Hewlett Packard Enterprise

SQL/MX 3.5 リリースの技術的な最新情報

部品番号: 875227-191
発行: 2017 年 3 月
版数: L17.02 RVU

© 2017, Hewlett Packard Enterprise Development LP

ご注意

本書の内容は、将来予告なしに変更されることがあります。Hewlett Packard Enterprise 製品およびサービスに対する保証については、当該製品およびサービスの保証規定書に記載されています。本書のいかなる内容も、新たな保証を追加するものではありません。本書の内容につきましては万全を期しておりますが、本書中の技術的あるいは校正上の誤り、脱落に対して、責任を負いかねますのでご了承ください。

本書で取り扱っているコンピューターソフトウェアは秘密情報であり、その保有、使用、または複製には、Hewlett Packard Enterprise から使用許諾を得る必要があります。FAR 12.211 および 12.212 に従って、商業用コンピューター・ソフトウェア、コンピューター・ソフトウェア資料、および商業用製品の技術情報は、ベンダー標準の商業用ライセンスのもとで米国政府に使用許諾が付与されます。

他社の Web サイトへのリンクは、Hewlett Packard Enterprise の Web サイトの外に移動します。Hewlett Packard Enterprise は、Hewlett Packard Enterprise の Web サイト以外にある情報を管理する権限を持たず、また責任を負いません。

商標

Intel®、インテル、Itanium®、Pentium®、Intel Inside®および Intel Inside ロゴは、インテルコーポレーションまたはその子会社のアメリカ合衆国およびその他の国における商標または登録商標です。

Microsoft®および Windows®は、Microsoft Corporation の商標です。

Java®および Oracle®は、Oracle および/またはその関連会社の登録商標です。

Motif、OSF/1、UNIX、X/Open、および"X"デバイスは、米国およびその他の国における登録商標であり、IT DialTone および The Open Group は、米国およびその他の国における The Open Group の商標です。

保証

Open Software Foundation、OSF、OSF ロゴ、OSF/1、OSF/Motif、および Motif は、Open Software Foundation, Inc.の商標です。

OSF MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE OSF MATERIAL PROVIDED HEREIN, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

OSF shall not be liable for errors contained herein or for incidental consequential damages in connection with the furnishing, performance, or use of this material.

© 1990, 1991, 1992, 1993 Open Software Foundation, Inc. This documentation and the software to which it relates are derived in part from materials supplied by the following:

© 1987, 1988, 1989 Carnegie-Mellon University. © 1989, 1990, 1991 Digital Equipment Corporation. © 1985, 1988, 1989, 1990 Encore Computer Corporation. © 1988 Free Software Foundation, Inc. © 1987, 1988, 1989, 1990, 1991 Hewlett Packard Enterprise. © 1985, 1987, 1988, 1989, 1990, 1991, 1992 International Business Machines Corporation. © 1988, 1989 Massachusetts Institute of Technology. © 1988, 1989, 1990 Mentat Inc. © 1988 Microsoft Corporation. © 1987, 1988, 1989, 1990, 1991, 1992 SecureWare, Inc. © 1990, 1991 Siemens Nixdorf Informationssysteme AG. © 1986, 1989, 1996, 1997 Sun Microsystems, Inc. © 1989, 1990, 1991 Transarc Corporation.

This software and documentation are based in part on the Fourth Berkeley Software Distribution under license from The Regents of the University of California. OSF acknowledges the following individuals and institutions for their role in its development: Kenneth C.R.C. Arnold, Gregory S. Couch, Conrad C. Huang, Ed James, Symmetric Computer Systems, Robert Elz. © 1980, 1981, 1982, 1983, 1985, 1986, 1987, 1988, 1989 Regents of the University of California.

本書に含まれる情報の輸出には、米国商務省の承認が必要な場合があります。

目次

このドキュメントについて

本マニュアルは、SQL/MX 3.5 の新機能およびで既存の機能へのアップデートに関する情報を提供します。

対象となるリリースバージョンアップデート（RVU）

本書では、L17.02 以降の L シリーズ RVU をサポートしています。

対象読者

このマニュアルは、データベース管理者および NonStop SQL/MX 3.5 を使用しているアプリケーションプログラマ向けです。この製品を使用するには、SQL（構造化照会言語）および American National Standard Database Language SQL:1999 について深く理解している必要があります。

発行履歴

部品番号	製品バージョン	発行
875227-191	NonStop SQL/MX 3.5	2017 年 3 月

SQL/MX 3.5 リリースの技術的な最新情報

技術的な最新情報は、SQL/MX リリース 3.5 の新機能について紹介します。

機能

- **データベース互換性** アプリケーションを SQL/MX へより簡単に移行するための機能のセットです。この機能としては、データ型 DATE type2、VARCHAR2、NUMBER、TO_DATE 関数、MERGE 文、マテリアライズドビューなどがあります。
- **PL/MX**、SQL/MX の手続き型言語です。
- **データベースサービス** ユーザーデータベースをクラウドベースの環境で作成して管理します。
- **ユーザー管理** DBS 操作がクラウドベース環境でできるように SQL/MX で導入されました。
- **スキーマレベル権限** 権限をスキーマレベルで付与したり取り消します。
- **MXOAS 構成可能な IP アドレス**のサポート。
- **MXCS AF_UNIX プロトコル**のサポートでデータベース接続が CLIM 障害の影響を受けなくなります。
- **テーブルサイズ指定** 大きなテーブルの初期および最大サイズを設定します。

サポート対象外のスキーマバージョン

SQL/MX リリース 3.5 が v1200 スキーマバージョンをサポートしません。

参考資料

- *SQL/MX 3.4 リファレンスマニュアル*
- *SQL/MX 3.5 データベースサービスマニュアル*
- *SQL/MX 3.5 Installation and Upgrade Guide*
- *SQL/MX 3.5 Procedural Language for SQL/MX (PL/MX) Reference Manual*
- *MXDM User Guide for SQL/MX Release 3.5*
- *SQL/MX 3.5 Messages Manual*
- *SQL/MX 3.4 Guide to Stored Procedures in Java*
- *SQL/MX 3.4 Management Manual*
- *SQL/MX 3.4 Query Guide*
- *SQL/MX Connectivity Service Manual for SQL/MX Release 3.4*
- *ODBC/MX Client Drivers User Guide for SQL/MX Release 3.4*
- *JDBC Type 2 Driver Programmer's Reference for SQL/MX Release 3.4*
- *JDBC Type 4 Driver Programmer's Reference for SQL/MX Release 3.2.1*

データベースの互換性

SQL/MX 3.5 には、SQL/MX へのアプリケーションの移行を容易にするために以下のデータベース互換機能が用意されています。

- **DATE type2** データ型
- **NUMBER** データ型
- **VARCHAR2** データ型
- **TO_DATE** 関数
- **マテリアライズドビュー**
- **MERGE ステートメント**

これらの機能のほとんどはデフォルトで有効になっていますが、いくつかは CQD DC_MODE を 1 に設定することで有効にする必要があります。

既存のデータベース互換の関数と式のリストは、**Oracle と互換性のある SQL/MX の関数と式**を参照してください。

CQD DC_MODE

データベースの互換性の機能を有効にする、または無効にするために DC_MODE CQD が導入されました。有効な値は 1（有効）および 0（無効）です。デフォルト値は、0 です。DC_DATE のような個々の機能固有の CQD を設定すると、DC_MODE 設定は上書きされます。

DATE type2

DATE type2 は、日付と時刻の値を表す新しいデータ型です。これは年、月、日、時間、分、および秒の値を使用して日付を表します。ストレージサイズは 7 バイトです。DATE データ型の SQL/MX 実装では、年、月、および日の値を使用して日付形式を表します。ストレージサイズは 4 バイトです。DATE type2 を使用する場合の構文は、DATE と同じです。デフォルトの表示書式は'YYYY-MM-DD'です。DATE type2 データ型の表示形式を変更する､新しいセッションデフォルトである DATE_FORMAT が導入されました。

アプリケーションは、DC_MODE CQD を 1 に設定することによってデータベース互換モードに切り替えることができます。データベース互換モードでは、DATE キーワードは、DDL、DML 文、および Utility コマンドの DATE type2 データ型を示します。

DATE type2 データ型をサポートするものは次のとおりです。

- すべての SQL/MX ユーティリティ
- Windows ANSI および UNICODE ODBC/MX ドライバー
- JDBC T2 および T4 ドライバー

組み込み型 C/C++および COBOL アプリケーション、OSS、Linux、および HP-UX ODBC/MX ドライバーは DATE type2 データ型をサポートしていません。

次の例では、DC_MODE CQD を 1 に設定して DATE type2 列のあるテーブルを作成し、値を挿入しています。

```
-- DC_MODE default attribute is set to 1.
>>control query default DC_MODE '1';

--- SQL operation complete.

-- Create a table with a DATE type2 column
>>create table dc_testtab1 (col1 DATE);

--- SQL operation complete.
```

```
-- DATE type2 column accepts date and time values as shown below
>>insert into dc_testtab1 values (DATE '2016-09-23 11:12:30');

--- 1 row(s) inserted.
```

日付時刻リテラル

日付時刻リテラルは、DATE type2 用の日付と時刻のフィールドを持つ値を受け入れます。DATE 部分を指定しない場合、デフォルト値は現在の月の初日です。TIME 部分を指定しない場合、デフォルト値は 00:00:00 A.M です。次の表では、日付時刻リテラルの解釈の例をいくつか示します。

ユーザー指定リテラル	DATE type2 リテラル
DATE '2016-10-11 13:20:10'	DATE '2016-10-11 13:20:10'
DATE '13:20:10'	DATE '**2017-02-01** 13:20:10'
DATE '2016-10-11'	DATE '2016-10-11 **00:00:00**'

DDL 文の DEFAULT 句もしくは FIRST KEY 句で DATE type2 列を使用する場合、日付部分を明示する必要があります。

算術演算

DATE type2 データ型は、SQL/MX DATE としてすべての算術セマンティクスに従います。ただし、時刻の部分は DATE type2 値の評価中に常に考慮されます。2 つの DATE type2 値の減算の結果は日数であり、返される型は NUMERIC（18、6）です。次の例では、DATE_FORMAT セッションデフォルト値を設定し、DATE type2 を別の DATE type2 から減算した結果を示します。

```
>>control query default DC_MODE ‘1’;

--- SQL operation complete.

>>create table t1(a DATE, b DATE);

--- SQL operation complete.


>>set session default DATE_FORMAT 'YYYY-MM-DD HH24:MI:SS';

--- SQL operation complete.

>>select a, b, a-b from t1;
A                    B                    (EXPR)
-------------------  -------------------  ---------------------
2016-10-25 11:10:20  2016-10-24 10:10:20               1.041666

--- 1 row(s) selected.
```

