



THE  
POWER  
TO KNOW.

# SAS<sup>®</sup> 9.4 データセットオプション リファレンス 第3版

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2014. *SAS® 9.4 Data Set Options:Reference, Third Edition*. Cary, NC: SAS Institute Inc.

**SAS® 9.4 Data Set Options:Reference, Third Edition**

Copyright © 2014, 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, North Carolina 27513-2414.

August 2014

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.

---

## 目次

本書について	v
SAS 9.4 データセットオプションの新機能	ix
<b>1 章・データセットオプションとは</b>	<b>1</b>
データセットオプションについて	1
構文	1
データセットオプションの使用	2
他の SAS ドキュメントで説明されているデータセットオプション	3
<b>2 章・データセットオプションリファレンス</b>	<b>5</b>
カテゴリ別データセットオプション	6
ディクショナリ	8
<b>推奨資料</b>	<b>77</b>
キーワード	79



# 本書について

---

## SAS 言語の構文規則

### SAS 言語の構文規則の概要

SAS 言語要素の構文は、標準的な表記規則を使用して文書化されます。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 3 項目に分けることができます。

- 構文のコンポーネント
- スタイル規則
- 特殊文字
- SAS ライブラリや外部ファイルへの参照

### 構文のコンポーネント

言語要素の多くでは、その構文の構成要素はキーワードと引数から構成されます。キーワードのみが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。

#### キーワード

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。

次の SAS 構文の例では、キーワードは構文の最初の単語です。

```
CHAR (string, position)
CALL RANBIN (seed, n, p, x);
ALTER (alter-password)
BEST w.
REMOVE <data-set-name>
```

次の例では、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**

#### 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が使用されます。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 (<>) で囲まれます。

次の例では、*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。次の SAS コードの例では、引数 *string* の値は 'summer'、引数 *position* の値は 4 です。x=char('summer', 4);

次の例では、*string* と *substring* が必須の引数で、*modifiers* と *startpos* は省略可能な引数です。

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

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

## スタイル規則

SAS 構文のドキュメントで使用されているスタイル規則には、太字の大文字、小文字、斜体があります。

#### 太字の大文字

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。次の例では、キーワード ERROR が太字の大文字で記述されています。

**ERROR**<message>;

#### 大文字

リテラルである引数を表します。

次の CMPMODEL=システムオプションの例には、BOTH、CATALOG、XML というリテラルが含まれています。

**CMPMODEL = BOTH | CATALOG | XML**

#### 斜体

ユーザー指定の引数または値を示します。斜体の項目は、ユーザーが指定する次のいずれかの値を表します。

- 非リテラル引数

次の LINK ステートメントの例では、引数 *label* はユーザーが指定する値であるため、斜体で記述されています。

**LINK** *label*;

- 引数に割り当てる非リテラル値

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

**FORMAT** = *variable-1* < ..., *variable-n format* ><DEFAULT = *default-format*>;

斜体の項目は、選択可能な引数リストの一般名 (*attribute-list* など) である場合もあります。斜体の項目を複数使用する場合、項目は *item-1*, ..., *item-n* のように表されます。

## 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。

次の MAPS システムオプションの例では、等号記号で 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。山かっこで囲まれていない引数は必須です。

次の CAT 関数の例では、少なくとも 1 つの項目が必須です。

**CAT** (*item-1* < ..., *item-n*>)

縦棒は一連の値の中から 1 つ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。

次の 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、省略記号の後に続く引数または一連の引数を繰り返せることを示します。省略記号とそれに続く引数が山かっこで囲まれている場合、その引数は省略可能です。

次の CAT 関数の例では、省略記号は複数の省略可能な項目を指定できることを示します。

**CAT** (*item-1* < ..., *item-n*>)

'value'または" value"

一重または二重引用符で囲まれた引数には、同じく一重または二重引用符で囲まれた値を指定する必要があることを示します。

次の FOOTNOTE ステートメントの例では、引数 *text* が引用符で囲まれています。

**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。

次の例では、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

多くの SAS ステートメントやその他の言語要素は、SAS ライブラリや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は、通常、SAS ステートメント(LIBNAME または FILENAME)を使用するか、動作環境の制御言語を使用して関連付けをするかを選択できます。SAS ライブラリや外部ファイルを参照するにはいくつかの方法がありますが、どのような方法が使えるかはお使いの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。SAS ドキュメント中の SAS ライブラリを使用する例では、*SAS-library* という語句を斜体で使用しています。*SAS-library* は次のように引用符で囲まれます。

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



# SAS 9.4 データセットオプションの新機能

---

## 概要

SAS 9.4 データセットオプションには、1 つの新しいデータセットオプションと、2 つの拡張されたデータセットがあります。

- 新しい `ENCRYPTKEY=` (23 ページ) データセットオプションは、キー値を指定します。これにより、AES (Advanced Encryption Standard) 暗号化を使用できます。
- 拡張された `ENCRYPT=` (18 ページ) データセットオプションは AES 暗号化をサポートするようになりました。これにより、AES アルゴリズムを使用してより強固な暗号化を行えます。
- `EXTENDOBSCOUNTER=` (26 ページ) データセットオプションのデフォルトが YES になり、拡張ファイル形式で SAS データファイルが作成されるようになりました。拡張ファイル形式は、32 ビット長の制限を超えてオブザベーションカウントを拡張します。

x SAS データセットオプション

## 1 章

## データセットオプションとは

---

データセットオプションについて .....	1
構文 .....	1
データセットオプションの使用 .....	2
入力または出力 SAS データセットにデータセットオプションを使用する .....	2
データセットオプションとシステムオプションの相互作用 .....	2
他の SAS ドキュメントで説明されているデータセットオプション .....	3

---

データセットオプションについて

データセットオプションは、データセットオプションが表示される SAS データセットにのみ適用されるアクションを指定します。データセットオプションは、次の動作を実行します。

- 変数名を変更する
- 最初または最後の  $n$  個のオブザベーションのみを処理対象として選択する
- 処理または出力データセットから変数を削除する
- データセットのパスワードを指定する

---

構文

SAS データセット名の後に、データセットオプションをカッコで囲んで指定します。複数のデータセットオプションを指定するには、スペースで区切ります。

*(option-1=value-1 < ...option-n=value-n>)*

次の例では、SAS ステートメント内のデータセットオプションを示します。

- `data scores(keep=team game1 game2 game3);`
- `data mydata(index=(b k) label='label for my data set' drop=p read=secret);`
- `data new(drop=i n index=(j combo=(x1 a1 a20 b1 b50 )));`
- `data idxdup2(compress=yes index=(ok1 ok2 ssn/unique ok3));`

- `proc print data=new(drop=year);`
- `set old(rename=(date=Start_Date));`

---

## データセットオプションの使用

### 入力または出力 SAS データセットにデータセットオプションを使用する

ほとんどの SAS データセットオプションは、DATA ステップまたはプロシジャ(PROC) ステップ内の入力または出力 SAS データセットに適用されます。データセットオプションが入力データセットに関連付けられている場合、アクションは読み込まれているデータセットに適用されます。データセットオプションが DATA ステートメント内に表示されるか、PROC ステップ内で出力データセットの指定の後に表示されると、アクションは出力データセットに適用されます。DATA ステップでは、出力データセットのデータセットオプションは、存在するであろう OUTPUT ステートメント内ではなく、DATA ステートメント内に表示される必要があります。

COMPRESS=などの一部のデータセットオプションは、そのデータセットの存続期間に存在する属性を設定するため、SAS データセットを作成するときのみ意味があります。大部分のデータセットオプションでは、変更またはキャンセルするには、データセットを再作成する必要があります。他のオプション(PW=や LABEL=など)は PROC DATASETS を使用して変更できます。詳細については、“DATASETS” (*Base SAS Procedures Guide*)を参照してください。

データセットオプションが同じ DATA または PROC ステップ内の入力データセットと出力データセットに表示される場合、データセットオプションは先に入力データセットに適用されます。次に、プログラミングステートメントが評価されるか、データセットオプションが出力データセットに適用されます。同様に、作成しているデータセットに指定されるデータセットオプションは、プログラミングステートメントが処理された後に適用されます。たとえば、RENAME=データセットオプションを使用しているときには、新しい名前は DATA ステップが終了するまで変数に関連付けられません。

ときには、同じステートメント内で使用されているデータセットオプションが競合する場合があります。たとえば、同じステートメント内の同じ変数に DROP=データセットオプションと KEEP=データセットオプションは指定できません。また、タイミングが問題になる場合もあります。たとえば、SET ステートメントで指定されたデータセットに対し KEEP=と RENAME=を使用する場合、KEEP=では元の変数名を使用する必要があります。KEEP=はデータセットが読み込まれる前に処理されます。RENAME=で指定した新しい名前は、SET ステートメントの後のプログラミングステートメントに適用されません。

### データセットオプションとシステムオプションの相互作用

システムオプションおよびデータセットオプションの多くは、同じ名前を持ち、同じ機能を備えています。システムオプションは、SAS ジョブまたはセッション内のすべての DATA ステップおよび PROC ステップに対して影響します。

データセットオプションは、表示されるステップ内でデータセットのシステムオプションよりも優先されます。この例では、OPTIONS ステートメント内の OBS=システムオプションが、SAS ジョブ内のデータセットから最初の 100 個のオブザベーションのみが処理されることを指定しています。ただし、SET ステートメント内の OBS=データオプションは、データセット TWO のシステムオプションよりも優先されます。OBS=は、データセット TWO から最初の 5 個のオブザベーションのみを読み込むように指定します。PROC PRINT ステップによりデータセット FINAL が出力されます。このデータセットには、デ

データセット TWO の最初の 5 個のオブザベーション、次にデータセット THREE の最初の 100 個のオブザベーションが含まれます。

```
options obs=100;

data final;
  set two(obs=5) three;
run;

proc print data=final;
run;
```

---

## 他の SAS ドキュメントで説明されているデータセットオプション

*SAS データセットオプション*: リファレンスに記載されているデータセットオプションに加えて、次のドキュメントにもデータセットオプションが記載されています。

- *Windows 版 SAS*
- *UNIX 版 SAS*
- *z/OS 版 SAS*
- *SAS 各国語サポート(NLS): リファレンスガイド*
- *SAS Scalable Performance Data Engine: リファレンス*
- *SAS/ACCESS for Relational Databases: Reference*
- *Base SAS プロシジャガイド*



## 2 章

## データセットオプションリファレンス

---

カテゴリ別データセットオプション	6
ディクショナリ	8
ALTER=データセットオプション	8
BUFNO=データセットオプション	9
BUFSIZE=データセットオプション	11
CNTLLEV=データセットオプション	12
COMPRESS=データセットオプション	13
DLDMGACTION=データセットオプション	16
DROP=データセットオプション	17
ENCRYPT=データセットオプション	18
ENCRYPTKEY=データセットオプション	23
EXTENDOBSCOUNTER=データセットオプション	26
FILECLOSE=データセットオプション	27
FIRSTOBS=データセットオプション	27
GENMAX=データセットオプション	29
GENNUM=データセットオプション	30
IDXNAME=データセットオプション	32
IDXWHERE=データセットオプション	33
INDEX=データセットオプション	35
IN=データセットオプション	36
KEEP=データセットオプション	37
LABEL=データセットオプション	38
OBSBUF=データセットオプション	40
OBS=データセットオプション	42
OUTREP=データセットオプション	50
POINTOBS=データセットオプション	52
PW=データセットオプション	54
PWREQ=データセットオプション	55
READ=データセットオプション	55
RENAME=データセットオプション	56
REPEMPTY=データセットオプション	60
REPLACE=データセットオプション	61
REUSE=データセットオプション	62
ROLE=データセットオプション	63
SORTEDBY=データセットオプション	64
SPILL=データセットオプション	66
TOBSNO=データセットオプション	70
TYPE=データセットオプション	71
WHERE=データセットオプション	72
WHEREUP=データセットオプション	74
WRITE=データセットオプション	75

## カテゴリ別データセットオプション

SAS データセットオプションのカテゴリは、SAS データセットオプショングループに対応します。

データセットコントロール	データセットに関連付けられたオプション
オブザベーションコントロール	オブザベーションに関連付けられたオプション
SAS インデックス使用のユーザーコントロール	インデックスに関連付けられたオプション
変数コントロール	変数に関連付けられたオプション
その他	テープの位置に関連付けられたオプション

カテゴリ	言語要素	説明
SAS インデックス使用のユーザーコントロール	IDXNAME=データセットオプション (p. 32)	WHERE 式の条件の照合に特定のインデックスを使用するように SAS に指示します。
	IDXWHERE=データセットオプション (p. 33)	SAS で WHERE 式の条件の照合にインデックス検索を使用するか、順次検索を使用するかを指定します。
オブザベーションコントロール	FIRSTOBS=データセットオプション (p. 27)	SAS データセット内で SAS が最初に処理するオブザベーションを指定します。
	IN=データセットオプション (p. 36)	データセットのデータが現在のオブザベーションに関与しているかどうかを示すブール値変数を作成します。
	OBS=データセットオプション (p. 42)	データセット内で SAS が最後に処理するオブザベーションを指定します。
	POINTOBS=データセットオプション (p. 52)	SAS でオブザベーションへのランダムアクセスまたは順次アクセスが可能な圧縮データセットを作成するかどうかを指定します。
	WHERE=データセットオプション (p. 72)	SAS データセットからオブザベーションを選択するために使用する特定の条件を指定します。
	WHEREUP=データセットオプション (p. 74)	新しいオブザベーションと変更されたオブザベーションを WHERE 式に対して評価するかどうかを指定します。
その他	FILECLOSE=データセットオプション (p. 27)	SAS データセットを閉じたときのテープの位置付けを指定します。
データセットコントロール	ALTER=データセットオプション (p. 8)	ALTER=パスワードを SAS ファイルに割り当てて、ユーザーによるファイルの置き換えや削除を防止し、読み取り保護や書き込み保護されたファイルへのアクセスを可能にします。
	BUFNO=データセットオプション (p. 9)	SAS データセットの処理に割り当てるバッファ数を指定します。
	BUFSIZE=データセットオプション (p. 11)	出力 SAS データセットの永久バッファページのサイズを指定します。



カテゴリ	言語要素	説明
	CNTLLEV=データセットオプション (p. 12)	SAS データセットへの共有アクセスのレベルを指定します。
	COMPRESS=データセットオプション (p. 13)	新しい出力 SAS データセットでのオブザベーションの圧縮方法を指定します。
	DLDMGACTION=データセットオプション (p. 16)	SAS ライブラリ内の SAS データセットの破損が検出されたときに実行するアクションを指定します。
	ENCRYPT=データセットオプション (p. 18)	出力 SAS データセットを暗号化するかどうかを指定します。
	ENCRYPTKEY=データセットオプション (p. 23)	AES (Advanced Encryption Standard)暗号化のキー値を指定します。
	EXTENDOBSCOUNTER=データセットオプション (p. 26)	新しい出力 SAS データファイル内の最大オブザベーションカウントを拡張するかどうかを指定します。
	GENMAX=データセットオプション (p. 29)	新しいデータセットの世代を要求し、既存のデータセットの世代数を変更し、バージョンの最大数を指定します。
	GENNUM=データセットオプション (p. 30)	SAS データセットの特定の世代を指定します。
	INDEX=データセットオプション (p. 35)	新しい出力 SAS データセットのインデックスを定義します。
	LABEL=データセットオプション (p. 38)	SAS データセットのラベルを指定します。
	OBSBUF=データセットオプション (p. 40)	DATA ステップビューを処理する表示バッファのサイズを決定します。
	OUTREP=データセットオプション (p. 50)	出力 SAS データセットのデータ表現を指定します。
	PW=データセットオプション (p. 54)	SAS ファイルに READ、WRITE および ALTER パスワードを割り当て、パスワード保護された SAS ファイルへのアクセスを可能にします。
	PWREQ=データセットオプション (p. 55)	パスワードダイアログボックスを表示するかどうか指定します。
	READ=データセットオプション (p. 55)	READ=パスワードを SAS ファイルに割り当てます。これにより、ユーザーはパスワードを入力しない限りファイルを読み取れなくなります。
	REPEMPTY=データセットオプション (p. 60)	同じ名前の新しい空のデータセットで、既存の SAS データセットを上書きできるかどうかを指定します。
	REPLACE=データセットオプション (p. 61)	データを含む新しい SAS データセットで、同じ名前の既存のデータセットを上書きできるかどうかを指定します。

カテゴリ	言語要素	説明
	REUSE=データセットオプション (p. 62)	圧縮 SAS データセットの空き領域に新しいオブザベーションを書き込みできるかどうかを指定します。
	ROLE=データセットオプション (p. 63)	スタースキーマ結合のファクトテーブルを指定します。
	SORTEDBY=データセットオプション (p. 64)	データセットの現在の並べ替え方法を示します。
	SPILL=データセットオプション (p. 66)	DATA ステップビューの非順次処理の予備ファイルを作成するかどうかを指定します。
	TOBSNO=データセットオプション (p. 70)	クライアント/サーバー間の転送で送信するオブザベーション数を指定します。
	TYPE=データセットオプション (p. 71)	特殊な構造の SAS データセットにデータセットの種類を指定します。
	WRITE=データセットオプション (p. 75)	WRITE パスワードを SAS ファイルに割り当てます。これにより、ユーザーはパスワードを入力しない限りファイルを書き込めなくなります。
変数コントロール	DROP=データセットオプション (p. 17)	入力データセットの場合は、指定された変数を処理対象から除外し、出力データセットの場合は、指定された変数をデータセットへの書き込み対象から除外します。
	KEEP=データセットオプション (p. 37)	入力データセットの場合は、処理する変数を指定し、出力データセットの場合は、データセットに書き込む変数を指定します。
	RENAME=データセットオプション (p. 56)	変数の名前を変更します。

## ディクショナリ

### ALTER=データセットオプション

ALTER=パスワードを SAS ファイルに割り当てて、ユーザーによるファイルの置き換えや削除を防止し、読み取り保護や書き込み保護されたファイルへのアクセスを可能にします。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** データセットコントロール

**注:** パスワードのセキュリティを確認するためにこの操作の後にログを確認してください。詳細については、“Blotting Passwords and Encryption Key Values” (*SAS Language Reference: Concepts*)を参照してください。

**参照項目:** 動作環境向けドキュメントの OpenVMS、UNIX、または z/OS の ALTER=データセットオプション。

## 構文

ALTER=*alter-password*

### 構文の説明

*alter-password*

有効な SAS 名である必要があります。詳細については、“Words in the SAS Language” (*SAS Language Reference: Concepts*)を確認してください。

### 詳細

ALTER=オプションは、カタログ以外のすべての種類の SAS ファイルに適用されません。このオプションは、パスワードを SAS ファイルに割り当てる場合や、読み取り保護、書き込み保護、または変更保護された SAS ファイルにアクセスする場合に使用できます。

ALTER パスワードで保護された SAS データセットを置き換えると、新しいデータセットに ALTER パスワードが継承されます。新しいデータセットの ALTER パスワードを変更するには、DATASETS プロシジャで MODIFY ステートメントを使用します。

注: SAS パスワードでは、SAS System 以外での SAS ファイルへのアクセスはコントロールされません。SAS 以外からの SAS ファイルへのアクセスをコントロールするには、オペレーティングシステムで提供されるユーティリティか、ファイルシステムのセキュリティコントロールを使用してください。

