(694 ページ\)](#)

KUPDATES 関数

文字値の内容を置換します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KUPDATES(*argument*,*position*,*n*<, *characters-to-replace*>)

KUPDATES(*argument*,*position*<,*n*> , *characters-to-replace*)

必須引数

argument

文字変数を指定します。

position

文字の開始位置となる数値式を指定します。

オプション引数

n

置き換える部分文字列の長さを示す数値式を指定します。

制限事項 *n* は、*argument* 内で、*position* より後に残った式の長さ以下である必要があります。

n の指定はオプションですが、この関数で n と *characters-to-replace* を両方とも省略することはできません。

ヒント n を省略すると、*characters-to-replace* のすべての文字を使用して *argument* の値を置き換えます。

characters-to-replace

argument の内容を置き換える文字式を指定します。

制限事項 *characters-to-replace* の指定はオプションですが、この関数で *characters-to-replace* と n を両方とも省略することはできません。

ヒント 文字のリテラル文字列を引用符で囲みます。

詳細

制限と詳しい情報については、“[SAS 文字列関数の国際化の互換性](#)” (555 ページ) を参照してください。

KUPDATES 関数は *argument* の値を *characters-to-replace* の式で置き換えます。KUPDATES は、*position* で指定された開始文字から n 文字分を置き換えます。

KUPDATES 関数が事前に長さを割り当てられていない変数に値を返す場合、デフォルトで変数の長さは 200 に割り当てられます。

SQL プロシジャまたはプロシジャの WHERE 句では、KUPDATES 関数によって返される単語の最大長は 200 文字になります。

比較

KUPDATES 関数は、第 3 引数 n に基づいて対象文字列 *argument* を更新します。KUPDATE 関数は、第 3 引数 n 値に関係なく、最後の引数 *characters-to-replace* の長さを使用して対象文字列 *argument* を更新します。

KUPDATE と KUPDATES の例を次に示します。

```
data _null_;
  x = 'abcde';
  y = kupdate(x, 1, 4, '12');
  z = kupdates(x, 1, 2, '12345');
  put y;
  put z;
run;
```

KUPDATE は abcd を 12 と置き換えます。KUPDATES は、abcd を 12 と空白埋め込みで置き換えます。

```
y=12e
y=12 e
```

例

例 1

これらの例では、UTF-8 エンコーディングを使用します。

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、漢字と絵文字を使用します。

```
data _null_;
  x1=kupdates('Renée Dupont',1,2,'Yè'); /* 1 */
  x2=kupdates(x1,1,2,'abcde');
  x3=kupdates(x1,1,3);
  x4=kupdates(x1,3,'π!€');
  y1=kupdates('感じのテスト',1,2,'漢字');
  y2=kupdates(y1,1,2,'kanji');
  y3=kupdates(y1,1,3);
  y4=kupdates(y1,3,'かんじ');
  put x1/x2/x3/x4/y1/y2/y3/y4;
run;
```

- 1 KUPDATES は文字列の最初の 2 文字を置き換えます。「RE」は「Yè」に置き換えられます。

上記のステートメントは、次の結果を生成します。

```
Yènée Dupont
abnée Dupont
ée Dupont
Yèπ!€
漢字のテスト
ka のテスト
テスト
漢字かんじ
```

例 2

次に、サポート代理ペア文字のサポート方法の例を示します。HTML 形式で表示されます。この例では、漢字と絵文字を使用します。

次の例では、KUPDATE と KUPDATES の違いを示します。

```
data _null_;
  x1='Renée Dupont';
  x2=kupdate(x1,1,2,'abcde');
  x3=kupdates(x1,1,2,'abcde');
  y1='感じのテスト';
  y2=kupdate(y1,1,2,'kanji');
  y3=kupdates(y1,1,2,'kanji');
  put x2/x3/y2/y3;
run;
```

上記のステートメントは、次の結果を生成します。

```

abcdenée Dupont
abnée Dupont
kanji のテスト
ka のテスト

```

関連項目

関数:

- [“KUPDATE 関数” \(694 ページ\)](#)

KVERIFY 関数

式に固有の最初の文字（文字ベースの値）の位置を返します。

カテゴリ: CAS

文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

ヒント: [“VERIFY 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)を参照してください。

構文

KVERIFY(*source*, *excerpt-1*, ...*excerpt-n*)

必須引数

source

SAS 文字式を指定します。

excerpt-1, ...*excerpt-n*

SAS 文字式を指定します。

ヒント 文字のリテラル文字列を引用符で囲みます。

複数の excerpt を指定する場合は、カンマで区切ります。

詳細

制限と詳しい情報については、[“SAS 文字列関数の国際化の互換性” \(555 ページ\)](#)を参照してください。

KVERIFY 関数は、いずれの *excerpt* 内にも存在しない文字のうち、最初の文字の *source* 内での位置を返します。KVERIFY は、*source* 内のすべての文字が、少なくとも 1 つの *excerpt* 内にあることを確認すると、0 を返します。

例

例 1

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;
  check='»é';
  result=kverify('ééé»ééABC', check); /* 1 */
  put result=;
run;
```

- 1 KVERIFY は文字「A」の位置を指定します。*excerpt* 値は「»é」なので、KVERIFY は「»é」ではない最初の文字を検索します。

上記のステートメントは、次の結果を生成します。

```
result=7
```

例 2

この例では、漢字を使用します。

```
data _null_;
  input grade :$3. @@;
  check='良否';
  x=kverify(grade, check);
  if x gt 0 then put 'Invalid grade value: ' grade;
  cards;
  良否良否良否可
  ;
```

上記のステートメントは、次の結果を生成します。

```
Invalid grade value: 可
```

KVERIFYB 関数

式に固有の最初の文字（バイトベースの値）の位置を返します。

カテゴリ: 文字操作

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

KVERIFYB(*source*, *excerpt-1*, <*excerpt-n*>)

必須引数

source

SAS 文字式を指定します。

excerpt

SAS 文字式を指定します。

ヒント 複数の *excerpt* を指定する場合は、カンマで区切ります。

詳細

KVERIFYB は、いずれの *excerpt* 内にも存在しない文字のうち、最初の文字の *source* 内での位置を返します。KVERIFYB は、*source* 内のすべての文字が、少なくとも 1 つの *excerpt* 内にあることを確認すると、0 を返します。

比較

KVERIFYB は、バイトベースの値を返します。KVERIFY は、文字ベースの値を返します。SBCS 文字列、(たとえば、wlatin1 エンコーディングの)を処理する場合、これらの関数は同一の結果を返します。一方、DBCS セッション(たとえば、EUC-JP または SHIFT-JIS エンコーディング)では、ほとんどの CJK 文字は 2 バイト幅です。このため、これらの関数は異なる結果を返します。

例

次の例では、UTF-8 エンコーディングを使用します。

```
data _null_;  
  check='»é';  
  result=kverifyb('ééé»ééABC', check); /* 1 */  
  put result=;  
run;
```

- 1 KVERIFYB は文字「A」のバイト位置を指定します。*excerpt* 値は「»é」なので、KVERIFYB は「»é」ではない最初の文字を検索します。

上記のステートメントは、次の結果を生成します。

```
result=13
```

NLDATE 関数

SAS 日付値を、日付のディスクリプターを使用して指定したロケールの日付値に変換します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

NLDATE(*date*,*descriptor*)

必須引数

date

SAS 日付値を指定します。

descriptor

出力での日付と時間のフォーマット方法を指定する、変数または式です。次のディスクリプターでは、大文字と小文字が区別されます。

#

先頭の 0 を出力結果から削除します。

%%

%文字を指定します。

%a

省略形の曜日のディスクリプターを指定します。省略形の曜日のディスクリプターの範囲は、Mon から Sun です。

%A

完全な曜日のディスクリプターを指定します。完全な曜日のディスクリプターの範囲は、Monday から Sunday です。

%b

省略形の月のディスクリプターを指定します。省略形の月のディスクリプターの範囲は、Jan から Dec です。

%B

完全な月のディスクリプターを指定します。完全な月のディスクリプターの範囲は、January から December です。

%C

完全な月のディスクリプターを指定し、空白埋め込みを使用します。完全な月のディスクリプターの範囲は、January から December です。

%d

日付のディスクリプターを指定し、0 埋め込みを使用します。日付修飾子の範囲は、01 から 31 です。

%e

日付のディスクリプターを指定し、空白埋め込みを使用します。日付のディスクリプターの範囲は、01 から 31 です。

%F

完全な曜日のディスクリプターを指定し、空白埋め込みを使用します。完全な曜日のディスクリプターの範囲は、Monday から Sunday です。

%j

通算日のディスクリプターを 10 進数で指定し、先頭に 0 を使用します。通算日のディスクリプターの範囲は、1 から 366 です。

%m

月のディスクリプターを指定し、0 埋め込みを使用します。月のディスクリプターの範囲は、01 から 12 です。

%o

月のディスクリプターを指定します。月のディスクリプターの範囲は 1 から 12 で、空白埋め込みを使用します。

%u

月曜日から日曜日を表す 1 から 7 の範囲の番号で、曜日のディスクリプターを指定します。

%U

年内の週数を使用してディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します(日曜日を週の開始日と見なします)。週数の値は、0 から 53 の範囲の 10 進数で表し、先頭に 0 を使用します。最大値は 53 です。

%V

ディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します。週数の値は、01 から 53 の範囲の 10 進数で表し、先頭に 0 を使用します。最大値は 53 です。各週は月曜日から始まり、年の第 1 週は 1 月 4 日と年の最初の木曜日の両方を含む週です。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。

%w

日曜日から土曜日を表す 0 から 6 の範囲の番号で、曜日のディスクリプターを指定します。

%W

年内の週数を使用してディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します(月曜日を週の開始日と見なしま

す)。週数の値は、0 から 53 の範囲の 10 進数で表し、先頭に 0 を使用します。最大値は 53 です。

%y
年(2 桁)の修飾子を指定します。年のディスクリプターの範囲は、00 から 99 です。

%Y
年(4 桁)のディスクリプターを指定します。年のディスクリプターの範囲は、1970 から 2069 です。

詳細

NLDATE 関数では、SAS 日付値を、日付のディスクリプターを使用して指定したロケールの日付値に変換します。

例

SAS 日付値から作成されたログファイル名の例を次に示します。

ステートメント	結果
options locale=English_Unitedstates; logfile=nldate('24Feb2003'd,'%B-%d.log'); put logfile;	February-24.log
options locale=German_Germany; logfile=nldate('24Feb2003'd,'%B-%d.log'); put logfile;	Februar-24.log

SAS 日付値から作成された曜日名の例を次に示します。

ステートメント	結果
	----+----1----+
options locale=English_unitedstates; weekname=nldate('24Feb2003'd,'%A'); put weekname;	Monday
options locale=German_Germany; weekname=nldate('24Feb2003'd,'%A'); put weekname;	Montag

関連項目

出力形式:

- [“NLDATEw. 出力形式” \(190 ページ\)](#)

NLDATEM 関数

SAS 日時値を、日時形式のディスクリプターを使用して指定したロケールの時間値に変換します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

NLDATEM(*datetime*,*descriptor*)

必須引数

datetime

SAS 日時値を指定します。

descriptor

出力での日付と時間のフォーマット方法を指定する、変数または式です。次のディスクリプターでは、大文字と小文字が区別されます。

#

先頭の 0 を出力結果から削除します。

%%

%文字を指定します。

%a

省略形の曜日のディスクリプターを指定します。省略形の曜日のディスクリプターの範囲は、Mon から Sun です。

%A

完全な曜日のディスクリプターを指定します。完全な曜日のディスクリプターの範囲は、Monday から Sunday です。

%b

省略形の月のディスクリプターを指定します。省略形の月のディスクリプターの範囲は、Jan から Dec です。

%B

完全な月のディスクリプターを指定します。完全な月のディスクリプターの範囲は、January から December です。

%c

完全な月のディスクリプターを指定し、空白埋め込みを使用します。完全な月のディスクリプターの範囲は、January から December です。

%d

日付のディスクリプターを指定し、0 埋め込みを使用します。日付のディスクリプターの範囲は、01 から 31 です。

%e

日付のディスクリプターを指定し、空白埋め込みを使用します。日付のディスクリプターの範囲は、01 から 31 です。

%F

完全な曜日のディスクリプターを指定し、空白埋め込みを使用します。完全な曜日のディスクリプターの範囲は、Monday から Sunday です。

%H

24 時間表示の時間のディスクリプターを指定します。時間のディスクリプターの範囲は、00 から 23 です。

%I

12 時間表示の時間のディスクリプターを指定します。時間のディスクリプターの範囲は、01 から 12 です。

%j

通算日のディスクリプターを 10 進数で指定し、先頭に 0 を使用します。通算日のディスクリプターの範囲は、1 から 366 です。

%m

月のディスクリプターを指定し、0 埋め込みを使用します。月のディスクリプターの範囲は、01 から 12 です。

%M

分のディスクリプターを指定します。分のディスクリプターの範囲は、00 から 59 です。

%o

月のディスクリプターを指定し、空白埋め込みを使用します。月のディスクリプターの範囲は、1 から 12 です。

%p

午前または午後のディスクリプターを指定します。

%S

秒のディスクリプターを指定します。秒のディスクリプターの範囲は、00 から 59 です。

%u

月曜日から日曜日を表す 1 から 7 の範囲の番号で、曜日のディスクリプターを指定します。

%U

年内の週数を使用してディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します(日曜日を週の開始日と見なしません)。週数の値は、0 から 53 の範囲の 10 進数で表します。先頭に 0 を使用し、最大値は 53 です。

%V

ディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します。週数の値は、01 から 53 の範囲の 10 進数で表します。先頭に 0 を使用し、最大値は 53 です。各週は月曜日から始まり、年の第 1 週は 1 月 4 日と年の最初の木曜日の両方を含む週です。1 月の最初の月曜日が 2 日、3 日または 4 日の場合、それより前の日は前年の最後の週に組み込まれます。

%w

日曜日から土曜日を表す 0 から 6 の範囲の番号で、曜日のディスクリプターを指定します。

%W

年内の週数を使用してディスクリプター値を SAS 日付値として計算することで、通算週のディスクリプターを指定します(月曜日を週の開始日と見なします)。週数の値は、0 から 53 の範囲の 10 進数で表します。先頭に 0 を使用し、最大値は 53 です。

%y

年(2 桁)のディスクリプターを指定します。年のディスクリプターの範囲は、00 から 99 です。

%Y

年(4 桁)のディスクリプターを指定します。年のディスクリプターの範囲は、1970 から 2069 です。

詳細

NLDATM 関数では、SAS 日時値を、日時のディスクリプターを使用して指定したロケールの日時値に変換します。

例

SAS 日時値から作成された時間(午前または午後)の例を次に示します。

ステートメント	結果
	----+----1----+
options locale=English; time_ampm=nldatm('24Feb2003:12:39:43'dt,'%I%p'); put time_ampm;	12PM
options locale=German; time_ampm=nldatm('24Feb2003:12:39:43'dt,'%I%p'); put time_ampm;	12nachm

関連項目

出力形式:

- “NLDATMw. 出力形式” (222 ページ)

NLTIME 関数

SAS 時間値または SAS 日時値を、NLTIME ディスクリプターを使用して指定したロケールの時間値に変換します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

NLTIME(*time* | *datetime*, *descriptor*, *startpos*)

必須引数

time

SAS 時間値を指定します。

datetime

SAS 日時値を指定します。

descriptor

ディスクリプターの値を指定する、変数または式です。大文字または小文字で次のディスクリプターを入力できます。

#

先頭の 0 を出力結果から削除します。

%%

%文字を指定します。

%H

24 時間表示の時間のディスクリプターを指定します。時間のディスクリプターの範囲は、00 から 23 です。

%I

12 時間表示の時間のディスクリプターを指定します。時間のディスクリプターの範囲は、01 から 12 です。

%M

分の修飾子を指定します。分のディスクリプターの範囲は、00 から 59 です。

%P

午前または午後のディスクリプターを指定します。

%S

秒のディスクリプターを指定します。秒のディスクリプターの範囲は、00 から 59 です。

startpos

検索を開始する位置を指定し、検索の方向を指定する整数です。

詳細

NLTIME 関数では、SAS 時間値または SAS 日時値を、時間のディスクリプターを使用して指定したロケールの時間値に変換します。

例

SAS 時間から作成された午前または午後の時間の例を次に示します。

ステートメント	結果
	----+----1----+
options locale=English; time_ampm=nltime('12:39:43't,'%i%p'); put time_ampm;	12PM
options locale=German; time_ampm=nltime('12:39:43't,'%i%p'); put time_ampm;	12nachm

関連項目

出力形式:

- [“NLTIMEw. 出力形式” \(499 ページ\)](#)

SASMSG 関数

データセットからメッセージを返します。返されるメッセージは現在のロケールと指定されたキーに基づきます。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

SASMSG (*BASENAME*, *"KEY"*, <<*"QUOTE"* | *"DQUOTE"* | *"NOQUOTE"*>>
<, *"substitution 1"*, ..., *"substitution 7"*>>)

必須引数

BASENAME
メッセージを含むデータセットの名前

KEY
メッセージキー

注: 無効なキー名を指定すると、キー名が返されます。

QUOTE | *DQUOTE* | *NOQUOTE*
メッセージテキストと代入文字列に追加する引用符のタイプを指定します。

デフォルト DQUOTE

substitution
文字列代入 文字列代入の最大件数は 7 です。SAS メッセージデータセットは 7 ビット ASCII データセットでなければなりません。

詳細

SAS メッセージデータセットは 7 ビット ASCII データセットでなければなりません。xxxx'は、文字の Unicode 値の 10 進数表現です。

SASMSG 関数で使用するデータセットは、この関数の使用専用で作成してください。データセットには次の変数を含める必要があります。

#	変数名	タイプ	長さ	説明
1	ロケール	char	5	メッセージの言語
2	key	char	60	メッセージを認識するキー
3	lineno	num	5	メッセージのライン番号(逆順)
4	text	text	1200	メッセージのテキスト

データセットは次の変数で並べ替えている必要があります。 *locale*、*key*、および *lineno*。変数 *lineno* は降順でなければなりません。 *locale* と *key* の複合インデックスを定義する必要があります。並べ替えとインデックス付きのデータセットを作成するサンプルプログラムは次のようになります。

```
%let basename=MyProduct;
proc sort data=t.&basename;
  by locale key descending lineno;
run;
proc datasets lib=t
  memtype=data;
  modify &basename;
  index create indx=(LOCALE KEY);
run;
quit;
```

返されるメッセージは LOCALE システムオプションに基づきます。LOCALE オプションは *ll_rr* で表されます。ll は 2 文字の言語コードを表し、PR は 2 文字の地域コードを表します。一致するものがない場合、関数は言語のみの一致を探します。 *locale* と *key* のペアが見つからない場合、関数はデフォルトの英語(en)を使用します。英語(en)のキーが存在しない場合、キー名が返されます。

フォーマティングは変更することができます。フォーマットコード %s を使用して、文字列代入を使用できます。代入順序を変更する事ができます。英語以外の言語へのメッセージの翻訳では、代入順序の変更が必要な場合があります。引数番号 *nn* を指定することにより、フォーマット文字列内で順序を変えることができます。 *nn* は代入リストの引数番号になります。次の例では順序について説明します。

ステートメント	結果
<pre>msg = sasmsg ("nls.mymsg","IN_CD_LOG", "noquote","cat","dog"); IN_CD_LOGINFO = My %#1s. Your %#2s</pre>	<pre>msg= My cat. Your dog.</pre>
<pre>IN_CD_LOGINFO = My %#2s. Your %#1s</pre>	<pre>msg= My dog. Your cat.</pre>

SASMSG 関数は %SYSFUNC マクロ関数と一緒にオープンコードマクロで使用されます。

%SYSFUNC マクロで呼び出される関数に渡す引数は、二重引用符で囲まないとくたさい。 %SYSFUNC 外部で SASMSG 関数に渡される引数は、二重引用符で囲む必要があります。

SASMSG 関数が %SYSFUNC マクロ関数と一緒に使われる場合、返される文字列は %NRBQUOTE 関数を使用して囲まれます。

例

例 1

次の例で SASMSG のフォーマティング機能を説明します。

```
%macro demo_sasmsg;
  data _null_;
    msg = sasmsg("nls.mymsg", "IN_APW_SAVE_OK", "noquote");
    put msg=;
  run;
%mend demo_sasmsg;
```

SAS ステートメント	結果
options locale = en_US; %demo_sasmsg ;	msg=The Access Control key was successfully saved.
options locale = es_ES; %demo_sasmsg;	msg=La clave de control de acceso se ha guardado.
options locale = french_France; %demo_sasmsg;	msg=La clé de contrôle d'accès a bien été enregistrée.

例 2

次の例でオープンマクロの機能を説明します。

```
%MACRO PRT(loc,tb,key);
  option locale=&loc;
  %PUT %SYSFUNC(SASMSG(&tb,&key) );
%MEND PRT;
```

SAS ステートメント	結果
%PRT(en_US,&TABLEID,IN_EDIT)	"Edit"
%PRT(es_ES,&TABLEID,IN_EDIT)	"Editar"
%PRT(fr_FR,&TABLEID,IN_EDIT)	"Modifier"

SASMSGGL 関数

データセットからメッセージを返します。メッセージは、指定されたロケール値と指定されたキー値に基づきます。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

```
SASMSGGL("BASENAME", "KEY", "LOCALE", (, , , <"Q"|"D"|"N">
<, "substitution 1", ..., "substitution 6">>)
```

必須引数

BASENAME

メッセージを含むデータセットの名前

KEY

メッセージキー

.....
注: 無効なキー名を指定すると、キー名が返されます。
.....

LOCALE

posix ロケール値 (ll_RR).

QUOTE|DQUOTE|NOQUOTE

メッセージテキストと代入文字列に追加する引用符のタイプを指定します。

デフォルト DQUOTE

substitution

文字列代入 文字列代入の最大件数は 7 です。SAS メッセージデータセットは 6 ビット ASCII データセットでなければなりません。

詳細

SAS メッセージデータセットは 7 ビット ASCII データセットでなければなりません。7 ビット ASCII エンコーディングで表現できない文字は 'luxxxx' のような Unicode エスケープ形式で表現されます。xxxx は、文字の Unicode 値の 10 進数表現です。

SASMSG 関数で使用するデータセットは、この関数の使用専用で作成してください。データセットには次の変数を含める必要があります。

#	変数名	タイプ	長さ	説明
1	ロケール	char	5	メッセージの言語
2	key	char	60	メッセージを認識するキー
3	lineno	num	5	メッセージのライン番号（逆順）
4	text	text	1200	メッセージのテキスト

データセットは次の変数で並べ替えている必要があります。locale、key、および lineno。変数 lineno は降順でなければなりません。locale と key の複合インデックスを定義する必要があります。並べ替えとインデックス付きのデータセットを作成するサンプルプログラムは次のようになります。

```
%let basename=MyProduct;

proc sort data=t.&basename;
  by locale key descending lineno;
run;

proc datasets lib=t
  memtype=data;
  modify &basename;
  index create indx=(LOCALE KEY);
run;
quit;
```

返されるメッセージは LOCALE システムオプションに基づきます。LOCALE オプションは *ll_rr* で表されます。*ll* は 2 文字の言語コードを表し、*rr* は 2 文字の地域コードを表します。一致するものがない場合、関数は言語のみの一致を探します。locale と key のペアが見つからない場合、関数はデフォルトの英語(en)を使用します。英語(en)のキーが存在しない場合、キー名が返されます。

フォーマティングは変更することができます。フォーマットコード %s を使用して、文字列代入を使用できます。代入順序を変更する事ができます。英語以外の言語へのメッセージの翻訳では、代入順序の変更が必要な場合があります。引数番号 *nn* を指定することにより、フォーマット文字列内で順序を変えることができます。*nn* は代入リストの引数番号になります。次の例では順序の変更について説明します。

ステートメント	結果
msg = sasmsgl ("nls.mymsg","IN_CD_LOG","en_US","N", "cat","dog"); IN_CD_LOGINFO = My %#1s. Your %#2s	msg= My cat. Your dog.
IN_CD_LOGINFO = My %#2s. Your %#1s	msg= My dog. Your cat.

SASMSGGL 関数は%SYSFUNC マクロ関数と一緒にオープンコードマクロで使用できます。

%SYSFUNC マクロで呼び出される関数に渡す引数は、二重引用符で囲まないとくささい。%SYSFUNC 外部で SASMSGGL 関数に渡される引数は、二重引用符で囲む必要があります。

SASMSGGL 関数が%SYSFUNC マクロ関数と一緒に使われる場合、返される文字列は%NRBQUOTE 関数を使用して囲まれます。

例

例 1

次の例で SASMSGGL のフォーマティング機能を説明します。

ステートメント	結果
<code>sasmsggl("nls.mymsg", "IN_APW_SAVE_OK", "en_US", "n");</code>	The Access Control key was successfully saved.
<code>sasmsggl("nls.mymsg", "IN_APW_SAVE_OK", "es_ES", "n");</code>	La clave de control de acceso se ha guardado.
<code>sasmsggl("nls.mymsg", "IN_APW_SAVE_OK", "fr", "n");</code>	La clé de contrôle d'accès a bien été enregistrée.

例 2

次の例でオープンマクロの機能を説明します。

SAS ステートメント	結果
<code>%PUT %SYSFUNC(SASMSGGL(NLS.MYDS, IN_ASD_LABEL, en_US));</code>	"Edit"
<code>%PUT %SYSFUNC(SASMSGGL(NLS.MYDS, IN_ASD_LABEL, es_ES));</code>	"Editor"
<code>%PUT %SYSFUNC(SASMSGGL(NLS.MYDS, IN_ASD_LABEL, fr));</code>	"Modifier"

SETLOCALE 関数

現在の SAS ロケールにロケールキーを指定します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

SAS ロケールの設定

SETLOCALE (*[sas_locale](#)*)

単一ロケール要素のカスタマイズ

SETLOCALE (*[key,value](#)*)

単一ロケール要素のカスタマイズ

SETLOCALE (*[category_name](#), [sas_locale](#)*)

必須引数

[sas_locale](#)

SAS 名または Posix 名を使用して SAS ロケール名を指定します。ロケールの別名を指定する事もできます。

[key](#)

SAS ロケール要素キーを指定します。詳細セクションの要素キーのリストを参照してください。

[value](#)

ロケール要素の値を指定します。

[category_name](#)

カテゴリ名を指定します。

- LC_TIME
- LC_MONETARY
- LC_NUMERIC
- LC_ALL

詳細

次のロケール要素を変更することができます。*key* の値は *max length* の値より小さくなくてはなりません。*type* に次の値を指定できます。

- 0 文字列
- 1 符号なし整数。二重引用符を使用しなくてはなりません。

ロケール要素キー	最大長	タイプ	カテゴリ
DATESTYLE	3	0	
PAPERSIZE	8	0	
FTITLE	512	0	
FTEXT	512	0	
SIMFONT	512	0	
SORTSEQ	8	0	
MESSAGES	8	0	
FORMATNAME_DATE	512	0	
FORMATNAME_DATETIME	512	0	
FORMATNAME_TIME	512	0	
FORMATNAME_NUMERIC	512	0	
FORMATNAME_PERCENT	512	0	
FONT_SERIF	32	0	
FONT_SANSERIF	32	0	
FONT_CURSIVE	32	0	
FONT_FANTASY	32	0	
FONT_MONOSPACE	32	0	
BRUSH	32	0	
SIMPLEX	32	0	
COMPLEX	32	0	
SWISS	32	0	
ITALIC	32	0	
DATE_FORMAT	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
DATE_SHORT_FORMAT	512	0	LC_TIME
DATETIME_AMPM_FORMAT	512	0	LC_TIME
DATETIME_FORMAT	512	0	LC_TIME
DATETIME_SHORT_FORMAT	512	0	LC_TIME
DATETIME_WEEK_FORMAT	512	0	LC_TIME
DATETIME_WEEK_SHORT_FORMAT	512	0	LC_TIME
TIME_AMPM_FORMAT	512	0	LC_TIME
TIME_FORMAT	512	0	LC_TIME
DATE_WEEK_FORMAT	512	0	LC_TIME
DATE_WEEK_SHORT_FORMAT	512	0	LC_TIME
DATE_YYMM_FORMAT	512	0	LC_TIME
DATE_YYMM_SHORT_FORMAT	512	0	LC_TIME
DATE_MMDD_FORMAT	512	0	LC_TIME
DATE_MMDD_SHORT_FORMAT	512	0	LC_TIME
DATE_YEAR_FORMAT	512	0	LC_TIME
DATE_YEAR_SHORT_FORMAT	512	0	LC_TIME
DATE_YYQQ_FORMAT	512	0	LC_TIME
DATE_YYQQ_SHORT_FORMAT	512	0	LC_TIME
DATE_YYWW_FORMAT	512	0	LC_TIME
DATE_YYWW_SHORT_FORMAT	512	0	LC_TIME
DATE_SEP	8	0	LC_TIME
ABMON01	512	0	LC_TIME
ABMON02	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
ABMON03	512	0	LC_TIME
ABMON04	512	0	LC_TIME
ABMON05	512	0	LC_TIME
ABMON06	512	0	LC_TIME
ABMON07	512	0	LC_TIME
ABMON08	512	0	LC_TIME
ABMON09	512	0	LC_TIME
ABMON10	512	0	LC_TIME
ABMON11	512	0	LC_TIME
ABMON12	512	0	LC_TIME
MON01	512	0	LC_TIME
MON02	512	0	LC_TIME
MON03	512	0	LC_TIME
MON04	512	0	LC_TIME
MON05	512	0	LC_TIME
MON06	512	0	LC_TIME
MON07	512	0	LC_TIME
MON08	512	0	LC_TIME
MON09	512	0	LC_TIME
MON10	512	0	LC_TIME
MON11	512	0	LC_TIME
MON12	512	0	LC_TIME
ABDAY1	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
ABDAY2	512	0	LC_TIME
ABDAY3	512	0	LC_TIME
ABDAY4	512	0	LC_TIME
ABDAY5	512	0	LC_TIME
ABDAY6	512	0	LC_TIME
ABDAY7	512	0	LC_TIME
DAY1	512	0	LC_TIME
DAY2	512	0	LC_TIME
DAY3	512	0	LC_TIME
DAY4	512	0	LC_TIME
DAY5	512	0	LC_TIME
DAY6	512	0	LC_TIME
DAY7	512	0	LC_TIME
AM	512	0	LC_TIME
PM	512	0	LC_TIME
ABQTR1	512	0	LC_TIME
ABQTR2	512	0	LC_TIME
ABQTR3	512	0	LC_TIME
ABQTR4	512	0	LC_TIME
QTR1	512	0	LC_TIME
QTR2	512	0	LC_TIME
QTR3	512	0	LC_TIME
QTR4	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
INT_CURRENCY_SYMBOL	3	0	LC_MONETARY
CURRENCY_SYMBOL	32	0	LC_MONETARY
MON_DECIMAL_POINT	8	0	LC_MONETARY
MON_THOUSANDS_SEP	8	0	LC_MONETARY
MON_GROUPING	3	1	LC_MONETARY
MON_POSITIVE_SIGN	8	0	LC_MONETARY
MON_NEGATIVE_SIGN	8	0	LC_MONETARY
MON_INT_FRAC_DIGITS	3	1	LC_MONETARY
MON_FRAC_DIGITS	3	1	LC_MONETARY
MON_P_CS_PRECEDES	3	1	LC_MONETARY
MON_P_SEP_BY_SPACE	3	1	LC_MONETARY
MON_P_SIGN_POSN	3	1	LC_MONETARY
MON_N_SIGN_POSN	3	1	LC_MONETARY
DECIMAL_POINT	1	0	LC_NUMERIC
THOUSANDS_SEP	1	0	LC_NUMERIC
GROUPING	3	1	LC_NUMERIC
POSITIVE_SIGN	8	0	LC_NUMERIC
NEGATIVE_SIGN	8	0	LC_NUMERIC
P_CS_PRECEDES	3	1	LC_NUMERIC
P_SEP_BY_SPACE	3	1	LC_NUMERIC
N_CS_PRECEDES	3	1	LC_NUMERIC
P_SEP_BY_SPACE	3	1	LC_NUMERIC
N_CS_PRECEDES	3	1	LC_NUMERIC

ロケール要素キー	最大長	タイプ	カテゴリ
N_SEP_BY_SPACE	3	1	LC_NUMERIC
P_SIGN_POSN	3	1	LC_NUMERIC
N_SIGN_POSN	3	1	LC_NUMERIC
HEIGHT	3	1	
WIDTH	3	1	

例

例 1

次の例では、SETLOCALE 関数は日本語(jp_JP)のロケールを指定します。SETLOCALE 関数は以前のロケールを返します。この例では、以前のロケールは English_United States です。

ステートメント	結果
<pre>data _null_; x=setlocale('ja_JP'); put x=; run;</pre>	<pre>x=English_UnitedStates</pre>

例 2

次の例では、SETLOCALE 関数は要素値が変更されているロケール名を返します。

ステートメント	結果
<pre>data _null_; x = setlocale("LC_MONETARY", 'zh_CN'); put x=; run;</pre>	<pre>x=japanese_japan</pre>

例 3

次の例では、SETLOCALE 関数は指定された DATE_YEAR_FORMAT キーの値を変更します。

ステートメント	結果
<pre>data _null; x=setlocale('DATE_YEAR_FORMAT', '¥%Y'); put x=; run;</pre>	<pre>x=%Y¥</pre>

SORTKEY 関数

言語並べ替えキーを作成します。

カテゴリ: ロケール

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

sortKey(*string*, <*locale*, *strength*, *case_order*, *numeric_order*, *collation_order*>)

必須引数

string
文字式

locale
POSIX 名(ja_JP)形式でロケール名を指定します。ロケール名と POSIX 値のリストについては、[表 21.1](#) を参照してください。

strength
強度の値は、照合レベルに関連付けられます。5 つの照合レベル値があります。次の表に、5 つのレベルの情報を示します。強度のデフォルト値は、ロケールに関連付けられます。

値	照合の種類	説明
PRIMARY または P	PRIMARY では、基本文字間の差異を指定します("a" < "b"など)。	この差異は最強です。たとえば、辞書は基本文字により異なるセクションに分割されません。

SECONDARY または S	文字のアクセント記号が第 2 差異となります("as" < "às" < "at" など)。	文字間のその他の差異も、言語に応じて第 2 差異と考慮される場合があります。文字列内に第 1 差異がある場合は、第 2 差異は無視されます。
TERTIARY または T	大文字と小文字の差異は、第 3 レベルで区別されます("ao" < "Ao" < "aò" など)。	例として、通常の仮名と小文字の仮名の違いがあります。文字列内に第 1 または第 2 差異がある場合は、第 3 差異は無視されます。
QUATERNARY または Q	レベル 1 から 3 で句読点が無視される場合、句読点の有無で単語を区別するために追加レベルが使用されます("ab" < "a-b" < "aB" など)。	第 1、第 2 または第 3 差異がある場合は、この差異は無視されます。句読点の無視が必須の場合や日本語テキストを処理する場合は、第 4 レベルを使用する必要があります。
IDENTICAL または I	他のすべてのレベルが等しい場合、最終的な決定をするために同一レベルが使用されます。各文字列の正規化形式 D の Unicode コードポイント値がこのレベルで比較され、レベル 1 から 4 で違いがないことが確認されます。	たとえば、ヘブライ語の詠唱マークのみがこのレベルで区別されます。2 つの文字列間でコードポイント値のみが異なることはほとんどないため、このレベルは必要最小限に使用します。

case order

大文字と小文字で並べ替えます。この引数は、TERTIARY、QUATERNARY または IDENTICAL レベルでのみ有効です。次の表に、case order 引数の値と情報を示します。

値	説明
UPPER または U	大文字、小文字の順序で並べ替えます。
LOWER または L	小文字、大文字の順序で並べ替えます。

numeric order

数を表す文字のかわりに数値で番号を並べ替えます。

値	説明
NUMERIC または N	数値で番号(整数)を並べ替えます。たとえば、"8 Main St."は"45 Main St."より前になります。

collation order

collation の値には、Phonebook と Traditional という 2 種類があります。照合値を選択しない場合、ユーザーのロケールのデフォルト照合が選択されます。次の表に、詳細を示します。

値	説明
PHONEBOOK または P	文字の電話帳スタイルの並べ替えを指定します。 PHONEBOOK はドイツ語でのみ選択します。
TRADITIONAL または T	文字の従来のスタイルの並べ替えを指定します。 TRADITIONAL はスペイン語でのみ選択します。

詳細

SORTKEY 関数により、データの言語並べ替えキーが作成されます。少なくとも 1 つの引数を入力する必要があります。キーを受け取る変数の長さが十分でない場合、データが切り捨てられて警告が表示されます。

locale

ロケール値には POSIX 名(LL_RR)を使用します。LL は 2 文字の言語コードを表し、RR は 2 文字の地域コードを表します。たとえば、en_US は英語で、米国の POSIX 名です。en は英語を表し、US は米国を表します。ロケール値が指定されていない場合、セッションロケールが使用されます。

strength

strength 引数は、アクセント記号または大文字、小文字が照合または一致テキストに影響するかどうかを決定します。strength に値が指定されていない場合、ロケールにより値が決まります。次の値を strength に指定できます。

PRIMARY

基本文字が含まれます。たとえば、文字 A、a、Å のすべてが同様に処理されます。

SECONDARY

データを PRIMARY と同様に処理し、さらにアクセント記号を処理します。文字 A と a は同等に処理され、Å はアクセント記号付き文字として処理されます。

TERTIARY

データを SECONDARY と同様に処理し、さらに大文字、小文字を処理します。たとえば、A、a、Å はすべて異なる文字として処理されます。

QUATERNARY

データを TERTIARY と同様に処理し、さらに句読点を処理します。

IDENTICAL

データを QUATERNARY と同様に処理し、さらにコードポイントを処理します。

case order

大文字または小文字でデータを並べ替えるように指定します。次の表に、UPPER 値または LOWER 値を指定した例を示します。

UPPER	LOWER
Aztec	aztec
aztec	Aztec
Mars	mars
mars	Mars

collation order

照合順序値 PHONEBOOK は、ロケールがドイツ語でない場合は無視されます。
照合順序値 TRADITIONAL は、ロケールがスペイン語でない場合は無視されます。
他のロケールでは警告メッセージが表示されます。

例

SORTKEY 関数の例を次に示します。

```
data mydata;
input name & $15.;
datalines;
côte
ôte
coté
45 Main St.
côte
zoo
cote
8 Main St.
;
run ;

data list ;
set mydata;
format key_fr $hex40.;
key_fr = sortkey(name,"fr_CA",,'N');
run ;

PROC SQL ;
create table my_fr_CA_sort as
SELECT * FROM list
ORDER BY key_fr;
QUIT ;
```

TRANTAB 関数

指定された変換テーブルを使用してデータをトランスコードします。

カテゴリ: 文字

制限事項: この関数は I18N レベル 0 ステータスがアサインされており、SBCS データ用に設計されています。この関数は DBCS または MBCS の処理に使用できません。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

TRANTAB(*string*, *trantab_name*)

必須引数

string

トランスコードされる入力データ。

trantab_name

変換テーブル。

詳細

TRANTAB 関数では、変換テーブルを使用して 1 つの内部表示形式から別の内部表示形式に文字を再マッピングすることによって、データ文字列をトランスコードします。入力文字列のデータのエンコーディングは、変換テーブルのテーブル 1 のエンコーディングと一致する必要があります。TRANTAB 関数では、テーブル 1 を使用してエンコーディングからデータを再マッピングします。

変換テーブルは各国語の要件をサポートするために SAS 6 で導入されました。SAS 8.2 では、変換テーブルの直接使用より優れた、LOCALE=および ENCODING=システムオプションが導入されました。SAS 9 では、変換テーブルは、外部ファイルのトランスコーディングにのみに使用されます。セッションエンコーディング間で、SAS ファイルに直接トランスコーディングされます。SAS 9.2 では、TRANTAB 関数が下位互換性を維持するためにサポートされています。

これ以降のリリースでは、LOCALE=システムオプションをお勧めします。

注意

TRANTAB 関数は、経験を積んだ SAS ユーザーのみが使用してください。

例

次の例では、Latin2 でエンコードされたデータを大文字の Latin2 エンコーディングにトランスコードする変換テーブルを使用します。

ステートメント	結果
teststrg=trantab('testing','lat2_ucs'); put teststrg;	TESTING

関連項目

プロシジャ:

- [20 章, “TRANTAB プロシジャ,” \(1067 ページ\)](#)

TZONEID 関数

現在のタイムゾーン ID を返します。

カテゴリ: 日付と時間

別名: TZID

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。
この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONEID(<*time-zone-id*>)

オプション引数

time-zone-id
SAS で定義されている region/area の値を指定します。ゾーン ID の指定時、タイムゾーン名と夏時間ルールに基づきタイムゾーンが決定されます。

詳細

TZONEID 関数は、TIMEZONE=オプションが空白の場合か、またはユーザー定義のタイムゾーンが指定されている場合に空白値を返します。

TZONEID 関数は、タイムゾーン ID を検証します。タイムゾーン ID が指定されている場合、その ID が有効であればタイムゾーン ID を返し、無効であれば空白値を返します。

例

最初の例では、TIMEZONE オプションが JST に設定されています。2 番目の例では、TIMEZONE が空白値に設定されています。3 番目の例では、TIMEZONE がユーザー指定のタイムゾーンに設定されています。4 番目の例では、有効なタイムゾーン ID と無効なタイムゾーン ID が表示されます。

ステートメント	結果
<pre>options timezone=jst; data _null_; tzid=tzoneid(); put tzid=; run;</pre>	<pre>tzid=ASIA/TOKYO</pre>
<pre>options timezone=""; data _null_; tzid=tzoneid(); put tzid=; run;</pre>	<pre>tzid=</pre>
<pre>options timezone='xxx-12'; /* user defined timezone */ data _null_; tzid=tzoneid(); put tzid=; run;</pre>	<pre>tzid=ETC/GMT-12</pre>
<pre>data null; name_valid=tzoneid('asia/tokyo'); name_invalid=tzoneid('Milky Way'); put name_valid =; put name_invalid=; run;</pre>	<pre>name_valid=ASIA/TOKYO name_invalid=</pre>

TZONENAME 関数

現在の標準時間または夏時間のタイムゾーン名を返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONENAME()

TZONENAME(<*time-zone-id*,*datetime*>)

オプション引数

time-zone-id

SAS で定義されている *region/area* の値を指定します。ゾーン ID の指定時、タイムゾーン名と夏時間ルールに基づきタイムゾーンが決定されます。

参照項目: タイムゾーン ID のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名” \(1145 ページ\)](#)を参照してください。

datetime

SAS 日時値を指定します。

詳細

TIMEZONE=オプションおよび time-zone-id 引数が空白の場合、TZONENAME 関数は空白値を返します。

TZONENAME は、指定されたタイムゾーンと日時に基づいてタイムゾーン名を返します。SAS 日時が指定されていない場合、現在の日時が使用されます。time-zone-id が指定されていない場合、TIMEZONE=オプションで指定された timezone ID が使用されます。

例

最初の例では、TIMEZONE オプションが空白値に設定されています。2 番目の例では、TIMEZONE がタイムゾーン名 JST に設定されています。3 番目の例では、TIMEZONE がユーザー指定のタイムゾーンに設定されています。4 番目の例では、TIMEZONE がタイムゾーン ID に設定されています。

ステートメント	結果
options tz=""; data _null; tzname=tzname();	tzname=

ステートメント	結果
<pre>put tzname =; run;</pre>	
<pre>options tz='jst'; data _null_; tzname=tzonenname(); put tzname =; run;</pre>	tzname=JST
<pre>options tz='xxx-12'; data _null_; tzname=tzonenname(); put tzname =; run;</pre>	tzname=XXX
<pre>options tz='America/Chicago'; data _null_; tzname=tzonenname('01SEP2014:01:01:01'dt); put tzname =; run;</pre>	tzname=CDT

TZONEOFF 関数

ユーザーのタイムゾーンオフセットを返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。
この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONEOFF()
TZONEOFF(<time-zone-id,datetime>)

オプション引数

time-zone-id
SAS で定義されている *region/area* の値を指定します。タイムゾーン ID の指定時、タイムゾーン名と夏時間ルールに基づきタイムゾーンが決定されます。

参照項目: タイムゾーン ID のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名” \(1145 ページ\)](#)を参照してください。

datetime
SAS 日時値を指定します。

詳細

引数が指定されていない場合は、TZONEOFF 関数は、指定された TIMEZONE オプションのタイムゾーンオフセットを返します。引数としてタイムゾーン ID を指定した TZONEOFF(time-zone-id)関数は、指定したタイムゾーン ID に対するタイムゾーンオフセットを返します。タイムゾーン ID 名を使用した TZONEOFF 関数は、指定したタイムゾーン名に対するタイムゾーンオフセットを返します。タイムゾーン ID はロケール依存でないため、タイムゾーン ID の使用をお勧めします。

SASDTM が指定されていない場合は、TZONEOFF は現在のタイムゾーンオフセットを返します。SASDTM が指定されている場合は、指定された時間値のローカル時間を取得するためのオフセットを返します。

例

最初の例では、引数がないため、TZONEOFF 関数は現在の SAS セッションに対するオフセットを返します。2 番目の例では、指定したタイムゾーン ID に基づいてオフセットを返します。3 番目の例では、指定したタイムゾーン ID および指定の日付と時間に基づいてオフセットを返します。4 番目の例では、TIMEZONE オプション値と指定の日付に基づいてオフセットを返します。

SAS 日時が指定されていない場合は、TZONEOFF 関数は現在のタイムゾーンオフセットを返します。SAS 日時が指定されている場合は、指定された時間値のローカル時間を提供するためのオフセットを返します。

ステートメント	結果
option TIMEZONE='AUSTRALIA/MELBOURNE'; %PUT %SYSFUNC(TZONEOFF());	39600
option TIMEZONE='AUSTRALIA/MELBOURNE'; %PUT %SYSFUNC(TZONEOFF(EUROPE/ROME));	3600
data _null_ ; dt1='05DEC2012:08:17:52'dt ; dt2='05JUN2012:08:17:52'dt ; offset1= TZONEOFF('EUROPE/MOSCOW', dt1); offset2= TZONEOFF('EUROPE/MOSCOW', dt2); put offset1= / offset2= ; run ;	offset1=14400 offset2=14400
option TIMEZONE='EUROPE/MOSCOW' ; data _null_ ; dt1='05DEC2012:08:17:52'dt ; dt2='05JUN2012:08:17:52'dt ; offset1= TZONEOFF(dt1); offset2= TZONEOFF(dt2); put offset1= / offset2= ;	offset1=14400 offset2=14400

ステートメント

結果

run;

TZONES2U 関数

SAS 日時値を UTC 日時値へ変換します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONES2U(*datetime*,<*time-zone-id*>)

必須引数

datetime

SAS 日時値を指定します。

オプション引数

time-zone-id

SAS で定義されている region/area の値を指定します。タイムゾーン ID の指定時、タイムゾーン名と夏時間ルールに基づきタイムゾーンが決定されます。

参照項目: タイムゾーン ID のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名” \(1145 ページ\)](#)を参照してください

詳細

TZONES2U()関数は、指定された TIMEZONE の UTC ベースの時間を返します。引数としてタイムゾーン ID を指定した TZONES2U(time-zone-id)関数は、指定したタイムゾーン ID に対する UTC ベースの時間を返します。タイムゾーン名が現在のロケールで無効な場合は、エラーを受け取ります。time-zone-id が指定されていない場合、TIMEZONE=オプションで指定された timezone ID が使用されます。

例

次の例では、SAS 日時値を UTC 時間値へ変換します。

ステートメント	結果
option locale=ja_JP TZ='JST' ; data _null_ ; dt='05Nov2012:08:17:52'dt ; utc1 = tzones2u(dt) ; utc2 = tzones2u(dt,'ASIA/TOKYO') ; utc3 = tzones2u(dt,'JST') ; put dt= /utc1= is8601dz. // ; put dt= /utc2= is8601dz. // ; put dt= /utc3= is8601dz. // ; run ;	dt=1667722672 utc1=2012-11-04T23:17:52+00:00 dt=1667722672 utc2=2012-11-04T23:17:52+00:00 dt=1667722672 utc3=2012-11-04T23:17:52+00:00

TZONEDSTNAME 関数

夏時間名を返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONEDSTNAME()

TZONEDSTNAME(<*time-zone-id*>)

オプション引数

time-zone-id

SAS で定義されている region/area の値を指定します。タイムゾーン ID を指定するとき、タイムゾーン名と夏時間ルールに基づいて、SAS で使用されるタイムゾーンが決定されます。

詳細

time-zone-id が指定されていない場合、TIMEZONE=オプションで指定された TIMEZONE ID が使用されます。指定されたタイムゾーンの夏時間が存在しない場合は、空白名が返されます。

例

次の例では、米国/シカゴのタイムゾーンを指定します。

ステートメント	結果
<pre>options timezone='America/Chicago'; dstname=CDT data null; dstname=tzonedstname(); put dstname; run;</pre>	

TZONEDSTOFF 関数

指定された夏時間のタイムゾーンオフセット値を返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONEDSTOFF()

TZONEDSTOFF(<time-zone-id>)

オプション引数

time-zone-id

SAS で定義されている region/area の値を指定します。タイムゾーン ID を指定するとき、タイムゾーン名と夏時間ルールに基づいて、SAS で使用されるタイムゾーンが決定されます。

詳細

time-zone-id が指定されていない場合、TIMEZONE=オプションで指定された TIMEZONE ID が使用されます。指定されたタイムゾーンの夏時間が存在しない場合は、空白名が返されます。

例

次の例では、米国/シカゴのタイムゾーンを指定します。

ステートメント	結果
<pre>options timezone='America/Chicago'; dstoff=18000 data null; dstoff=tzonedstoffs(); put dstoff; run;</pre>	

TZONESTTNAME 関数

標準のタイムゾーン名を返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONESTTNAME()

TZONESTTNAME(<*time-zone-id*>)

オプション引数

time-zone-id

SAS で定義された region 値または area 値を指定します。タイムゾーン ID を指定するとき、タイムゾーン名と夏時間ルールに基づいて、SAS で使用されるタイムゾーンが決定されます。

詳細

TZID が指定されていない場合、TIMEZONE=オプションで指定された TIMEZONE ID が使用されます。

例

次の例では、アジア/大阪のタイムゾーンを指定します。

ステートメント	結果
<pre>options timezone='Asia/Osaka'; JST data null; name=tzonesttname(); put name; run;</pre>	

TZONESTTOFF 関数

指定された標準時間のタイムゾーンオフセット値を返します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONESTTOFF()

TZONESTTOFF(<*time-zone-id*>)

オプション引数

time-zone-id

SAS で定義された region 値または area 値を指定します。タイムゾーン ID を指定するとき、タイムゾーン名と夏時間ルールに基づいて、SAS で使用されるタイムゾーンが決定されます。

詳細

time-zone-id が指定されていない場合、TIMEZONE=オプションで指定された TIMEZONE ID が使用されます。標準時が指定されていない場合は、空白名が返されます。

例

次の例では、アジア/大阪のタイムゾーンを指定します。

ステートメント	結果
<pre>options timezone='Asia/Osaka'; 32400 data null; name=tzonesttoff(); put name; run;</pre>	

TZONEU2S 関数

UTC 日時値を SAS 日時値へ変換します。

カテゴリ: 日付と時間

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

この関数は、CAS で実行される DATA ステップではサポートされていません。

構文

TZONEU2S(*UTC-date-time value*<, *time-zone-id*>)

必須引数

UTC-date time value

UTC (Coordinated Universal Time)日時値を指定します。

オプション引数

time-zone-id

SAS で定義されている region/area の値を指定します。ゾーン ID の指定時、タイムゾーン名と夏時間ルールに基づきタイムゾーンが決定されます。

詳細

TZONEU2S(datetime)関数は、指定された TIMEZONE オプションの UTC 時間に対する SAS 日時を返します。引数としてタイムゾーン ID を指定した TZONEU2S(datetime, time-zone-id)関数は、指定したタイムゾーン ID の UTC 時間に対する SAS 日時を返します。

例

次の例では、1 つの UTC 日時値を 3 つの特定の SAS 日時値へ変換します。

ステートメント	結果
<pre>option locale=fr_FR TZ='AMERICA/DENVER'; data _null_; utc_date = '2012-09-02T02:34:56+00:00'; udt = input(utc_date,js8601dz.); sdt1 = tzoneu2s(udt); sdt2 = tzoneu2s(udt,'EUROPE/AMSTERDAM'); sdt3 = tzoneu2s(udt,'CET'); put sdt1= datetime. / sdt2= datetime. / sdt3= datetime.; run;</pre>	<pre>sdt1=01SEP12:20:34:56 sdt2=02SEP12:04:34:56 sdt3=02SEP12:03:34:56</pre>

UNICODE 関数

Unicode 文字を現在の SAS セッションエンコーディングに変換します。

カテゴリ: 文字
CAS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

STR=UNICODE(<instr> (,<Unicode type>))

必須引数

str

現在の SAS セッションエンコーディングに変換されたデータ文字列。

オプション引数

instr

入力データ文字列。

Unicode type

Unicode 文字形式

ESC	Unicode エスケープ(たとえば、\u0042)。ESC はデフォルトの形式です。
NCR	数値文字表現(たとえば、大 や±))
PAREN	Unicode かっこエスケープ(たとえば、<u0061>)
UTF8	UTF8 エンコーディング
UTF16	ビッグエンディアンの UTF16 エンコーディング UCS2 は別名です。
UTF16B	ビッグエンディアンの UTF16 エンコーディング UCS2B は別名です。
UTF16L	リトルエンディアンの UTF16 エンコーディング UCS2L は別名です。
UTF32	ビッグエンディアンの UTF32 エンコーディング UCS4 は別名です。
UTF32B	ビッグエンディアンの UTF32 エンコーディング UCS4B は別名です。
UTF32L	リトルエンディアンの UTF32 エンコーディング UCS4L は別名です。

詳細

Unicode 文字を読み込み、現在の SAS セッションエンコーディングに変換します。

UNICODE および UNICODDEC 関数は、UTF-8、UTF16、または UTF32 エンコーディングの Unicode SMP(Supplementary Multilingual Plane)文字を受け入れることができますが、\uD84C または\uDF8F などの代理ペア形式の Unicode SMP 文字を受け入れることはできません。“[例 2:](#)”を参照してください。

例

例 1

この例は、UNICODE 関数の使用例を示しています。このコードは、どの CJK ロケールでも実行できます。

```
data _null;
```

```

str1=unicode("\u0041\u0042\u0043"); put str1=; /* default */
str2=unicode("\u0041\u0042\u0043", "esc"); put str2=; /* ESC - Unicode Escape */
str3=unicode("&#177;", "ncr"); put str3=; /* NCR - Numeric Character Representation */
str4=unicode("&#22823", "ncr"); put str4=; /* NCR - Numeric Character Representation */
str5=unicode('<\u0061>\u0062>', 'paren'); put str5=; /* PAREN - Unicode Parenthesis Escape */
str6=unicode('2759'x, 'ucs2'); put str6=; /* UCS2 - UCS2 encoding */
str7=unicode('5927'x, 'ucs2b'); put str7=; /* UCS2B - UCS2 encoding with big endian */
str8=unicode('2759'x, 'ucs2l'); put str8=; /* UCS2L - UCS2 encoding with little endian */
str9=unicode('27590000'x, 'ucs4'); put str9=; /* UCS4 - UCS4 encoding */
str10=unicode('00005927'x, 'ucs4b'); put str10=; /* UCS4B - UCS4 encoding with big endian */
str11=unicode('27590000'x, 'ucs4l'); put str11=; /* UCS4L - UCS4 encoding with little endian */
str12=unicode('E5A4A7'x, 'utf8'); put str12=; /* UTF8 - UTF8 encoding */
str13=unicode('2759'x, 'utf16'); put str13=; /* UTF16 - UTF16 encoding */
str14=unicode('5927'x, 'utf16b'); put str14=; /* UTF16B - UTF16 encoding with big endian */
str15=unicode('2759'x, 'utf16l'); put str15=; /* UTF16L - UTF16 encoding with little endian */
str16=unicode("\u00020bb7", "esc"); put str16=; /* ESC - Unicode Escape for Supplementary Character */
str17=unicode("&#134071;", "ncr"); put str17=; /* NCR - Numeric Character Representation for Supplementary Character */
str18=unicode('<\u00020BB7>', 'paren'); put str18=; /* PAREN - Unicode Parenthesis Escape for
Supplementary Character*/

run;

```

SAS は次の出力をログに書き込みます。

```

str1=ABC
str2=ABC
str3=±
str4=大
str5=ab
str6=大
str7=大
str8=大
str9=大
str10=大
str11=大
str12=大
str13=大
str14=大
str15=大

```

例 2

この例は、SMP 文字を処理するときの UNICODE 関数および UNICODDEC 関数の機能を示しています。

```

data _null_;
  length x x1 x2 $4 y $10 y1 $9 y2 $11 u16 $4 u32 $4 z z1 z2 $4;
  x = '醜'; /* SMP character */
  u16 = UNICODEC(x, "utf16b");
  x1 = UNICODE(u16, "utf16b");
  u32 = UNICODEC(x, "utf32b");
  x2 = UNICODE(u32, "utf32b");
  y = UNICODEC(x, "esc");
  z = UNICODE(y, "esc");
  y1 = UNICODEC(x, "ncr");
  z1 = UNICODE(y1, "ncr");
  y2 = UNICODEC(x, "paren");
  z2 = UNICODE(y2, "paren");
  put x= x=hex.;
  put x1= x1=hex.;
  put x2= x2=hex.;
  put u16=hex. /u32=hex.;
  put y= y=hex.;
  put y1= y1=hex.;
  put y2= y2=hex.;
  put z= z1= z2=;
run;

```

SAS は次の出力をログに書き込みます。

```

x=醜 x=F0A38E8F
x1=醜 x1=F0A38E8F
x2=醜 x2=F0A38E8F
u16=D84CDF8F
u32=0002338F
y=\u0002338F y=5C753030303233333846
y1=&#144271; y1=26233134343237313B20
y2=<\u0002338F> y2=3C7530303032333338463E
z=醜 z1=醜 z2=醜

```

UNICODEC 関数

現在の SAS セッションエンコーディングの文字を Unicode 文字に変換します。

カテゴリ: 文字
CAS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

STR=UNICODEC(<*instr*> (,<Unicode type>))

必須引数

str

Unicode エンコーディングに変換されたデータ文字列。

オプション引数

instr

入力データ文字列。

Unicode type

Unicode 文字形式

ESC	Unicode エスケープ(たとえば、\u0042)。ESC はデフォルトの形式です。
NCR	数値文字表現(たとえば、大 や±))
PAREN	Unicode かっこエスケープ(たとえば、<u0061>)
UTF8	UTF8 エンコーディング
UTF16	ビッグエンディアンの UTF16 エンコーディング UCS2 は別名です。
UTF16B	ビッグエンディアンの UTF16 エンコーディング UCS2B は別名です。
UTF16L	リトルエンディアンの UTF16 エンコーディング UCS2L は別名です。
UTF32	ビッグエンディアンの UTF32 エンコーディング UCS4 は別名です。
UTF32B	ビッグエンディアンの UTF32 エンコーディング UCS4B は別名です。
UTF32L	ビッグエンディアンの UTF32L エンコーディング UCS4L は別名です。

詳細

現在の SAS セッションエンコーディングに含まれる文字を読み込み、Unicode エンコーディングに変換します。

UNICODE および UNICODDEC 関数は、UTF-8、UTF16、または UTF32 エンコーディングの Unicode SMP(Supplementary Multilingual Plane)文字を受け入れることができますが、\uD84C または\uDF8F などの代理ペア形式の Unicode SMP 文字を受け入れることはできません。“例 2.”を参照してください。

例

例 1

この例は、UNICODEC 関数の使用例を示しています。

```
data _null_;
  str1=unicodec("ABC", 'utf8');
  put str1= $hex12.;
  str2=unicodec("ABCé", 'utf8');
  put str2= $hex12.;
run;
```

SAS は次の出力をログに書き込みます。

```
str1=414243202020
str2=414243C3A920
```

例 2

次の例は、SMP 文字を処理するときの UNICODE 関数および UNICODEC 関数の機能を示しています。

```
data _null_;
  length x x1 x2 $4 y $10 y1 $9 y2 $11 u16 $4 u32 $4 z z1 z2 $4;
  x = '𐀀'; /* SMP character */
  u16 = UNICODEC(x, "utf16b");
  x1 = UNICODE(u16, "utf16b");
  u32 = UNICODEC(x, "utf32b");
  x2 = UNICODE(u32, "utf32b");
  y = UNICODEC(x, "esc");
  z = UNICODE(y, "esc");
  y1 = UNICODEC(x, "ncr");
  z1 = UNICODE(y1, "ncr");
  y2 = UNICODEC(x, "paren");
  z2 = UNICODE(y2, "paren");
  put x= x=hex.;
  put x1= x1=hex.;
  put x2= x2=hex.;
  put u16=hex. /u32=hex.;
  put y= y=hex.;
  put y1= y1=hex.;
  put y2= y2=hex.;
  put z= z1= z2=;
run;
```

SAS は次の出力をログに書き込みます。