DATE type2 および TIMESTAMP データ型間の減算はサポートされています。戻り値は、INTERVAL SECOND 型となります。

リレーショナルと割り当ての操作

DATE type2 を別の DATE type2 と比較する場合、時間部分を含むすべてのフィールドが比較されます。次に例を示します。

`D1= 2016-06-06 17:38:43  D2=2016-06-06 17:40:02` とします。
この時、`D1` と `D2` を比較すると下記の結果となります。

`D1 = D2` `FALSE` が返ります。
`D1 > D2` `FALSE` が返ります。
`D1 < D2` `TRUE` が返ります。

DATE type2 値は、TIMESTAMP 列、またはリテラルに割り当てる、および比較することができ、またその逆も可能です。

CAST 式

式は DATE type2 データ型にキャストできます。主な CAST の有効な変換は、次のとおりです。

- 文字列への、または TIMESTAMP への DATE type2 値の CAST
- DATE type2 への TIMESTAMP 値の CAST

DATE type2 への TIME 値のキャストはサポートされていません。

留意事項

- DATE および DATE type2 は、同じアプリケーション内に存在できません。
- テーブルは、DATE データ型と DATE type2 データ型のいずれかの列を含むことができ、両方を含むことはできません。
- DATE type2 データ型を使用したクエリを MFC がキャッシュすることはできません。
- SPJRS での DATE type2 データ型を使用する場合、適切な長さの TIMESTAMP(0)または CHAR へのキャストが必要です。
- SYSDATE、SYSTIMESTAMP、および INSTR は、SQL/MX 3.5 で予約語として予約されており、識別子として使用することはできません。予約語を識別子として使用するには、二重引用符で囲みます。

注記:

DATE type2 列を含むテーブルやインデックスでのオンラインユーティリティの操作はサポートされていません。

日付時刻関数

DATE type2 はすべての日付時刻関数と互換性があり、引数または戻り値のタイプとして使用できます。DATE type2 の動作は、以下の日付時刻関数の既存の DATE データ型に似ています。

- DATE_ADD
- DATEADD
- DATE_SUB
- DATEDIFF
- DATEFORMAT
- DAY
- DAYNAME
- DAYOFMONTH
- DAYOFWEEK
- DAYOFYEAR
- JULIANTIMESTAMP
- LAST_DAY
- MONTH
- MONTHNAME
- MONTHS_BETWEEN
- QUARTER
- TO_CHAR(<DATETIME>)
- WEEK
- YEAR

DATE type2 の動作は、以下の日付時刻関数の既存の DATE データ型とは異なります。

- CURRENT_DATE
- EXTRACT
- MIN
- MAX

DATE type2 は、HOUR、MINUTE、および SECOND 関数とともに使用できます。

CURRENT_DATE

DC_MODE CQD が 1 に設定されている場合、CURRENT_DATE 関数は、現地の現在の日付を DATE type2 データ型の値として返します。出力の文字列の形式は、DATE_FORMAT セッションデフォルト設定に依存します。

例

```
>>set session default DATE_FORMAT 'yyyy-mm-dd hh24:mi:ss';

--- SQL operation complete.


>>select CURRENT_DATE from (values(1)) as t;
(EXPR)
-------------------
2016-10-24 11:25:30
--- 1 row(s) selected.
```

EXTRACT

EXTRACT 関数は、DATE type2 から時刻フィールド、HOUR、MINUTE、および SECOND を抽出します。

例

```
>>select CURRENT_DATE, EXTRACT (HOUR FROM CURRENT_DATE) from (values(1)) t;
(EXPR)                 (EXPR)
---------------------- ------
2016-10-24 11:25:30    11
--- 1 row(s) selected.
```

MIN

MIN 関数は、時間部分を考慮して DATE type2 の最小値を評価します。

MAX

MAX 関数は、時間部分を考慮して DATE type2 の最大値を評価します。

DATE_FORMAT セッションデフォルト

構文

```
SET SESSION DEFAULT DATE_FORMAT format-string;
```

説明

DATE_FORMAT は、DATE type2 を書式設定文字列に変換するための書式文字列を指定する、セッションのデフォルト属性です。デフォルトの書式文字列は'YYYY-MM-DD'です。このセッションデフォルト値は、DATE type2 データ型にのみ適用できます。

パラメーター

`format-string`

出力形式を定義する定数文字列を指定します。次の表は、サポートされる書式のリストです。

表 1: 書式文字列要素

要素	説明	*datetime-expression*	*`format-string`*	出力
- / , . ; : “text”	出力形式で日付時刻フィールドのセパレーターまたは区切り記号。引用符（""）の内側のテキストは、引用符を削除した後は何も変更されずに再現されます。	31-MAR-11 05.02.31.123457 AM	'DD/MM/YYYY'	'31/03/2011'
AD A.D. BC B.C.	ピリオドあり、またはピリオドなしの Anno Domini または Before Christ インジケーター。	31-MAR-11 05.02.31.123457 AM	'YYYY AD'	'2011 AD'
AM A.M. PM P.M.	ピリオドあり、またはピリオドなしの Ante Meridian（午前）または Post Meridian（午後）インジケーター。	31-MAR-11 05.02.31.123457 AM	'HH:MM AM'	'05:02 AM'
CC SCC	世紀	31-MAR-11 05.02.31.123457 AM	'CC'	‘21’
D	曜日（1～7） 日曜日 = 1	31-MAR-11 05.02.31.123457 AM	'D'	‘5’
DAY day Day	要素の大文字と小文字に基づいて大文字、小文字、または先頭を大文字とした、最も長い曜日の名前の幅を表示するために空白が埋め込まれた、曜日の名前。	31-MAR-11 05.02.31.123457 AM	'DAY' 'day' 'Day'	'THURSDAY' 'thursday' 'Thursday'
DD	月の日（日付）（1～31）	31-MAR-11 05.02.31.123457 AM	'DD'	‘31’

表は続く

要素	説明	*datetime-expression*	*format-string*	出力
DDD	年初からの日数（1～366）	31-MAR-11 05.02.31.123457 AM	'DDD'	‘090’
DY dy Dy	要素の大文字と小文字に基づいて、大文字、小文字、または先頭を大文字にした曜日の短縮形。	31-MAR-11 05.02.31.123457 AM	'DY' 'dy' 'Dy'	'THU' 'thu' 'Thu'
FM	埋め込みモード修飾子を使用して、先頭または末尾の空白を削除します。	31-MAR-11 05.02.31.123457 AM	'FMMONTH'	'MARCH'
HH HH12	時（1～12）	31-MAR-11 05.02.31.123457 AM	'HH'	‘05’
HH24	時（0～23）	31-MAR-11 05.02.31.123457 AM	‘HH24’	‘05’
IW	ISO 標準に基づいた年初からの週数（1～52 または 1～53）。	31-MAR-11 05.02.31.123457 AM	'IW'	‘13’
IYYY IYY IY I	ISO の年の最後の 4、3、2、または 1 桁。	31-MAR-11 05.02.31.123457 AM	'IYYY'	‘011’
YEAR year Year SYEAR Syear SYear	要素の大文字と小文字に基づいて年を大文字、小文字、または先頭を大文字にして略さずに記述します。	31-MAR-11 05.02.31.123457 AM	'YEAR' 'year' 'Year'	'TWENTY ELEVEN' 'twenty eleven' 'Twenty Eleven'
J	ユリウス日、つまり、紀元前 4712 年 1 月 1 日から経過した日数。	31-MAR-11 05.02.31.123457 AM	'J'	‘2455652’
MI	分（0～59）	31-MAR-11 05.02.31.123457 AM	'MI'	‘02’

表は続く

要素	説明	*datetime-expression*	*format-string*	出力
MM	月（01～12、つまり 1 月は 01）	31-MAR-11 05.02.31.123457 AM	'MM'	'03'
MON mon Mon	要素の大文字と小文字に基づいた、大文字、小文字、または先頭を大文字にした月の短縮形。	31-MAR-11 05.02.31.123457 AM	'MON' 'mon' 'Mon'	'MAR' 'mar' 'Mar'
MONTH month Month	要素の大文字と小文字に基づいて大文字、小文字、または先頭を大文字とした、最も長い月の名前の幅を表示するために空白が埋め込まれた月の名前。	31-MAR-11 05.02.31.123457 AM	'MONTH' 'month' 'Month'	'MARCH' 'march' 'March'
RM	ローマ数字（I～XII）の月。	31-MAR-11 05.02.31.123457 AM	'RM'	'III'
RR RRRR	年の最後の 2 桁または 4 桁。	31-MAR-11 05.02.31.123457 AM	'RR' 'RRRR'	'11' '2011'
SS	秒（0～59）	31-MAR-11 05.02.31.123457 AM	'SS'	'31'
SSSSS	午前 0 時から経過した秒数（0～86399）。	31-MAR-11 05.02.31.123457 AM	'SSSSS'	'18151'
X	現地の小数点文字。サポートされる現地の小数点文字は'.'（'ドット）のみです。	31-MAR-11 05.02.31.123457 AM	'HH:MM:SSXFF'	'05:03:31.123457'
Y,YYY	指定された位置にカンマが入った年。	31-MAR-11 05.02.31.123457 AM	'Y,YYY'	'2,011'
YYYY SYYYY	4 桁の年。S は BC 日付の先頭にマイナス記号を付けます。	31-MAR-11 05.02.31.123457 AM	'YYYY'	'2011'
YYY YY あり	年の最後の 3 桁、2 桁、または 1 桁。	31-MAR-11 05.02.31.123457 AM	'YY'	'11'

表は続く

要素	説明	*datetime-expression*	*format-string*	出力
WW	年初からの週数（1～53）	31-MAR-11 05.02.31.123457 AM	'WW'	‘13’
W	月の週（1～5）	31-MAR-11 05.02.31.123457 AM	'W'	‘5’
th	サフィックス st/nd/th を数値の出力形式に追加します。 注記：'th'サフィックスは任意の要素に追加できます。サフィックス st/nd/th は、数値以外の出力値の場合は無視されます。	31-MAR-11 05.02.31.123457 AM	'DDth' 'DYth'	‘31st’ 'THU'

留意事項

- *format-string* の長さは 74 文字以下にする必要があります。
- NULL、列名、式、または動的パラメーターは *format-string* として指定できません。
- DATE_FORMAT セッションのデフォルト属性は mxci インターフェイスと JDBC T2 ドライバーでサポートされています。

  ODBC または JDBC T4 アプリケーションから DATE_FORMAT を使用するには、書式設定された出力を保持する十分な長さのある CHARACTER 文字列として DATE type2 をキャストします。