### 関連項目:

- “File Protection” (*SAS Language Reference: Concepts*)
- “Manipulating Passwords” (*Base SAS Procedures Guide*)

### データセットオプション:

- “ENCRYPT=データセットオプション” (18 ページ)
- “PW=データセットオプション” (54 ページ)
- “READ=データセットオプション” (55 ページ)
- “WRITE=データセットオプション” (75 ページ)

---

## BUFNO=データセットオプション

SAS データセットの処理に割り当てるバッファ数を指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** データセットコントロール

**参照項目:** 動作環境向けドキュメントの BUFNO=データセットオプション。

## 構文

BUFNO= *n* | *nK* | *hexX* | MIN | MAX

**構文の説明***n* | *n*K

バッファの数を 1 (バイト)や 1,024 (キロバイト)の倍数で指定します。たとえば、値 8 では 8 個のバッファ、値 1k では 1,024 個のバッファが指定されます。

*hex*X

バッファ数を 16 進値で指定します。先頭が数値(0 から 9)、末尾が X の値を指定する必要があります。たとえば、値 2dx ではバッファ数が 45 バッファに設定されます。

MIN

最小バッファ数を 0 に設定します。これにより SAS では動作環境に最適な最小値が使用されます。これがデフォルト設定です。

MAX

バッファ数を動作環境で可能な最大数に設定します。4 バイト符号付き整数の最大値である  $2^{31}-1$  (約 20 億)以下の値になります。

**詳細**

バッファ数は、データセットの永続的的属性ではなく、現在の SAS セッションまたはジョブでのみ有効です。

BUFNO=は、入力、出力または更新用に開かれている SAS データセットに適用されません。

バッファ数を大きくするほど、特定の SAS データセットに必要な入力および出力(I/O)操作の数を制限して、実行時間を短縮できます。ただし、実行時間が改善するかわりにメモリ消費が増えます。

小さいデータセットに対する I/O 操作を減らすとともに実行時間をスピードアップするには、処理するデータのページごとに 1 個のバッファを割り当てます。この手法は、処理中に同じオブザベーションを複数回読み込むときに最も効果的です。

SAS でデータセットページとインデックスファイルページの数に基づいてバッファ数が割り当てられるように要求するには、SASFILE グローバルステートメントを使用します。

**動作環境の情報**

BUFNO=のデフォルト値は、動作環境に応じて決まり、順次アクセスを最適化するように設定されます。直接(ランダム)アクセスのパフォーマンスを向上させるには、BUFNO=の値を変更します。直接アクセスのデフォルト設定と使用可能な設定については、動作環境向け SAS ドキュメントの BUFNO=データセットオプションを参照してください。

**比較**

- BUFNO=データセットオプションが指定されていない場合、BUFNO=システムオプションの値が使用されます。同じ SAS セッションで両方の値が指定されている場合、BUFNO=データセットオプションに指定された値は、BUFNO=システムオプションに指定された値よりも優先されます。

**関連項目:****データセットオプション:**

- [“BUFSIZE=データセットオプション” \(11 ページ\)](#)

**システムオプション:**

- “BUFNO= System Option” (*SAS System Options: Reference*)

**ステートメント:**

- “SASFILE Statement” (*SAS Statements: Reference*)

---

## BUFSIZE=データセットオプション

出力 SAS データセットの永久バッファページのサイズを指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** データセットコントロール

**制限事項:** 出力データセットにのみ使用します。

**参照項目:** 動作環境向けドキュメントの UNIX、z/OS、または OpenVMS の BUFSIZE=データセットオプション。

### 構文

**BUFSIZE**=*n* | *nK* | *nM* | *nG* | *hexX* | **MAX**

#### 構文の説明

***n* | *nK* | *nM* | *nG***

ページサイズを 1 (バイト)、1,024 (キロバイト)、1,048,576 (メガバイト)、1,073,741,824 (ギガバイト)のいずれかの倍数で指定します。たとえば、値 8 では 8 バイトのページサイズ、値 4k では 4,096 バイトのページサイズが指定されます。

注: システムオプションとデータセットオプションのどちらも指定されていない場合、デフォルトは 0 です。その結果、動作環境に最適な最小ページサイズが使用されます。次のいずれかの場合は BUFSIZE=システムオプションが使用されません。

- BUFSIZE=データセットオプションが設定されていない
- BUFSIZE=データセットオプションがゼロに設定されている

バッファページサイズを動作環境のデフォルト値にリセットするには、BUFSIZE=0 を使用します。

***hexX***

バッファサイズを 16 進値で指定します。先頭が数値(0 から 9)、末尾が X の値を指定する必要があります。たとえば、値 2dx ではページサイズが 45 バイトに設定されます。

**MAX**

ページサイズを動作環境で可能な最大値に設定します。4 バイト符号付き整数の最大値である  $2^{31}-1$  (約 20 億バイト)以下の値になります。

### 詳細

ページサイズは、1 回の I/O 操作で 1 個のバッファに転送できるデータ量です。ページサイズは、データセットの永続的属性であり、データセットが処理されるときに使用されます。

ページサイズが大きいほど、ストレージメディアに対する必要な読み取りまたは書き込み回数を減らして、実行時間を短縮します。ただし、実行時間が改善するかわりにメモリ消費が増えます。

ページサイズを変更するには、DATA ステップを使用してデータセットをコピーし、新しいページを指定するか、SAS デフォルトを使用します。ページサイズを動作環境のデフォルト値にリセットするには、BUFSIZE=0 を使用します。

注: COPY プロシジャを使用してデータセットを別のエンジンで割り当てられた別のライブラリにコピーできます。指定されたデータセットのページサイズは保持されません。

#### 動作環境の情報

BUFSIZE=のデフォルト値は、動作環境に応じて決まり、順次アクセスを最適化するように設定されます。直接(ランダム)アクセスのパフォーマンスを向上させるには、BUFSIZE=の値を変更します。直接アクセスのデフォルト設定と使用可能な設定については、動作環境向け SAS ドキュメントの BUFSIZE=データセットオプションを参照してください。

### 関連項目:

#### データセットオプション:

- “BUFNO=データセットオプション” (9 ページ)

#### システムオプション:

- “BUFSIZE= System Option” (*SAS System Options: Reference*)

---

## CNTLLEV=データセットオプション

SAS データセットへの共有アクセスのレベルを指定します。

<b>該当要素:</b>	DATA ステップおよび PROC ステップ
<b>カテゴリ:</b>	データセットコントロール
<b>制限事項:</b>	入力データセットにのみ指定します。

---

### 構文

CNTLLEV=LIB | MEM | REC

#### 構文の説明

##### LIB

同時アクセスがライブラリレベルでコントロールされることを指定します。ライブラリレベルのコントロールにより、同時アクセスでは、ライブラリへの更新プロセスが 1 つのみに制限されます。

##### MEM

同時アクセスが SAS データセット(メンバ)レベルでコントロールされることを指定します。メンバレベルのコントロールにより、同時アクセスでは、SAS データセットへの更新または出力プロセスが 1 つのみに制限されます。データセットが更新または出力プロセス用に開かれている場合、他の操作からそのデータセットにはアクセスできません。データセットが入力プロセス用に開かれている場合、他の同時入力プロセスは許可されますが、更新または出力プロセスは許可されません。

**REC**

同時アクセスがオブザベーション(レコード)レベルでコントロールされることを指定します。レコードレベルのコントロールにより、同じ SAS データセットに対する複数の更新アクセスは許可されますが、同じオブザベーションの同時更新は拒否されます。

**詳細**

CNTLLEV=オプションでは、SAS データセットへの共有更新アクセスが拒否されるレベルを指定します。SAS データセットは、複数の SAS セッションによって、または 1 つのセッション内の複数のステートメント、ウィンドウまたはプロシジャによって開くことができます。SAS プロシジャのデフォルトでは、可能な最大同時アクセスレベルが許可されるとともに、データとデータ分析の一貫性が保証されます。そのため、次の状況では CNTLLEV=データセットオプションを使用します。

- SAS Component Language (SCL)、SAS/IML ソフトウェア、DATA ステッププログラミングなどのように、アプリケーションでデータへのアクセスをコントロールする場合
- データのメンバレベルコントロールを提供しないインターフェイスエンジンでデータにアクセスする場合

CNTLLEV=REC を使用し、SAS プロシジャでデータ分析の一貫性のためにメンバレベルのコントロールが必要な場合、SAS ログに警告が出力されます。この警告は、分析中に他のプロセスによりデータが更新されると、不正確または予測できない結果が発生する可能性があることを示します。

**例: 共有アクセスレベルの変更**

次の例では、最初の SET ステートメントに、共有アクセスレベルでデフォルトのメンバレベルコントロールよりレコードレベルコントロールを優先させるための CNTLLEV=データセットオプションが含まれます。2 つ目の SET ステートメントでは SAS データセットがデフォルトのメンバレベルコントロールで開かれます。

```
set datalib.fuel (cntllev=rec) point=obsnum;
.
.
.
set datalib.fuel;
by area;
```

---

**COMPRESS=データセットオプション**

新しい出力 SAS データセットでのオブザベーションの圧縮方法を指定します。

- 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
- 制限事項:** 出力データセットにのみ使用します。

**構文**

COMPRESS=NO | YES | CHAR | BINARY



## 構文の説明

### NO

新しく作成された SAS データセット内でオブザベーションは圧縮されないこと(固定長レコードの保持)を指定します。

### YES | CHAR

新しく作成された SAS データセット内でオブザベーションは SAS により RLE (Run Length Encoding)を使用して圧縮されること(可変長レコードの作成)を指定します。RLE では、繰り返し連続する同じ文字(空白を含む)を 2 バイトまたは 3 バイトの表現に削減することでオブザベーションが圧縮されます。

別名 ON

### BINARY

新しく作成された SAS データセット内でオブザベーションは RDC (Ross Data Compression)を使用して圧縮されること(可変長レコードの作成)を指定します。RDC では、Run Length Encoding とスライディングウィンドウ圧縮を組み合わせることで反復バイトパターンをより効果的に表現することでファイルが圧縮されます。

注: この方式は、中サイズから大サイズ(数百バイトまたはそれ以上)のバイナリデータ(文字変数と数値変数)のブロックを圧縮する場合に有効です。この圧縮関数は一度に 1 つのレコードに対してのみ動作するため、効果的に圧縮するには数百バイト以上のレコード長が必要です。

## 詳細

ファイルの圧縮は、各オブザベーションの表現に必要なバイト数を削減します。ファイル圧縮の利点として、ファイルのストレージ要件の削減、処理中のデータ読み取り/書き込みのための I/O 操作の削減などがあります。ただし、圧縮ファイルの読み取りには(各オブザベーションの圧縮を解除するオーバーヘッドのために)より多くの CPU リソースが必要になります。状況によっては、圧縮後のファイルサイズが減らずに増えることがあります。

COMPRESS=データセットオプションは、個別のファイルを圧縮するために使用します。オプションは出力データセットに対してのみ指定します。つまり DATA ステップの DATA ステートメント内または SAS プロシジャの OUT=オプション内に指定されたデータセットのみ指定します。COMPRESS=データセットオプションは、SAS データファイル(メンバタイプ DATA)を作成する場合にのみ使用します。SAS ビューは、データが含まれていないため圧縮できません。COPY プロシジャではデータセットオプションはサポートされません。したがって、PROC COPY かまたは PROC DATASETS の COPY ステートメントで COMPRESS=データセットオプションを使用することはできません。

**ヒント** PROC COPY で生成された OUTPUT データセットを圧縮するには、PROC COPY ステートメントの NOCLONE オプションの前に COMPRESS=YES システムオプションを使用します。

```
options compress=yes;
proc copy in=work out=new noclone;
select x;
run;
```

ファイルが圧縮された後、設定はファイルの永続的属性になります。設定を変更するには、ファイルを再作成する必要があります。そのため、ファイルを圧縮解除するには、圧縮ファイルをコピーする DATA ステップに COMPRESS=NO を指定します。

通常、COMPRESS=CHAR はシングルバイトが繰り返す場合に適した圧縮を提供します。COMPRESS=BINARY はバイト文字列が繰り返す場合に適した圧縮を適用しま



す。繰り返すシングルバイトの検索よりも、繰り返すバイト文字列の検索のほうがコストがかかります。たとえば、“例 1: Compress=CHAR” (15 ページ) および “例 2: COMPRESS=BINARY” (15 ページ) を参照してください。

## 比較

COMPRESS=データセットオプションは、LIBNAME ステートメントの COMPRESS=オプションと COMPRESS=システムオプションよりも優先されます。

データセットオプション POINTOBS=YES(デフォルト)により、圧縮データセットを順次アクセスではなく、ランダムアクセス(オブザベーション番号を指定)で処理できるように定義されます。ランダムアクセスでは、オブザベーション番号を FSEDIT プロシジャや、SET および MODIFY ステートメントの POINT=オプションに指定できます。

圧縮ファイルを作成するとき、空き領域の追跡と再利用のために(データセットオプションまたはシステムオプションとして)REUSE=YES を指定することもできます。REUSE=YES を指定すると、新しいオブザベーションは、他のオブザベーションの更新または削除によって空いた領域に挿入されます。デフォルトの REUSE=NO が有効な場合、新しいオブザベーションは既存のファイルに追加されます。

POINTOBS=YES と REUSE=YES は相互排他的です。つまり一緒に使用することはできません。REUSE=YES は、POINTOBS=YES よりも優先されます。REUSE=YES を設定すると、POINTOBS=NO が自動的に設定されます。

TAPE エンジンでは COMPRESS=データセットオプションがサポートされますが、COMPRESS=システムオプションはサポートされません。

XPORT エンジンでは圧縮はサポートされません。

## 例

### 例 1: Compress=CHAR

```
data mylib.CharRepeats (compress=char);
  length ca $ 200;
  do i=1 to 100000;
    ca='aaaaaaaaaaaaaaaaaaaaa';
    cb='bbbbbbbbbbbbbbbbbbbbbbb';
    cc='ccccccccccccccccccccccc';
    output;
  end;
run;
```

次のメッセージが SAS ログに書き込まれます。

```
NOTE: Compressing data set MYLIB.CHARREPEATS decreased size by 88.55 percent.
       Compressed is 45 pages; un-compressed would require 393 pages.
```

### 例 2: COMPRESS=BINARY

```
data mylib.StringRepeats (compress=binary);
  length cabcd $ 200;
  do i=1 to 1000000;
    cabcd='abcdabcdabcdabcdabcdabcdabcdabcd';
    cefgh='efghefghefghefghefghefghefghefghe';
    cijkl='ijklijklijklijklijklijklijklijkl';
    output;
  end;
run;
```

次のメッセージが SAS ログに書き込まれます。

NOTE: Compressing data set MYLIB.STRINGREPEATS decreased size by 70.27 percent.  
Compressed is 1239 pages; un-compressed would require 4167 pages.

### 関連項目:

- “Compressing Data Files” (*SAS Language Reference: Concepts*)

### データセットオプション:

- “POINTOBS=データセットオプション” (52 ページ)
- “REUSE=データセットオプション” (62 ページ)

### ステートメント:

- “LIBNAME Statement” (*SAS Statements: Reference*)

### システムオプション:

- “COMPRESS= System Option” (*SAS System Options: Reference*)
- “REUSE= System Option” (*SAS System Options: Reference*)

---

## DLDMGACTION=データセットオプション

SAS ライブラリ内の SAS データセットの破損が検出されたときに実行するアクションを指定します。

**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

**デフォルト:** Windows および UNIX では、出荷時のデフォルト値は、対話型モードでは REPAIR、バッチモードでは FAIL です。  
z/OS では、出荷時のデフォルト値は、対話型モードでは PROMPT、バッチモードでは REPAIR です。

### 構文

**DLDMGACTION=**FAIL | ABORT | REPAIR | NOINDEX | PROMPT

#### 構文の説明

##### FAIL

ただちにステップを停止し、エラーメッセージをログに発行します。これはバッチモードのデフォルトです。

##### ABORT

ステップを停止し、エラーメッセージをログに発行し、SAS セッションを終了します。

##### REPAIR

破損したファイルを次回開こうとすると、破損したデータセットの自動修復が試行されます。データセットは、破損時点で切り捨てられた可能性があります。REPAIR オプションでは、1 つまたは複数のインデックスが再作成されます。破損が非常に重大な場合、自動修復の試行が成功しないこともあります。

**NOINDEX**

インデックスと一貫性制約がないデータファイルを自動的に修復します。また、この修復では、インデックスファイルを削除し、データファイルを更新して無効にしたインデックスと一貫性制約を反映します。データファイルは INPUT モードでのみ開くように制限されます。無効にしたインデックスと一貫性制約を修正または削除するには PROC DATASETS の REBUILD ステートメントを実行するように指示する警告が SAS ログに書き込まれます。

参照項目 “DLDMGACTION= System Option” (*SAS System Options: Reference*)

“Recovering Disabled Indexes and Integrity Constraints” (*SAS Language Reference: Concepts*)

**PROMPT**

FAIL、ABORT、REPAIR、NOINDEX のいずれかのアクションを選択するように求めるダイアログボックスを表示します。

---

## DROP=データセットオプション

入力データセットの場合は、指定された変数を処理対象から除外し、出力データセットの場合は、指定された変数をデータセットへの書き込み対象から除外します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** 変数コントロール

---

**構文**

**DROP=***variable(s)*

**構文の説明**

*variable(s)*

1 つ以上の変数名のリストを指定します。SAS で許可される形式の変数のリストを指定できます。

**詳細**

オプションが入力データセットに関連付けられている場合、この変数は処理には使用できません。DROP=データセットオプションが出力データセットに関連付けられている場合、この変数は出力データセットには書き込まれませんが、処理には使用できません。

**比較**

DROP=データセットオプションと DROP ステートメントには次の違いがあります。

- DATA ステップでは、DROP=データセットオプションは入力および出力データセットに適用できます。DROP ステートメントは、出力データセットにのみ適用されます。
- DATA ステップで複数の出力データセットを作成する場合、異なる変数を異なるデータセットに書き込むには DROP=データセットオプションを使用します。DROP ステートメントはすべての出力データセットに適用されます。
- PROC ステップでは、DROP=データセットオプションのみを使用でき、DROP ステートメントは使用できません。

## 例

### 例 1: 入力から変数を除外する

この例では、変数 SALARY および GENDER は処理対象には含まれず、出力データセットにも書き込まれません。

```
data plan1 plan2;
  set payroll(drop=salary gender);
  if hired<'01jan98'd then output plan1;
  else output plan2;
run;
```

SALARY および GENDER は、DROP=により SET ステートメントで PAYROLL から読み取りができなくなるため、DATA ステップのどのロジックでも使用できません。

### 例 2: 変数をデータセットに書き込まずに処理する

この例では、SALARY および GENDER は PLAN2 には書き込まれませんが、PLAN1 には書き込まれます。

```
data plan1 plan2(drop=salary gender);
  set payroll;
  if hired<'01jan98'd then output plan1;
  else output plan2;
run;
```

## 関連項目:

### データセットオプション:

- [“KEEP=データセットオプション” \(37 ページ\)](#)

### ステートメント:

- [“DROP Statement” \(SAS Statements: Reference\)](#)

---

## ENCRYPT=データセットオプション

出力 SAS データセットを暗号化するかどうかを指定します。

**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

**デフォルト:** ENCRYPT=NO

**制限事項:** 出力データセットにのみ使用します。

---

## 構文

ENCRYPT=AES | NO | YES

## 構文の説明

### AES

AES (Advanced Encryption Standard)アルゴリズムを使用してファイルを暗号化します。AES では、Base SAS に含まれている SAS/SECURE を使用して拡張暗号化を行います。ENCRYPT=AES を使用する場合、ENCRYPTKEY=データセットオプションを指定する必要があります。詳細については、“[ENCRYPTKEY=データセットオプション](#)” (23 ページ)を参照してください。

**制限事項** TAPE エンジンでは ENCRYPT=AES はサポートされません。TAPE エンジン暗号化には、ENCRYPT=YES を使用します。

**注意** ENCRYPT=AES を使用する場合は ENCRYPTKEY=のすべての値をメモしてください。ENCRYPTKEY=の値をメモすることを忘れた場合、データを失うこととなります。SAS では ENCRYPTKEY=値の回復をサポートできません。次のメッセージが SAS ログに書き込まれます。

Note: If you lose or forget the ENCRYPTKEY= value, there will be no way to open the file or recover the data.