```

x=醜 x=F0A38E8F
x1=醜 x1=F0A38E8F
x2=醜 x2=F0A38E8F
u16=D84CDF8F
u32=0002338F
y=\u0002338F y=5C753030303233333846
y1=&#144271; y1=26233134343237313B20
y2=<u0002338F> y2=3C7530303032333338463E
z=醜 z1=醜 z2=醜

```

UNICODELEN 関数

Unicode データの文字単位の長さを示します。

カテゴリ: 文字
CAS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

UNICODELEN()

詳細

UNICODELEN 関数では、Unicode データの文字単位の長さを示します。

例

この例では、Linux オペレーティングシステムでサポートされている日本語 (Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
len1=unicodelen("abc 大");	len1=4

len2=unicodelen("\u0041\u0042\u0043\u5927','esc');	len2=4
len3=unicodelen("大','ncr');	len3=1
len4=unicodelen("<u0061><u0062>','paren');	len4=2

関連項目

関数:

- [“UNICODEWIDTH 関数” \(749 ページ\)](#)

UNICODEWIDTH 関数

Unicode データの表示単位の長さを示します。

カテゴリ: 文字
CAS

制限事項: この関数は I18N レベル 2 ステータスがアサインされており、SBCS、DBCS および MBCS(UTF8)で使用するために設計されています。 詳細については、[国際化の互換性 \(555 ページ\)](#)を参照してください。

構文

UNICODEWIDTH()

詳細

UNICODEWIDTH 関数では、Unicode データの表示単位の長さを示します。表示単位は、文字を固定幅フォントで表示する時の文字の幅を示します。

CJK 記号の文字、CJK 統合漢字、全角アルファベットと句読点、全角通貨記号、CJK 統合漢字拡張 B-D および CJK 互換漢字補助は、表示単位の値が 2 になります。他の文字は表示単位の値が 1 になります。

例

この例では、Linux 動作システムでサポートされる日本語(Shift_JIS)セッションエンコーディングを使用します。

ステートメント	結果
len1=unicodewidth("abc 太");	len1=5
len2=unicodewidth("\u0041\u0042\u0043\u5927", 'esc');	len2=5
len3=unicodewidth("大 ", 'ncr');	len3=2
len4=unicodewidth("<u0061><u0062>", 'paren');	len4=2

関連項目

関数:

- [“UNICODELEN 関数” \(748 ページ\)](#)

VARTRANSCODE 関数

SAS データセット変数のトランスコーディング属性を返します。

カテゴリ: 変数情報

構文

VARTRANSCODE(*data-set-id*, *var-num*)

必須引数

- data-set-id*
OPEN 関数が返すデータセット識別子を指定します。
- var-num*
SAS データセット内での変数の位置を指定します。

ヒント VARNUM 関数はこの値を返します。

詳細

トランスコーディングとは、1 つのエンコーディングから別のエンコーディングにデータを変換する処理のことです。VARTRANSCODE 関数は、*var-num* 変数とその

値をトランスコードしない場合は 0 を、*var-num* 変数とその値をトランスコードする場合は 1 を返します。

変数のトランスコーディングの詳細については *SAS 各国語サポート(NLS): リファレンスガイド*の[トランスコーディング \(33 ページ\)](#)を参照してください。値のエンコーディングおよびデータのトランスコーディングの詳細については、*SAS 各国語サポート(NLS): リファレンスガイド*の[AS データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値 \(1118 ページ\)](#)を参照してください。

例

次の例は、文字変数をトランスコードするかどうかの決定方法を示しています。

```
data a;
  attrib x length=$3. transcode=no;
  attrib y length=$3. transcode=yes;
  x='abc';
  y='xyz';
run;
data _null_;
  dsid=open('work.a','i');
  nobs=attrn(dsid,"nobs");
  nvars=attrn(dsid,"nvars");
  do i=1 to nobs;
    xrc=fetch(dsid,1);
    do j=1 to nvars;
      transcode = vartranscode(dsid,j);
      put transcode=;
    end;
  end;
run;
```

SAS は次の出力をログに書き込みます。

```
transcode=0
transcode=1
```

関連項目

関数:

- [“ATTRN 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)
- [“OPEN 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)
- [“VARNUM 関数” \(SAS 関数と CALL ルーチン: リファレンス\)](#)
- [“VTRANSCODE 関数” \(752 ページ\)](#)
- [“VTRANSCODEX 関数” \(753 ページ\)](#)

VTRANSCODE 関数

指定された文字変数でトランスコーディングが有効かどうかを示す値を返します。

カテゴリ: CAS
変数情報

構文

VTRANSCODE (*var*)

必須引数

var

スカラーまたは配列参照として表される文字変数を指定します。

制限事項 式は引数として使用できません。

詳細

VTRANSCODE 関数は、トランスコーディングがオフの場合は 0 を、オンの場合は 1 を返します。

デフォルトでは、DATA ステップのすべての文字変数がトランスコードされます。トランスコーディングをオフにするには、ATTRIB ステートメントの TRANSCODE=属性を使用します。

比較

- VTRANSCODE 関数は、指定された変数でトランスコーディングが有効かどうかを示す値を返します。一方、VTRANSCODEX 関数は、変数名を判別するために引数を評価します。その後、VTRANSCODEX 関数はその変数名に関連付けられたトランスコーディングステータス(オンまたはオフ)を返します。
- VTRANSCODE 関数は、引数として式を受け入れません。VTRANSCODEX 関数は式を受け入れますが、指定された式の値で配列参照を示すことはできません。
- 関連する関数が、変数名、種類、出力形式、長さなど他の変数属性の値を返しません。変数属性のリストについては、“変数情報”の関数(SAS 関数と CALL ルーチン: リファレンス)を参照してください。

例

ステートメント	結果
	----+----1----+
attrib x transcode = yes; attrib y transcode = no; rc1 = vtranscode(y); put rc1=;	rc1=0

関連項目

関数:

- [“VTRANSCODEX 関数” \(753 ページ\)](#)

ステートメント:

- [ATTRIB](#) in

VTRANSCODEX 関数

指定された引数でトランスコーディングが有効かどうかを示す値を返します。

カテゴリ: CAS
変数情報

構文

VTRANSCODEX (*var*)

必須引数

var

文字変数名に対して評価する SAS 文字式を指定します。

制限事項 指定された式の値で配列参照を示すことはできません。

詳細

VTRANSCODEX 関数は、トランスコーディングがオフの場合は 0 を、オンの場合は 1 を返します。

デフォルトでは、DATA ステップのすべての文字変数がトランスコードされます。トランスコーディングをオフにするには、ATTRIB ステートメントの TRANSCODE=属性を使用します。

比較

- VTRANSCODE 関数は、指定された変数でトランスコーディングが有効かどうかを示す値を返します。一方、VTRANSCODEX 関数は、変数名を判別するために引数を評価します。その後、VTRANSCODEX 関数はその変数名に関連付けられたトランスコーディングステータス(オンまたはオフ)を返します。
- VTRANSCODE 関数は、引数として式を受け入れません。VTRANSCODEX 関数は式を受け入れますが、指定された式の値で配列参照を示すことはできません。
- 関連する関数が、変数名、種類、出力形式、長さなど他の変数属性の値を返します。変数属性のリストについては、“変数情報”の関数(SAS 関数と CALL ルーチン: リファレンス)を参照してください。

例

ステートメント	結果
	----+----1----+
attrib x transcode = yes; attrib y transcode = no; rc1 = vtranscodex('y'); put rc1=;	rc1=0

関連項目

関数:

- [“VTRANSCODE 関数” \(752 ページ\)](#)

ステートメント:

- [ATTRIB](#)

6 部

各国語サポート関連の入力形式

14 章

各国語サポート関連の入力形式のディクショナリ	757
------------------------------	-----

各国語サポート関連の入力形式の ディクショナリ

カテゴリ別の入力形式	759
ディクショナリ	773
\$CPTDWw. 入力形式	773
\$CPTWDw. 入力形式	774
EUROW.d 入力形式	775
EUROXw.d 入力形式	778
JNENGOW. 入力形式	780
\$KANJIw. 入力形式	781
\$KANJIxw. 入力形式	782
\$LOGVSw. 入力形式	783
\$LOGVSRw. 入力形式	785
MINGUOW. 入力形式	786
NENGOW. 入力形式	788
NLDATEw. 入力形式	789
NLDATEWw. 入力形式	791
NLDATMw. 入力形式	792
NLDATMAP 入力形式	793
NLDATMWw. 入力形式	794
NLMNIAEDw.d 入力形式	795
NLMNIAUDw.d 入力形式	796
NLMNIBGNw.d 入力形式	798
NLMNIBRLw.d 入力形式	799
NLMNICADw.d 入力形式	800
NLMNICHfw.d 入力形式	801
NLMNICNYw.d 入力形式	803
NLMNICZKw.d 入力形式	804
NLMNIDKKw.d 入力形式	805
NLMNIEEKw.d 入力形式	806
NLMNIEGPw.d 入力形式	808
NLMNIEURw.d 入力形式	809
NLMNIGBPw.d 入力形式	810
NLMNIHKDw.d 入力形式	812
NLMNIHRKw.d 入力形式	813

NLMNIHUFw.d 入力形式	814
NLMNIIDRw.d 入力形式	815
NLMNIILSw.d 入力形式	817
NLMNIINRw.d 入力形式	818
NLMNIJPYw.d 入力形式	819
NLMNIKRWw.d 入力形式	820
NLMNITLw.d 入力形式	822
NLMNILVLw.d 入力形式	823
NLMNIMOPw.d 入力形式	824
NLMNIMXNw.d 入力形式	825
NLMNIMYRw.d 入力形式	827
NLMNINOKw.d 入力形式	828
NLMNINZDw.d 入力形式	829
NLMNIPLNw.d 入力形式	830
NLMNIRUBw.d 入力形式	832
NLMNISEKw.d 入力形式	833
NLMNISGDw.d 入力形式	834
NLMNITHBw.d 入力形式	835
NLMNITRYw.d 入力形式	837
NLMNITWDw.d 入力形式	838
NLMNIUSDw.d 入力形式	839
NLMNIZARw.d 入力形式	840
NLMNLAEDw.d 入力形式	842
NLMNLAUDw.d 入力形式	843
NLMNLBGNw.d 入力形式	844
NLMNLBRLw.d 入力形式	845
NLMNLCADw.d 入力形式	847
NLMNLCHFw.d 入力形式	848
NLMNLCNYw.d 入力形式	849
NLMNLCZKw.d 入力形式	850
NLMNLDKKw.d 入力形式	852
NLMNLEEKw.d 入力形式	853
NLMNLEGPw.d 入力形式	854
NLMNLEURw.d 入力形式	855
NLMNLGBPw.d 入力形式	857
NLMNLHKDw.d 入力形式	858
NLMNLHRKw.d 入力形式	859
NLMNLHUFw.d 入力形式	861
NLMNLIDRw.d 入力形式	862
NLMNLILSw.d 入力形式	863
NLMNLINRw.d 入力形式	864
NLMNLJPYw.d 入力形式	866
NLMNLKRWw.d 入力形式	867
NLMNLLTLw.d 入力形式	868
NLMNLLVLw.d 入力形式	869
NLMNLMOPw.d 入力形式	871
NLMNLMXNw.d 入力形式	872
NLMNLMYRw.d 入力形式	873
NLMNLNOKw.d 入力形式	874
NLMNLNZDw.d 入力形式	876
NLMNLPLNw.d 入力形式	877
NLMNLRUBw.d 入力形式	878
NLMNLSEKw.d 入力形式	879
NLMNLSGDw.d 入力形式	881

NLMNLTHBw.d 入力形式	882
NLMNLTRYw.d 入力形式	883
NLMNLTDWd.d 入力形式	884
NLMNLUSDw.d 入力形式	886
NLMNLZARw.d 入力形式	887
NLMNYw.d 入力形式	888
NLMNYIw.d 入力形式	890
NLNUMw.d 入力形式	892
NLNUMIw.d 入力形式	893
NLPCTw.d 入力形式	895
NLPCTIw.d 入力形式	897
NLSTRMON 入力形式	898
NLTIMAPw. 入力形式	900
NLTIMEw. 入力形式	901
\$REVERJw. 入力形式	902
\$REVERSw. 入力形式	903
\$UCS2Bw. 入力形式	904
\$UCS2BEw. 入力形式	906
\$UCS2Lw. 入力形式	907
\$UCS2LEw. 入力形式	908
\$UCS2Xw. 入力形式	910
\$UCS2XEw. 入力形式	911
\$UCS4Bw. 入力形式	912
\$UCS4Lw. 入力形式	913
\$UCS4Xw. 入力形式	915
\$UCS4XEw. 入力形式	916
\$UESCw. 入力形式	917
\$UESCEw. 入力形式	919
\$UNCRw. 入力形式	920
\$UNCREw. 入力形式	921
\$UPARENw. 入力形式	923
\$UPARENEw. 入力形式	924
\$UPARENpw. 入力形式	926
\$UTF8Xw. 入力形式	927
\$VSLOGw. 入力形式	928
\$VSLOGRw. 入力形式	930
YENw.d 入力形式	931

カテゴリ別の入力形式

一部の入力形式は SAS 内でのみ実行され、一部の入力形式は SAS 内および CAS エンジン上で実行されます。入力形式カテゴリとして CAS を指定した場合、入力形式は SAS 内および CAS サーバー上で実行されます。入力形式カテゴリとして CAS を指定しない場合、入力形式は SAS 内でのみ実行されます。たとえば、NLDATE 入力形式は SAS 内および CAS サーバー上で実行されるため、カテゴリとして CAS を指定します。\$UNCR 入力形式は SAS でしか実行されないため、カテゴリとして CAS を指定しません。

SAS 入力形式のこれらのカテゴリは NLS をサポートしています。

カテゴリ	説明
BIDI テキストの操作	データ変数から双方向のデータ値を読み込むように指示します。 入力形式のリストについては、 BIDI テキスト処理 を参照してください。
CAS	この入力形式を CAS サーバー上で実行するように指示します。 入力形式のリストについては、 CAS を参照してください。
文字	文字変数に文字データ値を読み込むように指示します。 入力形式のリストについては、 文字 を参照してください。
DBCS	さまざまなアジア言語を管理するように指示します。 入力形式のリストについては、 DBCS を参照してください。
日付と時間	日付、時刻および日時を表す変数にデータ値を読み込むように指示します。 入力形式のリストについては、 日付と時間 を参照してください。
ヘブライ語テキスト処理	データ変数からヘブライ語のデータを読み込むように指示します。 入力形式のリストについては、 ヘブライ語テキスト処理 を参照してください。
数値	数値変数に数値データ値を読み込むように指示します。 入力形式のリストについては、 数値 を参照してください。

次の表に、SAS 入力形式の概要を示します。詳細については、それぞれの各国語サポート関連の入力形式に関する情報を参照してください。

カテゴリ	言語要素	説明
BIDI テキストの操作	\$LOGVSw. 入力形式 (p. 783)	左から右に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。
	\$LOGVSRw. 入力形式 (p. 785)	右から左に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。
	\$VSLOGw. 入力形式 (p. 928)	視覚的順序で並ぶ文字列を読み込み、その文字列を左から右の論理的順序に変換します。
	\$VSLOGRw. 入力形式 (p. 930)	視覚的順序で並ぶ文字列を読み込み、その文字列を右から左の論理的順序に変換します。
CAS	EUROW.d 入力形式 (p. 775)	数値を読み込み、ヨーロッパ通貨の埋め込み文字を取り除いて、カンマと小数点を逆にします。
	EUROXw.d 入力形式 (p. 778)	数値を読み込み、ヨーロッパ通貨の埋め込み文字を削除します。

カテゴリ	言語要素	説明
	\$LOGVSw. 入力形式 (p. 783)	左から右に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。
	\$LOGVSRw. 入力形式 (p. 785)	右から左に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。
	NENGOW. 入力形式 (p. 788)	eyymmdd 形式の日本の日付値を読み込みます。
	NLDATEw. 入力形式 (p. 789)	日付値を、現在の SAS ロケールに適したローカルの SAS 日付値として読み取ります。
	NLDATEWw. 入力形式 (p. 791)	日付値を、現在の SAS ロケールに適したローカルの SAS 日付と曜日の形式で読み取ります。
	NLDATMw. 入力形式 (p. 792)	現在の SAS ロケールに適した日時値を読み込みます。
	NLDATMAP 入力形式 (p. 793)	日時値を、現在の SAS ロケールに適した現地の SAS 日時値 (午前または午後の指定付き)として読み取ります。
	NLDATMWw. 入力形式 (p. 794)	日付値を、現在の SAS ロケールに適した現地の SAS の曜日と日時として読み取ります。
	NLMNIAEDw.d 入力形式 (p. 795)	アラブ首長国連邦の国際表現の通貨出力形式を読み込みます。
	NLMNIAUDw.d 入力形式 (p. 796)	オーストラリアの国際表現の通貨出力形式を読み込みます。
	NLMNIBGNw.d 入力形式 (p. 798)	ブルガリアの国際表現の通貨出力形式を読み込みます。
	NLMNIBRLw.d 入力形式 (p. 799)	ブラジルの国際表現の通貨出力形式を読み込みます。
	NLMNICADw.d 入力形式 (p. 800)	カナダの国際表現の通貨出力形式を読み込みます。
	NLMNICHFw.d 入力形式 (p. 801)	リヒテンシュタインとスイスの国際表現の通貨出力形式を読み込みます。
	NLMNICNYw.d 入力形式 (p. 803)	中国の国際表現の通貨出力形式を読み込みます。
	NLMNICZKw.d 入力形式 (p. 804)	チェコの国際表現の通貨出力形式を読み込みます。
	NLMNIDKKw.d 入力形式 (p. 805)	デンマーク、フェロー諸島およびグリーンランドの国際表現の通貨出力形式を読み込みます。
	NLMNIEEKw.d 入力形式 (p. 806)	エストニアの国際表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNIEGPw.d 入力形式 (p. 808)	エジプトの国際表現の通貨出力形式を読み込みます。
	NLMNIEURw.d 入力形式 (p. 809)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際表現の通貨出力形式を読み込みます。
	NLMNIGBPw.d 入力形式 (p. 810)	英国の国際表現の通貨出力形式を読み込みます。
	NLMNIHKDw.d 入力形式 (p. 812)	香港の国際表現の通貨出力形式を読み込みます。
	NLMNIHRKw.d 入力形式 (p. 813)	クロアチアの国際表現の通貨出力形式を読み込みます。
	NLMNIHUFw.d 入力形式 (p. 814)	ハンガリーの国際表現の通貨出力形式を読み込みます。
	NLMNIIDRw.d 入力形式 (p. 815)	インドネシアの国際表現の通貨出力形式を読み込みます。
	NLMNIILSw.d 入力形式 (p. 817)	イスラエルの国際表現の通貨出力形式を読み込みます。
	NLMNIINRw.d 入力形式 (p. 818)	インドの国際表現の通貨出力形式を読み込みます。
	NLMNIJPYw.d 入力形式 (p. 819)	日本の国際表現の通貨出力形式を読み込みます。
	NLMNIKRWw.d 入力形式 (p. 820)	韓国の国際表現の通貨出力形式を読み込みます。
	NLMNITLw.d 入力形式 (p. 822)	リトアニアの国際表現の通貨出力形式を読み込みます。
	NLMNILLVw.d 入力形式 (p. 823)	ラトビアの国際表現の通貨出力形式を読み込みます。
	NLMNIMOPw.d 入力形式 (p. 824)	マカオの国際表現の通貨出力形式を読み込みます。
	NLMNIMXNw.d 入力形式 (p. 825)	メキシコの国際表現の通貨出力形式を読み込みます。
	NLMNIMYRw.d 入力形式 (p. 827)	マレーシアの国際表現の通貨出力形式を読み込みます。
	NLMNINOKw.d 入力形式 (p. 828)	ノルウェーの国際表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNINZDw.d 入力形式 (p. 829)	ニュージーランドの国際表現の通貨出力形式を読み込みます。
	NLMNIPLNw.d 入力形式 (p. 830)	ポーランドの国際表現の通貨出力形式を読み込みます。
	NLMNIRUBw.d 入力形式 (p. 832)	ロシアの国際表現の通貨出力形式を読み込みます。
	NLMNISEKw.d 入力形式 (p. 833)	スウェーデンの国際表現の通貨出力形式を読み込みます。
	NLMNISGDw.d 入力形式 (p. 834)	シンガポールの国際表現の通貨出力形式を読み込みます。
	NLMNITHBw.d 入力形式 (p. 835)	タイの国際表現の通貨出力形式を読み込みます。
	NLMNITRYw.d 入力形式 (p. 837)	トルコの国際表現の通貨出力形式を読み込みます。
	NLMNITWDw.d 入力形式 (p. 838)	台湾の国際表現の通貨出力形式を読み込みます。
	NLMNIUSDw.d 入力形式 (p. 839)	プエルトリコと米国の国際表現の通貨出力形式を読み込みます。
	NLMNIZARw.d 入力形式 (p. 840)	南アフリカの国際表現の通貨出力形式を読み込みます。
	NLMNLAEDw.d 入力形式 (p. 842)	アラブ首長国連邦の現地表現の通貨出力形式を読み込みます。
	NLMNLAUDw.d 入力形式 (p. 843)	オーストラリアの現地表現の通貨出力形式を読み込みます。
	NLMNLBGNw.d 入力形式 (p. 844)	ブルガリアの現地表現の通貨出力形式を読み込みます。
	NLMNLBRLw.d 入力形式 (p. 845)	ブラジルの現地表現の通貨出力形式を読み込みます。
	NLMNLCADw.d 入力形式 (p. 847)	カナダの現地表現の通貨出力形式を読み込みます。
	NLMNLCHFw.d 入力形式 (p. 848)	リヒテンシュタインとスイスの現地表現の通貨出力形式を読み込みます。
	NLMNLCNYw.d 入力形式 (p. 849)	中国の現地表現の通貨出力形式を読み込みます。
	NLMNLCZKw.d 入力形式 (p. 850)	チェコの現地表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNLDKKw.d 入力形式 (p. 852)	デンマーク、フェロー諸島およびグリーンランドの現地表現の通貨出力形式を読み込みます。
	NLMNLEEKw.d 入力形式 (p. 853)	エストニアの現地表現の通貨出力形式を読み込みます。
	NLMNLEGPw.d 入力形式 (p. 854)	エジプトの現地表現の通貨出力形式を読み込みます。
	NLMNLEURw.d 入力形式 (p. 855)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地表現の通貨出力形式を読み込みます。
	NLMNLGBPw.d 入力形式 (p. 857)	英国の現地表現の通貨出力形式を読み込みます。
	NLMNLHKDw.d 入力形式 (p. 858)	香港の現地表現の通貨出力形式を読み込みます。
	NLMNLHRKw.d 入力形式 (p. 859)	クロアチアの現地表現の通貨出力形式を読み込みます。
	NLMNLHUFw.d 入力形式 (p. 861)	ハンガリーの現地表現の通貨出力形式を読み込みます。
	NLMNLIDRw.d 入力形式 (p. 862)	インドネシアの現地表現の通貨出力形式を読み込みます。
	NLMNLILSw.d 入力形式 (p. 863)	イスラエルの現地表現の通貨出力形式を読み込みます。
	NLMNLINRw.d 入力形式 (p. 864)	インドの現地表現の通貨出力形式を読み込みます。
	NLMNLJPYw.d 入力形式 (p. 866)	日本の現地表現の通貨出力形式を読み込みます。
	NLMNLKRWw.d 入力形式 (p. 867)	韓国の現地表現の通貨出力形式を読み込みます。
	NLMNLLTLw.d 入力形式 (p. 868)	リトアニアの現地表現の通貨出力形式を読み込みます。
	NLMNLLVLw.d 入力形式 (p. 869)	ラトビアの現地表現の通貨出力形式を読み込みます。
	NLMNLMOPw.d 入力形式 (p. 871)	マカオの現地表現の通貨出力形式を読み込みます。
	NLMNLMXNw.d 入力形式 (p. 872)	メキシコの現地表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNLMYRw.d 入力形式 (p. 873)	マレーシアの現地表現の通貨出力形式を読み込みます。
	NLMNLNOKw.d 入力形式 (p. 874)	ノルウェーの現地表現の通貨出力形式を読み込みます。
	NLMNLNZDw.d 入力形式 (p. 876)	ニュージーランドの現地表現の通貨出力形式を読み込みます。
	NLMNLPLNw.d 入力形式 (p. 877)	ポーランドの現地表現の通貨出力形式を読み込みます。
	NLMNLRUBw.d 入力形式 (p. 878)	ロシアの現地表現の通貨出力形式を読み込みます。
	NLMNLSEKw.d 入力形式 (p. 879)	スウェーデンの現地表現の通貨出力形式を読み込みます。
	NLMNLSGDw.d 入力形式 (p. 881)	シンガポールの現地表現の通貨出力形式を読み込みます。
	NLMNLTHBw.d 入力形式 (p. 882)	タイの現地表現の通貨出力形式を読み込みます。
	NLMNLTRYw.d 入力形式 (p. 883)	トルコの現地表現の通貨出力形式を読み込みます。
	NLMNLTWDw.d 入力形式 (p. 884)	台湾の現地表現の通貨出力形式を読み込みます。
	NLMNLUSDw.d 入力形式 (p. 886)	プエルトリコと米国の現地表現の通貨出力形式を読み込みます。
	NLMNLZARw.d 入力形式 (p. 887)	南アフリカの現地表現の通貨出力形式を読み込みます。
	NLMNYw.d 入力形式 (p. 888)	現地表現の通貨出力形式を、現在の SAS ロケールに適した数値として読み取ります。
	NLMNYIw.d 入力形式 (p. 890)	通貨データを、現在の SAS ロケールに適した国際表現の数値として読み込みます。
	NLNUMw.d 入力形式 (p. 892)	現地表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。
	NLNUMIw.d 入力形式 (p. 893)	国際表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。
	NLPCTw.d 入力形式 (p. 895)	現地表現の百分率のデータを現在の SAS ロケールに適した数値として読み取ります。
	NLPCTIw.d 入力形式 (p. 897)	国際表現の百分率のデータを、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ	言語要素	説明
	NLSTRMON 入力形式 (p. 898)	月名を、現在の SAS ロケールに適した数値として読み込みます。
	NLTIMAPw. 入力形式 (p. 900)	時間値を、a.m.および p.m.表示を使用し現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。
	NLTIMEw. 入力形式 (p. 901)	時間値を、現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。
	\$REVERJw. 入力形式 (p. 902)	文字データを右から左の順に読み込み、空白を保持します。
	\$REVERSw. 入力形式 (p. 903)	文字データを右から左の順に左詰めで読み込みます。
	\$VSLOGw. 入力形式 (p. 928)	視覚的順序で並ぶ文字列を読み込み、その文字列を左から右の論理的順序に変換します。
	\$VSLOGRw. 入力形式 (p. 930)	視覚的順序で並ぶ文字列を読み込み、その文字列を右から左の論理的順序に変換します。
DBCS	YENw.d 入力形式 (p. 931)	埋め込まれた円記号、カンマおよび小数点を削除します。
	\$KANJIw. 入力形式 (p. 781)	DBCS データからシフトコードデータを削除します。
	\$KANJIXw. 入力形式 (p. 782)	シフトコードデータを DBCS データに追加します。
数値	EUROW.d 入力形式 (p. 775)	数値を読み込み、ヨーロッパ通貨の埋め込み文字を取り除いて、カンマと小数点を逆にします。
	EUROXw.d 入力形式 (p. 778)	数値を読み込み、ヨーロッパ通貨の埋め込み文字を削除します。
	NLMNIAEDw.d 入力形式 (p. 795)	アラブ首長国連邦の国際表現の通貨出力形式を読み込みます。
	NLMNIAUDw.d 入力形式 (p. 796)	オーストラリアの国際表現の通貨出力形式を読み込みます。
	NLMNIBGNw.d 入力形式 (p. 798)	ブルガリアの国際表現の通貨出力形式を読み込みます。
	NLMNIBRLw.d 入力形式 (p. 799)	ブラジルの国際表現の通貨出力形式を読み込みます。
	NLMNICADw.d 入力形式 (p. 800)	カナダの国際表現の通貨出力形式を読み込みます。
	NLMNICHw.d 入力形式 (p. 801)	リヒテンシュタインとスイスの国際表現の通貨出力形式を読み込みます。
	NLMNICNYw.d 入力形式 (p. 803)	中国の国際表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNICZKw.d 入力形式 (p. 804)	チェコの国際表現の通貨出力形式を読み込みます。
	NLMNIDKKw.d 入力形式 (p. 805)	デンマーク、フェロー諸島およびグリーンランドの国際表現の通貨出力形式を読み込みます。
	NLMNIEEKw.d 入力形式 (p. 806)	エストニアの国際表現の通貨出力形式を読み込みます。
	NLMNIEGPw.d 入力形式 (p. 808)	エジプトの国際表現の通貨出力形式を読み込みます。
	NLMNIEURw.d 入力形式 (p. 809)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際表現の通貨出力形式を読み込みます。
	NLMNIGBPw.d 入力形式 (p. 810)	英国の国際表現の通貨出力形式を読み込みます。
	NLMNIHKDw.d 入力形式 (p. 812)	香港の国際表現の通貨出力形式を読み込みます。
	NLMNIHRKw.d 入力形式 (p. 813)	クロアチアの国際表現の通貨出力形式を読み込みます。
	NLMNIHUFw.d 入力形式 (p. 814)	ハンガリーの国際表現の通貨出力形式を読み込みます。
	NLMNIIDRw.d 入力形式 (p. 815)	インドネシアの国際表現の通貨出力形式を読み込みます。
	NLMNIILSw.d 入力形式 (p. 817)	イスラエルの国際表現の通貨出力形式を読み込みます。
	NLMNIINRw.d 入力形式 (p. 818)	インドの国際表現の通貨出力形式を読み込みます。
	NLMNIJPYw.d 入力形式 (p. 819)	日本の国際表現の通貨出力形式を読み込みます。
	NLMNIKRWw.d 入力形式 (p. 820)	韓国の国際表現の通貨出力形式を読み込みます。
	NLMNILTlw.d 入力形式 (p. 822)	リトアニアの国際表現の通貨出力形式を読み込みます。
	NLMNILVLw.d 入力形式 (p. 823)	ラトビアの国際表現の通貨出力形式を読み込みます。
	NLMNIMOPw.d 入力形式 (p. 824)	マカオの国際表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNIMXNw.d 入力形式 (p. 825)	メキシコの国際表現の通貨出力形式を読み込みます。
	NLMNIMYRw.d 入力形式 (p. 827)	マレーシアの国際表現の通貨出力形式を読み込みます。
	NLMNINOKw.d 入力形式 (p. 828)	ノルウェーの国際表現の通貨出力形式を読み込みます。
	NLMNINZDw.d 入力形式 (p. 829)	ニュージーランドの国際表現の通貨出力形式を読み込みます。
	NLMNIPLNw.d 入力形式 (p. 830)	ポーランドの国際表現の通貨出力形式を読み込みます。
	NLMNIRUBw.d 入力形式 (p. 832)	ロシアの国際表現の通貨出力形式を読み込みます。
	NLMNISEKw.d 入力形式 (p. 833)	スウェーデンの国際表現の通貨出力形式を読み込みます。
	NLMNISGDw.d 入力形式 (p. 834)	シンガポールの国際表現の通貨出力形式を読み込みます。
	NLMNITHBw.d 入力形式 (p. 835)	タイの国際表現の通貨出力形式を読み込みます。
	NLMNITRYw.d 入力形式 (p. 837)	トルコの国際表現の通貨出力形式を読み込みます。
	NLMNITWDw.d 入力形式 (p. 838)	台湾の国際表現の通貨出力形式を読み込みます。
	NLMNIUSDw.d 入力形式 (p. 839)	プエルトリコと米国の国際表現の通貨出力形式を読み込みます。
	NLMNIZARw.d 入力形式 (p. 840)	南アフリカの国際表現の通貨出力形式を読み込みます。
	NLMNLAEDw.d 入力形式 (p. 842)	アラブ首長国連邦の現地表現の通貨出力形式を読み込みます。
	NLMNLAUDw.d 入力形式 (p. 843)	オーストラリアの現地表現の通貨出力形式を読み込みます。
	NLMNLBGWw.d 入力形式 (p. 844)	ブルガリアの現地表現の通貨出力形式を読み込みます。
	NLMNLBRLw.d 入力形式 (p. 845)	ブラジルの現地表現の通貨出力形式を読み込みます。
	NLMNLCADw.d 入力形式 (p. 847)	カナダの現地表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNLCHFw.d 入力形式 (p. 848)	リヒテンシュタインとスイスの現地表現の通貨出力形式を読み込みます。
	NLMNLCNYw.d 入力形式 (p. 849)	中国の現地表現の通貨出力形式を読み込みます。
	NLMNLCZKw.d 入力形式 (p. 850)	チェコの現地表現の通貨出力形式を読み込みます。
	NLMNLDKKw.d 入力形式 (p. 852)	デンマーク、フェロー諸島およびグリーンランドの現地表現の通貨出力形式を読み込みます。
	NLMNLEEKw.d 入力形式 (p. 853)	エストニアの現地表現の通貨出力形式を読み込みます。
	NLMNLEGPw.d 入力形式 (p. 854)	エジプトの現地表現の通貨出力形式を読み込みます。
	NLMNLEURw.d 入力形式 (p. 855)	オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地表現の通貨出力形式を読み込みます。
	NLMNLGBPw.d 入力形式 (p. 857)	英国の現地表現の通貨出力形式を読み込みます。
	NLMNLHKDw.d 入力形式 (p. 858)	香港の現地表現の通貨出力形式を読み込みます。
	NLMNLHRKw.d 入力形式 (p. 859)	クロアチアの現地表現の通貨出力形式を読み込みます。
	NLMNLHUFw.d 入力形式 (p. 861)	ハンガリーの現地表現の通貨出力形式を読み込みます。
	NLMNLIDRw.d 入力形式 (p. 862)	インドネシアの現地表現の通貨出力形式を読み込みます。
	NLMNLILSw.d 入力形式 (p. 863)	イスラエルの現地表現の通貨出力形式を読み込みます。
	NLMNLINRw.d 入力形式 (p. 864)	インドの現地表現の通貨出力形式を読み込みます。
	NLMNLJPYw.d 入力形式 (p. 866)	日本の現地表現の通貨出力形式を読み込みます。
	NLMNLKRWw.d 入力形式 (p. 867)	韓国の現地表現の通貨出力形式を読み込みます。
	NLMNLLTLw.d 入力形式 (p. 868)	リトアニアの現地表現の通貨出力形式を読み込みます。

カテゴリ	言語要素	説明
	NLMNLLVLw.d 入力形式 (p. 869)	ラトビアの現地表現の通貨出力形式を読み込みます。
	NLMNLMOPw.d 入力形式 (p. 871)	マカオの現地表現の通貨出力形式を読み込みます。
	NLMNLMXNw.d 入力形式 (p. 872)	メキシコの現地表現の通貨出力形式を読み込みます。
	NLMNLMYRw.d 入力形式 (p. 873)	マレーシアの現地表現の通貨出力形式を読み込みます。
	NLMNLNOKw.d 入力形式 (p. 874)	ノルウェーの現地表現の通貨出力形式を読み込みます。
	NLMNLNZDw.d 入力形式 (p. 876)	ニュージーランドの現地表現の通貨出力形式を読み込みます。
	NLMNLPLNw.d 入力形式 (p. 877)	ポーランドの現地表現の通貨出力形式を読み込みます。
	NLMNLRUBw.d 入力形式 (p. 878)	ロシアの現地表現の通貨出力形式を読み込みます。
	NLMNLSEKw.d 入力形式 (p. 879)	スウェーデンの現地表現の通貨出力形式を読み込みます。
	NLMNLSGDw.d 入力形式 (p. 881)	シンガポールの現地表現の通貨出力形式を読み込みます。
	NLMNLTHBw.d 入力形式 (p. 882)	タイの現地表現の通貨出力形式を読み込みます。
	NLMNLTRYw.d 入力形式 (p. 883)	トルコの現地表現の通貨出力形式を読み込みます。
	NLMNLTWDw.d 入力形式 (p. 884)	台湾の現地表現の通貨出力形式を読み込みます。
	NLMNLUSDw.d 入力形式 (p. 886)	プエルトリコと米国の現地表現の通貨出力形式を読み込みます。
	NLMNLZARw.d 入力形式 (p. 887)	南アフリカの現地表現の通貨出力形式を読み込みます。
	NLMNYw.d 入力形式 (p. 888)	現地表現の通貨出力形式を、現在の SAS ロケールに適した数値として読み取ります。
	NLMNYIw.d 入力形式 (p. 890)	通貨データを、現在の SAS ロケールに適した国際表現の数値として読み込みます。
	NLNUMw.d 入力形式 (p. 892)	現地表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ	言語要素	説明
	NLNUMIw.d 入力形式 (p. 893)	国際表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。
	NLPCTw.d 入力形式 (p. 895)	現地表現の百分率のデータを現在の SAS ロケールに適した数値として読み取ります。
	NLPCTIw.d 入力形式 (p. 897)	国際表現の百分率のデータを、現在の SAS ロケールに適した数値として読み取ります。
	NLSTRMON 入力形式 (p. 898)	月名を、現在の SAS ロケールに適した数値として読み込みます。
	YENw.d 入力形式 (p. 931)	埋め込まれた円記号、カンマおよび小数点を削除します。
日付と時間	JNENGOW. 入力形式 (p. 780)	yymmdd 形式の日本の漢字の日付値を読み込みます。
	MINGUOW. 入力形式 (p. 786)	台湾の形式の日付を読み込みます。
	NENGOW. 入力形式 (p. 788)	eyymmdd 形式の日本の日付値を読み込みます。
	NLDATEw. 入力形式 (p. 789)	日付値を、現在の SAS ロケールに適したローカルの SAS 日付値として読み取ります。
	NLDATEWw. 入力形式 (p. 791)	日付値を、現在の SAS ロケールに適したローカルの SAS 日付と曜日の形式で読み取ります。
	NLDATMw. 入力形式 (p. 792)	現在の SAS ロケールに適した日時値を読み込みます。
	NLDATMAP 入力形式 (p. 793)	日時値を、現在の SAS ロケールに適した現地の SAS 日時値 (午前または午後の指定付き)として読み取ります。
	NLDATMWw. 入力形式 (p. 794)	日付値を、現在の SAS ロケールに適した現地の SAS の曜日と日時として読み取ります。
	NLTIMAPw. 入力形式 (p. 900)	時間値を、a.m.および p.m.表示を使用し現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。
	NLTIMEw. 入力形式 (p. 901)	時間値を、現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。
ヘブライ語のテキストの操作	\$CPTDWw. 入力形式 (p. 773)	ヘブライ語 DOS (cp862)エンコーディングの文字列を読み込み、その文字列を Windows (cp1255)エンコーディングに変換します。
	\$CPTWDw. 入力形式 (p. 774)	Windows (cp1255)エンコーディングの文字列を読み込み、その文字列をヘブライ語 DOS (cp862)エンコーディングに変換します。
文字	\$REVERJw. 入力形式 (p. 902)	文字データを右から左の順に読み込み、空白を保持します。

カテゴリ	言語要素	説明
	\$REVERSw. 入力形式 (p. 903)	文字データを右から左の順に左詰めで読み込みます。
	\$UCS2Bw. 入力形式 (p. 904)	ビッグエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS2BEw. 入力形式 (p. 906)	現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列をビッグエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。
	\$UCS2Lw. 入力形式 (p. 907)	リトルエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS2LEw. 入力形式 (p. 908)	現在の SAS セッションのエンコーディングにある文字列を読み込み、その文字列をリトルエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。
	\$UCS2Xw. 入力形式 (p. 910)	16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS2XEw. 入力形式 (p. 911)	現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列を、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。
	\$UCS4Bw. 入力形式 (p. 912)	ビッグエンディアン、32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS4Lw. 入力形式 (p. 913)	リトルエンディアン、32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS4Xw. 入力形式 (p. 915)	32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UCS4XEw. 入力形式 (p. 916)	現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列を、32 ビット、UCS4 形式の UNICODE(ユニコード)に変換します。
	\$UESCw. 入力形式 (p. 917)	UESC 表記でエンコードされた文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UESCEw. 入力形式 (p. 919)	現在の SAS セッションのエンコーディングを使用する文字列を読み取り、その文字列を UESC 表記に変換します。
	\$UNCRw. 入力形式 (p. 920)	NCR 文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UNCREw. 入力形式 (p. 921)	現在の SAS セッションのエンコーディングを使用する文字列を読み込み、その文字列を NCR に変換します。

カテゴリ	言語要素	説明
	\$UPARENw. 入力形式 (p. 923)	UPAREN 表記でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。
	\$UPARENEW. 入力形式 (p. 924)	現在の SAS セッションのエンコーディングを使用する文字列を読み込み、その文字列を UPAREN 表記に変換します。
	\$UPARENpw. 入力形式 (p. 926)	UPAREN 表記でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換しますが、各国語文字は UPAREN 表記のエンコーディングのまま残ります。
	\$UTF8Xw. 入力形式 (p. 927)	UTF-8 でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

ディクショナリ

\$CPTDWw. 入力形式

ヘブライ語 DOS (cp862)エンコーディングの文字列を読み込み、その文字列を Windows (cp1255)エンコーディングに変換します。

カテゴリ: ヘブライ語のテキストの操作

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$CPTDWw.

構文の説明

w

入力幅を指定します。

デフォルト 200

範囲 1-32767

比較

\$CPTDWw.入力形式では、\$CPTWDw.入力形式とは逆の処理を行います。

例

次の例では、入力値として 808182 を使用します。

ステートメント	結果
	----+----1----+
x=input('808182',\$cptdw6.); put x;	808182

関連項目

出力形式:

- ["\\$CPTDWw. 出力形式" \(152 ページ\)](#)
- ["\\$CPTWDw. 出力形式" \(153 ページ\)](#)

入力形式:

- ["\\$CPTWDw. 入力形式" \(774 ページ\)](#)

\$CPTWDw. 入力形式

Windows (cp1255)エンコーディングの文字列を読み込み、その文字列をヘブライ語 DOS (cp862)エンコーディングに変換します。

カテゴリ: ヘブライ語のテキストの操作

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$CPTWDw.

構文の説明

w
入力幅を指定します。

デフォルト 200

範囲 1-32767

比較

\$CPTWDw.入力形式では、\$CPTDWw.入力形式とは逆の処理を行います。

例

次の例では、入力値として 123456789 を使用します。

ステートメント	結果
	-----1-----+
x=input(' 123456789',\$cptwd6.); put x;	€12,345,678.90

関連項目

- 出力形式:**
- ["\\$CPTWDw. 出力形式" \(153 ページ\)](#)
 - ["\\$CPTDWw. 出力形式" \(152 ページ\)](#)
- 入力形式:**
- ["\\$CPTDWw. 入力形式" \(773 ページ\)](#)

EUROw.d 入力形式

数値を読み込み、ヨーロッパ通貨の埋め込み文字を取り除いて、カンマと小数点を逆にします。

カテゴリ: CAS

数値

注:

2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

EUROw.d

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

値を除算する 10 のべき乗を指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

EUROw.d 入力形式では数値を読み込み、埋め込みユーロ記号(E)、カンマ、空白、パーセント記号、ハイフン、閉じかっこを入力データから削除します。小数点は、実数部と小数部を分ける区切り文字と見なされます。EUROw.d 入力形式では、フィールドの先頭にある開きかっこがマイナス記号に変換されます。

比較

- EUROw.d 入力形式は EUROXwd 入力形式に似ていますが、EUROXw.d では小数点とカンマの役割が逆になっています。この規則はヨーロッパの国で共通です。
- 入力にカンマもピリオドもない場合、EUROw.d 入力形式と EUROXw.d 入力形式は相互に交換可能です。

例

次の表に、ユーロでの通貨の入力値、適用される SAS ステートメントおよび結果を示します。

```
data _null_;
  input x euro10.;
  put x=;
  datalines;
E1
E1.23
1.23
1,234.56
;
run;
SAS Log:
x=1
x=1.23
x=1.23
x=1234.56
```

値	ステートメント	結果
		----+----1----2
E1	input x euro10.; put x;	1
E1.23	input x euro10.; put x;	1.23
1.23	input x euro10.; put x;	1.23
1,234.56	input x euro10.; put x;	1234.56

関連項目

出力形式:

- [“EUROw.d 出力形式” \(163 ページ\)](#)
- [“EUROXw.d 出力形式” \(166 ページ\)](#)

入力形式:

- [“EUROXw.d 入力形式” \(778 ページ\)](#)

EUROXw.d 入力形式

数値を読み込み、ヨーロッパ通貨の埋め込み文字を削除します。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

EUROXw.d

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

値を除算する 10 のべき乗を指定します。データに小数点を表すカンマが含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

EUROXw.d 入力形式では数値を読み込み、埋め込みユーロ記号(E)、ピリオド、空白、パーセント記号、ハイフン、閉じかっこを入力データから削除します。カンマは、実数部と小数部を分ける区切り文字と見なされます。EUROXw.d 入力形式では、フィールドの先頭にある開きかっこがマイナス記号に変換されます。

比較

- EUROXw.d 入力形式は EUROW.d 入力形式に似ていますが、EUROW.d では、カンマと小数点の役割が逆になっています。この規則は英語圏の国で共通です。
- 入力にカンマもピリオドもない場合、EUROXw.d 入力形式と EUROW.d 入力形式は相互に交換可能です。

例

次の表に、ユーロでの通貨の入力値、適用される SAS ステートメントおよび結果を示します。

```
data _null_;
  input x eurox10.;
  put x=;
  datalines;
E1
E1.23
1.23
1,234.56
; run;
SAS Log:
7  input x eurox10.;
8  put x=;
9  datalines;
x=1
x=123
x=123
x=1.23456
```

値	ステートメント	結果
		----+----1----2
E1	input x eurox10.; put x;	1
E1.23	input x eurox10.; put x;	123
1.23	input x eurox10.; put x;	123
1,234.56	input x eurox10.; put x;	1.23456

関連項目

出力形式:

- “EUROw.d 出力形式” (163 ページ)
- “EUROXw.d 出力形式” (166 ページ)

入力形式:

- “EUROw.d 入力形式” (775 ページ)

JNENGOW. 入力形式

yymmdd 形式の日本の漢字の日付値を読み込みます。

カテゴリ: 日付と時間

配置: 左

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

JNENGOW.

構文の説明

w

入力幅を指定します。

デフォルト 16

範囲 16–32

詳細

JNENGOW.入力形式は、日本語の漢字値を *yymmdd* の形式で読み取ります。年、月、日の値は、空白や特殊文字で区切ることができます。入力幅には、空白および特殊文字のスペースを考慮する必要があります。

注: SAS では、2 桁の年は YEARCUTOFF=システムオプションで定義された 100 年の期間内であると解釈します。

例

次の例は、JNENGO 入力形式を使用して漢字値を SAS 日付値に変換する方法を示しています。データ行の日付値は特殊文字で区切られます。

```
data _null_;
  input x jnengo.;
  datalines;
  明治1年4月6日
  明治45年7月29日
  大正1年7月30日
  大正15年12月24日
  昭和1年12月25日
  昭和64年 1月 7日
  平成1年1月8日
  平成10年12月 8日
  ;
```

\$KANJIw. 入力形式

DBCS データからシフトコードデータを削除します。

カテゴリ: DBCS

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$KANJIw.

構文の説明

w

入力幅を指定します。

範囲 入力形式の最小幅は 2 です。

制限事項 幅は偶数幅(2 バイト単位)で指定します。奇数幅を指定すると幅が切り捨てられます。この幅は、シフトコードデータの長さ以上にする必要があります。

詳細

\$KANJI 入力形式では、DBCS データからシフトコードデータが削除されます。
\$KANJI 入力形式では、ホストメインフレームデータが処理されます。\$KANJI は、他のプラットフォームで使用できます。\$KANJI 入力形式を非 EBCDIC (モーダルエンコーディングではない) ホストで使用する場合、データは変化しません。

SBCS の空データが返された場合を除き、データは SO で開始し、SI で終了する必要があります。入力データ長は、 $2 + (SO/SI \text{ length}) * 2$ である必要があります。

関連項目

出力形式:

- ["\\$KANJIw. 出力形式" \(177 ページ\)](#)
- ["\\$KANJIXw. 出力形式" \(178 ページ\)](#)

入力形式:

- ["\\$KANJIXw. 入力形式" \(782 ページ\)](#)

\$KANJIXw. 入力形式

シフトコードデータを DBCS データに追加します。

カテゴリ: DBCS

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$KANJIX_w.

構文の説明

w

入力幅を指定します。

範囲 入力形式の最小幅は、 $2 + (\text{現在の DBCSTYPE=設定で使用されるシフトコードの長さ}) * 2$ です。

制限事項 幅は偶数幅(2 バイト単位)で指定します。奇数幅を指定すると幅が切り捨てられます。この幅は、シフトコードデータの長さ以上にする必要があります。

詳細

\$KANJI 入力形式では、シフトコードデータが、シフトコードデータのない DBCS データに追加されます。入力データが空白の場合、シフトコードデータは追加されません。\$KANJI 入力形式で処理されるのはホストメインフレームデータですが、\$KANJI は他のプラットフォームで使用できます。\$KANJI 入力形式を非 EBCDIC (モーダルエンコーディングではない) ホストで使用する場合、データは変化しません。

関連項目

出力形式:

- “\$KANJIw. 出力形式” (177 ページ)
- “\$KANJIxw. 出力形式” (178 ページ)

入力形式:

- “\$KANJIw. 入力形式” (781 ページ)

\$LOGVSw. 入力形式

左から右に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。

カテゴリ: BIDI テキストの操作
CAS

構文

\$LOGVSw.

構文の説明

w

入力幅を指定します。

デフォルト 200

範囲 1-32767

比較

\$LOGVSw.入力形式では、LOGVSRw.入力形式とは逆の処理を行います。

例

次の例では、“טוֹסֵף flight”のヘブライ語の入力値を使用します。

ステートメント	結果
	----+----1----+
x=input ('טוֹסֵף flight',\$logvs12.); put x;	טוֹסֵף flight

次の例では、“كاسوت copmuter”のアラビア語の入力値を使用します。

ステートメント	結果
	----+----1----+
x=input ('كاسوت computer',\$logvs12.); put x;	كاسوت コンピューター

関連項目

出力形式:

- [“\\$LOGVSRw. 出力形式” \(181 ページ\)](#)
- [“\\$LOGVSw. 出力形式” \(179 ページ\)](#)

入力形式:

- [“\\$LOGVSRw. 入力形式” \(785 ページ\)](#)

\$LOGVSRw. 入力形式

右から左に論理的順序で並ぶ文字列を読み込み、その文字列を視覚的順序に変換します。

カテゴリ: BIDI テキストの操作
CAS

構文

\$LOGVSRw.

構文の説明

w
入力幅を指定します。
デフォルト 200
範囲 1-32767

比較

\$LOGVSRw.入力形式では、\$LOGVSw.入力形式とは逆の処理を行います。

例

次の例では、“תִּסְרֵן flight”のヘブライ語の入力値を使用します。

ステートメント	結果
	-----1-----+
x=input (' תִּסְרֵן flight', \$logvsr12.); put x;	flight תִּסְרֵן

次の例では、“تاذ computer”のアラビア語の入力値を使用します。

ステートメント	結果
	----+----1----+
<pre>x=input (' ٱٱٱٱ computer',\$logvsr12.); put x;</pre>	ٱٱٱٱ コンピューター

関連項目

出力形式:

- “\$LOGVSw. 出力形式” (179 ページ)
- “\$LOGVSRw. 出力形式” (181 ページ)

入力形式:

- “\$LOGVSw. 入力形式” (783 ページ)

MINGUOW. 入力形式

台湾の形式の日付を読み込みます。

カテゴリ: 日付と時間

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

MINGUOW.

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲

6-10

詳細

台湾の日付の通常の形式は、*yyyymmdd* です。

yyyy

年を表す整数です。

mm

月を表す 01 から 12 の整数です。

dd

月の日を表す 01 から 31 の整数です。

台湾のカレンダーでは、1912 年を基準年(1912 年 1 月 1 日が 01/01/01)として使用します。1912 年より前の日付は無効です。年の値は 100 年周期で繰り返されることはありません。年の値は、継続して増加していきます。

年、月、日の値は、YYMMDDw.入力形式で使用可能な、空白、スラッシュ、ハイフンなどの区切り文字で区切ることができます。区切り文字を使用する場合は、すべての値の間に挿入します。区切り文字を省略する場合は、値が 10 未満の日または月の先頭に 0 を使用してください。

例

次の例では、入力値に異なる日付を使用します。

```
input date minguo10.;
put date date9.;

data _null_;
  input date minguo10.;
  put date date9.;
  datalines;
49/01/01
891215
03-01-01
;
```

値	結果
	----+----1----+
49/01/01	01JAN1960
891215	15DEC2000
103-01-01	01JAN2014

関連項目

出力形式:

- [“MINGUOw. 出力形式” \(182 ページ\)](#)

入力形式:

- [“YYMMDDw. 入力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)

NENGOW. 入力形式

eyymmdd 形式の日本の日付値を読み込みます。

カテゴリ: CAS

日付と時間

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NENGOW.

構文の説明

w

入力幅を指定します。

デフォルト 10

範囲 7-32

詳細

日本の日付の通常の形式は、*eyymmdd* です。

e

元号(明治、大正、昭和、平成、または令和)の先頭の文字(M、T、S、H、R)です。

yy

年を表す整数です。

mm

月を表す 01 から 12 の整数です。

dd

月の日を表す 01 から 31 の整数です。

e の値と整数はピリオドで区切ることができます。*e* を省略すると、現在の元号が使用されます。年、月、日の値は、空白や数値以外の文字で区切ることができます。ただし、区切り文字を使用する場合は、すべての値の間に挿入します。区切り文字を省略する場合は、値が 10 未満の日または月の先頭に 0 を使用してください。

例

次の例では、異なる入力値を使用します。

```
data _null_;
input x nengo.;
put x= date9.;
datalines;
h11108
h.11108
11/10/08
;
run;
```

```
x=08OCT1999
x=08OCT1999
x=08OCT1999
```

関連項目

出力形式:

- [“NENGOW. 出力形式” \(184 ページ\)](#)

NLDATEw. 入力形式

日付値を、現在の SAS ロケールに適したローカルの SAS 日付値として読み取ります。

カテゴリ: CAS
日付と時間

別名: NLDATEW

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるよ

うになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATE_w.

構文の説明

w

入力幅を指定します。

デフォルト 10

範囲 10-200

例

次の例では、入力値として February 24, 2003 を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; dy='February 24, 2003'; y=input(dy,nldate200.); put y;	15760
options locale=German_Germany; dy='24. Februar 2003'; y=input(dy,nldate16.); put y;	15760

関連項目

出力形式:

■ [“NLDATE_w. 出力形式” \(190 ページ\)](#)

NLDATEWw. 入力形式

日付値を、現在の SAS ロケールに適したローカルの SAS 日付と曜日の形式で読み取ります。

- カテゴリ: CAS
日付と時間
- 配置: 左
- 注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATEWw.

構文の説明

w

入力幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 20

範囲 16-200

例

次の例では、入力値として February 24, 2014 を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; data; dy='Monday, February 24, 2014'; y=input(dy,nldatew200.); put y; run;	19778
options locale=German_Germany; dy='Mo. 24.Februar 2014';	19778

ステートメント	結果
---------	----

```
y=input(dy,nldatew16.);
put y;
run;
```

NLDATMw. 入力形式

現在の SAS ロケールに適した日時値を読み込みます。

カテゴリ: CAS

日付と時間

別名: NLDATMW、NLDATMAP

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATM_w.

構文の説明

w

入力幅を指定します。

デフォルト 19

範囲 19-200

例

次の例では、入力値として February 24, 2003 12:39:43 を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates;	1361709583
y=input('24.Feb03:12:39:43', nldatm.);	

ステートメント	結果
put y=;	
options locale=German_Germany;	1330171200
y=input('24.Februar 2003 12.39 Uhr', nldatm.);	
put y=;	

関連項目

出力形式:

- [“NLDATMw. 出力形式” \(222 ページ\)](#)

NLDATMAP 入力形式

日時値を、現在の SAS ロケールに適した現地の SAS 日時値(午前または午後の指定付き)として読み取ります。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMAP_w.

構文の説明

w

入力幅を指定します。必要に応じて、出力形式の幅に合うように日時値が短縮されます。

デフォルト 32

範囲 17-200

例

これらの例では、入力値として February 24, 2014 の 12:39:43 p.m.を使用します。

ステートメント	結果
	----+----1----+----2----+----3
options locale=English_UnitedStates; data; dy='February 24, 2014 12:39:43 PM'; y=input(dy,nldatmap200.); put y; run;	1708864783
options locale=Spanish_Mexico; data; dy='24/02/2003 12:39:43 PM'; y=input(dy,nldatmap200.); put y; run;	1708864783

NLDATMWw. 入力形式

日付値を、現在の SAS ロケールに適した現地の SAS の曜日と日時として読み取ります。

カテゴリ: CAS

日付と時間

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLDATMWw

構文の説明

w

入力幅を指定します。必要に応じて、出力形式の幅に合うように日付が短縮されます。

デフォルト 40
範囲 34-200

例

次の例では、入力値として Mon, Feb 24, 2014 12:39:43PM を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; data; dy='Mon, Feb 24, 2014 12:39:43 PM'; y=input(dy,nldatmw200.); put y; run;	1708864783
options locale=German_Germany; dy='Mo, 24. Feb 2014 12.39 Uhr'; y=input(dy,nldatmw16.); put y; run;	1708864783

NLMNIAEDw.d 入力形式

アラブ首長国連邦の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIAEDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 3

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniaed32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLAEDw.d 入力形式” \(842 ページ\)](#)

NLMNIAUDw.d 入力形式

オーストラリアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIAUD*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniaud32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLAUDw.d 入力形式” (843 ページ)

NLMNIBGNw.d 入力形式

ブルガリアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIBGNw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=put('(-1234.56789)',nlmnibgn32.2);  
y=put('(-1234.56789)',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLBGNw.d 入力形式” \(844 ページ\)](#)

NLMNIBRLw.d 入力形式

ブラジルの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIBRLw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnibr132.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLBRLw.d 入力形式” \(845 ページ\)](#)

NLMNICADw.d 入力形式

カナダの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICADw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnicad32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNICADw.d 出力形式” \(375 ページ\)](#)

NLMNICHFw.d 入力形式

リヒテンシュタインとスイスの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICHF*w.d*

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnichf32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ “NLMNICHw.d 出力形式” (376 ページ)

NLMNICNYw.d 入力形式

中国の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICNYw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnicny32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNICNYw.d 出力形式” \(378 ページ\)](#)

NLMNICZKw.d 入力形式

チェコの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNICZKw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniczk32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 入力形式:
- [“NLMNLCZKw.d 入力形式” \(850 ページ\)](#)

NLMNIDKKw.d 入力形式

デンマーク、フェロー諸島およびグリーンランドの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIDKKw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmndkk32.2);
y=input('$12,345.67'),dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIDKKw.d 出力形式” \(380 ページ\)](#)

NLMNIEEKw.d 入力形式

エストニアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEEKw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnieek32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLEEKw.d 入力形式” (853 ページ)

NLMNIEGPw.d 入力形式

エジプトの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEGPw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 3

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniegp32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLEGPw.d 入力形式” \(854 ページ\)](#)

NLMNIEURw.d 入力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIEUR*w.d*

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnieur32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 出力形式:**
- [“NLMNIEURw.d 出力形式” \(385 ページ\)](#)

NLMNIGBPw.d 入力形式

英国の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIGBPw.d

構文の説明

w
入力幅を指定します。
デフォルト 12
範囲 8-32

d
10^d で数値を除算するよう指定します。データに小数点が含まれる場合、d 値は無視されます。
デフォルト 2
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnigbp32.2);  
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 出力形式:
- [“NLMNIGBPw.d 出力形式” \(386 ページ\)](#)

NLMNIHKDw.d 入力形式

香港の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHKDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnihk32.2);  
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIHKDw.d 出力形式” \(388 ページ\)](#)

NLMNIHRKw.d 入力形式

クロアチアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHRKw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnihrk32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLHRKw.d 入力形式” \(859 ページ\)](#)

NLMNIHUFw.d 入力形式

ハンガリーの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIHUFw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniuhf32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLHUFw.d 入力形式” \(861 ページ\)](#)

NLMNIIDRw.d 入力形式

インドネシアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIIDR $w.d$

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniidr32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLIDRw.d 入力形式” (862 ページ)

NLMNIILSw.d 入力形式

イスラエルの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIILSw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmniils32.2);  
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIILSw.d 出力形式” \(393 ページ\)](#)

NLMNIINRw.d 入力形式

インドの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIINRw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniinr32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 入力形式:
- [“NLMNLINRw.d 入力形式” \(864 ページ\)](#)

NLMNIJPYw.d 入力形式

日本の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIJPYw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnijpy32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIJPYw.d 出力形式” \(396 ページ\)](#)

NLMNIKRWw.d 入力形式

韓国の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIKRWw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnkrw32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLKRWw.d 入力形式” (867 ページ)

NLMNILTlw.d 入力形式

リトアニアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNILTlw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmniltl32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLLTLw.d 入力形式” \(868 ページ\)](#)

NLMNILVLw.d 入力形式

ラトビアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNILVLw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnilvl32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 入力形式:
- [“NLMNLLVLw.d 入力形式” \(869 ページ\)](#)

NLMNIMOPw.d 入力形式

マカオの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIMOPw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnimop32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ [“NLMNLMOPw.d 入力形式” \(871 ページ\)](#)

NLMNIMXNw.d 入力形式

メキシコの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIMXN $w.d$

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnimxn32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLMXNw.d 入力形式” (872 ページ)

NLMNIMYRw.d 入力形式

マレーシアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIMYRw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnimyr32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIMYRw.d 出力形式” \(405 ページ\)](#)

NLMNINOKw.d 入力形式

ノルウェーの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNINOKw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmninok32.2);  
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 出力形式:
- [“NLMNINOKw.d 出力形式” \(406 ページ\)](#)

NLMNINZDw.d 入力形式

ニュージーランドの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNINZDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmninzd32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ [“NLMNINZDw.d 出力形式” \(407 ページ\)](#)

NLMNIPLNw.d 入力形式

ポーランドの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIPLNw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnipln32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント

結果

----+----1----+

put x=;

-12345.67

put y=;

-12345.67

関連項目

出力形式:

■ “NLMNIPLNw.d 出力形式” (409 ページ)

NLMNIRUBw.d 入力形式

ロシアの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIRUBw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnirub32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIRUBw.d 出力形式” \(410 ページ\)](#)

NLMNISEKw.d 入力形式

スウェーデンの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNISEKw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnisek32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNISEKw.d 出力形式” \(412 ページ\)](#)

NLMNISGDw.d 入力形式

シンガポールの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNISGDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnisgd32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNISGDw.d 出力形式” \(413 ページ\)](#)

NLMNITHBw.d 入力形式

タイの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNITHB*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnithb32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNLTHBw.d 入力形式” (882 ページ)

NLMNITRYw.d 入力形式

トルコの国際表現の通貨出力形式を読み込みます。

カテゴリ:	CAS 数値
配置:	左
注:	2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNITRYw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnitry32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNLTRYw.d 入力形式” \(883 ページ\)](#)

NLMNITWDw.d 入力形式

台湾の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNITWDw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnitwd32.2);  
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 出力形式:
- [“NLMNITWDw.d 出力形式” \(417 ページ\)](#)

NLMNIUSDw.d 入力形式

プエルトリコと米国の国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIUSDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmniusd32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNIUSDw.d 出力形式” \(419 ページ\)](#)

NLMNIZARw.d 入力形式

南アフリカの国際表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNIZARw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67'),nlmnizar32.2);
y=input('$12,345.67')dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ “NLMNIZARw.d 出力形式” (420 ページ)

NLMNLAEDw.d 入力形式

アラブ酋長国連邦の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLAEDw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 3

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnaed32.2);
y=input('$12,345.67',dollar32.2);
```


ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIAEDw.d 入力形式” \(795 ページ\)](#)

NLMNLAUDw.d 入力形式

オーストラリアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLAUDw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlud32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLAUDw.d 出力形式” \(423 ページ\)](#)