CQD DC_DATE

DC_DATE は日付タイプ 2 の動作を有効または無効にする新しい属性です。属性 DC_MODE より高い優先順位を持っています。

属性の有効な値は次のとおりです。-1、0、1

DC_DATE を-1 に設定すると、DC_MODE 設定に基づいて DATE type2 が有効に、または無効になります。これはデフォルト値です。

DC_DATE を 0 に設定すると、DC_MODE 設定に関係なく DATE type2 が無効になります。

DC_DATE を 1 に設定すると、DC_MODE 設定に関係なく DATE type2 が有効になります。

DC_DATE	DC_MODE を 0 に設定	DC_MODE1 に設定
-1	SQL/MX DATE	DATE type2
0	SQL/MX DATE	SQL/MX DATE
1	DATE type2	DATE type2

ODBC 変換

ODBC アプリケーションは、DATE type2 値をバインドおよび取得する SQL タイプとして、CHAR、DATE、TIME、または TIMESTAMP を使用します。

C データ型から SQL データ型、およびその逆への可能な変換の一覧を以下の表に示します。

C データ型	SQL データ型
SQL_C_TYPE_DATE または SQL_C_TYPE_TIME または SQL_C_TYPE_TIMESTAMP	SQL_CHAR
	SQL_VARCHAR
	SQL_LONG_VARCHAR
	SQL_TYPE_DATE
	SQL_TYPE_TIME
	SQL_TYPE_TIMESTAMP

Windows UNICODE ドライバーで可能な変換を以下の表に示します。

C データ型	SQL データ型
SQL_C_TYPE_DATE または SQL_C_TYPE_TIME または SQL_C_TYPE_TIMESTAMP	SQL_WCHAR
	SQL_WVARCHAR
	SQL_WLONG_VARCHAR
	SQL_TYPE_DATE
	SQL_TYPE_TIME
	SQL_TYPE_TIMESTAMP

1. SQLBindParameter ODBC API C タイプが SQL_C_TYPE_DATE であり、SQL タイプが SQL_DATE である場合、DATE type2 列に日付値が挿入されます。時間は日付の一部ではないため、デフォルトの時間は SQL/MX に挿入されます。たとえば、DATE 列から 2016-11-04 がバインドされる場合、SQL/MX に挿入される値は、2016-11-04 00:00:00 A.M.です。
2. SQLBindParameter ODBC API C タイプが SQL_C_TYPE_TIME であり、SQL タイプが SQL_DATE である場合、DATE type2 列に時間値が挿入されます。日付は時間の一部ではないため、デフォルトの日付が SQL/MX に挿入されます。たとえば、TIME 列に 10:58:59 がバインドされる場合、SQL/MX に挿入される値は 2016-11-01 10:58:59 A.M.です（ここで 2016-11-01 は現在の月の初日を表します）。
3. SQLBindParameter ODBC API C タイプが SQL_C_TYPE_TIMESTAMP であり、SQL タイプが SQL_DATE である場合、DATE type2 列にタイムスタンプ（0）値が挿入されます。たとえば、TIMESTAMP 列から 2016-08-23 11:38:49.23145 がバインドされる場合、SQL/MX に挿入される値は、2016-08-23 11:38:49 A.M.です。
4. SQLBindCol または SQLGetData ODBC API で結果を取得する間、アプリケーションが DATE type2 列のターゲットタイプを、DATE 値をアプリケーションバッファーにコピーする SQL_C_TYPE_DATE で設定したとします。この場合、たとえば挿入された値が 2016-11-04 11:12:30 A.M であれば、DATE でバインドされるため、取得される値は 2016-11-04 となります。
5. SQLBindCol または SQLGetData ODBC API で結果を取得する間、アプリケーションが DATE type2 列のターゲットタイプを、TIME 値をアプリケーションバッファーにコピーする SQL_C_TYPE_TIME で設定したとします。この場合、たとえば挿入された値が 2016-11-10 10:58:59 A.M であれば、TIME でバインドされるため、取得される値は 2016-11-04 となります。
6. SQLBindCol または SQLGetData ODBC API で結果を取得する間、アプリケーションが DATE type2 列のターゲットタイプを、TIMESTAMP 値をアプリケーションバッファーにコピーする SQL_C_TYPE_TIMESTAMP で設定したとします。この場合、たとえば挿入された値が 2016-08-23

10:58:59 A.M であれば、TIMESTAMP でバインドされるため、取得される値は 2016-08-23 10:58:59 となります。

JDBC 変換

JDBC アプリケーションでは、setObject()、setDate()、setString()、setTime()、および setTimeStamp()メソッドを使用して値をバインドでき、getObject()、getDate()、getTime()、getTimeStamp()、および getString()メソッドを使用して値を取得できます。DATE type2 は、Java の java.sql.Timestamp オブジェクトにマッピングされます。DATE type2 列をバインドするために setTime()メソッドが呼び出される場合、デフォルトの日付値は 1970-01-01 です。DATE type2 列をバインドするために setDate()メソッドが呼び出される場合、デフォルトの時間値は 00:00:00 AM です。

DC_MODE が 1 に設定されている場合、ResultSetMetadata メソッドで新しい DATE type2 列タイプが表示されます。

表 2: 設定メソッドの変換の例

オブジェクトタイプの例	SQL/MX に格納される値
java.sql.Date - 2016-09-23	2016-09-23 00:00:00
java.sql.Time - 00:21:56	1970-01-01 00:21:56
java.sql.Timestamp - 2016-09-23 17:03:50	2016-09-23 17:03:50

表 3: 取得メソッドの変換の例

SQL/MX からの例	取得メソッド - 表示される値
2016-09-23 17:03:50	getObject() - 2016-09-23 17:03:50.0
2016-10-25 00:00:00	getDate() - 2016-10-25
2016-11-23 00:21:56	getTime() - 00:21:56
2016-09-23 17:03:50	getTimeStamp() - 2016-09-23 17:03:50.0
2016-10-25 17:10:56	getString() - 2016-10-25 17:10:56

NUMBER

NUMBER は、有効桁数とスケールが固定された正確な数値を表す新しいデータ型です。

NUMBER データ型の構文は以下のようになります。

```
NUMBER [(precision [,scale])] [SIGNED|UNSIGNED]
```

上記において、

precision は、桁数の合計を指定します。サポートされる範囲は 1～128 です。

scale は、小数点右側の桁数を指定します。有効桁数を超えてはなりません。サポートされる範囲は 0～128 です。

有効桁数、スケール値および符号のタイプはオプションです。デフォルトは NUMBER (18,6) SIGNED です。有効桁数とスケールの両方の値が指定されていない場合は、デフォルトの有効桁数とスケールの値が

使用されます。有効桁数の値が指定されているけれどもスケールの値が指定されていない場合、スケールはゼロです。

NUMBER データ型は、バイナリに格納され、有効桁数が最大 9 の符号の付いた数と有効桁数が最大 18 の符号が付かない数がハードウェアでサポートされます。他のすべての場合、数はソフトウェアでサポートされますが、効率が下がります。

NUMBER_DEFAULT_PRECISION と NUMBER_DEFAULT_SCALE という 2 つの新しい属性が、NUMBER データ型のデフォルトの有効桁数とスケールの値をカスタマイズできるように導入されています。詳しくは、**属性**を参照してください。

NUMBER データ型は、以下によってサポートされます。

- すべての SQL/MX ユーティリティ
- すべての JDBC および ODBC ドライバー

組み込み型 C/C++および COBOL アプリケーションは NUMBER データ型をサポートしていません。

留意事項

- NUMBER データ型は、既存の NUMERIC データ型へのマッピングとして実装されます。SHOWDDL コマンドは、NUMBER 列の対応する NUMERIC マッピングを表示します。
- NUMBER データ型を使用したクエリを MFC がキャッシュすることはできません。

例

この例では、有効桁数とスケールの値を指定せずに作成された NUMBER 列を示します。デフォルトの有効桁数とスケールの値（18、6）が使用されています。SHOWDDL 出力で NUMERIC（18、6）と表示されます。

```
>>create table numtest1(col1 NUMBER);

--- SQL operation complete.
>>showddl numtest1;

CREATE TABLE CAT.SCH.NUMTEST1
  (
    COL1                              NUMERIC(18, 6) DEFAULT NULL
  )
  LOCATION \NSK.$DATA.ZSDC68HQ.HF6XZH00
  NAME NSK_DATA_ZSDC68HQ_HF6XZH00
  ATTRIBUTES BLOCKSIZE 4096
  NO PARTITION
  ;
--- SQL operation complete.
```

この例では、有効桁数の値だけを指定して作成された NUMBER 列を示しています。スケールの値は「0」と見なされます。

```
>>create table numtest2 ( col1 NUMBER(20));

--- SQL operation complete.

>>showddl numtest2;

CREATE TABLE CAT.SCH.NUMTEST2
  (
    COL1                    NUMERIC(20, 0) DEFAULT NULL
  )
```

```
  LOCATION \NSK.$DATA2.ZSDC68HQ.JRW5ZK00
  NAME NSK_DATA2_ZSDC68HQ_JRW5ZK00
  ATTRIBUTES BLOCKSIZE 4096
  NO PARTITION
  ;

--- SQL operation complete.
```

この例では、有効桁数値とスケール値の両方を指定して作成された NUMBER 列を示しています。

```
>>create table numtest3 ( col1 NUMBER(20,10));
--- SQL operation complete.

>>showddl numtest3;

CREATE TABLE CAT.SCH.NUMTEST3
  (
                COL1                    NUMERIC(20, 10) DEFAULT NULL
  )
  LOCATION \NSK.$DATA.ZSDC68HQ.X63DCR00
  NAME NSK_DATA_ZSDC68HQ_X63DCR00
  ATTRIBUTES BLOCKSIZE 4096
  NO PARTITION
  ;

--- SQL operation complete.
```

VARCHAR2

VARCHAR2 は可変長文字の文字列を格納する非 ANSI データ型です。VARCHAR2 データ型列には、mxci、rmxci、JDBC、および ODBC アプリケーションからアクセスできます。MXDM から VARCHAR2 列を含むテーブルを作成および管理できます。VARCHAR2 データ型をサポートするものは次のとおりです。

- すべての SQL/MX ユーティリティ
- Windows ANSI および UNICODE ODBC/MX ドライバー
- JDBC T2 および T4 ドライバー

組み込み型 C/C++および COBOL アプリケーション、OSS、Linux、および HP-UX ODBC/MX ドライバーは VARCHAR2 データ型をサポートしていません。

関連するデータ型が VARCHAR2 である場合、SQL/MX は空の文字列を NULL と見なします。NOT NULL VARCHAR2 列に空の文字列を挿入すると、操作は失敗しエラー 8421 が表示されます。NOT NULL VARCHAR2 列への空の文字列の割り当てがコンパイル時に検出された場合、文のコンパイルは失敗しエラー 4122 が表示されます。関連付けられたデータ型が VARCHAR2 である場合、NULL に関連付けられたすべてのセマンティックスが空の文字列に適用されます。