### NO

ファイルを暗号化しません。

### YES

SAS Proprietary アルゴリズムを使用してファイルを暗号化します。この暗号化では、データセットに保存されたパスワードが使用されます。ENCRYPT=YES を指定した場合、少なくとも READ=データセットオプションまたは PW=データセットオプションを同時に指定する必要があります。この暗号化方法ではパスワードが使用されるため、暗号化されたデータセットのパスワードは、データセットを再作成せずに変更することはできません。

#### 注意:

**ENCRYPT=YES を使用する場合はすべてのパスワードをメモしてください。**パスワードを忘れた場合、SAS のサポートなしではリセットできません。これは時間とリソースを消費する処理です。

## 詳細

ENCRYPT=YES のデータファイルを使用するには、SAS 6.11 以降を使用してください。ENCRYPT=YES を使用する場合、次のルールが適用されます。

- 暗号化されたデータファイルをコピーするには、出力エンジンで暗号化がサポートされている必要があります。
- データファイルが暗号化されると、関連付けられたインデックスもすべて暗号化されます。
- PROC CPORT は、SASProprietary 暗号化データファイルに対しては使用できません。

**注:** SAS ビューにはデータが含まれていません。よって、暗号化は必要ありません。

暗号化された AES データファイルを使用するには、SAS 9.4 以降と SAS/SECURE を使用してください。また、ENCRYPT=AES を使用する場合、次のルールが適用されます。

- AES 暗号化でデータセットを作成する場合は、ENCRYPTKEY=データセットオプションを使用する必要があります。

- 暗号化された AES データファイルをコピーするには、出力エンジンで AES 暗号化がサポートされている必要があります。
- Base SAS では、参照一貫性制約付きのデータファイルで AES 暗号化を使用できません。主キーおよび外部キーのデータファイルすべてで同じ暗号化キーを使用し、すべての参照外部キーと主キーのデータファイルを開ける必要があります。

AES 暗号化されたデータセットの ENCRYPTKEY=値は、データセットを再作成せずに変更することはできません。

注: 暗号化には、圧縮とおおよそ同じ量の CPU リソースが必要です。

## 例

### 例 1: ENCRYPT=YES オプションの使用

次の例では、SAS Proprietary アルゴリズムを使用してデータセットを暗号化します。

```
libname mylib "c:\mylib";

data mylib.salary(encrypt=yes read=green);
  input name $ yrsal bonuspct;
  datalines;
Muriel    34567  3.2
Bjorn     74644  2.5
Freda     38755  4.1
Benny     29855  3.5
Agnetha   70998  4.1
;
```

このデータセットを使用するには、READ=パスワードを指定します。

```
proc contents data=mylib.salary(read=green);
run;
```

アウトプット 2.1 ENCRYPT=YES

The SAS System			
The CONTENTS Procedure			
Data Set Name	MYLIB.SALARY	Observations	5
Member Type	DATA	Variables	3
Engine	V9	Indexes	0
Created	04/30/2014 13:37:05	Observation Length	24
Last Modified	04/30/2014 13:37:05	Deleted Observations	0
Protection	READ	Compressed	NO
Data Set Type		Sorted	NO
Encrypted	YES		
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	2715
Obs in First Data Page	5
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	c:\mylib\salary.sas7bdat
Release Created	9.0401M0
Host Created	X64_7PRO

**例 2: ENCRYPT=AES オプションの使用**

```
libname mylib "c:\mylib";

data mylib.salary(encrypt=aes encryptkey=green);
  input name $ yrsal bonuspct;
  datalines;
Muriel    34567  3.2
Bjorn     74644  2.5
Freda     38755  4.1
Benny     29855  3.5
```

```
Agnetha 70998 4.1
;
```

次の例では、SAS Proprietary アルゴリズムを使用してデータセットを暗号化します。

```
proc contents data=mylib.salary(encryptkey=green);
run;
```

#### アウトプット 2.2 ENCRYPT=AES

### The SAS System

#### The CONTENTS Procedure

<b>Data Set Name</b>	MYLIB.SALARY	<b>Observations</b>	5
<b>Member Type</b>	DATA	<b>Variables</b>	3
<b>Engine</b>	V9	<b>Indexes</b>	0
<b>Created</b>	04/30/2014 13:42:52	<b>Observation Length</b>	24
<b>Last Modified</b>	04/30/2014 13:42:52	<b>Deleted Observations</b>	0
<b>Protection</b>		<b>Compressed</b>	NO
<b>Data Set Type</b>		<b>Sorted</b>	NO
<b>Encrypted</b>	AES		
<b>Label</b>			
<b>Data Representation</b>	WINDOWS_64		
<b>Encoding</b>	wlatin1 Western (Windows)		

#### Engine/Host Dependent Information

<b>Data Set Page Size</b>	65536
<b>Number of Data Set Pages</b>	1
<b>First Data Page</b>	1
<b>Max Obs per Page</b>	2715
<b>Obs in First Data Page</b>	5
<b>Number of Data Set Repairs</b>	0
<b>ExtendObsCounter</b>	YES
<b>Filename</b>	c:\mylib\salary.sas7bdat
<b>Release Created</b>	9.0401M0
<b>Host Created</b>	X64_7PRO



**関連項目:**

- “SAS Data File Encryption” (*SAS Language Reference: Concepts*)

**データセットオプション:**

- “ALTER=データセットオプション” (8 ページ)
- “PW=データセットオプション” (54 ページ)
- “READ=データセットオプション” (55 ページ)
- “WRITE=データセットオプション” (75 ページ)

---

**ENCRYPTKEY=データセットオプション**

AES (Advanced Encryption Standard)暗号化のキー値を指定します。

**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

**範囲:** 1 バイトから 64 バイトまで

**制限事項:** SAS 9.4 以降でのみ使用します。  
AES 暗号化データファイルにのみ使用します。

**注:** 暗号化キーのセキュリティを確認するためにこの操作の後にログを確認してください。詳細については、“Blotting Passwords and Encryption Key Values” (*SAS Language Reference: Concepts*)を参照してください。

---

**構文**

ENCRYPTKEY=*key-value*

**構文の説明*****key-value***

暗号化キー値を割り当てます。ENCRYPT=AES を使用する場合、ENCRYPTKEY=データセットオプションを指定する必要があります。キー値の長さは最大で 64 バイトです。ENCRYPTKEY=キー値は、次のルールに従い、必要に応じて引用符を付けて作成します。

**引用符なし:**

- 英数字文字とアンダースコアのみ使用
- 最大 64 バイト
- 大文字と小文字の英字のみ使用
- 先頭は必ず英字にする
- 空白を含まない
- 大文字小文字を区別しない

```
%let mykey=abcdefghi12;
encryptkey=&mykey
encryptkey=key_value
encryptkey=key_value1
```

**一重引用符:**

- 英数字文字、特殊文字および DBCS 文字を使用
- 最大 64 バイト
- 大文字と小文字の英字のみ使用
- 空白を含むことができるが、空白のみは不可
- 大文字小文字を区別する

```
encryptkey='key_value'
encryptkey='1234*#mykey'
```

**二重引用符:**

- 英数字文字、特殊文字および DBCS 文字を使用
- 最大 64 バイト
- 大文字と小文字の英字のみ使用
- 空白を含むことができるが、空白のみは不可
- 大文字小文字を区別する

```
encryptkey="key_value"
encryptkey="1234*#mykey"
%let mykey=Abcdefghi12;
encryptkey("&mykey")
```

ENCRYPTKEY=キー値で DBCS 文字を使用する場合、文字列が UTF-8 エンコーディングに変換された後、その文字列には 64 バイトの制限が適用されません。DBCS によるキー値の長さを計算するには、次の DATA ステップを使用します。

```
data _null_;
  key=length(unicodec('key-value','UTF8'));
  put 'key length=' key;
run;
```

**操作** AES 暗号化されたデータセットのキー値は、そのデータセットを再作成しないと変更できません。

## 詳細

**注意:**

**ENCRYPT=AES** を使用する場合は **ENCRYPTKEY=**のすべての値をメモしてください。**ENCRYPTKEY=**の値をメモすることを忘れた場合、データを失うこととなります。SAS では **ENCRYPTKEY=**値の回復をサポートできません。次のメッセージが SAS ログに書き込まれます。

Note: If you lose or forget the ENCRYPTKEY= value, there will be no way to open the file or recover the data.

AES 暗号化を使用して SAS データセットを作成またはアクセスする場合、**ENCRYPTKEY=**オプションを使用する必要があります。

**ENCRYPTKEY=**データセットオプションは、ファイルを削除や置換から保護するものではありません。次のいずれかの方法を使用すれば、**ENCRYPTKEY=**キー値を指定せずに暗号化されたデータセットを削除できます。

- PROC DATASETS の KILL オプション
- PROC SQL の DROP ステートメント

- DELETE プロシジャ

ENCRYPTKEY=オプションでは、ファイルのコンテンツへのアクセスのみが防止されます。ファイルの削除や置き換えを防止するには、ファイルに ALTER=パスワードを含める必要があります。

AES で暗号化されたデータファイルをコピーする場合、ENCRYPTKEY=キー値を指定する必要があります。SELECT ステートメントで、データセット名の後に値を指定します。SELECT の使用例を次に示します。

```
copy in=OldLib out=NewLib;
  select salary(encryptkey=key-value);
run;
```

DATASETS プロシジャの ENCRYPTKEY=キー値で保護されているデータファイルを処理する場合、AGE、APPEND、AUDIT、CONTENTS、MODIFY、REBUILD、REPAIR の各ステートメントに値を指定できます。また、CHANGE ステートメントがその値への相対参照を使用して特定の世代データセットを参照する場合は、その値も指定する必要があります。

```
change OldName(gennum=-1 encryptkey=key-value)=NewName;
run;
```

オプションは、SAS データファイル名の後にかっこで囲んで指定するか、またはスラッシュの後に指定します。

**注意:**

参照一貫性制約を使用する場合、相互参照する主キーおよび外部キーデータファイルすべてで、同じ暗号化キーを使用する必要があります。詳細については、SAS 言語リファレンス: 解説編の“Encryption and Integrity Constraints”を参照してください。

マクロ変数を ENCRYPTKEY=キー値として使用できます。マクロ変数を定義する例を次に示します。

```
%let secret=myvalue;
```

次の例では、このマクロ変数を ENCRYPTKEY=キー値として使用しています。

```
data my.dsname(encrypt=aes encryptkey="&secret");
```

ENCRYPTKEY=キー値にマクロ変数を指定する場合、マクロ変数を二重引用符で囲む必要があります。二重引用符を使用しないと、予測できない結果を招く可能性があります。

## 例: ENCRYPTKEY=オプションの使用

次の例では ENCRYPT=AES オプションを使用しています。

```
data salary(encrypt=aes encryptkey=green);
  input name $ yrsal bonuspct;
  datalines;
Muriel    34567  3.2
Bjorn     74644  2.5
Freda    38755  4.1
Benny    29855  3.5
Agnetha  70998  4.1
```

このデータセットを使用するには、ENCRYPTKEY=キー値を指定します。

```
proc contents data=salary(encryptkey=green);
run;
```

**関連項目:**

“SAS Data File Encryption” (*SAS Language Reference: Concepts*)

**EXTENDOBSCOUNTER=データセットオプション**

新しい出力 SAS データファイル内の最大オブザベーションカウントを拡張するかどうかを指定します。

<b>該当要素:</b>	DATA ステップおよび PROC ステップ
<b>カテゴリ:</b>	データセットコントロール
<b>別名:</b>	EOC=
<b>デフォルト:</b>	YES
<b>制限事項:</b>	出力データセットにのみ使用します。 BASE Engine でのみ使用します。

**構文**

EXTENDOBSCOUNTER=YES | NO

**構文の説明****YES**

新しく作成された SAS データファイルに、32 ビット長の制限を超えてオブザベーションをカウントする拡張ファイル形式を要求します。この SAS データファイルは 32 ビット整数でオブザベーション数を保存する動作環境に対して作成されますが、データファイルはカウンタに関して 64 ビットファイルと同様に動作します。これがデフォルト設定です。

**制限事項** 拡張オブザベーションカウントで作成した SAS データファイルは、SAS9.3 より前のリリースとは互換性がありません。SAS データファイルが SAS 9.3 以降で EXTENDOBSCOUNTER が YES の設定で作成されていた場合は、その SAS データファイルを EXTENDOBSCOUNTER=NO で再作成する必要があります。

EXTENDOBSCOUNTER=YES は、内部データ表現でオブザベーション数が 32 ビット整数として保存される出力 SAS データファイルに対してのみ有効です。EXTENDOBSCOUNTER=YES は、64 ビット整数の SAS データファイルでは無視されます。EXTENDOBSCOUNTER=YES に適した動作環境と OUTREP=データ表現値の一覧表については、“When Extending the Observation Count Is Supported” (*SAS Language Reference: Concepts*)を参照してください。

**NO**

新しく作成した SAS データファイルの最大オブザベーションカウントを、動作環境の長整数型の大きさによって決めるように指定します。32 ビット整数を使用する動作環境では、最大数は  $2^{31}-1$ 、つまり約 20 億オブザベーション(2,147,483,647)です。64 ビット整数を使用する動作環境では、最大数は  $2^{63}-1$ 、つまり約 920 京オブザベーションです。

**関連項目:**

- “Extending the Observation Count for a 32-Bit SAS Data File” (*SAS Language Reference: Concepts*)

**ステートメント:**

- “EXTENDOBSCOUNTER=YES | NO” (*SAS Statements: Reference*)

**システムオプション:**

- “EXTENDOBSCOUNTER= System Option” (*SAS System Options: Reference*)

---

## FILECLOSE=データセットオプション

SAS データセットを閉じたときのテープの位置付けを指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** その他

**参照項目:** UNIX 動作環境向けドキュメントの FILECLOSE=データセットオプション。

**注意:** このオプション値は、すべての動作環境で認識されるわけではありません。一部の動作環境では、追加の値を使用できます。テープに保存された SAS ライブラリの使用の詳細については、ご使用の動作環境向け SAS ドキュメントを参照してください。

---

### 構文

FILECLOSE=DISP | LEAVE | REREAD | REWIND

#### 構文の説明

##### DISP

テープボリュームを、動作環境のコントロール言語で指定された配置に従って位置付けます。

##### LEAVE

テープを最後に処理したファイルの末尾に位置付けます。FILECLOSE=LEAVE は、SAS プログラムで同じファイルに繰り返しアクセスすることはなく、同じテープ上で後にある 1 つ以上の SAS ファイルにアクセスする場合に使用します。

##### REREAD

テープボリュームを最後に処理したファイルの先頭に位置付けます。FILECLOSE=REREAD は、SAS プログラムで同じテープ上にある同じ SAS データセットに複数回アクセスする場合に使用します。

##### REWIND

テープを最初まで巻き戻します。FILECLOSE=REWIND は、SAS プログラムで同じファイルに繰り返しアクセスすることはなく、同じテープ上で前にある 1 つ以上の SAS ファイルにアクセスする場合に使用します。

---

## FIRSTOBS=データセットオプション

SAS データセット内で SAS が最初に処理するオブザベーションを指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** オブザベーションコントロール

**制限事項:** 入力(読み取り)処理でのみ有効です。

PROC SQL ビューには使用できません。

## 構文

**FIRSTOBS**=*n* | *nK* | *nM* | *nG* | *hexX* | MIN | MAX

### 構文の説明

***n* | *nK* | *nM* | *nG***

最初に処理するオブザベーションの番号を、1(バイト)、1,024(キロバイト)、1,048,576(メガバイト)、1,073,741,824(ギガバイト)のいずれかの倍数で指定します。たとえば、値 8 では 8 番目のオブザベーション、値 3k では 3,072 が指定されます。

***hexX***

最初に処理するオブザベーションの番号を 16 進値で指定します。先頭が数値(0 から 9)、末尾が X の値を指定する必要があります。たとえば、値 2dx では 45 番目のオブザベーションが最初に処理するオブザベーションとして設定されます。

**MIN**

最初に処理するオブザベーションの番号を 1 に設定します。これがデフォルト設定です。

**MAX**

最初に処理するオブザベーションの番号を、データセットの最大オブザベーション数に設定します。8 バイト符号付き整数の最大値である  $2^{63}-1$  (約 920 京オブザベーション)以下の値になります。

## 詳細

FIRSTOBS=データセットオプションは、既存の 1 つの SAS データセットに対してのみ有効です。現在の SAS セッション存続中のすべてのステップで有効にするには、FIRSTOBS=システムオプションを使用します。

FIRSTOBS=は入力(読み取り)処理でのみ有効です。FIRSTOBS=を指定しても、出力または更新処理には無効です。

WHERE 処理には FIRSTOBS=処理を適用できます。詳細については、“Processing a Segment of Data That Is Conditionally Selected” (*SAS Language Reference: Concepts*) を参照してください。

**VIEWTABLE** ウィンドウを使用してデータセットオブザベーションを削除する際に、FIRSTOBS=オプションに割り当てられた値が削除されたオブザベーションの番号より大きい場合、FIRSTOBS=オプションは誤った結果をもたらします。この動作は、**VIEWTABLE** ウィンドウを使用したデータセットオブザベーションの削除はオブザベーションに削除フラグを設定するにすぎないために発生します。オブザベーションは物理的にデータセットから削除されずに、使用できないようになっています。詳細については、“Working with VIEWTABLE” (*SAS Language Reference: Concepts*) を参照してください。

## 比較

- FIRSTOBS=データセットオプションは、個々のデータセットの FIRSTOBS=システムオプションよりも優先されます。
- FIRSTOBS=データセットオプションでは処理の開始点を指定しますが、OBS=データセットオプションでは終了点を指定します。この 2 つのオプションは、多くの場合、処理するオブザベーションの範囲を定義するために使用されます。

- 外部ファイルを読み取る場合、INFILE ステートメントの FIRSTOBS=オプションで最初に読み取るレコードが指定されます。

## 例: FIRSTOBS=データセットオプションの使用

この PROC ステップでは、データセット STUDY をオブザベーション 20 から印刷します。

```
proc print data=study(firstobs=20);
run;
```