NLMNLBGNw.d 入力形式

ブルガリアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLBGNw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input(-12345.67,nlmnlbgn32.2);
y=input(-12345.67,dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ [“NLMNIBGNw.d 入力形式” \(798 ページ\)](#)

NLMNLBRLw.d 入力形式

ブラジルの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLBRL*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlbrl32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNIBRLw.d 入力形式” (799 ページ)

NLMNLCADw.d 入力形式

カナダの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCADw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlcad32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLCADw.d 出力形式” \(427 ページ\)](#)

NLMNLCHFw.d 入力形式

リヒテンシュタインとスイスの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCHFw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlchf32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 出力形式:
- [“NLMNLCHFW.d 出力形式” \(429 ページ\)](#)

NLMNLCNYw.d 入力形式

中国の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCNYw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlcny32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLCNYw.d 出力形式” \(430 ページ\)](#)

NLMNLCZKw.d 入力形式

チェコの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLCZKw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlczk32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNICZKw.d 入力形式” (804 ページ)

NLMNLDKKw.d 入力形式

デンマーク、フェロー諸島およびグリーンランドの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLDKKw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlldkk32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLDKKw.d 出力形式” \(433 ページ\)](#)

NLMNLEEKw.d 入力形式

エストニアの現地表現の通貨出力形式を読み込みます。

- カテゴリ:

CAS
- 数値
- 配置:

左
- 注:

2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEEKw.d

構文の説明

- w

入力幅を指定します。

デフォルト 12

範囲 8-32
- d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、d 値は無視されます。

デフォルト 4
範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnleek32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

- 入力形式:
- [“NLMNIEEKw.d 入力形式” \(806 ページ\)](#)

NLMNLEGPw.d 入力形式

エジプトの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEGPw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 3

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmlegp32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIEGPw.d 入力形式” \(808 ページ\)](#)

NLMNLEURw.d 入力形式

オーストリア、ベルギー、キプロス、エストニア、フィンランド、フランス、ドイツ、ギリシャ、アイルランド、イタリア、ラトビア、リトアニア、ルクセンブルク、マルタ、オランダ、ポルトガル、スロバキア、スロベニア、スペインの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS

数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLEUR*w.d*

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnleur32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLEURw.d 出力形式” \(437 ページ\)](#)

NLMNLGBPw.d 入力形式

英国の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLGBPw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlgbp32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLGBPw.d 出力形式” \(439 ページ\)](#)

NLMNLHKDw.d 入力形式

香港の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHKDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlhkd32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLHKDw.d 出力形式” \(440 ページ\)](#)

NLMNLHRKw.d 入力形式

クロアチアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHRK w,d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlhrk32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-1,234.57

関連項目

入力形式:

- [“NLMNIHRK \$w,d\$ 入力形式” \(813 ページ\)](#)

NLMNLHUFw.d 入力形式

ハンガリーの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLHUFw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlhuf32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIHUFw.d 入力形式” \(814 ページ\)](#)

NLMNLIDRw.d 入力形式

インドネシアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLIDRw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlidr32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIIDRw.d 入力形式” \(815 ページ\)](#)

NLMNLILSw.d 入力形式

イスラエルの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS

数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLILSw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlils32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLILSw.d 出力形式” \(446 ページ\)](#)

NLMNLINRw.d 入力形式

インドの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLINRw.d

構文の説明

w
 入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlr32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNIINRw.d 入力形式” (818 ページ)

NLMNLJPYw.d 入力形式

日本の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLJPY w . d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10 d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 0

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnljpy32.2);
y=input('$12,345.67',dollar32.2);
```


ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLJPYw.d 出力形式” \(449 ページ\)](#)

NLMNLKRWw.d 入力形式

韓国の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLKRWw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlkrw32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIKRWw.d 入力形式” \(820 ページ\)](#)

NLMNLLTLw.d 入力形式

リトアニアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLLTLw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnl32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ [“NLMNILLw.d 入力形式” \(822 ページ\)](#)

NLMNLLVLw.d 入力形式

ラトビアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLLVL w . d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10 d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlvl(32.2));
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ “NLMNILVLw.d 入力形式” (823 ページ)

NLMNLMOPw.d 入力形式

マカオの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLMOPw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlmop32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIMOPw.d 入力形式” \(824 ページ\)](#)

NLMNLMXNw.d 入力形式

メキシコの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLMXNw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlmxn32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNIMXNw.d 入力形式” \(825 ページ\)](#)

NLMNLMYRw.d 入力形式

マレーシアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS

数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLMYRw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlmyr32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ [“NLMNLMYRw.d 出力形式” \(457 ページ\)](#)

NLMNLNOKw.d 入力形式

ノルウェーの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLNOKw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlnok32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ “NLMNLNOKw.d 出力形式” (459 ページ)

NLMNLNZDw.d 入力形式

ニュージーランドの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLNZDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlzdz32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLNZDw.d 出力形式” \(460 ページ\)](#)

NLMNLPLNw.d 入力形式

ポーランドの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLPLNw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlpln32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLRUBw.d 出力形式” \(462 ページ\)](#)

NLMNLRUBw.d 入力形式

ロシアの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLRUBw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlrub32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLRUBw.d 出力形式” \(463 ページ\)](#)

NLMNLSEKw.d 入力形式

スウェーデンの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLSEK $w.d$

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlsek32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ “NLMNLSEKw.d 出力形式” (464 ページ)

NLMNLSGDw.d 入力形式

シンガポールの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS

数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLSGDw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlsgd32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLSGDw.d 出力形式” \(466 ページ\)](#)

NLMNLTHBw.d 入力形式

タイの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTHBw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlthb32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

- [“NLMNITHBw.d 入力形式” \(835 ページ\)](#)

NLMNLTRYw.d 入力形式

トルコの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTRYw.d

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 4

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnltry32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

入力形式:

■ [“NLMNITRYw.d 入力形式” \(837 ページ\)](#)

NLMNLTWDw.d 入力形式

台湾の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注:

2025.12以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLTWD*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 12

範囲 8-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnltd32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

■ “NLMNLTWDw.d 出力形式” (470 ページ)

NLMNLUSDw.d 入力形式

プエルトリコと米国の現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLUSDw.d

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
10^d で数値を除算するよう指定します。データに小数点が含まれる場合、*d* 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlusd32.2);  
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	----+----1----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLUSDw.d 出力形式” \(471 ページ\)](#)

NLMNLZARw.d 入力形式

南アフリカの現地表現の通貨出力形式を読み込みます。

カテゴリ: CAS
数値

配置: 左

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNLZAR $w.d$

構文の説明

w
入力幅を指定します。

デフォルト 12

範囲 8-32

d
 10^d で数値を除算するよう指定します。データに小数点が含まれる場合、 d 値は無視されます。

デフォルト 2

範囲 0-28

例

次の例では、LOCALE=システムオプションが English_UnitedStates に設定されています。

```
x=input('$12,345.67',nlmnlzar32.2);
y=input('$12,345.67',dollar32.2);
```

ステートメント	結果
	-----1-----+
put x=;	-12345.67
put y=;	-12345.67

関連項目

出力形式:

- [“NLMNLZARw.d 出力形式” \(473 ページ\)](#)

NLMNYw.d 入力形式

現地表現の通貨出力形式を、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNYw.d

構文の説明

w

入力幅を指定します。

デフォルト 9

範囲 1-32

d

10^d で除算するかどうかを指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLMNYw.d 入力形式では、指定されたロケールで、通貨データを現地表現で読み込み、そのデータを数値に変換します。入力データから、桁区切り文字、小数点区切り文字、空白、通貨記号、閉じかっこを削除します。

比較

NLMNYw.d 入力形式は、DOLLARw.d 入力形式と似ていますが、NLMNYw.d 入力形式はロケール固有である点が違います。

数値データをアポストロフィで囲むと、NLNUM はエラーメッセージを返します。アポストロフィはデータが文字であることを指定します。次の例では、数値 1 がアポストロフィで囲まれているため、エラーメッセージが生成されます。

```
data;
x=input("'1'",NLMNY32.);
put x;
run;
```

例

次の例では、入力値として\$12,345.67を使用します。

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates;	-12345.67
x=input('\$12,345.67',nlmny32);	-12345.67

ステートメント

結果

```
y=input('$12,345.67',dollar32);
put x=;
put y=;
```

関連項目

出力形式:

- [“NLMNYw.d 出力形式” \(474 ページ\)](#)
- [“NLMNYIw.d 出力形式” \(478 ページ\)](#)

入力形式:

- [“NLMNYIw.d 入力形式” \(890 ページ\)](#)

NLMNYIw.d 入力形式

通貨データを、現在の SAS ロケールに適した国際表現の数値として読み込みます。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLMNYI*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 9

範囲 1-32

d

10^d で除算するかどうかを指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLMNYIw.d 入力形式では、指定ロケールで、通貨データを国際表現で読み込み、そのデータを数値に変換します。入力データから、桁区切り文字、小数点区切り文字、空白、通貨記号、閉じかっこを削除します。

例

次の例では、入力値として 12,345.67 を使用します。

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates;	-12345.67
x=input('(USD12,345.67)',nlmnyi32.2);	-12345.67
y=input('\$-12,345.67)',dollar32.2);	
put x=;	
put y=;	

関連項目

出力形式:

- [“NLMNYw.d 出力形式” \(474 ページ\)](#)
- [“NLMNYIw.d 出力形式” \(478 ページ\)](#)

入力形式:

- [“NLMNYw.d 入力形式” \(888 ページ\)](#)

NLNUMw.d 入力形式

現地表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLNUM*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

10^d で除算するかどうかを指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。小数点が見つからない場合、データは整数と見なされます。

デフォルト 0

範囲 0-31

詳細

NLNUM*w.d* 入力形式では、指定されたロケールで、数値データを現地表現で読み込み、そのデータを数値に変換します。入力データから、桁区切り文字、小数点区切り文字、空白、通貨記号、閉じかっこを削除します。

数値データをアポストロフィで囲むと、NLNUM はエラーメッセージを返します。アポストロフィはデータが文字であることを指定します。次の例では、数値 1 がアポストロフィで囲まれているため、エラーメッセージが生成されます。

```
data;
```

```
x=input("'1'",NLNUM32.);
put x=;
run;
```

比較

NLNUMw.d 入力形式 では、NLNUMIw.d 入力形式とは逆の処理を行います。

例

次の例では、入力値として-1234356.78 を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; x=input('1,234,356.78',nlnum32.2); put x=;	-1234356.78

関連項目

出力形式:

- [“NLNUMw.d 出力形式” \(482 ページ\)](#)
- [“NLMNYIw.d 出力形式” \(478 ページ\)](#)

入力形式:

- [“NLNUMIw.d 入力形式” \(893 ページ\)](#)

NLNUMIw.d 入力形式

国際表現の数値データを、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるよ

うになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLNUMI*w.d*

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

10^{*d*} で数値を除算するよう指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLNUMI*w.d* 入力形式では、指定されたロケールで、数値データを国際表現で読み込み、そのデータを数値に変換します。入力データから、桁区切り文字、小数点区切り文字、空白、通貨記号、閉じかっこを削除します。

比較

NLNUMI*w.d* 入力形式 では、NLNUM*w.d* 入力形式とは逆の処理を行います。

例

次の例では、入力値として-1234356.78 を使用します。

ステートメント	結果
	----+-----1-----+
options locale=English_UnitedStates;	-1234356.78
x=input('-1,234,356.78', nlnumi32.2);	

ステートメント

結果

put x=;

関連項目

出力形式:

- “NLNUMw.d 出力形式” (482 ページ)
- “NLNUMiw.d 出力形式” (484 ページ)
- “NLNUMw.d 入力形式” (892 ページ)

NLPCTw.d 入力形式

現地表現の百分率のデータを現在の SAS ロケールに適した数値として読み取ります。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTw.d

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

10^{*d*} で除算するかどうかを指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

詳細

NLPCTw.d 入力形式では、指定されたロケールで百分率データを現地表現で読み込み、そのデータを数値に変換します。値を 100 で除算し、桁区切り文字、小数点区切り文字、空白、パーセント記号、閉じかっこを入力データから削除します。

比較

NLPCTw.d 入力形式 では、NLPCTIw.d 入力形式とは逆の処理を行います。NLPCTw.d 入力形式 は、PERCENTw.d 入力形式と似ていますが、NLPCTw.d 入力形式は ロケール固有である点が違います。

数値データをアポストロフィで囲むと、NLPCT はエラーメッセージを返します。アポストロフィはデータが文字であることを指定します。次の例では、数値 1 がアポストロフィで囲まれているため、エラーメッセージが生成されます。

```
data;
x=input("'1'",NLPCT32.);
put x=;
run;
```

例

次の例では、入力値として-12,345.67%を使用します。

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates;	-123.4567
x=input('-12,345.67%',nlpct32.2);	-123.4567
y=input('(12,345.67%)',percent32.2);	
put x=;	
put y=;	

関連項目

出力形式:

- “NLPCTw.d 出力形式” (486 ページ)

- “NLPCTIw.d 出力形式” (488 ページ)
- “NLPCTIw.d 入力形式” (897 ページ)

NLPCTIw.d 入力形式

国際表現の百分率のデータを、現在の SAS ロケールに適した数値として読み取ります。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLPCTIw.d

構文の説明

w

入力幅を指定します。

デフォルト 6

範囲 1-32

d

10^d で除算するかどうかを指定します。データに小数点区切り文字が含まれる場合、*d* 値は無視されます。

デフォルト 0

範囲 0-31

詳細

NLPCTIw.d 入力形式では、指定されたロケールで、百分率データを国際表現で読み込み、そのデータを数値に変換します。値を 100 で除算し、桁区切り文字、小数点区切り文字、空白、パーセント記号、閉じかっこを入力データから削除します。

比較

NLPCTIw.d 入力形式 では、NLPCTw.d 入力形式とは逆の処理を行います。

例

次の例では、入力値として-12,345.67%を使用します。

ステートメント	結果
	----+----1----+
options LOCALE=English_UnitedStates;	-123.4567
x=input('-12,345.67%',nlpct32.2);	-123.4567
y=input('(12,345.67%)',percent32.2);	
put x=;	
put y=;	

関連項目

出力形式:

- [“NLPCTw.d 出力形式” \(486 ページ\)](#)
- [“NLPCTIw.d 出力形式” \(488 ページ\)](#)

入力形式:

- [“NLPCTw.d 入力形式” \(895 ページ\)](#)

NLSTRMON 入力形式

月名を、現在の SAS ロケールに適した数値として読み込みます。

カテゴリ: CAS
数値

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLSTRMON

詳細

NLSTRMON 入力形式は、指定されたロケールの月名を読み取り、数値に変換します。これらの例では English_United States ロケールを使用します。

- 第 1 月(January) = 1
- 第 2 月(February) = 2
- 第 3 月(March) = 3
- 第 4 月(April) = 4
- 第 5 月(May) = 5
- 第 6 月(June) = 6
- 第 7 月(July) = 7
- 第 8 月(August) = 8
- 第 9 月(September) = 9
- 第 10 月(October) = 10
- 第 11 月(November) = 11
- 第 12 月(December) = 12

例

この例では French_France (fr_fr)ロケールを使用します。

ステートメント	結果
options locale = fr_fr;	x=1
data test;	x=janvier
x = input('janvier',nlstrmon.);	x=5
put x = ;	x=mai
put x= nlstrmon.;	
x = input('mai',nlstrmon.);	
put x=;	
put x= nlstrmon.;	
run;	
option locale =en_us;	x=1;
data test;	x=2;
x = input("JANUARY",nlstrmon.);	x=3;
put x = ;	x=4;
x = input("FEB",nlstrmon.);	
put x = ;	
x = input("march",nlstrmon.);	

ステートメント	結果
<pre>put x =; x = input("apr",nlstrmon.); put x =; run;</pre>	

NLTIMAPw. 入力形式

時間値を、a.m.および p.m.表示を使用し現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。

カテゴリ: CAS
日付と時間

注: 2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMAPw.

構文の説明

w
入力幅を指定します。

デフォルト 10

範囲 4-200

例

次の例では、入力値として 04:24:43 p.m.を使用します。

ステートメント	結果
	----+----1----+
<pre>options locale=English_UnitedStates; y=input('04:24:43 PM',nltimap11.);</pre>	16:24:43

ステートメント	結果
put y time.;	
options locale=German_Germany;	16:24:00
y=input('16.24 Uhr',nltimap11.);	
put y time.;	

関連項目

出力形式:

- [“NLTIMAPw. 出力形式” \(498 ページ\)](#)

NLTIMew. 入力形式

時間値を、現在の SAS ロケールに適したローカルの SAS 時間値として読み込みます。

カテゴリ:	CAS
	日付と時間
別名:	NLTIMAP
注:	2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

NLTIMew.

構文の説明

<i>w</i>	入力幅を指定します。
デフォルト	20
範囲	10-200

例

次の例では、入力値として 16:24:43 を使用します。

ステートメント	結果
	----+----1----+
options locale=English_UnitedStates; y=input('16:24:43',nltime.); put y time.;	16:24:43
options locale=German_Germany; y=input('16.24 Uhr',nltime.); put y time.;	16:24:00

関連項目

出力形式:

- [“NLTIMEw. 出力形式” \(499 ページ\)](#)

\$REVERJw. 入力形式

文字データを右から左の順に読み込み、空白を保持します。

カテゴリ: CAS
文字

構文

\$REVERJ*w*.

構文の説明

w

入力幅を指定します。

デフォルト *w* が指定されていない場合は 1

範囲 1-32767

比較

\$REVERJw.入力形式は\$REVERSw.入力形式とほぼ同じですが、\$REVERSw.入力形式では先頭の空白がすべて削除され、結果は左詰めになります。

例

次の例では、入力値として ABCD を使用します。

```
input @1 name $reverj7.;
```

値	結果
	----+----1
ABCD	###DCBA
ABCD	DCBA### 1

1 #文字は空白を表します。

関連項目

入力形式:

- ["\\$REVERSw. 入力形式" \(903 ページ\)](#)

\$REVERSw. 入力形式

文字データを右から左の順に左詰めで読み込みます。

カテゴリ: CAS
文字

構文

\$REVERSw.

構文の説明

w
入力幅を指定します。

デフォルト `w` が指定されていない場合は 1
範囲 1-32767

比較

`$REVERSw`.入力形式は`$REVERJw`.入力形式とほぼ同じですが、`$REVERJw`.入力形式では先頭と末尾の空白がすべて保持されます。

例

次の例では、入力値として ABCD を使用します。

`input @1 name $revers7.;`

値	結果
	----+----1
ABCD	DCBA###
ABCD	DCBA### 1

1 #文字は空白を表します。

関連項目

入力形式:

■ [“\\$REVERJw. 入力形式” \(902 ページ\)](#)

\$UCS2Bw. 入力形式

ビッグエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2Bw.

構文の説明

w

入力幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

比較

\$UCS2Bw.入力形式では、\$UCS2BEw.入力形式とは逆の処理を行います。同一の動作環境内のデータを処理する場合は、\$UCS2Xw.入力形式を使用します。異なる動作環境からのデータを処理する場合は、\$UCS2Bw.入力形式や\$UCS2Lw.入力形式を使用します。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
x=input('5927'x,\$ucs2b.); put x=\$hex4.;	x=91e5

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- “\$UCS2Lw. 入力形式” (907 ページ)
- “\$UCS2Xw. 入力形式” (910 ページ)
- “\$UTF8Xw. 入力形式” (927 ページ)

\$UCS2BEw. 入力形式

現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列をビッグエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2BEw.

構文の説明

w

入力幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 1-32767

比較

\$UCS2BEw.入力形式では、\$UCS2Bw.入力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	-----1-----+

ステートメント	結果
ucs2str=input('大', \$ucs2be2.); put ucs2str=\$hex4.;	ucs2str=2020

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2BEw. 出力形式" \(507 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)

\$UCS2Lw. 入力形式

リトルエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2Lw.

構文の説明

w

入力幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

比較

\$UCS2Lw.入力形式では、\$UCS2LEw.入力形式とは逆の処理を行います。同一の動作環境内のデータを処理する場合は、\$UCS2Xw.入力形式を使用します。異なる動作環境からのデータを処理する場合は、\$UCS2Bw.入力形式や\$UCS2Lw.入力形式を使用します。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
x=input('2759'x,\$ucs2l.); put x=\$hex4.;	x=91e5

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS2LEw. 入力形式

現在の SAS セッションのエンコーディングにある文字列を読み込み、その文字列をリトルエンディアン、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2LEw.

構文の説明

w

入力幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

比較

\$UCS2LEw.入力形式では、\$UCS2Lw.入力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
ucs2str=input ('犬', \$ ucs2le2.); put ucs2str=\$hex4;	ucs2str=2759

関連項目

出力形式:

- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2LEw. 出力形式" \(510 ページ\)](#)

入力形式:

- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)

\$UCS2Xw. 入力形式

16 ビット、UCS2 形式の UNICODE(ユニコード)文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2Xw.

構文の説明

w

出力フィールドの幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

比較

\$UCS2Xw.入力形式では、\$UCS2XEw.入力形式とは逆の処理を行います。同一の動作環境内のデータを処理する場合は、\$UCS2Xw.入力形式を使用します。異なる動作環境からのデータを処理する場合は、\$UCS2Bw.入力形式や\$UCS2Lw.入力形式を使用します。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。この例では、リトルエンディアン形式を使用します。

ステートメント	結果
	----+----1----+
x=input('5927'x,\$ucs2x.); put x=\$hex4.;	x=91e5

関連項目

出力形式:

- “\$UCS2Bw. 出力形式” (506 ページ)
- “\$UCS2Lw. 出力形式” (509 ページ)
- “\$UCS2Xw. 出力形式” (512 ページ)
- “\$UTF8Xw. 出力形式” (532 ページ)

入力形式:

- “\$UCS2Bw. 入力形式” (904 ページ)
- “\$UCS2Lw. 入力形式” (907 ページ)
- “\$UTF8Xw. 入力形式” (927 ページ)

\$UCS2XEw. 入力形式

現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列を、16 ビット、UCS2 形式の UNICODE(ユニコード)に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS2XEw.

構文の説明

w

入力幅を指定します。16 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 2-32767

比較

\$UCS2XEw.入力形式では、\$UCS2Xw.入力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
ucs2str=input('大', \$ ucs2xe2.); put ucs2str=\$hex6;	ucs2str=5927

関連項目

出力形式:

- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UCS2xEw. 出力形式" \(513 ページ\)](#)

入力形式:

- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)

\$UCS4Bw. 入力形式

ビッグエンディアン、32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS4Bw.

構文の説明

w

入力幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

比較

同一の動作環境内のデータを処理する場合は、\$UCS4Xw.入力形式を使用します。異なる動作環境からのデータを処理する場合は、\$UCS4Bw.入力形式や\$UCS4Lw.入力形式を使用します。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
ucs4str=input(' 大', \$ucs4be8.);	ucs4str=00000020000005927

関連項目

出力形式:

- ["\\$UCS4Bw. 出力形式" \(515 ページ\)](#)

入力形式:

- ["\\$UCS4Lw. 入力形式" \(913 ページ\)](#)
- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)

\$UCS4Lw. 入力形式

リトルエンディアン、32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS4Lw.

構文の説明

w

入力幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

比較

同一の動作環境内のデータを処理する場合は、\$UCS4Xw.入力形式を使用します。異なる動作環境からのデータを処理する場合は、\$UCS4Bw.入力形式や\$UCS4Lw.入力形式を使用します。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+----2----+----3----+
z=put('com',\$UCS4L16.); put z \$hex32.;	2E000000630000006F0000006D000000

関連項目

出力形式:

- ["\\$UCS4Lw. 出力形式" \(518 ページ\)](#)

入力形式:

- ["\\$UCS4Bw. 入力形式" \(912 ページ\)](#)
- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)

\$UCS4Xw. 入力形式

32 ビット、UCS4 形式の UNICODE(ユニコード)文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS4Xw.

構文の説明

w

入力幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

比較

\$UCS4Xw.入力形式では、\$UCS4XEw.入力形式とは逆の処理を行います。\$UCS4Xw.入力形式は、同一の動作環境内のデータを処理する場合に使用します。\$UCS4Bw.入力形式や\$UCS4Lw.入力形式は、異なる動作環境からのデータを処理する場合に使用します。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。この例では、リトルエンディアン形式を使用します。

ステートメント	結果
	----+----1----+
ucs4=put('91e5'x,\$ucs4x.);	ucs4=27590000
sjis=input(ucs4,\$ucs4x.);	sjis=91E52020
put ucs4=\$hex8. sjis=\$hex8.;	

ステートメント	結果
run;	

関連項目

出力形式:

- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS4Xw. 出力形式" \(521 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UTF8Xw. 入力形式" \(927 ページ\)](#)

\$UCS4XEw. 入力形式

現在の SAS セッションのエンコーディングの文字列を読み込み、その文字列を、32 ビット、UCS4 形式の UNICODE(ユニコード)に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UCS4XEw.

構文の説明

w

入力幅を指定します。32 ビットサイズの UNICODE 文字が入る十分な幅を指定します。

デフォルト 8

範囲 4-32767

比較

\$UCS4XEw.入力形式では、\$UCS4Xw.入力形式とは逆の処理を行います。

例

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	-----1-----+
ucs4str=input('犬', \$ucs4xe2.); put ucs4str=\$hex8;	ucs4str=00005927

関連項目

出力形式:

- ["\\$UCS4Xw. 出力形式" \(521 ページ\)](#)
- ["\\$UCS4XEw. 出力形式" \(523 ページ\)](#)

入力形式:

- ["\\$UCS4Xw. 入力形式" \(915 ページ\)](#)

\$UESCw. 入力形式

UESC 表記でエンコードされた文字列を読み取り、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UESCw.

構文の説明

w

出力フィールドの幅を指定します。

デフォルト 8

範囲 1-32767

詳細

0 から 9、a から z、A から Z など、すべての動作環境で使用可能でない文字は、UESC 表記で表す必要があります。`$UESCw`入力形式はネストできます。

比較

`$UESCw`入力形式では、`$UESCEw`入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
<pre>x=input('¥u5927', \$uesc10.); y=input('¥uu5927', \$uesc10.); z=input('¥uuu5927', \$uesc10.); put x; put y; put z;</pre>	<pre>大 ¥u5927 ¥uu5927</pre>

関連項目

出力形式:

- ["\\$UESCw. 出力形式" \(524 ページ\)](#)
- ["\\$UESCEw. 出力形式" \(526 ページ\)](#)

入力形式:

\$UESCEw. 入力形式

現在の SAS セッションのエンコーディングを使用する文字列を読み取り、その文字列を UESC 表記に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UESCEw.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1-32767

詳細

\$UESCEw.入力形式はネストできます。

比較

\$UESCEw.入力形式では、\$UESCw.入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+

ステートメント	結果
x=input('大', \$uesc10.);	¥u5927
y=input('¥u5927', \$uesc10.);	¥uu5927
z=input('¥uu5927', \$uesc10.);	¥uuu5927
put x y z;	

関連項目

出力形式:

- ["\\$UESCw. 出力形式" \(524 ページ\)](#)
- ["\\$UESCEw. 出力形式" \(526 ページ\)](#)

入力形式:

- ["\\$UESCw. 入力形式" \(917 ページ\)](#)

\$UNCRw. 入力形式

NCR 文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UNCRw.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1–32767

詳細

入力文字列に含まれるのは、文字と NCR のみにする必要があります。各国語の文字は、NCR で表記する必要があります。

比較

\$UNCRw.入力形式では、\$UNCREw.入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
x=input ('大', \$uncr10.);	大
y=input('abc', \$uncr10.);	abc
put X;	
put Y;	

関連項目

出力形式:

- ["\\$UNCRw. 出力形式" \(527 ページ\)](#)
- ["\\$UNCREw. 出力形式" \(528 ページ\)](#)

入力形式:

- ["\\$UNCREw. 入力形式" \(921 ページ\)](#)

\$UNCREw. 入力形式

現在の SAS セッションのエンコーディングを使用する文字列を読み込み、その文字列を NCR に変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UNCRE w .

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1-32767

詳細

出力文字列は、プレーンテキストと NCR に変換されます。各国語の文字は、NCR に変換されます。

比較

\$UNCRE w .入力形式では、\$UNCR w .入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
x=input(' ㄨ abc', \$uncre12.); put x;	大abc

関連項目

出力形式:

- “\$UNCRw. 出力形式” (527 ページ)
- “\$UNCREw. 出力形式” (528 ページ)

入力形式:

- “\$UNCRw. 入力形式” (920 ページ)

\$UPARENw. 入力形式

UPAREN 表記でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UPAREN_w.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1-32767

詳細

SAS セッションのエンコーディングに対応する UNICODE 表記がない場合、その表記は現在の SAS セッションのエンコーディングのまま残ります。

比較

\$UPARENw.入力形式では、\$UPARENw.入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
<pre>v=input('<u0061>',\$uparen10.); w=input('<u0062>',\$uparen10.); x=input('<u0063>',\$uparen10.); y=input('<u0033>',\$uparen10.); z=input('<u5927>',\$uparen10.); put v; put w; put x; put y; put z;</pre>	<pre>a b c 3 大</pre>

関連項目

出力形式:

- ["\\$UPARENw. 出力形式" \(530 ページ\)](#)
- ["\\$UPARENw. 出力形式" \(531 ページ\)](#)

入力形式:

- ["\\$UPARENw. 入力形式" \(924 ページ\)](#)
- ["\\$UPARENpw. 入力形式" \(926 ページ\)](#)

\$UPARENw. 入力形式

現在の SAS セッションのエンコーディングを使用する文字列を読み込み、その文字列を UPAREN 表記に変換します。

- カテゴリ:
- 文字
- 制限事項:
- この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UPARENw.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1-32767

比較

\$UPARENEw.入力形式では、\$UPARENw.入力形式とは逆の処理を行います。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
v=input('a',\$uparen10.);	<u0061>
w=input('b',\$uparene10.);	<u0062>
x=input('c',\$uparene10.);	<u0063>
y=input('3',\$uparene10.);	<u0033>
z=input(' 太',\$uparen10.);	<u5927>
put v;	
put w;	
put x;	
put y;	
put z;	

関連項目

出力形式:

- ["\\$UPARENw. 出力形式" \(530 ページ\)](#)
- ["\\$UPARENEw. 出力形式" \(531 ページ\)](#)

入力形式:

- ["\\$UPARENw. 入力形式" \(923 ページ\)](#)
- ["\\$UPARENpw. 入力形式" \(926 ページ\)](#)

\$UPARENpw. 入力形式

UPAREN 表記でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換しますが、各国語文字は UPAREN 表記のエンコーディングのまま残ります。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UPARENpw.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1–32767

詳細

UPAREN 表記に各国語文字が含まれ、その値が UNICODE の 0x00ff より大きい場合、その表記は UPAREN 表記のまま残ります。

例

これらの例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
<pre>v=input('<u0061>',\$uparen10.); w=input('<u0062>',\$uparenp10.); x=input('<u0063>',\$uparenp10.); y=input('<u0033>',\$uparenp10.); z=input('<u5927>',\$uparepn10.); put v; put w; put x;</pre>	<pre>a b c 3 <u5927></pre>

ステートメント

結果

```
put y;
put z;
```

関連項目

出力形式:

- ["\\$UPARENw. 出力形式" \(530 ページ\)](#)
- ["\\$UPARENEw. 出力形式" \(531 ページ\)](#)

入力形式:

- ["\\$UPARENw. 入力形式" \(923 ページ\)](#)
- ["\\$UPARENEw. 入力形式" \(924 ページ\)](#)

\$UTF8Xw. 入力形式

UTF-8 でエンコードされた文字列を読み込み、その文字列を現在の SAS セッションのエンコーディングに変換します。

カテゴリ: 文字

制限事項: この入力形式は、CAS で実行される DATA ステップではサポートされていません。

構文

\$UTF8Xw.

構文の説明

w

入力幅を指定します。

デフォルト 8

範囲 1–32767

比較

この例では、Linux 動作環境でサポートされる日本語(Shift_JIS)エンコーディングを使用します。

ステートメント	結果
	----+----1----+
<pre>x=input(' e5a4a7' x, \$utf8x3.); put x;</pre>	大

関連項目

出力形式:

- ["\\$UCS2Bw. 出力形式" \(506 ページ\)](#)
- ["\\$UCS2Lw. 出力形式" \(509 ページ\)](#)
- ["\\$UCS2Xw. 出力形式" \(512 ページ\)](#)
- ["\\$UTF8Xw. 出力形式" \(532 ページ\)](#)

入力形式:

- ["\\$UCS2Bw. 入力形式" \(904 ページ\)](#)
- ["\\$UCS2Lw. 入力形式" \(907 ページ\)](#)
- ["\\$UCS2Xw. 入力形式" \(910 ページ\)](#)

\$VSLOGw. 入力形式

視覚的順序で並ぶ文字列を読み込み、その文字列を左から右の論理的順序に変換します。

カテゴリ: BIDI テキストの操作
CAS

構文

\$VSLOGw.

構文の説明

w
入力幅を指定します。

デフォルト 200

範囲 1-32000

比較

\$VSLOGw.入力形式では、\$VSLOGRw.入力形式とは逆の処理を行います。

例

次の例では、ヘブライ語の入力値"תִּירוּם הַמָּטָו flight"を使用します。

ステートメント	結果
	----+----1----+
x=input (' תִּירוּם הַמָּטָו ', \$vslog12.); put x;	תִּירוּם הַמָּטָו flight

次の例では、“computer”の“ت اذ” “アラビア語の入力値を使用します。

ステートメント	結果
	----+----1----+
x=input (' ت اذ computer', \$vslog12.); put x;	ت اذ コンピューター

関連項目

- 出力形式:**
- “\$VSLOGRw. 出力形式” (536 ページ)
 - “\$VSLOGw. 出力形式” (535 ページ)

入力形式:

■ “\$VSLOGRw. 入力形式” (930 ページ)

\$VSLOGRw. 入力形式

視覚的順序で並ぶ文字列を読み込み、その文字列を右から左の論理的順序に変換します。

カテゴリ: BIDI テキストの操作
CAS

構文

\$VSLOGRw.

構文の説明

w
入力幅を指定します。

デフォルト 200

範囲 1-32000

比較

\$VSLOGRw.入力形式では、\$VSLOGw.入力形式とは逆の処理を行います。

例

次の例では、“flight”פֿלייט のヘブライ語の入力値を使用します。

ステートメント	結果
	----+----1----+
x=input (' פֿלייט ',\$vslogr12.); put x;	flight פֿלייט

次の例では、“computer”מחשב “アラビア語の入力値を使用します。

ステートメント	結果
	-----1-----+
x=input (' حاسوب computer' computer', \$vslogr12.); put x;	حاسوب computer

関連項目

出力形式:

- ["\\$VSLOGw. 出力形式" \(535 ページ\)](#)
- ["\\$VSLOGRw. 出力形式" \(536 ページ\)](#)

入力形式:

- ["\\$VSLOGw. 入力形式" \(928 ページ\)](#)

YENw.d 入力形式

埋め込まれた円記号、カンマおよび小数点を削除します。

カテゴリ:	CAS 数値
制限事項:	YEN 入力形式は UTF8 エンコーディングをサポートしていません。 NLMNLJPY 出力形式 も参照してください。
注:	2025.12 以降、この入力形式は、DS2 言語で実行する場合に BIGINT データ型をサポートします。BIGINTPROCESSING=YES を設定すると、大きい整数値(BIGINT として保存)が丸めや切り捨てを行うことなく 19 桁までの完全な精度で表示されるようになります。15 桁または 16 桁を超える大きな値の場合は、BIGINTPROCESSING=YES をお勧めします。

構文

YENw.d

構文の説明

w

入力幅を指定します。

デフォルト 1

範囲 1-32

d

値を除算する 10 のべき乗を指定します。

要件 *d* は 0 または 2 でなければなりません。

ヒント *d* が 2 の場合、YEN*w.d* は小数点と小数点以下 2 桁を読み込みます。*d* が 0 の場合、YEN*w.d* は整数部分のみの値を読み込みます。

詳細

円記号文字のコードの 16 進表現は、EBCDIC のシステムでは 5B、ASCII のシステムでは 5C です。これらのコードが表す通貨記号は、他の国では異なる場合があります。

例

次の例では、入力値として円を使用します。

input value yen10.2;

値	結果
	----+----1----+
¥1254.71	1254.71

関連項目

出力形式:

■ [“YEN*w.d* 出力形式” \(544 ページ\)](#)

7 部

各国語サポート関連のマクロ関数

15 章

各国語サポート関連のマクロ関数のディクショナリ	935
-------------------------------	-----

15

各国語サポート関連のマクロ関数のディクショナリ

カテゴリ別のマクロ関数	935
ディクショナリ	936
%KCMPRES 関数	936
%KINDEX マクロ関数	937
%KLEFT マクロ関数	938
%KLENGTH マクロ関数	938
%KSCAN マクロ関数	939
%KSUBSTR マクロ関数	941
%KUPCASE マクロ関数	943
%QKCMPRES 関数	945
%QKLEFT マクロ関数	945
%QKSCAN マクロ関数	946
%QKSUBSTR マクロ関数	948
%QKUPCASE マクロ関数	950
%VALIDCHS マクロ関数	951

カテゴリ別のマクロ関数

次の表に、SAS 各国語サポートマクロ関数の概要を示します。詳細については、それぞれの各国語サポート関連のマクロ関数に関する情報を参照してください。

カテゴリ	言語要素	説明
DBCS	%KCMPRES 関数 (p. 936)	複数の空白を圧縮し、先頭と末尾の空白を削除します。
	%KINDEX マクロ関数 (p. 937)	文字列の先頭文字の位置を返します。

カテゴリ	言語要素	説明
	%KLEFT マクロ関数 (p. 938)	先頭の空白を削除し、引数を左詰めにします。
	%KLENGTH マクロ関数 (p. 938)	文字列の長さを返します。
	%KSCAN マクロ関数 (p. 939)	文字列内の単語を位置指定して検索します。
	%KSUBSTR マクロ関数 (p. 941)	文字列の部分文字列を生成します。
	%KUPCASE マクロ関数 (p. 943)	値を大文字に変換します。
	%QKCMPRES 関数 (p. 945)	複数の空白を圧縮し、先頭と末尾の空白を削除します。
	%QKLEFT マクロ関数 (p. 945)	先頭の空白を削除し、引数を左詰めにします。
	%QKSCAN マクロ関数 (p. 946)	文字列内の単語を位置指定して検索します。
	%QKSUBSTR マクロ関数 (p. 948)	文字列の部分文字列を生成します。
	%QKUPCASE マクロ関数 (p. 950)	値を大文字に変換します。

ディクショナリ

%KCMPRES 関数

複数の空白を圧縮し、先頭と末尾の空白を削除します。

カテゴリ: DBCS

種類: NLS マクロ関数

参照項目: ["%QKCMPRES 関数" \(945 ページ\)](#)

構文

%KCMPRES (*text* | *text expression*)

必須引数

text

テキストまたはテキスト式。

詳細

%KCMPRES マクロ関数および%QKCMPRES マクロ関数は、複数の空白を圧縮し、先頭と末尾の空白を削除します。%KCMPRES は、引数が引用符で囲まれている場合でも、引用符で囲まれていない結果を返します。%QKCMPRES は、引用符で囲まれた結果を返します。“[%QKCMPRES 関数](#)” (945 ページ)を参照してください。

```
& % ' " ( ) + - * / < > = ^ ~ ; , # blank AND OR NOT EQ NE LE LT GE GT IN
```

%KINDEX マクロ関数

文字列の先頭文字の位置を返します。

カテゴリ: DBCS

種類: NLS マクロ関数

構文

%KINDEX (*source*, *string*)

必須引数

source

文字列またはテキスト式を指定します。

string

文字列またはテキスト式を指定します。

詳細

%KINDEX 関数は、*source* を検索して、最初に出現する *string* の先頭文字の位置を返します。*string* が見つからない場合、関数は 0 を返します。

例: 文字の検索

次のステートメントは、文字列内に文字 V が最初に現れる位置を返します。

```
%let a=a very long value;  
%let b=%kindex(&a,v);  
%put V appears at position &b.;
```

ステートメントが実行されると、SAS ログに次の行が出力されます。

V appears at position 3.

%KLEFT マクロ関数

先頭の空白を削除し、引数を左詰めにします。

カテゴリ: DBCS
要件 MAUTOSOURCE システムオプション
参照項目: ["%QKLEFT マクロ関数"](#)

構文

%KLEFT (*text* | *text expression*)

詳細

%KLEFT マクロ関数は、先頭の空白を削除し、引数を左詰めにします。

%KLEFT は、引数が引用符で囲まれている場合でも、引用符で囲まれていない結果を返します。%QKLEFT は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します。

```
& % ' " ( ) + - * / < > = ~ ^ ~ ; , # blank  
AND OR NOT EQ NE LE LT GE GT IN
```

%KLENGTH マクロ関数

文字列の長さを返します。

カテゴリ: DBCS
種類: NLS マクロ関数

構文

%KLENGTH (*character string* | *text expression*)

詳細

引数が文字列の場合、%KLENGTH は文字列の長さを返します。引数がテキスト式の場合、%KLENGTH は変換された値の長さを返します。引数が NULL 値の場合、%KLENGTH は 0 を返します。

例: 文字列長を返す

次のステートメントは、文字列とテキスト式の長さを返します。

```
%let a=Happy;  
%let b=Birthday;  
%let c=René;  
%put The length of &a is %length(&a);  
%put The length of &b is %length(&b);  
%put The length of &c is %length(&c);  
%put The length of &a &b To &c is %length(&a &b to &c);
```

```
%put with LENGTH macro %length(&c);
```

ステートメントが実行されると、SAS ログに次の行が出力されます。

```
The length of Happy is 5.  
The length of Birthday is 8.  
The length of René is 4.  
The length of Happy Birthday To René is 22.
```

```
with LENGTH macro 5
```

%KSCAN マクロ関数

文字列内の単語を位置指定して検索します。

カテゴリ: DBCS

種類: NLS マクロ関数

参照項目: “%QKSCAN マクロ関数”

構文

%KSCAN (*argument*, *n*<,*delimiters* >)

必須引数

argument

文字列またはテキスト式を指定します。*argument* が後述する特殊文字または二ーモニック演算子を含む場合は、%QKSCAN を使用します。

n

この関数が返すワードの位置を表す整数、またはそのような整数を生成するテキスト式です。*n* が *argument* にある単語数より大きい場合、関数は NULL 文字列を返します。*n* が負の場合、%KSCAN は文字列の最後の単語から逆方向に検索して、単語を選択します。

delimiter

%QKSCAN が文字式内で単語区切り文字として使用する文字を指定します。

詳細

%KSCAN と %QKSCAN 関数は、*argument* を検索して、*n* 番目の単語を返します。単語とは、1 つ以上の区切り文字によって区切られた 1 つ以上の文字のことです。

%KSCAN では、引数がマクロ引用関数によりマスクされていても、特殊文字と二ーモニック演算子をマスクせずに値を返します。%QKSCAN は、次の特殊文字と二ーモニック演算子をマスクした値を返します。

```
& % ' ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

%KSCAN 関数では、次の条件をすべて満たす部分文字列を *word* (単語) とします。

- 左境界が、区切り文字または文字列の先頭である
- 右境界が、区切り文字または文字列の末尾である
- 区切り文字を含まない

単語は、その文字列の先頭または末尾に区切り文字がある場合、またはその文字列に 2 つ以上の連続した区切り文字が含まれる場合には長さが 0 となります。

%KSCAN 関数に指定する引数が 2 つのみの場合、デフォルトの区切り文字は、コンピューターの使用文字が ASCII であるか EBCDIC であるかによって異なります。

- コンピューターの使用文字が ASCII の場合、次の文字がデフォルトの区切り文字です。

```
blank ! $ % & ( ) * + , - . / ; < ^ | / ; < ^ |
```

^ 文字が含まれない ASCII 環境では、%KSCAN 関数は代わりに ~ 文字を使用します。

- コンピューターの使用文字が EBCDIC の場合、次の文字がデフォルトの区切り文字です。

```
blank ! $ % & ( ) * + , - . / ; < ^ | / ; < ~ | ¢ |
```

%KSCAN 関数では、文字引数に NULL を指定できます。NULL 引数は、長さがゼロの文字列として処理されます。数値引数には NULL を指定できません。

例: %KSCAN 関数と%QKSCAN 関数のアクションの比較

次の例に、%KSCAN と%QKSCAN の処理を示します。

```
%macro a;
  aaaaaa
%mend a;
%macro b;
  bbbbbb
%mend b;
%macro c;
  ccccc
%mend c;
%let x=%nrstr(%a*%b*%c);
%put X: &x;
%put The third word in X, with KSCAN: %kscan(&x,3,*);
%put The third word in X, with QKSCAN: %qkscan(&x,3,*);
```

%PUT ステートメントは、ログに次の行を出力します。

```
X: %a*%b*%c
The third word in X, with KSCAN: ccccc
The third word in X, with QKSCAN: %c
```

%KSUBSTR マクロ関数

文字列の部分文字列を生成します。

カテゴリ: DBCS

種類: NLS マクロ関数

構文

%KSUBSTR (*argument*, *position*<, *length*>)

必須引数

argument

文字列またはテキスト式を指定します。*argument* が後述する特殊文字またはニーモニック演算子を含む場合は、%QKSUBSTR を使用します。

position

整数または整数を生成する (テキスト、論理、算術) 式です。この整数は、部分文字列の先頭文字の位置を表します。*position* が文字列の文字数より大きい場合、%KSUBSTR と %QKSUBSTR は警告メッセージを発行して、NULL 値を返します。

オプション引数

length

部分文字列内の文字数を表すオプション整数、またはそのような整数を生成する式 (テキスト式、論理式、演算式) です。*length* が *argument* 内の *position* 以降の文字数より大きい場合、%KSUBSTR と %QKSUBSTR は警告メッセージを発行して、*position* から文字列の末尾までの文字で構成される部分文字列を返します。デフォルトで、%KSUBSTR と %QKSUBSTR は *position* から文字列の末尾までの文字で構成される文字列を作成します。

詳細

%KSUBSTR と %QKSUBSTR 関数は、*argument* 内の *position* 位置から *length* の文字数分の文字で構成される部分文字列を生成します。

%KSUBSTR は、特殊文字とニーモニック演算子をマスクせずに値を返します。

%QKSUBSTR は、次の特殊文字とニーモニック演算子をマスクします。

& % ' " () + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN

例

例 1: ファイル参照名を 8 文字に制限する例

マクロ MAKEFREF は %KSUBSTR を使用して、ユーザーが割り当てたパラメーターが 8 文字より長い場合、先頭の 8 文字をファイル参照名に割り当てます。

```
%macro makefref(fileref,file);
  %if %klength(&fileref) gt 8 %then
    %let fileref = %ksubstr(&fileref,1,8);
  filename &fileref "&file";
%mend makefref;
%makefref(humanresource,/dept/humanresource/report96)
```

SAS は、次のステートメントを読み込みます。

```
FILENAME HUMANRES "/dept/humanresource/report96";
```

例 2: セグメントに長いマクロ変数値を保存する

マクロ SEPMSG は、マクロ変数 MSG の値を 40 文字単位に分割して、単位ごとに異なる変数に保存します。

```
%macro sepmsg(msg);
  %let i=1;
  %let start=1;
```

```

%if %length(&msg)>40 %then
%do;
%do %until(%klength(&&msg&i)<40);
%let msg&i=%qksubstr(&msg,&start,40);
%put Message &i is: &&msg&i;
%let i=%eval(&i+1);
%let start=%eval(&start+40);
%let msg&i=%qksubstr(&msg,&start);
%end;
%put Message &i is: &&msg&i;
%end;
%else %put No subdivision was needed.;
%mend sepmsg;
%sepmsg(%nrstr(A character operand was found in the %EVAL function
or %IF condition where a numeric operand is required. A character
operand was found in the %EVAL function or %IF condition where a
numeric operand is required.));

```

このプログラムが実行されると、次の行が SAS ログに出力されます。

```

Message 1 is: A character operand was found in the %EV
Message 2 is: AL function or %IF condition where a nu
Message 3 is: meric operand is required. A character
Message 4 is: operand was found in the %EVAL function
Message 5 is: or %IF condition where a numeric operan
Message 6 is: d is required.

```

例 3: %KSUBSTR と %QKSUBSTR の処理の比較

%KSUBSTR は、C 言語で特殊文字とニーモニック演算子をマスクせずに結果を処理するため、名前が解決された結果を生成します。

```

%let a=one;
%let b=two;
%let c=%nrstr(&a &b);
%put C: &c;
%put With KSUBSTR: %ksubstr(&c,1,2);
%put With QKSUBSTR: %qksubstr(&c,1,2);

```

これらのステートメントが実行されると、次の行が SAS ログに出力されます。

```

C: &a &b
With KSUBSTR: one
With QKSUBSTR: &a

```

%KUPCASE マクロ関数

値を大文字に変換します。

カテゴリ: DBCS

種類: NLS マクロ関数

参照項目: ["%QKUPCASE マクロ関数"](#)

構文

%KUPCASE (*character string* | *text expression*)

詳細

%KUPCASE と %QKUPCASE 関数は、引数内の小文字を大文字に変換します。
%KUPCASE は、特殊文字とニーモニック演算子をマスクせずに値を返します。

引数が次の特殊文字またはニーモニック演算子を含む場合は、%QKUPCASE を使用します。
%QKUPCASE は、次の特殊文字とニーモニック演算子をマスクして値を返します。

```
& % ' ' ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

値を比較する場合、マクロ機能では比較前に小文字から大文字に自動的に変換しないため、%KUPCASE と %QKUPCASE が役に立ちます。

例

例 1: 比較する値を大文字に変換する

この例では、マクロ RUNREPT は、マクロ変数 MONTH に入力された値を文字列 DEC と比較します。大文字に変換された値が DEC の場合、PROC FSVIEW はデータセット REPORTS.ENDYEAR で実行されます。DEC ではない場合、PROC FSVIEW は、REPORTS データライブラリの月名のデータセットで実行されます。

```
%macro runrept(month);
  %if %kupcase(&month)=DEC %then
    %str(proc fsview data=reports.endyear; run;);
  %else %str(proc fsview data=reports.&month; run;);
%mend runrept;
```

このマクロを次のどの形式で呼び出した場合でも、同マクロ内の %IF 条件が true になります。

```
%runrept(DEC)
%runrept(Dec)
%runrept(dec)
```

例 2: %KUPCASE と %QKUPCASE の比較

次のステートメントは、%KUPCASE と %QKUPCASE により生成される結果を示します。

```
%let a=begin;
%let b=%nrstr(&a);
%put KUPCASE produces: %kupcase(&b);
%put QKUPCASE produces: %qkupcase(&b);
```

ステートメントが実行されると、SAS ログに次の行が出力されます。

KUPCASE produces: begin
QKUPCASE produces: &A

%QKCMPRES 関数

複数の空白を圧縮し、先頭と末尾の空白を削除します。

カテゴリ: DBCS
種類: NLS マクロ関数
参照項目: ["%KCMPRES 関数"](#)

構文

%QKCMPRES (*text* | *text expression*)

必須引数

text
テキストまたはテキスト式。

詳細

%KCMPRES マクロ関数および%QKCMPRES マクロ関数は、複数の空白を圧縮し、先頭と末尾の空白を削除します。%KCMPRES は、引数が引用符で囲まれている場合でも、引用符で囲まれていない結果を返します。%QKCMPRES は、引用符で囲まれた結果を返します。["%QKCMPRES 関数" \(945 ページ\)](#)を参照してください。

```
& % ' " ( ) + - * / < > = ^ ~ ; , # blank AND OR NOT EQ NE LE LT GE GT IN
```

%QKLEFT マクロ関数

先頭の空白を削除し、引数を左詰めにします。

カテゴリ: DBCS
要件 MAUTOSOURCE システムオプション
参照項目: ["%KLEFT マクロ関数"](#)

構文

%QKLEFT (*text* | *text expression*)

詳細

%QKLEFT マクロ関数は、先頭の空白を削除し、引数を左詰めにします。引数が次の特殊文字またはニーモニック演算子を含む場合は、QKLEFT を使用します。

```
& % ' " ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

比較

%KLEFT は、引数が引用符で囲まれている場合でも、引用符で囲まれていない結果を返します。%QKLEFT は、次の特殊文字とニーモニック演算子がマスクされた結果を生成します。そのためマクロプロセッサは、それらの結果を、マクロ言語の要素ではなくテキストとして解釈します。

%QKSCAN マクロ関数

文字列内の単語を位置指定して検索します。

カテゴリ: DBCS
種類: NLS マクロ関数
参照項目: ["%KSCAN マクロ関数"](#)

構文

%QKSCAN (*argument*, *n*<,delimiters >)

必須引数

argument

文字列またはテキスト式を指定します。*argument* が後述する特殊文字またはニーモニック演算子を含む場合は、%QKSCAN を使用します。

n

この関数が返すワードの位置を表す整数、またはそのような整数を生成するテキスト式です *n* が *argument* にある単語数より大きい場合、関数は NULL 文字列を返します。*n* が負の場合、%KSCAN は文字列の最後の単語から逆方向に検索して、単語を選択します。

delimiter

%QKSCAN が文字式内で単語区切り文字として使用する文字を指定します。

詳細

%KSCAN と %QKSCAN 関数は、*argument* を検索して、*n* 番目の単語を返します。単語とは、1 つ以上の区切り文字によって区切られた 1 つ以上の文字のことです。

%KSCAN では、引数がマクロ引用関数によりマスクされていても、特殊文字とニーモニック演算子をマスクせずに値を返します。%QKSCAN は、次の特殊文字とニーモニック演算子をマスクした値を返します。

```
& % ' " ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

%KSCAN 関数では、次の条件をすべて満たす部分文字列を *word* (単語) とします。

- 左境界が、区切り文字または文字列の先頭である
- 右境界が、区切り文字または文字列の末尾である
- 区切り文字を含まない

単語は、その文字列の先頭または末尾に区切り文字がある場合、またはその文字列に 2 つ以上の連続した区切り文字が含まれる場合には長さが 0 となります。

%KSCAN 関数に指定する引数が 2 つのみの場合、デフォルトの区切り文字は、コンピューターの使用文字が ASCII であるか EBCDIC であるかによって異なります。

- コンピューターの使用文字が ASCII の場合、次の文字がデフォルトの区切り文字です。

```
blank ! $ % & ( ) * + , - . / ; < ^ ' / ; < ^ |
```

^ 文字が含まれない ASCII 環境では、%KSCAN 関数は代わりに ~ 文字を使用します。

- お使いのコンピューターが EBCDIC 文字を使用している場合、デフォルトの区切り文字は次のようになります。

```
blank ! $ % & ( ) * + , - . / ; < ^ ' / ; < ~ | ¢ |
```

%KSCAN 関数では、文字引数に NULL を指定できます。NULL 引数は、長さがゼロの文字列として処理されます。数値引数には NULL を指定できません。

例: %KSCAN 関数と %QKSCAN 関数のアクションの比較

次の例に、%KSCAN と %QKSCAN の処理を示します。

```
%macro a;
  aaaaaa
%mend a;
%macro b;
  bbbbbb
```

```

%mend b;
%macro c;
    ccccc
%mend c;
%let x=%nrstr(%a*%b*%c);
%put X: &x;
%put The third word in X, with KSCAN: %kscan(&x,3,*);
%put The third word in X, with QKSCAN: %qkscan(&x,3,*);

%PUT ステートメントは、ログに次の行を出力します。

X: %a*%b*%c
The third word in X, with KSCAN: ccccc
The third word in X, with QKSCAN: %c

```

%QKSUBSTR マクロ関数

文字列の部分文字列を生成します。

カテゴリ: DBCS

種類: NLS マクロ関数

構文

%QKSUBSTR (*argument*, *position*<, *length*>)

必須引数

argument

文字列またはテキスト式を指定します。*argument* が後述する特殊文字または二一モニック演算子を含む場合は、%QKSUBSTR を使用します。

position

整数または整数を生成する (テキスト、論理、算術) 式です。この整数は、部分文字列の先頭文字の位置を表します。*position* が文字列の文字数より大きい場合、%KSUBSTR と %QKSUBSTR は警告メッセージを発行して、NULL 値を返します。

オプション引数

length

部分文字列内の文字数を表すオプション整数、またはそのような整数を生成する式 (テキスト式、論理式、演算式) です。*length* が *argument* 内の *position* 以降の文字数より大きい場合、%KSUBSTR と %QKSUBSTR は警告メッセージを発行して、*position* から文字列の末尾までの文字で構成される部分文字列を返します。デフォルトで、%KSUBSTR と %QKSUBSTR は *position* から文字列の末尾までの文字で構成される文字列を作成します。

詳細

%KSUBSTR と %QKSUBSTR 関数は、*argument* 内の *position* 位置から *length* の文字数分の文字で構成される部分文字列を生成します。

%KSUBSTR は、特殊文字とニーモニック演算子をマスクせずに値を返します。

%QKSUBSTR は、次の特殊文字とニーモニック演算子をマスクします。

```
& % ' " ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

例

例 1: ファイル参照名を 8 文字に制限する例

マクロ MAKEFREF は %KSUBSTR を使用して、ユーザーが割り当てたパラメーターが 8 文字より長い場合、先頭の 8 文字をファイル参照名に割り当てます。

```
%macro makefref(fileref,file);
  %if %klength(&fileref) gt 8 %then
    %let fileref = %ksubstr(&fileref,1,8);
  filename &fileref "&file";
%mend makefref;
%makefref(humanresource,/dept/humanresource/report96)
```

SAS は、次のステートメントを読み込みます。

```
FILENAME HUMANRES "/dept/humanresource/report96";
```

例 2: セグメントに長いマクロ変数値を保存する

マクロ SEPMSG は、マクロ変数 MSG の値を 40 文字単位に分割して、単位ごとに異なる変数に保存します。

```
%macro sepmsg(msg);
  %let i=1;
  %let start=1;
  %if %length(&msg)>40 %then
    %do;
      %do %until(%klength(&&msg&i)<40);
        %let msg&i=%qksubstr(&msg,&start,40);
        %put Message &i is: &&msg&i;
        %let i=%eval(&i+1);
        %let start=%eval(&start+40);
        %let msg&i=%qksubstr(&msg,&start);
      %end;
      %put Message &i is: &&msg&i;
    %end;
  %else %put No subdivision was needed.;
%mend sepmsg;
%sepmsg(%nrstr(A character operand was found in the %EVAL function
or %IF condition where a numeric operand is required. A character
operand was found in the %EVAL function or %IF condition where a
numeric operand is required.));
```

このプログラムが実行されると、次の行が SAS ログに出力されます。

```
Message 1 is: A character operand was found in the %EV
Message 2 is: AL function or %IF condition where a nu
Message 3 is: meric operand is required. A character
Message 4 is: operand was found in the %EVAL function
Message 5 is: or %IF condition where a numeric operan
Message 6 is: d is required.
```

例 3: %KSUBSTR と %QKSUBSTR の処理の比較

%KSUBSTR は、C 言語で特殊文字とニーモニック演算子をマスクせずに結果を処理するため、名前が解決された結果を生成します。

```
%let a=one;
%let b=two;
%let c=%nrstr(&a &b);
%put C: &c;
%put With KSUBSTR: %ksubstr(&c,1,2);
%put With QKSUBSTR: %qksubstr(&c,1,2);
```

これらのステートメントが実行されると、次の行が SAS ログに出力されます。

```
C: &a &b
With KSUBSTR: one
With QKSUBSTR: &a
```

%QKUPCASE マクロ関数

値を大文字に変換します。

カテゴリ: DBCS
種類: NLS マクロ関数
参照項目: ["%KUPCASE マクロ関数"](#)

構文

%QKUPCASE (*character string* | *text expression*)

詳細

%KUPCASE と %QKUPCASE 関数は、引数内の小文字を大文字に変換します。
%KUPCASE は、特殊文字とニーモニック演算子をマスクせずに値を返します。

引数が次の特殊文字またはニーモニック演算子を含む場合は、%QKUPCASE を使用します。
%QKUPCASE は、次の特殊文字とニーモニック演算子をマスクして値を返します。

```
& % ' " ( ) + - * / < > = ~ ^ ~ ; , # blank
AND OR NOT EQ NE LE LT GE GT IN
```

値を比較する場合、マクロ機能では比較前に小文字から大文字に自動的に変換しないため、%KUPCASE と %QKUPCASE が役に立ちます。

例

例 1: 比較する値を大文字に変換する

この例では、マクロ RUNREPT は、マクロ変数 MONTH に入力された値を文字列 DEC と比較します。大文字に変換された値が DEC の場合、PROC FSVIEW はデータセット REPORTS.ENDYEAR で実行されます。DEC ではない場合、PROC FSVIEW は、REPORTS データライブラリの月名のデータセットで実行されます。

```
%macro runrept(month);
  %if %kupcase(&month)=DEC %then
    %str(proc fsview data=reports.endyear; run;);
  %else %str(proc fsview data=reports.&month; run;);
%mend runrept;
```

このマクロを次のどの形式で呼び出した場合でも、同マクロ内の %IF 条件が true になります。

```
%runrept(DEC)
%runrept(Dec)
%runrept(dec)
```

例 2

これらのステートメントが実行されると、SAS ログに次の行が出力されます。

```
KUPCASE produces: BEGIN
QKUPCASE produces: &A
```

%VALIDCHS マクロ関数

データセット変数の文字エンコーディングの互換性を検証します。

制限事項: エンコーディング値が DEFAULT に等しい場合、このマクロは使用しないでください。

構文

```
%VALIDCHS(dsnm=dataset_name,
libnm=libref_name<ENCODING=encoding_name,>><COMPATIBLE=encoding_name>>
```

必須引数

`dsnm=dataset_name`

検証するデータセットの名前。

`libnm=libref_name`

検証する SAS ライブラリの名前。

オプション引数

`ENCODING=encoding_name`

指定したエンコーディングでデータセット内の文字が有効であることを検証するために%VALIDCHS マクロ関数を使用するオプションのエンコーディング名。ENCODING=オプションが指定されていない場合、データセットのエンコーディングが使用されます。

`COMPATIBLE=encoding_name`

指定されたエンコーディングとデータセット内のデータの互換性を検証するために%VALIDCHS マクロ関数を使用するオプションのエンコーディング。これはエンコーディングの異なるサーバーとクライアントの間でデータセットを正常に転送できるかどうか判断するのに役立ちます。互換性のない文字により、データ転送の問題が発生する場合があります。