SQL/MX は、VARCHAR2 データ型の文字列を比較するときに、末尾のスペースを考慮します。

ODBC および JDBC アプリケーションは、値をバインドおよび取得するために、ANSI タイプの VARCHAR または CHAR を SQL タイプとして使用します。VARCHAR2 データ型を使用したクエリを MFC がキャッシュすることはできません。

VARCHAR2 の構文は次のとおりです。

```
VARCHAR2 (length) [CHARACTER SET char-set-name] [COLLATE DEFAULT] [UPSHIFT]
```

ここで、

length は、列内で許可される文字数を指定する正の整数です。length の値を指定する必要があります。

char-set-name は文字セットであり、これには ISO88591 でも UCS2 でも指定できます。

UPSHIFT 句は、列に格納する前に文字を大文字に変換するよう NonStop SQL/MX に指示します。

VARCHAR2 が他のデータ型とともに述語で使用される場合、その述語の評価中に、VARCHAR2 セマンティクスが適用されます。

パーティショニングキーとして VARCHAR2 列を使用するには、列の文字セットは ISO88591 である必要があります。

留意事項

- VARCHAR2 は ISO88591 と UCS2 の文字セットをサポートし、デフォルトは ISO88591 です。
- VARCHAR2 構文は、CREATE TABLE 文、ALTER TABLE 文、および CAST 式で使用できます。

注記:

VARCHAR2 列を含むテーブルまたはインデックスに対するオンラインユーティリティ操作はサポートされていません。

VARCHAR と VARCHAR2 の違い

シナリオ	VARCHAR	VARCHAR2
空文字列	空の文字列として保存されます。 NULL としては扱われません。	NULL として保存され扱われます。
キー列の値	末尾のスペースは無視されます。	末尾のスペースは受け入れられます。
文字列比較	末尾のスペースは無視されます。	末尾のスペースは受け入れられます。
予約済みのキーワード	はい	いいえ

例

次の例では、末尾のスペースに対する VARCHAR と VARCHAR2 の動作を示します。VARCHAR2 データ型では、2 つの文字列の比較中に末尾のスペースを受け入れます。

```
>>create table vc (a varchar(10) not null primary key, b varchar(10));

--- SQL operation complete.
>>create table vc2 (a varchar2(10) not null primary key, b varchar2(10));

--- SQL operation complete.
>>insert into vc values ('abcd','abcd    ');

--- 1 row(s) inserted.
>>select * from vc where a = b;

A           B
```

```
----------  ----------

abcd        abcd

--- 1 row(s) selected.
>>insert into vc2 values ('abcd','abcd    ');

--- 1 row(s) inserted.
>>select * from vc2 where a = b;

--- 0 row(s) selected.
>>
```

次の例では、末尾のスペースを含むキー列に対する VARCHAR と VARCHAR2 の動作を示します。

```
>>insert into vc values ('abcd    ','abcd    ');

*** ERROR[8102] The operation is prevented by a primary key
VC_651493923_3328 on table CAT.SCH.VC.

--- 0 row(s) inserted.

>>insert into vc2 values ('abcd    ','abcd    ');

--- 1 row(s) inserted.
```

VARCHAR2 の SQL データ型への変換

ODBC および JDBC アプリケーションは、VARCHAR2 値をバインドおよび取得するために、ANSI タイプの VARCHAR または CHAR を SQL データ型として使用します。

C データ型から SQL データ型、およびその逆への可能な変換の一覧を以下の表に示します。

C データ型	**SQL データ型**
SQL_C_CHAR	SQL_CHAR
	SQL_VARCHAR
	SQL_LONG_VARCHAR
	SQL_ DECIMAL
	SQL_ NUMERIC

Windows UNICODE ドライバーで可能な変換を以下の表に示します。

C データ型	**SQL データ型**
SQL_C_CHAR または SQL_C_WCHAR	SQL_WCHAR
	SQL_WVARCHAR
	SQL_WLONG_VARCHAR

TO_DATE 関数

構文

```
TO_DATE (value_string, format_specifier)
```

説明

TO_DATE 関数は、日付または日付時刻値を含む文字列を DATE データ型に変換します。この関数は、データベース互換モードでは DATE type2 オブジェクトを返します。それ以外の場合は DATE オブジェクトを返します。

パラメーター

value_string

DATE データ型に変換される日付または日付時刻値です。

format_specifier

value_string を解釈するために使用される形式です。デフォルトの形式は、SQL/MX の日付の場合は *YYYY MM-DD*、DATE type2 の場合は *YYYY MM-DD HH24:MI:SS* です。format_specifier はオプションパラメーターです。指定しないと、デフォルト形式が使用されます。SQL/MX でサポートされている format_specifier エレメントを表示するには、**書式文字列要素**を参照してください。

このパラメーターは次のいずれかのタイプに指定できます。

- 文字列リテラル
- ホスト変数
- 列参照
- 動的パラメーター
- 関数の出力

留意事項

- 日付フィールドに日付が指定されていない場合、デフォルトで指定の月の最初の日になります。
- 日付フィールドに月が指定されていない場合、デフォルトで現在の月になります。
- 日付フィールドに年が指定されていない場合、デフォルトで現在の年になります。
- この関数は、入力引数での余分な先頭と末尾のスペースを無視します。
- 入力引数での余分な区切り文字は無視されます。
- value_string と format_specifier では異なる区切り文字を使用できます。
- format_specifier エレメントでは大文字と小文字が区別されません。
- ISO88591 文字セットだけが入力文字列でサポートされています。CHAR データ型と VARCHAR データ型だけがサポートされています。
- タイムゾーン、NULL 値、および SQL/MP テーブルはサポートされていません。
- MV 定義での TO_DATE 関数の使用は、データベース互換モードではサポートされていません。

例

- 次の例では、値文字列と形式文字列で使用されている区切り文字が異なっている場合でも、日付値が返されます。

```
>>select TO_DATE ('31-DEC-2015','DD-MON-YYYY') from (values(1))t;

(EXPR)
----------

2015-12-31
```

```
--- 1 row(s) selected.
```

- 次の例では、format_specifier が指定されていないので、デフォルトの形式が使用されます。

```
>>select TO_DATE ('2016-08-16') from (values(1))t;

(EXPR)
----------

2016-08-16

--- 1 row(s) selected.
```

- 次の例では、時間が指定されていないので、エラーメッセージが返されます。

```
>>select TO_DATE ('29-FEB-2016', 'DD-MON-YYYY HH' ) from (values(1))t;

*** ERROR[8973] Value string '29-FEB-2016' doesnot match with format 'DD-
MON-YYYY HH'

--- 0 row(s) selected.
```

- 次の例では、値文字列が無効な日付なので、エラーメッセージが表示されます。

```
>>select TO_DATE ('32-DEC-2015','DD-MON-YYYY') from (values(1))t;

*** ERROR[8972] Invalid value 32 for DAY.

--- 0 row(s) selected.
```

マテリアライズドビュー

ビューの実体化は、Business Intelligence アプリケーションでよく使用されるデータベース操作です。実体化後に、通常のテーブルと同様にビューのタプルが保存され、直接マテリアライズドビュー（MV）をクエリできます。頻繁に実行される複雑な結合によるクエリがある場合、そのクエリの結果を実体化し、実体化された結果にアクセスすることで、結合は一回だけ実行され、何度もアクセスすることで、そのデータへのアクセスを高速化できます。クエリ率が高く、クエリが複雑なアプリケーションでは速度の違いが重要となる場合があります。たとえば、複雑な結合は大量のデータを含む多くのテーブルにまたがります。MV は、データウェアハウス、レプリケーションサーバー、課金、記録システム、モバイルシステムなどのアプリケーションで役に立ちます。

MV は、実テーブルに対する増分の変更で更新することも、再計算することもできます。実テーブルで発生する増分変更でメンテナンスされる場合、更新される MV は増分更新 MV と呼ばれます。実テーブルの変更とともに MV のメンテナンスが発生している場合は、ON STATEMENT 句を使用して増分の更新を行うことができます。

SQL/MX リリース 3.5 では、結合クエリを持つ MV 定義をサポートしています。結合クエリとして定義される MV は、結合マテリアリズドビューと呼ばれます。集約クエリを含む、または集約クエリと結合クエリの両方を含む MV を定義すると、SQL/MX はエラーを返します。

SHOWDDL、mxtool 操作、SQL/MXHealthCheck ツール、および UPDATE STATISTICS は MV をサポートするように拡張されています。

ここでは、MV をサポートするためにコマンドとユーティリティに行われた変更について詳しく説明します。

CREATE MV

構文

```
CREATE {MATERIALIZED VIEW  | MV} mv-name [column-name-list]
{ REFRESH refresh-type }{ initialization-type } [file-options]
AS query-expr

mv-name is:
[catalog-name.][schema-name.]mv-name

column-name-list is:
        (column-name [, column-name]...)

refresh-type is:
        ON STATEMENT

initialization-type is:
        INITIALIZE ON CREATE

file-options is:
        STORE BY store-option
        | LOCATION [\node.]$volume[.subvolume.file-name]
        [NAME partition-name]
        | partn-file-option
        | ATTRIBUTE[S] attribute [,attribute]...

store-option is:
        (key-column-list)

partn-file-option is:
        { [RANGE] PARTITION
        [BY (partitioning-column [,partitioning-column]...)]
        [(ADD range-partn-defn [,ADD range-partn-defn]...)]
        | HASH PARTITION
        [BY (partitioning-column [,partitioning-column]...)]
        [(ADD partn-defn [,ADD partn-defn]...)]}


range-partn-defn is:
            FIRST KEY {col-value | (col-value [,col-value ]...)}
            partn-defn

partn-defn is:
        LOCATION [\node.]$volume[.subvolume.file-name]
  [NAME partition-name]
        [EXTENT ext-size | (pri-ext-size [,sec-ext-size])]
        [MAXEXTENTS num-extents]

attribute is:
     {AUDITCOMPRESS | NO AUDITCOMPRESS}
        | BLOCKSIZE number-bytes
        | {CLEARONPURGE | NO CLEARONPURGE}
        | EXTENT ext-size | (pri-ext-size [,sec-ext-size])
        | MAXEXTENTS num-extents
```

```
query-expr is:
        SELECT column-expr [, column-expr]...
        FROM table-ref [, table-ref] ...
        [WHERE search-condition]

table-ref is:
        simple-table | joined-table

simple-table is:
        {base-table-name}
        [[AS] corr-name [(col-expr-list)]]

joined-table is:
        table-ref [NATURAL] [INNER] JOIN table-ref [join-spec]

join-spec is:
        ON search-condition

column-expr is:
        {non-aggregate-column-expr [[AS] derived-name]
```

説明

この文は、SQL/MX テーブルに MV を作成するために使用されます。MV は、通常のテーブルと同様に、クエリの結果を保存するデータベースオブジェクトです。MV は、実テーブルを問い合わせることなく直接問い合わせることができます。