この SET ステートメントでは、FIRSTOBS=と OBS=を使用して、データセット STUDY から、オブザベーション 5 から 10 までのみを読み取ります。データセット NEW には 6 個のオブザベーションが含まれます。

```
data new;
  set study(firstobs=5 obs=10);
run;

proc print data=new;
run;
```

## 関連項目:

### データセットオプション:

- “OBS=データセットオプション” (42 ページ)

### ステートメント:

- “INFILE Statement” (*SAS Statements: Reference*)
- “WHERE Statement” (*SAS Statements: Reference*)

### システムオプション:

- “FIRSTOBS= System Option” (*SAS System Options: Reference*)

---

## GENMAX=データセットオプション

新しいデータセットの世代を要求し、既存のデータセットの世代数を変更し、バージョンの最大数を指定します。

- 該当要素:** DATA ステップおよび PROC ステップ
- カテゴリ:** データセットコントロール
- 制限事項:** 出力データセットにのみ使用します。
- 

## 構文

GENMAX=*number-of-generations*

## 構文の説明

### *number-of-generations*

データセットの世代を要求し、維持する最大バージョン数を指定します。0 から 1,000 までの値を指定できます。デフォルトの GENMAX=0 は、世代データセットを要求しないことを意味します。

## 詳細

GENMAX=は、新しいデータセットの世代を要求したり、既存のデータセットの世代数を変更したりする場合に使用します。データセットが初めて置き換えられるとき、SAS では置き換えられたバージョンが保持され、メンバ名に 4 文字のバージョン番号が付加されます。メンバ名には、#と 3 桁の数字が含まれます。たとえば、A という名前のデータセットの場合、履歴バージョンは A#001 になります。

データセットの世代が要求されると、メンバ名は 28 文字に制限されます(32 文字ではない)。末尾の 4 文字は、付加されるバージョン番号用に予約されます。GENMAX=データセットオプションを 0 に設定すると、メンバ名を最大 32 文字にすることができます。

既存のデータセットの世代数を減らすと、新しい制限を超えるバージョンは SAS により古い方から削除されます。

## 例

### 例 1: データセット作成時の世代の要求

DATA ステップで WORK.A という名前のデータセットが作成されます。このデータセットは最大 10 世代を持つことができます(1 個の最新バージョンと 9 個の履歴バージョン)。

```
data a (genmax=10);
  x=1;
  output;
run;
```

### 例 2: 既存のデータセットの世代数の変更

データセット MYLIB.A の世代数を 4 に変更します。

```
proc datasets lib=mylib;
  modify a (genmax=4);
run;
```

## 関連項目:

- “Understanding Generation Data Sets” (*SAS Language Reference: Concepts*)

### データセットオプション:

- “GENNUM=データセットオプション” (30 ページ)

---

## GENNUM=データセットオプション

SAS データセットの特定の世代を指定します。

**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  
 制限事項: 入力データセットにのみ使用します。

## 構文

GENNUM=*integer*

### 構文の説明

#### *integer*

世代グループ内の特定のバージョンを参照する番号です。正の数を指定すると、データセット名に付加されている特定の世代番号が絶対参照されます。負の数を指定すると、基本(現在の)バージョンに対する履歴バージョンが新しい順に相対参照されます。通常、値 0 が基本バージョンを示します。

DATASETS プロシジャでは、さまざまなステートメントで GENNUM=を指定して追加機能を使用できます。

- DATASETS および DELETE ステートメントの場合、GENNUM=では追加の値 ALL、HIST および REVERT がサポートされます。
- DELETE プロシジャの場合、GENNUM=では追加の値 ALL、HIST および REVERT がサポートされます。
- CHANGE ステートメントの場合、GENNUM=では追加の値 ALL がサポートされます。
- CHANGE ステートメントの場合、GENNUM=0 を指定すると基本バージョンだけでなくすべてのバージョンを示します。

## 詳細

GENMAX=データセットオプションを使用してデータセットの世代を要求した後、特定のバージョンを要求するには GENNUM=を使用します。たとえば、GENNUM=3 を指定すると履歴バージョン#003 を示し、GENNUM=-1 を指定すると最も新しい履歴バージョンを示します。

置き換えが 999 回行われると、最も新しいバージョンは#999 になります。置き換えが 1,000 回行われると、最も新しいバージョン番号は#000 に戻ります。そのため、履歴バージョン#000 が必要な場合は、GENNUM=1000 を指定します。

絶対参照と相対参照のどちらも、特定のバージョンを参照します。相対参照では、削除されたバージョンがスキップされません。そのため、削除されたバージョンが 1 つ以上含まれる世代グループを操作する場合、相対参照を使用して参照したバージョンが削除されているとエラーが発生します。たとえば、基本バージョン AIR と 3 つの履歴バージョン(AIR#001、AIR#002、AIR#003)がある場合に、AIR#002 を削除したとします。次のステートメントでは AIR#002 が存在しないためにエラーが返されます。

```
proc print data=air (gennum= -2);
run;
```

## 例

### 例 1: 絶対参照を使用してバージョンを要求する

この例では、絶対参照を使用してデータセット A の履歴バージョン#003 を印刷します。

```
proc print data=a(gennum=3);
```

```
run;
```

### 例 2: 相対参照を使用してバージョンを要求する

次の PRINT プロシジャでは、基本バージョンから 3 バージョン前のデータセットを印刷します。

```
proc print data=a(gennum=-3);
run;
```

### 関連項目:

- “Understanding Generation Data Sets” (*SAS Language Reference: Concepts*)
- “DATASETS” (*Base SAS Procedures Guide*)

### データセットオプション:

- “GENMAX=データセットオプション” (29 ページ)

---

## IDXNAME=データセットオプション

WHERE 式の条件の照合に特定のインデックスを使用するように SAS に指示します。

- 該当要素:** DATA ステップおよび PROC ステップ
- カテゴリ:** SAS インデックス使用のユーザーコントロール
- 制限事項:** 入力データセットにのみ使用します。  
IDXWHERE=データセットオプションと相互排他的

### 構文

IDXNAME=*index-name*

### 構文の説明

#### *index-name*

SAS データセットの単一インデックスまたは複合インデックスの名前(32 文字以内)を指定します。指定したインデックスが最適なものであるかどうか、順次検索の方がリソースをより効率的に利用できるかどうかについて SAS は確認しません。

**操作** この指定はデータセットの永続的な属性ではなく、現在の使用に対してのみ有効です。

**ヒント** IDXNAME=の使用を SAS ログに表示するように要求するには、システムオプション MSGLEVEL=I を指定します。

### 詳細

インデックス付き SAS データセットの WHERE 式の条件を満たすため、SAS は WHERE 式の最適化に使用できる 0 個以上の候補インデックスを識別します。SAS は、候補インデックスリストの中から次を決定します。

- 最適なパフォーマンスを提供する候補インデックス
- データを順次に確認した方がより効率的な場合は、すべてのインデックスを拒否

SAS が選択するインデックスは常に最適であるとは限らないため、IDXNAME=データセットオプションを指定することで、いずれかの候補インデックスを使用するように SAS に指示できます。SAS によって候補インデックスとして識別されないインデックスを指定した場合、IDXNAME=では要求が処理されません。つまり、IDXNAME=には結果が無効となるインデックスを指定できません。

## 比較

IDXWHERE=では、インデックスを使用するかどうかの SAS の決定を変更できますが、IDXNAME=では、特定のインデックスを使用するように SAS に指示できます。

## 例: インデックスの指定

この例では、IDXNAME=データセットオプションを使用して、WHERE 式の最適化に特定のインデックスを使用するように SAS に指示します。SAS は、データセットの順次検索の方がリソースをより効率的に利用できる可能性を無視します。SAS は、指定したインデックスが最適なものであるかどうかについて確認しません。(EMPNUM インデックスは NOMISS オプションを使用して作成されていません)

```
data mydata.empnew;
  set mydata.employee (idxname=empnum);
  where empnum < 2000;
run;
```

## 関連項目:

- “Using an Index for WHERE Processing” (*SAS Language Reference: Concepts*)

## データセットオプション:

- “IDXWHERE=データセットオプション” (33 ページ)

---

## IDXWHERE=データセットオプション

SAS で WHERE 式の条件の照合にインデックス検索を使用するか、順次検索を使用するかを指定します。

<b>該当要素:</b>	DATA ステップおよび PROC ステップ
<b>カテゴリ:</b>	SAS インデックス使用のユーザーコントロール
<b>制限事項:</b>	入力データセットにのみ使用します。 IDXNAME=データセットオプションと相互排他的

## 構文

IDXWHERE=YES | NO

### 構文の説明

YES

WHERE 式の最適化に最適なインデックスを選択し、データセットの順次検索の方がリソースをより効率的に利用できる可能性を無視するように SAS に指示します。

NO

すべてのインデックスを無視し、データセットの順次検索で WHERE 式の条件を満たすように SAS に指示します。

注: IDXWHERE=を使用して、BY ステートメントの処理に対するインデックスの使用を無効にすることはできません。

## 詳細

デフォルトでは、インデックス付き SAS データセットの WHERE 式の条件を満たすため、SAS はインデックスを使用するか、データセットを順次に読み取るかを決定します。ソフトウェアによって相対的な効率性が推定され、より効率的な方法が選択されません。

IDXWHERE=データセットオプションを指定して、ソフトウェアの決定を無効にすることが必要な場合があります。ソフトウェアの決定は、一般的なルールに基づいているため、必ずしも最適な結果になるとは限りません。つまり、IDXWHERE=データセットオプションを指定することによって、処理方法を決定できます。

注: この指定はデータセットの永続的な属性ではなく、現在の使用に対してのみ有効です。

注: システムオプション MSGLEVEL=I を指定すると、設定がインデックス処理に影響する場合は IDXWHERE=の使用を SAS ログに表示するように要求できます。

## 比較

IDXNAME=では、特定のインデックスを使用するように SAS に指示できますが、IDXWHERE=では、インデックスを使用するかどうかの SAS の決定を無効にできません。

## 例

### 例 1: インデックスの使用を指定する

この例では、IDXWHERE=データセットオプションを使用して、WHERE 式の最適化に最適なインデックスを決定するように SAS に指示します。SAS は、データセットの順次検索の方がリソースをより効率的に利用できる可能性を無視します。

```
data mydata.empnew;
  set mydata.employee (idxwhere=yes);
  where empnum < 2000;
```

### 例 2: インデックスを使用しないように指定する

この例では、IDXWHERE=データセットオプションを使用して、次のタスクを実行します。

- すべてのインデックスを無視する
- データセットの順次検索で WHERE 式の条件を満たす

```
data mydata.empnew;
  set mydata.employee (idxwhere=no);
  where empnum < 2000;
```

## 関連項目:

- “Understanding SAS Indexes” (*SAS Language Reference: Concepts*)
- WHERE 式の処理

## データセットオプション:

- “IDXNAME=データセットオプション” (32 ページ)

## INDEX=データセットオプション

新しい出力 SAS データセットのインデックスを定義します。

<b>該当要素:</b>	DATA ステップおよび PROC ステップ
<b>カテゴリ:</b>	データセットコントロール
<b>制限事項:</b>	出力データセットにのみ使用します。

### 構文

INDEX=(*index-specification* <*index-specification-2* ...> )

#### 構文の説明

##### *index-specification*

作成する単一インデックスまたは複合インデックスの名前および説明を指定します。*Index-specification* は次のような形式になります。

*index* = (*variable(s)* </UNIQUE> </NOMISS> )

##### *index*

インデックスを形成する変数名または複合インデックス用に選択する名前です。

##### *variable(s)*

複合インデックスの作成に使用する変数のリストです。

##### UNIQUE

キー変数の値は一意である必要があることを指定します。新しいデータセットに UNIQUE を指定し、複数のオブザベーションのインデックス変数の値が同じ場合、インデックスは作成されません。UNIQUE オプションの前にスラッシュ(/)を付ける必要があります。

##### NOMISS

欠損値を含むすべてのオブザベーションをインデックスから除外します。欠損値を含むオブザベーションは引き続きデータセットから読み込まれますが、インデックスからは読み込まれません。NOMISS オプションの前にスラッシュ(/)を付ける必要があります。

### 例

#### 例 1: 単一インデックスの定義

次の INDEX=データセットオプションでは、SSN 変数の単一インデックスを定義します。

```
data new(index=(ssn));
```

#### 例 2: 複合インデックスの定義

次の INDEX=データセットオプションでは、CITY 変数および STATE 変数を使用する CITYST という名前の複合インデックスを定義します。

```
data new(index=(cityst=(city state)));
```

**例 3: 単一インデックスおよび複合インデックスの定義**

次の INDEX=データセットオプションでは、SSN の単一インデックスおよび CITY と STATE の複合インデックスを定義します。

```
data new(index=(ssn cityst=(city state)));
```

**例 4: UNIQUE オプションを使用した単一インデックスの定義**

次の INDEX=データセットオプションでは、一意の値を含む SSN 変数の単一インデックスを定義します。

```
data new(index=(ssn /unique));
```

**例 5: NOMISS オプションを使用した単一インデックスの定義**

次の INDEX=データセットオプションでは、欠損値を含むすべてのオブザベーションをインデックスから除外した SSN 変数の単一インデックスを定義します。

```
data new(index=(ssn /nomiss));
```

**例 6: UNIQUE オプションおよび NOMISS オプションを使用して複数のインデックスを定義する**

次の INDEX=データセットオプションでは、SSN 変数の単一インデックスおよび CITY と STATE の複合インデックスを定義します。各変数に UNIQUE オプションおよび NOMISS オプションが設定されている必要があります。

```
data new(index=(ssn /unique/nomiss cityst=(city state) /unique/nomiss));
```

**関連項目:**

- “INDEX CREATE Statement” (*Base SAS Procedures Guide*)
- “CREATE INDEX Statement” (*SAS SQL Procedure User's Guide*)
- “Understanding SAS Indexes” (*SAS Language Reference: Concepts*)

---

**IN=データセットオプション**

データセットのデータが現在のオブザベーションに関与しているかどうかを示すブール値変数を作成します。

**該当要素:** DATA ステップ

**カテゴリ:** オブザベーションコントロール

**制限事項:** SET、MERGE、MODIFY、UPDATE ステートメントのみとともに使用します。

---

**構文**

IN=*variable*

**構文の説明**

*variable*

入力データセットのデータが現在のオブザベーションに関与しているかどうかを示す値を含む、新しい変数の名前を指定します。DATA ステップ内での変数の値は、データセットが現在のオブザベーションに関与している場合は 1 です。それ以外の場合は 0 です。

## 詳細

IN=データセットオプションは SET、MERGE、MODIFY、UPDATE ステートメントのみに使用し、SAS データセット名の後にかっこで囲んで指定します。IN=変数の値は、DATA ステップ内のプログラムステートメントで使用できます。この変数は、新しい変数に割り当てられない限り、作成される SAS データセットには含まれません。

BY グループ処理で IN=を使用し、現在の BY グループでデータセットがオブザベーションに関与する場合、IN=の値は 1 です。この値は BY グループの処理中はそのまま保持され、プログラミングロジックによってリセットされません。

## 例

この例では、国際飛行便を示す新しい変数、OVERSEAS を IN=で作成します。変数 I の値は、オブザベーションが NONUSA データセットから読み込まれている場合は 1 になります。それ以外の場合、この変数の値は 0 になります。IF-THEN ステートメントで I の値を確認し、データセット NONUSA のデータが現在のオブザベーションに関与しているかどうかを判断します。I=1 の場合、変数 OVERSEAS には値としてアスタリスク(\*)が渡されます。

```
data allflts;
  set usa nonusa(in=i);
  by fltnum;
  if i then overseas='*';
run;
```

## 関連項目:

- BY-GROUP 処理

## ステートメント:

- “BY Statement” (*SAS Statements: Reference*)
- “MERGE Statement” (*SAS Statements: Reference*)
- “MODIFY Statement” (*SAS Statements: Reference*)
- “SET Statement” (*SAS Statements: Reference*)
- “UPDATE Statement” (*SAS Statements: Reference*)

---

## KEEP=データセットオプション

入力データセットの場合は、処理する変数を指定し、出力データセットの場合は、データセットに書き込む変数を指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** 変数コントロール

## 構文

KEEP=*variable(s)*

## 構文の説明

### *variable(s)*

1 つ以上の変数名のリストを指定します。SAS で許可される形式の変数のリストを指定できます。

## 詳細

KEEP=データセットオプションが入力データセットに関連付けられている場合、KEEP=データセットオプションの後に示された変数のみを処理に使用できます。KEEP=データセットオプションが出力データセットに関連付けられている場合、このオプションの後に示された変数のみが出力データセットに書き込まれます。すべての変数を処理に使用できます。

## 比較

KEEP=データセットオプションと KEEP ステートメントには次の違いがあります。

- DATA ステップでは、KEEP=データセットオプションは入力および出力データセットに適用できます。KEEP ステートメントは、出力データセットにのみ適用されます。
- DATA ステップで複数の出力データセットを作成する場合、異なる変数を異なるデータセットに書き込むには KEEP=データセットオプションを使用します。KEEP ステートメントはすべての出力データセットに適用されます。
- PROC ステップでは、KEEP=データセットオプションのみを使用でき、KEEP ステートメントは使用できません。

## 例

この例では、IDNUM および SALARY のみが PAYROLL から読み込まれ、この 2 つのみが処理で使用できる PAYROLL 内の変数です。

```
data bonus;
  set payroll(keep=idnum salary);
  bonus=salary*1.1;
run;
```

## 関連項目:

### データセットオプション:

- [“DROP=データセットオプション” \(17 ページ\)](#)

### ステートメント:

- [“KEEP Statement” \(SAS Statements: Reference\)](#)

---

## LABEL=データセットオプション

SAS データセットのラベルを指定します。

**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

---



## 構文

LABEL=*'label'*

### 構文の説明

*'label'*

256 文字までのテキスト文字列を指定します。ラベルテキストに一重引用符が含まれる場合、ラベルを二重引用符で囲みます。データセットからラベルを削除するには、ラベルに引用符で囲んだ空白を割り当てます。

ラベルテキストに 2 つの一重引用符を使用し、この文字列を一重引用符で囲むこともできます。

## 詳細

LABEL=オプションは、入力データセットと出力データセットで使用できます。入力データセットで LABEL=を指定する場合、ファイルに割り当てられるラベルは DATA ステップまたは PROC ステップの間のみ有効です。出力データセットに LABEL=を指定する場合、ラベルは永続的にそのファイルのラベルになります。ファイルを出力するには CONTENTS または DATASETS プロシジャを使用し、変更するには PROC DATASETS を使用します。

データセットに割り当てられたラベルは、APPEND プロシジャまたは MODIFY ステートメントなどを使用してデータセットを更新する場合はそのデータセットに関連付けられたままになります。ただし、以前にラベルが割り当てられているデータセットを使用して DATA ステップで新しいデータセットを作成する場合、ラベルは失われます。例:

```
data one;
    set one;
run;
```

## 比較

- LABEL=データセットオプションでは、データセットのみにラベルを割り当てられます。データセット内の変数のラベルは、LABEL ステートメントを使用して指定できます。
- ATTRIB ステートメントの LABEL=オプションでも、変数にラベルを割り当てることができます。