詳細

基本

%VALIDCHS マクロ関数はデータセットをスキャンし、データセットの文字変数内の文字を検査して、指定されたエンコーディングで有効かどうか判定します。ENCODING=オプションが指定されている場合、%VALIDCHS マクロ関数は ENCODING=の値に基づいて検証を行います。ENCODING=オプションが指定されていない場合、%VALIDCHS はデータセットのエンコーディングを使用します。COMPATIBLE=オプションが指定されている場合、%VALIDCHS はデータセットのエンコーディング(または ENCODING=オプションのエンコーディング)の文字データの内容を検証します。その後、COMPATIBLE=オプションで指定されたエンコーディングに基づいて互換性を検証します。たとえば、euc-cn エンコーディング(漢字)のデータセットで、COMPATIBLE=wlatin1 が指定されているとします。%VALIDCHS は文字が euc-cn エンコーディングで有効であることを検証します。その後%VALIDCHS は文字の互換性を検証し、それらの文字が wlatin1 エンコーディングでサポートされているかどうか確認します。%VALIDCHS がデータセット内で無効な文字を検出した場合、メッセージが表示され、検証が続行されます。

自動マクロ変数

`_validchars_rc` は%VALIDCHS により生成される自動マクロ変数です。これはテストのステータスを示します。次に、例を示します。

```

2: No invalid characters are found but data will cause the truncation
   1: No invalid characters are found
   0: Invalid characters are found
  -1: Invalid characters are found and also data will cause the truncation
  -2: General error

```

例

例 1: データセット内の文字データの単純な検証

%VALIDCHS は入力データセットのエンコーディングを使用します。

```

%VALIDCHS(dsnm=sashelp.class);
%put &_validchars_rc;

```

データセットのエンコーディングは us-ascii です。%VALIDCHS マクロ関数はすべての文字データが us-ascii で有効であることを確認し、次のメッセージをログに返します。

```

NOTE: All characters in the dataset: sashelp.class (encoding:us-ascii) are valid.
0

```

例 2: 指定されたエンコーディングでデータセット内の文字データを検証

%VALIDCHS は指定されたエンコーディングを使用して次の文字を検証します。

```

%VALIDCHS(dsnm=sashelp.class, encoding=utf-8);
%put &_validchars_rc;

```

次のメッセージが SAS ログに書き込まれます。

```

NOTE: All characters in the dataset: sashelp.class (encoding:utf-8) are valid.
0

```

例 3: 文字データの検証と互換性のチェック

```

%VALIDCHS(dsnm=sashelp.class, encoding=euc-cn, compatible=wlatin1);
%put &_validchars_rc;

```

次のメッセージが SAS ログに書き込まれます。

```

NOTE: All characters in the dataset: sashelp.class (encoding:euc-cn) are
compatible to wlatin1 encoding.
0

```

例 4: 文字データの検証と互換性のチェック

次のデータセットには wlatin1 エンコーディングと各国文字が含まれます。

```

data mydata;

```

```
length ch $1.;  
ch = '€';  
run;  
  
%VALIDCHS(dsnm=mydata, encoding=wlatin1, compatible=utf-8);  
%put &_validchars_rc;
```

次のメッセージが SAS ログに書き込まれます。

```
WARNING: Found possible truncation in the variable: ch at observation number: 1  
ERROR: Detected the issue in the dataset: mydata (encoding:wlatin1) when  
validating the compatibility to ::utf-8 encoding.  
2
```


8 部

各国語サポート関連のシステムオプション

16 章

各国語サポート関連のシステムオプションのディクショナリ 957

各国語サポート関連のシステムオプションのディクショナリ

カテゴリ別のシステムオプションエントリ	958
ディクショナリ	959
BOMFILE システムオプション	959
DATESTYLE= システムオプション	960
DBCS システムオプション	962
DBCSLANG システムオプション	963
DBCSTYPE システムオプション	964
DFLANG= システムオプション	966
ENCODING システムオプション	968
FSDBTYPE システムオプション	970
FSIMM システムオプション	971
FSIMMOPT システムオプション	972
LOCALE システムオプション	973
LOCALEDATA システムオプション	974
LOGLANGCHG システムオプション	975
LOGLANGENG システムオプション	976
LSWLANG システムオプション	978
MAPEBCDIC2ASCII= システムオプション	979
NLDECSEPARATOR システムオプション	980
ODSLANGCHG システムオプション	982
RSASIOTRANSERROR システムオプション	984
SORTSEQ= システムオプション	984
TIMEZONE= システムオプション	986
TRANTAB= システムオプション	988
VALIDMEMNAME= システムオプション	990
VALIDVARNAME= システムオプション	992
URLENCODING= システムオプション	995

カテゴリ別のシステムオプションエントリ

SAS システムオプションの言語コントロールカテゴリは、各国語サポートの影響を受けます。次の表に、SAS システムオプションの概要を示します。詳細については、各 SAS システムオプションのディクショナリエントリを参照してください。

カテゴリ	言語要素	説明
CAS	DATESTYLE= システムオプション (p. 960)	ANYDTDTTE、ANYDTDTM または ANYDTTME 入力形式データがあいまいな場合の月、日、年の順序を指定します。
SORT: プロシジャオプション	SORTSEQ= システムオプション (p. 984)	SORT および SQL プロシジャが現在の SAS セッションで使用する、言語固有の照合順序を指定します。
環境コントロール: 言語コントロール	DATESTYLE= システムオプション (p. 960)	ANYDTDTTE、ANYDTDTM または ANYDTTME 入力形式データがあいまいな場合の月、日、年の順序を指定します。
	DBCS システムオプション (p. 962)	2 バイト文字セット(DBCS)を認識します。
	DBCSLANG システムオプション (p. 963)	2 バイト文字セット(DBCS)言語を指定します。
	DBCSTYPE システムオプション (p. 964)	2 バイト文字セット(DBCS)言語に使用するエンコーディング方法を指定します。
	DFLANG= システムオプション (p. 966)	国際化対応の日付の入力形式および出力形式の言語を指定します。
	ENCODING システムオプション (p. 968)	SAS セッションのデフォルト文字セットエンコーディングを指定します。
	FSDBTYPE システムオプション (p. 970)	フルスクリーンの 2 バイト文字セット(DBCS)エンコーディング方法を指定します。
	FSIMM システムオプション (p. 971)	フルスクリーンの 2 バイト文字セット(DBCS)の入力方法モジュール(IMM)を指定します。
	FSIMMOPT システムオプション (p. 972)	フルスクリーンの 2 バイト文字セット(DBCS)で使用する入力方法モジュール(IMM)のオプションを指定します。
	LOCALE システムオプション (p. 973)	地域の言語、地域の規則および文化を反映する、SAS セッションの属性一式を指定します。
	LOCALEDATA システムオプション (p. 974)	ロケール情報のソースデータベースを指定します。

カテゴリ	言語要素	説明
	LOGLANGENG システムオプション (p. 976)	起動後に LOCALE オプションを設定した場合に、SAS ログメッセージテキストに英語を使用するかを指定します。
	MAPEBCDIC2ASCII= システムオプション (p. 979)	EBCDIC から ASCII へのトランスコードと、ASCII から EBCDIC へのトランスコードで SAS ユーザーが使用する変換テーブルを指定します。
	ODSLANGCHG システムオプション (p. 982)	ODS 出力のテキスト言語を変更できるかを設定します。
	TIMEZONE= システムオプション (p. 986)	ユーザーローカルタイムゾーンを指定します。
	TRANTAB= システムオプション (p. 988)	SAS のさまざまな構成要素で使用される変換テーブルを指定します。
	URLENCODING= システムオプション (p. 995)	SAS セッションエンコーディングと UTF-8 エンコーディングのどちらを使用して URLENCODE 関数と URLDECODE 関数の引数が解釈されるのかを指定します。
言語コントロール	LSWLANG システムオプション (p. 978)	SAS 起動時に LOGLANGCHG オプションまたは ODSLANGCHG オプションを設定する際に、言語切り替え機能で使用する言語を指定します。
入力コントロール: データ処理	DATESTYLE= システムオプション (p. 960)	ANYDTDTE、ANYDTDTM または ANYDTTME 入力形式データがあいまいな場合の月、日、年の順序を指定します。
ファイル: SAS ファイル	RSASIoTERROR システムオプション (p. 984)	リモートアプリケーションから無効なデータが読み込まれたときにトランスコーディングエラーを表示します。
	VALIDMEMNAME= システムオプション (p. 990)	SAS データセット、SAS データビューおよびアイテムストアの命名規則を指定します。
	VALIDVARNAME= システムオプション (p. 992)	SAS セッション中に作成および処理可能な有効な SAS 変数名の規則を指定します。
ファイル: 外部ファイル	BOMFILE システムオプション (p. 959)	Unicode でエンコードされた外部ファイルにバイトオーダーマーク(BOM)接頭辞を書き出すかどうかを指定します。

ディクショナリ

BOMFILE システムオプション

Unicode でエンコードされた外部ファイルにバイトオーダーマーク(BOM)接頭辞を書き出すかどうかを指定します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント
 カテゴリ: ファイル: 外部ファイル
 PROC OPTIONS EXTFILES
 GROUP=

構文

BOMFILE

NOBOMFILE

構文の説明

BOMFILE

Unicode でエンコードされたファイルを外部ファイルに書き出すときに、バイトオーダーマーク(BOM)接頭辞を書き出すように指定します。

NOBOMFILE

Unicode でエンコードされたファイルを外部ファイルに書き出すときに、BOM 接頭辞を書き出さないように指定します。

詳細

BOMFILE システムオプションは、Unicode でエンコードされた外部ファイルを読み込むときには適用されません。

BOM は、Unicode データストリームの先頭の署名です。BOM のサイズはエンコーディングにより異なります。

DATESTYLE= システムオプション

ANYDTDTE、ANYDTDTM または ANYDTTME 入力形式データがあいまいな場合の月、日、年の順序を指定します。

該当要素: 自動実行ファイル、OPTIONS ステートメント
 カテゴリ: 環境コントロール: 言語コントロール
 入力コントロール: データ処理
 CAS
 PROC OPTIONS INPUTCONTROL
 GROUP= LANGUAGECONTROL

デフォルト: デフォルト値は、LOCALE=システムオプションの値によって決定されます。

操作: DATESTYLE=システムオプションの値は、新しい CAS セッションに使用されます。
 DATESTYLE=セッションオプションは、CAS セッション内の DATESTYLE=システム

オプションの値より優先されます。DATESTYLE=セッションオプションは、CAS ステートメントの SESSOPTS=オプションを使用するか、PROC CAS の sessionProp.setSessOpt アクションを使用して設定されます。

注: サイト管理者はこのオプションを制限できません。詳細については、“[制限されたオプション](#)” (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。

参照項目:

構文

DATESTYLE= [MDY](#) | [YMD](#) | [DMY](#) | [LOCALE](#)

構文の説明

MDY

SAS に設定される順序を月、日、年に指定します。

YMD

SAS に設定される順序を年、月、日に指定します。

DMY

SAS に設定される順序を日、月、年に指定します。

LOCALE

LOCALE=システムオプション値に対応する値に基づいて、SAS に設定する順序を MDY、YMD、DMY のいずれかに指定します。

詳細

システムオプション DATESTYLE=は、月、日、年の順序を識別します。デフォルト値は LOCALE です。デフォルトの LOCALE システムオプションの値は英語です。したがって、デフォルトの DATESTYLE の順序は MDY になります。

各ロケールオプション値のデフォルト設定については、[ロケール値](#)を参照してください。

関連項目

入力形式:

- [“ANYDTDTew. 入力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)
- [“ANYDTDTMw. 入力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)
- [“ANYDTTMEw. 入力形式” \(SAS 出力形式と入力形式: リファレンス\)](#)

システムオプション:

- [“LOCALE システムオプション” \(973 ページ\)](#)

セッションオプション:

- [“DATESTYLE= セッションオプション” \(SAS Cloud Analytic Services: ユーザーガイド\)](#)

DBCS システムオプション

2 バイト文字セット(DBCS)を認識します。

該当要素:	構成ファイル、SAS 起動時
カテゴリ:	環境コントロール: 言語コントロール
PROC OPTIONS GROUP=	LANGUAGECONTROL
デフォルト:	NODBCS

構文

-DBCS

-NODBCS

必須引数

DBCS

値のエンコーディングで 2 バイト文字セット(DBCS)を認識します。DBCS エンコーディングは、東アジア言語のサポートに使用されます。

NODBCS

値のエンコーディングで DBCS を認識しません。かわりに、1 バイト文字セット(SBCS)が値のエンコーディングに使用されます。1 バイトで文字セットの各文字が表記されます。

詳細

DBCS システムオプションは、中国語、日本語、韓国語、台湾語のような東アジア各国の言語をサポートするために使用されます。

関連項目

概念的な情報:

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)

- [“SAS セッションの DBCS 値” \(1115 ページ\)](#)
- [23 章, “SAS 言語要素中のエンコーディング値” \(1117 ページ\)](#)

システムオプション:

- [“DBCSLANG システムオプション” \(963 ページ\)](#)
- [“DBCSTYPE システムオプション” \(964 ページ\)](#)

DBCSLANG システムオプション

2 バイト文字セット(DBCS)言語を指定します。

該当要素: 構成ファイル、SAS 起動時
 カテゴリ: 環境コントロール: 言語コントロール
 PROC OPTIONS LANGUAGECONTROL
 GROUP=
 デフォルト: なし

構文

-DBCSLANG *language*

必須引数

language

動作環境に依存します。有効な言語値は、次の表を参照してください。

表 16.1 各動作環境でサポートされる DBCS 言語

言語	LINUX
CHINESE (簡体字中国語)	はい
JAPANESE	はい
KOREAN	はい
TAIWANESE (繁体字中国語)	はい
NONE	いいえ
UNKNOWN	いいえ

詳細

DBCSLANG システムオプションの適切な設定は、DBCSTYPE システムオプションに使用された設定に応じて異なります。すべての DBCSLANG 言語をサポートする DBCSTYPE 設定もありますが、日本語のみサポートする DBCSTYPE 設定もあります。

CHINESE では、簡体字中国語とも呼ばれる、中華人民共和国で使用する言語が指定されます。TAIWANESE では、繁体字中国語とも呼ばれる、台湾で使用する中国語が指定されます。

関連項目

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)
- [“SAS セッションの DBCS 値” \(1115 ページ\)](#)
- [23 章, “SAS 言語要素中のエンコーディング値” \(1117 ページ\)](#)

システムオプション:

- [“DBCS システムオプション” \(962 ページ\)](#)
- [“DBCSTYPE システムオプション” \(964 ページ\)](#)

DBCSTYPE システムオプション

2 バイト文字セット(DBCS)言語に使用するエンコーディング方法を指定します。

該当要素:	構成ファイル、SAS 起動時
カテゴリ:	環境コントロール: 言語コントロール
PROC OPTIONS GROUP=	LANGUAGECONTROL

構文

-DBCSTYPE *encoding-method*

必須引数

encoding-method

2 バイト文字セット(DBCS)言語のエンコードに使用する方法を指定します。
encoding-method の有効値は、コンピューターのハードウェアメーカーが動作環境に適用した標準に依存します。

詳細

DBCS エンコーディング方法は、コンピューターのハードウェアメーカーと標準組織によって異なります。

DBCSLANG=システムオプションは、エンコーディング方法が適用される言語を指定します。DBCSTYPE=は、DBCS および DBCSLANG=システムオプションも指定した場合にのみ指定する必要があります。

比較

表 16.2 LINUX の DBCS エンコーディング方法

DBCSTYPE=値	説明
DEC	DEC エンコーディング方法
EUC	拡張 LINUX コードエンコーディング方法
HP15	Hewlett Packard エンコーディング方法
PCIBM	IBM PC エンコーディング方法
PCMS	Microsoft PC エンコーディング方法
SJIS	日本語にのみ有効な Shift-JIS エンコーディング方法
NONE	DBCS 処理の無効化

関連項目

概念的な情報:

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)
- [“SAS セッションの DBCS 値” \(1115 ページ\)](#)
- [23 章, “SAS 言語要素中のエンコーディング値” \(1117 ページ\)](#)

システムオプション:

- [“DBCS システムオプション” \(962 ページ\)](#)
- [“DBCSLANG システムオプション” \(963 ページ\)](#)

DFLANG= システムオプション

国際化対応の日付の入力形式および出力形式の言語を指定します。

該当要素: SAS Environment Manager, 自動実行ファイル、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: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

デフォルト: English

構文

DFLANG=*'language'*,LOCALE

構文の説明

'language'

国際化対応の日付の入力形式および出力形式に使用する言語を指定します。

language の値として有効なのは、次の言語です。

- Afrikaans
- Catalan
- Croatian
- Czech
- Danish
- Dutch
- English
- Finnish
- French
- German
- Hungarian
- Italian
- Japanese
- Macedonian
- Norwegian
- Polish

- Portuguese
- Russian
- Slovenian
- Spanish
- Swedish
- Swiss_French
- Swiss_German

LOCALE

LOCALE システムオプションで指定されたロケールがアクティブなロケールになります。

詳細

DFLANG システムオプションの値は、SAS セッション中に変更できますが、一度に使用できる言語は 1 つのみです。*language* の値では、大文字と小文字は区別されません。

DFLANG=locale を指定すると、LOCALE ステートメントのシステムオプションに指定されているロケールが有効なロケールになります。locale または language は DFLANG システムオプションでサポートされる言語である必要があります。

次の例では、国際化対応の日付の入力形式および出力形式にはドイツ語(German)が使用されます。ドイツ語ロケールの POSIX 名は de_DE になります。ドイツ語ロケールは DFLANG システムオプションでサポートされています。

```
option locale=de_DE; /* German locale */
option DFLANG=locale;
```

次の例では、国際化対応の日付の入力形式および出力形式には英語(English)が使用されます。マルタ語(Maltese)は DFLANG でサポートされていないので、デフォルトロケールは English になります。

```
option locale=mt_MT; /* Maltese locale */
option DFLANG=locale;
```

DFLANG=locale を指定すると、日付形式の出力は、LOCALE システムオプションで指定されたロケールで表示されます。出力の日付形式をコントロールするために、DFLANG のロケールとして起動時に設定された LOCALE システムオプションに基づく値を使用します。DFLANG が有効な言語に設定されていない場合、出力の日付形式はデフォルトでは英語になります。次の例では、ロケールはフランス語に設定されています:

```
Sas.exe -locale French
Proc print data=sashelp.class; run;
```

```
mercredi 09 mars 2011 14 h 25
```

DFLANG=JAPANESE を設定した場合、DFLANG は DFLANG=LOCALE の場合と同じように動作します。

値 JAPANESE は、日本語版イメージを実行している顧客が、日本語向けに書式化された SAS ログや SAS リスティングで日付および時刻スタンプを確認できるようにするためにサポートされています。

- 起動時に DFLANG が設定されていない場合には、LOCALE オプションによってオプション値が設定されます。
- SAS ログの日付および時刻スタンプのデフォルトの表示形式は英語です。DFLANG として LOCALE または JAPANESE が設定されている場合、SAS ログ内の日付および時刻スタンプは現在の SAS LOCALE に従ってフォーマットされます。
- SAS リストの日付および時刻スタンプのデフォルト表示形式は英語です。DFLANG として LOCALE または JAPANESE が設定されている場合、SAS リストまたは ODS 宛先に送信される出力の各ページに含まれる日付および時刻スタンプは、DFLANG=LOCALE が設定された直後に有効になった SAS LOCALE に従ってフォーマットされます。
- LOCALE オプションに対する変更は、DTRESET オプションが有効な場合を除き、日付および時刻スタンプのフォーマット方法に影響しません。その場合、日付および時刻スタンプは現在の SAS LOCALE に従ってフォーマットされます。

値 JAPANESE を使用して日本語イメージを処理して、SAS ログの日時スタンプや日本語向けにフォーマットされたリストを表示できます。値 LOCALE もサポートされています。

関連項目

[DTRESET システムオプション](#)

ENCODING システムオプション

SAS セッションのデフォルト文字セットエンコーディングを指定します。

該当要素:	構成ファイル、SAS 起動時
カテゴリ:	環境コントロール: 言語コントロール
PROC OPTIONS GROUP=	LANGUAGECONTROL

構文

-ENCODING= *ASCIIANY encoding-value*

ENCODING=*encoding-value*

必須引数

ASCIANY

トランスコーディングは、通常は SAS によりセッションエンコーディングとデータセットエンコーディングが異なることが検出されたときに発生します。

ASCIANY では、データセットにアクセスする SAS セッションに ASCII の値をエンコードするセッションがある場合に、SAS でトランスコードされないデータセットを作成できます。

注: ANY は BINARY の指定と同等です。データはバイナリであるため、実際のエンコーディングには意味がありません。

encoding-value

すべての動作環境の有効な値については、[24 章, “SAS セッションのエンコーディング値” \(1131 ページ\)](#)を参照してください。

詳細

文字セットエンコーディングは、コードポイントと呼ばれる数値にマップされた文字のセットです。

SAS セッションのエンコーディングは、次のように ENCODING=、LOCALE=、DBCSTYPE=、DBCSLANG=システムオプションの値で決まります。

- ENCODING オプションが指定されていない場合、ENCODING 値は LOCALE の値と SAS を起動しているオペレーティングシステムから判別されます。また、LOCALE が設定されていない場合、デフォルト LOCALE は en_US になります。
- LOCALE=と ENCODING=の両方が指定された場合、セッションエンコーディングは ENCODING=オプションで指定された値です。
- LOCALE=が指定されていて ENCODING=が指定されていない場合、SAS は LOCALE=値から適切なエンコーディング値を決定します。
- DBCS オプションが設定された場合、DBCSLANG=と DBCSTYPE=システムオプションの値により ENCODING=と LOCALE=の値が決まります。

関連項目

概念的な情報:

- [“各国語サポート関連のロケール概念の概要”](#)
- [“概要: 各国語サポート関連のエンコーディング”](#)についての概念的な議論
- [“トランスコーディングの概要”](#)についての概念的な議論
- [表 21.1](#)
- [22 章, “DBCS データ処理に関連する SAS システムオプション”](#)
- [23 章, “SAS 言語要素中のエンコーディング値”](#)

FSDBTYPE システムオプション

フルスクリーンの 2 バイト文字セット(DBCS)エンコーディング方法を指定します。

該当要素: 構成ファイル、SAS 起動時、SASV9_OPTIONS 環境変数
カテゴリ: 環境コントロール: 言語コントロール
PROC OPTIONS LANGUAGECONTROL
GROUP=
デフォルト: DEFAULT

構文

-FSDBTYPE *encoding-method*

詳細

FSDBTYPE=システムオプションでは、フルスクリーン DBCS を可能にする方法に適切なエンコーディング方法を指定します。フルスクリーン DBCS エンコーディング方法は、コンピューターのハードウェアメーカーと標準組織により異なります。

表 16.3 フルスクリーン DBCS エンコーディング方法

FSDBTYPE=エンコーディング方法	説明
dec	Digital Equipment Corporation (DEC)エンコーディング方法
euc	拡張 LINUX エンコーディング方法
hp15	HP-UX エンコーディング方法
jis7	日本語の X Windows 環境でのみ使用される 7 ビット Shift-JIS エンコーディング方法
pcibm	IBM PC エンコーディング方法
SJIS	日本語にのみ有効な Shift-JIS エンコーディング方法

FSDBTYPE=エンコーディング方法	説明
デフォルト	特定のホストで使用するデフォルトの方法

関連項目

概念的な情報:

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)
- [“SAS セッションの DBCS 値” \(1115 ページ\)](#)
- [23 章, “SAS 言語要素中のエンコーディング値” \(1117 ページ\)](#)

FSIMM システムオプション

フルスクリーンの 2 バイト文字セット(DBCS)の入力方法モジュール(IMM)を指定します。

該当要素: 構成ファイル、SAS 起動時、SASV9_OPTIONS 環境変数
 カテゴリ: 環境コントロール: 言語コントロール
 PROC OPTIONS GROUP= LANGUAGECONTROL
 デフォルト: なし

構文

-FSIMM*fsdevice_name=IMM-name1*<,*fsdevice_name=IMM-name2*> ...

詳細

IMM-name に次の値を指定できます。

TTY | SASWUJT

/dev/tty のインターフェイスを提供します。この IMM では、DBCS 入力機能のある端末エミュレータ経由で DBCS 文字列を入力できます。

PIPE | SASWUJP

パイプインターフェイスを提供します。このインターフェイスでは、DBCS 入力サーバープロセスを分岐します。デフォルトのサーバー名は *saswujms* で、ベンダーで提供される MOTIF ツールキットを使用します。

たとえば、X11 ドライバーに PIPE 入力方法モジュールを使用するには、次のように指定します。

```
-FSIMM X11=PIPE
```

注: サーバーは FSIMMOPT オプションを使用して指定します。

関連項目

概念的な情報:

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)

システムオプション:

- [“FSIMMOPT システムオプション” \(972 ページ\)](#)

FSIMMOPT システムオプション

フルスクリーンの 2 バイト文字セット(DBCS)で使用される入力方法モジュール(IMM)のオプションを指定します。

該当要素:	構成ファイル、SAS 起動時、SASV9_OPTIONS 環境変数
カテゴリ:	環境コントロール: 言語コントロール
PROC OPTIONS GROUP=	LANGUAGECONTROL
デフォルト:	なし

構文

-FSIMMOPT *fullscreen-IMM:IMM-option*

詳細

FSIMMOPT システムオプションでは、各フルスクリーン IMM (入力方法モジュール)のオプションを指定します。各 IMM に 1 つの FSIMMOPT オプションのみ指定できます。同じ IMM に複数の FSIMMOPT オプションを指定した場合、最後の指定のみが使用されます。

各 IMM のオプション値については、SAS Technical Report J-121*DBCS Support Usage Guide* (日本語)を参照してください。

たとえば、PIPE IMM で使用するサーバー名 MOTIF の指定に、FSIMMOPT オプションを使用できます。

-fsimmopt PIPE:MOTIF

関連項目

概念的な情報:

- [5 章, “2 バイト文字セット\(DBCS\)” \(53 ページ\)](#)

システムオプション:

- [“FSIMM システムオプション” \(971 ページ\)](#)

LOCALE システムオプション

地域の言語、地域の規則および文化を反映する、SAS セッションの属性一式を指定します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

デフォルト: English_UnitedStates

構文

LOCALE=*locale-name*

必須引数

locale-name

ロケール値(SAS 名および POSIX 名)の一覧については、[表 21.1](#) を参照してください。

詳細

LOCALE=システムオプションは、地域の規則、言語および文化を反映するロケールの指定に使用します。

LOCALE=システムオプションの値に ENCODING=システムオプションの値との互換性がない場合、文字セットエンコーディングは、ENCODING=システムオプションの値によって決まります。

DBCS=システムオプションがアクティブな場合、DBCSTYPE=および DBCSLANG=システムオプションの値によってロケールと文字セットエンコーディングが決まります。

LOCALE=の値を設定すると、次のシステムオプションの値は、明示的に値を指定しない限り、変更されます。

ENCODING=

設定したロケールには、SAS が実行する動作環境の多くで使用される一般的なエンコーディング値があります。ENCODING= オプションが構成ファイルやコマンドラインに明示的に指定されていない場合、SAS は LOCALE とオペレーションシステムのデフォルトのエンコーディングを使用します。LOCALE は明示的に指定されるか、デフォルトが使用されます。ENCODING=システムオプションが設定された場合、TRANTAB=システムオプションも設定されます。

DATESTYLE=

LOCALE=が設定されると、DATESTYLE=システムオプションには選択されたロケールに対応する値が使用されます。

DFLANG=

LOCALE=が設定されると、DFLANG=システムオプションは選択されたロケールに対応する値に設定されます。

PAPERSIZE=

LOCALE=が設定されると、PAPERSIZE=システムオプションは選択されたロケールに対応する値に設定され、ODS プリンターがそのロケールの優先測定単位(インチまたはセンチメートル)に設定されます。

関連項目

概念的な情報:

- [2 章, “各国語サポート関連のロケール” \(5 ページ\)](#)
- [表 21.1](#)

システムオプション:

- [“ENCODING システムオプション” \(968 ページ\)](#)
- [“DATESTYLE= システムオプション” \(960 ページ\)](#)
- [“DFLANG= システムオプション” \(966 ページ\)](#)
- [“PAPERSIZE= システムオプション” \(SAS システムオプション: リファレンス\)](#)
- [“TRANTAB= システムオプション” \(988 ページ\)](#)

LOCALEDATA システムオプション

ロケール情報のソースデータベースを指定します。

該当要素: 構成ファイル、SAS 起動時

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

デフォルト: SASLOCALE

構文

LOCALEDATA= [SASLOCALE](#) | [REGISTRY](#)

構文の説明

SASLOCALE

内部 SAS データベースのロケール定義を使用します。

REGISTRY

SAS レジストリからのロケール定義を使用します。

詳細

SASLOCALE 値システムオプションは、ロケール定義のソースとして内部データベースを使用することを示します。このデータベースでは、カスタマイズは行えません。

REGISTRY 値は、外部データベースを指定します。外部データベースでは、PROC LOCALEDATA を使用してロケール定義をカスタマイズできます。

LOGLANGCHG システムオプション

SAS ログの言語を変更できるかどうかを指定します。

該当要素: 構成ファイル、SAS 起動時

PROC OPTIONS LOGCONTROL
GROUP=

デフォルト: UNICODE サーバーを除くすべてのサーバーでは、LOGLANGCHG が NOLOGLANGCHG (OFF)に設定されます。

構文

LOGLANGCHG

NOLOGLANGCHG

構文の説明

LOGLANGCHG

起動後に SAS ログメッセージの言語を変更できるようにする場合に指定します。

NOLOGLANGCHG

起動後に SAS ログメッセージの言語を変更できないようにする場合に指定します。

詳細

LOGLANGCHG は起動オプションであり、構成ファイルまたは SAS 起動コマンドラインで指定されます。起動後に値を変更することはできません。

結果は条件によって異なります。

ODSLANGCHG	LOGLANGCHG	結果
ON	ON	ODS 出力および SAS ログに使用する言語の変更可。変更には LSWLANG=オプション設定を使用。
OFF	OFF	ODS 出力と SAS ログ出力に使用する言語の変更不可。
OFF	ON	SAS ログに使用する言語の変更可能。変更には LSWLANG=オプション設定を使用。
ON	OFF	ODS 出力に使用する言語の変更可能。変更には LSWLANG=オプション設定を使用。

関連項目

- [“ODSLANGCHG システムオプション” \(982 ページ\)](#)
- [“LSWLANG システムオプション” \(978 ページ\)](#)
- [“LOGLANGENG システムオプション” \(976 ページ\)](#)

LOGLANGENG システムオプション

起動後に LOCALE オプションを設定した場合に、SAS ログメッセージテキストに英語を使用するかを指定します。

該当要素: 構成ファイル、SAS 起動時

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=
デフォルト: NOLOGLANGENG

構文

LOGLANGENG

NOLOGLANGENG

構文の説明

LOGLANGENG

英語の SAS ログメッセージを表示するように言語スイッチ機能を使用します。

NOLOGLANGENG

英語の SAS ログメッセージを表示するように言語スイッチ機能を使用しません。

詳細

LOGLANGENG オプションと、LOGLANGCHG、ODSLANGCHG および LSWLANG オプションは互いに影響します。

- LOGLANGENG が設定されると、LOGLANGCHG と LSWLANG=EN が自動的に設定されます。
- ODSLANGCHG を指定しても、LOGLANGENG を指定できます。
- NOLOGLANGENG は、SAS セッションに影響しません。
- 起動時は、構成ファイル(!sasroot\nls)と LOCALE オプションに基づき SAS メッセージの言語が決定されます。起動後は LOCALE オプションを設定した場合でも、LOGLANGENG オプションを指定しているならば、SAS ログは英語で出力されます。
- 起動時に LOGLANGENG を指定すると、セッション中に LOCALE の値が変更された場合でも、SAS ログは引き続き英語で出力されます。
- LOGLANGENG を指定すると、LOGLANGCHG は自動的に ON に設定され、英語のメッセージが SAS ログに書き込まれます。
- LOGLANGENG が OFF に設定され、LOGLANGCHG が ON に設定されている場合、SAS ログは LSWLANG=オプションによって制御されます。
- ODSLANGCHG を指定すると、ODS テキストは LSWLANG=オプションによって制御されます。
- 起動時に LOGLANGENG を指定せずに、LOCALE をセッション中に変更した際、起動時に設定した言語で SAS ログが出力されることがあります。この現象は、LOGLANGCHG と LSWLANG=をともに指定し、LSWLANG の値に LOCALE 以外を指定している場合に発生します。

例

例として、LOGLANGENG が指定されているフランス語の SAS セッションを考えます。

LOGLANGENG が設定されている場合、LOGLANGCHG は無効です。この SAS セッションでは、SAS ログは常に英語で出力されますが、ODS 出力の言語は現在の SAS 構成に基づき決定されます。

関連項目

- [“LSWLANG システムオプション” \(978 ページ\)](#)
- [“LOGLANGCHG システムオプション” \(975 ページ\)](#)
- [“ODSLANGCHG システムオプション” \(982 ページ\)](#)

LSWLANG システムオプション

SAS 起動時に LOGLANGCHG オプションまたは ODSLANGCHG オプションを設定する際に、言語切り替え機能で使用する言語を指定します。

該当要素:	構成ファイル、SAS 起動時
カテゴリ:	言語コントロール
PROC OPTIONS GROUP=	LANGUAGECONTROL
デフォルト:	LOCALE

構文

LSWLANG=[LOCALE](#) | *language*

必須引数

LOCALE

LOCALE 引数を指定すると、SAS 9.4 より前の動作が維持されます。SAS メッセージテキストの言語として、LOCALE=オプションの値が使用されます。

language

ODS 出力と SAS ログ出力に使用する言語を指定します。次の値を指定できません。

コード	言語
EN	英語
FR	フランス語
IT	イタリア語
DE	ドイツ語
ES	スペイン語
ZH	簡体字中国語
ZT	繁体字中国語
JA	日本語
KO	韓国語
PL	ポーランド語
RU	ロシア語

関連項目

- [“ODSLANGCHG システムオプション” \(982 ページ\)](#)
- [“LOGLANGCHG システムオプション” \(975 ページ\)](#)
- [“LOGLANGENG システムオプション” \(976 ページ\)](#)

MAPEBCDIC2ASCII= システムオプション

EBCDIC から ASCII へのトランスコードと、ASCII から EBCDIC へのトランスコードで SAS ユーザーが使用する変換テーブルを指定します。

該当要素: 構成ファイル、SAS 起動時
 カテゴリ: 環境コントロール: 言語コントロール
 PROC OPTIONS LANGUAGECONTROL
 GROUP=
 別名: MAPE2A

操作: MAPEBCDIC2ASCII=システムオプションは、SAS セッションに使用する変換テーブルを指定します。MAPEBCDIC2ASCII で指定されたテーブルは、\$EBCDIC 出力形式や\$ASCII 出力形式などのコンポーネントで使用されます。

構文

MAPEBCDIC2ASCII= TRANTAB *catalog-entry*

必須引数

catalog-entry

変換テーブルを含む SAS カタログの TRANTAB エントリを指定します。*entry-name.type* を指定した場合、SAS は指定された名前を SASUSER.PROFILE、SASHELP.LOCALE、SASHELP.HOST の順に検索します。

詳細

MAPEBCDIC2ASCII=は、各国語の要件をサポートします。

注意

利用方法がよくわからない場合は変換テーブルを変更しないでください。 変換テーブルは SAS により各国語サポートを提供するために内部的に利用されます。変換テーブルの利用方法がよくわからない場合や、技術的な支援がない場合は設定を変更しないでください。

NLDECSEPARATOR システムオプション

小数点にロケール依存の数値出力を生成するか、引き続き米国英語設定の数値形式を出力するかを指定します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント

PROC OPTIONS
GROUP= LANGUAGECONTROL

デフォルト: NONLDECSEPARATOR

制限事項: ODS GRAPHICS が有効なプロシジャで NLDECSEPARATOR を使用すると、構文エラーまたは null デバッグエラーが生成されます。数値変数には適切な NL 出力形式を使用してください。

構文

NLDECSEPARATOR

NONLDECSEPARATOR

必須引数

NLDECSEPARATOR

別名 NLD

小数点区切り文字について、ロケール依存の数値出力を有効にします。

NONLDECSEPARATOR

小数点区切り文字について、ロケール依存の数値出力を無効にします。

詳細

小数点としてピリオドを使用する数値を生成する場合は、NLDECSEPARATOR オプションとともに BESTDOTX 出力形式を使用してください。BESTDOTX 出力形式を使用しないと、出力にカンマが含まれます(例: 1,2)。SAS は、この出力を数値とは解釈しません。この値が後続の代入ステートメントで使用されると、エラーになります。

CALL SYMPUT などの一部の SAS 関数は、数値の文字列への変換にデフォルトで BEST 出力形式を使用します。NLDECSEPARATOR が有効の場合、BEST 出力形式は現在のロケールの指定に応じた区切り文字を使用します。ロケールで区切り文字としてピリオドが使用されている場合、出力される文字列はこの値を数値として評価する以降の SAS ステートメントで使用できます。ただし、区切り文字がカンマなど他の文字である場合は、SAS コードがそれを数値として使用しようとしたときに構文エラーになります。

このタイプのエラーを避けるため、値の出力形式に BESTDOTX を使用してから CALL SYMPUT ステートメント内で代入してください。BESTDOTX は、出力で小数点として必ずピリオドを使用します。たとえば、次の SAS コードでは、変数 X の値の出力形式に BESTDOTX が使用されてから、変数 X の値がマクロ変数 &MACX に代入されています。&MACX の値が以降で SAS 変数 Y に代入されても、構文エラーになりません。

```
data a;
  x=1.2;
  CALL SYMPUT('macx', put(x, bestdotx.));
run;

data b;
  y=&macx
run;
```

関連項目

[“BESTDOTXw. 出力形式” \(149 ページ\)](#)

ODSLANGCHG システムオプション

ODS 出力のテキスト言語を変更できるかを設定します。

該当要素: 構成ファイル、SAS 起動時

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

デフォルト: UNICODE サーバー以外のサーバーでは、ODSLANGCHG は設定しません。

ヒント: 言語切り替え機能に使用される言語は、LSWLANG=オプションで制御されます。デフォルトでは、LSWLANG は LOCALE に設定され、言語切り替え機能の言語には LOCALE の言語が使用されます。

構文

ODSLANGCHG

NOODSLANGCHG

構文の説明

ODSLANGCHG

起動後に ODS 出力の言語を変更できるようにする場合に指定します。

NOODSLANGCHG

起動後に ODS 出力の言語を変更できないようにする場合に指定します。

詳細

言語切り替え機能(LSW)により、起動後に SAS メッセージと ODS テンプレートの言語を変更できます。ODS 出力の言語切り替え機能を使用できるようにするには、ODSLANGCHG を指定する必要があります。

起動時は、構成ファイルに基づき、SAS メッセージと ODS テンプレートの言語が決定されます。ODSLANGCHG オプションが指定されると、地域化された SAS メッセージと ODS テンプレートが使用できる場合は、ODS 出力の言語は LSWLANG=設定に合わせて変更されます。

ユーザーは ODSLANGCHG は指定できますが、ロケール言語への翻訳はできません。たとえば、ODSLANGCHG を指定している場合に、フランス語の SAS セッション

ンを開始してからロケールをギリシャ語に設定すると、NLDATE はギリシャ語で表示されますが、出力はフランス語で表示されます。SAS ではギリシャ語への翻訳は行っていないため、出力はフランス語で表示されます。ODSLANGCHG が指定されて LSWLANG=LOCALE が設定されている場合、ODS PATH は LOCALE 設定に基づいて更新されて、地域化されたテンプレートが含まれるようになります。

比較

また、その地域化されたメッセージが、ODS 出力の生成に使用されます。

起動時に ODSLANGCHG が指定されない場合、ODS 出力は構成ファイルに指定された言語で表示されます。

例

例 1 では、ODSLANGCHG が指定されていない(NOODSLANGCHG が指定された)フランス語のサーバーを考えます。

フランス語のクライアントアプリケーションがこのサーバーに接続すると、ODS 出力や、NL 出力形式を使用してフォーマットされた日付はフランス語で表示されます。英語のクライアントアプリケーションがこのサーバーに接続し、サーバーのロケールを英語に変更すると、ODS 出力はフランス語で表示され、NL 出力形式を使用してフォーマットされた日付は英語で表示されます。

例 2 では、ODSLANGCHG が指定され、LSWLANG=LOCALE が設定されたフランス語のサーバーを考えます。

フランス語のクライアントアプリケーションがこのサーバーに接続すると、ODS 出力や、NL 出力形式を使用してフォーマットされた日付はフランス語で表示されます。英語のクライアントアプリケーションがこのサーバーに接続し、サーバーのロケールを英語に変更すると、ODS 出力や、NL 出力形式を使用してフォーマットされた日付は英語で表示されます。

例 3 では、ODSLANGCHG が指定され、LSWLANG=English が設定されたフランス語のサーバーを考えます。

フランス語のクライアントアプリケーションがこのサーバーに接続すると、ODS 出力は英語で表示され、NL 出力形式を使用してフォーマットされた日付はフランス語で表示されます。英語のクライアントアプリケーションがこのサーバーに接続し、サーバーのロケールを英語に変更すると、ODS 出力や、NL 出力形式を使用してフォーマットされた日付は英語で表示されます。

関連項目

- [“LOGLANGENG システムオプション” \(976 ページ\)](#)
- [“LOGLANGCHG システムオプション” \(975 ページ\)](#)
- [“LSWLANG システムオプション” \(978 ページ\)](#)

RSASIOTRANSERROR システムオプション

リモートアプリケーションから無効なデータが読み込まれたときにトランスコーディングエラーを表示します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント
カテゴリ: ファイル: SAS ファイル
PROC OPTIONS
GROUP= SASFILES
デフォルト: RSASIOTRANSERROR

構文

RSASIOTRANSERROR

NORSASIOTRANSERROR

構文の説明

RSASIOTRANSERROR

リモートアプリケーションから無効な値が読み込まれた場合に、トランスコーディングエラーを表示するように指定します。

NORSASIOTRANSERROR

リモートアプリケーションから無効な値が読み込まれた場合に、トランスコーディングエラーを表示しないように指定します。

詳細

RSASIOTRANSERROR システムオプションを使用すると、SASIO のリモート ユーザーは無効なデータ値を無視できます。通常は、無効なデータ値がリモートアプリケーションから読み込まれると、トランスコーディングエラーが発生します。

SORTSEQ= システムオプション

SORT および SQL プロシジャが現在の SAS セッションで使用する、言語固有の照合順序を指定します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント
カテゴリ: SORT: プロシジャオプション

PROC OPTIONS SORT
GROUP=

構文

SORTSEQ=<*sort-table*> | <LINGUISTIC>

構文の説明

sort-table

PROC TRANTAB で作成した変換テーブルを指定します。

LINGUISTIC

指定された言語のルールに従って文字を並べ替えます。ルールおよびデフォルトの照合順序は、現在のロケール設定で指定された言語に基づきます。実装は、International Components for Unicode (ICU)ライブラリによって提供されています。詳細については“[Linguistic Sorting of Data Sets and ICU](#)” (*Base SAS Procedures Guide*)を参照してください。

例

この例は、SORTSEQ=オプションと一緒に PROC SORT および PROC SQL を使用した場合の機能を示しています。

```
options sortseq=linguistic;
proc sort data=sashelp.class out=foo1;
by name;
run;
proc sql;
create table foo2 as select * from sashelp.class order by name;
quit;
run;
```

言語オプション(SORTSEQ=LINGUISTIC)が Base SAS プロシジャの PROC SORT で指定された場合、SAS システムは ICU 照合を提供します。また、SQL プロシジャ内で SORTSEQ=を使用することや、システムオプション SORTSEQ=LINGUISTIC を指定することによっても、言語照合を指定できるようになりました。

注: SORTSEQ=LINGUISTIC システムオプションが指定された場合、PROC SORT と PROC SQL のみが影響を受けます。

言語オプションが指定された場合、SAS システムは ICU ライブラリを、Unicode 照合アルゴリズム(UCA)のリファレンス実装および事実上の標準として使用します。

SAS Viya プラットフォームでは、SAS に組み込まれ、PROC SORT で使用される ICU ライブラリのバージョンは ICU 63 です。この ICU バージョンでは、バージョン 34 の Unicode Common Locale Data Repository (CLDR)のロケールデータを使用します。

関連項目

- [“照合順序” \(17 ページ\)](#)

システムオプション:

- [“TRANTAB= システムオプション” \(988 ページ\)](#)

TIMEZONE= システムオプション

ユーザーローカルタイムゾーンを指定します。

該当要素: 自動実行ファイル、OPTIONS ステートメント

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS
GROUP= LANGUAGECONTROL

別名: TZ=

制限事項: TIMEZONE=システムオプションは、In-Database 処理には影響しません。SAS Embedded Process は、データソースのタイムゾーンを使用します。詳細については、[“In-Database Processing” \(SAS In-Database Products: User's Guide\)](#)を参照してください。

注: 別名 TZ=は、制限付きオプション構成ファイルでは使用できません。TIMEZONE=は、制限付きオプション構成ファイルで使用する必要があります。サイト管理者はこのオプションを制限できます。詳細については、[“制限されたオプション” \(SAS システムオプション: リファレンス\)](#)を参照してください。

構文

TIMEZONE=['time-zone-name'](#) | ['time-zone-ID'](#)

構文の説明

time-zone-name

3 または 4 文字のタイムゾーン名を指定します。たとえば、EST は東部標準時間のタイムゾーン名です。

デフォルト BLANK、SAS Compute Server のタイムゾーンと SAS クライアントのタイムゾーンが同じであることを示します

参照項目: タイムゾーン名のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名” \(1145 ページ\)](#)を参照してください。

<->

英国グリニッジからのオフセット時間を<->*hh*<:*mm*<:*ss*>>: -という形式で指定します。

- 英国グリニッジの西のタイムゾーンオフセットを指定します。グリニッジの東のタイムゾーンオフセットには符号表記は必要ありません。
- hh* 1 または 2 桁を使って時間を指定します。オフセットを指定する場合、時間は必須です。
- mm* 1 または 2 桁を使って分を指定します。分はオプションです。
- ss* 1 または 2 桁を使って秒を指定します。秒はオプションです。

time-zone-ID

SAS で定義されている *region/area* の値を指定します。タイムゾーン ID を指定したとき、SAS が使用するタイムゾーンはタイムゾーン名と夏時間ルールを考慮して決定されます。

注 タイムゾーン ID は JAVA タイムゾーン名と互換性があります。

参照項目 タイムゾーン名のリストについては、[付録 2, “タイムゾーン ID とタイムゾーン名” \(1145 ページ\)](#)を参照してください。

詳細

システムが特定のタイムゾーンを使用するように、TIMEZONE=オプションはタイムゾーン ID またはタイムゾーン名に対して設定します。タイムゾーン設定は次の SAS コンポーネントに影響します。

- イベントやログに記録される時間
- データセット作成や変更の時間
- DATE()関数
- DATETIME()関数
- TIME()関数
- TODAY()関数
- タイムゾーン関数 TZONEOFF(), TZONEID(), TZONENAME(), TZONES2U(), TZONEU2S()
- タイムゾーン出力形式 B8601DXw., E8601DXw., B8601LXw., E8601LXw., B8601TXw., E8601TXw., NLDATMTZw., NLDATMTZW., NLDATMWZw.

タイムゾーンを設定するには、タイムゾーン ID とタイムゾーン名を指定します。タイムゾーン ID では、リージョンとエリアをスラッシュ(/)で区切ります。たとえば、America/New_York や Asia/Osaka がタイムゾーン ID です。

タイムゾーン名は 3 または 4 文字のタイムゾーンの名前です。たとえば、EST は東部標準時間で JST は日本標準時間です。SAS は、時間値を使う前に夏時間を考慮したタイムゾーンルールに則って時間を決定します。

タイムゾーン名には異なるロケールで有効のものもあります。たとえば、CST は中部標準時間、キューバ標準時間、そして中国標準時間を意味します。SAS は LOCALE=システムオプションの値を使って、どのリージョンとエリアを使えばいいのか判断します。TIMEZONE='CST'で LOCALE='zh_CN'の場合、SAS は Asia/Beijing タイムゾーンを使用します。タイムゾーン名がそのロケールに存在しない場合、シ

システムはすべてのタイムゾーンを検索して最初にマッチしたタイムゾーンを設定します。

このオプションが制限されていて TIMEZONE=の値がデフォルト値の BLANK である場合、タイムゾーンの動作はタイムゾーン情報を使用しないものになります。

関連項目

[“SAS でのタイムゾーンの指定” \(59 ページ\)](#)

TRANTAB= システムオプション

SAS のさまざまな構成要素で使用される変換テーブルを指定します。

該当要素: 構成ファイル、SAS 起動時、OPTIONS ステートメント

カテゴリ: 環境コントロール: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

デフォルト: SAS Viya プラットフォーム: 出荷時のデフォルト値は (lat1wlt1,wlt1lat1,lat1_ucs,lat1_lcs,lat1_ccl,,,,,)です。

操作: TRANTAB=システムオプションでは、ファイル転送を含む、SAS セッションで使用される変換テーブルを指定します。TRANTAB ステートメントでは、エクスポートまたは転送される SAS ファイルの文字セットに適用する、カスタマイズした変換テーブル(たとえば、EBCDIC 文字の ASCII 文字へのマッピング)を指定します。

構文

TRANTAB=(*catalog-entries*)

構文の説明

catalog-entries

変換テーブルを含む SAS カタログエントリを指定します。*entry-name.type* を指定した場合、SASUSER.PROFILE、SASUSER.HOST の順に検索されます。

詳細

変換テーブルは、10 個の位置を含むかっこで囲まれたリストで指定します。リスト内でのテーブルの位置によって指定する変換テーブルの種類が決まります。リストの各エントリはカンマで区切られます。次の位置と種類のリストを参照してください。

位置	変換テーブルの種類
1 番目	local-to-transport-format
2 番目	transport-to-local-format
3 番目	lowercase-to-uppercase
4 番目	uppercase-to-lowercase
5 番目	文字の分類
6 番目	スキャナ変換
7 番目	デルタ文字
8 番目	スキャナ文字の分類
9 番目	未使用
10 番目	DBCS ユーザーテーブル

注意

利用方法がよくわからない場合は変換テーブルを変更しないでください。 変換テーブルは SAS スーパーバイザーにより各国語サポートを提供するために内部的に利用されます。変換テーブルの利用方法がよくわからない場合や、技術的な支援がない場合は設定を変更しないでください。

1 つのテーブルを変更するには、変更しない他のテーブルには NULL エントリを指定します。たとえば、リストの 3 番目にある lowercase-to-uppercase テーブルを変更するには、uppercase を次のように指定します。

```
options trantab = ( , , new-uppercase-table);
```

他のテーブルは変更されません。OPTIONS プロシジャの出力には、TRANTAB=オプションのすべての指定が反映されるのではなく、最後の指定のみが反映されます。次に、例を示します。

```
options trantab = ( , , new-uppercase-table);
options trantab = ( , , new-lowercase-table);
```

PROC OPTIONS では、TRANTAB=の値は

(, , new-lowercase-table)と表示されますが、new-uppercase テーブルと new-lowercase テーブルの両方が有効な値です。

関連項目

[20 章, "TRANTAB プロシジャ," \(1067 ページ\)](#)

VALIDMEMNAME= システムオプション

SAS データセット、SAS データビューおよびアイテムストアの命名規則を指定します。

該当要素: 自動実行ファイル、OPTIONS ステートメント

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS SASFILES

GROUP=

デフォルト: 出荷時のデフォルト値は COMPATIBLE です。

適用対象: Base SAS エンジンと SPD エンジン

制限事項: VALIDMEMNAME=オプションは、テープエンジン V9TAPE、V8TAPE、V7TAPE、V6TAPE ではサポートされません。
VALIDMEMNAME の値に関係なく、メンバー名の最後に特殊文字#を付けて、その後 3 桁の数字を続けることはできません。これは、世代データセットの命名規則と競合するためです。このようなメンバー名を使用すると、結果としてエラーが発生します。

注: サイト管理者はこのオプションを制限できます。詳細については、“[制限されたオプション](#)” (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。

構文

VALIDMEMNAME=[COMPATIBLE](#) | [EXTEND](#)

構文の説明

COMPATIBLE

SAS データセット名、SAS データビュー名またはアイテムストア名が次の規則に従う必要があることを指定します。

- 名前の長さは、最大 32 文字まで指定できます。
- 名前は、英字(A から Z、a から z)またはアンダースコアで始める必要があります。開始文字以外には、英字、数字、アンダースコアを使用できます。
- 名前には、空白またはアンダースコア以外の特殊文字を含めることができません。
- 名前には大文字と小文字を混在させることができます。メンバー名は SAS 内部で大文字に変換されます。このため、メンバー名の大文字と小文字の組み合わせを変更して、異なる変数を表すことはできません。たとえば、**customer**、**Customer**、および **CUSTOMER** はすべて同じメンバー名を表します。ディスク上に名前がどのように保存されるかは、動作環境によって決まります。

別名 COMPAT

EXTEND

SAS データセット名、SAS データビュー名またはアイテムストア名が次の規則に従う必要があることを指定します。

- 名前には各国語文字を含めることができます。
- 名前には / \ * ? " < > | : - . 以外の特殊文字を含めることができます。

注: SPD エンジンでは、メンバー名のどこにも '.' (ピリオド) を使用できません。

- 名前には、少なくとも 1 文字が必要です(文字、数字、有効な特殊文字、および各国語文字)。
- 名前の長さは、最大 32 バイトまで指定できます。
- NULL のバイトは使用できません。
- 名前は空白または '.' (ピリオド) で始めることはできません。

注: SPD エンジンでは、メンバー名の最初の文字に '\$' を使用できません。

- メンバーが作成されるときに先頭と末尾の空白は削除されます。
- 名前には大文字と小文字を混在させることができます。メンバー名は SAS 内部で大文字に変換されます。このため、メンバー名の大文字と小文字の組み合わせを変更して、異なる変数を表すことはできません。たとえば、**customer**、**Customer**、および **CUSTOMER** はすべて同じメンバー名を表します。名前がどのように表示されるかは、動作環境によって決まります。

要件 VALIDMEMNAME=EXTEND が設定されていて、SAS データセット名、SAS データビュー名およびアイテムストア名に空白や特殊文字が含まれる場合、名前が数字で始まる場合や各国語文字で始まる場合、それらは SAS 名リテラルとして書き込まれなければなりません。パーセント記号(%)またはアンパサンド(&)のいずれかを使用する場合、SAS マクロ機能との交互作用を避けるために、名前リテラルに単一引用符を使用する必要があります。詳細については、“[SAS Name Literals](#)” ([SAS プログラマガイド: エッセンシャル](#))を参照してください。

動作環境 LINUX 動作環境では、VALIDMEMNAME=EXTEND が設定されている場合、すべての Base SAS ウィンドウで拡張規則がサポートされます。

LINUX 動作環境では、SAS ファイルを物理名で直接参照する場合、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。ファイルの物理参照でピリオドを含む SAS メンバー名を使う場合、ファイルの拡張子を必ず追加してください。たとえば、データセット名 my.member を物理ファイルとして参照する場合は、SET ステートメント set './saslib/my.member.sas7bdat' のように、参照名にファイル拡張子 sas7bdat を追加します。

ヒント 名前は大文字で表示されます。

参照項目: “SAS 名の拡張” (SAS プログラマガイド: エッセンシャル)

例 data “August Purchases”n;
data ‘Años de empleo’n.;

注意 SAS 全体で、名前リテラル構文に 32 バイト制限を超える SAS メンバー名を指定したり、埋め込まれている引用符が多すぎたりする場合、予期しない結果になる可能性があります。VALIDMEMNAME=EXTEND システムオプションの目的は、埋め込み空白や各国語文字を許可するなど、他の DBMS メンバーの命名規則との互換性を持たせることです。

詳細

VALIDMEMNAME=EXTEND の場合、SAS データセット名、SAS データビュー名およびアイテムストア名でが使用できる有効な文字は、次の文字まで拡張されます。

- 各国語文字
- サードパーティのデータベースでサポートされる文字
- ファイル名で一般的に使用される文字

DATA、VIEW および ITEMSTORE という SAS メンバーの種類のみで、文字の拡張がサポートされます。CATALOG や PROGRAM などのその他のメンバーの種類では、拡張文字はサポートされません。関連付けられた DATA メンバーがある場合のみ存在する INDEX と AUDIT では、拡張文字がサポートされます。

関連項目

- “SAS 単語” (SAS プログラマガイド: エッセンシャル)

システムオプション:

- “VALIDVARNAME= システムオプション” (992 ページ)

VALIDVARNAME= システムオプション

SAS セッション中に作成および処理可能な有効な SAS 変数名の規則を指定します。

該当要素: 自動実行ファイル、OPTIONS ステートメント

カテゴリ: ファイル: SAS ファイル

PROC OPTIONS SASFILES
GROUP=

デフォルト: 出荷時のデフォルト値は V7 です。

操作: SAS/ACCESS のコンテキストでは、VALIDVARNAME= は DBMS の列名を SAS に読み込んで SAS 変数名に変換する方法を制御します。一般に、VALIDVARNAME= は、任意の SAS 変数を作成するときに使用しなければならない規則を指定します。

注: サイト管理者はこのオプションを制限できます。詳細については、“[制限されたオプション](#)” (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。

構文

VALIDVARNAME=V7 | UPCASE | ANY

構文の説明

V7

変数名が次の規則に従う必要があることを指定します。

- 名前は最大 32 文字指定できます。
- 最初の文字には、英字(A から Z、a から z)またはアンダースコアを使用する必要があります。開始文字以外には、英字、数字、アンダースコアを使用できます。
- 末尾の空白は無視されます。変数名の配置は左揃えになります。
- 変数名には、空白またはアンダースコア以外の特殊文字を含めることができません。
- 変数名には大文字と小文字を混在させることができます。SAS では、変数を初めて参照した際に使用されていた大文字と小文字の組み合わせと同じ組み合わせで変数名の保存、書き込みが行われますが、変数名の処理時には、SAS 内部ではすべて大文字に変換されます。このため、変数名の大文字と小文字の組み合わせを変更して、異なる変数を表すことはできません。たとえば、**cat**、**Cat**、および **CAT** はすべて同じ変数を表します。
- 変数には、特殊な SAS 自動変数名(_N_、_ERROR_ など)や、変数リスト名(_NUMERIC_、_CHARACTER_、_ALL_ など)を割り当てないでください。

例 `season='summer';`
`percent_of_profit=percent;`

UPCASE

変数名が V7 と同じ規則に従うように指定します。ただし、SAS の以前のバージョンにある変数名が大文字という規則は除きます。

ANY

SAS 変数名が次の規則に従う必要があることを指定します。

- 名前には、空白、各国語文字、特殊文字、マルチバイト文字など、どの文字でも使用できます。
- 名前は最大 32 バイトです。

- 名前に NULL のバイトを含めることはできません。
- 先頭の空白は保持されますが、末尾の空白は無視されます。
- 名前には、少なくとも 1 文字が必要です。すべて空白の名前は使用できません。
- 名前には大文字と小文字を混在させることができます。SAS では、変数を初めて参照した際に使用されていた大文字と小文字の組み合わせと同じ組み合わせで変数名の保存、書き込みが行われますが、変数名の処理時には、SAS 内部ではすべて大文字に変換されます。このため、変数名の大文字と小文字の組み合わせを変更して、異なる変数を表すことはできません。たとえば、**cat**、**Cat**、および **CAT** はすべて同じ変数を表します。

要件 VALIDVARNAME システムオプションが V7 に設定されているときに、有効な文字(英数字またはアンダースコア)以外の文字を使用する場合は、変数名を名前リテラルとして表す必要があり、VALIDVARNAME=ANY に設定する必要があります。名前にパーセント記号(%)またはアンパサンド(&)のいずれかが含まれている場合、SAS マクロ機能との交互作用を避けるために、名前リテラルに単一引用符を使用する必要があります。[“SAS Name Literals” \(SAS プログラマガイド: エッセンシャル\)](#)および[“名前リテラル使用時のエラーの回避” \(SAS プログラマガイド: エッセンシャル\)](#)を参照してください。

参照項目: [“SAS 名の拡張” \(SAS プログラマガイド: エッセンシャル\)](#)

例 '% of profit'n=percent;
 'items@warehouse'n=itemnum;

注意 SAS 全体で、名前リテラル構文に 32 バイト制限を超えたり、埋め込まれている引用符が多すぎたりする SAS 変数名を使用すると、予期しない結果になる可能性があります。VALIDVARNAME=ANY システムオプションの目的は、埋め込み空白や各国語文字を許可するなど、他の DBMS 変数(列)の命名規則との互換性を持たせることです。

関連項目

- [“ほとんどの SAS 名の規則” \(SAS プログラマガイド: エッセンシャル\)](#)

システムオプション:

- [“VALIDMEMNAME= システムオプション” \(990 ページ\)](#)

URLENCODING= システムオプション

SAS セッションエンコーディングと UTF-8 エンコーディングのどちらを使用して URLENCODE 関数と URLDECODE 関数の引数が解釈されるのかを指定します。

該当要素: SAS Environment Manager, 自動実行ファイル、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: 言語コントロール

PROC OPTIONS LANGUAGECONTROL
GROUP=

参照項目: ["URLENCODING= システムオプション" \(SAS システムオプション: リファレンス\)](#)

構文

URLENCODING=[SESSION](#) | [UTF8](#)

9 部

各国語サポート関連のコマンド、ステートメント、プロシジャオプション

17 章

各国語サポート関連のコマンド、ステートメントおよびプロシジャオプションのディクショナリ	999
---	-----

17

各国語サポート関連のコマンド、 ステートメントおよびプロシジャ オプションのディクショナリ

カテゴリ別の各国語サポート関連のコマンド、ステートメント、およびプロシジャオプション	1000
ディクショナリ	1001
CHARSET=オプション	1001
照合順序オプション	1002
CORRECTENCODING=オプション	1009
CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、 CVPMULTIPLIER=、CVPVARCHAR=、CVPEXCLUDE=、 CVPINCLUDE=オプション	1010
ENCODING=オプション	1021
INENCODING= OUTENCODING=オプション	1024
ODSCHARSET=オプション	1026
ODSTRANTAB=オプション	1027
PROC SQL の TRANSCODE=列修飾子	1028
RENCODING=オプション	1029
TRANSCODE=オプション	1032
TRANTAB=オプション	1034
XMLENCODING=オプション	1035
TRANTAB ステートメント	1036

カテゴリ別の各国語サポート関連のコマンド、ステートメント、およびプロシジャオプション

選択した SAS ステートメントのオプションに対するデータアクセスカテゴリとデータセットコントロールカテゴリは、各国語サポートの影響を受けます。次の表に、ステートメントオプションの概要を示します。詳細については、各ステートメントオプションのディクショナリエントリを参照してください。