オプション

mv-name

MV の ANSI 論理名を指定します。*mv-name* は、*[catalog-name.][schema-name.]mv-name* 形式の一部で、ここで名前の各部分は有効な SQL 識別子です。最大 128 文字を指定できます。*mv-name* は、スキーマ内で一意でなければなりません。

column-name-list

MV に列の名前を指定します。リスト内の列名は、*query-expr* の列に対応します。

refresh-type

MV がいつどのように更新されるかを指定します。

ON STATEMENT

query-expr が ON STATEMENT である場合、MV は実テーブルの更新時にただちに更新されます。MV は実テーブルを更新する文の一部として自動的に維持されます。そのため、初期化されると、MV は常に INSERT、UPDATE、または DELETE 文を介して実テーブルと一致します。

refresh-type が ON STATEMENT である場合、MV 更新では、対応する MV に対する同時更新が必要です。

initialization-type

MV が実テーブルから初期コンテンツをいつ取得するかを指定します。

INITIALIZE ON CREATE

MV の作成時に、MV が挿入されることを指定します。

STORE BY store-option

MV のクラスタリングキーの基になる一連の列を指定します。クラスタリングキーによって、MV を保持する物理ファイル内の行の順序が決定されます。

STORE BY 句を指定しない場合、SQL/MX は実テーブルのストレージの順序とプライマリキーからストア順を決定します。

search-condition および *column-expr* の説明については、*SQL/MX 3.4 リファレンスマニュアル*を参照してください。

許可

MV を作成するには、スキーマを所有する、または Super ID を持つ、あるいは、MV を作成する必要があるスキーマに対して CREATE アクセス許可を持つ必要があります。すべての基礎となるテーブルとビューに対する SELECT 権限が必要です。

留意事項

- MV 定義は、2 つまたは複数のテーブルの結合を持つことができますが、内部等値結合のみが許可されます。
- *query-expr* の式または関数が暗黙的な名前を持たない場合、SELECT リストの AS 句を使用して明示的な名前を指定するか、*column-name-list* を指定する必要があります。そうでない場合、エラーが返されます。
- 結合クエリの予測リストの 2 つ以上の列が同じ名前である場合は、これらの列に明示的な一意のエイリアス名を付けるか*列名リスト*指定する必要があります。そうでない場合、エラーが返されます。
- 参照整合性制約が設定されているテーブルに MV を作成することは推奨しません。
- MV では、SELECT 権限のみ付与できます。
- MV 定義クエリに*が含まれている場合、基礎となるテーブルの既存の列が MV 定義に含まれます。

注記:

MV の作成時に SQL/MX によって列やインデックスが追加されることがあります。SHOWDDL では、これらのインデックスおよび列が表示されます。MV の作成時、警告 12112 は追加のインデックスの作成を示します。

制約

- MV 定義のクエリで集約関数を指定することはできません。
- MV 定義クエリには、DISTINCT、GROUP BY、HAVING、または ORDER BY 句を指定することはできません。
- MV に UNION 句を指定することはできません。
- MV では、サブクエリは使用できません。
- MV 定義では、Sequence Generator 擬似列を指定することはできません。
- MV 定義では、クエリの予測リストの実テーブルの IDENTITY 列を指定することはできません。
- MV は、依存するトリガーのあるテーブルに定義することはできません。MV を含むテーブルにトリガーを作成することはできません。
- MV には、インデックス、トリガー、ビュー、およびその他の MV を作成できません。参照整合性の制約により MV 列は参照できません。
- 組み込まれた INSERT、UPDATE、および DELETE 操作は、依存 MV を持つテーブルでは許可されていません。
- PROTOTYPE 句と mxrpm ツールは、MV 上の類似性チェックをサポートしていません。
- MV がテーブルに存在する場合、テーブルのクラスタリングキーの自己参照更新は許可されていません。
- 実テーブルが依存 MV を含む場合、更新可能なビューから行を更新、挿入、または削除することはできません。

コマンド例

以下の文では、MV、hr_details が作成されます。

```
create table employee (e_empid int not null, empname varchar (50) not null,
address varchar (100) not null, contact varchar (15), primary key (e_empid));
create table salary (s_empid int not null, sal numeric (18, 6), dept varchar
(20), primary key (s_empid));
create materialized view hr_details
refresh on statement
initialize on create
as
select empname, address, sal from employee join salary on e_empid = s_empid
where dept = 'HR';
```

ALTER MATERIALIZED VIEW

構文

```
ALTER {MATERIALIZED VIEW | MV} mv-name mv-alter-action

mv-alter-action is:
     RENAME TO new-name
```

説明

この文は MV の名前を変更します。

オプション

mv-name

MV の現在の名前を指定します。*mv-name* は、*[catalog-name.][schema-name.]mv-name* 形式の一部で、ここで名前の各部分は有効な SQL 識別子です。最大 128 文字を指定できます。

RENAME TO

オブジェクトの ANSI 名を変更します。

new-name

VM の新しい名前を指定します *new-name* には、最大 128 文字を指定できます。

許可

MV を変更するにはスキーマを所有しているか Super ID を持っている、またはオブジェクトのオーナーであるか、MV が含まれるスキーマに対する ALTER 許可を持っている必要があります。

留意事項

新しい ANSI 名は、スキーマ内で一意でなければなりません。

コマンド例

```
>>ALTER MATERIALIZED VIEW CAT.SCH.HR_DETAILS RENAME TO HR_INFO;

--- SQL operation complete.
```

DROP MATERIALIZED VIEW

構文

```
DROP { MATERIALIZED VIEW | MV } mv-name
```

説明

この文は、MV を削除します。

オプション

mv-name

削除する MV の現在の名前を指定します。*mv-name* は、*[catalog-name.][schema-name.]mv-name* 形式の一部で、ここで名前の各部分は有効な SQL 識別子です。最大 128 文字を指定できます。

許可

MV を削除するにはスキーマを所有しているか Super ID を持っている、またはオブジェクトのオーナーであるか、MV が含まれるスキーマに対する DROP 許可を持っている必要があります。

コマンド例

```
>>DROP MATERIALIZED VIEW HR_DETAILS;

--- SQL operation complete.
```

MERGE 文

MERGE は ANSI 2003 標準文です。一連のソースレコードとターゲットテーブルを指定すると、MERGE 文によって、効率的な方法でソースレコードがターゲットテーブルのデータにマージされます。ソースの行と一致するターゲットテーブルの行は更新され、一致しない行はターゲットテーブルに挿入されます。ソースレコードはクエリから取得できます。

ユーザーにはターゲットテーブルに対するオブジェクトの INSERT および UPDATE 権限、およびソースターゲットに対するオブジェクトの READ または SELECT 権限が必要です。DELETE 句は、SQL/MX リリース 3.5 でサポートされていません。以下はターゲットにすることはできません。

- トリガー、RI 制約、マテリアライズドビュー、ユーザー定義 ID 列、シーケンスジェネレーターオブジェクトを持つテーブル。
- ビューやサブクエリのような、導出されたテーブル。
- SQL/MP のテーブルおよびエイリアス。

MERGE 構文

構文

```
MERGE INTO table [ [AS] corr ]
    [using-clause]
    on-clause
    {[when-matched-clause] | [when-not-matched-clause]}
    [[FOR] access-option ACCESS]
    [WITH control-query-default control-list]
    [statement-level-hint-list]

using-clause is:
    USING (select-statement) [AS] corr
```

```
 [col-expr-list]

on-clause is:
    ON search-condition

when-matched-clause is:
    WHEN MATCHED THEN UPDATE SET set-clause [, set-clause]...

set-clause is:
    column-name = {expression||NULL}

when-not-matched-clause is:
    WHEN NOT MATCHED THEN INSERT insert-values-list

insert-values-list is:
    [(column1, ..., columnN)] VALUES (value1, ..., valueN)

access-option is:
    READ COMMITTED

control-query-default is:
    CONTROL QUERY DEFAULT

control-list is:
    attribute 'attr-value'
    | attribute 'attr-value' [, control-list]...

statement-level-hint-list is:
    USING <<+ hint-list >>

hint-list is:
    hint [, hint-list]

hint is:
    statement-hint
    | table-hint

statement-hint is:
    join-type-hint

table-hint is:
    access-path-hint
    | cardinality-hint
    | selectivity-hint

join-type-hint is:
    USE_HASHJOIN (table-1 [,table-2])
    | NO_HASHJOIN (table-1 [,table-2])
    | USE_MERGEJOIN (table-1 [,table-2])
    | NO_MERGEJOIN (table-1 [,table-2])
    | USE_NESTEDJOIN (table-1 [,table-2])
    | NO_NESTEDJOIN (table-1 [,table-2])

access-path-hint is:
    BASETABLE table
    | NO_BASETABLE table
    | INDEX table( index-list )
```

```
    | NO_INDEX table( index-list )
    | INDEXJOIN table( index-list )
    | NO_INDEXJOIN table( index-list )

index-list is:
    index [, index]...

cardinality-hint is:
    CARDINALITY table( cardinality-value )

selectivity-hint is:
    SELECTIVITY table( selectivity-value )
```

パラメーター

table

MERGE 操作が実行されるターゲットテーブルです。テーブル名は ANSI 論理名でなければなりません。

corr

テーブルの相関名です。

using-clause

SELECT 文です。この文の出力は MERGE 文のソースとして使用されます。

SELECT 文は、

- GROUP BY 句を含むことができます。
- ORDER BY 句を含むことはできません。
- EMBEDDED UPDATE 文や DELETE 文にすることができません。
- STREAM ACCESS 句や SET ON ROLLBACK 句を使用できません。

USING 句で指定されたサブクエリの相関名は必須です。

col-expr-list

導出された列のリストです。

search-condition

検索条件です。この検索条件に対して MERGE 文は行を更新したり挿入します。INSERT 句が指定されていると、検索条件はテーブルのクラスタリングキーで述部にする必要があります。INSERT 句が MERGE 文にない場合、検索条件は非キー列に指定できます。

- 述部がすべてのデータタイプと文字セットの列でサポートします。式も述部でサポートされます。
- SYSKEY の述部が MERGE 文の ON 句では許可されません。

when-matched-clause

ON 句の検索条件が TRUE になると、ターゲットテーブルで 1 つ以上の列を更新します 。WHEN MATCHED 句は、UPDATE 句であるとも見なされます。

WHERE 句や DELETE 句を WHEN MATCHED 句で指定することはできません。

set-clause

ターゲットテーブルの特定の列に値を関連付けます。その列をクラスタリングキーの一部にすることはできません。各 set-clause で、更新ソースの expression の値（または NULL）は、指定したターゲットの column-name の値に設定されます。ターゲットの各列のデータタイプが、ソース値のデータタイプと互換性がある必要があります。

注記:

UPDATE 文の SET 句はサブクエリを含むことができません。

when-not-matched-clause

ON 条件が FALSE になると、1 つまたは複数のレコードをターゲットテーブルに挿入します。INSERT キーワードの後の列リストが指定されていない場合、ターゲットテーブルの列の数は VALUES 句の値の数と一致しなければなりません。WHEN NOT MATCHED 句は、INSERT 句であるとも見なされます。

注記:

ON 句と VALUES 句で指定されたキー値は同じでなければなりません。

MERGE 文には、オプティマイザーのヒントとインライン CQD を含めることができます。オプティマイザーのヒントとインライン CQD パラメーターの説明については、*SQL/MX 3.4 リファレンス マニュアル* を参照してください。

留意事項

MERGE および MATCHED は、ANSI 予約済みのキーワードです。

必要な権限

MERGE 文を実行するためには、ユーザーに以下が必要です。

- ソーステーブルでの SELECT 権限
- ターゲットテーブルでの INSERT および UPDATE 権限

トランザクションの開始と終了

- MERGE は 1 つの文であるため、UPDATE と INSERT はどちらも同じトランザクションの下で作動します。いずれかの操作が失敗した場合、両方の操作は AUTOCOMMIT が ON のときロールバックされます。
- MERGE 文がユーザートランザクション内で失敗すると、トランザクション全体が中断します。

Reporting(レポート)

MERGE 文が正常に実行されると、SQL/MX は、挿入された行と更新された行の数を別々に報告するのでなく、マージした行として合計を報告します。同じ行を複数回更新した場合、各更新インスタンスは、マージした行の合計数に追加されます。

ON 述部

INSERT 句がある場合、ON 述部はターゲットテーブルからすべてのクラスタリングキー列を参照して、1 つの行だけが選択されるようにしなければなりません。次の例を考えてみましょう。

```
create table table1 (a1 int not null,  a2 int not null, a3 int not null, a4
int, primary key (a1,a2,a3));
create table table2 (a1 int not null, a2 int, a3 int, a4 int, primary key
(a1));
insert into table2 values (1,2,3), (4,5,6);
merge into table1 A using (select * from table2) B ON (A.a1=B.a1 and
A.a2=B.a2 and A.a3=B.a3) when not matched then insert values (B.a1, B.a2,
B.a3, 8);
```

MERGE 文では、すべてのプライマリキー列が ON 句で指定されているため、タプル（1、2、3、8）と（4、5、6、8）をテーブル 1 に挿入します。ただし以下の文はプライマリキー列 a3 が ON 述部にないため、SQL/MX コンパイラはエラーを返します。

```
merge into table1 A using (select * from table2) B ON (A.a1=B.a1 and
A.a2=B.a2)
when not matched then insert values (B.a1, B.a2, B.a3, 8);

*** ERROR[4504] Non-unique ON clause cannot be specified with an INSERT
clause.
*** ERROR[8822] The statement was not prepared.
```

注記:

テーブルにプライマリキー制約を追加したのがテーブルの作成中ではなく、ALTER TABLE 文を使用して追加した場合、クラスタリングキーは SYSKEY です。プライマリキー制約で指定された `key-column-list` ではありません。このような表では、すべてのプライマリキー列に等価述語が定義されていても、ON 句は常に一意ではありません。したがって、INSERT 句との MERGE を、プライマリキー制約が追加されたテーブルで発行することはできません。

MERGE 文は、ソース行とターゲットテーブルの行の結合を行ないます。ON 述部で複数の行を選択していると、ターゲット行は、次の例に示すように複数回更新される場合があります。

例

次の例では、tabTwo の各行で、tabOne のすべての行は 1 回更新されます。tabTwo には行が 5 つあるため、tabOne の各行は 5 回更新されます。

```
create table tabOne (a int , b int, c int not null, d int default 50 not
null, primary key (c,d));
create index IndexOneTabOne on tabOne(a) ;
create index IndexTwoTabOne on tabOne(b) ;

create table tabTwo (a int , b int, c int not null, d int default 50 not
null, primary key (c,d));
create index IndexOneTabTwo on tabTwo(c);

insert into tabOne Values (10,20,30,40);
insert into tabOne Values (10,20,30,50);
insert into tabOne Values (10,20,40,50);

insert into tabTwo Values (10,200,30,40);
insert into tabTwo Values (10,200,30,50);
insert into tabTwo Values (10,200,40,50);
insert into tabTwo Values (10,200,100,100);
insert into tabTwo Values (10,200,200,200);

>>select * from tabOne;

A            B            C            D
-----------  -----------  -----------  -----------

         10           20           30           40
         10           20           30           50
         10           20           40           50

--- 3 row(s) selected.
```

```
>> MERGE INTO tabOne
+> USING (select * from tabTwo) tabTwo
+> ON (tabOne.c=tabOne.c AND tabOne.d=tabOne.d)
+> WHEN MATCHED THEN UPDATE SET a=tabOne.a*10;

--- 15 row(s) merged.
>>select * from tabOne;

A            B            C            D
-----------  -----------  -----------  -----------

    1000000           20           30           40
    1000000           20           30           50
    1000000           20           40           50

--- 3 row(s) selected.
```

オプティマイザーのヒントとインライン CQD

- MERGE 文と、USING 句で指定されたサブクエリの両方で、すべてのオプティマイザーのヒントとインライン CQD を、MERGE 文の最後で指定する必要があります。
- USING 句で指定されたサブクエリ内の個々のテーブル参照のテーブルヒントを、テーブルの参照の横で指定できます。

データタイプ変換

既存の NonStop SQL/MX 変換規則を使用して、MERGE 文で指定された定数、動的パラメーター、およびホスト変数を変換します。

- ON 句述部で指定された定数、動的パラメーター、ホスト変数は、変換のターゲット列データタイプに変換されません。
- MERGE 文の UPDATE および INSERT 句で指定された定数、動的パラメーター、およびホスト変数は、値が更新または挿入される前に、ターゲット列データタイプに変換されます。

例

この MERGE 文では、120 の値がタイプ NUMERIC (4)の列 EMPNUM に、変換や切り捨ての後で挿入されます。

```
>>select * from employee;
--- 0 row(s) selected.

>> MERGE INTO employee
+>   ON empnum = 120
+>   WHEN MATCHED THEN UPDATE SET jobcode = 1000
+>   WHEN NOT MATCHED THEN
+>       INSERT VALUES (120.54, 'TIM', 'WALKER', 3000, 1000, 32000.00);

--- 1 row(s) merged.
>>select * from employee;

Employee/Number  First Name       Last Name            Dept/Num  Job/Code
Salary
---------------  ---------------  -------------------  --------  --------
-----------

            120  TIM              WALKER                   3000
1000     32000.00
```

この MERGE 文では、タイプ NUMERIC(4)の列 jobcode が、変換や切り捨ての後で、1234 の値で更新されます。

```
>> MERGE INTO employee
+>      ON empnum = 120
+>    WHEN MATCHED THEN UPDATE SET jobcode = 1234.54;

--- 1 row(s) merged.
>> select * from employee;

Employee/Number  First Name       Last Name              Dept/Num  Job/Code
Salary
---------------  ---------------  --------------------  --------  --------
-----------

            120  TIM              WALKER                     3000
1234     32000.00

--- 1 row(s) selected.
```

この MERGE 文では、ON 句の述部は、120.54 の値の変換または切り捨てにより評価されます。EMPNUM 120.54 のテーブル employee に行がないため、述部は FALSE になり、MERGE 文の INSERT 句が評価されます。INSERT 句は値 120 を切り捨ての後で挿入しようとし、MERGE は一意制約違反エラーで失敗します。

```
>>MERGE INTO employee
+> ON empnum = 120.54
+> WHEN MATCHED THEN UPDATE SET jobcode = 1000
+> WHEN NOT MATCHED THEN
+> INSERT VALUES (120.54, 'TIM', 'WALKER', 3000, 1000, 32000.00);

*** ERROR[8102] The operation is prevented by a primary key
EMPLOYEE_172678469_2978 on table UPSERT.SCH.EMPLOYEE.

--- 0 row(s) merged.
```

制限事項

注記:

制限事項については、ON 述部の項の例で指定された値を持つテーブル tabOne と tabTwo を検討してください。

- MERGE 文のターゲットテーブルを、以下にすることはできません。
  - トリガー、RI 制約、マテリアライズドビュー、ユーザー定義 ID 列、シーケンスジェネレーターオブジェクトを持つテーブル
  - ビューやサブクエリのような、導出されたテーブル
  - MP テーブルおよび MP エイリアス
- シーケンスジェネレーターオブジェクトを、MERGE 文の USING、ON、UPDATE、INSERT のいずれの句でも参照することはできません。
- SET ON ROLLBACK 句が MERGE 文では許可されていません。
- ON 句と VALUE 句で指定されたキー値は同じでなければなりません。以下の文は許可されていません。

```
MERGE INTO tabOne ON c = 30 and d = 40
    WHEN NOT MATCHED THEN INSERT VALUES (10, 20, 40, 50);
```

- ON 句にサブクエリを含めることはできません。以下の文は許可されていません。

```
MERGE INTO tabOne ON c = (SELECT a FROM tabTwo) WHEN...
```

- MERGE 文の UPDATE SET 句にサブクエリを含めることはできません。以下の文は許可されていません。

```
MERGE INTO tabOne ON c = 30 and d = 40
    WHEN MATCHED THEN UPDATE SET b = (select b from tabTwo);
```

- WHERE 条件が UPDATE および INSERT 句ではサポートされません。以下の文はサポートされていません。

```
>> MERGE INTO tabOne c = 30 and d = 40
+> WHEN MATCHED THEN UPDATE SET b = 2 WHERE b > 0
+> WHEN NOT MATCHED THEN INSERT VALUES (1, 2);
*** ERROR [15001] A syntax error occurred at or before:
MERGE INTO tabOne c = 30 and d = 40 WHEN MATCHED THEN UPDATE SET b = 2
WHERE b > 0 WHEN NOT
^ (63 characters from start of SQL statement)
*** ERROR [8822] The statement was not prepared.
```

  >> MERGE INTO tabOne c = 30 and d = 40

- ON 句の定数フィルター述部はサポートされていません。以下の文はサポートされていません。

```
>> MERGE INTO tabOne ON 0 = 1 -- constant filter predicate
+> WHEN MATCHED THEN UPDATE SET b = 2 WHERE b > 0
+> WHEN NOT MATCHED THEN INSERT VALUES (1, 2);
*** ERROR [3241] This MERGE statement is not supported.
Reason: Non-unique ON clause not allowed with INSERT.
```

- MERGE 文の INSERT VALUES 句にサブクエリを含めることはできません。以下の文は許可されていません。

```
MERGE INTO tabOne ON c = 30 and d = 40
WHEN MATCHED THEN INSERT values (10, select 20 from tabTwo, 30, 40);
```