## 例: データセットにラベルを割り当てる

```
data w2(label='1976 W2 Info, Hourly');
data new(label='Peter''s List');
data new(label="Hillside's Daily Account");
data sales(label='Sales For May(NE)');
```

## 関連項目:

### ステートメント:

- “ATTRIB Statement” (*SAS Statements: Reference*)
- “LABEL Statement” (*SAS Statements: Reference*)
- “MODIFY Statement” (*SAS Statements: Reference*)

**プロシジャ:**

- “CONTENTS” (*Base SAS Procedures Guide*)
- “DATASETS” (*Base SAS Procedures Guide*)

**OBSBUF=データセットオプション**

DATA ステップビューを処理する表示バッファのサイズを決定します。

- 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
- 制限事項:** DATA ステップビューでのみ有効です。

**構文**

**OBSBUF=*n***

**構文の説明**

*n*

一度に表示バッファに読み込まれるオブザベーション数を指定します。

**デフォルト** デフォルトの表示バッファには 32K バイトのメモリが割り当てられます。このため、一度に表示バッファに読み込むことができるデフォルトのオブザベーション数は、オブザベーションの長さによって異なります。そのため、デフォルトは 32K バイトに収まるオブザベーション数です。オブザベーションの長さが 32K バイトを超える場合、一度にバッファに読み込むことができるオブザベーションは 1 つのみです。

**ヒント** オブザベーションの長さ(バイト)を確認するには、DATA ステップビューに PROC CONTENTS を使用します。

**注意** **OBSBUF=オプションの最大値は、使用可能なメモリ量によって異なります。** 指定した値が大きすぎて表示バッファのメモリ割り当てに失敗した場合、メモリ不足エラーが発生します。

**詳細**

OBSBUF=データセットオプションでは、一度に表示バッファに読み込むことができるオブザベーション数を指定します。表示バッファは、DATA ステップビューから生成される出力オブザベーションを保持するために割り当てられたメモリのセグメントです。バッファのサイズによって、一度にメモリ内に保持できるデータ量が決定されます。

OBSBUF=では、DATA ステップビューからのデータ読み込みのパフォーマンスを調整できます。

表示バッファは、DATA ステップビューを開く要求(SAS プロシジャなど)と DATA ステップビュー自体の間で共有されます。2 つのコンピュータタスクは、データの要求とデータの生成や返送間で次のように調整されます。

- 要求タスク(PRINT プロシジャなど)がデータを要求すると、要求タスクから表示タスクへのタスクの切り替えが発生します。このアクションは DATA ステップビューを実行してオブザベーションを生成します。DATA ステップビューによって、表示バッファに可能な限り多くのオブザベーションが挿入されます。

- 表示バッファがいっぱいになると、要求されたデータを返すために表示タスクから要求タスクへのタスクの切り替えが発生します。オブザベーションは表示バッファから解除されます。

表示バッファのサイズによって、保持できる生成されたオブザベーション数が決定されます。生成されたオブザベーション数によって、コンピュータが要求タスクと表示タスク間の切り替えを行う必要がある回数決定されます。たとえば、OBSBUF=1 では各オブザベーションごとにタスクの切り替えが発生します。OBSBUF=10 では一度に 10 個のオブザベーションが表示バッファに読み込まれます。表示バッファを大きくすると、DATA ステップビューの処理に必要なタスクの切り替え数は少なくなり、実行時間が短縮できます。

効率を高めるには、デフォルトのバッファサイズに収まるオブザベーション数を確認してから、より多くの生成されたオブザベーションを保持できるように表示バッファを設定します。

注: OBSBUF=を使用し、タスクの切り替え数を減らすことで処理効率を高めることができます。ただし、表示バッファサイズを大きくすると、バッファがいっぱいになるまでにかかる時間が長くなります。この処理により、表示タスクから要求タスクへ戻すタスクの切り替えで遅延が発生します。この遅延は、対話型アプリケーションでより顕著です。たとえば、Viewtable ウィンドウを使用する場合、表示バッファを大きくすると、要求したオブザベーションの表示にかかる時間が長くなります。1 個のオブザベーションのみが Viewtable に返される場合でも、その前に表示バッファがいっぱいになります。表示バッファサイズをかなり大きく設定する前には、次の情報を検討します。

- DATA ステップビューの処理に使用するアプリケーションの種類
- 使用可能なメモリ量

## 例

この例では、オブザベーションの長さは 10K です。デフォルトの表示バッファサイズ (32K) では、一度に 3 個のオブザベーションが表示バッファに読み込まれます。デフォルトの表示バッファサイズでは、3 個のオブザベーションが生成されるたびにコンピュータはタスクを切り替える必要があるため、実行時間が長くなります。

パフォーマンスを向上させるには、OBSBUF=データセットオプションを 100 に設定します。このアクションでは、一度に 100 個のオブザベーションが表示バッファに読み込まれます。また、PRINT プロシジャで DATA ステップビューを処理するためのタスクの切り替え数が削減されます。

```
data testview / view=testview;
    ... more SAS statements ...
run;
proc print data=testview (obsbuf=100);
run;
```

## 関連項目:

### データセットオプション:

- “SPILL=データセットオプション” (66 ページ)

## OBS=データセットオプション

データセット内で SAS が最後に処理するオブザベーションを指定します。

<b>該当要素:</b>	DATA ステップおよび PROC ステップ
<b>カテゴリ:</b>	オブザベーションコントロール
<b>デフォルト:</b>	MAX
<b>制限事項:</b>	入力データセットにのみ使用します。 PROC SQL ビューには使用できません。

### 構文

OBS= *n* | *nK* | *nM* | *nG* | *nT* | *hexX* | MIN | MAX

#### 構文の説明

*n* | *nK* | *nM* | *nG* | *nT*

処理を停止するタイミングを示す数を整数 *n* で指定します。いずれかの文字表記法を使用すると、整数 × 指定値の結果になります。具体的には、指定表記が K (キロ) の場合は 1,024、M (メガ) の場合は 1,048,576、G (ギガ) の場合は 1,073,741,824、T (テラ) の場合は 1,099,511,627,776 の整数倍になります。たとえば、値 20 では 20 オブザベーション、値 3m では 3,145,728 オブザベーションが指定されます。

*hexX*

処理を停止するタイミングを示す数を 16 進値で指定します。先頭が数値(0 から 9)、末尾が X の値を指定する必要があります。たとえば、10 進値の 248 に相当する 16 進値 F8 を指定するには、0F8x と指定する必要があります。値 2dx では、10 進値での 45 が指定されます。

MIN

処理を停止するタイミングを示す数を 0 に設定します。OBS=0 を使用すると空のデータセットが作成されます。このデータセットは、もう 1 つのデータセットの構造を備え、そのオブザベーションは含みません。

**操** OBS=0 で NOREPLACE オプションが有効になっている場合、SAS は特定の  
**作** 操作を引き続き実行できます。実際、SAS は、オブザベーションを使用せずにプログラムの各 DATA ステップおよび PROC ステップを実行します。たとえば、ライブラリまたは SAS データセットを処理する、CONTENTS および DATASETS などのプロシジャを実行します。

MAX

処理を停止するタイミングを示す数をデータセット内の最大オブザベーション数に指定します。8 バイト符号付き整数の最大値である  $2^{63}-1$  (約 920 京オブザベーション) 以下の値になります。これがデフォルト設定です。

### 詳細

OBS=では、オブザベーションの処理を停止するタイミングを SAS に指示します。処理を停止するタイミングを判断するため、SAS は計算式で OBS=の値を使用します。この計算式には OBS=の値と FIRSTOBS=の値が含まれます。

$(\text{obs} - \text{firstobs}) + 1 = \text{results}$

たとえば、OBS=10 で FIRSTOBS=1(FIRSTOBS=のデフォルト値)の場合、結果は 10 個のオブザベーションになります。つまり、 $(10 - 1) + 1 = 10$  となります。OBS=10 で FIRSTOBS=2 の場合の結果は、9 個のオブザベーションになります。つまり、 $(10 - 2) + 1 = 9$  となります。OBS=は、既存の SAS データセットが読み込まれている場合にのみ有効です。

OBS=データセットオプションは、個々のデータセットの OBS=システムオプションよりも優先されます。

## 比較

- OBS=データセットオプションでは処理の終点を指定しますが、FIRSTOBS=データセットオプションでは開始点を指定します。この 2 つのオプションは、多くの場合、処理するオブザベーションの範囲を定義するために使用されます。
- OBS=データセットオプションでは、SAS データセットからオブザベーションを選択できます。INFILE ステートメントで OBS=オプションを使用することで、外部データファイルから読み込まれるオブザベーションを選択できます。

## 例

### 例 1: OBS=を使用したオブザベーションの処理を停止するタイミングを指定する

この例では、SAS データセットを作成し、FIRSTOBS=2 および OBS=12 を設定した PRINT プロシジャを実行します。結果は、11 個のオブザベーションになります。つまり、 $(12 - 2) + 1 = 11$  となります。OBS=の結果は SAS が最後に処理するオブザベーション番号であるように見えます。

```
data Ages;
    input Name $ Age;
    datalines;
Miguel 53
Brad 27
Willie 69
Marc 50
Sylvia 40
Arun 25
Gary 40
Becky 51
Alma 39
Tom 62
Kris 66
Paul 60
Randy 43
Barbara 52
Virginia 72
;
proc print data=Ages (firstobs=2 obs=12);
run;
```

## アウトプット 2.3 OBS=およびFIRSTOBS=を使用した PROC PRINT の出力

**The SAS System**

Obs	Name	Age
2	Brad	27
3	Willie	69
4	Marc	50
5	Sylvia	40
6	Arun	25
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
11	Kris	66
12	Paul	60

**例 2: WHERE ステートメントを使用した PROC PRINT**

この例では、例 1 で作成されたデータセットを使用し、それには 15 個のオブザベーションが含まれています。

WHERE ステートメントを含む PRINT プロシジャをここに示します。データのサブセットの結果は 12 個のオブザベーションになります。

```
proc print data=Ages;  
  where Age LT 65;  
run;
```

## アウトプット 2.4 WHERE ステートメントを使用した PROC PRINT の出力

**The SAS System**

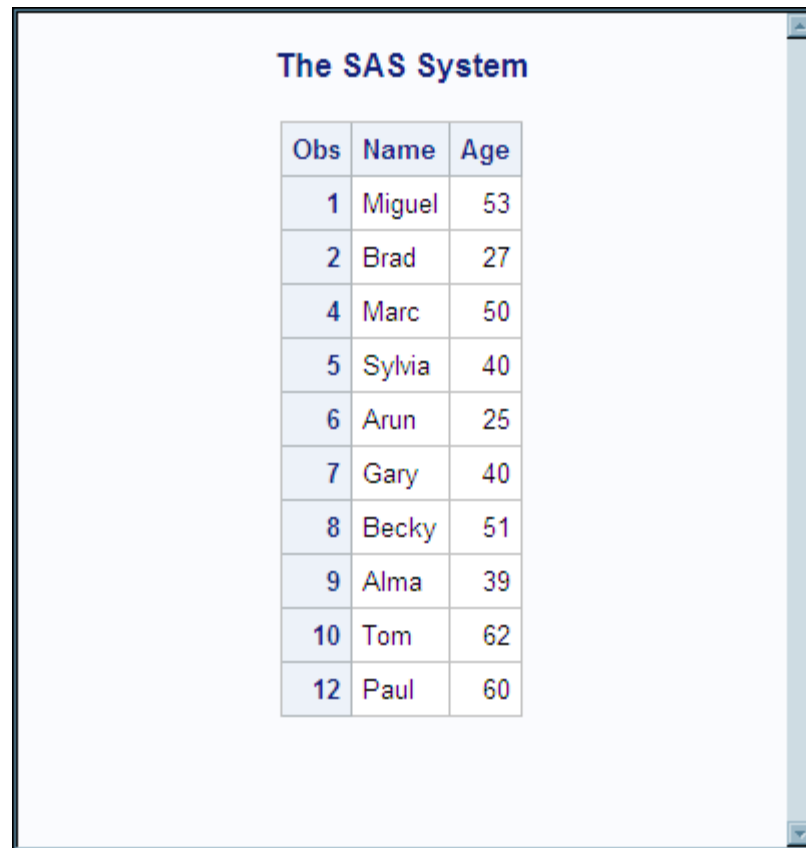
Obs	Name	Age
1	Miguel	53
2	Brad	27
4	Marc	50
5	Sylvia	40
6	Arun	25
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
12	Paul	60
13	Randy	43
14	Barbara	52

**例 3: WHERE ステートメントおよび OBS=を使用した PROC PRINT**

WHERE ステートメントおよび OBS=10 を使用した PRINT プロシジャを実行すると、結果は 10 個のオブザベーションになります。つまり、 $(10 - 1) + 1 = 10$  となります。WHERE 処理を使用して、SAS はデータをサブセット化し、そのサブセットに OBS= を適用します。

```
proc print data=Ages (obs=10);  
  where Age LT 65;  
run;
```

## アウトプット 2.5 WHERE ステートメントおよび OBS=を使用した PROC PRINT の出力



The SAS System

Obs	Name	Age
1	Miguel	53
2	Brad	27
4	Marc	50
5	Sylvia	40
6	Arun	25
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
12	Paul	60

**例 4: WHERE ステートメント、OBS=および FIRSTOBS=を使用した PROC PRINT**

OBS=の結果は SAS が処理するオブザベーション番号であるように見えます。FIRSTOBS=2 および OBS=10 をサブセットに適用した場合、結果は 9 個のオブザベーションになります。つまり、 $(10 - 2) + 1 = 9$  となります。OBS=は、最後のオブザベーション番号でも、処理するオブザベーション数でもありません。値は、処理を停止するタイミングを判別する計算式で使用されます。

```
proc print data=Ages (firstobs=2 obs=10);  
  where Age LT 65;  
run;
```



**アウトプット 2.6** WHERE ステートメント、OBS=およびFIRSTOBS=を使用した PROC PRINT の出力



The screenshot shows a window titled "The SAS System" containing a table with three columns: Obs, Name, and Age. The table lists 12 observations, with observation 6 missing. The data is as follows:

Obs	Name	Age
2	Brad	27
4	Marc	50
5	Sylvia	40
6	Arun	25
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
12	Paul	60

**例 5: 削除されたオブザベーションを示す PROC PRINT**

この例では、例 1 で作成されたデータセットからオブザベーション 6 が削除されたデータセットを使用します。

変更されたファイルの PROC PRINT の出力を次に示します。

```
proc print data=Ages;  
run;
```

## アウトプット 2.7 オブザベーション 6 が削除されたことを示す PROC PRINT の出力

**The SAS System**

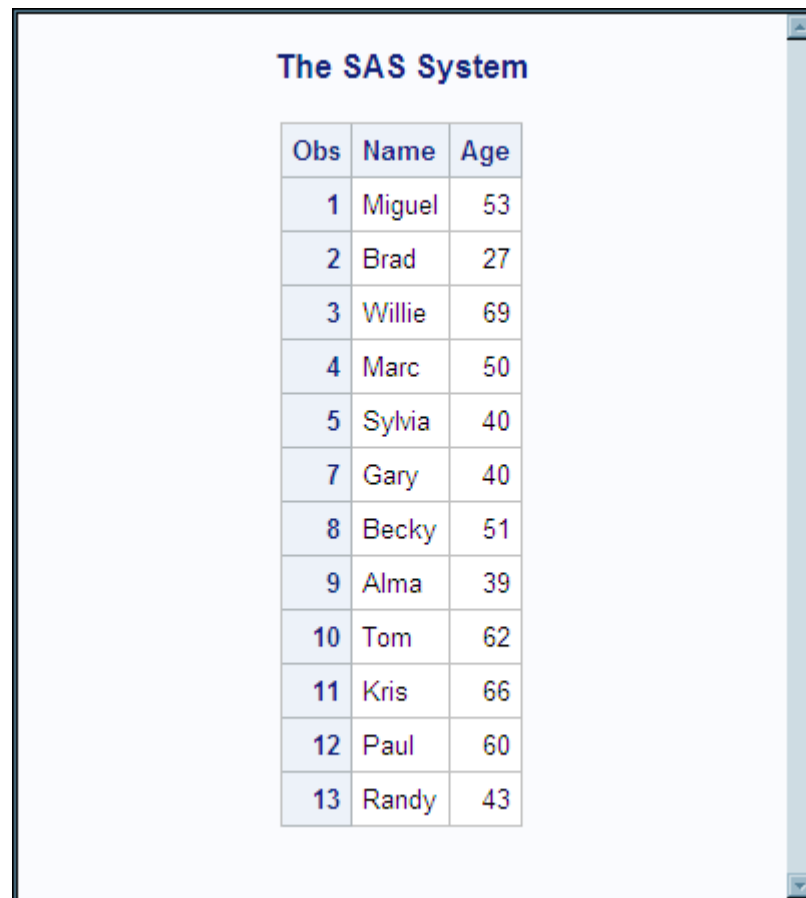
Obs	Name	Age
1	Miguel	53
2	Brad	27
3	Willie	69
4	Marc	50
5	Sylvia	40
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
11	Kris	66
12	Paul	60
13	Randy	43
14	Barbara	52
15	Virginia	72

**例 6: OBS=を使用した PROC PRINT**

OBS=12 を使用した PRINT プロシジャを実行すると、結果は 12 個のオブザベーションになります。つまり、 $(12 - 1) + 1 = 12$  となります。

```
proc print data=Ages (obs=12);  
run;
```

## アウトプット 2.8 OBS=を使用した PROC PRINT の出力



The SAS System

Obs	Name	Age
1	Miguel	53
2	Brad	27
3	Willie	69
4	Marc	50
5	Sylvia	40
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
11	Kris	66
12	Paul	60
13	Randy	43

**例 7: オブザベーションが削除された場合に OBS=を使用する**

OBS=の結果は SAS が処理するオブザベーション番号であるように見えます。ただし、FIRSTOBS=2 で OBS=12 を適用する場合の結果は、11 個のオブザベーションになります。つまり、 $(12 - 2) + 1 = 11$  となります。OBS=は、最後のオブザベーション番号でも、処理するオブザベーション数でもありません。値は、処理を停止するタイミングを判別する計算式で使用されます。

```
proc print data=Ages (firstobs=2 obs=12);  
run;
```

アウトプット 2.9 OBS=およびFIRSTOBS=を使用した PROC PRINT の出力



The SAS System

Obs	Name	Age
2	Brad	27
3	Willie	69
4	Marc	50
5	Sylvia	40
7	Gary	40
8	Becky	51
9	Alma	39
10	Tom	62
11	Kris	66
12	Paul	60
13	Randy	43

**関連項目:**

- “Processing a Segment of Data That Is Conditionally Selected” (*SAS Language Reference: Concepts*)

**データセットオプション:**

- “FIRSTOBS=データセットオプション” (27 ページ)

**ステートメント:**

- “INFILE Statement” (*SAS Statements: Reference*)
- “WHERE Statement” (*SAS Statements: Reference*)
- SAS ステートメント: リファレンス

**システムオプション:**

- “OBS= System Option” (*SAS System Options: Reference*)