カテゴリ	言語要素	説明
ODS: サードパーティによるフォーマティング	CHARSET=オプション (p. 1001)	出力の META 宣言で生成される文字セットを指定します。
	TRANTAB=オプション (p. 1034)	文字データをトランスコーディングする場合に使用する変換テーブルを指定します。
情報	TRANSCODE=オプション (p. 1032)	文字変数をトランスコードするかどうかを指示する、ATTRIB ステートメントの属性(1 つ以上の変数に出力形式、入力形式、ラベル、長さに関連付ける)を指定します。
データアクセス	CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE=オプション (p. 1010)	SAS ファイルをトランスコードするために必要とされる文字変数の属性を指定します。
	ENCODING=オプション (p. 1021)	外部ファイルの入力または出力処理のエンコーディングを優先指定してトランスコードします。
	INENCODING=OUTENCODING=オプション (p. 1024)	SAS ライブラリの SAS データセットの読み込みまたは書き込み時のエンコーディングを無効にして変更します。
	ODSCHARSET=オプション (p. 1026)	出力の META 宣言で生成される文字セットを指定します。
	ODSTRANTAB=オプション (p. 1027)	出力ファイルの XML ドキュメントをトランスコードするときに使用する変換テーブルを指定します。
	RENCODING=オプション (p. 1029)	EBCDICANY または ASCIIANY セッションエンコーディングを使用する SAS/SHARE サーバーセッションのデータをトランスコードする場合に使用する、ASCII ベースまたは EBCDIC ベースのエンコーディングを指定します。

カテゴリ	言語要素	説明
	XMLENCODING=オプション (p. 1035)	外部ドキュメントのインポートまたはエクスポート時に、XML ドキュメントのエンコーディングより優先される値を指定します。

ディクショナリ

CHARSET=オプション

出力の META 宣言で生成される文字セットを指定します。

該当要素: ODS MARKUP ステートメントと ODS HTML ステートメント

カテゴリ: ODS: サードパーティによるフォーマティング

構文

CHARSET=*character-set* ;

必須引数

character-set

HTML 出力の META タグで使用する文字セットを指定します。

エンコーディングの例は UTF-8 です。インターネットで使用する公式の文字セットは、IANA (Internet Assigned Numbers Authority)によって登録されます。IANA は、ポート、プロトコルとエンタープライズ番号、オプション、コードとタイプなどのさまざまなインターネットプロトコルパラメーターの中央レジストリです。文字セット値の一覧については、www.unicode.org/reports/tr22/index.html および www.iana.org/assignments/character-sets を参照してください。

character set は、このコンテキストの *encoding-value* に似ています。ただし、*character set* という用語は、インターネットでの使用に適したエンコーディングを識別するために使用されます。

例: ODS MARKUP ステートメントの META 宣言で生成される出力

```
<META http-equiv="Content-Type" content="text/html; charset=utf-8">
```

関連項目

概念的な情報:

- [3 章, “各国語サポート関連のエンコーディング” \(11 ページ\)](#)

ステートメント:

- [“ODS MARKUP Statement” \(SAS Output Delivery System: User's Guide\)](#)

照合順序オプション

PROC SORT の照合順序を指定します。

該当要素: PROC SORT ステートメント

注: PROC SORT ステートメントは、SAS データセットのオブザベーションを 1 つ以上の文字変数または数値変数を基準に並べ替えます。

構文

PROC SORT *collating-sequence-option* <*other options*> ;

オプション

オプションには、1 つの *collating-sequence-option* と複数の *other options* を含めることができます。2 種類のオプションの順序は重要でなく、両方の種類を同じ PROC SORT ステップに含める必要はありません。PROC SORT *collating-sequence-option* についてのみ次に説明します。

動作環境の情報: DANISH、FINNISH、NORWEGIAN、SWEDISH のいずれかの *collating-sequence-option* に対する動作環境固有の動作の詳細については、現在の動作環境向けの SAS ドキュメントを参照してください。

ASCII

ASCII 照合順序を使用して文字変数を並べ替えます。このオプションは、EBCDIC がネイティブ照合順序のシステムで ASCII 並べ替えを実行する場合にのみ必要です。

DANISH NORWEGIAN

デンマーク語とノルウェー語に従って文字を並べ替えます。

デンマーク語とノルウェー語の照合順序は、[図 17.6 \(1004 ページ\)](#)に示されています。

EBCDIC

EBCDIC 照合順序を使用して文字変数を並べ替えます。このオプションは、ASCII がネイティブ照合順序のシステムで EBCDIC 並べ替えを実行する場合にのみ必要です。

POLISH

ポーランド語の規則に従って文字を並べ替えます。

FINNISH SWEDISH

フィンランド語とスウェーデン語の規則に従って文字を並べ替えます。フィンランド語とスウェーデン語の照合順序は、[図 17.6 \(1004 ページ\)](#)に示されています。

NORWEGIAN

DANISH を参照してください。

SWEDISH

FINNISH を参照してください。

SORTSEQ=collating-sequence

照合順序を指定します。*collating-sequence* には、collating-sequence-option、変換テーブル、エンコーディング、キーワード LINGUISTIC のいずれかを指定できます。1 つの照合順序のみ指定できます。詳細については、“[照合順序](#)” (17 ページ)を参照してください。

照合順序の説明を次に示します。

collating-sequence-option | translation_table

SAS で提供される変換テーブルかユーザー定義の変換テーブル、または 1 つの PROC SORT ステートメントの Collating-Sequence-Options を指定します。SORTSEQ=とともに PROC TRANTAB と PROC SORT を使用した例については、“[例 6: 並べ替えへのさまざまな変換テーブルの使用](#)” (1089 ページ)を参照してください。

使用可能な変換テーブルは、次のとおりです。

- ASCII
- DANISH
- EBCDIC
- FINNISH
- ITALIAN
- NORWEGIAN
- POLISH
- REVERSE
- SPANISH
- SWEDISH

次の図に、各言語で英数字が並べ替えられる方法を示します。

図 17.1 英数字の各言語での並べ替え

```

Danish:      0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅabcde fghijklmnopqrstuvwxyzæøå
Finnish:     0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZÅÖabcde fghijklmnopqrstuvwxyzåö
Italian:     0123456789AÀBÇÇDEÉÊËFGHIÌJJKLMNOÒPQRSTUÙVWXYZaàbcçdeéêëfghiìjklmnoòpqrstuùvwxyz
Norwegian:   0123456789ABCDEF GHIJ KLMNOPQRSTU VWXYZÆØÅabcde fghijklmnopqrstuvwxyzæøå
Spanish:     0123456789AÁaśBbCcDdEÉeëFfGgHhIíiJjKkLlMmNnÑñOÓoóPpQqRrSsTtUÚuúÜüVvWwXxYyZz
Swedish:     0123456789ABCDEF GHIJ KLMNOPQRSTU VWXYZÅÖabcde fghijklmnopqrstuvwxyzåö

```

制限事項 PROC SORT ステップでは、1 つの collating-sequence-option のみ指定できます。

ヒント SORTSEQ=照合順序オプションは、かっこを使用せず、引数を関連付けずに指定します。照合順序を指定する例を次に示します。proc sort data=mydata SORTSEQ=ASCII;

encoding-value

エンコーディング値を指定します。この結果は、指定したエンコーディングで表示される文字データのバイナリ照合と同じです。サポートされるエンコーディング値は“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ)を参照してください。

制限事項 SORTSEQ=オプションで指定したエンコーディングを認識する SAS System の一部またはプロシジャは、PROC SORT のみです。

ヒント エンコーディング値に英数字またはアンダースコア以外の文字が含まれる場合、その値を引用符で囲む必要があります。

参照項目: “データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ)で指定可能なエンコーディングの一覧。

LINGUISTIC<(collating-rules)>

指定した言語のルールに従って文字を並べ替える、言語照合を指定します。ルールおよびデフォルトの照合順序オプションは、現在のロケール設定で指定した言語に基づきます。International Components for Unicode (ICU) ライブラリにより実装され、Unicode 照合アルゴリズム(UCA)と大部分で互換性のある結果を作成します。

別名 UCA

制限事項 SORTSEQ=LINGUISTIC オプションは、PROC SORT および PROC SQL のみ使用できます。照合規則は、PROC SORT でサポートされており、システムオプションや PROC SQL の SORTSEQ=オプションではサポートされていません。

注 また、SQL プロシジャ内で SORTSEQ=を使用することや、SORTSEQ=LINGUISTIC システムオプションを指定することで、言語照合を指定できるようになりました。詳細については、“SORTSEQ= システムオプション” (984 ページ)および“SORT Procedure” (Base SAS Procedures Guide)を参照してください。

ヒント collating-rules はかっこで囲む必要があります。複数の照合ルールを指定できます。

言語照合で並べ替えたデータセットで BY 処理を実行する場合、データセットが適切に処理されるように **NOBYSORTED** システムオプションの指定が必要な場合があります。BY 処理は、照合順序処理とは異なる方法で実行されます。

参照 言語照合の詳細については、“**照合順序**” (17 ページ)を参照してください。
項目:

PROC SORT での言語的な並べ替えについては、“**SORTSEQ=sort-table | LINGUISTIC**” (*SAS SQL プロシジャユーザーガイド*)を参照してください。

Unicode 照合アルゴリズム(UCA)の仕様は、<http://www.unicode.org> で確認できます。

LINGUISTIC オプションで指定可能な collation-rules を示します。これらのルールにより言語照合順序が変更されます。

ALTERNATE_HANDLING=SHIFTED

スペース、句読点、記号などの変数文字の処理を制御します。このオプションを指定しない(デフォルト値の Non-Ignorable を使用する)場合、これらの変数文字間の違いは、通常の文字間の違いと同様に重要です。

ALTERNATE_HANDLING オプションを指定する場合、これらの変数文字はあまり重要ではありません。

デフォルト NON_IGNOREABLE

ヒント SHIFTED 値は、QUATERNARY に設定された STRENGTH=と組み合わせてよく使用されます。このような場合、スペース文字、句読点、記号が文字列の比較時に考慮されますが、文字列のその他の要素(基本文字、アクセント記号、大文字と小文字)が同一の場合に限定されます。

CASE_FIRST=

大文字と小文字の順序を指定します。この引数は、TERTIARY、QUATERNARY または IDENTICAL レベルでのみ有効です。次の表に、CASE_FIRST 引数の値と情報を示します。

値	説明
UPPER	大文字、小文字の順序で並べ替えます。
LOWER	小文字、大文字の順序で並べ替えます。

COLLATION=

利用可能な COLLATION=値を次の表に示します。照合値を選択しない場合、ユーザーのロケールのデフォルト照合が選択されます。

値	説明
BIG5HAN	ラテン語にピンインの並べ替えを指定し、中国語、日本語、韓国語の文字に big5 文字セットの並べ替えを指定します。
DIRECT	ヒンディー語のバリエントを指定します。
GB2312HAN	ラテン語にピンインの並べ替えを指定し、中国語、日本語、韓国語の文字に gb2312han 文字セットの並べ替えを指定します。
PHONEBOOK	文字の並べ替えに電話帳スタイルを指定します。 PHONEBOOK はドイツ語でのみ選択します。
PINYIN	1 文字ずつピンインに変換する方法を使用した中国語、日本語、韓国語の文字の並べ替えを指定します。この並べ替えは、簡体字中国語でよく使用されます。
POSIX	ポータブルオペレーティングシステムインターフェイスです。このオプションでは、文字の"C"ロケール並べ替えを指定します。
STROKE	英字以外の書体の並べ替えを指定します。中国語、日本語、韓国語、ベトナム語で STROKE を選択します。この並べ替えは、繁体字中国語でよく使用されます。
TRADITIONAL	文字の並べ替えに従来のスタイルを指定します。たとえば、スペイン語で TRADITIONAL を選択します。

LOCALE=locale_name

POSIX 名形式でロケール名を指定します。たとえば、ja_JP です。PROC SORT でサポートされるロケールと POSIX 値の一覧については、[表 21.1](#) を参照してください。

制限事項 次のロケールは PROC SORT でサポートされていません。

- Afrikaans_SouthAfrica, af_ZA
- Cornish_UnitedKingdom, kw_GB

NUMERIC_COLLATION=

数を示す文字のかわりに、数値でテキスト内の整数値を並べ替えます。

値	説明
ON	数値で番号を並べ替えます。たとえば、 "8 Main St."は"45 Main St."より前になります。
OFF	文字の値で数を並べ替えます。たとえば、 "45 Main St."は"8 Main St."より前になります。

デフォルト OFF

STRENGTH=

強度の値は、照合レベルに関連付けられます。5つの照合レベル値があります。次の表に、5つのレベルの情報を示します。強度のデフォルト値は、ロケールに関連付けられます。

値	照合の種類	説明
PRIMARY または 1	PRIMARY では、基本文字間の差異を指定します("a" < "b"など)。	この差異は最強です。たとえば、辞書は基本文字により異なるセクションに分割されます。
SECONDARY または 2	文字のアクセント記号が第 2 差異となります("as" < "às" < "at"など)。	文字列内に第 1 差異がある場合は、第 2 差異は無視されます。文字間のその他の差異も、言語に応じて第 2 差異と考慮される場合があります。
TERTIARY または 3	大文字と小文字の差異は、第 3 レベルで区別されます("ao" < "Ao" < "aò"など)。	文字列内に第 1 または第 2 差異がある場合は、第 3 差異は無視されます。別の例として、通常の仮名と小文字の仮名の違いがあります。
QUATERNARY または 4	レベル 1 から 3 で句読点が無視される場合、句読点の有無で単語を区別するために追加レベルが使用されます("ab" < "a-b" < "aB"など)。	句読点の無視が必須の場合や日本語テキストを処理する場合は、第 4 レベルを使用する必要があります。第 1、第 2 または第 3 差異がある場合は、この差異は無視されます。
IDENTICAL または 5	他のすべてのレベルが等しい場合、最終的な決定をするために同一レベルが使用されます。各	2つの文字列間でコードポイント値のみが異なることはほとんどないため、このレベル

値	照合の種類	説明
	文字列の正規化形式 D (NFD) の Unicode コードポイント値がこのレベルで比較され、レベル 1 から 4 で違いがないことが確認されます。	は必要最小限に使用します。たとえば、ヘブライ語の詠唱マークのみがこのレベルで区別されます。

別名 Level=

注意
データの並べ替えにホストの並べ替えユーティリティを使用する場合、**SORTSEQ=**オプションで変換テーブルベースの照合順序を指定すると、**BY** 変数の文字が破損する可能性があります。詳細については、現在の作環境向けのドキュメントの **PROC SORT** を参照してください。

詳細

PROC SORT ステートメントの照合順序オプションは、SAS データセットのオブザベーションを 1 つ以上の文字または数値変数で並べ替えます。

表 17.1 オプション

タスク	オプション
照合順序の指定	
ASCII の指定	ASCII (1002 ページ)
EBCDIC の指定	EBCDIC (1002 ページ)
デンマーク語の指定	DANISH (1002 ページ)
フィンランド語の指定	FINNISH (1003 ページ)
ノルウェー語の指定	NORWEGIAN (1002 ページ)
ポーランド語の指定	POLISH (1003 ページ)
スウェーデン語の指定	SWEDISH (1003 ページ)
カスタマイズされた順序の指定	NATIONAL (1003 ページ)
前述の照合順序(ASCII、EBCDIC、DANISH、FINNISH、ITALIAN、	SORTSEQ= (1003 ページ)

タスク	オプション
	NORWEGIAN、POLISH、SPANISH、SWEDISH、NATIONAL)、システムで提供されたその他の変換テーブル名(POLISH、SPANISH)、ユーザーが作成した変換テーブル名のいずれかを指定します。エンコーディングを指定できます。また、ロケールに適切な照合順序にするために、キーワード LINGUISTIC または UCA を指定できます。

関連項目

- [“照合順序” \(17 ページ\)](#)
- [“SORT Procedure” \(Base SAS Procedures Guide\)](#)

システムオプション:

- [“SORTSEQ= システムオプション” \(984 ページ\)](#)
- [“TRANTAB= システムオプション” \(988 ページ\)](#)

CORRECTENCODING=オプション

SAS ファイルのエンコーディング属性を、SAS ファイル内のデータのエンコーディングに一致させるように明示的に変更します。

該当要素: DATASETS プロシジャの MODIFY ステートメント

構文

MODIFY SAS file </>CORRECTENCODING=*encoding-value*> >;

オプション引数

</ <CORRECTENCODING=*encoding-value*> >

ファイル内のデータの実際のエンコーディングに一致させるため、ファイルのディスクリプター情報に記録されるエンコーディングインジケータの変更を可能にします。各 SAS ファイル名の後のこのオプションには、かっこは使用できません。フォワードスラッシュの後に CORRECTENCODING=を指定する必要があります。例:

```
modify mydata / correctencoding=latin2;
```

トランスコーディングの有効なエンコーディング値のリストについては、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ)を参照してください。

制限事項 CORRECTENCODING=は、SAS ファイルがデフォルトのベースエンジンである SAS 9 の V9 を使用する場合にのみ、使用できます。

例: SAS セッションエンコーディングと SAS ファイルエンコーディングを解決するための CORRECTENCODING=オプションの使用

ファイルのエンコーディングインジケータは、データのエンコーディングとは別にできます。たとえば、SAS 9 より前のバージョンで作成された SAS ファイルには、エンコーディングインジケータは保存されていません。このようなエンコーディングが記録されていない SAS ファイルを SAS 9 のセッションで開くと、現在のセッションのエンコーディングが割り当てられます。たとえば、データのエンコーディングがデンマーク語 EBCDIC で現在のセッションのエンコーディングが Western Wlatin1 の場合、ファイル内のデータの実際のエンコーディングとファイルのディスクリプター情報に保存されたエンコーディングインジケータは一致しません。この場合、データは正常にトランスコードされず、出力が読み込めなくなる可能性があります。次の MODIFY ステートメントを使用し、EBCDIC エンコーディングを明示的に割り当てることでこの問題を解決できます。

注: CEDA では読み取り専用のコピーが作成されます。データを永続的にトランスコードするには、PROC COPY または DATA ステップでデータをコピーする必要があります。

```
proc datasets library=myfiles;
  modify olddata / correctencoding=ebcdic1142;
quit;
```

CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE=オプション

SAS ファイルをトランスコードするために必要とされる文字変数の属性を指定します。

該当要素: LIBNAME ステートメント

カテゴリ: データアクセス

PROC OPTIONS LINUX での LIBNAME ステートメント。
GROUP=

	CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、 CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE=オプション 1011
制限事項:	CVPEXCLUDE=および CVPINCLUDE=オプションは、SAS Viya プラットフォーム 3.5 以降で使用できます。
参照項目:	LIBNAME、SAS/ACCESS

構文

```
LIBNAME libref 'SAS data-library' <CVPBYTES=bytes> <CVPENGINE=engine>
<CVPEXCLUDE=variable expression><CVPFORMATWIDTH=YES|
NO><CVPINCLUDE=variable expression><CVPMULTIPLIER=multiplier>
<CVPVARCHAR=YES | NO>;
```

オプション引数

CVPBYTES=bytes

トランスコーディングを必要とする SAS データファイル进行处理するときに、拡張する文字変数の長さをバイト数で指定します。CVP エンジン、長さを拡張することにより、文字データの切り捨てが発生しないようにします。文字変数の長さは、指定した値を現在の長さに追加して増やします。0 から 32766 までの値を指定できます。

たとえば、次の LIBNAME ステートメントでは、CVPBYTES=オプションを指定して CVP エンジンを暗黙的に割り当てています。

```
libname expand 'SAS data-library' cvpbytes=5;
```

文字変数の長さは、5 バイト追加して増やします。長さが 10 の文字変数は長さ 15 に、長さが 100 の文字変数は長さ 105 に増加します。

デフォルト CVPBYTES=を指定すると、指定値に従って文字変数の長さを拡張するために CVP エンジンを自動的に使用します。CVP エンジンを明示的に割り当てても、CVPBYTES=または CVPMULTIPLIER=を指定しない場合は、SAS は乗数(CVPMULTIPLIER=0)を自動的に選択して、文字変数を長くします。

制限事項 CVP エンジン、SAS データファイルをサポートしていますが、SAS のビュー、カタログおよびアイテムストアなどはサポートしていません。

CVP エンジン、入力(読み込み)処理でのみ使用できます。

CVP エンジンなどのエンジンが混在するライブラリ連結では、SAS データファイルのみが処理されます。たとえば、COPY プロシジャを実行する場合は、SAS データファイルのみコピーされます。

要件 指定するバイト数は、どのような拡張にも適応できる大きさにする必要があります。この大きさが不十分だと、データが切り捨てられ、その結果 SAS ログにエラーメッセージが書き込まれます。

操作 CVPBYTES=オプションと CVPMULTIPLIER=オプションの両方を指定することはできません。いずれか 1 つのオプションのみを指定してください。

参照項目: ["CVP エンジンの使用による文字データ切り捨ての回避" \(44 ページ\)](#)

CVPENGINE=engine

トランスコーディングを必要とする SAS データファイル进行处理するために使用するエンジンを指定します。トランスコードする文字変数の長さが CVP エンジンによって拡張されるため、文字データの切り捨ては発生しません。次に、指定されたエンジンで実際のファイル进行处理します。

別名 CVPENG

デフォルト SAS は、デフォルトの SAS エンジンを使用します。

参照項目: [“CVP エンジンの使用による文字データ切り捨ての回避” \(44 ページ\)](#)

CVPEXCLUDE=variable expression

データ処理中に除外される変数を指定します。

この例では、文字変数 var1、var2、および var3 がデータ処理から除外されています。cvpexclude="^(var1|var2|var3)\$"

CVPINCLUDE=と CVPEXCLUDE=は一緒に使用することはできません。両方のオプションを省略すると、すべての変数が処理されます。

CVPEXCLUDE=var が指定されている場合、var を含むすべての変数(var、var1、invariant など)は除外されます。var 変数のみを除外するには、CVPEXCLUDE="^var\$"を指定します。

CVPFORMATWIDTH=YES | NO

文字出力形式の幅を拡張するかどうかを指定します。

CVPVARCHAR=を指定しないと、新しい出力形式の幅は CVPMULTIPLIER=および CVPBYTES=オプションによって決まります。

CVPVARCHAR=を指定すると、CVP エンジンに変換された文字変数の最大バイト長を満たすように、出力形式の幅を自動的に調整します。たとえば、UTF-8 セッションでは、出力形式の幅に 4 を掛けます。

別名 CVPFMTW

デフォルト YES

CVPINCLUDE=variable expression

データ処理中に含まれる変数を指定します。指定された変数のみが処理されます。

CVPINCLUDE=と CVPEXCLUDE=は一緒に使用することはできません。両方のオプションを省略すると、すべての変数が処理されます。

注: CVPEXCLUDE=および CVPINCLUDE=オプションは、SAS Viya プラットフォーム 3.5 以降で使用できます。

CVPMULTIPLIER=multiplier

トランスコーディングを必要とする SAS データファイル进行处理するときに、文字変数の長さを拡張するための乗数値を指定します。CVP エンジンは、長さを拡張することにより、文字データの切り捨てが発生しないようにします。文字変数の長さは、指定した値を現在の長さに乗算して増やします。1 から 5 までの乗数を指定することも、0 を指定して CVP エンジンが乗数を自動的に決定するようにすることもできます。

CVPBYTES=、CVPENGINE=、CVPFORMATWIDTH=、CVPMULTIPLIER=、
CVPVARCHAR=、CVPEXCLUDE=、CVPINCLUDE=オプション [1013](#)

たとえば、次の LIBNAME ステートメントでは、CVPMULTIPLIER=オプションを指定して CVP エンジンを暗黙的に割り当てています。

```
libname expand 'SAS data-library' cvpmultiplier=2.5;
```

文字変数の長さは 2.5 倍になります。長さが 10 の文字変数は長さ 25 に、長さが 100 の文字変数は長さ 250 に増加します。

別名 CVPMULT

デフォルト CVPMULTIPLIER=オプションを指定すると、指定値に従って文字変数の長さを拡張するために CVP エンジンを自動的に使用します。CVP エンジンを明示的に割り当てても、CVPMULTIPLIER=オプションまたは CVPBYTES=オプションを指定しない場合は、CVPMULTIPLIER=0(AUTO)を使用して文字変数を長くします。AUTO(0)は、SAS セッションと入力データセットのエンコーディングに基づいて CVP エンジンの値を設定します。

制限事項 CVP エンジンは、SAS データファイルをサポートしていますが、SAS のビュー、カタログおよびアイテムストアなどはサポートしていません。

CVP エンジンは、入力(読み込み)処理でのみ使用できます。

CVP エンジンなどのエンジンが混在するライブラリ連結では、SAS データファイルのみが処理されます。たとえば、COPY プロシジャを実行する場合は、SAS データファイルのみコピーされます。

要件 指定するバイト数は、どのような拡張にも適応できる大きさにする必要があります。この大きさが不十分だと、データが切り捨てられ、その結果 SAS ログにエラーが書き込まれます。

操作 CVPMULTIPLIER=オプションと CVPBYTES=オプションの両方を指定することはできません。いずれか 1 つのオプションのみを指定してください。

参照 [“CVP エンジンの使用による文字データ切り捨ての回避” \(44 ページ\)](#)
項目:

CVPVARCHAR=YES | NO

入力ファイル処理の際に、固定幅文字変数を可変幅文字に変換するか否かを指定します。固定幅文字変数のバイト長は、SAS セッションエンコーディングにおける 1 文字あたり最大バイト数に、指定された固定幅文字長を掛けた値です。

デフォルト No

操作 CVPVARCHAR=YES を指定した場合、CVPMULTIPLIER=オプションや CVPBYTES=オプションを指定しても無視されます。

注 CHAR カラム以下にある文字列データの末尾に空白があれば削除します。

TRANSCODE=NO という形式の固定幅文字変数は、変換の対象外です。

詳細

CVPVARCHAR=YES である場合、文字変数パディング(CVP、Character Variable Padding)エンジンは、CHAR データ型で定義された変数を VARCHAR データ型に変換します。CVP は入力処理専用なので、自動的に VARCHAR データ型で保存されることはありません。

SAS セッションの終了時、データは保存されず、VARCHAR への変換結果も失われてしまいます。変更を保存したければ、SET ステートメントまたは PROC COPY と、VARCHAR に対応したエンジンを使ってください。BASE エンジンなど、VARCHAR に非対応のエンジンでデータを保存した場合、新しいデータセットの文字カラムは CHAR データ型に戻ってしまいます。新しいデータセットの CHAR 変数の長さは、VARCHAR の保存に必要なバイト数と同じです。

VARCHAR の詳細については、“[文字列関数の VARCHAR データ型](#)” (579 ページ)を参照してください。

データを読み込むプロシジャが VARCHAR に対応していなければ、CVPVARCHAR=YES と指定しても無視されます。CVP エンジンは、CVPBYTES、CVPMULTIPLIER、またはデフォルトの乗数を使って、データ内の文字カラムの長さを拡張します。

デフォルトで、CVP エンジンは変数の長さの 1.5 倍の拡張を使用します。西ヨーロッパ言語のテキストは通常、各国語に対する ASCII 文字の比率が高いため、この拡張は通常、西ヨーロッパ言語のデータをサポートするのに十分です。1.5 倍の拡張は、ほとんどのアジア言語データにも十分な可能性があります。多くのアジア文字は UTF-8 で 3 バイトを必要としますが、これらの言語をサポートする 2 バイト (DBCS)エンコーディングでは、アジア文字ごとに 2 バイトがすでに必要です。

ただし、1 バイト(SBCS)エンコーディングで表される多くの言語では、テキストで各国語の比率が高く、UTF-8 では 2 バイトまたは 3 バイトが必要です。たとえば、ロシア語のテキストは主にキリル文字の文字を使用します。これには、WCYRILLIC(CP1251 および WINDOWS-1251)をエンコードする SBCS で 1 バイトが必要です。UTF-8 では各キリル文字に 2 バイトが必要なため、変数の長さによっては、1.5 倍の拡張では切り捨てを防ぐのに十分でない場合があります。

拡張を変更する必要がある場合は、[CVPMULTIPLIER](#)=または、LIBNAME ステートメントの CVPBYTES=オプションのいずれかを指定できます。CVPMULTIPLIER=は、変数の長さに適用される乗数値を指定します。[CVPBYTES](#)=は、文字列の長さに特定のバイト数を追加します。

例を次に示します。

```
libname mylib 'path-to-SAS-files' cvpmultiplier=2.5;
libname mylib 'path-to-SAS-files' cvpbytes=5;
```

また、“[NCHARMULTIPLIER= LIBNAME ステートメントオプション](#)” ([SAS Cloud Analytic Services: ユーザーガイド](#))を使用して乗数を指定し、CAS サーバーで実行することもできます。

これらのデータテーブルは、データを UTF-8 として保存するときに追加のストレージ要件を決定するのに役立ちます。

[表 17.186](#) は、指定されたエンコーディングから UTF-8 にデータを移動するときに予想されるストレージサイズの増加を示しています。

表 17.2 UTF-8 としてのデータの保存と可能なストレージサイズの増加

ソースエンコーディング	サポートされている言語	UTF-8 でのストレージサイズの増加
ASCII	英語、マレー語、およびその他のサポートされている言語	0%
ISO-8859-1 (LATIN1)、および Windows cp1252 (WLATIN1)	西ヨーロッパ	10%
ISO-8859-7、プレーンテキスト	ギリシャ語	90%
ISO-8859-7、50%マークアップ	ギリシャ語	45%
TIS-620、プレーンテキスト	タイ語	190%
TIS-620、50%マークアップ	タイ語	95%
EUC-KR、プレーンテキスト	韓国語	50%
EUC-KR、50%マークアップ	韓国語	55%

表 17.187 に、リスト内の言語の文字を表すために必要なバイト数を示します。

表 17.3 言語別の UTF-8 文字の長さ

文字の長さ	言語
1 バイト	US_ASCII 文字
2 バイト	東および西ヨーロッパ語、バルト語、ギリシャ語、トルコ語、キリル語、ヘブライ語、アラビア語、およびその他のサポートされている文字セット
3 バイト	中国語、日本語、韓国語(CJK)、タイ語、インド語、および特定の制御文字
4 バイト	絵文字、あまり一般的ではない CJK 文字、およびさまざまな歴史的なスクリプト

例: CVP エンジンの使い方

次の例に、CVP エンジンを使用して文字データの切り捨てを回避する方法を示します。例では、Wlatin2 エンコーディングの一部の各国語文字を含む MYFILES.WLATIN2 という名前の SAS データセットを使用します。MYFILES.WLATIN2 は、wlatin2 エンコーディングを使用して SAS セッションで作成する必要があります。

```
libname myfiles 'SAS Data-Library';
data myfiles.wlatin2 (encoding=wlatin2);
  var1='A';
  var2='Š';
  var3='š';
  var4='ı';
;
proc print data=myfiles.wlatin2;
run;
```

Obs	var1	var2	var3	var4
1	A	Š	š	ı

これは、SAS 9 で実行した、MYFILES.WLATIN2 での PROC CONTENTS 出力です。エンコーディングが Wlatin2 で、各文字変数の長さが 1 バイトであることを示しています。

アウトプット 17.1 MYFILES.WLATIN2 の PROC CONTENTS 出力

Data Set Name	MY FILES.WLATIN2	Observations	1
Member Type	DATA	Variables	4
Engine	V9	Indexes	0
Created	07/28/2017 12:02:40	Observation Length	4
Last Modified	07/28/2017 12:02:40	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	wlatin2 Central Europe (Windows)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	15881
Obs in First Data Page	1
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\SAS Data-Library\wlatin2.sas7bdat
Release Created	9.0401M5
Host Created	X64_7PRO
Owner Name	BUILTIN\Administrators
File Size	128KB
File Size (bytes)	131072

Alphabetic List of Variables and Attributes			
#	Variable	Type	Len
1	var1	Char	1
2	var2	Char	1
3	var3	Char	1
4	var4	Char	1

次のコードがセッションエンコーディング Wlatin2 を使用して実行されます。

```
options msglevel=i;
libname myfiles 'SAS Data-Library';
data myfiles.utf8 (encoding="utf-8");
  set myfiles.wlatin2;
run;
```

この DATA ステップでは、新しいデータセット MYFILES.UTF8 を要求します。さらに、データを UTF-8 エンコーディングのこの新しいデータセットに読み込む(つまり、Wlatin2 から UTF-8 へのデータのトランスコードが必要)ことを要求します。トランスコードの際に文字データの切り捨てが発生した場合、要求結果はエラーとなります。新規データセット MYFILES.UTF8 が作成されますが、データは何も含まれていません。

例のコード 17.1 トランスコーディングエラーの SAS ログ

```
INFO: Data file MYFILES.UTF8.DATA is in a format native to another
host, or the file encoding does not match the session encoding.
Cross Environment Data Access will be used, which might require additional
CPU resources and might reduce performance.
ERROR: Some character data was lost during transcoding in the dataset
MYFILES.UTF8. Either the data contains characters that are not
representable in the new encoding or truncation
occurred during transcoding.
NOTE: The data step has been abnormally terminated.
NOTE: The SAS System stopped processing this step because of errors.
NOTE: There were 1 observations read from the data set MYFILES.WLATIN2.
WARNING: The data set MYFILES.UTF8 may be incomplete. Either the data contains characters that are not
representable in the new encoding or truncation
occurred during transcoding.
```

次のコードが実行され、変数の長さが拡張されます。

```
options msglevel=i;
libname myfiles 'SAS Data-Library';
libname expand cvp 'SAS Data-Library' cvpbytes=2;
data myfiles.utf8 (encoding="utf-8");
  set expand.wlatin2;
run;
```

この例で CVP エンジンでは、それぞれの長さに 2 バイトを追加して文字変数の長さを拡張します。データは、Wlatin2 から UTF-8 にトランスコードされることにより、UTF-8 エンコーディングで新規ファイルに読み込まれます。拡張された文字変数の長さが原因のデータの切り捨ては発生せず、新規データセットが正常に作成されます。

例のコード 17.2 MYFILES.UTF8 の PROC ログ出力

```
57 options msglevel=i;
58 libname myfiles 'C:\SAS Data-Library';

NOTE: Libref MYFILES was successfully assigned as follows:
      Engine:      V9
      Physical Name: C:\SAS Data-Library
59 libname expand cvp 'C:\SAS Data-Library' cvpbytes=2;
WARNING: Libref EXPAND refers to the same physical library as MYFILES.
NOTE: Libref EXPAND was successfully assigned as follows:
      Engine:      CVP
      Physical Name: 'C:\SAS Data-Library'
60 data myfiles.utf8 (encoding="utf-8");;
61 set expand.wlatin2;
INFO: Data file EXPAND.WLATIN2.DATA is in a format that is native to another host, or the file encoding
does not match the session
encoding.Cross Environment Data Access will be used, which might require additional CPU resources and
might reduce performance.
62 run;

NOTE: There were 1 observations read from the data set EXPAND.WLATIN2.
NOTE: The data set MYFILES.UTF8 has 1 observations and 4 variables.
```

最後に示すのは MYFILES.UTF8 の PROC CONTENTS 出力です。これは、UTF-8 エンコーディングで表示され、各文字変数の長さは 3 です。

アウトプット 17.2 MYFILES.UTF8 の PROC CONTENTS 出力

Data Set Name	MYFILES.UTF8	Observations	1
Member Type	DATA	Variables	4
Engine	V9	Indexes	0
Created	07/28/2017 12:02:41	Observation Length	12
Last Modified	07/28/2017 12:02:41	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_64		
Encoding	utf-8 Unicode (UTF-8)		

Engine/Host Dependent Information	
Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	5403
Obs in First Data Page	1
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\SAS Data-Library\utf8.sas7bdat
Release Created	9.0401M5
Host Created	X64_7PRO
Owner Name	BUILTIN\Administrators
File Size	128KB
File Size (bytes)	131072

Alphabetic List of Variables and Attributes			
#	Variable	Type	Len
1	var1	Char	3
2	var2	Char	3
3	var3	Char	3
4	var4	Char	3

ENCODING=オプション

外部ファイルの入力または出力処理のエンコーディングを優先指定してトランスコードします。

該当要素: %INCLUDE ステートメント、FILE ステートメント、FILENAME ステートメント、FILENAME ステートメントの EMAIL (SMTP) アクセス方式、INFILE ステートメント、ODS ステートメント、FILE コマンド、INCLUDE コマンド

カテゴリ: データアクセス

構文

ENCODING= 'encoding-value'

オプション引数

ENCODING= 'encoding-value'

外部ファイルの読み込み、書き込み、コピー、保存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。

外部ファイルを使用してデータの読み込み、書き込み、コピー、保存を行うと、SAS はデータを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します。

詳細については、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ) を参照してください。

デフォルト SAS は、現在のセッションエンコーディングを使用します。

詳細

次のテーブルでは、ENCODING オプションが対応するステートメントでどのように使われるかについて説明します。

%INCLUDE ステートメント:	SAS ステートメントおよびデータ行の指定したソースファイルからの読み込み。
-------------------	--

FILE ステートメント:	外部ファイルへの書き込み
---------------	--------------

FILENAME ステートメント:	外部ファイルからの読み込み、外部ファイルへの書き込み
-------------------	----------------------------

FILENAME ステートメント の EMAIL (SMTP) アクセス 方式:	プログラムによる電子メール送信
INFILE ステートメント:	外部ファイルからの読み込み
ODS ステートメント:	SAS プロシジャおよび DATA ステップの出力の生成、保存または再作成に使用する Output Delivery System (ODS)機能の制御
FILE コマンド:	ウィンドウのコンテンツの外部ファイルへの保存
INCLUDE コマンド:	外部ファイルの現在のウィンドウへのコピー

一部のエンコーディングでは、バイトオーダーマーク(BOM)を使用します。BOM はエンコーディングの指定時に生成されます。UTF-8 エンコーディングの場合、BOM を生成するには、ファイル名とファイルの DATA ステップステートメントに *encoding=utf-8* を指定する必要があります。

例

例 1: FILE ステートメントを使用した外部ファイルへの書き込みのエンコーディングの指定

この例では、SAS データセットから外部ファイルを作成します。現在のセッションエンコーディングは Wlatin1 ですが、外部ファイルのエンコーディングは UTF-8 にする必要があります。デフォルトでは、外部ファイルは現在のセッションエンコーディングを使用して書き込まれます。

外部ファイルへのデータの書き込みに使用するエンコーディングを指定するには、ENCODING=オプションを次のように指定します。

```
libname myfiles 'SAS data-library';
filename outfile 'external-file';
data _null_;
  set myfiles.cars;
  file outfile encoding="utf-8";
  put Make Model Year;
run;
```

外部ファイルを UTF-8 エンコーディングにするように指示すると、Wlatin1 から指定した UTF-8 エンコーディングにデータがトランスコードされます。

例 2: FILENAME ステートメントを使用した外部ファイルの読み込みのエンコーディングの指定

この例では、外部ファイルから SAS データセットを作成します。外部ファイルは UTF-8 文字セットエンコーディングで、現在の SAS セッションは Wlatin1 エンコーディングです。デフォルトでは、SAS は外部ファイルがセッションエンコーディン

グと同じエンコーディングであると想定します。この場合、文字データは新しい SAS データセットに正しく書き出されません。

外部ファイルの読み込み時に使用するエンコーディングを指定するには、ENCODING=オプションを次のように指定します。

```
libname myfiles 'SAS data-library';

filename extfile 'external-file' encoding="utf-8";
data myfiles.unicode;
  infile extfile;
  input Make $ Model $ Year;
run;
```

外部ファイルが UTF-8 エンコーディングであると指定すると、SAS は、新しい SAS データセットへの書き込み時に外部ファイルを UTF-8 から現在のセッションエンコーディングにトランスコードします。これで、新しいデータセットにデータが Wlatin1 のエンコーディングで正しく書き込まれるようになります。

例 3: FILENAME ステートメントを使用した外部ファイルへの書き込みのエンコーディングの指定

この例では、SAS データセットから外部ファイルを作成します。デフォルトで、SAS は外部ファイルを現在のセッションエンコーディングを使用して書き込みます。現在のセッションエンコーディングは Wlatin1 ですが、外部ファイルのエンコーディングは UTF-8 にする必要があります。

外部ファイルへのデータ書き込み時に使用するエンコーディングを指定するには、ENCODING=オプションを次のように指定します。

```
libname myfiles 'SAS data-library';
filename outfile 'external-file' encoding="utf-8";
data _null_;
  set myfiles.cars;
  file outfile;
  put Make Model Year;
run;
```

外部ファイルを UTF-8 エンコーディングにするように指定すると、外部ファイルへの書き込み時に、Wlatin1 から指定した UTF-8 エンコーディングにデータがトランスコードされます。

例 4: メッセージ本文と添付ファイルのエンコーディングの変更

この例では、メッセージ本文と添付ファイルのテキストエンコーディングを変更する方法を説明します。

```
filename mymail email 'Joe.Developer@sas.com';
data _null_;
  file mymail
    subject='Text Encoding'
    encoding=greek
    attach=('C:\My Files\Test.out'
      content_type='text/plain'
      encoding='ebcdic1047'
      outencoding='latin1');
run;
```

プログラムでは、次の操作が実行されます。

- ENCODING=メールオプションで、メッセージ本文を送信前にギリシャ語(ISO)でエンコードするように指定します。
- ATTACH=メールオプションの場合、添付ファイルオプション ENCODING=で、SAS に読み込まれる添付ファイルのエンコーディングを西ヨーロッパ諸語 (EBCDIC)に指定します。
- SMTP やその他のメールインターフェイスは EBCDIC をサポートしないため、添付ファイルオプション OUTENCODING=によって、添付ファイルは送信前に西ヨーロッパ諸語(ISO)に変換されます。

例 5: INFILE=ステートメントを使用した外部ファイル読み込みのエンコーディングの指定

この例では、外部ファイルから SAS データセットを作成します。外部ファイルのエンコーディングは UTF-8 であり、現在の SAS セッションのエンコーディングは Wlatin1 です。デフォルトでは、SAS は、外部ファイルのエンコーディングがセッションエンコーディングと同じであると見なします。このため、文字データは新しい SAS データセットに正しく書き込まれません。

外部ファイルの読み込み時に使用するエンコーディングを指定するには、ENCODING=オプションを次のように指定します。

```
libname myfiles 'SAS data-library';
filename extfile 'external-file';
data myfiles.unicode;
  infile extfile encoding="utf-8";
  input Make $ Model $ Year;
run;
```

外部ファイルが UTF-8 エンコーディングであると指定すると、SAS は、新しい SAS データセットへの書き込み時に外部ファイルを UTF-8 から現在のセッションエンコーディングにトランスコードします。これで、新しいデータセットにデータが Wlatin1 のエンコーディングで正しく書き込まれるようになります。

関連項目

ステートメント:

- [“FILE ステートメント” \(SAS DATA ステップステートメント: リファレンス\)](#)
- [“INFILE ステートメント” \(SAS DATA ステップステートメント: リファレンス\)](#)

INENCODING=

SAS ライブラリの SAS データセットの読み込みまたは書き込み時のエンコーディングを無効にして変更します。

該当要素: LIBNAME ステートメント

カテゴリ: データアクセス

構文

INENCODING= ANY | ASCIIANY | EBCDICANY | *encoding-value*

OUTENCODING= ANY | ASCIIANY | EBCDICANY | *encoding-value*

構文の説明

ANY

ASCII エンコーディングと EBCDIC エンコーディング間でトランスコーディングを行わないように指定します。

NOTE: ANY は BINARY の指定と同等です。データはバイナリであるため、実際のエンコーディングには意味がありません。

ASCIIANY

混在したエンコーディングが ASCII エンコーディングと想定して、トランスコーディングを行わないように指定します。

EBCDICANY

混在したエンコーディングが EBCDIC エンコーディングと想定して、トランスコーディングを行わないように指定します。

encoding-value

エンコーディング値を指定します。エンコーディング値の一覧については、[23 章, “SAS 言語要素中のエンコーディング値” \(1117 ページ\)](#)を参照してください。

詳細

INENCODING=オプションは、SAS ライブラリの SAS データセットの読み込みに使用されます。OUTENCODING=オプションは、SAS ライブラリの SAS データセットの書き込みに使用されます。

INENCODING=値または OUTENCODING=値は、LIST 引数を使用する場合、SAS ログに書き出されます。

INENCODING=および OUTENCODING=は、混在したエンコーディングが含まれる既存のライブラリを使用する場合に、最も適しています。混在したエンコーディングが含まれるライブラリを読み取るには、INENCODING=を ASCIIANY または EBCDICANY に設定できます。別個のデータセットを書き出すには、OUTENCODING=を使用して特定のエンコーディングを指定できます。このエンコーディングがデータセットの作成時に適用されます。

比較

- セッションエンコーディングは、ENCODING=システムオプションを使用して指定します。

- SAS ライブラリのデータセットを読み取るためのエンコーディングを指定するには、LIBNAME ステートメントの INENCODING=オプションを入力ファイルに使用します。LIBNAME ステートメントオプションと ENCODING=データセットオプションを両方とも指定すると、データセットオプションが使用されます。
- SAS ライブラリにデータセットを書き出すためのエンコーディングを指定するには、LIBNAME ステートメントの OUTENCODING=オプションを出力ファイルに使用します。LIBNAME ステートメントオプションと ENCODING=データセットオプションを両方とも指定すると、データセットオプションが使用されます。
- COPY プロシジャの場合、デフォルトの CLONE オプションでは、OUTENCODING=オプションで指定したエンコーディング値ではなく、入力データセットのエンコーディング属性を使用します。CLONE および NOCLONE の詳細については、[COPY ステートメント](#)を参照してください。

注: この操作は、SAS/CONNECT または SAS/SHARE 使用時には適用されません。

関連項目

- [“概要: 各国語サポート関連のエンコーディング” \(12 ページ\)](#)

システムオプション:

- [“ENCODING システムオプション” \(968 ページ\)](#)
- [“LOCALE システムオプション” \(973 ページ\)](#)

データセットオプション:

- [“ENCODING= データセットオプション” \(84 ページ\)](#)

ODSCHARSET=オプション

出力の META 宣言で生成される文字セットを指定します。

該当要素: XML V2 エンジンの LIBNAME ステートメント
 カテゴリ: データアクセス

構文

ODSCHARSET=*character-set*;

必須引数

character-set

XML エンジンの LIBNAME ステートメントでは、ENCODING=属性で使用する文字セットを指定します。

エンコーディングの例は ISO-8859-1 です。インターネットで使用する公式の文字セットは、IANA (Internet Assigned Numbers Authority)によって登録されます。IANA は、ポート、プロトコルとエンタープライズ番号、オプション、コードとタイプなど、さまざまなインターネットプロトコルパラメーターの中央レジストリです。文字セット値の一覧については、www.unicode.org/reports/tr22/index.html および www.iana.org/assignments/character-sets を参照してください。

character set は、このコンテキストの *encoding-value* に似ています。ただし、*character set* という用語は、インターネットでの使用に適したエンコーディングを識別するために使用されます。

詳細

XML ドキュメントによっては、XML 宣言が不要なものもあります。XML 宣言は、ドキュメントの文字エンコーディングがデフォルトの UTF-8 または UTF-16 以外で、上位プロトコルでエンコーディングが決まっていなかった場合にのみ必要となります。

XML エンジンの LIBNAME ステートメントの ODSCHARSET オプションで 出力する XML ドキュメントに使う文字セットを指定します。

関連項目

概念的な情報:

- [3 章, "各国語サポート関連のエンコーディング" \(11 ページ\)](#)

ステートメント:

- [SAS XMLV2 およびXML LIBNAME エンジン: ユーザーガイド](#)

ODSTRANTAB=オプション

出力ファイルの XML ドキュメントをトランスコードするときに使用する変換テーブルを指定します。

該当要素: XML V2 エンジンの LIBNAME ステートメント

カテゴリ: データアクセス

構文

TRANTAB ='translation-table'

オプション引数

translation-table

出力ファイルで使用する変換テーブルを指定します。変換テーブルは、文字セットの文字(通常の文字、表語文字、数字、句読点、記号、コントロール文字など)を数値にマッピングするエンコーディング方法です。変換テーブルの例として、文字を EBCDIC から ASCII-ISO に変換するものがあります。*table-name* には、SAS で提供される変換テーブルまたはユーザー定義の変換テーブルを指定できます。値は、SASUSER.PROFILE カタログまたは SASHELP.HOST カタログの SAS カタログエントリ名である必要があります。

詳細

SAS 9.2 では、下位互換性のために、XML エンジンの LIBNAME ステートメントで ODSTRANTAB=オプションの使用がサポートされています。エンコーディングを指定する場合、LOCALE=システムオプションを使用する方法が推奨されます。

関連項目

概念的な情報:

- [“トランスコーディングと変換テーブル” \(34 ページ\)](#)
- [2 章, “各国語サポート関連のロケール” \(5 ページ\)](#)の概念の説明

システムオプション:

- [“TRANTAB= システムオプション” \(988 ページ\)](#)
- [“LOCALE システムオプション” \(973 ページ\)](#)

プロシジャ:

- [20 章, “TRANTAB プロシジャ,” \(1067 ページ\)](#)

ステートメント:

- [SAS XMLV2 およびXML LIBNAME エンジン: ユーザーガイド](#)

PROC SQL の TRANSCODE=列修飾子

文字列に値をトランスコードできるかどうかを指定します。

該当要素: SQL プロシジャの列修飾子の構成要素

構文

TRANSCODE=YES|NO

必須引数

TRANSCODE=YES | NO

文字列について、値をトランスコードできるかどうかを指定します。トランスコーディングを実行しないようにするには、TRANSCODE=NO を使用します。
CREATE TABLE AS ステートメントを使用してテーブルを作成する場合、作成されるテーブルの特殊文字の列のトランスコーディング属性は、TRANSCODE=列修飾子を使用して変更しない限り、入力元テーブルの属性と同じであることに注意してください。

デフォルト YES

制限事項 トランスコーディングの抑止は、V6TAPE エンジンではサポートされていません。

関連項目

概念的な情報:

- [4 章, “各国語サポート関連のトランスコーディング” \(33 ページ\)](#)

RENCODING=オプション

EBCDICANY または ASCIIANY セッションエンコーディングを使用する SAS/SHARE サーバーセッションのデータをトランスコードする場合に使用する、ASCII ベースまたは EBCDIC ベースのエンコーディングを指定します。

該当要素: SAS/SHARE の LIBNAME ステートメントのみ

カテゴリ: データアクセス

注: LIBNAME ステートメントの RENCODING=オプションは、エンコーディングが混在するコンピューティング環境を維持するためにセッションエンコーディングを EBCDICANY または ASCIIANY に設定した SAS/SHARE Server を使用する場合にはのみ関係します。

参照項目: SAS/SHARE User's Guide の LIBNAME ステートメント

構文

RENCODING=ASCII-encoding-value | EBCDIC-encoding-value

構文の説明

ASCII-encoding-value

UNIX および Windows での ASCII エンコーディングの有効な値の一覧については、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ) を参照してください。

EBCDIC-encoding-value

z/OS での EBCDIC エンコーディングの有効な値の一覧については、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値” (1118 ページ) を参照してください。

詳細

エンコーディングが混在する環境(たとえば、Latin1 や Latin2 などの互換性のないエンコーディングを使用する SAS/SHARE クライアントセッション)で SAS/SHARE を使用する場合は、次のオプションを設定できます。

- SAS/SHARE サーバーセッションでは、SAS システムオプション **ENCODING=EBCDICANY** または **ENCODING=ASCIIANY** を設定します。
- SAS/SHARE クライアントセッションでは、次の条件において、LIBNAME ステートメントで **RENCODING=オプション** を設定します。
 - ASCII ベースのエンコーディングを使用するクライアントセッションが EBCDICANY サーバーにアクセスする場合
 - EBCDIC ベースのエンコーディングを使用するクライアントセッションが ASCIIANY サーバーにアクセスする場合

RENCODING=オプションにより、SAS/SHARE Client は、クライアント/サーバーセッションのトランスコーディング時に想定するサーバーのデータのエンコーディングを指定できます。

SAS 9 では、SAS/SHARE クライアント/サーバーセッションで複数の SBCS または DBCS エンコーディングのデータを処理する場合は、UTF8 エンコーディングの使用をお勧めします。UTF8 セッションエンコーディングを実行する Unicode サーバーの詳細については、<http://support.sas.com/rnd/migration/nls/index.html> にアクセスし、*Processing Multilingual Data with the SAS® 9.2 Unicode Server* および *Multilingual Computing with SAS® 9.4* を検索してください。

比較

SAS 9 では、UTF8 エンコーディングを使用して、従来の SBCS または DBCS エンコーディングの文字を複数含む多言語データを SAS データセットで保持できます。SAS/SHARE を使用してそのデータへの更新アクセスを共有するには、UTF8 のセッションエンコーディングを使用して SAS/SHARE Server を実行する必要もありま

す。SAS では、必要に応じてデータをクライアントのエンコーディングにトランスコードします。

SAS 9 以前は、SAS/SHAR Client と SAS/SHARE Server が共通のアーキテクチャで実行される(たとえば、クライアントとサーバーが UNIX マシン上で実行される)場合、文字データの自動トランスコーディングは実行されませんでした。1 つの SAS/SHARE Server 内に含まれる EBCDIC または ASCII エンコーディングが異なるデータセットにアクセスするアプリケーションや、1 つのデータセット内にさまざまなエンコーディングが混在するデータセットにアクセスするアプリケーションを作成できました。この方法はあまり一般的ではなく、異なる動作環境で実行するクライアントからトランスコーディングテーブルを 慎重にプログラミングする必要があります。

次のステップは、必要に応じて SAS 9 で混在するエンコーディングを保持する方法を示しています。

- SAS/SHARE Server は、EBCDIC エンコーディングが混在する場合は EBCDICANY のセッションエンコーディングを、ASCII エンコーディングが混在する場合は ASCIIANY のセッションエンコーディングを使用して実行する必要があります。

これにより、バージョン 8 以前のリリースの動作を復元し、同じ EBCDIC または ASCII ファミリでの異なるクライアントエンコーディングとサーバーエンコーディングとの間の自動的な文字トランスコーディングを回避します。つまり、次の条件下ではトランスコーディングが行われません。

- クライアントセッションエンコーディングが EBCDIC エンコーディングで、サーバーセッションエンコーディングが EBCDICANY の場合
- クライアントセッションエンコーディングが ASCII エンコーディングで、サーバーセッションエンコーディングが ASCIIANY の場合

- ASCIIANY または EBCDICANY サーバーと同じエンコーディングファミリを共有しない SAS/SHARE Client は、そのサーバーにアクセスする最初の LIBNAME ステートメントで RENCODING=オプションを使用して、必要なトランスコーディングを制御できます。

たとえば、ポーランド語ロケールで実行される ASCII クライアントは、z/OS EBCDICANY サーバーにアクセスし、RENCODING=EBCDIC870 と指定して、ポーランド語でエンコードされたデータを含むことをクライアントが認識するデータにアクセスできます。ドイツ語ロケールで実行される別の ASCII クライアントは、同じ z/OS EBCDICANY サーバーにアクセスし、RENCODING=EBCDIC1141 と指定して、ドイツ語のデータを含むことをクライアントが認識しているデータにアクセスできます。同様に、ASCIIANY サーバーにアクセスする EBCDIC クライアントは、LIBNAME ステートメントの RENCODING=オプションを使用して、アクセスしようとするデータの正確な ASCII エンコーディングを指定できます。

関連項目

概念的な情報:

- [“トランスコーディングの概要” \(33 ページ\)](#)

TRANSCODE=オプション

文字変数をトランスコードするかどうかを指示する、ATTRIB ステートメントの属性(1 つ以上の変数に出力形式、入力形式、ラベル、長さを関連付ける)を指定します。

該当要素:	DATA ステップの ATTRIB ステートメント
カテゴリ:	情報
種類:	宣言型

構文

ATTRIB *variable-list(s) attribute-list(s)* ;

必須引数

variable-list

属性を設定する変数を指定します。

ヒント SAS で使用可能な任意の形式で変数をリストします。

attribute-list

variable-list に割り当てる 1 つまたは複数の属性を指定します。ATTRIB ステートメントには複数の属性を指定できます。属性の一覧については、“[ATTRIB ステートメント](#)” ([SAS DATA ステップステートメント: リファレンス](#)) を参照してください。

TRANSCODE= YES | NO

文字変数をトランスコードするかどうかを指定します。トランスコーディングを実行しないようにするには、TRANSCODE=NO を使用します。詳細については、“[トランスコーディングの概要](#)” (33 ページ) を参照してください。

デフ YES
オル
ト

制限 SAS Compute Server クライアントによっては、TRANSCODE=NO 属性がサポートされない場合があります。TRANSCODE=NO が指定された変数は、Viya では返されません。

操作 VTRANSCODE 関数および VTRANSCODEX 関数を使用すると、文字変数についてトランスコーディングがオンとオフのどちらであるかを返すことができます。

TRANSCODE=属性が NO に設定された文字変数がデータセット内にあると、PROC CONTENTS は、データセット内の変数ごとに、TRANSCODE=値が含まれるトランスコーディング列を出力します。

データセット内のすべての変数がデフォルトの TRANSCODE=値(YES)に設定された場合、トランスコーディング列は出力されません。

例

例 1: TRANSCODE=オプションと SET ステートメントの併用

SET ステートメントを使用して複数のデータセットからデータセットを 1 つ作成すると、出力データセット内の変数の TRANSCODE=属性が、最初のデータセット内の変数の TRANSCODE=値と同じに設定されます。この例では、B が最初のデータセットで、データセット B 内の Z の TRANSCODE=属性が NO であるため、データセット A 内の変数 Z の TRANSCODE=属性は NO です。

```
data b;
  length z $4;
  z = 'ice';
  attrib z transcode = NO;
data c;
  length z $4;
  z = 'snow';
  attrib z transcode = YES;
data a;
  set b;
  set c;
  /* Check transcode setting for variable Z */
  rc1 = vtranscode(z);
  put rc1=;
run;
```

例 2: TRANSCODE=オプションと MERGE ステートメントの併用

MERGE ステートメントを使用して複数のデータセットからデータセットを 1 つ作成すると、出力データセット内の変数の TRANSCODE=属性が、最初のデータセット内の変数の TRANSCODE=値と同じに設定されます。この例では、C が最初のデータセットで、データセット C 内の Z の TRANSCODE=属性が YES であるため、データセット A 内の変数 Z の TRANSCODE=属性は YES です。

```
data b;
  length z $4;
  z = 'ice';
  attrib z transcode = NO;
data c;
  length z $4;
  z = 'snow';
  attrib z transcode = YES;
data a;
  merge c b;
  /* Check transcode setting for variable Z */
  rc1 = vtranscode(z);
  put rc1=;
run;
```

注: TRANSCODE=属性は、変数が入力データセットまたは ATTRIB TRANSCODE=ステートメントで最初に検出されたときに設定されます。SET または MERGE ステートメントが、ATTRIB TRANSCODE=ステートメントよりも前に現れ、TRANSCODE=属性が SET ステートメントと矛盾する場合、エラーメッセージが表示されます。

関連項目

関数:

- ["VTRANSCODE 関数" \(752 ページ\)](#)
- ["VTRANSCODEX 関数" \(753 ページ\)](#)

TRANTAB=オプション

文字データをトランスコーディングする場合に使用する変換テーブルを指定します。

該当要素: ODS MARKUP ステートメントと ODS RTF ステートメント

カテゴリ: ODS: サードパーティによるフォーマティング

構文

TRANTAB = (*translation-table*)

オプション引数

translation-table

出力ファイルで使用する変換テーブルを指定します。変換テーブルは、文字セットの文字(通常の文字、表語文字、数字、句読点、記号、コントロール文字など)を数値にマッピングするエンコーディング方法です。変換テーブルの例として、文字を EBCDIC から ASCII-ISO に変換するものがあります。*table-name* には、SAS で提供される変換テーブルまたはユーザー定義の変換テーブルを指定できます。値は、SASUSER.PROFILE カタログまたは SASHELP.HOST カタログの SAS カタログエントリ名である必要があります。

詳細

SAS 9.4 では、下位互換性のために、ODS MARKUP ステートメントの TRANTAB=オプションがサポートされています。エンコーディングを指定する場合は ENCODING=の使用をお勧めします。

関連項目

概念的な情報:

- [“トランスコーディングと変換テーブル” \(34 ページ\)](#)

プロシジャ:

- [20 章, “TRANTAB プロシジャ,” \(1067 ページ\)](#)

ステートメント:

- [“ODS MARKUP Statement” \(SAS Output Delivery System: User's Guide\)](#)
- [“ODS RTF Statement” \(SAS Output Delivery System: User's Guide\)](#)

XMLENCODING=オプション

外部ドキュメントのインポートまたはエクスポート時に、XML ドキュメントのエンコーディングより優先される値を指定します。

該当要素: XML エンジンの LIBNAME ステートメント

カテゴリ: データアクセス

構文

XMLENCODING= *'encoding-value'*

詳細

XML エンジンの LIBNAME ステートメントは、外部ドキュメントをインポートまたはエクスポートするために、SAS ライブラリ参照名を XML ドキュメントに関連付けます。

比較

オプション

encoding-value

外部ファイルの読み込み、書き込み、コピー、保存に使用するエンコーディングを指定します。XMLENCODING=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エンコーディングとは異なることを示します。

詳細については、“データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値”(1118 ページ)を参照してください。

encoding-value のデフォルトは現在のセッションのエンコーディングです。

関連項目

ステートメント:

- SAS XMLV2 および XML LIBNAME エンジン: ユーザーガイド

TRANTAB ステートメント

SAS ファイルをエクスポートまたは転送するための文字データのトランスコーディング時に使用する、変換テーブルを指定します。

該当要素:	CPORT プロシジャ、UPLOAD プロシジャ、DOWNLOAD プロシジャ
制限事項:	1 つの TRANTAB ステートメントに対して指定できる変換テーブルは 1 つのみです。追加の変換テーブルを指定するには、追加の TRANTAB ステートメントを使用します。
操作:	TRANTAB ステートメントでは、エクスポートまたは転送される SAS ファイルの文字セットに適用する、カスタマイズした変換テーブル(たとえば、EBCDIC 文字の ASCII 文字へのマッピング)を指定します。TRANTAB=システムオプションでは、ファイル転送を含む、SAS セッションで使用される変換テーブルを指定します。

構文

TRANTAB NAME=*translation-table-name* <TYPE=(*etype-list*) <OPT=DISP | SRC | (DISP SRC)> > ;

必須引数

NAME=*translation-table-name*

エクスポート(PROC CPORT)または転送(PROC UPLOAD または PROC DOWNLOAD)する SAS カタログに適用する変換テーブルの名前を指定します。SASUSER.PROFILE カタログまたは SASHELP.HOST カタログのカatalog エントリの名前として指定する *translation-table-name*。SASUSER.PROFILE カタログが先に検索され、SASHELP.HOST カタログはその次に検索されます。

大半はデフォルトの変換テーブルが使用されますが、アプリケーションに異なる各国語文字が必要な場合など、追加の変換テーブルの適用が必要になることもあります。

デフォルト以外の変換テーブルを指定するには、次の 2 つの方法があります。

- プロシジャの起動時に変換テーブルを指定するには、必要に応じて、プロシジャで TRANTAB ステートメントを使用します。
- SAS セッションまたはジョブ全体(すべてのファイルエクスポートまたは転送を含む)に変換テーブルを指定するには、TRANTAB=システムオプションを使用します。

オプション引数

TYPE=(etype-list)

指定したタイプのエントリにのみ変換テーブルを適用します。etype-list には、1 つ以上のエントリのタイプを指定できます。たとえば、カタログエントリのタイプには、DATA や FORMAT などがあります。etype-list に指定するエントリタイプが 1 つのみの場合、かっこを省略します。

デフォルトで、UPLOAD、DOWNLOAD および CPORT プロシジャは、指定したすべてのカタログエントリに変換テーブルを適用します。

OPT=DISP | SRC(|)

OPT=DISP 指定したカタログエントリのうち、ウィンドウ表示を作成するエントリにのみ変換テーブルを適用します。

OPT=SRC 指定したカタログエントリのうち、タイプが SOURCE のエントリにのみ変換テーブルを適用します。

OPT=(DISP SRC) 指定したカタログエントリのうち、ウィンドウ表示を作成するか、タイプが SOURCE のエントリにのみ変換テーブルを適用します。

OPT=オプションを指定しない場合、UPLOAD または DOWNLOAD プロシジャは、変換テーブルを指定したカタログ内のすべてのエントリに適用します。

デフォルト PROC CPORT、PROC UPLOAD および PROC DOWNLOAD は、変換テーブルを指定したカタログ内のすべてのエントリとデータセットに適用します。

詳細

変換テーブルは、各国語の要件をサポートするために SAS 6 で導入されました。SAS 8.2 では、変換テーブルの直接使用より優れた、LOCALE=システムオプションが導入されました。SAS 9.3 では、TRANTAB ステートメントが下位互換性を維持するためにサポートされています。ただし、SAS 8.2 以降のリリースでは、LOCALE=システムオプションを使用することをお勧めします。詳細については、TS-639:Data Conversion Issues in V6-V8 を参照してください。このテクニカルサポートの注記では、英語以外の言語を使用している場合について説明しています。<http://support.sas.com/techsup/technote/ts639.pdf>

PROC CPORT はネットワークを介して SAS ファイルをエクスポートする場合に使用します。PROC UPLOAD と PROC DOWNLOAD はネットワークを介して SAS ファイルを転送する場合に使用します。

TRANTAB ステートメントの使用時、PROC UPLOAD または PROC DOWNLOAD ステートメントに INCAT=および OUTCAT=オプションを指定する必要があります。

例

例 1

次の情報をプロシジャ機能に適用します。

- PROC CPORT ステートメントオプション: FILE=
- TRANTAB ステートメントオプション: TYPE=

この例では、カスタマイズした変換テーブルを、PROC CPORT がエクスポートする前にトランスポートファイルに適用する方法を示します。この例では、カスタマイズした変換テーブル TTABLE1 はすでに作成されたと想定します。

例 2: プログラム

ライブラリ参照を割り当てます。</userSuppliedValue>LIBNAME ステートメントと FILENAME ステートメントは、ソースライブラリのライブラリ参照名とトランスポートファイルのファイル参照名をそれぞれ割り当てます。

```
libname source
'\\sashq\root\pub\pubdoc\doc\901\authoring\proc\miscsrc\sasfiles\cport';
filename tranfile 'trans3';
proc trantab table=ascii;
save table=ttable1;
```

```
libname source 'SAS data-library';
filename tranfile 'transport-file'
             host-options-for-file-characteristics;
```

変換の詳細を適用します。</userSuppliedValue>TRANTAB ステートメントは、カスタマイズされた変換テーブル TTABLE1 で指定した変換を適用します。TYPE=によって、変換の適用対象が FORMAT エントリだけに制限されます。

```
proc cport catalog=source.formats file=tranfile;
  trantab name=ttable1 type=(format);
run;
```

例 3: SAS ログ

NOTE: Proc CPORT begins to transport catalog SOURCE.FORMATS
 NOTE: The catalog has 2 entries and its maximum logical record length is 104.
 NOTE: Entry REVENUE.FORMAT has been transported.
 NOTE: Entry DEPT.FORMATC has been transported.