- 更新している列をクラスタリングキーの一部にすることはできません。
- MFC、ROWSETS、およびノンアトミックの ROWSETS が MERGE 文ではサポートされていません。
- 組み込まれた UPDATE や DELETE、ストリームアクセスでは MERGE がサポートされていません。
- DP2 セーブポイントが MERGE 文では無効です。

MERGE 文の例

custnum = 1177 を持つ行が存在しない場合は、新しい行（1177、'ARROTECH'、'192 16TH ST.'、'BANGALORE'、'KARNATAKA'、'560048'、'F2'）を挿入します。

```
MERGE INTO customer ON custnum = 1177
WHEN NOT MATCHED THEN
    INSERT VALUES( 1177, 'ARROTECH', '192 16TH ST.',
                   'BANGALORE', 'KARNATAKA', '560048', 'F2');

--- 1 row(s) merged.

>>select * from customer;

Cust/Num  Customer Name       Street                  City
State         Post Code   CR
--------  ------------------  ----------------------  --------------
```

```
------------  ----------  --

    1177  ARROTECH             192 16TH ST.            BANGALORE
KARNATAKA     560048      F2

--- 1 row(s) selected.
```

キー custnum = 1177 を持つ行が存在する場合は、列 custname を‘ARROTECH’に更新します。この行が存在しない場合は、新しい行（1177、'ARROTECH'、'192 16TH ST.'、'BANGALORE'、'KARNATAKA'、'560048'、'F2'）が挿入されます。

```
MERGE INTO customer ON custnum = 1177
WHEN MATCHED THEN UPDATE SET custname = 'ARROTECH'
WHEN NOT MATCHED THEN
    INSERT VALUES (1177, 'ARROTECH', '192 16TH ST.’,
                   'BANGALORE', 'KARNATAKA', '560048', 'F2');

--- 1 row(s) merged.
```

custnum = 1177 を持つ行が存在する場合、次の MERGE 文は列 custname を‘ARRO’に更新します。

```
>>select * from customer;

Cust/Num  Customer Name       Street                  City
State         Post Code   CR
--------  ------------------  ----------------------  --------------
------------  ----------  --

    1177  ARROTECH            192 16TH ST.            BANGALORE
KARNATAKA     560048      F2

--- 1 row(s) selected.

MERGE INTO customer ON custnum = 1177
WHEN MATCHED THEN UPDATE SET custname = 'ARRO';

--- 1 row(s) merged.

>>select * from customer;

Cust/Num  Customer Name       Street                  City
State         Post Code   CR
--------  ------------------  ----------------------  --------------
------------  ----------  --

    1177  ARRO                192 16TH ST.            BANGALORE
KARNATAKA     560048      F2

--- 1 row(s) selected.
```

empnum = 120 を持つ行が存在しない場合、次の MERGE 文は新しい行（120、'TIM'、'WALKER'、3000、job.jobcode、32000.00）を employee テーブルに挿入します。empnum = 120 を持つ行が存在する場合、この行の jobcode は job テーブルの jobcode とマージされます。

```
MERGE INTO employee USING (SELECT jobcode FROM job) job
ON empnum = 120
WHEN MATCHED THEN UPDATE SET jobcode = job.jobcode
WHEN NOT MATCHED THEN
INSERT VALUES (120, 'TIM', 'WALKER', 3000, job.jobcode, 32000.00);
--- 1 row(s) merged.
```

次の MERGE 文は、動的パラメーターを INSERT 句で使用します。MERGE 文は正常にコンパイルします。しかしこの文は、動的パラメーターの値に基づいて、正常に実行される場合とされない場合があります。

```
>>prepare xx from
+>MERGE INTO employee
+>ON empnum = ?
+>WHEN MATCHED THEN UPDATE SET jobcode = 1000
+>WHEN NOT MATCHED THEN
+>INSERT VALUES (?, 'TIM', 'WALKER', 3000, 1000, 32000.00);

--- SQL command prepared.
>>
>>execute xx using 121,120;

*** ERROR[8500] Key values specified in the INSERT part of a MERGE statement
must be the same as those specified in the ON clause.

--- 0 row(s) merged.
>>
>>execute xx using 120,121;

*** ERROR[8500] Key values specified in the INSERT part of a MERGE statement
must be the same as those specified in the ON clause.

--- 0 row(s) merged.
>>
>>execute xx using 120,120;

--- 1 row(s) merged.
```

C 言語の例

```
EXEC SQL MERGE INTO employee
USING (SELECT jobcode into :outparam1 FROM job) job
ON empnum = 120
WHEN MATCHED THEN UPDATE SET jobcode = job.jobcode
WHEN NOT MATCHED THEN
INSERT VALUES (120, 'TIM', 'WALKER', 3000, 300, 32000.00);
END-EXEC;
```

COBOL 言語の例

```
EXEC SQL MERGE INTO employee
USING (SELECT jobcode into :outparam1 FROM job) job
ON empnum = 120
WHEN MATCHED THEN UPDATE SET jobcode = job.jobcode
WHEN NOT MATCHED THEN
INSERT VALUES (120, 'TIM', 'WALKER', 3000, 300, 32000.00);
END-EXEC;
```

MERGE セマンティックス

MERGE 文は、一致する行をソースクエリまたはテーブルからターゲットテーブルにマージする DML 文です。ソースクエリまたはテーブルの行がターゲットテーブルに存在しない場合、MERGE によってターゲットテーブルに行が挿入されます。

MERGE 文は以下を含みます。

- ソースクエリまたはテーブル。ソースクエリがある場合、USING 句とも呼ばれます。
- ターゲットテーブル
- ソースから行を評価する ON 述部。
- UPDATE または INSERT 句、あるいはその両方。ソースからの特定の行について ON 述部が TRUE まで評価する場合、その行に対して UPDATE 句が実行されます。そうでない場合、INSERT 句が実行されます（ターゲットテーブルに行が挿入されます）。
- MERGE 構文では、INSERT 句は WHEN NOT MATCHED THEN INSERT として指定されます。UPDATE 句は WHEN MATCHED THEN UPDATE として指定されます。
- UPDATE 句は 1 つまたは複数の列を参照することがあります。これらの列は、クラスタリングキーの一部であってはなりません。

MERGE 文に INSERT 句があれば、ON 述部は 1 つの行を選択する必要があります（ターゲットテーブルのクラスタリングキーに存在するすべての列において検索条件は述部でなければなりません）。UPDATE 句のみある場合（INSERT 指定なし）、ON 述部では複数の行を選択できます（検索条件は非キー列に指定できます）。

セマンティックスの制限事項は次のとおりです。

- ターゲットテーブルを、ビュー、マテリアライズドビュー、またはシーケンスジェネレーターとすることはできません。
- シーケンスジェネレーターは、いずれの句でも使用できません。
- ターゲットテーブルがトリガー、マテリアライズドビュー、または参照整合性制約を含む場合、MERGE は許可されません。
- USING 句の MERGE 文と SELECT 文には、SET ON ROLLBACK、埋め込み更新または削除、およびストリームアクセスは指定できません。
- ON 句で指定したキー値は、INSERT 句で指定したキー値と一致する必要があります。
- ON 句、WHEN MATCHED 句、および WHEN NOT MATCHED 句にはサブクエリを含めることはできません。
- クラスタリングキーの列の更新はサポートされません。

PL/MX とユーザー定義関数

NonStop SQL/MX の手続き型言語（PL/MX）は、Oracle PL/SQL や ANSI SQL/PSM と多くの点で類似した手続き型言語です。ユーザー定義関数（UDF）を PL/MX で作成、使用、および管理することができます。UDF は、データベースの一部となるカスタムビジネスロジックを実装する方法を提供します。他のプログラミング言語の関数と同じように、UDF はパラメーターを受け取ってスカラー値を返します。UDF は、ソースコードとオブジェクトコードの両方がデータベース内で完全に保存および管理されるという点で、 真のストアドファンクションです。そのため、UDF はストアドファンクションと呼ばれることもあります。

PL/MX と PL/MX を使用したプログラミング言語について詳しくは、*SQL/MX 3.5 Procedural Language for NonStop SQL/MX (PL/MX) Reference Manual* を参照してください。

データベースサービス

データベースサービス（DBS）では、ユーザーデータベースをクラウドベースの環境で作成して管理できます。NonStop サーバー上でホストされているユーザーデータベースは、ユーザーの分離、リソースの分離、メタデータの分離などの分離機能を使用して互いに分離されます。ユーザーデータベースにアクセスするユーザーまたはアプリケーションは、データベースに関連付けられているこれらのオブジェクトとリソースのみにアクセスできます。

ユーザーデータベースに必要なリソースで DBS 環境を構成するために新しいインストールスクリプトが追加されています。mxdbs という新しい OS コマンドラインツールは、ユーザーデータベースの作成や管理に使用できます。mxdbs ツールは OSS コマンド行から実行できます。または SQL/MX データベースにアクセスするエンドユーザーに対してセルフサービスモデルを提供する管理ツールに統合できます。

DBS 機能およびマルチテナント機能について詳しくは、*SQL/MX 3.5 データベースサービス マニュアル*を参照してください。

ユーザー管理

ユーザー管理は、クラウドベース環境で DBS 操作を行えるように SQL/MX で導入されました。ユーザー管理機能を使用すれば、ユーザーおよび権限グループを作成し管理できます。ユーザーは、Guardian ユーザー名、Guardian ユーザー ID、およびオプションの外部名を持ち、「権限グループ」と呼ばれる 1 つまたは複数のユーザーグループのメンバーになれます。データベースサービスはユーザー管理機能を使用します。

権限グループの主な目的は、オブジェクトとスキーマに対する権限の付与対象者として機能することです。特権グループに付与されたアクセス権限は、権限グループの現在および将来のすべてのメンバーに拡張されます。テーブルおよびビューに関する権限が権限グループに付与されていても、グループメンバーは、そのテーブルまたはビューに依存するオブジェクトを作成できません。

外部ユーザー名は SQL 識別子であり、これは Guardian ユーザー名の別名です。Safeguard エイリアスとは関係しておらず、MXCS 認証を除いて NonStop システムへのログオンには使用できません。Guardian ユーザー名とは 1 対 1 で関連付けられています。

ユーザー管理機能では次の操作を行えます。

- メタデータ内のオーナー、付与者、付与対象者の Guardian ユーザー名を表す。
- 外部ユーザー名を Guardian ユーザー名に関連付け、現在 Guardian ユーザー名を受け入れているすべてのコマンドでその名前を使用する。
- 権限グループと呼ばれるユーザーのグループを作成する。
- GRANT コマンドおよび REVOKE コマンドで付与対象者として権限グループを受け入れる。

ユーザー管理は次のように分類されます。

自動ユーザー管理

GRANT、GIVE、DDL、またはオブジェクトを暗黙的に作成するいずれかのユーティリティコマンドが実行されるときには常に、オブジェクトオーナーまたは付与対象者の情報がユーザー管理メタデータに格納されます。