---

**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現を指定します。

**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:** z/OS 版 SAS の OUTREP=データセットオプション

## 構文

OUTREP=*format*

### 構文の説明

#### *format*

特定の動作環境でデータを保存するときの形式のデータ表現を指定します。動作環境が異なると、使用される標準や規則も異なります。たとえば、浮動小数点数を保存する場合は IEEE や IBM メインフレーム、文字エンコーディングの場合は ASCII または EBCDIC、メモリ内のバイトの順序付けではビッグエンディアンまたはリトルエンディアン、ワード配置では 4 バイト境界または 8 バイト境界、整数のデータ型の長さでは 16 ビット、32 ビットまたは 64 ビットが使用され、倍精度の場合はバイト交換される場合とされない場合があります。

デフォルトでは、SAS を実行している CPU のデータ表現を使用して新しい SAS データセットが作成されます。OUTREP=オプションを指定することで、異なるデータ表現を使用して SAS データセットを作成できます。たとえば、UNIX 環境で、Windows データ表現を使用する SAS データセットを作成できます。互換性およびデータ表記の詳細については、“Processing Data Using Cross-Environment Data Access (CEDA)” (*SAS Language Reference: Concepts*) を参照してください。

OUTREP=の値については、次の表を参照してください。

**表 2.1** OUTREP=オプションのデータ表現の値

OUTREP=の値	エイリアス*	環境
ALPHA_TRU64	ALPHA_OSF	Tru64 UNIX
ALPHA_VMS_32	ALPHA_VMS	OpenVMS Alpha
ALPHA_VMS_64		OpenVMS Alpha
HP_IA64	HP_ITANIUM	HP-UX (Itanium プロセッサファミリアーキテクチャ)
HP_UX_32	HP_UX	HP-UX (PA-RISC)
HP_UX_64		HP-UX (PA-RISC)、64 ビット
INTEL_ABI		ABI (Intel アーキテクチャ)
LINUX_32	LINUX	Linux (Intel アーキテクチャ)
LINUX_IA64		Linux (Itanium ベースシステム)
LINUX_X86_64		Linux (x64)

OUTREP=の値	エイリアス*	環境
MIPS_ABI		MIPS ABI
MVS_32	MVS	31 ビット SAS (z/OS)
MVS_64_BFP		64 ビット SAS (z/OS)
OS2		OS/2 (Intel)
RS_6000_AIX_32	RS_6000_AIX	AIX
RS_6000_AIX_64		AIX
SOLARIS_32	SOLARIS	Solaris (SPARC)
SOLARIS_64		Solaris (SPARC)
SOLARIS_X86_64		Solaris (x64)
VAX_VMS		OpenVMS VAX
VMS_IA64		OpenVMS (HP Integrity)
WINDOWS_32	WINDOWS	32 ビット SAS (Microsoft Windows)
WINDOWS_64		64 ビット SAS (Itanium ベースシステムと x64 両方の Microsoft Windows)

\* 現在の値を使用することをお勧めします。エイリアスは互換性のためにだけ使用します。

## 詳細

### 注意:

エンコーディングに互換性がない場合、トランスコーディングで文字データが失われる可能性があります。エンコーディングおよび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*を参照してください。

### 関連項目:

- “OUTREP=format” (*SAS Statements: Reference*)
- “Processing Data Using Cross-Environment Data Access (CEDA)” (*SAS Language Reference: Concepts*)

---

## POINTOBS=データセットオプション

SAS でオブザベーションへのランダムアクセスまたは順次アクセスが可能な圧縮データセットを作成するかどうかを指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** オブザベーションコントロール

**制限事項:** POINTOBS=は、圧縮データセットを作成する場合にのみ適用されます。それ以外の場合は無視されます。

## 構文

POINTOBS=YES | NO

### 構文の説明

#### YES

オブザベーション番号でランダムにアクセスできる圧縮データセットを SAS ソフトウェアで作成します。これがデフォルト設定です。

オブザベーション番号でデータに直接アクセスする例を次に示します。

- DATA ステップの MODIFY ステートメントおよび SET ステートメントの POINT=オプションを介して
- PROC FSEDIT を使用して特定のオブザベーション番号を介して

**ヒント** POINTOBS=YES と指定した場合、データセットからの情報の取得の効率性には影響しません。圧縮データセットの作成時およびその圧縮データセットの情報の更新または追加時に、CPU 使用率が約 10%増加します。

#### NO

圧縮されたデータセットのオブザベーションをオブザベーション番号でランダムにアクセスする機能を抑制します。

**ヒント** 次の状況では、圧縮データセットで、オブザベーション番号でデータにアクセスする必要がない場合、POINTOBS=NO と指定するとパフォーマンスは約 10%向上します。

- 圧縮データセットの作成時
- 圧縮データセットへのオブザベーションの更新または追加時

## 詳細

REUSE=YES は、POINTOBS=YES よりも優先されます。例:

```
data test (compress=yes pointobs=yes reuse=yes);
```

このデータセットオプションでは、データセットに POINTOBS=NO が指定されます。圧縮を使用する場合のデフォルトは POINTOBS=YES であるため、REUSE=YES によって POINTOBS=が NO に変更されます。

## 関連項目:

### データセットオプション:

- “COMPRESS=データセットオプション” (13 ページ)
- “REUSE=データセットオプション” (62 ページ)

### システムオプション:

- “COMPRESS= System Option” (*SAS System Options: Reference*)

- “REUSE= System Option” (*SAS System Options: Reference*)

---

## PW=データセットオプション

SAS ファイルに READ、WRITE および ALTER パスワードを割り当て、パスワード保護された SAS ファイルへのアクセスを可能にします。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** データセットコントロール

**注:** パスワードのセキュリティを確認するためにこの操作の後にログを確認してください。詳細については、“Blotting Passwords and Encryption Key Values” (*SAS Language Reference: Concepts*)を参照してください。

---

### 構文

PW=*password*

#### 構文の説明

*password*

パスワードを 8 文字に制限し、大文字と小文字を区別しない有効な SAS 名を指定する必要があります。“Words in the SAS Language” (*SAS Language Reference: Concepts*)を参照してください。

### 詳細

PW= オプションはカタログ以外の全てのタイプの SAS ファイルに適用されます。このオプションは、パスワードを SAS ファイルに割り当てる場合や、パスワード保護された SAS ファイルにアクセスする場合に使用できます。

ALTER パスワードで保護された SAS データセットを置き換えると、新しいデータセットに ALTER パスワードが継承されます。新しいデータセットの ALTER パスワードを変更するには、DATASETS プロシジャで MODIFY ステートメントを使用します。

#### 動作環境の情報

パスワードの使用の詳細については、SAS のドキュメントの該当セクションを参照してください。

**注:** SAS パスワードでは、SAS System 以外での SAS ファイルへのアクセスはコントロールされません。SAS 以外で SAS ファイルへのアクセスをコントロールするには、オペレーティングシステムで提供されるユーティリティか、ファイルシステムのセキュリティコントロールを使用してください。

### 関連項目:

- “File Protection” (*SAS Language Reference: Concepts*)
- “Manipulating Passwords” (*Base SAS Procedures Guide*)

#### データセットオプション:

- “ALTER=データセットオプション” (8 ページ)
- “ENCRYPT=データセットオプション” (18 ページ)
- “READ=データセットオプション” (55 ページ)



- “WRITE=データセットオプション” (75 ページ)

---

## PWREQ=データセットオプション

パスワードダイアログボックスを表示するかどうか指定します。

**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

PWREQ=YES | NO

### 構文の説明

#### YES

ダイアログボックスを表示します。

#### NO

ダイアログボックスを表示しません。存在しないパスワードまたは無効なパスワードが入力された場合、データセットは開かずにエラーメッセージが SAS ログに書き込まれます。

### 詳細

対話型 SAS セッションでの PWREQ=オプションは、パスワード保護された SAS データセットに無効なパスワードまたは存在しないパスワードが入力された場合にダイアログボックスを表示するかどうかを制御します。PWREQ=は、READ=、WRITE=または ALTER=パスワードのあるデータセットに適用されます。PWREQ=は SCL アプリケーションで最も役に立ちます。

### 関連項目:

#### データセットオプション:

- “ALTER=データセットオプション” (8 ページ)
- “ENCRYPT=データセットオプション” (18 ページ)
- “PW=データセットオプション” (54 ページ)
- “READ=データセットオプション” (55 ページ)
- “WRITE=データセットオプション” (75 ページ)

---

## READ=データセットオプション

READ=パスワードを SAS ファイルに割り当てます。これにより、ユーザーはパスワードを入力しない限りファイルを読み取れなくなります。

**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

**注:** パスワードのセキュリティを確認するためにこの操作の後にログを確認してください。詳細については、“Blotting Passwords and Encryption Key Values” (*SAS Language Reference: Concepts*)を参照してください。

## 構文

READ=*read-password*

### 構文の説明

#### *read-password*

有効な SAS 名である必要があります。詳細については、“Words in the SAS Language” (*SAS Language Reference: Concepts*)を参照してください。

## 詳細

READ=オプションは、カタログ以外のすべての種類の SAS ファイルに適用されます。このオプションは、パスワードを SAS ファイルに割り当てる場合や、読み取り保護された SAS ファイルにアクセスする場合に使用できます。

**注:** SAS パスワードでは、SAS System 以外での SAS ファイルへのアクセスはコントロールされません。SAS 以外で SAS ファイルへのアクセスをコントロールするには、オペレーティングシステムで提供されるユーティリティか、ファイルシステムのセキュリティコントロールを使用してください。

## 関連項目:

- “File Protection” (*SAS Language Reference: Concepts*)
- “Manipulating Passwords” (*Base SAS Procedures Guide*)

### データセットオプション:

- “ALTER=データセットオプション” (8 ページ)
- “ENCRYPT=データセットオプション” (18 ページ)
- “PW=データセットオプション” (54 ページ)
- “WRITE=データセットオプション” (75 ページ)

## RENAME=データセットオプション

変数の名前を変更します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** 変数コントロール

## 構文

RENAME=(*old-name-1=new-name-1 <old-name-2=new-name-2 ...>*)

## 構文の説明

### *old-name*

名前を変更する変数です。

### *new-name*

変数の新しい名前です。有効な SAS 名を指定する必要があります。

## 詳細

データセットを作成するときに RENAME=データセットオプションを使用する場合は、出力データセットに新しい変数名を含める必要があります。入力データセットに対して RENAME=を使用すると、DATA ステップのプログラミングステートメントで新しい名前が使用されます。

SAS プロシジャで使用される入力データセットに対して RENAME=を使用すると、そのプロシジャ内の変数名が変更されます。WHERE ステートメントや WHERE=データセットオプションなどの WHERE 処理と一緒に RENAME=を使用すると、新しい名前はデータの処理前に適用されます。WHERE 式では新しい名前を使用する必要があります。

DROP=データセットオプションでも KEEP=データセットオプションでも、同じ DATA ステップで RENAME=を使用します。DROP=および KEEP=データセットオプションは RENAME=の前に適用されます。DROP=および KEEP=データセットオプションでは、古い名前を使用する必要があります。同じステートメント内で、同じ変数を削除したり名前変更したりすることはできません。

注: RENAME=データセットオプションは、出力モードで開くデータセットにのみ適用されます。

プログラムロジックで変数名を変更する必要がある場合は、RENAME ステートメントまたは RENAME=データセットオプションを使用します。たとえば、2 つの入力データセットに同じ名前の変数が存在する場合があります。ファイル管理タスクとして変数名を変更するには、DATASETS プロシジャを使用します。

処理が開始される前に変数名を変更するには、入力データセットに対して RENAME=データセットオプションを使用する必要があります。

## 比較

RENAME=データセットオプションと RENAME ステートメントには次の違いがあります。

- PROC ステップでは、RENAME=データセットオプションのみを使用でき、RENAME ステートメントは使用できません。
- それぞれのデータセットの個々の変数名を変更する場合は、RENAME=データセットオプションを使用する必要があります。RENAME ステートメントはすべての出力データセットに適用されます。

## 例

### 例 1: 出力時の変数名の変更

この例では、DATA ステートメントで RENAME=を使用し、出力データセットへの書き込み時に変数名が変更されることを示します。DATA ステップ処理時には、変数の元の名前 X が使用されます。

```
data one;
  input x y z;
  datalines;
```

```

24 595 439
243 343 034
;
proc print data=one;
run;

data two(rename=(x=keys));
  set one;
  z=x+y;
run;
proc print data=two;
run;

```

**アウトプット 2.10** データセット One および Two

**The SAS System**

Obs	x	y	z
1	24	595	439
2	243	343	34

**The SAS System**

Obs	keys	y	z
1	24	595	619
2	243	343	586

### 例 2: 入力時の変数名の変更

この例では、DATA ステップ処理前に、SET ステートメントで変数 X を KEYS という変数名に変更します。

```

data three;
  set one(rename=(x=keys));
  z=keys+y;
run;

proc print data=three;
run;

```

## アウトプット 2.11 データセット Three

**The SAS System**

Obs	keys	y	z
1	24	595	619
2	243	343	586

**例 3: WHERE 処理で SAS プロシジャの変数名を変更する**

この例では、PRINT プロシジャの変数 score1 を score2 という変数名に変更します。新しい名前はデータの処理前に適用されるため、WHERE ステートメントではこの新しい名前を指定する必要があります。

```
data test;
  input score1;
  datalines;
26
76
86
56
;

proc print data=test (rename=(score1=score2));
  where score2 gt 75;
run;
```

## アウトプット 2.12 データセット Test

**The SAS System**

Obs	score2
2	76
3	86

**関連項目:****データセットオプション:**

- “DROP=データセットオプション” (17 ページ)
- “KEEP=データセットオプション” (37 ページ)

**ステートメント:**

- “RENAME Statement” (*SAS Statements: Reference*)

**プロシジャ:**

- “DATASETS” (*Base SAS Procedures Guide*)

---

## REPEMPTY=データセットオプション

同じ名前の新しい空のデータセットで、既存の SAS データセットを上書きできるかどうかを指定します。

- 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  
**制限事項:** 出力データセットにのみ使用します。
- 

### 構文

REPEMPTY=YES | NO

#### 構文の説明

##### YES

同じ名前の新しい空のデータセットで、既存のデータセットを置き換えるように指定します。これがデフォルト設定です。

**操作** REPEMPTY=YES に設定され、REPLACE=NO に設定されている場合、データセットは置き換えられません。

---

##### NO

同じ名前の新しい空のデータセットで、既存の SAS データセットを置き換えないように指定します。

**ヒント** データを含む新しいデータセットで既存のデータセットを置き換えるには、REPLACE=YES および REPEMPTY=NO を設定します。

---

次の構文エラーで誤って作成された新しい空のデータセット B で既存のデータセット B が置き換えられないようにする場合、REPEMPTY=NO を使用します。

```
data mylib.a set b;
```

---

誤って作成された新しい空のデータセットで既存のデータセットが上書きされないようにするには、REPLACE=YES および REPEMPTY=NO を設定します。

---

### 詳細

個々のデータセットの場合、REPEMPTY=データセットオプションは LIBNAME ステートメントの REPEMPTY=オプションよりも優先されます。

### 比較

- REPEMPTY=および REPLACE=データセットオプションは、永続的な SAS データセットと一時的な SAS データセットに適用されます。ただし、REPLACE システムオプションは永続的な SAS データセットのみに適用されます。

**関連項目:****データセットオプション:**

- “REPLACE=データセットオプション” (61 ページ)

**ステートメントオプション:**

- “REPEMPTY=YES|NO” (*SAS Statements: Reference*)

**システムオプション:**

- “REPLACE System Option” (*SAS System Options: Reference*)

---

**REPLACE=データセットオプション**

データを含む新しい SAS データセットで、同じ名前の既存のデータセットを上書きできるかどうかを指定します。

**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

**制限事項:** 出力データセットにのみ使用します。  
このオプションは、SAS データセットの作成時にのみ有効です。

**注:** PROC CONTENTS または PROC DATASETS 内で CONTENTS ステートメントと組み合わせると OUT2= *PermanentLibrary.\_ALL\_* オプションを使用する場合、REPLACE=YES データセットオプションまたは REPLACE システムオプションも設定する必要があります。

---

**構文**

REPLACE=NO | YES

**構文の説明****NO**

同じ名前の新しいデータセットで、既存のデータセットを置き換えないように指定します。

**YES**

同じ名前の新しいデータセットで、既存のデータセットを置き換えるように指定します。

**詳細**

- REPLACE=データセットオプションは、個々のデータセットの REPLACE システムオプションよりも優先されます。
- REPLACE システムオプションは、永続的な SAS データセットにのみ適用されます。

データを含む新しいデータセットで既存のデータセットを置き換えるには、REPLACE=YES および REPEMPTY=NO を設定します。

## 例

この DATA ステートメントで REPLACE=データセットオプションを使用すると、MYLIB によって参照されるライブラリ内の ONE という名前の永続的な SAS データセットは置き換えられなくなります。

```
data mylib.one(replace=no);
```

ファイルが置き換えられていないことを示すメッセージがログに書き込まれます。

## 関連項目:

### システムオプション:

- “REPLACE System Option” (*SAS System Options: Reference*)

---

## REUSE=データセットオプション

圧縮 SAS データセットの空き領域に新しいオブザベーションを書き込みできるかどうかを指定します。

- 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  
**制限事項:** 出力データセットにのみ使用します。
- 

## 構文

REUSE=NO | YES

### 構文の説明

#### NO

圧縮データセット内の領域を追跡および再利用しません。新しいオブザベーションは、既存のデータセットの最後に追加されます。SAS データセット内の多数のオブザベーションを削除したり更新したりする場合には、NO 引数を指定するとデータの格納効率が下がります。

圧縮データセットには、SAS データセットの末尾にオブザベーションを追加するプロシジャ(APPEND プロシジャや FSEDIT プロシジャなど)を使用します。

#### YES

圧縮 SAS データセット内の領域を追跡して再利用します。新しいオブザベーションは、他のオブザベーションの更新または削除によって空いた領域に挿入されます。

REUSE=YES を指定すると、新しいオブザベーションは必ずしも末尾ではない、ファイル内の領域に追加されます。

## 詳細

デフォルトでは、新しいオブザベーションは既存の圧縮データセットの最後に追加されます。他のオブザベーションの削除または更新によって空いた領域を追跡し、再利用する場合は、圧縮 SAS データセットを作成するときに REUSE=データセットオプションを使用します。

REUSE=は、COMPRESS=YES データセットオプションまたはシステムオプションを使用してデータセットを新規作成するときのみ使用します。



REUSE=データセットオプションは、REUSE=システムオプションよりも優先されます。

REUSE=YES は、POINTOBS=YES よりも優先されます。たとえば、次のステートメントを使用すると、データセットの設定は POINTOBS=NO になります。

```
data test (compress=yes pointobs=yes reuse=yes);
```

圧縮を使用する場合のデフォルトは POINTOBS=YES であるため、REUSE=YES によって POINTOBS=が NO に変更されます。

## 関連項目:

### データセットオプション:

- “COMPRESS=データセットオプション” (13 ページ)

### システムオプション:

- “REUSE= System Option” (*SAS System Options: Reference*)