関連項目

概念的な情報:

- [4 章, “各国語サポート関連のトランスコーディング” \(33 ページ\)](#)

システムオプション:

- [“TRANTAB= システムオプション” \(988 ページ\)](#)

プロシジャ:

- [20 章, “TRANTAB プロシジャ,” \(1067 ページ\)](#)
- [“CPORT Procedure” \(*Base SAS Procedures Guide*\)](#)
- [“UPLOAD Procedure” \(*SAS/CONNECT User’s Guide*\)](#)
- [“DOWNLOAD Procedure” \(*SAS/CONNECT User’s Guide*\)](#)

10 部**各国語サポート関連のプロシジャ**

18 章		
	DBCSTAB プロシジャ	1043
19 章		
	LOCALEDATA プロシジャ	1051
20 章		
	TRANTAB プロシジャ	1067

DBCSTAB プロシジャ

概要: DBCSTAB プロシジャ	1043
DBCSTAB プロシジャの機能	1043
構文: DBCSTAB プロシジャ	1044
PROC DBCSTAB ステートメント	1044
例: DBCSTAB プロシジャ	1045
例 1: DBCSTAB プロシジャを使用したコンバージョンテーブルの作成	1045
例 2: DBCSTAB プロシジャを使用した日本語コンバージョンテーブルの作成 ..	1047

概要: DBCSTAB プロシジャ

DBCSTAB プロシジャの機能

DBCSTAB プロシジャは SAS でサポートされる 2 バイト文字セット(DBCS)に対するコンバージョンテーブルを作成します。

次の場合には、DBCSTAB プロシジャを使用して既存の DBCS テーブルを変更します。

- 使用している DBCS エンコーディングシステムが SAS によりサポートされていない場合
- 使用している DBCS エンコーディングシステムが標準でないコンバージョンテーブルを含んでいる場合

なお、有効な DBCSTYPE=の値が利用できない場合に、DBCSTAB プロシジャを使用することもできます。これらの値は動作環境に依存します。このような場合、DBCSTAB プロシジャを使用して似た変換テーブルを変更してから、TRANTAB オプションで新しいテーブルを使用するように指定できます。

構文: DBCSTAB プロシジャ

```
PROC DBCSTAB TABLE=table-name
<BASETYPE=base-type> <CATALOG=<libref.>catalog-name>
<DATA=<libref.>table-name> <DBCSLANG=language>
<DESC='description'> <FORCE> <VERIFY> <VERBOSE>;
```

PROC DBCSTAB ステートメント

2 バイト文字セット(DBCS)に対するコンバージョンテーブルを作成します。

構文

```
PROC DBCSTAB TABLE=table-name
<options>;
```

必須引数

TABLE=*table-name*

作成する 2 バイトコードテーブルの名前を指定します。このテーブル名は、CATALOG=オプションで指定されたカタログ内のタイプ DBCSTAB のエントリになります。カタログのデフォルト名は、SASUSER.DBCS です。

別名 NAME=

N=

オプション引数

BASETYPE=*base-type*

2 バイトコードテーブル変換の基準となるタイプを指定します。このオプションを使用すると、生成されるテーブルの数を減らすことができます。

BASETYPE=を指定すると、すべての 2 バイトコードがまずベースコードに変換され、続いてそれらが必要なコードへと変換されます。 n 個のコードがある場合、 $n(n-1)$ 回の変換を行う必要があります。

別名 BTYPE=

CATALOG=<*libref*.>*catalog-name*

テーブルの保存先となるカタログの名前を指定します。カタログが存在しない場合は、作成されます。

デフォルト Sasuser.DBCS

DATA=<libref.>table-name

2 バイトコードテーブルを作成するためのデータを指定します。テーブルを作成するには、複数の 2 バイト文字変数が必要となります。DBCSTYPE=システムオプションの値と等しく、KCVT 関数で認識される変数名を使用します。

DBCSLANG=language

2 バイトコードテーブルで使用される言語を指定します。このオプションの値は、DBCSLANG システムオプションの値と一致する必要があります。

別名 DBLANG=

DESC='description'

エントリの DESCRIPTION フィールドに設定するテキスト文字列を指定します。

FORCE

エラーがある場合でもコンバージョンテーブルを生成します。

VERIFY

コードに基づいて入力テーブルのデータ範囲を確認します。このオプションは、無効な 2 バイトコードがないかどうかを確認します。

VERBOSE

DBCS テーブル作成時に統計の詳細を出力します。

例: DBCSTAB プロシジャ

例 1: DBCSTAB プロシジャを使用したコンバージョンテーブルの作成

要素:	PROC DBCSTAB ステートメントオプション: CATALOG= DBLANG= BASETYPE= VERIFY
-----	--

次の例では、日本語変換テーブル CUSTAB を作成し、TRANTAB=オプションで新しい変換テーブルを指定する方法を説明します。DBCS=、DBCSLANG=、DBCSTYPE=の各オプションは起動時に指定されます。

プログラム

```
proc dbcstab
  /* name of the new translate table */
  name=custtab
  /* based on pcibm encoding */
  basetype=pcms
  /* data to create the new table */
  data=trantab
  /* japanese language */
  dbcslang=japanese
  /* catalog descriptor */
  desc='Modified Japanese Trantab'
  /* where the table is stored */
  catalog=sasuser.dbcs
  /* checks for invalid DBCS in the new data */
  verify;
run;
```

コンバージョンテーブルの作成

```
invoke Japanese SAS with Shift_JIS encoding.

data trantab;

pcms='8342'x; dec='b9b3'x;
run;

proc dbcstab;
name=custtab;
basetype=pcms;
data=trantab;
dbcslang=japanese;
desc='Modified Japanese Trantab';
catalog=sasuser.dbcs;
verify;
run;

options trantab=(,,,,,,,,custtab);
```

プログラムの説明

SAS 環境を設定します。

invoke Japanese SAS with Shift_JIS encoding.

TRANTAB データセットを作成します。

```
data trantab;
```

pcms パラメーターと dec パラメーターを定義し、コードを実行します。

```
pcms='8342'x; dec='b9b3'x;
run;
```

DBCSTAB プロシジャを呼び出します。

```
proc dbcstab;
```

変換テーブルに名前を割り当てます。

```
name=custtab;
```

pcibm エンコーディングを適用します。

```
basetype=pcms;
```

TranTab データセットのデータにアクセスします。

```
data=trantab;
```

言語を日本語に設定します。

```
dbcslang=japanese;
```

変更した日本語変換テーブルである旨の説明をカタログディスクリプターに設定します。

```
desc='Modified Japanese Trantab';
```

テーブルの場所を設定します。

```
catalog=sasuser.dbcs;
```

無効な ABCS データがないか確認し、プログラムを処理してください。

```
verify;
run;
```

TRANTAB=オプションで変換テーブルを選択します。 変換テーブルは、SAS/CONNECT、PROC CPORT や PROC CIMPORT、DATA ステップ関数 KCVT を使用した DBCS 変換で使用されます。TRANTAB=オプションにより DBCS 変換テーブルを指定することもできます。SAS 8.2 以前のバージョンでは、9 番目の引数を使用して DBCS システムテーブルを指定していました。ただし、SAS 9 以降では、9 番目の引数を指定せずに、モジュールに含まれているシステムテーブルを使用するようになりました。SYSTAB 名として、Japanese、Korean、Chinese、Taiwanese を指定できます。10 番目の引数は DBCS ユーザーテーブルを指定します。

```
options trantab=(,,,,,,,,custtab);
```

例 2: DBCSTAB プロシジャを使用した日本語コンバージョンテーブルの作成

要素: PROC DBCSTAB ステートメントオプション:
 TABLE=
 DATA=
 DBLANG=
 BASETYPE=
 VERIFY

プログラム

```
data ja_jpn;  
length ibm jis euc pcibm $2.;  
ibm='4040'x;  
jis='2121'x;  
euc='a1a1'x;  
pcibm='8140'x;  
run;  
  
proc dbcstab;  
table=japanese;  
data=ja_jpn;  
dblang=japanese;  
basetype=jis;  
verify;  
run;
```

プログラムの説明

ja_jpn データセットを定義します。

```
data ja_jpn;
```

指定したエンコーディングの長さを 2 に設定します。

```
length ibm jis euc pcibm $2.;
```

値 **4040** を **ibm** エンコーディングに代入します。

```
ibm='4040'x;
```

値 **2121** を **jis** エンコーディングに代入します。

```
jis='2121'x;
```

値 **a1a1** を **euc** エンコーディングに代入します。

```
euc='a1a1'x;
```

値 **8140** を **pcibm** エンコーディングに割り当て、コードを実行します。

```
pcibm='8140'x;  
run;
```

DBCSTAB プロシジャを呼び出します。

```
proc dbcstab;
```

japanese という名前でテーブルを定義します。

```
table=japanese;
```

Ja_Jpn データセットにアクセスします。

```
data=ja_jpn;
```

言語を日本語に設定します。

```
dblang=japanese;
```

jis エンコーディングを指定します。

```
basetype=jis;
```

無効な ABCS データがないか確認し、プログラムを処理してください。

```
verify;
```

```
run;
```

SAS ログ

```
1 proc dbcstab
2 table=ja_jpn
3 data=work.ja_jpn
4 dblang=japanese
5 basetype=jis
6 verify;
7 run;
```

NOTE: Base table for JIS created.

NOTE: IBM table for JIS created.

NOTE: PCIBM table for JIS created.

NOTE: EUC table for JIS created.

NOTE: Base table for IBM created.

NOTE: JIS table for IBM created.

NOTE: Base table for PCIBM created.

NOTE: JIS table for PCIBM created.

NOTE: Base table for EUC created.

NOTE: JIS table for EUC created.

NOTE: 10 DBCS tables are generated.Each table has 1 DBCS characters.

NOTE: Each table is 2 bytes in size.

NOTE: Required table memory size is 612.

NOTE: There were 1 observations read from the data set WORK.JA_JPN.

LOCALEDATA プロシジャ

概要: LOCALEDATA プロシジャ	1051
LOCALEDATA プロシジャの機能	1051
構文: LOCALEDATA プロシジャ	1051
PROC LOCALEDATA ステートメント	1052
LOAD ステートメント	1052
MODIFY ステートメント	1053
CONTENTS ステートメント	1059
SAVE ステートメント	1060
例: LOCALEDATA プロシジャを使用したロケールデータの変更	1060

概要: LOCALEDATA プロシジャ

LOCALEDATA プロシジャの機能

LOCALEDATA プロシジャを使用すると、データをカスタマイズできます。カスタマイズしたロケールデータの表示、オープン、変更、保存が行えます。

構文: LOCALEDATA プロシジャ

```
PROC LOCALEDATA;
  LOAD SASLOCALE | REGISTRY;
```

```
MODIFY key=key-name value=key-value | category=category-name
value=locale;
CONTENTS _ALL_ | key-name | category-name;
SAVE REGISTRY <_ALL_ | key-name | category-name syntax=SAS>;
```

ステートメント	タスク
"PROC LOCALEDATA ステートメント"	カスタマイズするロケールデータのソースを指定します。
"LOAD ステートメント"	ロケール要素キーまたはロケールカテゴリのロケールデータをカスタマイズする方法を指定します。
"CONTENTS ステートメント"	破棄するロケール要素の値を指定します。
"SAVE ステートメント"	カスタマイズされたロケールデータを SAS レジストリに保存します。
"MODIFY ステートメント"	ロケール要素キーまたはロケールカテゴリのロケールデータをカスタマイズします。

PROC LOCALEDATA ステートメント

カスタマイズするロケールデータのソースを指定します。

構文

```
PROC LOCALEDATA;
```

LOAD ステートメント

カスタマイズ用にロードするロケールデータのソースを指定します。

デフォルト: レジストリ

構文

```
LOAD SASLOCALE | REGISTRY;
```

必須引数

SASLOCALE

工場出荷時のデフォルトロケールデータベースからロケールデータをロードします。

REGISTRY

SAS レジストリからロケールデータをロードします。このロケールデータは、カスタマイズされ、SAS レジストリに保存されたロケールデータの可能性があります。

MODIFY ステートメント

ロケール要素キーまたはロケールカテゴリのロケールデータをカスタマイズします。

構文

MODIFY *key=key-name value=key-value | category=category-name value=locale*

必須引数

key=key-name value=key-value

1 つのロケール要素値をカスタマイズします。

category=category-name value=locale

カテゴリ内のすべてのロケールデータをカスタマイズします。次のカテゴリのうち 1 つを選択できます。

- LC_NUMERIC
- LC_MONETARY
- LC_TIME

ユーザーが変更可能な要素キーを次の表に示します。

ロケール要素キーの値は、最大長の値を超えてはなりません。

次の値をタイプに使用できます。

0 string

1 二重引用符で囲んだ符号なし整数

ロケール要素キー	最大長	タイプ	カテゴリ
DATESTYLE	3	0	
PAPERSIZE	8	0	

ロケール要素キー	最大長	タイプ	カテゴリ
FTITLE	512	0	
FTEXT	512	0	
SIMFONT	512	0	
SORTSEQ	8	0	
MESSAGES	8	0	
FORMATNAME_DATE	512	0	
FORMATNAME_DATE TIME	512	0	
FORMATNAME_TIME	512	0	
FORMATNAME_NUM ERIC	512	0	
FORMATNAME_PERC ENT	512	0	
FONT_SERIF	32	0	
FONT_SANSERIF	32	0	
FONT_CURSIVE	32	0	
FONT_FANTASY	32	0	
FONT_MONOSPACE	32	0	
BRUSH	32	0	
SIMPLEX	32	0	
COMPLEX	32	0	
SWISS	32	0	
ITALIC	32	0	
DATE_FORMAT	512	0	LC_TIME
DATE_SHORT_FORM AT	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
DATETIME_AMPM_FORMAT	512	0	LC_TIME
DATETIME_FORMAT	512	0	LC_TIME
DATETIME_SHORT_FORMAT	512	0	LC_TIME
DATETIME_WEEK_FORMAT	512	0	LC_TIME
DATETIME_WEEK_SHORT_FORMAT	512	0	LC_TIME
TIME_AMPM_FORMAT	512	0	LC_TIME
TIME_FORMAT	512	0	LC_TIME
DATE_WEEK_FORMAT	512	0	LC_TIME
DATE_WEEK_SHORT_FORMAT	512	0	LC_TIME
DATE_YYMM_FORMAT	512	0	LC_TIME
DATE_YYMM_SHORT_FORMAT	512	0	LC_TIME
DATE_MMDD_FORMAT	512	0	LC_TIME
DATE_MMDD_SHORT_FORMAT	512	0	LC_TIME
DATE_YEAR_FORMAT	512	0	LC_TIME
DATE_YEAR_SHORT_FORMAT	512	0	LC_TIME
DATE_YYQQ_FORMAT	512	0	LC_TIME
DATE_YYQQ_SHORT_FORMAT	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
DATE_YYWW_FORMAT	512	0	LC_TIME
DATE_YYWW_SHORT_FORMAT	512	0	LC_TIME
DATE_SEP	8	0	LC_TIME
ABMON01	512	0	LC_TIME
ABMON02	512	0	LC_TIME
ABMON03	512	0	LC_TIME
ABMON04	512	0	LC_TIME
ABMON05	512	0	LC_TIME
ABMON06	512	0	LC_TIME
ABMON07	512	0	LC_TIME
ABMON08	512	0	LC_TIME
ABMON09	512	0	LC_TIME
ABMON10	512	0	LC_TIME
ABMON11	512	0	LC_TIME
ABMON12	512	0	LC_TIME
MON01	512	0	LC_TIME
MON02	512	0	LC_TIME
MON03	512	0	LC_TIME
MON04	512	0	LC_TIME
MON05	512	0	LC_TIME
MON06	512	0	LC_TIME
MON07	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
MON08	512	0	LC_TIME
MON09	512	0	LC_TIME
MON10	512	0	LC_TIME
MON11	512	0	LC_TIME
MON12	512	0	LC_TIME
ABDAY1	512	0	LC_TIME
ABDAY2	512	0	LC_TIME
ABDAY3	512	0	LC_TIME
ABDAY4	512	0	LC_TIME
ABDAY5	512	0	LC_TIME
ABDAY6	512	0	LC_TIME
ABDAY7	512	0	LC_TIME
DAY1	512	0	LC_TIME
DAY2	512	0	LC_TIME
DAY3	512	0	LC_TIME
DAY4	512	0	LC_TIME
DAY5	512	0	LC_TIME
DAY6	512	0	LC_TIME
DAY7	512	0	LC_TIME
AM	512	0	LC_TIME
PM	512	0	LC_TIME
ABQTR1	512	0	LC_TIME
ABQTR2	512	0	LC_TIME

ロケール要素キー	最大長	タイプ	カテゴリ
ABQTR3	512	0	LC_TIME
ABQTR4	512	0	LC_TIME
QTR1	512	0	LC_TIME
QTR2	512	0	LC_TIME
QTR3	512	0	LC_TIME
QTR4	512	0	LC_TIME
INT_CURRENCY_SYM BOL	3	0	LC_MONETARY
CURRENCY_SYMBOL	32	0	LC_MONETARY
MON_DECIMAL_POI NT	8	0	LC_MONETARY
MON_THOUSANDS_S EP	8	0	LC_MONETARY
MON_GROUPING	3	1	LC_MONETARY
MON_POSITIVE_SIG N	8	0	LC_MONETARY
MON_NEGATIVE_SIG N	8	0	LC_MONETARY
MON_INT_FRAC_DIG ITS	3	1	LC_MONETARY
MON_FRAC_DIGITS	3	1	LC_MONETARY
MON_P_CS_PRECEDE S	3	1	LC_MONETARY
MON_P_SEP_BY_SPA CE	3	1	LC_MONETARY
MON_N_CS_PRECED ES	3	1	LC_MONETARY
MON_N_SEP_BY_SPA CE	3	1	LC_MONETARY

ロケール要素キー	最大長	タイプ	カテゴリ
MON_P_SIGN_POSN	3	1	LC_MONETARY
MON_N_SIGN_POSN	3	1	LC_MONETARY
DECIMAL_POINT	1	0	LC_NUMERIC
THOUSANDS_SEP	1	0	LC_NUMERIC
GROUPING	3	1	LC_NUMERIC
POSITIVE_SIGN	8	0	LC_NUMERIC
NEGATIVE_SIGN	8	0	LC_NUMERIC
P_CS_PRECEDES	3	1	LC_NUMERIC
P_SEP_BY_SPACE	3	1	LC_NUMERIC
N_CS_PRECEDES	3	1	LC_NUMERIC
N_SEP_BY_SPACE	3	1	LC_NUMERIC
P_SIGN_POSN	3	1	LC_NUMERIC
N_SIGN_POSN	3	1	LC_NUMERIC
HEIGHT	3	1	
WIDTH	3	1	

CONTENTS ステートメント

指定されたロケール要素キーの値を表示します。

構文

CONTENTS[_ALL_](#) | *locale-element-key* | *category-name*

必須引数

ALL

すべてのロケール要素キーの値を表示します。

locale-element-key

指定されたロケール要素キーの値を表示します。

category-name

指定したカテゴリのロケール要素の値を表示します。

SAVE ステートメント

カスタマイズされたロケールデータを SAS レジストリに保存します。

構文

SAVE REGISTRY < *_ALL_* | *key-name* | *category-name syntax=SAS* >

必須引数

REGISTRY

ロケールデータを SASUSER ライブラリに保存し、SAS レジストリに登録します。保存されたロケールデータのエンコーディングは UNICODE エスケープになります。SAS レジストリ内にある保存されたロケール用の階層は次のようになります。**SAS_REGISTRY** ⇒ **LOCALE** ⇒ **SAS ロケール** ⇒ **ロケール要素キーロケールカテゴリ** ⇒ **ロケール要素キー**

オプション引数

ALL

すべてのロケールデータ要素を保存します。

key-name

saves the specified locale element key name and value.

category-name syntax=SAS

すべての要素を指定のカテゴリ内に保存します。

例: LOCALEDATA プロシジャを使用したロケールデータの変更

要素:

PROC LOCALEDATA ステートメント

LOAD ステートメント
 MODIFY ステートメント
 SAVE ステートメント
 CONTENTS ステートメント
 DATA ステップ
 PUT ステートメント

次のプログラムでは、ロケールデータを変更します。SAS システムは 2 回起動されます。

ロケール定義の変更

```
localedata=registry;
PROC LOCALEDATA;
LOAD SASLOCALE;
MODIFY KEY=QTR1 VALUE='QTR1';
MODIFY KEY=QTR2 VALUE='QTR2';
MODIFY KEY=QTR3 VALUE='QTR3';
MODIFY KEY=QTR4 VALUE='QTR4';
MODIFY KEY=DATE_SHORT_FORMAT VALUE='%Y//%m//%d';
MODIFY KEY=DATE_YEAR_FORMAT VALUE='Year(%Y)';
SAVE REGISTRY / _ALL_ syntax=SAS;
CONTENTS _ALL_;
QUIT;
data _null_;
a = 19208;
put a nldate12. / a nldateyq. / a nldateyr.;
run;
```

プログラムの説明

“[LOCALEDATA システムオプション](#)”を設定する必要があります。

LOCALEDATA システムオプションを registry に設定します。

```
localedata=registry;
```

LOCALEDATA プロシジャを呼び出します。

```
PROC LOCALEDATA;
```

Load the locale definitions from the local locale database.

```
LOAD SASLOCALE;
```

定義値を変更します。

```
MODIFY KEY=QTR1 VALUE='QTR1';
```

```
MODIFY KEY=QTR2 VALUE='QTR2';  
MODIFY KEY=QTR3 VALUE='QTR3';  
MODIFY KEY=QTR4 VALUE='QTR4';  
MODIFY KEY=DATE_SHORT_FORMAT VALUE='%Y///%m//%d';  
MODIFY KEY=DATE_YEAR_FORMAT VALUE='Year(%Y)';
```

変更した定義をレジストリに保存します。 この操作によりレジストリに保存されている定義が上書きされるので、重複するキー値はありません。

```
SAVE REGISTRY / _ALL_syntax=SAS;
```

レジストリの変更されたロケール定義を表示します。

```
CONTENTS _ALL_;
```

QUIT コマンドを呼び出して、処理を終了します。

```
QUIT;
```

変更されたロケール定義をテストするための DATA ステップを作成します。

```
data _null_;
```

値 19208 を変数 a に代入します。

```
a = 19208;
```

変数値 a の値を、NLDATE.、NLDATEYQ.、NLDATEYR.の形式で生成します。

```
put a nldate12. / a nldateyq. / a nldateyr.;  
run;
```

SAS ログ

```

1  PROC LOCALEDATA;
2
3  /* Load locale defintion from locale DB */
4  LOAD SASLOCALE;
5
6  /* Change the value of the elements */
7  MODIFY KEY=QTR1 VALUE='QTR1';
8  MODIFY KEY=QTR2 VALUE='QTR2';
9  MODIFY KEY=QTR3 VALUE='QTR3';
10  MODIFY KEY=QTR4 VALUE='QTR4';
11  MODIFY KEY=DATE_SHORT_FORMAT VALUE='%Y//%m//%d';
12  MODIFY KEY=DATE_YEAR_FORMAT VALUE='Year(%Y)' ;
13  /* Store the locale definition into registry, this will override the existing definition,
13 ! so there is no duplicate key issue */
14  SAVE REGISTRY / _ALL_ syntax=SAS;
15
16  /* View the locale definition from registry */
17  CONTENTS _ALL_;
LOCALE="English_UnitedStates"
LANGUAGE="English"
LANGUAGE_SCRIPT="en"
TERRITORY="UnitedStates"
LANGID="en"
LOCID="US"
DATESTYLE="MDY"
PAPERSIZE="LETTER"
FTITLE=""
FTEXT=""
SIMFONT=""
SORTSEQ=""
MESSAGES=""
FORMATNAME_DATE="NLDATE16."
FORMATNAME_DATETIME="NLDATM24."
FORMATNAME_TIME="NLTIME8."
FORMATNAME_NUMERIC="BEST12."
FORMATNAME_PERCENT="PERCENT12."
FONT_SERIF="COMPLEX"
FONT_SANSERIF="SIMPLEX"
FONT_CURSIVE="ITALIC"
FONT_FANTASY="BRUSH"
FONT_MONOSPACE="SIMPLEX"
BRUSH="BRUSH"
SIMPLEX="SIMPLEX"
COMPLEX="COMPLEX"
SWISS="SWISS"
ITALIC="ITALIC"
DATE_FORMAT="%B %d, %Y"
DATE_SHORT_FORMAT="%Y//%m//%d"
DATE_MEDIUM_FORMAT="%b %e, %Y"
DATE_LONG_FORMAT="%B %e, %Y"
DATE_FULL_FORMAT="%A, %B %e, %Y"
DATE_FULL_AB_FORMAT="%a, %b %e, %Y"
DATE_YYMM_FORMAT="%B %Y"
DATE_YYMM_SHORT_FORMAT="%b %Y"
DATE_MMDD_FORMAT="%B %d"
DATE_MMDD_SHORT_FORMAT="%b %d"
DATE_YEAR_FORMAT="Year(%Y)"
DATE_YEAR_SHORT_FORMAT="%y"
DATE_YYQQ_FORMAT="%Q %Y"
DATE_YYQQ_SHORT_FORMAT="%q %Y"
DATE_YYWW_FORMAT="Week %U %Y"
DATE_YYWW_SHORT_FORMAT="W%U %y"

```

```

DATETIME_AMPM_FORMAT="%B %e, %Y %I:%M:%S %p"
DATETIME_AMPM_AB_FORMAT="%b %e, %Y %I:%M:%S %p"
DATETIME_FORMAT="%d%b%Y-%H:%M:%S"
DATETIME_SHORT_FORMAT="%o/%e/%Y %I:%M:%S %p"
DATETIME_MEDIUM_FORMAT="%b %e, %Y %I:%M:%S %p"
DATETIME_LONG_FORMAT="%B %e, %Y %I:%M:%S %p"
DATETIME_FULL_FORMAT="%A, %B %e, %Y %I:%M:%S %p"
DATETIME_FULL_AB_FORMAT="%a, %b %e, %Y %I:%M:%S %p"
TIME_AMPM_FORMAT="%I:%M:%S %p"
TIME_FORMAT="%H:%M:%S"
TIME_SHORT_FORMAT="%I:%M %p"
TIME_MEDIUM_FORMAT="%I:%M:%S %p"
TIME_LONG_FORMAT="%I:%M:%S %p %Z"
TIME_FULL_FORMAT="%I:%M:%S %p %Z"
DATETIME_PATTERN="%x %X"
ABMON01="Jan"
ABMON02="Feb"
ABMON03="Mar"
ABMON04="Apr"
ABMON05="May"
ABMON06="Jun"
ABMON07="Jul"
ABMON08="Aug"
ABMON09="Sep"
ABMON10="Oct"
ABMON11="Nov"
ABMON12="Dec"
MON01="January"
MON02="February"
MON03="March"
MON04="April"
MON05="May"
MON06="June"
MON07="July"
MON08="August"
MON09="September"
MON10="October"
MON11="November"
MON12="December"
SA_ABMON01="Jan"
SA_ABMON02="Feb"
SA_ABMON03="Mar"
SA_ABMON04="Apr"
SA_ABMON05="May"
SA_ABMON06="Jun"
SA_ABMON07="Jul"
SA_ABMON08="Aug"
SA_ABMON09="Sep"
SA_ABMON10="Oct"
SA_ABMON11="Nov"
SA_ABMON12="Dec"
SA_MON01="January"
SA_MON02="February"
SA_MON03="March"
SA_MON04="April"
SA_MON05="May"
SA_MON06="June"
SA_MON07="July"
SA_MON08="August"
SA_MON09="September"
SA_MON10="October"
SA_MON11="November"
SA_MON12="December"

```

```

ABDAY1="Sun"
ABDAY2="Mon"
ABDAY3="Tue"
ABDAY4="Wed"
ABDAY5="Thu"
ABDAY6="Fri"
ABDAY7="Sat"
DAY1="Sunday"
DAY2="Monday"
DAY3="Tuesday"
DAY4="Wednesday"
DAY5="Thursday"
DAY6="Friday"
DAY7="Saturday"
SA_ABDAY1="Sun"
SA_ABDAY2="Mon"
SA_ABDAY3="Tue"
SA_ABDAY4="Wed"
SA_ABDAY5="Thu"
SA_ABDAY6="Fri"
SA_ABDAY7="Sat"
SA_DAY1="Sunday"
SA_DAY2="Monday"
SA_DAY3="Tuesday"
SA_DAY4="Wednesday"
SA_DAY5="Thursday"
SA_DAY6="Friday"
SA_DAY7="Saturday"
ABQTR1="Q1"
ABQTR2="Q2"
ABQTR3="Q3"
ABQTR4="Q4"
QTR1="QTR1"
QTR2="QTR2"
QTR3="QTR3"
QTR4="QTR4"
SA_ABQTR1="Q1"
SA_ABQTR2="Q2"
SA_ABQTR3="Q3"
SA_ABQTR4="Q4"
SA_QTR1="1st quarter"
SA_QTR2="2nd quarter"
SA_QTR3="3rd quarter"
SA_QTR4="4th quarter"
AM="AM"
PM="PM"
DATE_SEP="/"
FIRST_DAY_OF_WEEK="0"
INT_CURRENCY_SYMBOL="USD"
CURRENCY_SYMBOL="$"
MON_DECIMAL_POINT="."
MON_THOUSANDS_SEP=","
MON_GROUPING="3"
MON_POSITIVE_SIGN=""
MON_NEGATIVE_SIGN="-"
MON_INT_FRAC_DIGITS="2"
MON_FRAC_DIGITS="2"
MON_P_CS_PRECEDES="1"
MON_P_SEP_BY_SPACE="0"
MON_N_CS_PRECEDES="1"
MON_N_SEP_BY_SPACE="0"
MON_P_SIGN_POSN="1"
MON_N_SIGN_POSN="0"

```

```

CURR_FMT_L="#,##0.00;(#,##0.00)"
CURR_FMT_I="#,##0.##;(#,##0.##)"
NUM_DECIMAL_POINT="."
NUM_THOUSANDS_SEP=","
NUM_GROUPING="3"
NUM_POSITIVE_SIGN=""
NUM_NEGATIVE_SIGN="-"
NUM_P_CS_PRECEDES="1"
NUM_P_SEP_BY_SPACE="0"
NUM_N_CS_PRECEDES="1"
NUM_N_SEP_BY_SPACE="0"
NUM_P_SIGN_POSN="1"
NUM_N_SIGN_POSN="1"
DEC_FMT_L="#,##0.###"
DEC_FMT_I="#,##0.##"
PCT_FMT_L="#,##0%"
PCT_FMT_I="#,##0.##%"
HEIGHT="279"
WIDTH="216"
18
19  QUIT;

```

NOTE: PROCEDURE LOCALEDATA used (Total process time):

real time	0.24 seconds
cpu time	0.18 seconds

```

20
21  data _null_ ;
22  a = 19208 ;
23  put a nldate./ a NLDATEYQ./ a NLDATEYR.;
24  run ;

```

August 03, 2012
3rd quarter 2012
2012

NOTE: DATA statement used (Total process time):

real time	2.96 seconds
cpu time	0.28 seconds

TRANTAB プロシジャ

概要: TRANTAB プロシジャ	1067
TRANTAB プロシジャの機能	1068
概念: TRANTAB プロシジャ	1069
PROC TRANTAB で使用する変換テーブルと文字セットについて	1069
PROC TRANTAB を使用した変換テーブルの保存	1069
PROC TRANTAB を使用した SAS 変換テーブルの変更	1070
PROC TRANTAB 外での変換テーブルの使用	1070
構文: TRANTAB プロシジャ	1072
PROC TRANTAB ステートメント	1073
CLEAR ステートメント	1074
INVERSE ステートメント	1074
LIST ステートメント	1075
LOAD ステートメント	1075
REPLACE ステートメント	1076
SAVE ステートメント	1077
SWAP ステートメント	1078
例: TRANTAB プロシジャ	1079
例 1: 変換テーブルの表示	1079
例 2: 変換テーブルの作成	1080
例 3: 開始位置に 10 進値を指定した編集	1083
例 4: 開始位置に引用符付き文字を使用した編集	1085
例 5: 逆変換テーブルの作成	1088
例 6: 並べ替えへのさまざまな変換テーブルの使用	1089
例 7: テーブル 1 とテーブル 2 の編集	1091

概要: TRANTAB プロシジャ

TRANTAB プロシジャの機能

TRANTAB プロシジャは、カスタマイズで変換テーブルを作成、編集、表示します。また、PROC TRANTAB を使用して、SAS が提供する変換テーブルを表示および変更することもできます。これらの SAS 提供のテーブルは、Sashelp.Host カタログに保存されています。作成またはカスタマイズした変換テーブルは、Sasuser.Profile カタログに格納されます。変換テーブルには TRANTAB のエントリタイプがあります。

変換テーブルは動作環境に特有の SAS カタログエントリで、ある(コード化済み)文字セットの値を、別の文字セットの値に変換するものです。変換テーブルは、次の 2 つのテーブルにわかれています。テーブル 1 は、ASCII から EBCDIC などの変換で使います。テーブル 2 は、EBCDIC から ASCII などの反対(逆)の変換で使います。変換テーブルの半分は 2 桁の位置が 256 個含まれる配列で、それぞれの位置に、コーディングされた文字に対応する 1 バイトの符号なし数値があります。

SAS では次の目的で変換テーブルを使います。

- SORT プロシジャで照合順序を判断するため
- 動作環境とグラフィックデバイス間のデータ通信のため(IBM 環境で SAS/GRAPH ソフトウェアを実行する場合)
- U.S. English 以外の国際言語文字セットに対応するため

PROC TRANTAB プロシジャを実行しても、出力はありません。SAS ログの変換テーブルとメモを表示できます。

注: 変換テーブルは各国語の要件をサポートするために SAS 6 で導入されました。SAS 8.2 では、変換テーブルの直接使用を改善するために、LOCALE=システムオプションが導入されました。SAS 9.2 では、TRANTAB プロシジャが下位互換性のためにサポートされています。ただし、SAS 8.2 以降のリリースでは、LOCALE=システムオプションを使用することをお勧めします。PROC TRANTAB は対話型プロシジャです。PROC TRANTAB ステートメントをサブミットすると、PROC TRANTAB ステートメントを繰り返すことなく、引き続きステートメントを入力して実行できます。プロシジャを終了するには、QUIT ステートメントをサブミットするか、別の DATA または PROC ステートメントをサブミットします。

概念: TRANTAB プロシジャ

PROC TRANTAB で使用する変換テーブルと文字セットについて

変換テーブルの K 番目の要素は、順序どおりに並べた文字セットの K 番目の要素に対応します。たとえば、変換テーブルの位置 00 (1 バイト目)には、コード化された値が含まれ、その値は、順序どおりに並べた文字セットの最初の要素に対応します。動作環境の文字セットに含まれる文字の位置を特定するには、RANK 関数を使用します。次の例に、RANK の使用方法を示します。

```
data _null_;  
  x=rank('a');  
  put "The position of a is " x ".";  
run;
```

SAS ログには、次のメッセージが出力されます。The position of a is 97.

変換テーブルの各位置には、0 ('00'x)から 255 ('FF'x)の範囲内にある 16 進数が含まれます。16 進値の末尾には常に x が付きます。連続する 1 つ以上の 16 進値を表すには、それらの値をまとめて引用符で囲み、その後に x を 1 つ付けます。たとえば、連続する 3 個の 16 進値からなる文字列は、'08090A'x と表すことができます。SAS ログには、変換テーブルの各行が、引用符で囲まれて、後ろに x が付けられた 16 進値で表示されます。SAS ログではまた、テーブル内の位置に対応する縦方向と横方向の余白に参照番号が表示されます。[“例 1: 変換テーブルの表示” \(1079 ページ\)](#)に、SAS ログに変換テーブルが表示されている様子を示します。

PROC TRANTAB を使用した変換テーブルの保存

PROC TRANTAB を使用してカスタマイズした変換テーブルを作成すると、プロシジャが自動的にテーブルを Sasuser.Profile カタログに保存します。これにより、他のユーザーに影響を与えることなくカスタマイズした変換テーブルを使用できます。SORT プロシジャまたは GOPTIONS ステートメントで変換テーブルを指定すると、最初に Sasuser.Profile カタログでテーブルが検索されます。指定した変換テーブルが Sasuser.Profile カタログにない場合、次に Sashelp.Host カタログが検索されます。

作成した変換テーブルにグローバルにアクセスするには、SAS インストール担当者がこのテーブルを Sasuser.Profile カタログから Sashelp.Host カタログに (CATALOG プロシジャを使用して)コピーする必要があります。Sashelp.Host カタ

ログに変換テーブルが見つからない場合、続いて Sashelp.Locale でテーブルが検索されます。

PROC TRANTAB を使用した SAS 変換テーブルの変更

SAS 提供の変換テーブルがニーズに合わない場合、PROC TRANTAB を使用してテーブルを編集したり、新しいテーブルを作成したりできます。つまり、SAS テーブルを指定する PROC TRANTAB ステートメントを発行し、テーブルを編集してから、SAVE ステートメントを使用してそのテーブルを保存できます。変更した変換テーブルは、Sasuser.Profile カタログに保存されます。SAS インストール担当者は、PROC TRANTAB で変換テーブルを変更し、CATALOG プロシジャを用いて、Sasuser.Profile カタログから Sashelp.Host カタログに、変更済みのテーブルをコピーできます。その例を次に示します。

```
proc catalog c=sasuser.profile;  
  copy out=sashelp.host entrytype=trantab;  
run;
```

PROC TRANTAB を使用して Sashelp.Host カタログに保存された変換テーブルを変更するには、そのデータライブラリとカタログへの更新(書き込み)アクセス権が必要です。

PROC TRANTAB 外での変換テーブルの使用

SORT プロシジャでの変換テーブルの使用

PROC SORT は、変換テーブルを使用して並べ替えの照合順序を決定します。代替の変換テーブルは、PROC SORT の SORTSEQ=オプションで指定できます。たとえば、動作環境のデフォルトが EBCDIC 順による並べ替えで、ASCII 順に並べ替えようとする場合、ASCII 変換テーブルを指定するために次のステートメントを発行します。

```
proc sort sortseq=ascii;  
run;
```

PROC TRANTAB でカスタマイズした変換テーブルを作成し、PROC SORT で新しいテーブルを指定することもできます。このテーブルは、英語(U.S.)以外の言語に並べ替え順序を指定する場合に便利です。

変換テーブルを使って違う方法でデータを並べ替える例については、[“例 6: 並べ替えへのさまざまな変換テーブルの使用” \(1089 ページ\)](#)を参照してください。並べ替えに使用できるテーブルと SORTSEQ=オプションの詳細については、[“SORTSEQ= システムオプション” \(984 ページ\)](#)を参照してください。

外部ファイルのトランスコード

変換テーブルは、LOCALE=システムオプションによって暗黙的に設定されます。これらは外部ファイルのトランスコーディングにのみ使用されます。

変換テーブルは OPTIONS プロシジャを使用して表示できます。次の例では、LOCALE=システムオプションが en_US に設定されている z/OS 環境を使用しています。

```
TRANTAB=(eollwlt1,wlt1eoll,elat_ucs,elat_lcs,elat_ccl,,,elat_scc)
```

SAS/GRAPH での変換テーブルの使用

SAS/GRAPH ソフトウェアの場合、変換テーブルが最もよく使われるのは IBM の動作環境です。IBM の動作環境はグラフィックコマンドを EBCDIC で出力しますが、これを非同期グラフィックデバイスに ASCII 表現で届けなければならないので、テーブルは必須です。厳密にいうと、SAS/GRAPH ソフトウェアはこのようなデバイス向けのコマンドストリームを、内部的には ASCII 表現で組み立てます。しかし、コマンドを通信ソフトウェアに渡してデバイスに送信できるように、EBCDIC 表現に変換する必要があります。SAS/GRAPH は、内部的に変換テーブルを使用して、ASCII から EBCDIC への初期変換を行います。通信ソフトウェアは、コマンドストリームを ASCII 表現に逆変換した上で、グラフィックデバイスにアクセスします。

変換テーブルは動作環境によって異なります。多くの場合、デフォルトの変換テーブルである SASGTAB0 か、SAS が提供するいずれかのグラフィック変換テーブルをそのまま使用できます。ただし、これらのテーブルでは正しく変換できないものがある場合、PROC TRANTAB を使用して独自の変換テーブルを作成できます。SASGTAB0 テーブルは、英語(U.S.)以外の言語の文字が検出される場合に正しく変換できないことがあります。

SAS/GRAPH に代替変換テーブルを指定するには、GOPTIONS ステートメントで TRANTAB=オプションを使用するか、デバイスエントリの TRANTAB デバイスパラメーターを変更します。たとえば次の GOPTIONS ステートメントは、GTABTCAM グラフィック変換テーブルを指定します。

```
goptions trantab=gtabtcam;
```

SAS/GRAPH で使用される変換テーブルは、デバイス-動作環境間 変換および動作環境-デバイス間変換の両方を実行します。したがって、変換テーブルは 512 バイトで構成され、最初の 256 バイトはデバイス-動作環境間変換(IBM メインフレームでの ASCII から EBCDIC)に、次の 256 バイトは動作環境-デバイス間変換(IBM メインフレームでの EBCDIC から ASCII)に使用されます。PROC TRANTAB では、デバイス-動作環境間変換の変換テーブルの領域はテーブル 1 と見なされ、動作環境-デバイス間変換の領域はテーブル 2 と見なされます。テーブルの両方の領域を示す ASCII 変換テーブル(SAS 提供の変換テーブル)のリストについては、[“例 1: 変換テーブルの表示” \(1079 ページ\)](#)を参照してください。

IBM メインフレーム以外の動作環境では、変換テーブルを使用してドライバーが作成したデータストリーム内の特定の文字を変換できます。たとえば、ドライバーによって通常データストリーム内に縦棒が生成される場合、その縦棒のかわりに別の文字を生成するには、縦棒を別の文字に変換する変換テーブルを作成できます。

SAS/GRAPH は、キーボードで生成されたコードを指定された文字にマッピングする場合、および文字コードをグラフィック出力デバイスで要求されるコードにマッピングする場合に、キーマップおよびデバイスマップも使用します。これらのマップは SAS/GRAPH に固有です。詳細については、SAS のテクニカルサポートにお問い合わせください。

構文: TRANTAB プロシジャ

ヒント: RUN グループ処理がサポートされます。

```
PROC TRANTAB TABLE=table-name <NLS>;
  CLEAR <ONE | TWO | BOTH>;
  INVERSE;
  LIST <ONE | TWO | BOTH>;
  LOAD TABLE=table-name <NLS>;
  REPLACE position value-1 <...value-n>;
  SAVE <TABLE=table-name> <CAT=libname.catalog> <ONE | TWO | BOTH>;
  SWAP;
```

ステートメント	タスク
"PROC TRANTAB ステートメント"	変換テーブルを作成、編集、表示します
"CLEAR ステートメント"	変換テーブルのすべての位置をゼロに設定します
"INVERSE ステートメント"	テーブル 1 の逆変換テーブルを作成します
"LIST ステートメント"	変換テーブルを 16 進表現で表示します
"LOAD ステートメント"	変換テーブルを編集できるようにメモリにロードします
"REPLACE ステートメント"	変換テーブル内の文字を指定値で置換します
"SAVE ステートメント"	変換テーブルを Sasuser.Profile カタログに保存します
"SWAP ステートメント"	テーブル 1 をテーブル 2 と交換します

PROC TRANTAB ステートメント

変換テーブルを作成、編集、表示します。

ヒント: PROC TRANTAB ステートメントに誤ったテーブル名がある場合は、LOAD ステートメントを使用して正しいテーブルを読み込みます。PROC TRANTAB を再起動する必要はありません。新しいテーブルは、SAVE ステートメントを発行するまでカタログに保存されません。そのため、不要なテーブルがカタログに保存されることはありません。

構文

```
PROC TRANTAB TABLE=table-name <NLS>;
```

必須引数

TABLE=*table-name*

テーブルを作成、編集、表示するための変換テーブルを指定します。指定されたテーブル名は、8 文字以内の有効な 1 レベルの SAS 名である必要があります。

オプション引数

NLS

TABLE=引数で指定したテーブルが、SAS で提供される 5 つの特殊な内部変換テーブルの 1 つであることを指定します。TABLE=引数に 5 つの特殊なテーブルのいずれかを指定する場合は、NLS オプションを使用する必要があります。NLS は各国語サポート(National Language Support)を表します。このオプションおよび関連する変換テーブルによって、米国英語以外の言語の文字を変換する手段が提供されます。変更した NLS テーブルを使用するには、SAS システムオプションの TRANTAB=にその名前を指定します。これらの特殊な変換テーブルのいずれかを読み込む場合、SAS ログには、テーブル 2 が初期化されていないという注意が表示されます。つまり、テーブル 2 はゼロのみが含まれた空のテーブルです。PROC TRANTAB が、これらの特殊なケースで変換にテーブル 2 を使用することはありません。

SASXPT

ローカルからトランスポートへの形式変換テーブル(CPORT プロシジャで使用)を指定します。

SASLCL

トランスポートからローカルへの形式変換テーブル(CIMPORT プロシジャで使用)を指定します。

SASUCS

小文字から大文字への変換テーブル(UPCASE 関数で使用)を指定します。

SASLCS

大文字から小文字への変換テーブル(LOWCASE マクロで使用)を指定します。

SASCCL

文字分類テーブル(内部でのみ使用)を指定します。このテーブルには、各文字位置に対応するフラグバイトが含まれています。これらの位置は、各文字が所属しているクラス(複数可)を表しています。

CLEAR ステートメント

変換テーブルのすべての位置をゼロに設定します。新しいテーブルを作成するときに使用されます。

構文

CLEAR <ONE | TWO | BOTH>;

オプション引数

ONE | TWO | BOTH

ONE

テーブル 1 をクリアします。

TWO

テーブル 2 をクリアします。

BOTH

テーブル 1 とテーブル 2 を両方ともクリアします。

デフォルト ONE

INVERSE ステートメント

変換テーブル内にテーブル 1 の逆変換テーブルを作成します。INVERSE ステートメントはテーブル 2 を作成します。

構文

INVERSE;

詳細

INVERSE ステートメントでは、複数の変換は保持されません。テーブル 1 に、同じ値に変換される複数の文字があるとします。たとえば、"A"と"B"がどちらも"1"に変換される状況です。テーブル 2 に対して、INVERSE ステートメントは最後に変換された文字を値として使用します。つまり、最初のテーブルでは"A"が"B"より前に現れるという前提のもと、"1"は常に"B"に変換され、"A"には変換されません。並べ替えプログラムが正しく動作するには、逆変換テーブルが必要です。

LIST ステートメント

変換テーブルを SAS ログに 16 進表現で表示します。

構文

```
LIST <ONE | TWO | BOTH>;
```

オプション引数

ONE | TWO | BOTH

ONE

テーブル 1 を表示します。

TWO

テーブル 2 を表示します。

BOTH

テーブル 1 とテーブル 2 を両方とも表示します。

デフォルト ONE

LOAD ステートメント

変換テーブルを編集できるようにメモリに読み込みます。

ヒント:

LOAD ステートメントは、PROC TRANTAB ステートメントに誤ったテーブル名を指定した場合に使用します。プロシジャを再起動せずに、正しい名前を指定できます。1 つの PROC TRANTAB ステップで複数の変換テーブルを編集するには、LOAD ステートメントを使用します(必ず最初のテーブルを保存してから、別のテーブルを読み込んでください)。

構文

LOAD *TABLE=table-name* <NLS>;

必須引数

TABLE=*table-name*

編集する既存の変換テーブルの名前を指定します。指定されたテーブル名は、有効な 1 レベルの SAS 名である必要があります。

オプション引数

NLS

TABLE=引数で指定したテーブルが、SAS で提供される 5 つの特殊な内部変換テーブルの 1 つであることを指定します。TABLE=引数に 5 つの特殊なテーブルのいずれかを指定する場合は、NLS オプションを使用する必要があります。

SASXPT

ローカルからトランスポートへの形式変換テーブルを指定します。

SASLCL

トランスポートからローカルへの形式変換テーブルを指定します。

SASUCS

小文字から大文字への変換テーブルを指定します。

SASLCS

大文字から小文字への変換テーブルを指定します。

SASCCL

文字分類テーブルを指定します。このテーブルには、各文字位置に対応するフラグバイトが含まれています。これらの位置は、各文字が所属しているクラス(複数可)を表しています。

NLS は各国語サポート(National Language Support)を表します。このオプションおよび関連する変換テーブルによって、英語以外の言語の文字をプログラム、表示およびファイルにマップする手段が提供されます。これらの特殊な変換テーブルのいずれかを読み込む場合、SAS ログには、テーブル 2 が初期化されていないという注意が表示されます。つまり、テーブル 2 はゼロのみが含まれた空のテーブルです。PROC TRANTAB が、これらの特殊なケースで変換にテーブル 2 を使用することはありません。

REPLACE ステートメント

指定された開始位置から変換テーブル内の文字を指定値で置き換えます。

別名: REP

ヒント: 編集内容を保存するには、SAVE ステートメントを発行する必要があります。

構文

REPLACE *position value-1* < ...*value-n*>;

必須引数

position

変換テーブル内の置換の開始位置を指定します。変換テーブル内の編集可能な位置は、10 進値の 0 から 255 までです。位置は次のいずれかの方法で指定します。

- 10 進値または 16 進値で位置そのものを指定する方法。10 進値(20 など)を指定すると、PROC TRANTAB は、テーブル内の位置 20 (21 バイト目)を検索します。16 進値(たとえば'14'x)で指定した場合、PROC TRANTAB はこれと同等の 10 進値で表される位置を検索します。この例の場合、テーブル内の位置 20(21 バイト目)に当たります。
- 引用符付きの文字で指定する方法。PROC TRANTAB は、テーブル内の引用符付きの文字(つまり、引用符で囲まれた文字の 16 進値)を検索し、その文字の位置を開始位置として使用します。たとえば、次の REPLACE ステートメントを指定すれば、"a"に相当する 16 進値が最初に現れた箇所、およびこれに続く 2 つの 16 進値を、"ABC"に相当する 16 進値に置き換えることになります。
replace 'a' 'ABC';

このアクションが有用なのは、英字や数字を検索したいけれども、実際の位置は分からない、という場合です。指定された文字が見つからなかった場合、PROC TRANTAB はエラーメッセージを表示し、ステートメントを無視します。

位置 256 - 511(テーブル 2)の編集は以下の手順で行います。

- 1 SWAP ステートメントを発行します。
- 2 適切な REPLACE ステートメントを発行します。
- 3 SWAP ステートメントをもう一度発行し、改めてテーブルの位置決めをする必要があります。

value-1 < ...*value-n*>

テーブル内の *position* から配置する実際の値を、1 つ以上の 10 進、16 進、または文字定数で指定します。これらの値を組み合わせて使用することもできます。つまり、10 進値、16 進値および文字値を 1 つの REPLACE ステートメントに指定できます。[“例 3: 開始位置に 10 進値を指定した編集” \(1083 ページ\)](#)に、REPLACE ステートメントに 3 種類すべての値を指定している様子を示します。

SAVE ステートメント

変換テーブルを Sasuser.Profile カタログに保存します。

構文

SAVE <TABLE=*table-name*> <CAT=*libname.catalog*> <ONE | TWO | BOTH>;

オプション引数

TABLE=*table-name*

現在のテーブルを保存するときの名前を指定します。テーブル名は、有効な 1 レベルの SAS 名である必要があります。

デフォルト TABLE=オプションを省略すると、現在のテーブルは PROC TRANTAB ステートメントまたは LOAD ステートメントで指定した名前で作保存されます。

CAT=*libname.catalog*

TRANTAB エントリを保存する際に出カカタログの名前を指定します。

ONE | TWO | BOTH

ONE

テーブル 1 を保存します。

TWO

テーブル 2 を保存します。

BOTH

テーブル 1 とテーブル 2 を両方とも保存します。

デフォルト BOTH

SWAP ステートメント

テーブル 1 をテーブル 2 と交換して、位置 256 から 511 までを編集できるようにします。

ヒント: テーブルの編集後、SWAP ステートメントをもう一度発行し、改めてテーブルの位置決めをする必要があります。

構文

SWAP;

例: TRANTAB プロシジャ

例 1: 変換テーブルの表示

要素: LIST ステートメント

この例では、PROC TRANTAB を使用して SAS が提供する ASCII 変換テーブルを表示します。例はすべて Linux 環境で作成されました。

オプションを設定し、変換テーブルを指定します。

```
options nodate pageno=1 linesize=80 pagesize=60;  
proc trantab table=ascii;
```

変換テーブルの半分を両方とも表示します。 LIST BOTH ステートメントでは、変換を行うテーブルと逆変換を行うテーブルの両方を表示します。

```
list both;
```

SAS ログ

NOTE: Table specified is ASCII.

ASCII table 1:

```

0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '606162636465666768696A6B6C6D6E6F'x
70 '707172737475767778797A7B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAFAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

```

ASCII table 2:

```

0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '606162636465666768696A6B6C6D6E6F'x
70 '707172737475767778797A7B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAFAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

```

例 2: 変換テーブルの作成

要素: プロシジャの機能:
 LIST ステートメント
 REPLACE ステートメント
 SAVE ステートメント

この例では、PROC TRANTAB を使用してカスタマイズした変換テーブルを作成します。例はすべて Linux 環境で作成されました。

システムオプションを設定し、編集する変換テーブルを指定します。

```
options nodate pageno=1 linesize=80
pagesize=60;
proc trantab table=newtable;
```

変換テーブル内の文字を指定された開始位置から置き換えます。 この REPLACE ステートメントでは、テーブル内の開始位置 0 から値を配置します。REPLACE ステートメントには、任意の長さの 16 進値を使用できます。この例では、SAS ログに表示されたときの変換テーブルと同じ状態にするために長さ 16 の文字列を使用します。

```
replace 0
'00010203a309e57ff9ecc40b0c0d0e0f'x
'10111213a5e008e71819c6c51c1d1e1f'x
'c7fce9e2e40a171beaebe8efee050607'x
'c9e616f4f6f2fb04ffd6dca2b6a7501a'x
'20e1edf3faf1d1aababfa22e3c282b7c'x
'265facbdbca1abbb5f5f21242a293bac'x
'2d2f5fa6a6a6a62b2ba6a62c255f3e3f'x
'a62b2b2b2b2b2b2d2d603a2340273d22'x
'2b6162636465666768692d2ba6a62b2b'x
'2d6a6b6c6d6e6f7071722da62d2b2d2d'x
'2d7e737475767778787a2d2b2b2b2b2b'x
'2b2b2b5f5fa65f5f5fd5fb65f5fb55f'x
'7b4142434445464748495f5f5f5f5f'x
'7d4a4b4c4d4e4f5051525f5f5fb15f5f'x
'5c83535455565758595a5f5ff75f5fb0'x
'30313233343536373839b75f6eb25f5f'x
;
```

テーブルを保存します。 SAVE ステートメントでは、テーブルを PROC TRANTAB ステートメントで指定された名前で保存します。デフォルトでは、テーブルは Sasuser.Profile カタログに保存されます。

```
save;
```

SAS ログ内に変換テーブルの半分を両方とも表示します。 LIST BOTH ステートメントでは、変換を行うテーブルと逆変換を行うテーブルの両方を表示します。

```
list both;
```

SAS ログ

```
-->Create
and edit table 2. table 2 is
empty; that is, it
consists entirely of 0s.To create table 2, you can use the INVERSE statement.
(See: SAS(R) Statements: Reference)
```

```
)To edit table 2, you can use the SWAP statement with the REPLACE statement.
(See: SAS(R) Statements: Reference)
```

```
.)" commented-out by old2new conversion -->
```

```
NOTE: Table specified is NEWTABLE.
WARNING: Table NEWTABLE not found! New table is assumed.
NOTE: NEWTABLE table 1 is uninitialized.
NOTE: NEWTABLE table 2 is uninitialized.
```

```
NOTE: Saving table NEWTABLE.
NOTE: NEWTABLE table 2 will not be saved because it is uninitialized.
NEWTABLE table 1:
```

```
  0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '00010203A309E57FF9ECC40B0C0D0E0F'x
10 '10111213A5E008E71819C6C51C1D1E1F'x
20 'C7FCE9E2E40A171BEAEBE8EFEE050607'x
30 'C9E616F4F6F2FB04FFD6DCA2B6A7501A'x
40 '20E1EDF3FAF1D1AABABFA22E3C282B7C'x
50 '265FACBDBCA1ABBB5F5F21242A293BAC'x
60 '2D2F5FA6A6A6A62B2BA6A62C255F3E3F'x
70 'A62B2B2B2B2B2B2D2D603A2340273D22'x
80 '2B6162636465666768692D2BA6A62B2B'x
90 '2D6A6B6C6D6E6F7071722DA62D2B2D2D'x
A0 '2D7E737475767778787A2D2B2B2B2B'x
B0 '2B2B2B5F5FA65F5FDF5FB65F5FB55F'x
C0 '7B4142434445464748495F5F5F5F5F'x
D0 '7D4A4B4C4D4E4F5051525F5F5FB15F5F'x
E0 '5C83535455565758595A5F5FF75F5FB0'x
F0 '30313233343536373839B75F6EB25F5F'x
```

```
NOTE: NEWTABLE table 2 is uninitialized.
NEWTABLE table 2:
```

```
  0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000000000000000000000000000000'x
10 '000000000000000000000000000000'x
20 '000000000000000000000000000000'x
30 '000000000000000000000000000000'x
40 '000000000000000000000000000000'x
50 '000000000000000000000000000000'x
60 '000000000000000000000000000000'x
70 '000000000000000000000000000000'x
80 '000000000000000000000000000000'x
90 '000000000000000000000000000000'x
A0 '000000000000000000000000000000'x
B0 '000000000000000000000000000000'x
C0 '000000000000000000000000000000'x
D0 '000000000000000000000000000000'x
E0 '000000000000000000000000000000'x
F0 '000000000000000000000000000000'x
```

例 3: 開始位置に 10 進値を指定した編集

要素: LIST ステートメント
 REPLACE ステートメント
 SAVE ステートメント

この例では、“[例 2: 変換テーブルの作成](#)” (1080 ページ) で作成した変換テーブルを編集します。REPLACE ステートメントで指定する 10 進値は、テーブル変更の開始位置を表します。

両方の SAS ログ内の縦矢印によって、変更の開始点が表示されます。例はすべて Linux 環境で作成されました。

プログラム 1: 元のテーブルの表示

```
options nodate pageno=1 linesize=80 pagesize=60; proc trantab table=newtable;  
list one;
```

プログラムの説明

システムオプションを設定し、編集する変換テーブルを指定します。

```
options nodate pageno=1 linesize=80 pagesize=60; proc trantab table=newtable;
```

元のテーブルを表示します。LIST ステートメントは元の NEWTABLE 変換テーブルを表示します。

```
list one;
```

SAS ログ

元の NEWTABLE 変換テーブル

```

Table specified is NEWTABLE.
NOTE: NEWTABLE table 2 is uninitialized.
NEWTABLE table 1:
      ↓
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '00010203A309E57FF9ECC40B0C0D0E0F'x
10 '10111213A5E008E71819C6C51C1D1E1F'x
20 'C7FCE9E2E40A171BEAEBE8EFEE050607'x
30 'C9E616F4F6F2FB04FFD6DCA2B6A7501A'x
40 '20E1EDF3FAF1D1AABABFA22E3C282B7C'x
50 '265FACBDBC1A1BBB5F5F21242A293BAC'x
60 '2D2F5FA6A6A6A62B2BA6A62C255F3E3F'x
70 'A62B2B2B2B2B2B2D2D603A2340273D22'x
80 '2B6162636465666768692D2BA6A62B2B'x
90 '2D6A6B6C6D6E6F7071722DA62D2B2D2D'x
A0 '2D7E737475767778787A2D2B2B2B2B2B'x
B0 '2B2B2B5F5FA65F5F5FDF5FB65F5FB55F'x
C0 '7B4142434445464748495F5F5F5F5F5F'x
D0 '7D4A4B4C4D4E4F5051525F5F5FB15F5F'x
E0 '5C83535455565758595A5F5FF75F5FB0'x
F0 '30313233343536373839B75F6EB25F5F'x

```

プログラム 2: テーブルの編集

```

replace 10
20 10 200 'x' 'ux' '092040'x;

save;

list one;

```

プログラムの説明

変換テーブル内の文字を指定された開始位置から置き換えます。この REPLACE ステートメントでは、10 進数で 10 の開始位置(元のテーブルの 11 バイト目)から、指定した値を基にバイト単位の置換を行います。

```

replace 10
20 10 200 'x' 'ux' '092040'x;

```

変更を保存します。SAVE ステートメントでは、NEWTABLE 変換テーブルに加えた変更を保存します。

```

save;

```

新しいテーブルを表示します。2 番目の LIST ステートメントでは、編集された NEWTABLE 変換テーブルを表示します。

```

list one;

```

SAS ログ

```

Saving table NEWTABLE.
NOTE: NEWTABLE table 2 will not be saved because it is uninitialized.
NEWTABLE table 1:
      ↓
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '00010203A309E57FF9EC140AC8787578'x
10 '09204013A5E008E71819C6C51C1D1E1F'x
20 'C7FCE9E2E40A171BEAEBE8EFEE050607'x
30 'C9E616F4F6F2FB04FFD6DCA2B6A7501A'x
40 '20E1EDF3FAF1D1AABABFA22E3C282B7C'x
50 '265FACBDBCA1ABBB5F5F21242A293BAC'x
60 '2D2F5FA6A6A62B2BA6A62C255F3E3F'x
70 'A62B2B2B2B2B2B2D2D603A2340273D22'x
80 '2B6162636465666768692D2BA6A62B2B'x
90 '2D6A6B6C6D6E6F7071722DA62D2B2D2D'x
A0 '2D7E737475767778787A2D2B2B2B2B'x
B0 '2B2B2B5F5FA65F5F5FDF5FB65F5FB55F'x
C0 '7B4142434445464748495F5F5F5F5F'x
D0 '7D4A4B4C4D4E4F5051525F5F5FB15F5F'x
E0 '5C83535455565758595A5F5FF75F5FB0'x
F0 '30313233343536373839B75F6EB25F5F'x

```

出力詳細

位置 10 (11 バイト目)では、縦矢印が変換テーブルへの変更の開始位置を示しています。

11 バイト目では、10 進値の 20 (16 進値の 14)で 16 進値の C4 が置き換えられます。

12 バイト目では、10 進値の 10 (16 進値の 0A)で 16 進値の 0B が置き換えられます。

13 バイト目では、10 進値の 200 (16 進値の C8)で 16 進値の 0C が置き換えられます。

14 バイト目では、文字'x' (16 進値の 78)で 16 進値の 0D が置き換えられます。

15 バイト目、16 バイト目では、文字'ux'(16 進値はそれぞれ 75 と 78)で 16 進値 0E および 0F が置き換えられます。

17 バイト目、18 バイト目および 19 バイト目では、16 進値の 092040 で 16 進値の 101112 が置き換えられます。

例 4: 開始位置に引用符付き文字を使用した編集

要素:

- LIST ステートメント
- LOAD ステートメント
- REPLACE ステートメント
- SAVE ステートメント

この例では、ASCII 変換テーブルを編集して新しい変換テーブルを作成します。REPLACE ステートメントで指定した引用符付き文字と等しい 16 進値が最初に出現する位置が、テーブルへの変更開始位置になります。この方法は、テーブルへの変更を開始する正確な位置を知る必要がないという点が、“[例 3: 開始位置に 10 進値を指定した編集 \(1083 ページ\)](#)”とは異なります。PROC TRANTAB が、ユーザーに代わって正確な位置を検索します。

編集されたテーブルは、新しい名前で作成されます。両方の SAS ログ内の横矢印は、変換テーブル内の編集された行を示します。例はすべて Linux 環境で作成されました。

プログラム 1: 元のテーブルの表示

```
options nodate pageno=1 linesize=80 pagesize=60;
proc trantab table=ascii;

list one;
```

プログラムの説明

システムオプションを設定し、編集する変換テーブルを指定します。

```
options nodate pageno=1 linesize=80 pagesize=60;
proc trantab table=ascii;
```

変換テーブルを表示します。この LIST ステートメントでは、SAS ログに元の変換テーブルを表示します。

```
list one;
```

SAS ログ

```
NOTE: Table specified is ASCII.
ASCII table 1:
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '606162636465666768696A6B6C6D6E6F'x ←
70 '707172737475767778797A7B7C7D7E7F'x ←
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAFAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFFF'x
```

プログラム 2: テーブルの編集

```
replace 'a' 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
save table=upper;
load table=upper;
list one;
```

プログラムの説明

変換テーブル内の文字を指定された開始位置から置き換えます。REPLACE ステートメントは、16 進値"a"(値は 61)が最初に現れる箇所を見つけ、これと続く 25 個の 16 進値を、大文字"A"から"Z"に当たる 16 進値で置き換えます。

```
replace 'a' 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
```

変更を保存します。この SAVE ステートメントでは、ASCII 変換テーブルに加えた変更を UPPER という新しいテーブル名で保存します。ASCII 変換テーブルに保存された内容は、変更されずに残ります。

```
save table=upper;
```

変換テーブルを読み込んで表示します。この LOAD ステートメントでは、編集された変換テーブル UPPER を読み込みます。この LIST ステートメントでは、SAS ログに変換テーブル UPPER を表示します

```
load table=upper;
list one;
```

SAS ログ

```
NOTE: Table UPPER
being loaded.
UPPER table 1:
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '604142434445464748494A4B4C4D4E4F'x ←
70 '505152535455565758595A7B7C7D7E7F'x ←
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBBCDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x
```

例 5: 逆変換テーブルの作成

要素: INVERSE ステートメント
 LIST ステートメント
 SAVE ステートメント

この例では、“[例 4: 開始位置に引用符付き文字を使用した編集](#)” (1085 ページ) で作成した変換テーブルの逆変換テーブルを作成します。この例で作成した新しい変換テーブルは、データ通信の際、動作環境からデバイスへの変換のために使います。

```
options nodate pageno=1 linesize=80 pagesize=60;  
proc trantab table=upper;
```

逆変換テーブルを作成し、各テーブルを保存して表示します。この INVERSE ステートメントでは、元のテーブル 1 (UPPER) を逆にしてテーブル 2 を作成します。この SAVE ステートメントでは、変換テーブルを保存します。この LIST BOTH ステートメントでは、元の変換テーブルと逆変換テーブルの両方を表示します。

```
inverse;  
save;  
list both;
```

SAS ログ

INVERSE ステートメントでは、変換テーブルを逆転させるときに検出した複数の変換をすべて SAS ログに表示します。“[例 4: 開始位置に引用符付き文字を使用した編集](#)” (1085 ページ) において、すべての小文字が変換テーブル UPPER で大文字に変換されています。これは UPPER に大文字が 2 セットあることを意味しています。INVERSE で変換できなかった場合、PROC TRANTAB は値を 00 で埋めます。変換テーブル UPPER の逆変換テーブルには、00 の値が数多く含まれていることに注意してください。

SAS ログには、テーブル 1 の逆変換テーブルを作成するときに検出された重複値がすべて表示されます。スペースを節約するために、この例ではこのようなメッセージの多くは削除されています。

```

NOTE: This table cannot be mapped one to one.
duplicate of '41'x found at '61'x in table one.
duplicate of '42'x found at '62'x in table one.
duplicate of '43'x found at '63'x in table one.
.
.
.
duplicate of '58'x found at '78'x in table one.
duplicate of '59'x found at '79'x in table one.
duplicate of '5A'x found at '7A'x in table one.
NOTE: Saving table UPPER.
UPPER table 1:
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '604142434445464748494A4B4C4D4E4F'x
70 '505152535455565758595A7B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

UPPER table 2:
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '600000000000000000000000000000'x
70 '00000000000000000000000007B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

```

例 6: 並べ替えへのさまざまな変換テーブルの使用

要素: PROC SORT ステートメントオプション: SORTSEQ=
PRINT プロシジャ

この例では、デフォルトの並べ替え順序とは異なる順序でデータを並べ替えるために、異なる変換テーブルを指定する方法を示します。英語(U.S.)以外の言語で記述さ

れた文字は、デフォルトの順序とは異なる並べ替え順序が必要になる場合があります。

SAS サンプルライブラリにある TRABASE プログラムを使用すると、複数言語用の変換テーブルを作成できます。例はすべて Linux 環境で作成されました。

SAS システムオプションを設定します。

```
options nodate pageno=1 linesize=80 pagesize=60;
```

TESTSORT データセットを作成します。 DATA ステップで、4 組の単語のペアを使用して SAS データセットを作成します。各ペアの単語は最初の文字が大文字か小文字かという点のみが異なります。

```
data testsort;
  input Values $10.;
  datalines;
Always
always
Forever
forever
Later
later
Yesterday
yesterday
;
```

デフォルトの並べ替え順序とは異なる順序でデータを並べ替えます。 PROC SORT はデフォルトの変換テーブルを使ってデータを並べ替えます。小文字のみからなる語を先に、大文字のみからなる語を後にする並べ替えです。

```
proc sort;
  by values;
run;
```

データセットを出力します。 PROC PRINT では、並べ替えたデータセットが出力されます。

```
proc print noobs;
  title 'Default Sort Sequence';
run;
```

SAS 出力

次の出力はデフォルトの変換テーブルを使用した値の並べ替えの出力です。デフォルトの並べ替え順序を使用すると、最初にアルファベット順で大文字の単語すべて、次に小文字の単語すべてを並べます。


```
Default Sort Sequence
1
```

Values

Always
Forever
Later
Yesterday
always
forever
later
yesterday

変換テーブル UPPER に従ってデータを並べ替え、新しいデータセットを出力します。SORTSEQ=システムオプションは、PROC SORT がカスタマイズされた変換テーブル UPPER に従ってデータを並べ替えるよう指定します。小文字と大文字を区別せずに扱う変換テーブルです。この方法は、大文字か小文字かを区別せずに並べ替えたい場合に有用です。PROC PRINT では、並べ替えたデータセットが出力されます。

```
proc sort sortseq=upper;
  by values;
run;
proc print noobs;
  title 'Customized Sort Sequence';
run;
```

SAS 出力

次の出力はカスタマイズされた変換テーブルを使用した値の並べ替えの結果です。このカスタマイズされた並べ替えでは、先頭文字が大文字か小文字かにかかわらず、すべての語をアルファベット順に並べ替えます。

Customized Sort Sequence

2

Values

Always
always
Forever
forever
Later
later
Yesterday
yesterday

例 7: テーブル 1 とテーブル 2 の編集

要素: LIST ステートメント

REPLACE ステートメント

SAVE ステートメント

SWAP ステートメント

この例では、変換テーブルの両方の領域を編集する方法を示します。位置 256～511(テーブル 2)の編集は以下の手順で行います。

- SWAP ステートメントを発行して、テーブル 2 とテーブル 1 の位置を交換します。
- 適切な REPLACE ステートメントでテーブル 2 に変更を施します。
- SWAP ステートメントをもう一度発行し、改めてテーブルの位置決めをする必要があります。
- SAS ログ内の矢印によって、変更された列と行が示されます。

SAS システムオプションを設定し、変換テーブルを指定します。

```
options nodate pageno=1 linesize=80 pagesize=60;  
proc trantab table=upper;
```

元の変換テーブルを表示します。この LIST ステートメントでは、元の UPPER 変換テーブルを表示します。

```
list both;
```

SAS ログ

次の出力は、元の UPPER 変換テーブルです。

```

NOTE: Table specified is UPPER.
UPPER table 1:
    ↓
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x ←
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '604142434445464748494A4B4C4D4E4F'x
70 '505152535455565758595A7B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

UPPER table 2:
    ↓
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000102030405060708090A0B0C0D0E0F'x ←
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '60000000000000000000000000000000'x
70 '0000000000000000000000007B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

```

変換テーブル内の文字を指定された開始位置から置き換えます。REPLACE ステートメントは開始位置 1 から開始し、01 の現在の値を'0A'で置き換えます。

```
replace 1 '0A'x;
```

テーブル 2 を編集用に準備します。最初の SWAP ステートメントでは、テーブル 2 を編集できるように配置します。2 番目の REPLACE ステートメントでは、テーブル 1 と同じ変更をテーブル 2 に加えます。

```
swap;
replace 1 '0A'x;
```

元の位置にテーブルを保存して表示します。2 番目の SWAP ステートメントでは、テーブル 1 とテーブル 2 を元の位置に戻します。SAVE ステートメントでは、デフォルトで変換テーブルの両方の領域を保存します。LIST ステートメントでは、テーブルの両方の領域を表示します。

```
swap;
save;
list both;
```

SAS ログ

UPPER テーブル 1 を変更します。The hexadecimal value '0A' replaces hexadecimal value 01 in byte 2 for both areas of the translation table.この変換が行われたテーブル内の行と列が矢印で示されています。

```
NOTE: Table specified is UPPER.
UPPER table 1:
      [darr]
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000A02030405060708090A0B0C0D0E0F'x  <--
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '604142434445464748494A4B4C4D4E4F'x
70 '505152535455565758595A7B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x

UPPER table 2:
      [darr]
      0 1 2 3 4 5 6 7 8 9 A B C D E F
00 '000A02030405060708090A0B0C0D0E0F'x  <--
10 '101112131415161718191A1B1C1D1E1F'x
20 '202122232425262728292A2B2C2D2E2F'x
30 '303132333435363738393A3B3C3D3E3F'x
40 '404142434445464748494A4B4C4D4E4F'x
50 '505152535455565758595A5B5C5D5E5F'x
60 '60000000000000000000000000000000'x
70 '00000000000000000000000007B7C7D7E7F'x
80 '808182838485868788898A8B8C8D8E8F'x
90 '909192939495969798999A9B9C9D9E9F'x
A0 'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'x
B0 'B0B1B2B3B4B5B6B7B8B9BABBBCBDBEBF'x
C0 'C0C1C2C3C4C5C6C7C8C9CACBCCCDCECF'x
D0 'D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF'x
E0 'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF'x
F0 'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF'x
```

11 部

ロケール、エンコーディングおよびトランスコーディングの値

21 章	LOCALE=システムオプションの値	1097
22 章	DBCS データ処理に関連する SAS システムオプション	1115
23 章	SAS 言語要素中のエンコーディング値	1117
24 章	SAS セッションのエンコーディング値	1131

21

LOCALE=システムオプションの値

ロケールからエンコーディングへのマッピング 1097

ロケールからエンコーディングへのマッピング

このテーブルは、SAS 名または Posix 名を使用して指定した有効な LOCALE=値を示しています。別名も示されています。別名のないロケールもあります。DATESTYLE と PAPERSIZE システムオプションは、自動的に設定されます。

次に、例を示します。

```
sas -locale arabic_algeria
```

Arabic_Algeria の LOCALE=値を指定した場合の、対応するシステムオプションのデフォルト設定は次のとおりです。

```
DATESTYLE=DMY
PAPERSIZE=A4
```

表 21.1 に、SAS 名、別名、POSIX 値、LINUX エンコーディング値、DATESTYLE=システムオプション、および PAPERSIZE=システムオプション値をリストします。

表 21.1 POSIX ロケール値および Linux エンコーディング値に基づく DATESTYLE=および PAPERSIZE=システムオプションのデフォルト値

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Afrikaans_SouthAfrica 別名: Afrikaans, af	af_ZA	latin1	YMD	A4
Albanian_Albania 別名: Albanian, sq	sq_AL	latin2	YMD	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Arabic_Bahrain	ar_BH	arabic	DMY	A4
Arabic_Algeria	ar_DZ	arabic	DMY	A4
Arabic_Egypt	ar_EG	arabic	DMY	A4
Arabic_India	ar_IN	arabic	DMY	A4
Arabic_Iraq	ar_IQ	arabic	DMY	A4
Arabic_Jordan	ar_JO	arabic	DMY	A4
Arabic_Kuwait	ar_KW	arabic	DMY	A4
Arabic_Lebanon	ar_LB	arabic	DMY	A4
Arabic_Libya	ar_LY	arabic	DMY	A4
Arabic_Morocco	ar_MA	arabic	DMY	A4
Arabic_Oman	ar_OM	arabic	DMY	A4
Arabic_Qatar	ar_QA	arabic	DMY	A4
Arabic_SaudiArabia	ar_SA	arabic	DMY	A4
Arabic_Sudan	ar_SD	arabic	DMY	A4
Arabic_Syria	ar_SY	arabic	DMY	A4
Arabic_Tunisia	ar_TN	arabic	DMY	A4
Arabic_UnitedArabEmirates 別名: Arabic, ar	ar_AE	arabic	DMY	A4
Arabic_Yemen	ar_YE	arabic	DMY	A4
Basque_Spain 別名: Basque, Euskara, eu	eu_ES	latin9	DMY	A4
Bengali_India 別名: Bengali, bn	bn_IN	latin1	DMY	A4
Bosnian_BosniaHerzegovina 別名: Bosnian, bs	bs_BA	latin2	YMD	A4
Bulgarian_Bulgaria 別名: Bulgarian, bg	bg_BG	cyrillic	YMD	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Byelorussian_Belarus 別名: Byelorussian, Belarusian, Byelorussian_Belarus, be	be_BY	cyrillic	DMY	A4
Catalan_Spain 別名: Catalan, ca	ca_ES	latin1	DMY	A4
Chinese_China 別名: Chinese、zh_HANS、zh_HANS_CN	zh_CN	euc-cn	YMD	A4
Chinese_HongKong 別名: zh_HANT_HK	zh_HK	euc-tw、ms-950 ¹	YMD	A4
Chinese_Macau 別名: zh_HANT_MO	zh_MO	euc-tw、mis-950 ¹	YMD	A4
Chinese_Singapore 別名: zh_HANS_SG	zh_SG	euc-cn	DMY	A4
Chinese_Taiwan 別名: zh_HANT_TW	zh_TW	euc-tw、ms-950 ¹	YMD	A4
Cornish_UnitedKingdom 別名: Cornish, kw	kw_GB	latin1	DMY	A4
Croatian_BosniaHerzegovina	hr_BA	latin2	YMD	A4
Croatian_Croatia 別名: Croatian, hr	hr_HR	latin2	YMD	A4
Czechia 別名: Czech_CzechRepublic, cs	cs_CZ	latin2	DMY	A4
Danish_Denmark 別名: Danish, da	da_DK	latin9	DMY	A4
Dutch_Belgium	nl_BE	latin1	DMY	A4
Dutch_Netherlands 別名: Dutch, nl	nl_NL	latin1	DMY	A4
English_Australia	en_AU	latin1	DMY	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
English_Belgium	en_BE	latin9	DMY	A4
English_Botswana	en_BW	latin1	DMY	A4
English_Canada	en_CA	latin1	DMY	letter
English_Caribbean	en_CB	latin1	MDY	A4
English_HongKong	en_HK	latin9	DMY	A4
English_Ireland	en_IE	latin9	DMY	A4
English_India	en_IN	latin9	DMY	A4
English_Jamaica	en_JM	latin1	DMY	A4
English_Malta	en_MT	latin9	DMY	A4
English_NewZealand	en_NZ	latin1	DMY	A4
English_Philippines	en_PH	latin1	MDY	letter
English_Singapore	en_SG	latin9	DMY	A4
English_SouthAfrica	en_ZA	latin1	DMY	A4
English_UnitedStates 別名: English, en	en_US	latin1	MDY	letter
English_UnitedKingdom	en_GB	latin9	DMY	A4
English_Zimbabwe	en_ZW	latin1	DMY	A4
Estonian_Estonia	et_EE	latin6	DMY	A4
Faroese_FaroeIslands 別名: Faroese, fo	fo_FO	latin1	DMY	A4
Finnish_Finland 別名: Finnish, fi	fi_FI	latin9	DMY	A4
French_Belgium	fr_BE	latin9	DMY	A4
French_Canada	fr_CA	latin1	DMY	letter
French_France 別名: French, fr	fr_FR	latin9	DMY	A4
French_Luxembourg	fr-LU	latin9	DMY	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
French_Switzerland	fr_CH	latin9	DMY	A4
German_Austria	de_AT	latin9	DMY	A4
German_Germany 別名: German, de	de_DE	latin9	DMY	A4
German_Liechtenstein	de_LI	latin9	DMY	A4
German_Luxembourg	de_LU	latin9	DMY	A4
German_Switzerland	de_CH	latin9	DMY	A4
Greek_Cyprus	el_CY	greek	DMY	A4
Greek_Greece 別名: Greek, el	el_GR	greek	DMY	A4
Greenlandic_Greenland 別名: Greenlandic, kl	kl_GL	latin1	DMY	A4
Hebrew_Israel 別名: Hebrew, he	he_IL	hebrew	DMY	A4
Hindi_India 別名: Hindi, hi	hi_IN	latin1	DMY	A4
Hungarian_Hungary 別名: Hungarian, hu	hu_HU	latin2	YMD	A4
Icelandic_Iceland 別名: Icelandic, is	is_IS	latin1	DMY	A4
Indonesian_Indonesia 別名: Indonesian, id	id_ID	latin1	DMY	A4
Irish_Ireland 別名: Irish, ga	ga_IE	Latin8	DMY	A4
Italian_Italy 別名: Italian, it	it_IT	latin9	DMY	A4
Italian_Switzerland	it_CH	latin9	DMY	A4
Japanese_Japan 別名: Japanese, ja	ja_JP	euc-jp、 shift-jis ¹	YMD	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Kazakh_Kazakhstan 別名: Kazakh, kk	kk_KZ	Cyrillic	DMY	A4
Korean_Korea 別名: Korean, ko, Korean_SouthKorea	ko_KR	euc-kr	YMD	A4
Latvian_Latvia 別名: Latvian, lv	lv_LV	latin6	YMD	A4
Lithuanian_Lithuania 別名: Lithuanian, lt	lt_LT	latin6	YMD	A4
Macedonian_Macedonia 別名: Macedonian, mk	mk_MK	cyrillic	DMY	A4
Malay_Malaysia 別名: Malay, ms	ms_MY	latin1	DMY	A4
Maltese_Malta	mt_MT	latin3	DMY	A4
Marathi_India 別名: Marathi, mr	mr_IN	latin1	DMY	A4
NorwegianBokmal_Norway 別名: NorwegianBokmal, nb	nb_NO	latin9	DMY	A4
NorwegianNynorsk_Norway 別名: NorwegianNynorsk, nn	nn_NO	latin9	DMY	A4
Norwegian_Norway 別名: Norwegian, no	no_NO	latin9	DMY	A4
Persian_India	fa_IN	arabic	YMD	A4
Persian_Iran 別名: Persian, fa	fa_IR	arabic	YMD	A4
Polish_Poland 別名: Polish, pl	pl_PL	latin2	YMD	A4
Portuguese_Brazil	pt_BR	latin1	DMY	A4
Portuguese_Portugal 別名: Portuguese, pt	pt_PT	latin1	DMY	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Romanian_Romania 別名: Romanian, ro	ro_RO	latin2	DMY	A4
Russian_Russia 別名: Russian, ru	ru_RU	cyrillic	DMY	A4
Russian_Ukraine	ru_UA	cyrillic	DMY	A4
Serbian_BosniaHerzegovina	sr_BA	cyrillic	DMY	A4
Serbian_Montenegro	sr_ME	cyrillic	DMY	A4
Serbian_Serbia 別名: sr_RS、sr_RS	sr_RS sr_CYRL_RS	cyrillic	DMY	A4
SerbianLatin_BosniaHerzegovina	sh_BA	latin2	DMY	A4
SerbianLatin_Montenegro	sh_ME	latin2	DMY	A4
SerbianLatin_Serbia 別名: sh_LATN、sh_LATN_RS	sh_RS sh_RS	latin2	DMY	A4
Slovak_Slovakia 別名: Slovak, Slovakian,Slovakian_Slovakia , sk	sk_SK	latin2	DMY	A4
Slovenian_Slovenia 別名: Slovenian, si	sl_SI	latin2	YMD	A4
Spanish_Argentina	es_AR	latin1	DMY	A4
Spanish_Bolivia	es_BO	latin1	DMY	A4
Spanish_Chile	es_CL	latin1	DMY	letter
Spanish_Colombia	es_CO	latin1	DMY	letter
Spanish_CostaRica	es_CR	latin1	DMY	letter
Spanish_DominicanRepublic	es_DO	latin1	DMY	A4
Spanish_Ecuador	es_EC	latin1	DMY	A4
Spanish_ElSalvador	es_SV	latin1	MDY	letter
Spanish_Guatemala	es_GT	latin1	DMY	letter

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
Spanish_Honduras	es_HN	latin1	MDY	A4
Spanish_Mexico	es_MX	latin1	DMY	letter
Spanish_Nicaragua	es_NI	latin1	MDY	letter
Spanish_Panama	es_PA	latin1	MDY	letter
Spanish_Paraguay	es_PY	latin1	DMY	A4
Spanish_Peru	es_PE	latin1	DMY	A4
Spanish_PuertoRico	es_PR	latin1	MDY	letter
Spanish_Spain 別名: Spanish, es	es_ES	latin9	DMY	A4
Spanish_UnitedStates	es_US	latin1	DMY	letter
Spanish_Uruguay	es_UY	latin1	DMY	A4
Spanish_Venezuela	es_VE	latin1	DMY	letter
Swedish_Sweden 別名: Swedish, sv	sv_SE	latin9	YMD	A4
Tagalog_Philippines 別名: Tagalog, Filipino, Filipino_Philippines, ti	ti_ph	latin1	MDY	letter
Tamil_India 別名: Tamil, ta	ta_IN	latin1	DMY	A4
Telugu_India 別名: Telugu, te	te_IN	latin1	DMY	A4
Thai_Thailand 別名: Thai, th	th_TH	thai	DMY	A4
Turkish_Turkey 別名: Turkish, tr	tr_TR	latin5	DMY	A4
Ukrainian_Ukraine 別名: Ukrainian, uk	uk_UA	cyrillic	DMY	A4
Vietnamese_Vietnam	vi_VN	latin1	DMY	A4

SAS 名	Posix ロケール	LINUX エンコーディング	DATESTYLE=	PAPERSIZE=
別名: Vietnamese, vi				

1 プラットフォームに依存

表 21.2 に、SAS 名、別名、POSIX 値、LINUX エンコーディング値、および DFLANG= システムオプション値をリストします。

表 21.2 POSIX ロケール値および Linux エンコーディング値に基づく DFLANG=システムオプションのデフォルト値

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Afrikaans_SouthAfrica 別名: Afrikaans, af	af_ZA	latin1	English
Albanian_Albania 別名: Albanian, sq	sq_AL	latin2	English
Arabic_Bahrain	ar_BH	arabic	English
Arabic_Algeria	ar_DZ	arabic	English
Arabic_Egypt	ar_EG	arabic	English
Arabic_India	ar_IN	arabic	English
Arabic_Iraq	ar_IQ	arabic	English
Arabic_Jordan	ar_JO	arabic	English
Arabic_Kuwait	ar_KW	arabic	English
Arabic_Lebanon	ar_LB	arabic	English
Arabic_Libya	ar_LY	arabic	English
Arabic_Morocco	ar_MA	arabic	English
Arabic_Oman	ar_OM	arabic	English
Arabic_Qatar	ar_QA	arabic	English
Arabic_SaudiArabia	ar_SA	arabic	English
Arabic_Sudan	ar_SD	arabic	English

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Arabic_Syria	ar_SY	arabic	English
Arabic_Tunisia	ar_TN	arabic	English
Arabic_UnitedArabEmirates 別名: Arabic, ar	ar_AE	arabic	English
Arabic_Yemen	ar_YE	arabic	English
Basque_Spain 別名: Basque, Euskara, eu	eu_ES	latin9	English
Bengali_India 別名: Bengali, bn	bn_IN	latin1	English
Bosnian_BosniaHerzegovina 別名: Bosnian, bs	bs_BA	latin2	English
Bulgarian_Bulgaria 別名: Bulgarian, bg	bg_BG	cyrillic	English
Byelorussian_Belarus 別名: Byelorussian, Belarusian, Byelorussian_Belarus, be	be_BY	cyrillic	English
Catalan_Spain 別名: Catalan, ca	ca_ES	latin1	English
Chinese_China 別名: Chinese、zh_HANS、 zh_HANS_CN	zh_CN	euc-cn	Locale
Chinese_HongKong 別名: zh_HANT_HK	zh_HK	euc-tw、ms-950	Locale
Chinese_Macau 別名: zh_HANT_MO	zh_MO	euc-tw、ms-950	Locale
Chinese_Singapore 別名: zh_HANS_SG	zh_SG	euc-cn	Locale
Chinese_Taiwan	zh_TW	euc-tw、ms950	Locale

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
別名: zh_HANT_TW			
Cornish_UnitedKingdom 別名: Cornish, kw	kw_GB	latin1	English
Croatian_BosniaHerzegovina	hr_BA	latin2	Croatian
Croatian_Croatia 別名: Croatian, hr	hr_HR	latin2	Croatian
Czech_CzechRepublic alias: Czech, cs	cs_CZ	latin2	Czech
Danish_Denmark 別名: Danish, da	da_DK	latin9	Danish
Dutch_Belgium	nl_BE	latin1	Dutch
Dutch_Netherlands 別名: Dutch, nl	nl_NL	latin1	Dutch
English_Australia	en_AU	latin1	English
English_Belgium	en_BE	latin9	English
English_Botswana	en_BW	latin1	English
English_Canada	en_CA	latin1	English
English_Caribbean	en_CB	latin1	English
English_HongKong	en_HK	latin9	English
English_Ireland	en_IE	latin9	English
English_India	en_IN	latin9	English
English_Jamaica	en_JM	latin1	English
English_Malta	en_MT	latin9	English
English_NewZealand	en_NZ	latin1	English
English_Philippines	en_PH	latin1	English

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
English_Singapore	en_SG	latin9	English
English_SouthAfrica	en_ZA	latin1	English
English_UnitedStates 別名: English, en	en_US	latin1	English
English_UnitedKingdom	en_GB	latin9	English
English_Zimbabwe	en_ZW	latin1	English
Estonian_Estonia	et_EE	latin6	English
Faroese_FaroeIslands 別名: Faroese, fo	fo_FO	latin1	English
Finnish_Finland 別名: Finnish, fi	fi_FI	latin9	Finnish
French_Belgium	fr_BE	latin9	French
French_Canada	fr_CA	latin1	French
French_France 別名: French, fr	fr_FR	latin9	French
French_Luxembourg	fr-LU	latin9	French
French_Switzerland	fr_CH	latin9	Swiss_French
German_Austria	de_AT	latin9	German
German_Germany 別名: German, de	de_DE	latin9	German
German_Liechtenstein	de_LI	latin9	German
German_Luxembourg	de_LU	latin9	German
German_Switzerland	de_CH	latin9	Swiss_ German
Greek_Cyprus	el_CY	greek	English

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Greek_Greece 別名: Greek, el	el_GR	greek	English
Greenlandic_Greenland 別名: Greenlandic, kl	kl_GL	latin1	English
Hebrew_Israel 別名: Hebrew, he	he_IL	hebrew	English
Hindi_India 別名: Hindi, hi	hi_IN	latin1	English
Hungarian_Hungary 別名: Hungarian, hu	hu_HU	latin2	Hungarian
Icelandic_Iceland 別名: Icelandic, is	is_IS	latin1	English
Indonesian_Indonesia 別名: Indonesian, id	id_ID	latin1	English
Irish_Ireland 別名: Irish, ga	ga_IE	Latin8	English
Italian_Italy 別名: Italian, it	it_IT	latin9	Italian
Italian_Switzerland	it_CH	latin9	Italian
Japanese_Japan 別名: Japanese, ja	ja_JP	euc-jp、 shift-jis	Japanese
Kazakh_Kazakhstan 別名: Kazakh, kk	kk_KZ	Cyrillic	English
Korean_Korea 別名: Korean, ko, Korean_SouthKorea	ko_KR	euc-kr	Locale
Latvian_Latvia 別名: Latvian, lv	lv_LV	latin6	English

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Lithuanian_Lithuania 別名: Lithuanian, lt	lt_LT	latin6	English
Macedonian_Macedonia 別名: Macedonian, mk	mk_MK	cyrillic	English
Malay_Malaysia 別名: Malay, ms	ms_MY	latin1	English
Maltese_Malta	mt_MT	latin3	English
Marathi_India 別名: Marathi, mr	mr_IN	latin1	English
NorwegianBokmal_Norway 別名: NorwegianBokmal, nb	nb_NO	latin9	Norwegian
NorwegianNynorsk_Norway 別名: NorwegianNynorsk, nn	nn_NO	latin9	Norwegian
Norwegian_Norway 別名: Norwegian, no	no_NO	latin9	Norwegian
Persian_India	fa_IN	arabic	English
Persian_Iran 別名: Persian, fa	fa_IR	arabic	English
Polish_Poland 別名: Polish, pl	pl_PL	latin2	Polish
Portuguese_Brazil	pt_BR	latin1	Portuguese
Portuguese_Portugal 別名: Portuguese, pt	pt_PT	latin1	Portuguese
Romanian_Romania 別名: Romanian, ro	ro_RO	latin2	English
Russian_Russia 別名: Russian, ru	ru_RU	cyrillic	Russian

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Russian_Ukraine	ru-UA	cyrillic	Russian
Serbian_BosniaHerzegovina	sr_BA	cyrillic	English
Serbian_Montenegro	sr_ME	cyrillic	English
Serbian_Serbia 別名: sr_RS、sr_RS	sr_RS sr_CYRL_RS	cyrillic	English
SerbianLatin_BosniaHerzegovina	sh_BA	latin2	English
SerbianLatin_Montenegro	sh_ME	latin2	English
SerbianLatin_Serbia 別名: sh_LATN、sh_LATN_RS	sh_RS sh_RS	latin2	English
Slovak_Slovakia alias: Slovak, Slovakian, Slovakian_Slovakia ,sk	sk_SK	latin2	English
Slovenian_Slovenia 別名: Slovenian, si	sl_SI	latin2	Slovenian
Spanish_Argentina	es_AR	latin1	Spanish
Spanish_Bolivia	es_BO	latin1	Spanish
Spanish_Chile	es_CL	latin1	Spanish
Spanish_Colombia	es_CO	latin1	Spanish
Spanish_CostaRica	es_CR	latin1	Spanish
Spanish_DominicanRepublic	es_DO	latin1	Spanish
Spanish_Ecuador	es_EC	latin1	Spanish
Spanish_ElSalvador	es_SV	latin1	Spanish
Spanish_Guatemala	es_GT	latin1	Spanish
Spanish_Honduras	es_HN	latin1	Spanish
Spanish_Mexico	es_MX	latin1	Spanish

SAS 名	Posix ロケール	LINUX エンコーディング	DFLANG=
Spanish_Nicaragua	es_NI	latin1	Spanish
Spanish_Panama	es_PA	latin1	Spanish
Spanish_Paraguay	es_PY	latin1	Spanish
Spanish_Peru	es_PE	latin1	Spanish
Spanish_PuertoRico	es_PR	latin1	Spanish
Spanish_Spain 別名: Spanish, es	es_ES	latin9	Spanish
Spanish_UnitedStates	es_US	latin1	Spanish
Spanish_Uruguay	es_UY	latin1	Spanish
Spanish_Venezuela	es_VE	latin1	Spanish
Swedish_Sweden 別名: Swedish, sv	sv_SE	latin9	Swedish
Tagalog_Philippines 別名: Tagalog, Filipino, Filipino_Philippines, ti	ti_ph	latin1	English
Tamil_India 別名: Tamil, ta	ta_IN	latin1	English
Telugu_India 別名: Telugu, te	te_IN	latin1	English
Thai_Thailand 別名: Thai, th	th_TH	thai	English
Turkish_Turkey 別名: Turkish, tr	tr_TR	latin5	English
Ukrainian_Ukraine 別名: Ukrainian, uk	uk_UA	cyrillic	English
Vietnamese_Vietnam 別名: Vietnamese, vi	vi_VN	latin1	English

DBCS データ処理に関連する SAS システムオプション

DBCS の SAS セッションで使用されるシステムオプションの概要.....	1115
SAS セッションの DBCS 値.....	1115

DBCS の SAS セッションで使用されるシステムオプションの概要

SAS セッションの DBCS エンコーディング値を指定するには、DBCSLANG=と DBCSTYPE=システムオプションを使用できます。ただし、DBCS の SAS セッションの設定では、ENCODING=システムオプションの使用をお勧めします。ENCODING=システムオプションは優先されます。詳細については、“[SAS セッションエンコーディングの設定](#)” (23 ページ)を参照してください。詳細については“[ENCODING システムオプション](#)” (968 ページ)を参照してください。

SAS セッションの DBCS 値

次のテーブルに LINUX 動作環境でサポートされる DBCSLANG=および DBCSTYPE=システムオプションの値を示します。

注: エンコーディング値にハイフン(-)が含まれる場合は、エンコーディング値を引用符で囲んでください。

表 22.1 DBCSLANG=および DBCSTYPE=システムオプションでの DBCS サポート
値

DBCSLANG=	LINUX DBCSTYPE=
Chinese	dec
Chinese	hp15
Chinese	euc
Chinese	pcms
Japanese	dec
Japanese	pcibm
Japanese	hp15
Japanese	SJIS
Korean	pcibm
Korean	pcms
Korean	dec
Korean	euc
Korean	hp15
Taiwanese	dec
Taiwanese	pcibm
Taiwanese	hp15
Taiwanese	pcms

SAS 言語要素中のエンコーディング値

エンコーディング値を使用する SAS 言語要素の概要	1117
データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値	1118

エンコーディング値を使用する SAS 言語要素の概要

SAS セッションのエンコーディングが SAS ファイルのエンコーディングまたは SAS ファイル内に存在するデータと異なる場合、トランスコーディングが必要になります。SAS ファイルが Western Latin1 エンコーディングで作成され、ドイツ語 EBCDIC エンコーディングを使用する IBM メインフレームに移動されたとします。IBM メインフレームがそのファイルに正常にアクセスできるように、SAS データファイルを Western Latin1 エンコーディングからドイツ語 EBCDIC エンコーディングにトランスコードする必要があります。トランスコーディングオプションを含む SAS 言語要素などの、トランスコーディングの概念の詳細については、[4 章, “各国語サポート関連のトランスコーディング” \(33 ページ\)](#)を参照してください。

データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値

次の表は、各動作環境でデータのトランスコーディングに使用される SBCS、DBCS および Unicode エンコーディング値のリストです。エンコーディング値は、トランスコーディングのオプションを含む SAS 言語要素で有効です。

注: エンコーディング値にハイフン(-)が含まれる場合は、エンコーディング値を引用符で囲んでください。

表 23.1 データのトランスコーディングに使用する SBCS、DBCS および Unicode エンコーディング値

エンコーディング名	短縮名	説明	1 文字の最大バイト
aarabic	aara	アラビア語 (Macintosh)	1
acroatian	acro	MacOS 36 Croatian	1
acyrillic	acyr	MacOS 7 Cyrillic	1
agreek	agrk	ギリシャ語 (Macintosh)	1
ahebrew	aheb	ヘブライ語 (Macintosh)	1
aiceland	aice	アイスランド語 (Macintosh)	1
any	anye	トランスコーディング指定無し	1
arabic	arab	アラビア語(ISO)	1
aroman	arom	Roman (Macintosh)	1

エンコーディング名	短縮名	説明	1 文字の最大バイト
aromania	arma	MacOS 38 Romania	1
athai	atha	MacOS 21-Thai	1
aturkish	atur	トルコ語 (Macintosh)	1
aukrainian	aukr	ウクライナ語 (Macintosh)	1
big5	big5	繁体字中国語(Big5)	2
cyrillic	cyr1	キリル語(ISO)	1
dec-cn	zvms	簡体字中国語(DEC)	4
dec-jp	jvms	日本語(DEC)	2
dec-tw	yvms	繁体字中国語(DEC)	4
ebcdic037	e037	北米諸語(EBCDIC)	1
ebcdic275	e275	ブラジル(EBCDIC)	1
ebcdic424	e424	ヘブライ語 (EBCDIC)	1
ebcdic425	e425	アラビア語 (EBCDIC)	1
ebcdic500	e500	国際化対応 (EBCDIC)	1
ebcdic838	e838	タイ語(EBCDIC)	1
ebcdic870	e870	中央ヨーロッパ諸語(EBCDIC)	1
ebcdic875	e875	ギリシャ語 (EBCDIC)	1
ebcdic905	e905	ラテン諸語 3 EBCDIC	1

エンコーディング 名	短縮名	説明	1 文字の最大バイ ト
ebcdic924	e924	ヨーロッパ諸語 (EBCDIC)	1
ebcdic1025	ecyr	キリル語(EBCDIC)	1
ebcdic1026	etur	トルコ語(EBCDIC)	1
ebcdic1047	elat	西欧(EBCDIC)	1
ebcdic1097	e097	ペルシア諸語 (EBCDIC)	1
ebcdic1112	ebal	バルト語(EBCDIC)	1
ebcdic1122	eest	エストニア語 (EBCDIC)	1
ebcdic1130	evie	ベトナム語 (EBCDIC)	1
ebcdic1137	e137	デーバナーガリー 文字(EBCDIC)	1
ebcdic1140	e140	北米諸語(EBCDIC)	1
ebcdic1141	e141	オーストリア/ドイ ツ(EBCDIC)	1
ebcdic1142	e142	デンマーク/ノルウ エー(EBCDIC)	1
ebcdic1143	e143	フィンランド/スウ エーデン(EBCDIC)	1
ebcdic1144	e144	イタリア(EBCDIC)	1
ebcdic1145	e145	スペイン(EBCDIC)	1
ebcdic1146	e146	英国(EBCDIC)	1
ebcdic1147	e147	フランス(EBCDIC)	1
ebcdic1148	e148	国際化対応 (EBCDIC)	1

エンコーディング名	短縮名	説明	1 文字の最大バイト
ebcdic1149	e149	アイスランド語 (EBCDIC)	1
ebcdic1153	e153	Latin 2 Euro EBCDIC	1
ebcdic1154	e154	Cyrillic Euro EBCDIC	1
ebcdic1155	e155	Turkey Euro EBCDIC	1
ebcdic1156	e156	Baltic Euro EBCDIC	1
ebcdic1157	e157	Estonia Euro EBCDIC	1
ebcdic1158	e158	Cyrillic Ukraine Euro EBCDIC	1
ebcdic1160	e160	cp1160 EBCDIC	1
ebcdic1164	e164	cp1164 EBCDIC	1
ebcdicany	eany	すべての EBCDIC エンコーディングと互換性のあるデータセットを作成可能	1
euc-cn	zeuc	簡体字中国語(EUC)	2
euc-jp	jeuc	日本語(EUC)	4
euc-kr	keuc	韓国語(EUC)	4
euc-tw	yeuc	繁体字中国語(EUC)	4
fujitsu-cn	zfuj	簡体字中国語 (FACOM)	4
fujitsu-jp	jfuj	日本語(FACOM)	4
fujitsu-ko	kfuj	韓国語(FACOM)	4

エンコーディング 名	短縮名	説明	1 文字の最大バイ ト
fujitsu-tw	yfuj	繁体字中国語 (FACOM)	4
gb18030	gbke	簡体字中国語 (GB18030)	4
greek	grek	ギリシャ語(ISO)	1
hebrew	hebr	ヘブライ語(ISO)	1
hitachi-cn	zhit	簡体字中国語 (HITAC)	6
hitachi-jp	jhit	日本語(HITAC)	6
hitachi-ko	khit	韓国語(HITAC)	6
hitachi-tw	yhit	繁体字中国語 (HITAC)	4
hitsas-jp	jhts	日本語(XHITAC)	4
hitsas-ko	khts	韓国語(XHITAC)	4
hitsas-tw	yhts	繁体字中国語 (XHITAC)	4
hp15-tw	yhpx	繁体字中国語 (HP15)	2
ibm-1381	zpce	簡体字中国語 (PCIBM)	2
ibm-930	j930	Japanese katakana	4
ibm-933	kibm	韓国語(IBM)	4
ibm-935	zibm	簡体字中国語(IBM)	4
ibm-937	yibm	繁体字中国語(IBM)	4
ibm-939	jibm	日本語(IBM)	4
ibm-942	j942	日本語(PCIBM)	2

エンコーディング名	短縮名	説明	1 文字の最大バイト
ibm-949	kpce	韓国語(PCIBM)	2
iso2022cncns	zist	繁体字中国語 (ISO-2022)	4
iso2022cn gb	ziso	簡体字中国語 (ISO-2022)	4
iso2022jp	jiso	日本語(ISO-2022)	8
iso2022kr	kiso	韓国語(ISO-2022)	4
kz1048	kz48	Kazakh cp1048 Cyrillic	1
latin1	lat1	西ヨーロッパ諸語 (ISO)	1
latin2	lat2	中央ヨーロッパ諸語(ISO)	1
latin3	lat3	ラテン諸語 3(ISO 8859/3)	1
latin4	lat4	ラテン諸語 4(ISO 8859/4)	1
latin5	lat5	トルコ語(ISO)	1
latin6	lat6	バルト語(ISO)	1
latin7	lat7	バルト語	1
latin8	lat8	ラテン諸語 8(ISO 8859/14)	1
latin9	lat9	ヨーロッパ諸語 (ISO)	1
latin10	lat10	南東ヨーロッパ言語	1
macos-1	jmac	日本語(PCMAC)	2
macos-2	ymac	繁体字中国語 (PCMAC)	2

エンコーディング 名	短縮名	説明	1 文字の最大バイ ト
macos-3	kmac	韓国語(PCMAC)	2
macos-25	zmac	簡体字中国語 (PCMAC)	2
ms-932	j932	日本語(PCMS)	2
ms-936	zwin	簡体字中国語 (PCMS)	2
ms-949	kwin	韓国語(PCMS)	2
ms-950	ywin	繁体字中国語 (PCMS)	2
msdos720	p720	アラビア語(MS- DOS)	1
msdos737	p737	ギリシャ語(MS- DOS)	1
msdos775	p775	バルト語(MS-DOS)	1
open_ed-037	eous	米国(Open Edition)	1
open_ed-275	eobr	ブラジル (OpenEdition)	1
open_ed-424	eiw	ヘブライ語 (OpenEdition)	1
open_ed-425	ea2	アラビア語 (OpenEdition)	1
open_ed-838	eoth	タイ語 (OpenEdition)	1
open_ed-870	eol2	中央ヨーロッパ諸 語(OpenEdition)	1
open_ed-875	eoel	ギリシャ語 (OpenEdition)	1
open_ed-905	eol3	ラテン諸語 3(Open Edition EBCDIC)	1

エンコーディング名	短縮名	説明	1 文字の最大バイト
open_ed-924	eolt	ヨーロッパ諸語 (OpenEdition)	1
open_ed-930	oe30	カタカナ (OpenEdition)	4
open_ed-933	oe33	韓国語 (OpenEdition)	4
open_ed-935	oe35	簡体字中国語 (OpenEdition)	4
open_ed-937	oe37	繁体字中国語 (OpenEdition)	4
open_ed-939	oe39	日本語(IBM)	4
open_ed-939E	oe9e	日本語(IBM 拡張)	4
open_ed-1025	eocy	キリル語 (OpenEdition)	1
open_ed-1026	eotr	トルコ語 (OpenEdition)	1
open_ed-1047	eol1	西ヨーロッパ諸語 (OpenEdition)	1
open_ed-1097	eofa	ペルシア諸語 (OpenEdition EBCDIC)	1
open_ed-1112	eobl	バルト語 (OpenEdition)	1
open_ed-1122	eoet	エストニア語 (OpenEdition)	1
open_ed-1130	eovi	ベトナム語 (OpenEdition)	1
open_ed-1140	eo40	北米諸語 (OpenEdition)	1
open_ed-1141	eo41	オーストリア/ドイツ (OpenEdition)	1

エンコーディング 名	短縮名	説明	1 文字の最大バイ ト
open_ed-1142	eo42	デンマーク/ノルウ エー(OpenEdition)	1
open_ed-1143	eo43	フィンランド/スウ エーデン (OpenEdition)	1
open_ed-1144	eo44	イタリア (OpenEdition)	1
open_ed-1145	eo45	スペイン (OpenEdition)	1
open_ed-1146	eo46	英国(OpenEdition)	1
open_ed-1147	eo47	フランス (OpenEdition)	1
open_ed-1148	eo48	国際化対応 (OpenEdition)	1
open_ed-1149	eoIS	アイスランド (OpenEdition EBCDIC)	1
open_ed-1153	eo53	Latin 2 Euro OpenEdition EBCDIC	1
open_ed-1154	eo54	Cyrillic Euro OpenEdition EBCDIC	1
open_ed-1155	eo55	Turkey Euro OpenEdition EBCDIC	1
open_ed-1156	eo56	Baltic Euro OpenEdition EBCDIC	1
open_ed-1157	eo57	Estonia Euro OpenEdition EBCDIC	1
open_ed-1158	eo58	Cryrillic Ukraine Euro 1158	1

エンコーディング名	短縮名	説明	1 文字の最大バイト
		OpenEdition EBCDIC	
open_ed-1160	eo60	1160 OpenEdition EBCDIC	1
open_ed-1164	eo64	1164 OpenEdition EBCDIC	1
open_ed-1166	eo66	1166 OpenEdition kz-cyr	1
pciscii806	p806	インド(PC)	1
pcoem437	p437	米国(IBM-PC)	1
pcoem850	p850	西ヨーロッパ諸語 (IBM-PC)	1
pcoem852	p852	中央ヨーロッパ諸 語(IBM-PC)	1
pcoem857	p857	トルコ語(IBM-PC)	1
pcoem858	p858	ヨーロッパ諸語 (IBM-PC)	1
pcoem860	p860	ポルトガル語(MS- DOS)	1
pcoem862	p862	ヘブライ語(IBM- PC)	1
pcoem863	p863	フランス語(カナダ) (IBM-PC)	1
pcoem864	p864	アラビア語(IBM- PC)	1
pcoem865	p865	北欧諸語(IBM-PC)	1
pcoem866	p866	キリル語(IBM-PC)	1
pcoem869	p869	ギリシャ語(IBM- PC)	1

エンコーディング名	短縮名	説明	1 文字の最大バイト
pcoem874	p874	タイ語(IBM-PC)	1
pcoem921	p921	バルト語(IBM-PC)	1
pcoem922	p922	エストニア語(IBM-PC)	1
pcoem1129	pvie	ベトナム語(IBM-PC)	1
pc1098	po98	ペルシア諸語(PC)	1
roman8	rom8	HP Roman 8	1
shift-jis	SJIS	日本語(SJIS)	2
shift-jis04	sjs4	Shift_JIS-2004 サロゲートペアサポート	2
thai	thai	タイ語(ISO)	1
us-ascii	ansi	すべての ASCII エンコーディングと互換性のあるデータセットを作成可能	1
utf-8	utf8	Unicode (UTF-8)	4
utf-16be	u16b	Unicode (UTF-16BE)	2 ¹
utf-16le	u16l	Unicode (UTF-16LE)	2 ¹
utf-32be	u32b	Unicode (UTF-32BE)	4 ²
utf-32le	u32l	Unicode (UTF-32LE)	4 ²
warabic	wara	アラビア語 (Windows)	1

エンコーディング 名	短縮名	説明	1 文字の最大バイ ト
wbaltic	wbal	バルト語 (Windows)	1
wcyrillic	wcyr	キリル語 (Windows)	1
wgreek	wgrk	ギリシャ語 (Windows)	1
whebrew	wheb	ヘブライ語 (Windows)	1
wlatin1	wlt1	西ヨーロッパ諸語 (Windows)	1
wlatin2	wlt2	中央ヨーロッパ諸 語(Windows)	1
wturkish	wtur	トルコ語 (Windows)	1
wvietnamese	wvie	ベトナム語 (Windows)	1

1 UTF-16BE と UTF-16LE は各文字 2 バイトの固定長になります。

2 UTF-32BE と UTF-32LE は各文字 4 バイトの固定長になります。

SAS セッションのエンコーディング値

LINUX のエンコーディング値 1131

LINUX のエンコーディング値

次の表のエンコーディングは、LINUX 環境で有効です。

注: エンコーディング値にハイフン(-)が含まれる場合は、エンコーディング値を引用符で囲んでください。

表 24.1 LINUX の 1 バイトエンコーディング

ENCODING=値	説明
arabic	アラビア語(ISO 8859-6)
cyrillic	キリル語(ISO 8859-5)
greek	ギリシャ語(ISO 8859-7)
hebrew	ヘブライ語(ISO 8859-8)
latin1	西ヨーロッパ諸語(ISO 8859-1)
latin2	中央ヨーロッパ諸語(ISO 8859-2)
latin5	トルコ語(ISO 8859-9)

ENCODING=値	説明
latin6	バルト語(ISO 8859-4)
Latin7	バルト語(ISO 8859-13)
latin8	ケルト語(ISO 8859-14)
latin9	ヨーロッパ諸語(ISO 8859-15)
latin10	南東ヨーロッパ諸語(ISO 8859-16)
thai	タイ語(ISO 8859-11)

表 24.2 LINUX の 2 バイトエンコーディング

ENCODING=値	説明
big5	繁体字中国語(Big5)
euc-cn	簡体字中国語(EUC)
euc-jp	日本語(EUC)
euc-kr	韓国語(EUC)
euc-tw	繁体字中国語(EUC)
shift-jis	日本語(SJIS)

LINUX では、UTF-8 Unicode エンコーディングもサポートしています。

12 部

付録

付録 1	
エンコーディング名と別名	1135
付録 2	
タイムゾーン ID とタイムゾーン名	1145
付録 3	
TK_NLS_REPLACE_ICVEXT	1183

付録 1

エンコーディング名と別名

エンコーディングとその別名とエンコーディング文字セットの互換性..... 1135

エンコーディングとその別名とエンコーディング文字セットの互換性

この表には、よく用いられるエンコーディング方式とそれに対応するエンコーディング名および別名がリストされています。

エンコーディング文字セットの互換性テーブルは、同じ文字セットをサポートする言語(ロケール)を示しています

表 A12.1 エンコーディング名と別名

エンコーディング方式と文字セット	SAS のエンコーディング名	別名
Unicode	UTF-8	UTF-8
SBCS	WLATIN2	CP1250、WINDOWS-1250
	WCYRILLIC	CP1251、WINDOWS-1251
	WLATIN1	CP1252、WINDOWS-1252
	WGREEK	CP1253、WINDOWS-1253
	WTURKISH	CP1254、WINDOWS-1254

エンコーディング方式と文字セット	SAS のエンコーディング名	別名
	WHEBREW	CP1255, WINDOWS-1255
	WARABIC	CP1256, WINDOWS-1256
	WBALTIC	CP1257, WINDOWS-1257
	WVIETNAMESE	CP1258, WINDOWS-1258
	LATIN1	ISO8859_1、ISO8859-1
	LATIN2	ISO8859_2、ISO8859-2
	LATIN3	ISO8859_3、ISO8859-3
	CYRILLIC	ISO8859_5、ISO8859-5
	ARABIC	ISO8859_6、ISO8859-6
	GREEK	ISO8859_7、ISO8859-7
	HEBREW	ISO8859_8、ISO8859-8
	TURKISH	ISO8859_9、ISO8859-9
	THAI	ISO8859_11、ISO8859-11
	LATIN7	ISO8859_13、ISO8859-13
	LATIN9	ISO8859_15、ISO8859-15
	LATIN10	ISO8859_16、ISO8859-16
	OPEN_ED-838 EBCDIC838	CP838
	OPEN_ED-870 EBCDIC870	CP870
	OPEN_ED-875 EBCDIC875	CP875
	OPEN_ED-1047 EBCDIC1047	CP1047
	OPEN_ED-1122	CP1122

エンコーディング方式と文字セット	SAS のエンコーディング名	別名
	EBCDIC1122	
	OPEN_ED-1130, EBCDIC1130	CP1130
	OPEN_ED-1141 EBCDIC1141	CP1141
	OPEN_ED-1142 EBCDIC1142	CP1142
	OPEN_ED-1143 EBCDIC1143	CP1143
	OPEN_ED-1144, EBCDIC1144	CP1144
	OPEN_ED-1145 EBCDIC1145	CP1145
	OPEN_ED-1146, EBCDIC1146	CP1146
	OPEN_ED-1147, EBCDIC1147	CP1147
	OPEN_ED-1148, EBCDIC1148	CP1148
DBCS	SHIFT-JIS	MS-932、IBM-942、MACOS-1
	MS-949	MACOS-3、EUC-KR
	EUC-CN	MS-936、MACOS-25、DEC-CN、GBK、GB2312
	EUC-TW	DEC-TW
	MS-950	MACOS-2、BIG5

この表は、同じ文字セットをサポートする言語(ロケール)を示しています。たとえば、最初の列のアラビア語ロケールには、5つの互換性のある文字列 WARABIC、

ARABIC、MSDOS720、PCOEM864、OPEN_ED-425 があります。SAS で文字セットに使用されるエンコーディング名が 4 列目にリストされています。

表 A12.2 エンコーディング文字セットの互換性

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(java)
アラビア語	ar	アラビア語	WARABIC、 ARABIC、 MSDOS720、 PCOEM864、 OPEN_ED-425	CP1256、 ISO8859_6、 CP864、 WINDOWS-1256
ベンガル語	bn	ベンガル語		
ペルシャ語	fa	アラビア語	WARABIC、 ARABIC、 MSDOS720、 PCOEM864、 OPEN_ED-425	CP1256、 ISO8859_6、 CP864、 WINDOWS-1256
中国語(簡 体字)	zh	中国語(簡体 字)	MS-936、EUC- CN、IBM-935	GBK、GBK、 GB2312、CP935
中国語(繁 体字)	zh	中国語(繁体 字)	MS-950、EUC- TW、IBM-937	BIG5、BIG5、 CP937
ベラルーシ 語	be	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251
ブルガリア 語	bg	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251
マケドニア 語	mk	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251
ロシア語	ru	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251
セルビア語	sr	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(Java)
ウクライナ 語	uk	キリル語	WCYRILLIC、 CYRILLIC、 PCOEM866、 OPEN_ED- 1025	CP1251、 ISO8859_5、 CP866、CP1025、 WINDOWS-1251
ヒンディー 語	hi	デーバナー ガリー文字	PCISCII806、 OPEN_ED-1137	
マラーティ ー語	mr	デーバナー ガリー文字	PCISCII806、 OPEN_ED-1137	
ギリシャ語	el	ギリシャ語	WGREEK、 GREEK、 MSDOS737、 PCOEM869、 OPEN_ED-875	CP1253、 ISO8859_7、 CP869、CP875、 WINDOWS-1253
ヘブライ語	he	ヘブライ語	WHEBREW、 HEBREW、 PCOEM862、 OPEN_ED-424	CP1255、 ISO8859_8、 CP862、CP424、 WINDOWS-1255
日本語	ja	日本語	SHIFT-JIS、EUC- JP、IBM-939	SJIS、EUC_JP、 CP939
韓国語	ko	韓国語	MS-949、EUC- KR、IBM-933	MS649、EUC_KR、 CP933
アフリカー ンス語	af	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252
カタロニア 語	ca	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1145	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1145、 WINDOWS-1252
コーンウォ ール語	kw	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1146	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1146、 WINDOWS-1252
デンマーク 語	da	LATIN1	WLATIN1、 LATIN1、	CP1252、 ISO8859_1、 ISO8859_15、

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(Java)
			PCOEM850、 OPEN_ED- 1142	CP850、CP1142、 WINDOWS-1252
オランダ語	nl	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252
英語	ja	LATIN1	WLATIN1、 LATIN1、 PCOEM437、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP437、CP1047、 WINDOWS-1252
フェロー語	fo	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1142	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1142、 WINDOWS-1252
フィンランド語	fi	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1143	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1143、 WINDOWS-1252
フランス語	fr	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1147	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1147、 WINDOWS-1252
ドイツ語	de	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1141	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1141、 WINDOWS-1252
グリーンランド語	kl	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1142	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1142、 WINDOWS-1252
アイスランド語	is	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(java)
インドネシ ア語	id	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252
イタリア語	it	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1144	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1144、 WINDOWS-1252
マレー語	ms	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252
マンクス語	gv	LATIN1	WLATIN1、 LATIN8、 PCOEM850、 OPEN_ED- 1148	CP1252、 ISO8859_14、 CP850、CP1148、 WINDOWS-1252
ノルウェー 語	no	LATIN1	WLATIN1、 LATIN1、 PCOEM850、 OPEN_ED- 1142	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1142、 WINDOWS-1252
ポルトガル 語	pt	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1047	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1047、 WINDOWS-1252
スペイン語	es	LATIN1	WLATIN1、 LATIN9、 PCOEM850、 OPEN_ED- 1145	CP1252、 ISO8859_1、 ISO8859_15、 CP850、CP1145、 WINDOWS-1252
アルバニア 語	sq	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、CP870、 WINDOWS-1250
ボスニア語	bs	LATIN2	WLATIN2、 LATIN2、	CP1250、 ISO8859_2、