---

## ROLE=データセットオプション

スタースキーマ結合のファクトテーブルを指定します。

<b>該当要素:</b>	PROC SQL
<b>カテゴリ:</b>	データセットコントロール
<b>制限事項:</b>	入力データセットにのみ使用します。

---

## 構文

**ROLE=**[FACT](#) | [DIMENSION](#) | [DIM](#)

### 構文の説明

#### FACT

SAS データセットをスタースキーマのファクトテーブルとして指定します。

#### DIMENSION | DIM

SAS データセットをスタースキーマのディメンションテーブルとして指定します。

## 詳細

スタースキーマは複数のテーブルの配列で、大きなファクトテーブルが複数のディメンションテーブルに結合されています。たとえば、スタースキーマを作成するために、SQL プロシジャの構文を使用して SAS データセットを結合できます。

結合テーブルを処理するアプリケーションのパフォーマンスを向上するには、ROLE=データセットオプションを指定します。たとえば、特定のファクトテーブルを指定するには、ROLE=FACT を指定します。また、ROLE=DIMENSION を指定して、個々のディメンションテーブルを指定することもできます。

テーブルの役割はクエリごとに変わる可能性があるため、ROLE=の指定は現在のステップにのみ有効で、データセットには保存されません。

## 例: ファクトテーブルの指定

次の例では、ROLE=データセットオプションの使用により、PROC SQL のパフォーマンスが向上します。ORDERS がファクトテーブルで、PRODUCT、PERIOD、CUSTOMER がディメンションテーブルです。

```
proc sql;
  select orders.Order_Total
  from orders (role=fact), product, period, customer
  where orders.Product_ID = product.Product_ID
        and orders.Period_ID = period.Period_ID
        and orders.Customer_ID = customer.Customer_ID
        and product.Product_Name = "camera"
        and period.Period_Name = "1997"
        and customer.Customer_Name = "Walmart";
quit;
```

### 関連項目:

“SQL” (*SAS SQL Procedure User's Guide*)

---

## SORTEDBY=データセットオプション

データセットの現在の並べ替え方法を示します。

**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

**SORTEDBY=***by-clause**</ collate-name>* | **\_NULL\_**

#### 構文の説明

***by-clause*** *</ collate-name>*

データの現在の並べ替え方法を示します。

***by-clause***

PROC SORT ステップの BY ステートメントで使用する変数とオプションを示します。

***collate-name***

並べ替えに使用する照合順序を示します。デフォルトでは、動作環境の照合順序になります。照合順序の前にスラッシュ(/)を付ける必要があります。

**動作環境の情報**

照合順序の詳細については、使用している動作環境向けの SAS ドキュメントを参照してください。

***\_NULL\_***

既存のソートインジケータを削除します。

### 詳細

ソートインジケータを確認することにより、データセットがすでにキー変数に基づき昇順に並べ替えられているかどうか判断されます。ソートインジケータは、データセットのディスクリプタ情報に保存され、以前の並べ替えから設定されます。ソートインジケータ

タの使用方法和パフォーマンス改善の詳細については、“The Sort Indicator” (*SAS Language Reference: Concepts*) および“SORTVALIDATE System Option” (*SAS System Options: Reference*)を参照してください。

次の例の CONTENTS プロシジャは、SORTEDBY=データセットオプションを使用して、データセットが並べ替えられたことを示しています。

```
Sort Information Sortedby var1 Validated NO Character Set ANSI
```

## 比較

- DATASETS プロシジャの CONTENTS ステートメントは、データセットの並べ替え方法を示します。
- SORTEDBY=オプションは、データの並べ替え方法を示しますが、データセットの並べ替えは実行しません。

## 例

この例では、SORTEDBY=データセットオプションを使用して、データの現在の並べ替え方法を示します。データセット ORDERS は、PRIORITY に基づき、INDATE 値の降順に並べ替えられています。データセットが作成されると、ソートインジケータと一緒に保存されます。これらのステートメントにより、データセット ORDERS が作成され、ソートインジケータが記録されます。

```
libname mylib 'SAS-library';
options yearcutoff=1926;
data mylib.orders(sortedby=priority
                  descending indate);
  input priority 1. +1 indate date7.
        +1 office $ code $;
  format indate date7.;
  datalines;
1 03may01 CH J8U
1 21mar01 LA M91
1 01dec00 FW L6R
1 27feb99 FW Q2A
2 15jan08 FW I9U
2 09jul99 CH P3Q
3 08apr10 CH H5T
3 31jan12 FW D2W
;
```

### アウトプット 2.13 PROC CONTENTS Sort Information

Sort Information	
Sortedby	priority DESCENDING indate
Validated	NO
Character Set	ANSI

**関連項目:**

- “DATASETS” (*Base SAS Procedures Guide*)
- “SORT” (*Base SAS Procedures Guide*)
- “SQL” (*SAS SQL Procedure User's Guide*)

**SPILL=データセットオプション**

DATA ステップビューの非順次処理の予備ファイルを作成するかどうかを指定します。

**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

**制限事項:** DATA ステップビューでのみ有効です。

**構文**

**SPILL=**YES | NO

**構文の説明****YES**

DATA ステップビューの非順次処理の予備ファイルを作成します。これがデフォルト設定です。

**操作** DATA ステップビューの順次処理に予備ファイルが作成されることはありません。

**NO**

予備ファイルを作成しないか、予備ファイルのサイズを小さくします。

**操作** 直接(ランダム)アクセスの場合は、SPILL=NO を指定しても必ず予備ファイルが作成されます。

**注** 大量のデータを生成する DATA ステップビューの予備ファイルに対応する十分なディスク容量がない場合は、SPILL=NO を指定してください。

**ヒント** BY グループデータを処理する SAS プロシジャの場合は、現在の BY グループのみが予備ファイルに書き込まれるように SPILL=NO を指定することを検討してください。

**詳細**

DATA ステップビューが非順次処理で開かれるときに、デフォルトで予備ファイルが作成されます。spill file には、DATA ステップビューによって生成されるオブザベーションが含まれます。以降のデータ要求では、DATA ステップビューを再び実行するかわりに、予備ファイルからオブザベーションが読み込まれます。予備ファイルは WORK ライブラリにある一時ファイルです。

非順次処理には、一部の SAS ステートメントおよびプロシジャでサポートされる、次のアクセスメソッドが含まれます。SPILL=データセットオプションが各アクセスメソッドでどのように動作するかを次に説明します。

### ランダムアクセス

すべてのオブザベーションを順次に読み込まずにインデックスを使用して、オブザベーション番号が変数の値のいずれかで、オブザベーションを直接取得します。オブザベーションごとに DATA ステップビューを再起動すると処理時間が非常に長くなるため、SPILL=YES と SPILL=NO のどちらであっても、予備ファイルは必ず作成されます。

### BY-グループアクセス

BY ステートメントを使用して、変数の値に基づいて順序付け、グループ化またはインデックス付けされるオブザベーションを処理します。SPILL=YES の場合、DATA ステップビューから要求されるすべてのデータと同じサイズの予備ファイルが作成されます。SPILL=NO の場合、現在の BY グループのみを予備ファイルに書き込みます。予備ファイルのサイズは BY グループのサイズに依存します。

### 2 パスアクセス

データのパススルーを順次に複数回実行します。SPILL=NO の場合、予備ファイルは作成されません。かわりに、データの 1 回目のパススルーの後、後続のパススルーごとに DATA ステップビューが再起動されます。再起動ごとに DATA ステップビューによって少量のデータが返されると、ビューの再起動処理時間がかかなりの長さになる可能性があります。

注: SPILL=NO では、データの後続のパススルーで、異なるデータが生成される場合があります。予備ファイルの使用が必要になる処理もあります。たとえば、ランダム関数を使用した現在時刻に基づく値の計算結果は、データに影響する可能性があります。

## 例

### 例 1: 少数の大きな BY グループに予備ファイルを使用する

この例では、大量のランダムデータを生成し、UNIVARIATE プロシジャと BY ステートメントを使用する、DATA ステップビューを作成します。この例は、少数の大きな BY グループで SPILL=を使用した場合の影響を示します。

SPILL=YES の場合、DATA ステップビューから要求されるすべてのオブザベーションが予備ファイルに書き込まれます。SPILL=NO の場合、現在の BY グループにあるオブザベーションのみが予備ファイルに書き込まれます。この例によって生成される出力メッセージは、SPILL=NO で縮小される予備ファイルのサイズを示します。ただし、BY グループごとに予備ファイルを切り捨てる時間により、DATA ステップビューの全体的な処理時間が増えることがあります。

```
options msglevel=i;
data vw_few_large / view=vw_few_large;
  drop i;
  do byval = 'Group A', 'Group B', 'Group C';
    do i = 1 to 500000;
      r = ranuni(4);
      output;
    end;
  end;
run;
proc univariate data=vw_few_large (spill=yes) noprint;
  var r;
  by byval;
run;
proc univariate data=vw_few_large (spill=no) noprint;
  var r;
  by byval;
```

```
run;
```

### ログ2.1 SAS ログの出力

```
1  options msglevel=i; 2  data vw_few_large / view=vw_few_large; 3
drop i; 4 5  do byval = 'Group A', 'Group B', 'Group C'; 6  do i =
1 to 500000; 7  r = ranuni(4); 8  output; 9  end;
10  end; 11  run; NOTE:DATA STEP view saved on file
WORK.VW_FEW_LARGE.NOTE:A stored DATA STEP view cannot run under a different
operating system.NOTE:DATA statement used (Total process time): real
time 21.57 seconds cpu time 1.31 seconds 12  proc
univariate data=vw_few_large (spill=yes) noprint; INFO:View WORK.VW_FEW_LARGE
open mode:BY-group rewind.13  var r; 14  by byval; 15  run; INFO:View
WORK.VW_FEW_LARGE opening spill file for output observations.INFO:View
WORK.VW_FEW_LARGE deleting spill file.File size was 22506120 bytes.NOTE:View
WORK.VW_FEW_LARGE.VIEW used (Total process time): real time 40.68
seconds cpu time 12.71 seconds NOTE:PROCEDURE UNIVARIATE used (Total
process time): real time 57.63 seconds cpu time 13.12
seconds 16 17  proc univariate data=vw_few_large (spill=no) noprint; INFO:View
WORK.VW_FEW_LARGE open mode:BY-group rewind.18  var r; 19  by byval;
20  run; INFO:View WORK.VW_FEW_LARGE opening spill file for output
observations.INFO:View WORK.VW_FEW_LARGE truncating spill file.File size was
7502040 bytes.NOTE:The above message was for the following by-group: byval=Group
A INFO:View WORK.VW_FEW_LARGE truncating spill file.File size was 7534800
bytes.NOTE:The above message was for the following by-group: byval=Group B
INFO:View WORK.VW_FEW_LARGE truncating spill file.File size was 7534800
bytes.NOTE:The above message was for the following by-group: byval=Group C
INFO:View WORK.VW_FEW_LARGE deleting spill file.File size was 32760
bytes.NOTE:View WORK.VW_FEW_LARGE.VIEW used (Total process time): real
time 11.03 seconds cpu time 10.95 seconds NOTE:PROCEDURE
UNIVARIATE used (Total process time): real time 11.04 seconds cpu
time 10.96 seconds
```

### 例2: 多数の小さなBYグループに予備ファイルを使用する

この例では、大量のランダムデータを生成し、UNIVARIATE プロシジャとBY ステートメントを使用する、DATA ステップビューを作成します。この例は、多数の小さなBYグループでSPILL=を使用した場合の影響を示します。

SPILL=YES の場合、DATA ステップビューから要求されるすべてのオブザベーションが予備ファイルに書き込まれます。SPILL=NO の場合、現在のBYグループにあるオブザベーションのみが予備ファイルに書き込まれます。この例によって生成される出力メッセージは、SPILL=NO で縮小される予備ファイルのサイズを示します。小さなBYグループは、大量のディスク容量の節約につながります。

```
options msglevel=i;
data vw_many_small / view=vw_many_small;
drop i;
do byval = 1 to 100000;
do i = 1 to 5;
r = ranuni(4);
output;
end;
end;
run;
proc univariate data=vw_many_small (spill=yes) noprint;
var r;
by byval;
run;
proc univariate data=vw_many_small (spill=no) noprint;
var r;
by byval;
```

```
run;
```

## ログ2.2 SAS ログの出力

```
1  options msglevel=i; 2  data vw_many_small / view=vw_many_small; 3
drop i; 4 5  do byval = 1 to 100000; 6  do i = 1 to 5;
7  r = ranuni(4); 8  output; 9  end; 10  end;
11 run; NOTE:DATA STEP view saved on file WORK.VW_MANY_SMALL.NOTE:A stored
DATA STEP view cannot run under a different operating system.NOTE:DATA statement
used (Total process time): real time 0.56 seconds cpu time
0.03 seconds 12  proc univariate data=vw_many_small (spill=yes) noprint;
INFO:View WORK.VW_MANY_SMALL open mode:BY-group rewind.13  var r; 14  by
byval; 15  run; INFO:View WORK.VW_MANY_SMALL opening spill file for output
observations.INFO:View WORK.VW_MANY_SMALL deleting spill file.File size was
8024240 bytes.NOTE:View WORK.VW_MANY_SMALL.VIEW used (Total process time): real
time 30.73 seconds cpu time 29.59 seconds NOTE:PROCEDURE
UNIVARIATE used (Total process time): real time 30.96 seconds cpu
time 29.68 seconds 16 17  proc univariate data=vw_many_small
(spill=no) noprint; INFO:View WORK.VW_MANY_SMALL open mode:BY-group rewind.
18  var r; 19  by byval; 20  run; INFO:View WORK.VW_MANY_SMALL opening
spill file for output observations.INFO:View WORK.VW_MANY_SMALL truncating spill
file.File size was 65504 bytes.NOTE:The above message was for the following by-
group: byval=410 INFO:View WORK.VW_MANY_SMALL truncating spill file.File size
was 65504 bytes.NOTE:The above message was for the following by-group: byval=819
INFO:View WORK.VW_MANY_SMALL truncating spill file.File size was 65504
bytes.NOTE:The above message was for the following by-group:
byval=1229 ..Deleted many INFO and NOTE messages for BY groups .INFO:View
WORK.VW_MANY_SMALL truncating spill file.File size was 65504 bytes.NOTE:The
above message was for the following by-group: byval=99894 INFO:View
WORK.VW_MANY_SMALL deleting spill file.File size was 32752 bytes.NOTE:View
WORK.VW_MANY_SMALL.VIEW used (Total process time): real time 29.43
seconds cpu time 28.81 seconds NOTE:PROCEDURE UNIVARIATE used (Total
process time): real time 29.43 seconds cpu time 28.81
seconds
```

### 例3: 2 パスアクセスで予備ファイルを使用する

この例では、大量のランダムデータを生成し、TRANSPOSE プロシジャを使用する、DATA ステップビューを作成します。この例は、2 パスアクセス処理が必要なプロシジャで SPILL=を使用した場合の影響を示します。

PROC TRANSPOSE が DATA ステップビューを処理するとき、プロシジャはビューが生成するオブザベーションを 2 回パススルーする必要があります。1 回目のパスでオブザベーション数を数え、2 回目のパスで転置が実行されます。SPILL=YES の場合、1 回目のパス時に予備ファイルが作成され、2 回目のパスで予備ファイルからオブザベーションが読み込まれます。SPILL=NO の場合、予備ファイルは作成されません。1 回目のパスの後に DATA ステップビューが再起動されます。

最初の TRANSPOSE プロシジャでは、デフォルトで予備ファイルが使用される場合でも、SPILL=データセットオプションを含みません。Open モードに関する SAS ログメッセージは表示されません。

```
options msglevel=i;
data vw_transpose/view=vw_transpose;
  drop i j;
  array x[10000];
  do i = 1 to 10;
    do j = 1 to dim(x);
      x[j] = ranuni(4);
    end;
  output;
end;
```

```

run;
proc transpose data=vw_transpose out=transposed;
run;
proc transpose data=vw_transpose(spill=yes) out=transposed;
run;
proc transpose data=vw_transpose(spill=no) out=transposed;
run;

```

### ログ2.3 SAS ログの出力

```

1  options msglevel=i; 2  data vw_transpose/view=vw_transpose; 3  drop
i j; 4  array x[10000]; 5  do i = 1 to 10; 6  do j = 1 to
dim(x); 7  x[j] = ranuni(4); 8  end; 9  output;
10 end; 11 run; NOTE:DATA STEP view saved on file
WORK.VW_TRANSPOSE.NOTE:A stored DATA STEP view cannot run under a different
operating system.NOTE:DATA statement used (Total process time): real
time 0.68 seconds cpu time 0.18 seconds 12 proc transpose
data=vw_transpose out=transposed; 13 run; INFO:View WORK.VW_TRANSPOSE opening
spill file for output observations.INFO:View WORK.VW_TRANSPOSE deleting spill
file.File size was 880000 bytes.NOTE:View WORK.VW_TRANSPOSE.VIEW used (Total
process time): real time 2.37 seconds cpu time 1.17 seconds
NOTE:There were 10 observations read from the data set
WORK.VW_TRANSPOSE.NOTE:The data set WORK.TRANSPOSED has 10000 observations and
11 variables.NOTE:PROCEDURE TRANSPOSE used (Total process time): real
time 4.17 seconds cpu time 1.51 seconds 14 proc transpose
data=vw_transpose (spill=yes) out=transposed; INFO:View WORK.VW_TRANSPOSE open
mode: sequential.15 run; INFO:View WORK.VW_TRANSPOSE reopen mode: two-
pass.INFO:View WORK.VW_TRANSPOSE opening spill file for output
observations.INFO:View WORK.VW_TRANSPOSE deleting spill file.File size was
880000 bytes.NOTE:View WORK.VW_TRANSPOSE.VIEW used (Total process time): real
time 0.95 seconds cpu time 0.92 seconds NOTE:There were 10
observations read from the data set WORK.VW_TRANSPOSE.NOTE:The data set
WORK.TRANSPOSED has 10000 observations and 11 variables.NOTE:PROCEDURE TRANSPOSE
used (Total process time): real time 1.01 seconds cpu time
0.98 seconds 16 proc transpose data=vw_transpose (spill=no) out=transposed;
INFO:View WORK.VW_TRANSPOSE open mode: sequential.17 run; INFO:View
WORK.VW_TRANSPOSE reopen mode: two-pass.INFO:View WORK.VW_TRANSPOSE restarting
for another pass through the data.NOTE:View WORK.VW_TRANSPOSE.VIEW used (Total
process time): real time 1.34 seconds cpu time 1.32 seconds
NOTE:The View WORK.VW_TRANSPOSE was restarted 1 times.The following view
statistics only apply to the last view restart.NOTE:There were 10 observations
read from the data set WORK.VW_TRANSPOSE.NOTE:The data set WORK.TRANSPOSED has
10000 observations and 11 variables.NOTE:PROCEDURE TRANSPOSE used (Total process
time): real time 1.42 seconds cpu time 1.40 seconds

```

### 関連項目:

#### データセットオプション:

- [“OBSBUF=データセットオプション” \(40 ページ\)](#)

---

## TOBSNO=データセットオプション

クライアント/サーバー間の転送で送信するオブザベーション数を指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** データセットコントロール



**制限事項:** TOBSNO=オプションは、REMOTE エンジンを経由して SAS Sever からアクセスするデータセットでのみ有効です。

## 構文

TOBSNO=*n*

### 構文の説明

*n*  
送信するオブザベーション数を指定します。

## 詳細

TOBSNO=オプションが指定されていない場合、オブザベーションの長さ、サーバーの送信バッファサイズに基づいて値が計算されます。このアクションは PROC SERVER ステートメントの TBUFSIZE=オプションで指定されます。

TOBSNO=オプションは、REMOTE エンジンを経由して SAS Server からアクセスするデータセットでのみ有効です。更新するために開かれるデータセット、または別のエンジン経由でアクセスしたデータセットにこのオプションが指定されている場合は無視されます。

## 関連項目:

“FOPEN Function” (*SAS Functions and CALL Routines: Reference*)

## TYPE=データセットオプション

特殊な構造の SAS データセットにデータセットの種類を指定します。

**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

## 構文

TYPE=*data-set-type*

### 構文の説明

*data-set-type*  
特殊なデータセットの種類を指定します。

## 詳細

次のタスクの実行時に、DATA ステップで TYPE=データセットオプションを使用します。

- 適切な出力形式で特殊な SAS データセットを作成する
- プロシジャステートメントで特殊な種類の SAS データセットを識別する

データセットの種類を特定するには、CONTENTS プロシジャを使用できます。

ほとんどの SAS データセットには、種類は指定されていません。ただし、一部の SAS/STAT プロシジャで使用される、特殊な構造の SAS データセットがいくつかありま

す。これらの SAS データセットには、特殊な変数とオブザベーションが含まれており、これらは通常 SAS 統計プロシジャで作成されます。特殊な SAS データセットのほとんどは SAS/STAT で使用されるため、これらの詳細については *SAS/STAT User's Guide* を参照してください。特殊データセットには、CORR、COV、SSPC、EST、FACTOR などがあります。

他の SAS ソフトウェア製品で使用できる追加の値があります。詳細については、適切な製品のドキュメントを参照してください。

注: DATA ステップと SET ステートメントを一緒に使用して特殊な SAS データセットを変更する場合は、DATA ステートメントで TYPE=オプションを指定する必要があります。data-set-type は作成するデータセットには自動的にコピーされません。

## 関連項目:

“CONTENTS” (*Base SAS Procedures Guide*)

---

## WHERE=データセットオプション

SAS データセットからオブザベーションを選択するために使用する特定の条件を指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** オブザベーションコントロール

**制限事項:** SET および MODIFY ステートメント内で POINT=オプションと一緒に使用できません。

## 構文

WHERE=(*where-expression-1* <*logical-operator* *where-expression-2*> )

### 構文の説明

#### *where-expression*

一連の演算子、オペランドおよび SAS 関数で構成される演算式または論理式です。オペランドは変数、SAS 関数または定数です。演算子は、比較、論理演算または算術計算を要求する記号です。式はかっこで囲む必要があります。

#### *logical-operator*

AND、AND NOT、OR、OR NOT を指定できます。

## 詳細

- WHERE 式で指定した条件を満たすオブザベーションを選択するには、入力データセットと一緒に WHERE=データセットオプションを使用します。オブザベーションは、処理のために DATA ステップまたは PROC ステップに渡されます。WHERE 式の条件を満たすオブザベーションの選択は、DATA ステップの各反復で最初に行われる操作です。

また、出力データセットに書き込まれるオブザベーションを選択することもできます。一般的には、入力時点でオブザベーションを選択するほうが、出力時点で選択するより効率的です。ただし、入力時点でのオブザベーションの選択が実用的でない場合や不可能な場合があります。

- WHERE 処理には、OBS=処理および FIRSTOBS=処理を適用できます。詳細については、“Processing a Segment of Data That Is Conditionally Selected” (*SAS Language Reference: Concepts*)を参照してください。

- SET および MODIFY ステートメント内で、WHERE=データセットオプションと POINT=オプションを一緒に使用することはできません。
- 同じ DATA ステップで WHERE=データセットオプションと WHERE ステートメントの両方を使用できます。WHERE データセットオプションを使用するデータセットの WHERE ステートメントは無視されます。ただし、SAS/FSP では、WHERE=データセットオプションと WHERE コマンドを一緒に使用できます。

注: インデックス付き SAS データセットを使用すると、WHERE 式を使用して SAS データセット内のオブザベーションのサブセットにアクセスするときに、パフォーマンスを大幅に向上できます。インデックス付きデータセットでの WHERE 式の処理の詳細、および SAS データセットにインデックスを付ける前に検討するガイドラインの一覧については、“Understanding SAS Indexes” (*SAS Language Reference: Concepts*) を参照してください。

## 比較

- WHERE ステートメントはすべての入力データセットに適用されますが、WHERE=データセットオプションは指定されたデータセットのみからオブザベーションを選択します。
- DROP=および KEEP=データセットオプションでは処理対象の変数が選択されますが、WHERE=データセットオプションではオブザベーションが選択されます。

## 例

### 例 1: 入力データセットからオブザベーションを選択する

この例では、WHERE=データセットオプションを使用して、別のデータセットに読み込まれるときに SALES データセットをサブセット化します。

```
data whizmo;
  set sales(where=(product='whizmo'));
run;
```

アウトプット 2.14 入力データセットオブザベーション

Obs	product	sales	store
1	whizmo	234	mountain
2	whizmo	273	lakeside
3	whizmo	234	parkview
4	whizmo	233	central

### 例 2: 出力データセットからオブザベーションを選択する

この例では、WHERE=データセットオプションを使用して、SALES 出力データセットをサブセット化します。

```
data whizmo(where=(product='whizmo'));
  set sales;
```

```
run;
```

アウトプット 2.15 出力データセットオブザベーション

Whizmo Data Set			
Obs	product	sales	store
1	whizmo	234	mountain
2	whizmo	273	lakeside
3	whizmo	234	parkview
4	whizmo	233	central

### 関連項目:

- “WHERE Statement” (*SAS Statements: Reference*)
- WHERE 式の処理

---

## WHEREUP=データセットオプション

新しいオブザベーションと変更されたオブザベーションを WHERE 式に対して評価するかどうかを指定します。

**該当要素:** DATA ステップおよび PROC ステップ

**カテゴリ:** オブザベーションコントロール

---

### 構文

WHEREUP=NO | YES

#### 構文の説明

##### NO

追加されたオブザベーションと変更されたオブザベーションを WHERE 式に対して評価しません。

##### YES

追加されたオブザベーションと変更されたオブザベーションを WHERE 式に対して評価します。

### 詳細

追加されたオブザベーションまたは変更されたオブザベーションを指定した WHERE 式と照合する場合、WHEREUP=YES を指定します。

## 例

### 例 1: WHERE 式に一致しない更新の受け入れ

この例では、変更されたオブザベーションが WHERE 式に一致しない場合でもオブザベーションの更新および追加を WHEREUP=で許可する方法を示します。

```
data a;
  x=1;
  output;
  x=2;
  output;
run;
data a;
  modify a(where=(x=1) whereup=no);
  x=3;
  replace; /* Update does not match WHERE expression */
  output; /* Add does not match WHERE expression */
run;
```

この例では、SAS によってオブザベーションが更新され、新しいオブザベーションがデータセットに追加されます。

### 例 2: WHERE 式に一致しない更新の拒否

この例では、更新および追加が WHERE 式に一致しない場合、WHEREUP=はオブザベーションの更新または追加を許可しません。

```
data a;
  x=1;
  output;
  x=2;
  output;
run;
data a;
  modify a(where=(x=1) whereup=yes);
  x=3;
  replace; /* Update does not match WHERE expression */
  output; /* Add does not match WHERE expression */
run;
```

この例では、SAS によってオブザベーションは更新されず、新しいオブザベーションはデータセットに追加されません。

## 関連項目:

### データセットオプション:

- [“WHERE=データセットオプション” \(72 ページ\)](#)

---

## WRITE=データセットオプション

WRITE パスワードを SAS ファイルに割り当てます。これにより、ユーザーはパスワードを入力しない限りファイルを書き込めなくなります。

**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

注: パスワードのセキュリティを確認するためにこの操作の後にログを確認してください。詳細については、“Blotting Passwords and Encryption Key Values” (*SAS Language Reference: Concepts*)を参照してください。

---

## 構文

WRITE=*write-password*

### 構文の説明

*write-password*

有効な SAS 名である必要があります。詳細については、“Words in the SAS Language” (*SAS Language Reference: Concepts*)を参照してください。

## 詳細

WRITE=オプションは、カタログ以外のすべての種類の SAS ファイルに適用されます。このオプションは、パスワードを SAS ファイルに割り当てる場合や、書き込み保護された SAS ファイルにアクセスする場合に使用できます。

注: SAS パスワードでは、SAS System 以外での SAS ファイルへのアクセスはコントロールされません。SAS 以外で SAS ファイルへのアクセスをコントロールするには、オペレーティングシステムで提供されるユーティリティか、ファイルシステムのセキュリティコントロールを使用する必要があります。

## 関連項目:

- “File Protection” (*SAS Language Reference: Concepts*)
- “Manipulating Passwords” (*Base SAS Procedures Guide*)

## データセットオプション:

- “ALTER=データセットオプション” (8 ページ)
- “ENCRYPT=データセットオプション” (18 ページ)
- “PW=データセットオプション” (54 ページ)
- “READ=データセットオプション” (55 ページ)

# 推奨資料

---

このタイトルに関連した推奨される参考資料のリストを次に示します。

- *An Array of Challenges--Test Your SAS Skills*
- *Base SAS Glossary*
- *Base SAS プロシジャガイド*
- *Combining and Modifying SAS Data Sets: Examples*
- *Learning SAS by Example*
- *SAS 出力形式と入力形式: リファレンス*
- *SAS 関数と CALL ルーチン: リファレンス*
- *SAS 言語リファレンス: 解説編*
- *SAS 各国語サポート(NLS): リファレンスガイド*
- *SAS Scalable Performance Data Engine: リファレンス*
- *SAS ステートメント: リファレンス*
- *SAS システムオプション: リファレンス*
- *はじめよう SAS System*
- *The Little SAS Book: A Primer*

SAS 刊行物の一覧については、[sas.com/store/books](https://sas.com/store/books) から入手できます。必要な書籍についての質問は SAS 担当者までお寄せください:

SAS Books  
SAS Campus Drive  
Cary, NC 27513-2414  
電話: 1-800-727-0025  
ファクシミリ: 1-919-677-4444  
メール: [sasbook@sas.com](mailto:sasbook@sas.com)  
Web アドレス: [sas.com/store/books](https://sas.com/store/books)





# キーワード

## 2

2 パスアクセス  
予備ファイル 67

## A

ALTER=データセットオプション 8  
ALTER パスワード  
SAS ファイルへの割り当て 54

## B

BUFNO=データセットオプション 9  
BUFSIZE=データセットオプション 11  
BY-グループアクセス  
予備ファイル 67

## C

CNTLLEV=データセットオプション 12  
COMPRESS=データセットオプション 13

## D

DATA ステップビュー  
非順次処理の予備ファイル 66  
表示バッファのサイズ 40  
DLDMGACTION=データセットオプション 16  
DROP=データセットオプション 17

## E

ENCRYPT=データセットオプション 18  
ENCRYPTKEY データセットオプション 23  
EOC=  
EXTENDOBSCOUNTER=データセットオプション 26  
EXTENDOBSCOUNTER=データセット 26

## F

FILECLOSE=データセットオプション 27  
FIRSTOBS=データセットオプション 27

## G

GENMAX=データセットオプション 29  
GENNUM=データセットオプション 30

## I

IDXNAME=データセットオプション 32  
IDXWHERE=データセットオプション 33  
IN=データセットオプション 36  
INDEX=データセットオプション 35

## K

KEEP=データセットオプション 37

## L

LABEL=データセットオプション 38

## O

OBS=データセットオプション 42  
WHERE 処理 44  
オブザベーションが削除されたデータセット 47  
比較 43  
OBSBUF=データセットオプション 40  
OUTREP= データセットオプション 50

## P

POINTOBS= データセットオプション 52  
PW=データセットオプション 54  
PWREQ=データセットオプション 55

## R

RDC (Ross Data Compression) 13  
READ=データセットオプション 55

READ パスワード

SAS ファイルへの割り当て 54, 55

RENAME=データセットオプション 56

REPEMPTY=データセットオプション 60

REPLACE=データセットオプション 61

REUSE=データセットオプション 62

RLE (Run Length Encoding) 13

ROLE=データセットオプション 63

Ross Data Compression (RDC) 13

Run Length Encoding (RLE) 13

## S

SAS ファイル

ALTER パスワード 8

書き込みを禁止する 75

パスワードの割り当て 54

読み込みを禁止する 55

SORTEDBY=データセットオプション 64

SPILL=データセットオプション 66

## T

TOBSNO=データセットオプション 70

TYPE=データセットオプション 71

## W

WHERE=データセットオプション 72

WHEREUP=データセットオプション 74

WHERE 式

インデックス検索 33

インデックスの無効化 33

更新されたデータセットの評価 74

順次検索 33

条件に一致するインデックスを指定する 32

WHERE 処理

OBS=データセットオプション 44

プロシジャの変数名を変更する 59

WRITE=データセットオプション 75

WRITE パスワード

SAS ファイルへの割り当て 54, 75

## あ

アクセスメソッド

非順次処理 66

圧縮データセット

空き領域の再利用 62

暗号化

出力データセット 18

インデックス

インデックス検索の指定 33

候補インデックスの指定 32

出力データセットの定義 35

破損したデータセット 16

無効化 33

永久バッファページ

サイズ 11

オブザベーション

1つのデータセットで最初に処理するオブザベーション 27

空き領域への書き込み 62

カウントの拡張 26

クライアント/サーバー間の転送 70

更新されたデータセットと WHERE 式 74

出力データセットの圧縮 13

処理の終点 42

選択条件 72

データセットの関与 36

オブザベーションカウント

拡張 26

## か

空のデータセット 60

共有アクセスのレベル

データセット 12

クライアント/サーバー間の転送

送信するオブザベーションの数 70

結合

スタースキーマ結合のファクトテーブル 63

## さ

システムオプション

データセットの相互作用 2

出力データセット

暗号化 18

インデックスの定義 35

永久バッファページのサイズ 11

オブザベーションカウントの拡張 26

オブザベーションの圧縮 13

オブザベーションの選択 73

書き込みから変数を除外する 17

書き込む変数の指定 37

データセットオプション 2

データ表現 50

スタースキーマ結合

ファクトテーブル 63

世代

数の変更 29

データセットの指定 30

データセットの要求 29

バージョンの最大数 29

## た

ダイアログボックス

データセットパスワードの入力 55  
 データセット  
   1つのデータセットで最初に処理するオブザベーション 27  
   圧縮 13  
   暗号化 18  
   上書き 60, 61  
   同じ名前 60, 61  
   オブザベーションカウントの拡張 26  
   オブザベーションの選択 72  
   オブザベーションの選択条件 72  
   空 60  
   共有アクセスのレベル 12  
   現在のオブザベーションへの関与 36  
   更新, WHERE 式の評価 74  
   最後に処理するオブザベーション 42  
   修復 16  
   終了時のテープボリューム位置 27  
   処理に割り当てられるバッファ 9  
   世代 29  
   世代, 指定 30  
   置換 61  
   特殊な構造 71  
   並べ替え 64  
   パスワード入力ダイアログボックス 55  
   破損 16  
   変数の削除 17  
   変数の保持 37  
   ラベル 38  
 データセットオプション 1  
   構文 1  
   システムオプションの相互作用 2  
   出力データセット 2  
   入力データセット 2  
   例 1  
 データセットの圧縮 13  
   ランダムアクセスと順次アクセス 52  
 データセットの修復 16  
 データセットの種類  
   特殊な構造のデータセット 71  
 データ表現  
   出力データセット 50  
 テープボリューム  
   データセット終了時の位置 27

**な**  
 並べ替え  
   データセットの並べ替え情報 64  
 並べ替え情報  
   データセット 64  
 入力データセット

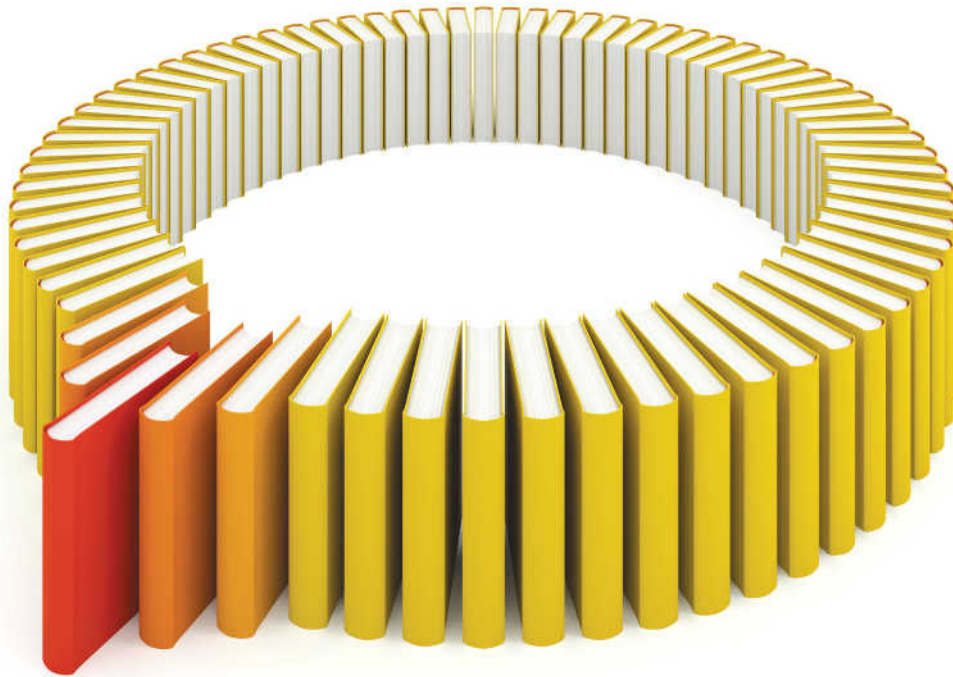
オブザベーションの選択 73  
 処理から変数を除外する 17  
 処理する変数の指定 37  
 データセットオプション 2

**は**  
 パスワード  
   ALTER パスワード 8  
   READ パスワード 55  
   SAS ファイルへの割り当て 54  
   WRITE パスワード 75  
   入力ダイアログボックス 55  
 パスワード保護されたファイル  
   アクセスの有効化 54  
 破損したデータセット 16  
 バッファ  
   永久バッファページのサイズ 11  
   データセット処理に割り当て 9  
   表示バッファのサイズ 40  
   ページサイズ 11  
 非順次処理  
   アクセスメソッド 66  
   予備ファイル 66  
 表示バッファ 40  
   サイズ 40  
 ファクトテーブル  
   スタースキーマ結合 63  
 プロシジャ  
   WHERE 処理, 変数名の変更 59  
 ページサイズ  
   バッファ 11  
 変数  
   データセットの処理から削除する 17  
   データセットの処理時に保持する 37  
   名前の変更 56  
   変数名の変更 56  
   WHERE 処理時のプロシジャ 59  
   出力時 57  
   入力時 58

**や**  
 予備ファイル 66

**ら**  
 ラベル  
   データセット 38  
 ランダムアクセス  
   予備ファイル 67





# Gain Greater Insight into Your SAS<sup>®</sup> Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

 [support.sas.com/bookstore](http://support.sas.com/bookstore)  
for additional books and resources.

  
THE POWER TO KNOW.

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. © 2013 SAS Institute Inc. All rights reserved. S107969US.0613