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(java)
			PCOEM852、 OPEN_ED- 870	ISO8859_16、 CP852、 CP870、 WINDOWS-1250
クロアチア 語	hr	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
チェコ語	cs	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
ハンガリー 語	hu	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
ポーランド 語	pl	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
ルーマニア 語	ro	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
セルビア語 (ラテン)	sh	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
スロバキア 語	sk	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、 CP852、 CP870、 WINDOWS-1250
スロベニア 語	sl	LATIN2	WLATIN2、 LATIN2、 PCOEM852、 OPEN_ED- 870	CP1250、 ISO8859_2、 ISO8859_16、

言語	言語コード (ISO 639)	スクリプト グループ	エンコーディン グ名(SAS)	エンコーディング 名(java)
				CP852、CP870、 WINDOWS-1250
マルタ語	mt	LATIN3	CP1254、 ISO8859_9、 CP857、CP1026	ISO8859_3
トルコ語	tr	LATIN5	WTURKISH、 TURKISH、 PCOEM857、 OPEN_ED-1026	CP1254、 ISO8859_9、 CP857、CP1026、 WINDOWS-1254
エストニア 語	et	LATIN6	WBALTIC、 LATIN6、 MSDOS775、 PCOEM921、 OPEN_ED-1112	CP1257、 ISO8859_6、 CP775、CP1112、 WINDOWS-1257
ラトビア語	lv	LATIN6	WBALTIC、 LATIN6、 MSDOS775、 PCOEM921、 OPEN_ED-1112	CP1257、 ISO8859_6、 CP775、CP1112、 WINDOWS-1257
リトアニア 語	lt	LATIN6	WBALTIC、 LATIN6、 MSDOS775、 PCOEM921、 OPEN_ED-1112	CP1257、 ISO8859_6、 CP775、CP1112、 WINDOWS-1257
タミール語	ta	タミール語		
テルグ語	te	テルグ語		
タイ語	th	タイ語	PCOEM874、 THAI	CP874、TIS620
ベトナム語	vi	ベトナム語	WVIETNAMESE、 OPEN_ED-1130	CP1258、CP1130、 WINDOWS-1258

付録 2

タイムゾーン ID とタイムゾーン名

エリア: Africa(アフリカ)	1145
エリア: America(アメリカ-北、中央、および南)	1148
エリア: Antarctica(南極)	1159
エリア: Arctic(北極)	1160
エリア: Asia(アジア)	1160
エリア: Atlantic(大西洋)	1165
エリア: Australia(オーストラリア)	1166
エリア: その他	1168
エリア: Europe(ヨーロッパ)	1174
エリア: Pacific(太平洋)	1179

エリア: Africa(アフリカ)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Africa/Abidjan	GMT	CI	グリニッジ標準時間	00:00
Africa/Accra	GMT	GH	グリニッジ標準時間	00:00
Africa/Algiers	CET	DZ	中央ヨーロッパ標準時間	01:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Africa/Bamako	GMT	ML	グリニッジ標準時間	00:00
Africa/Bangui	WAT	CF	西アフリカ標準時間	01:00
Africa/Banjul	GMT	GM	グリニッジ標準時間	00:00
Africa/Bissau	GMT	GW	グリニッジ標準時間	00:00
Africa/Brazzaville	WAT	CG	西アフリカ時間	01:00
Africa/Casablanca	+01	MA	西ヨーロッパ標準時間	01:00
Africa/Casablanca	+00	MA	西ヨーロッパ夏時間	00:00
Africa/Ceuta	CET	ES	中央ヨーロッパ標準時間	01:00
Africa/Ceuta	CEST	ES	中央ヨーロッパ夏時間	02:00
Africa/Conakry	GMT	GN	グリニッジ標準時間	00:00
Africa/Dakar	GMT	SN	グリニッジ標準時間	00:00
Africa/Douala	WAT	CM	西アフリカ時間	01:00
Africa/El_Aaiun	+01	EH	西ヨーロッパ標準時間	01:00
Africa/El_Aaiun	+00	EH	西ヨーロッパ夏時間	00:00
Africa/Freetown	GMT	SL	グリニッジ標準時間	00:00
Africa/Lome	GMT	TG	グリニッジ標準時間	00:00
Africa/Monrovia	GMT	LR	グリニッジ標準時間	00:00
Africa/Nouakchott	GMT	MR	グリニッジ標準時間	00:00
Africa/Ouagadougou	GMT	BF	グリニッジ標準時間	00:00
Africa/Sao_Tome	GMT	ST	グリニッジ標準時間	00:00
Africa/Timbuktu	GMT	CI	グリニッジ標準時間	00:00
Africa/Tunis	CET	TN	中央ヨーロッパ標準時間	01:00
Africa/Kinshasa	WAT	CD	西アフリカ標準時間	01:00
Africa/Lagos	WAT	NG	西アフリカ標準時間	01:00
Africa/Libreville	WAT	GA	西アフリカ標準時間	01:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Africa/Luanda	WAT	AO	西アフリカ標準時間	01:00
Africa/Malabo	WAT	GQ	西アフリカ標準時間	01:00
Africa/Ndjamena	WAT	TD	西アフリカ標準時間	01:00
Africa/Niamey	WAT	NE	西アフリカ標準時間	01:00
Africa/Porto-Novo	WAT	BJ	西アフリカ標準時間	01:00
Africa/Windhoek	CAT	NA	中央アフリカ時間	02:00
Africa/Blantyre	CAT	MW	中央アフリカ時間	02:00
Africa/Bujumbura	CAT	BI	中央アフリカ時間	02:00
Africa/Gaborone	CAT	BW	中央アフリカ時間	02:00
Africa/Harare	CAT	ZW	中央アフリカ時間	02:00
Africa/Kigali	CAT	RW	中央アフリカ時間	02:00
Africa/Lubumbashi	CAT	CD	中央アフリカ時間	02:00
Africa/Lusaka	CAT	ZM	中央アフリカ時間	02:00
Africa/Maputo	CAT	MZ	中央アフリカ時間	02:00
Africa/Cairo	EET	EG	東ヨーロッパ標準時間	02:00
Africa/Tripoli	EET	LY	東ヨーロッパ標準時間	02:00
Africa/Johannesburg	SAST	ZA	南アフリカ標準時間	02:00
Africa/Maseru	SAST	LS	南アフリカ標準時間	02:00
Africa/Mbabane	SAST	SZ	南アフリカ標準時間	02:00
Africa/Addis_Ababa	EAT	ET	東アフリカ時間	03:00
Africa/Asmara	EAT	ET	東アフリカ時間	3.00
Africa/Asmera	EAT	ER	東アフリカ時間	03:00
Africa/Dar_es_Salaam	EAT	TZ	東アフリカ時間	03:00
Africa/Djibouti	EAT	DJ	東アフリカ時間	03:00
Africa/Juba	EAT	SS	東アフリカ時間	03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Africa/Kampala	EAT	UG	東アフリカ時間	03:00
Africa/Khartoum	CAT	SD	中央アフリカ時間	02:00
Africa/Mogadishu	EAT	SO	東アフリカ時間	03:00
Africa/Nairobi	EAT	KE	東アフリカ時間	03:00

エリア: America(アメリカ-北、中央、および南)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Adak	HST	US	ハワイ・アリューシャン標準時間	-10:00
America/Adak	HDT	US	ハワイ・アリューシャン夏時間	-09:00
America/Anchorage	AKST	US	アラスカ標準時間	-09:00
America/Anchorage	AKDT	US	アラスカ夏時間	-08:00
America/Anguilla	AST	AI	アトランティック標準時間	-04:00
America/Antigua	AST	AG	アトランティック標準時間	-04:00
America/Araguaina	-03	BR	ブラジル標準時間	-03:00
America/Argentina/Buenos_Aires	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Catamarca	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/ComodRivadavia	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Cordoba	-03	AR	アルゼンチン時間	-03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Argentina/Jujuy	-03	AR	アルゼンチン標準標準時	-03:00
America/Argentina/La_Rioja	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Mendoza	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Rio_Gallegos	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Salta	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/San_Juan	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/San_Luis	-03	AR	西アルゼンチン標準時間	-03:00
America/Argentina/Tucuman	-03	AR	アルゼンチン標準時間	-03:00
America/Argentina/Ushuaia	-03	AR	アルゼンチン標準時間	-03:00
America/Aruba	AST	AW	アトランティック標準時間	-04:00
America/Asuncion	-04	PY	パラグアイ標準時間	-04:00
America/Asuncion	-03	PY	パラグアイ夏時間	-03:00
America/Atikokan	EST	CA	東部標準時間	-05:00
America/Atka	HST	US	ハワイ・アリューシャン標準時間	-10:00
America/Atka	HDT	US	ハワイ・アリューシャン夏時間	-09:00
America/Bahia	-03	BR	ブラジル標準時間	-03:00
America/Bahia_Banderas	CST	MX	中部標準時間	-06:00
America/Bahia_Banderas	CDT	MX	中部夏時間	-05:00
America/Barbados	AST	BB	アトランティック標準時間	-04:00
America/Belem	-03	BR	ブラジル標準時間	-03:00
America/Belize	CST	BZ	中部標準時間	-06:00
America/Blanc-Sablon	AST	CA	アトランティック標準時間	-04:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Boa_Vista	-04	BR	アマゾン標準時間	-04:00
America/Bogota	-05	CO	コロンビア標準時間	-05:00
America/Boise	MST	US	山岳部標準時間	-07:00
America/Boise	MDT	US	山岳部夏時間	-06:00
America/Buenos_Aires	-03	AR	アルゼンチン標準時間	-03:00
America/Cambridge_Bay	MST	CA	山岳部標準時間	-07:00
America/Cambridge_Bay	MDT	CA	山岳部夏時間	-06:00
America/Campo_Grande	-04	BR	アマゾン標準時間	-04:00
America/Campo_Grande	-03	BR	アマゾン夏時間	-03:00
America/Cancun	EST	MX	東部標準時間	-05:00
America/Caracas	-04	VE	ベネズエラ時間	-04:00
America/Catamarca	-03	AR	アルゼンチン標準時間	-03:00
America/Cayenne	-03	GF	フランス領ギアナ時間	-03:00
America/Cayman	EST	KY	東部標準時間	-05:00
America/Chicago	CST	US	中部標準時間	-06:00
America/Chicago	CDT	US	中部夏時間	-05:00
America/Chihuahua	MST	MX	メキシコ太平洋標準時間	-07:00
America/Chihuahua	MDT	MX	メキシコ太平洋夏時間	-06:00
America/Coral_Harbour	EST	CA	東部標準時間	-05:00
America/Cordoba	-03	AR	アルゼンチン標準時間	-03:00
America/Costa_Rica	CST	CR	中部標準時間	-06:00
America/Creston	MST	CA	山岳部標準時間	-07:00
America/Cuiaba	-04	BR	アマゾン標準時間	-04:00
America/Cuiaba	-03	BR	アマゾン夏時間	-03:00
America/Curacao	AST	CW	アトランティック標準時間	-04:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Danmarkshavn	GMT	GL	グリニッジ標準時間	00:00
America/Dawson	PST	CA	太平洋標準時間	-08:00
America/Dawson	PDT	CA	太平洋夏時間	-07:00
America/Dawson_Creek	MST	CA	山岳部標準時間	-07:00
America/Denver	MST	US	山岳部標準時間	-07:00
America/Denver	MDT	US	山岳部夏時間	-06:00
America/Detroit	EST	US	東部標準時間	-05:00
America/Detroit	EDT	US	東部夏時間	-04:00
America/Dominica	AST	DM	アトランティック標準時間	-04:00
America/Edmonton	MST	CA	山岳部標準時間	-07:00
America/Edmonton	MDT	CA	山岳部夏時間	-06:00
America/Eirunepe	-05	BR	アクレ標準時間	-05:00
America/El_Salvador	CST	SV	中部標準時間	-06:00
America/Ensenada	PST	MX	太平洋時間	-08:00
America/Ensenada	PDT	MX	太平洋夏時間	-07:00
America/Fort_Nelson	MST	CA	山岳部標準時間	-07:00
America/Fort_Wayne	EST	US	東部標準時間	-05:00
America/Fort_Wayne	EDT	US	東部夏時間	-04:00
America/Fortaleza	-03	BR	ブラジリア標準時間	-03:00
America/Glace_Bay	AST	CA	アトランティック標準時間	-04:00
America/Glace_Bay	ADT	CA	アトランティック夏時間	-03:00
America/Godthab	-03	GL	西グリーンランド標準時間	-03:00
America/Godthab	-02	GL	西グリーンランド夏時間	-02:00
America/Goose_Bay	AST	CA	アトランティック標準時間	-04:00
America/Goose_Bay	ADT	CA	アトランティック夏時間	-03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Grand_Turk	EDT	TC	東部夏時間	-04:00
America/Grand_Turk	EST	TC	東部標準時間	-05:00
America/Grenada	AST	GD	アトランティック標準時間	-04:00
America/Guadeloupe	AST	GP	アトランティック標準時間	-04:00
America/Guatemala	CST	GT	中部標準時間	-06:00
America/Guayaquil	-05	EC	エクアドル時間	-05:00
America/Guyana	-04	GY	ガイアナ時間	-04:00
America/Halifax	AST	CA	アトランティック標準時間	-04:00
America/Halifax	ADT	CA	アトランティック夏時間	-03:00
America/Havana	CST	CU	キューバ標準時間	-05:00
America/Havana	CDT	CU	キューバ夏時間	-04:00
America/Hermosillo	MST	MX	メキシコ太平洋標準時間	-07:00
America/Indiana/ Indianapolis	EST	US	東部標準時間	-05:00
America/Indiana/ Indianapolis	EDT	US	東部夏時間	-04:00
America/Indiana/Knox	CST	US	中部標準時間	-06:00
America/Indiana/Knox	CDT	US	中部夏時間	-05:00
America/Indiana/Marengo	EST	US	東部標準時間	-05:00
America/Indiana/Marengo	EDT	US	東部夏時間	-04:00
America/Indiana/ Petersburg	EST	US	東部標準時間	-05:00
America/Indiana/ Petersburg	EDT	US	東部夏時間	-04:00
America/Indiana/Tell_City	CST	US	中部標準時間	-06:00
America/Indiana/Tell_City	CDT	US	中部夏時間	-05:00
America/Indiana/Vevay	EST	US	東部標準時間	-05:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Indiana/Vevay	EDT	US	東部夏時間	-04:00
America/Indiana/Vincennes	EST	US	東部標準時間	-05:00
America/Indiana/Vincennes	EDT	US	東部夏時間	-04:00
America/Indiana/Winamac	EST	US	東部標準時間	-05:00
America/Indiana/Winamac	EDT	US	東部夏時間	-04:00
America/Indianapolis	EST	US	東部標準時間	-05:00
America/Indianapolis	EDT	US	東部夏時間	-04:00
America/Inuvik	MST	CA	山岳部標準時間	-07:00
America/Inuvik	MDT	CA	山岳部夏時間	-06:00
America/Iqaluit	EST	CA	東部標準時間	-05:00
America/Iqaluit	EDT	CA	東部夏時間	-04:00
America/Jamaica	EST	JM	東部標準時間	-05:00
America/Jujuy	-03	AR	アルゼンチン標準時間	-03:00
America/Juneau	AKST	US	アラスカ標準時間	-09:00
America/Juneau	AKDT	US	アラスカ夏時間	-08:00
America/Kentucky/Louisville	EST	US	東部標準時間	-05:00
America/Kentucky/Louisville	EDT	US	東部夏時間	-04:00
America/Kentucky/Monticello	EST	US	東部標準時間	-05:00
America/Kentucky/Monticello	EDT	US	東部夏時間	-04:00
America/Knox_IN	CST	US	中部標準時間	-06:00
America/Knox_IN	CDT	US	中部夏時間	-05:00
America/Kralendijk	AST	BQ	アトランティック標準時間	-04:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/La_Paz	-04	BO	ボリビア時間	-04:00
America/Lima	-05	PE	ペルー標準時間	-05:00
America/Los_Angeles	PST	US	太平洋標準時間	-08:00
America/Los_Angeles	PDT	US	太平洋夏時間	-07:00
America/Louisville	EST	US	東部標準時間	-05:00
America/Louisville	EDT	US	東部夏時間	-04:00
America/Lower_Princes	AST	SX	アトランティック標準時間	-04:00
America/Maceio	-03	BR	ブラジリア標準時間	-03:00
America/Managua	CST	NI	中部標準時間	-06:00
America/Manaus	-04	BR	アマゾン標準時間	-04:00
America/Marigot	AST	MF	アトランティック標準時間	-04:00
America/Martinique	AST	MQ	アトランティック標準時間	-04:00
America/Matamoros	CST	MX	中部標準時間	-06:00
America/Matamoros	CDT	MX	中部夏時間	-05:00
America/Mazatlan	MST	MX	メキシコ太平洋標準時間	-07:00
America/Mazatlan	MDT	MX	メキシコ太平洋夏時間	-06:00
America/Mendoza	-03	AR	アルゼンチン標準時間	-03:00
America/Menominee	CST	US	中部標準時間	-06:00
America/Menominee	CDT	US	中部夏時間	-05:00
America/Merida	CST	MX	中部標準時間	-06:00
America/Merida	CDT	MX	中部夏時間	-05:00
America/Metlakatla	AKST	US	アラスカ標準時間	-09:00
America/Metlakatla	AKDT	US	アラスカ夏時間	-08:00
America/Mexico_City	CST	MX	中部標準時間	-06:00
America/Mexico_City	CDT	MX	中部夏時間	-05:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Miquelon	-03	PM	サンピエール島とミクロン島標準時	-03:00
America/Miquelon	-02	PM	サンピエール島とミクロン島夏時間	-02:00
America/Moncton	AST	CA	アトランティック標準時間	-04:00
America/Moncton	ADT	CA	アトランティック夏時間	-03:00
America/Monterrey	CST	MX	中部標準時間	-06:00
America/Monterrey	CDT	MX	中部夏時間	-05:00
America/Montevideo	-03	UY	ウルグアイ標準時間	-03:00
America/Montreal	EST	CA	東部標準時間	-05:00
America/Montreal	EDT	CA	東部夏時間	-04:00
America/Montserrat	AST	MS	アトランティック標準時間	-04:00
America/Nassau	EST	BS	東部標準時間	-05:00
America/Nassau	EDT	BS	東部夏時間	-04:00
America/New_York	EST	US	東部標準時間	-05:00
America/New_York	EDT	US	東部夏時間	-04:00
America/Nipigon	EST	CA	東部標準時間	-05:00
America/Nipigon	EDT	CA	東部夏時間	-04:00
America/Nome	AKST	US	アラスカ標準時間	-09:00
America/Nome	AKDT	US	アラスカ夏時間	-08:00
America/Noronha	-02	BR	フェルナンド・デ・ノローニャ標準時間	-02:00
America/North_Dakota/Beulah	CST	US	中部標準時間	-06:00
America/North_Dakota/Beulah	CDT	US	中部夏時間	-05:00
America/North_Dakota/Center	CST	US	中部標準時間	-06:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/North_Dakota/Center	CDT	US	中部夏時間	-05:00
America/North_Dakota/New_Salem	CST	US	中部標準時間	-06:00
America/North_Dakota/New_Salem	CDT	US	中部夏時間	-05:00
America/Ojinaga	MST	MX	山岳部標準時間	-07:00
America/Ojinaga	MDT	MX	山岳部夏時間	-06:00
America/Panama	EST	PA	東部標準時間	-05:00
America/Pangnirtung	EST	CA	東部標準時間	-05:00
America/Pangnirtung	EDT	CA	東部夏時間	-04:00
America/Paramaribo	-03	SR	スリナム時間	-03:00
America/Phoenix	MST	US	山岳部標準時間	-07:00
America/Port-au-Prince	EST	HT	東部標準時間	-05:00
America/Port-au-Prince	EDT	HT	東部夏時間	-04:00
America/Port_of_Spain	AST	TT	アトランティック標準時間	-04:00
America/Porto_Acre	-05	BR	アクレ標準時間	-05:00
America/Porto_Velho	-04	BR	アマゾン標準時間	-04:00
America/Puerto_Rico	AST	PR	アトランティック標準時間	-04:00
America/Punta_Arenas	-03	CL	チリ標準時間	-03:00
America/Rainy_River	CST	CA	中部標準時間	-06:00
America/Rainy_River	CDT	CA	中部夏時間	-05:00
America/Rankin_Inlet	CST	CA	中部標準時間	-06:00
America/Rankin_Inlet	CDT	CA	中部夏時間	-05:00
America/Recife	-03	BR	ブラジリア標準時間	-03:00
America/Regina	CST	CA	中部標準時間	-06:00
America/Resolute	CST	CA	中部標準時間	-06:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Resolute	CDT	CA	中部夏時間	-05:00
America/Rio_Branco	-05	BR	アクレ標準時間	-05:00
America/Rosario	-03	AR	アルゼンチン標準時間	-03:00
America/Santa_Isabel	PST	MX	太平洋標準時間	-08:00
America/Santa_Isabel	PDT	MX	太平洋夏時間	-07:00
America/Santarem	-03	BR	ブラジリア標準時間	-03:00
America/Santiago	-04	CL	チリ標準時間	-04:00
America/Santiago	-03	CL	チリ夏時間	-03:00
America/Santo_Domingo	AST	DO	アトランティック標準時間	-04:00
America/Sao_Paulo	-03	BR	ブラジリア標準時間	-03:00
America/Sao_Paulo	-02	BR	ブラジリア夏時間	-02:00
America/Scoresbysund	-01	GL	東グリーンランド標準時間	-01:00
America/Scoresbysund	+00	GL	東グリーンランド夏時間	00:00
America/Shiprock	MST	US	山岳部標準時間	-07:00
America/Shiprock	MDT	US	山岳部夏時間	-06:00
America/Sitka	AKST	US	アラスカ標準時間	-09:00
America/Sitka	AKDT	US	アラスカ夏時間	-08:00
America/St_Barthelemy	AST	BL	アトランティック標準時間	-04:00
America/St_Johns	NST	CA	ニューファンドランド標準時間	-03:30
America/St_Johns	NDT	CA	ニューファンドランド夏時間	-02:30
America/St_Kitts	AST	KN	アトランティック標準時間	-04:00
America/St_Lucia	AST	LC	アトランティック標準時間	-04:00
America/St_Thomas	AST	VI	アトランティック標準時間	-04:00
America/St_Vincent	AST	VC	アトランティック標準時間	-04:00
America/Swift_Current	CST	CA	中部標準時間	-06:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
America/Tegucigalpa	CST	HN	中部標準時間	-06:00
America/Thule	AST	GL	アトランティック標準時間	-04:00
America/Thule	ADT	GL	アトランティック夏時間	-03:00
America/Thunder_Bay	EST	CA	東部標準時間	-05:00
America/Thunder_Bay	EDT	CA	東部夏時間	-04:00
America/Tijuana	PST	MX	太平洋標準時間	-08:00
America/Tijuana	PDT	MX	太平洋夏時間	-07:00
America/Toronto	EST	CA	東部標準時間	-05:00
America/Toronto	EDT	CA	東部夏時間	-04:00
America/Tortola	AST	VG	アトランティック標準時間	-04:00
America/Vancouver	PST	CA	太平洋標準時間	-08:00
America/Vancouver	PDT	CA	太平洋夏時間	-07:00
America/Virgin	AST	AI	アトランティック標準時間	-04:00
America/Whitehorse	PST	CA	太平洋標準時間	-08:00
America/Whitehorse	PDT	CA	太平洋夏時間	-07:00
America/Winnipeg	CST	CA	中部標準時間	-06:00
America/Winnipeg	CDT	CA	中部夏時間	-05:00
America/Yakutat	AKST	US	アラスカ標準時間	-09:00
America/Yakutat	AKDT	US	アラスカ夏時間	-08:00
America/Yellowknife	MST	CA	山岳部標準時間	-07:00
America/Yellowknife	MDT	CA	山岳部夏時間	-06:00

エリア: Antarctica(南極)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Antarctica/Casey	+08	AQ	オーストラリア西標準時間	08:00
Antarctica/Davis	+07	AQ	デービス時間	07:00
Antarctica/DumontDURville	+10	AQ	デュモン・デュルヴィル時間	10:00
Antarctica/Macquarie	+11	AU	マッコリー島時間	11:00
Antarctica/Mawson	+05	AQ	モーソン時間	05:00
Antarctica/McMurdo	NZST	AQ	ニュージーランド標準時間	12:00
Antarctica/McMurdo	NZDT	AQ	ニュージーランド夏時間	13:00
Antarctica/Palmer	-03	AQ	チリ標準時間	-04:00
Antarctica/Palmer	CLST	AQ	チリ夏時間	-03:00
Antarctica/Rothera	-03	AQ	ロゼラ時間	-03:00
Antarctica/South_Pole	NZST	AQ	ニュージーランド標準時間	12:00
Antarctica/South_Pole	NZDT	AQ	ニュージーランド夏時間	13:00
Antarctica/Syowa	+03	AQ	昭和時間	03:00
Antarctica/Troll	+00	AQ	グリニッジ標準時間	00:00
Antarctica/Troll	+02	AQ	グリニッジ標準時間	02:00
Antarctica/Vostok	+06	AQ	ボストーク時間	06:00

エリア: Arctic(北極)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Arctic/Longyearbyen	CET	SJ	中央ヨーロッパ標準時間	01:00
Arctic/Longyearbyen	CEST	SJ	中央ヨーロッパ夏時間	02:00

エリア: Asia(アジア)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Aden	+03	YE	アラビア標準時間	03:00
Asia/Almaty	+06	KZ	東カザフスタン時間	06:00
Asia/Amman	EET	JO	東ヨーロッパ標準時間	02:00
Asia/Amman	EEST	JO	東ヨーロッパ夏時間	03:00
Asia/Anadyr	+12	RU	アナディリ標準時間	12:00
Asia/Aqtau	+05	KZ	西カザフスタン時間	05:00
Asia/Aqtobe	+05	KZ	西カザフスタン時間	05:00
Asia/Ashgabat	+05	TM	トルクメニアン標準時間	05:00
Asia/Ashkhabad	+05	TM	トルクメニアン標準時間	05:00
Asia/Atyrau	+05	KZ	西カザフスタン時間	05:00
Asia/Baghdad	+03	IQ	アラビア標準時間	03:00
Asia/Bahrain	+03	BH	アラビア標準時間	03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Baku	+04	AZ	アゼルバイジャン標準時間	04:00
Asia/Baku	+05	AZ	アゼルバイジャン時間	05:00
Asia/Bangkok	+07	TH	インドシナ時間	07:00
Asia/Barnaul	+07	RU	クラスノヤルスク時間	07:00
Asia/Beijing	CST	CN	中国時間	08:00
Asia/Beirut	EET	LB	東ヨーロッパ標準時間	02:00
Asia/Beirut	EEST	LB	東ヨーロッパ夏時間	03:00
Asia/Bishkek	+06	KG	キルギスタン時間	06:00
Asia/Brunei	+08	BN	ブルネイ・ダルッサラム時間	08:00
Asia/Calcutta	IST	IN	インド標準時間	05:30
Asia/Chita	+09	RU	ヤクーツク標準時間	09:00
Asia/Choibalsan	+08	MN	チョイバルサン標準時間	08:00
Asia/Choibalsan	+09	MN	チョイバルサン夏時間	09:00
Asia/Chongqing	CST	CN	中国標準時間	08:00
Asia/Chungking	CST	CN	中国標準時間	08:00
Asia/Colombo	+0530	LK	インド標準時間	05:30
Asia/Dacca	+06	BD	バングラデシュ標準時間	06:00
Asia/Damascus	EET	SY	東ヨーロッパ標準時間	02:00
Asia/Damascus	EEST	SY	東ヨーロッパ夏時間	03:00
Asia/Dhaka	+06	BD	バングラデシュ標準時間	06:00
Asia/Dili	+09	TL	東ティモール時間	09:00
Asia/Dubai	+04	AE	(ペルシア)湾標準時間	04:00
Asia/Dushanbe	+05	TJ	タジキスタン時間	05:00
Asia/Famagusta	EET	CY	東ヨーロッパ標準時間	02:00
Asia/Famagusta	EEST	CY	東ヨーロッパ夏時間	03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Gaza	EET	PS	東ヨーロッパ標準時間	02:00
Asia/Gaza	EEST	PS	東ヨーロッパ夏時間	03:00
Asia/Harbin	CST	CN	中国標準時間	08:00
Asia/Hebron	EET	PS	東ヨーロッパ標準時間	02:00
Asia/Hebron	EEST	PS	東ヨーロッパ夏時間	03:00
Asia/Ho_Chi_Minh	+07	VN	インドシナ時間	07:00
Asia/Hong_Kong	HKT	HK	香港標準時間	08:00
Asia/Hovd	+07	MN	ホブド標準時間	07:00
Asia/Hovd	+08	MN	ホブド夏時間	08:00
Asia/Irkutsk	+08	RU	イルクーツク標準時間	08:00
Asia/Istanbul	+03	TR	東ヨーロッパ時間	03:00
Asia/Jakarta	WIB	ID	西インドネシア時間	07:00
Asia/Jayapura	WIT	ID	東インドネシア時間	09:00
Asia/Jerusalem	IST	IL	イスラエル標準時間	02:00
Asia/Jerusalem	IDT	IL	イスラエル夏時間	03:00
Asia/Kabul	+0430	AF	アフガニスタン時間	04:30
Asia/Kamchatka	+12	RU	ペトロパブロフスク・カムチャツキー標準時間	12:00
Asia/Karachi	PKT	PK	パキスタン標準時間	05:00
Asia/Kashgar	+06	CN	中国時間	06:00
Asia/Katmandu	+0545	NP	ネパール時間	05:45
Asia/Khandyga	+09	RU	ヤクーツク標準時間	09:00
Asia/Kolkata	IST	IN	インド標準時間	05:30
Asia/Krasnoyarsk	+07	RU	クラスノヤルスク標準時間	08:00
Asia/Kuala_Lumpur	+08	MY	マレーシア時間	08:00
Asia/Kuching	+08	MY	マレーシア時間	08:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Kuwait	+03	KW	アラビア標準時間	03:00
Asia/Macao	CST	MO	中国標準時間	08:00
Asia/Macau	CST	MO	中国標準時間	08:00
Asia/Magadan	+11	RU	マガダン標準時間	11:00
Asia/Makassar	WITA	ID	中央インドネシア時間	08:00
Asia/Manila	PST	PH	フィリピン標準時間	08:00
Asia/Muscat	+04	OM	(ペルシア)湾標準時間	04:00
Asia/Nicosia	EET	CY	東ヨーロッパ標準時間	02:00
Asia/Nicosia	EEST	CY	東ヨーロッパ夏時間	03:00
Asia/Novokuznetsk	+07	RU	クラスノヤルスク標準時	07:00
Asia/Novosibirsk	+07	RU	ノヴォシビルスク標準時間	07:00
Asia/Omsk	+06	RU	オムスク標準時間	06:00
Asia/Qostanay	+06	KZ	東カザフスタン時間	06:00
Asia/Oral	+05	KZ	西カザフスタン時間	05:00
Asia/Osaka	JST	JP	日本時間	09:00
Asia/Phnom_Penh	+07	KH	インドシナ時間	07:00
Asia/Pontianak	WIB	ID	西インドネシア時間	07:00
Asia/Pyongyang	KST	KP	韓国標準時間	09:00
Asia/Qatar	+03	QA	アラビア標準時間	03:00
Asia/Qyzylorda	+05	KZ	西カザフスタン時間	05:00
Asia/Rangoon	+0630	MM	ミャンマー時間	06:30
Asia/Riyadh	+03	SA	アラビア標準時間	03:00
Asia/Saigon	+07	VN	インドシナ時間	07:00
Asia/Sakhalin	+11	RU	サハリン標準時間	11:00
Asia/Samarkand	+05	UZ	ウズベキスタン標準時間	05:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Sapporo	JST	JP	日本時間	09:00
Asia/Seoul	KST	KR	韓国標準時間	09:00
Asia/Shanghai	CST	CN	中国標準時間	08:00
Asia/Singapore	+08	SG	シンガポール標準時間	08:00
Asia/Srednekolymsk	+11	RU	マガダン標準時間	11:00
Asia/Taipei	CST	TW	台北標準時間	08:00
Asia/Tashkent	+05	UZ	ウズベキスタン標準時間	05:00
Asia/Tbilisi	+04	GE	グルジア標準時間	04:00
Asia/Tehran	+0330	IR	イラン標準時間	03:30
Asia/Tehran	+0430	IR	イラン夏時間	04:30
Asia/Tel_Aviv	IST	IL	イスラエル標準時間	02:00
Asia/Tel_Aviv	IDT	IL	イスラエル夏時間	03:00
Asia/Thimbu	+06	BT	ブータン時間	06:00
Asia/Thimphu	BTT	BT	ブータン時間	06:00
Asia/Tokyo	JST	JP	日本標準時間	09:00
Asia/Tomsk	+07	RU	オムスク時間	07:00
Asia/Ujung_Pandang	WITA	ID	中央インドネシア時間	08:00
Asia/Ulan_Bator	+08	MN	ウランバートルバートル標準時	08:00
Asia/Urumqi	XJT	CN	中国標準時間	08:00
Asia/Ust-Nera	+10	RU	ウラジオストック標準時間	10:00
Asia/Vientiane	+07	LA	インドシナ時間	07:00
Asia/Vladivostok	+10	RU	ウラジオストック標準時間	10:00
Asia/Yakutsk	+09	RU	ヤクーツク標準時間	09:00
Asia/Yangon	+0630	MM		06:30
Asia/Yekaterinburg	+05	RU	エカテリンブルク標準時間	05:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Asia/Yerevan	+04	AM	アルメニア標準時間	04:00
Asia/Yerevan	+05	AM	アルメニア時間	05:00

エリア: Atlantic(大西洋)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Atlantic/Azores	AZOT	PT	アゾレス時間	-01:00
Atlantic/Azores	+00	PT	アゾレス夏標準時	00:00
Atlantic/Bermuda	AST	BM	アトランティック標準時間	-04:00
Atlantic/Bermuda	ADT	BM	アトランティック夏時間	-03:00
Atlantic/Canary	WET	ES	西ヨーロッパ標準時間	00:00
Atlantic/Canary	WEST	ES	西ヨーロッパ夏時間	01:00
Atlantic/Cape_Verde	-01	CV	カーボベルデ標準時間	-01:00
Atlantic/Faeroe	WET	FO	西ヨーロッパ標準時間	00:00
Atlantic/Faeroe	WEST	FO	西ヨーロッパ夏時間	01:00
Atlantic/Faroe	WET	FO	西ヨーロッパ時間	00:00
Atlantic/Faroe	WEST	FO	西ヨーロッパ夏時間	01:00
Atlantic/Jan_Mayen	CET	SJ	中央ヨーロッパ標準時間	01:00
Atlantic/Jan_Mayen	CEST	SJ	中央ヨーロッパ夏時間	02:00
Atlantic/Madeira	WET	PT	西ヨーロッパ標準時間	00:00
Atlantic/Madeira	WEST	PT	西ヨーロッパ夏時間	01:00
Atlantic/Reykjavik	GMT	IS	グリニッジ標準時間	00:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Atlantic/South_Georgia	-02	GS	サウスジョージア時間	-02:00
Atlantic/St_Helena	GMT	SH	グリニッジ標準時間	00:00
Atlantic/Stanley	-03	FK	フォークランド諸島標準時間	-03:00

エリア: Australia(オーストラリア)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Australia/ACT	AEST	AU	オーストラリア東部標準時間	10:00
Australia/ACT	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/Adelaide	ACST	AU	オーストラリア中部標準時間	09:30
Australia/Adelaide	ACDT	AU	オーストラリア中部夏時間	10:30
Australia/Brisbane	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Broken_Hill	ACST	AU	オーストラリア中部標準時間	09:30
Australia/Broken_Hill	ACDT	AU	オーストラリア中部夏時間	10:30
Australia/Canberra	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Canberra	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/Currie	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Currie	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/Darwin	ACST	AU	オーストラリア中部標準時間	09:30
Australia/Eucla	+0845	AU	オーストラリア中西部標準時間	08:45
Australia/Hobart	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Hobart	AEDT	AU	オーストラリア東部夏時間	11:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Australia/LHI	+1030	AU	ロード・ハウ標準時間	10:30
Australia/LHI	+11	AU	ロード・ハウ夏時間	11:00
Australia/Lindeman	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Lord_Howe	+1030	AU	ロード・ハウ標準時間	10:30
Australia/Lord_Howe	+11	AU	ロード・ハウ夏時間	11:00
Australia/Melbourne	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Melbourne	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/NSW	AEST	AU	オーストラリア東部標準時間	10:00
Australia/NSW	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/North	ACST	AU	オーストラリア中部標準時間	09:30
Australia/Perth	AWST	AU	オーストラリア西標準時間	08:00
Australia/Queensland	AEST	AU	オーストラリア東部標準時間	10:00
Australia/South	ACST	AU	オーストラリア中部標準時間	09:30
Australia/South	ACDT	AU	オーストラリア中部夏時間	10:30
Australia/Sydney	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Sydney	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/Tasmania	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Tasmania	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/Victoria	AEST	AU	オーストラリア東部標準時間	10:00
Australia/Victoria	AEDT	AU	オーストラリア東部夏時間	11:00
Australia/West	AWST	AU	オーストラリア西標準時間	08:00
Australia/Yancowinna	ACST	AU	オーストラリア中部標準時間	09:30
Australia/Yancowinna	ACDT	AU	オーストラリア中部夏時間	10:30

エリア: その他

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Brazil/Acre	-05	BR	アクレ標準時間	-05:00
Brazil/DeNoronha	-02	BR	フェルナンド・デ・ノローニャ標準時間	-02:00
Brazil/East	-03	BR	ブラジリア標準時間	-03:00
Brazil/East	-02	BR	ブラジリア夏時間	-02:00
Brazil/West	-04	BR	アマゾン標準時間	-04:00
CST6CDT	CST	US	中部標準時間	-06:00
CST6CDT	CDT	US	中部夏時間	-05:00
Canada/Atlantic	AST	CA	アトランティック標準時間	-04:00
Canada/Atlantic	ADT	CA	アトランティック夏時間	-03:00
Canada/Central	CST	CA	中部標準時間	-06:00
Canada/Central	CDT	CA	中部夏時間	-05:00
Canada/Eastern	EST	CA	東部標準時間	-05:00
Canada/Eastern	EDT	CA	東部夏時間	-04:00
Canada/Mountain	MST	CA	山岳部標準時間	-07:00
Canada/Mountain	MDT	CA	山岳部夏時間	-06:00
Canada/Newfoundland	NST	CA	ニューファンドランド標準時間	-03:30
Canada/Newfoundland	NDT	CA	ニューファンドランド夏時間	-02:30
Canada/Pacific	PST	CA	太平洋標準時間	-08:00
Canada/Pacific	PDT	CA	太平洋夏時間	-07:00
Canada/Saskatchewan	CST	CA	中部標準時間	-06:00
Canada/Yukon	PST	CA	太平洋標準時間	-08:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Canada/Yukon	PDT	CA	太平洋夏時間	-07:00
Chile/Continental	-04	CL	チリ標準時間	-04:00
Chile/Continental	-03	CL	チリ夏時間	-03:00
Chile/EasterIsland	-06	CL	イースター島標準時間	-06:00
Chile/EasterIsland	-05	CL	イースター島夏時間	-05:00
Cuba	CST	CU	キューバ標準時間	-05:00
Cuba	CDT	CU	キューバ夏時間	-04:00
EST5EDT	EST	US	東部標準時間	-05:00
EST5EDT	EDT	US	東部夏時間	-04:00
Egypt	EET	EG	東ヨーロッパ標準時間	02:00
Eire	GMT	IE	グリニッジ標準時間	00:00
Eire	IST	IE	グリニッジ標準時間	01:00
Etc/GMT	GMT		グリニッジ標準時間	00:00
Etc/GMT+0	GMT		グリニッジ標準時間	00:00
Etc/GMT+1	GMT-01		GMT より 1 時間遅い	-01:00
Etc/GMT+2	GMT-02		GMT より 2 時間遅い	-02:00
Etc/GMT+3	GMT-03		GMT より 3 時間遅い	-03:00
Etc/GMT+4	GMT-04		GMT より 4 時間遅い	-04:00
Etc/GMT+5	GMT-05		GMT より 5 時間遅い	-05:00
Etc/GMT+6	GMT-06		GMT より 6 時間遅い	-06:00
Etc/GMT+7	GMT-07		GMT より 7 時間遅い	-07:00
Etc/GMT+8	GMT-08		GMT より 8 時間遅い	-08:00
Etc/GMT+9	GMT-09		GMT より 9 時間遅い	-09:00
Etc/GMT+10	GMT-10		GMT より 10 時間遅い	-10:00
Etc/GMT+11	GMT-11		GMT より 11 時間遅い	-11:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Etc/GMT+12	GMT-12		GMT より 12 時間遅い	-12:00
Etc/GMT-0	GMT		グリニッジ標準時間	00:00
Etc/GMT-1	GMT+01		GMT より 1 時間早い	01:00
Etc/GMT-2	GMT+02		GMT より 2 時間早い	02:00
Etc/GMT-3	GMT+03		GMT より 3 時間早い	03:00
Etc/GMT-4	GMT+04		GMT より 4 時間早い	04:00
Etc/GMT-5	GMT+05		GMT より 5 時間早い	05:00
Etc/GMT-6	GMT+06		GMT より 6 時間早い	06:00
Etc/GMT-7	GMT+07		GMT より 7 時間早い	07:00
Etc/GMT-8	GMT+08		GMT より 8 時間早い	08:00
Etc/GMT-9	GMT+09		GMT より 9 時間早い	09:00
Etc/GMT-10	GMT+10		GMT より 10 時間早い	10:00
Etc/GMT-11	GMT+11		GMT より 11 時間早い	11:00
Etc/GMT-12	GMT+12		GMT より 12 時間早い	12:00
Etc/GMT-13	GMT+13		GMT より 13 時間早い	13:00
Etc/GMT-14	GMT+14		GMT より 14 時間早い	14:00
Etc/GMT0	GMT		グリニッジ標準時間	00:00
Etc/Greenwich	GMT		グリニッジ標準時間	00:00
Etc/UCT	UCT		グリニッジ標準時間	00:00
Etc/UTC	UTC		グリニッジ標準時間	00:00
Etc/Universal	UTC		グリニッジ標準時間	00:00
Etc/Zulu	UTC		グリニッジ標準時間	00:00
GB	GMT	GB	グリニッジ標準時間	00:00
GB	BST	GB	グリニッジ標準時間	01:00
GB-Eire	GMT	GB	グリニッジ標準時間	00:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
GB-Eire	BST	GB	グリニッジ標準時間	01:00
GMT	GMT		グリニッジ標準時間	00:00
GMT+0	GMT		グリニッジ標準時間	00:00
GMT-0	GMT		グリニッジ標準時間	00:00
GMT0	GMT		グリニッジ標準時間	00:00
Greenwich	GMT		グリニッジ標準時間	00:00
Hongkong	HKT	HK	香港標準時間	08:00
Iceland	GMT	IS	グリニッジ標準時間	00:00
Indian/Antananarivo	EAT	MG	東アフリカ時間	03:00
Indian/Chagos	+06	IO	インド洋時間	06:00
Indian/Christmas	+07	CX	クリスマス島時間	07:00
Indian/Cocos	+0630	CC	ココス島時間	06:30
Indian/Comoro	EAT	KM	東アフリカ時間	03:00
Indian/Kerguelen	+05	TF	フランス領南方および南極時代	05:00
Indian/Mahe	+04	SC	セーシェル時間	04:00
Indian/Maldives	+05	MV	モルディブ時間	05:00
Indian/Mauritius	+04	MU	モーリシャス標準時間	04:00
Indian/Mayotte	EAT	YT	東アフリカ時間	03:00
Indian/Reunion	+04	RE	レユニオン時間	04:00
Iran	+0330	IR	イラン標準時間	03:30
Iran	+0430	IR	イラン夏時間	04:30
Israel	IST	IL	イスラエル標準時間	02:00
Israel	IDT	IL	イスラエル夏時間	03:00
Jamaica	EST	JM	東部標準時間	-05:00
Japan	JST	JP	日本標準時間	09:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Kwajalein	+12	MH	マーシャル諸島時間	12:00
Libya	EET	LY	東ヨーロッパ標準時間	02:00
MST7MDT	MST	US	山岳部標準時間	-07:00
MST7MDT	MDT	US	山岳部夏時間	-06:00
Mexico/BajaNorte	PST	MX	メキシコ太平洋標準時間	-08:00
Mexico/BajaNorte	PDT	MX	メキシコ太平洋夏時間	-07:00
Mexico/BajaSur	MST	MX	メキシコ太平洋標準時間	-07:00
Mexico/BajaSur	MDT	MX	メキシコ太平洋夏時間	-06:00
Mexico/General	CST	MX	中部標準時間	-06:00
Mexico/General	CDT	MX	中部夏時間	-05:00
NZ	NZST	AQ	ニュージーランド標準時間	12:00
NZ	NZDT	AQ	ニュージーランド夏時間	13:00
NZ-CHAT	+1245	NZ	チャタム標準時間	12:45
NZ-CHAT	+1345	NZ	チャタム夏時間	13:45
Navajo	MST	US	山岳部標準時間	-07:00
Navajo	MDT	US	山岳部夏時間	-06:00
PRC	CST	CN	中国標準時間	08:00
PST8PDT	PST	US	太平洋標準時間	-08:00
PST8PDT	PDT	US	太平洋夏時間	-07:00
Poland	CET	PL	中央ヨーロッパ標準時間	01:00
Poland	CEST	PL	中央ヨーロッパ夏時間	02:00
Portugal	WET	PT	西ヨーロッパ標準時間	00:00
Portugal	WEST	PT	西ヨーロッパ夏時間	01:00
ROC	CST	TW	台北標準時間	08:00
ROK	KST	KR	韓国標準時間	09:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Singapore	+08	SG	シンガポール標準時間	08:00
Turkey	+03	TR	東ヨーロッパ時間	03:00
UCT	UCT		グリニッジ標準時間	00:00
US/Alaska	AKST	US	アラスカ標準時間	-09:00
US/Alaska	AKDT	US	アラスカ夏時間	-08:00
US/Aleutian	HST	US	ハワイ・アリューシャン標準時間	-10:00
US/Aleutian	HDT	US	ハワイ・アリューシャン夏時間	-09:00
US/Arizona	MST	US	山岳部標準時間	-07:00
US/Central	CST	US	中部標準時間	-06:00
US/Central	CDT	US	中部夏時間	-05:00
US/East-Indiana	EST	US	東部標準時間	-05:00
US/East-Indiana	EDT	US	東部夏時間	-04:00
US/Eastern	EST	US	東部標準時間	-05:00
US/Eastern	EDT	US	東部夏時間	-04:00
US/Hawaii	HST	US	ハワイ・アリューシャン標準時間	-10:00
US/Indiana-Starke	CST	US	中部標準時間	-06:00
US/Indiana-Starke	CDT	US	中部夏時間	-05:00
US/Michigan	EST	US	東部標準時間	-05:00
US/Michigan	EDT	US	東部夏時間	-04:00
US/Mountain	MST	US	山岳部標準時間	-07:00
US/Mountain	MDT	US	山岳部夏時間	-06:00
US/Pacific	PST	US	太平洋標準時間	-08:00
US/Pacific	PDT	US	太平洋夏時間	-07:00
US/Pacific-New	PST	US	太平洋標準時間	-08:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
US/Pacific-New	PDT	US	太平洋夏時間	-07:00
US/Samoa	SST	UM	サモア標準時間	-11:00
UTC	UTC		グリニッジ標準時間	00:00
Universal	UTC		グリニッジ標準時間	00:00
W-SU	MSK	RU	モスクワ標準時間	03:00
Zulu	UTC		グリニッジ標準時間	00:00

エリア: Europe(ヨーロッパ)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Amsterdam	CET	NL	中央ヨーロッパ標準時間	01:00
Europe/Amsterdam	CEST	NL	中央ヨーロッパ夏時間	02:00
Europe/Andorra	CET	AD	中央ヨーロッパ標準時間	01:00
Europe/Andorra	CEST	AD	中央ヨーロッパ夏時間	02:00
Europe/Astrakhan	+04	RU	モスクワ標準時間	04:00
Europe/Athens	EET	GR	東ヨーロッパ標準時間	02:00
Europe/Athens	EEST	GR	東ヨーロッパ夏時間	03:00
Europe/Belfast	GMT	GB	グリニッジ標準時間	00:00
Europe/Belfast	BST	GB	グリニッジ標準時間	01:00
Europe/Belgrade	CET	RS	中央ヨーロッパ標準時間	01:00
Europe/Belgrade	CEST	RS	中央ヨーロッパ夏時間	02:00
Europe/Berlin	CET	DE	中央ヨーロッパ標準時間	01:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Berlin	CEST	DE	中央ヨーロッパ夏時間	02:00
Europe/Bratislava	CET	SK	中央ヨーロッパ標準時間	01:00
Europe/Bratislava	CEST	SK	中央ヨーロッパ夏時間	02:00
Europe/Brussels	CET	BE	中央ヨーロッパ標準時間	01:00
Europe/Brussels	CEST	BE	中央ヨーロッパ夏時間	02:00
Europe/Bucharest	EET	RO	東ヨーロッパ標準時間	02:00
Europe/Bucharest	EEST	RO	東ヨーロッパ夏時間	03:00
Europe/Budapest	CET	HU	中央ヨーロッパ標準時間	01:00
Europe/Budapest	CEST	HU	中央ヨーロッパ夏時間	02:00
Europe/Busingen	CET	DE	中央ヨーロッパ標準時間	01:00
Europe/Busingen	CEST	DE	中央ヨーロッパ夏時間	02:00
Europe/Chisinau	EET	MD	東ヨーロッパ標準時間	02:00
Europe/Chisinau	EEST	MD	東ヨーロッパ夏時間	03:00
Europe/Copenhagen	CET	DK	中央ヨーロッパ標準時間	01:00
Europe/Copenhagen	CEST	DK	中央ヨーロッパ夏時間	02:00
Europe/Dublin	GMT	IE	グリニッジ標準時間	00:00
Europe/Dublin	IST	IE	グリニッジ標準時間	01:00
Europe/Gibraltar	CET	GI	中央ヨーロッパ標準時間	01:00
Europe/Gibraltar	CEST	GI	中央ヨーロッパ夏時間	02:00
Europe/Guernsey	GMT	GG	グリニッジ標準時間	00:00
Europe/Guernsey	BST	GG	グリニッジ標準時間	01:00
Europe/Helsinki	EET	FI	東ヨーロッパ標準時間	02:00
Europe/Helsinki	EEST	FI	東ヨーロッパ夏時間	03:00
Europe/Isle_of_Man	GMT	IM	グリニッジ標準時間	00:00
Europe/Isle_of_Man	BST	IM	グリニッジ標準時間	01:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Istanbul	+03	TR	東ヨーロッパ時間	03:00
Europe/Jersey	GMT	JE	グリニッジ標準時間	00:00
Europe/Jersey	BST	JE	グリニッジ標準時間	01:00
Europe/Kaliningrad	EET	RU	東ヨーロッパ標準時間	02:00
Europe/Kiev	EET	UA	東ヨーロッパ標準時間	02:00
Europe/Kiev	EEST	UA	東ヨーロッパ夏時間	03:00
Europe/Kirov	+03	RU	モスクワ時間	03:00
Europe/Lisbon	WET	PT	西ヨーロッパ標準時間	00:00
Europe/Lisbon	WEST	PT	西ヨーロッパ夏時間	01:00
Europe/Ljubljana	CET	SI	中央ヨーロッパ標準時間	01:00
Europe/Ljubljana	CEST	SI	中央ヨーロッパ夏時間	02:00
Europe/London	GMT	GB	グリニッジ標準時間	00:00
Europe/London	BST	GB	グリニッジ標準時間	01:00
Europe/Luxembourg	CET	LU	中央ヨーロッパ標準時間	01:00
Europe/Luxembourg	CEST	LU	中央ヨーロッパ夏時間	02:00
Europe/Madrid	CET	ES	中央ヨーロッパ標準時間	01:00
Europe/Madrid	CEST	ES	中央ヨーロッパ夏時間	02:00
Europe/Malta	CET	MT	中央ヨーロッパ標準時間	01:00
Europe/Malta	CEST	MT	中央ヨーロッパ夏時間	02:00
Europe/Mariehamn	EET	AX	東ヨーロッパ標準時間	02:00
Europe/Mariehamn	EEST	AX	東ヨーロッパ夏時間	03:00
Europe/Minsk	+03	BY	モスクワ標準時間	03:00
Europe/Monaco	CET	MC	中央ヨーロッパ標準時間	01:00
Europe/Monaco	CEST	MC	中央ヨーロッパ夏時間	02:00
Europe/Moscow	MSK	RU	モスクワ標準時間	03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Nicosia	EET	CY	東ヨーロッパ標準時間	02:00
Europe/Nicosia	EEST	CY	東ヨーロッパ夏時間	03:00
Europe/Oslo	CET	NO	中央ヨーロッパ標準時間	01:00
Europe/Oslo	CEST	NO	中央ヨーロッパ夏時間	02:00
Europe/Paris	CET	FR	中央ヨーロッパ時間	01:00
Europe/Paris	CEST	FR	中央ヨーロッパ夏時間	02:00
Europe/Podgorica	CET	ME	中央ヨーロッパ標準時間	01:00
Europe/Podgorica	CEST	ME	中央ヨーロッパ夏時間	02:00
Europe/Prague	CET	CZ	中央ヨーロッパ標準時間	01:00
Europe/Prague	CEST	CZ	中央ヨーロッパ夏時間	02:00
Europe/Riga	EET	LV	東ヨーロッパ標準時間	02:00
Europe/Riga	EEST	LV	東ヨーロッパ夏時間	03:00
Europe/Rome	CET	IT	中央ヨーロッパ標準時間	01:00
Europe/Rome	CEST	IT	中央ヨーロッパ夏時間	02:00
Europe/Samara	+04	RU	サマラ標準時間	04:00
Europe/San_Marino	CET	SM	中央ヨーロッパ標準時間	01:00
Europe/San_Marino	CEST	SM	中央ヨーロッパ夏時間	02:00
Europe/Sarajevo	CET	BA	中央ヨーロッパ標準時間	01:00
Europe/Sarajevo	CEST	BA	中央ヨーロッパ夏時間	02:00
Europe/Saratov	+04	RU	モスクワ標準時間	04:00
Europe/Simferopol	MSK	UA	モスクワ標準時	03:00
Europe/Skopje	CET	MK	中央ヨーロッパ標準時間	01:00
Europe/Skopje	CEST	MK	中央ヨーロッパ夏時間	02:00
Europe/Sofia	EET	BG	東ヨーロッパ標準時間	02:00
Europe/Sofia	EEST	BG	東ヨーロッパ夏時間	03:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Stockholm	CET	SE	中央ヨーロッパ標準時間	01:00
Europe/Stockholm	CEST	SE	中央ヨーロッパ夏時間	02:00
Europe/Tallinn	EET	EE	東ヨーロッパ標準時間	02:00
Europe/Tallinn	EEST	EE	東ヨーロッパ夏時間	03:00
Europe/Tirane	CET	AL	中央ヨーロッパ標準時間	01:00
Europe/Tirane	CEST	AL	中央ヨーロッパ夏時間	02:00
Europe/Tiraspol	EET	MD	東ヨーロッパ標準時間	02:00
Europe/Tiraspol	EEST	MD	東ヨーロッパ夏時間	03:00
Europe/Ulyanovsk	+04	RU	モスクワ標準時間	04:00
Europe/Uzhgorod	EET	UA	東ヨーロッパ標準時間	02:00
Europe/Uzhgorod	EEST	UA	東ヨーロッパ夏時間	03:00
Europe/Vaduz	CET	LI	中央ヨーロッパ標準時間	01:00
Europe/Vaduz	CEST	LI	中央ヨーロッパ夏時間	02:00
Europe/Vatican	CET	VA	中央ヨーロッパ標準時間	01:00
Europe/Vatican	CEST	VA	中央ヨーロッパ夏時間	02:00
Europe/Vienna	CET	AT	中央ヨーロッパ標準時間	01:00
Europe/Vienna	CEST	AT	中央ヨーロッパ夏時間	02:00
Europe/Vilnius	EET	LT	東ヨーロッパ標準時間	02:00
Europe/Vilnius	EEST	LT	東ヨーロッパ夏時間	03:00
Europe/Volgograd	+04	RU	ヴォルゴグラード標準時	04:00
Europe/Warsaw	CET	PL	中央ヨーロッパ標準時間	01:00
Europe/Warsaw	CEST	PL	中央ヨーロッパ夏時間	02:00
Europe/Zagreb	CET	HR	中央ヨーロッパ標準時間	01:00
Europe/Zagreb	CEST	HR	中央ヨーロッパ夏時間	02:00
Europe/Zaporozhye	EET	UA	東ヨーロッパ標準時間	02:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Europe/Zaporozhye	EEST	UA	東ヨーロッパ夏時間	03:00
Europe/Zurich	CET	CH	中央ヨーロッパ標準時間	01:00
Europe/Zurich	CEST	CH	中央ヨーロッパ夏時間	02:00

エリア: Pacific(太平洋)

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Pacific/Apia	+13	WS	アピア標準時間	13:00
Pacific/Apia	+14	WS	アピア夏時間	14:00
Pacific/Auckland	NZST	NZ	ニュージーランド標準時間	12:00
Pacific/Auckland	NZDT	NZ	ニュージーランド夏時間	13:00
Pacific/Bougainville	+11	PG	パプアニューギニア時間	11:00
Pacific/Chatham	+1245	NZ	チャタム標準時間	12:45
Pacific/Chatham	+1345	NZ	チャタム夏時間	13:45
Pacific/Chuuk	+10	FM	チューク時間	10:00
Pacific/Easter	-06	CL	イースター島標準時間	-06:00
Pacific/Easter	-05	CL	イースター島夏時間	-05:00
Pacific/Efate	+11	VU	バヌアツ標準時間	11:00
Pacific/Enderbury	+13	KI	フェニックス諸島時間	13:00
Pacific/Fakaofu	+13	TK	トケラウ時間	13:00
Pacific/Fiji	+12	FJ	フィジー標準時間	12:00
Pacific/Fiji	+13	FJ	フィジー夏時間	13:00
Pacific/Funafuti	+12	TV	ツバル時間	12:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Pacific/Galapagos	-06	EC	ガラパゴス時間	-06:00
Pacific/Gambier	-09	PF	ガンビア時間	-09:00
Pacific/Guadalcanal	+11	SB	ソロモン諸島時間	11:00
Pacific/Guam	CHST	GU	チャモロ標準時間	10:00
Pacific/Honolulu	HST	US	ハワイ・アリューシャン標準時間	-10:00
Pacific/Johnston	HST	US	ハワイ・アリューシャン標準時間	-10:00
Pacific/Kiritimati	+14	KI	ライン諸島時間	14:00
Pacific/Kosrae	+11	FM	コスラエ時間	11:00
Pacific/Kwajalein	+12	MH	マーシャル諸島時間	12:00
Pacific/Majuro	+12	MH	マーシャル諸島時間	12:00
Pacific/Marquesas	-0930	PF	マルキーズ時間	-09:30
Pacific/Midway	SST	UM	サモア標準時間	-11:00
Pacific/Nauru	+12	NR	ナウル時間	12:00
Pacific/Niue	-11	NU	ニウエ時間	-11:00
Pacific/Norfolk	+11	NF	ノーフォーク島時間	11:00
Pacific/Noumea	+11	NC	ニューカレドニア標準時間	11:00
Pacific/Pago_Pago	SST	AS	サモア標準時間	-11:00
Pacific/Palau	+09	PW	パラオ時間	09:00
Pacific/Pitcairn	-08	PN	ピトケアン時間	-08:00
Pacific/Pohnpei	PONT	FM	ポンペイ(ポナベ)時間	11:00
Pacific/Ponape	+11	FM	ポンペイ(ポナベ)時間	11:00
Pacific/Port_Moresby	+10	PG	パプアニューギニア時間	10:00
Pacific/Rarotonga	-10	CK	クック諸島標準時間	-10:00
Pacific/Saipan	CHST	MP	チャモロ標準時間	10:00
Pacific/Samoa	SST	UM	サモア標準時間	-11:00

タイムゾーン情報	タイムゾーン名	リージョン	タイムゾーンの説明	タイムゾーンのオフセット
Pacific/Tahiti	-10	PF	タヒチ時間	-10:00
Pacific/Tarawa	+12	KI	ギルバート諸島時間	12:00
Pacific/Tongatapu	+13	TO	トンガ標準時間	13:00
Pacific/Tongatapu	+14	TO	トンガ夏時間	14:00
Pacific/Truk	+10	FM	チューク時間	10:00
Pacific/Wake	+12	UM	ウェーク島時間	12:00
Pacific/Wallis	+12	WF	ウォリス・フツナ時間	12:00
Pacific/Yap	+10	FM	チューク時間	10:00

TK_NLS_REPLACE_ICVEXT

ディクショナリ	1183
TK_NLS_REPLACE_ICVECT 環境変数	1183

ディクショナリ

TK_NLS_REPLACE_ICVECT 環境変数

起動時に新しいトランスコーディングテーブルにアクセスします。

該当要素: 構成ファイル

制限事項: 香港の補助文字セットを正しく表示するには、icvywinu 変換テーブルを指定する必要があります。次に例を示します。

```
export TK_NLS_REPLACE_ICVEXT=icvywinu;
```

注: 2025.03 以降、この環境変数は新しくなりました。

例: `export TK_NLS_REPLACE_ICVEXT=translation-table (sh/bash)`

```
setenv TK_NLS_REPLACE_ICVEXT translation-table (csh/tcsh)
```

構文

```
TK_NLS_REPLACE_ICVEXT=translation-table;
```

必須引数

translation-table

新しい変換テーブルを指定します。

詳細

TK_NLS_REPLACE_ICVECT は、デフォルトのトランスコーディングテーブルを置き換えることにより、起動時に新しいトランスコーディングテーブルを指定します。新しいトランスコーディングテーブルを無効または有効にできます。新しいトランスコーディングテーブルは、SAS セッション中はアクティブなままです。

Viya4 で TK_NLS_REPLACE_ICVECT を有効にするには、Environment Manager を使用して計算サーバーセッションを構成します。

ディレクトリから新しいトランスコーディングテーブルを削除し、SAS セッションを再起動すると、元のトランスコーディングテーブルを復元できます。

香港の補助文字セットのユーザーは、正確な文字を表示するために icvywinu 変換テーブルを指定する必要があります。

表 A14.1 変換テーブル - icvywinu

変換テーブル		説明
icvywinu	BIG5	BIG5-HKSCS マッピングテーブル
	MS-950(YWIN)	
	MACOS-2(YMAC)	