Safeguard を使用するシステムに Guardian ユーザーを追加しても、自動的には SQL/MX ユーザー管理は呼び出されません。1 人または複数の Guardian ユーザーに対してユーザー管理を実行するには、CREATE USER コマンドを使用する必要があります。

明示的なユーザー管理

データベースプロバイダーが Guardian ユーザーに対して外部ユーザー名を追加したり削除したりできるようにします。Guardian ユーザーから外部名を削除すると、外部ユーザー名だけが削除されます。Guardian ユーザー情報はメタデータに残ります。

権限グループ管理

明示的な権限グループの管理を可能にします。

外部ユーザー名を受け入れるように拡張された文

以下の文が、ターゲットオーナーまたは付与対象者として外部ユーザー名を受け入れるように拡張されました。

- GIVE CATALOG
- GRANT CREATE CATALOG
- GRANT CREATE SCHEMA
- GIVE Object
- GIVE SCHEMA
- GRANT SECURITY_ADMIN
- REVOKE CREATE CATALOG

- REVOKE CREATE SCHEMA
- REVOKE SECURITY_ADMIN

以下のコマンドがユーザー管理機能用に追加されました。

- CREATE USER
- DROP USER
- CREATE PRIVILEGE GROUP
- ALTER PRIVILEGE GROUP
- GIVE PRIVILEGE GROUP
- DROP PRIVILEGE GROUP

CREATE USER

構文

外部のユーザー名を作成するには、次のコマンドを発行します。

```
CREATE USER external-user-name FOR guardian-user-name;
```

ユーザー管理のメタデータに 1 つまたは複数の Guardian ユーザー名を追加するには、次のコマンドを発行します。

```
CREATE USER FOR (guardian-user-name [, guardian-user-name] ...);
```

説明

Guardian ユーザーの外部ユーザー名を作成したり、ユーザー管理のメタデータに Guardian ユーザー名を追加したりするには、CREATE USER 文を使用します。Guardian ユーザー名には 1 つだけ外部ユーザー名を関連付けることができます。ユーザー情報は、メタデータに格納されます。このコマンドを実行した後は、GRANT、REVOKE、または GIVE ステートメントで Guardian ユーザー名の代わりに外部ユーザー名を使用することができます。システムカタログのメタデータは、3500 以上である必要があります。

パラメーター

external-user-name

Guardian ユーザーの外部名です。

guardian-user-name

Guardian のユーザー名です。

注記:

SQL/MX オブジェクトへのアクセスは、ANSI/ISO/IEC 9075:1999 SQL 標準（SQL:1999）の規則に従います。SQL:1999 では、SQL 文の処理中に認証 ID を使用してユーザーを識別します。SQL/MX の認証 ID は、二重引用符で囲まれた有効な Guardian ユーザー名か、特別な認証 ID である PUBLIC または DB_PUBLIC のいずれかです。

ノードへのログオンを許可された各ユーザーは、グループとユーザーの識別から構成される Guardian ユーザー ID によって識別されます。ユーザー ID の形式は、*group_number,user_number* または *group_name.user_name* のいずれかです。

留意事項

- 外部ユーザー名について

- 外部ユーザー名の形式は、有効な Guardian ユーザー名や、または特別な認証 ID である PUBLIC、DB_PUBLIC、または SYSTEM のいずれかと同じにすることはできません。
- 外部ユーザー名にスペースを含めることはできません。
- Guardian ユーザー名には 1 つだけ外部ユーザー名を関連付けることができます。

権限の要件

Guaridan ユーザーの外部ユーザー名を作成するには、セキュリティ管理者であるか、または Super ID（Super ID がセキュリティ管理者のグループの一部である場合、またはセキュリティ管理者のグループが存在しない場合）を持つ必要があります。

コマンド例

- 外部ユーザー名を作成し、Guardian ユーザー名に関連付けるには、次のコマンドを発行します。

```
CREATE USER "EMEA\Customer" FOR "DBUSERS.ADMIN";
```

- ユーザー管理メタデータに Guardian ユーザー名を追加するには、次のコマンドを発行します。

```
CREATE USER FOR ("SALES.JANE", "PAYROLL.JOE");
```

DROP USER

構文

```
DROP USER external-user-name;
```

説明

DROP USER 文では、Guardian ユーザー名に関連付けられている外部のユーザー名を削除します。ただし、Guardian ユーザー情報はメタデータに残ります。

パラメーター

external-user-name

削除する外部ユーザー名です。

留意事項

システムカタログのメタデータは、3500 以上である必要があります。

権限の要件

外部ユーザー名を削除するには、セキュリティ管理者、または Super ID（Super ID がセキュリティ管理者のグループに属する場合、またはセキュリティ管理者のグループが存在しない場合）でなければなりません。

コマンド例

次のコマンドは、外部ユーザー名を削除します。

```
DROP USER "EMEA\Customer";
```

このコマンドは、対応する Guardian ユーザーを削除せず、権限グループのメンバーシップやその Guardian ユーザーが保持する権限に影響を与えることもありません。

CREATE PRIVILEGE GROUP

構文

```
CREATE PRIVILEGE GROUP pg-name [ ADD ( authid [, authid ] ... ) ];
```

説明

CREATE PRIVILEGE GROUP 文は、明示的な権限グループを作成し、必要に応じてグループメンバーとして 1 つまたは複数のユーザーを追加します。

パラメーター

pg-name

権限グループの名前で SQL 識別子。Guardian の有効なグループ名と同じ形式にすることはできません。

authid

二重引用符で囲まれた有効な既存 Guardian のユーザー名、または Guardian ユーザーのいずれかに関連付けられている外部ユーザー名を指定します。*authid* が Guardian ユーザー名である場合、大文字と小文字は区別されません。

留意事項

- システムカタログのメタデータは、3500 以上である必要があります。
- オプションの ADD 句を指定しないと、CREATE PRIVILEGE GROUP ではメンバーを持たない権限グループが作成され、作成したユーザーによって所有されます。
- 権限グループの作成中には、同じユーザー名を 2 回指定することはできません。

権限の要件

権限グループを作成するユーザーがセキュリティ管理者のメンバーである場合、ADD 句ではセキュリティ管理者のグループのメンバーを含めることができません。

コマンド例

- 権限グループを作成するには、次のコマンドを発行します。

```
CREATE PRIVILEGE GROUP temp_users;
```

- 権限グループを作成し、外部ユーザー名を持つユーザーを 1 人追加するには、次のコマンドを発行します。

```
CREATE PRIVILEGE GROUP "Database Admins"
  ADD ( "customer@hpe.com" );
```

ALTER PRIVILEGE GROUP

構文

```
ALTER PRIVILEGE GROUP pg-name alter-members;

alter-members is: { add-members | delete-members }

add-members is: ADD ( authid [, authid ] ... )

delete-members is: DELETE ( authid [, authid ] ... )
```

説明

ALTER PREVILEGE GROUP 文は、明示的な権限グループから 1 つ以上のメンバーを追加または削除します。

パラメーター

pg-name

既存の明示的な権限グループである SQL 識別子の名前です。Guardian の有効なグループ名と同じ形式にはできません。

alter-members

ADD オプションまたは DELETE オプションを使用してメンバーを権限グループから追加または削除する必要があるかどうかを指定します。

authid

二重引用符で囲まれた有効な既存 Guardian ユーザー名、または既に Guardian ユーザーのいずれかに関連付けられている外部ユーザー名を指定します。*authid* が Guardian ユーザー名である場合、大文字小文字の区別はありません。

留意事項

- システムカタログのメタデータは、3500 以上である必要があります。
- 権限グループにメンバーを追加する際に、同じユーザー名を 2 回指定したり、既に権限グループのメンバーであるユーザーを指定したりすることはできません。
- 追加されたメンバーは、権限グループに付与されるアクセス権を取得します。権限グループからメンバーを削除すると、グループのアクセス権が取り消されます。

認証の要件

- 権限グループを変更するには、次のいずれかである必要があります。
  - 権限グループのオーナー
  - セキュリティ管理者
  - Super ID（Super ID がセキュリティ管理者のグループの一部である場合、またはセキュリティ管理者のグループが存在しない場合）
- 権限グループを変更するユーザーがセキュリティ管理者のグループのメンバーである場合、セキュリティ管理者のグループのメンバーは追加できません。

コマンド例

- Guardian ユーザーを権限グループに追加するには、次のコマンドを実行します。

```
ALTER PRIVILEGE GROUP temp_users ADD ( "USER1", "USER2" );
```

- ユーザーを権限グループから削除するには、次のコマンドを実行します。

```
ALTER PRIVILEGE GROUP temp_users DELETE ( "USER2" );
```

GIVE PRIVILEGE GROUP

構文

```
GIVE PRIVILEGE GROUP pg-name TO authid;
```

説明

GIVE PRIVILEGE GROUP 文は、権限グループの所有権を別のユーザーに転送します。

パラメーター

pg-name

既存の明示的な権限グループの SQL 識別子の名前です。Guardian の有効なグループ名、および暗黙的権限グループ名と同じ形式にはできません。

authid

二重引用符で囲まれた有効な Guardian ユーザー名、または既に Guardian ユーザーのいずれかに関連付けられている外部ユーザー名を指定します。*authid* が Guardian ユーザー名である場合、大文字小文字の区別はありません。

留意事項

- システムカタログのメタデータは、3500 以上である必要があります。
- 権限グループの所有権はシステム上の任意の既存のユーザーに転送できます。
- 権限グループに付与された既存の権限は GIVE PRIVILEGE GROUP コマンドの影響を受けません。

権限の要件

- GIVE PRIVILEGE GROUP を使用して所有権を転送するには、次のいずれかである必要があります。
  - 権限グループのオーナー
  - セキュリティ管理者
  - Super ID（Super ID がセキュリティ管理者のグループの一部である場合、またはセキュリティ管理者のグループが存在しない場合）
- 権限グループの所有権は、付与者が権限グループのオーナーである場合を除き、セキュリティ管理者のグループのメンバーに転送することはできません。

コマンド例

権限グループの所有権を転送するには、次のコマンドを発行します。

```
GIVE PRIVILEGE GROUP temp_users TO "customer@hpe.com";
```

DROP PRIVILEGE GROUP

構文

```
DROP PRIVILEGE GROUP pg-name [RESTRICT | CASCADE];
```

説明

DROP PRIVILEGE GROUP 文は、権限グループを削除します。

パラメーター

pg-name

既存の明示的な権限グループである SQL 識別子の名前です。Guardian の有効なグループ名、または暗黙的権限グループ名と同じ形式にはできません。

RESTRICT

このオプションを指定すると、権限グループは空である場合にのみ削除されます。これは、デフォルトのオプションです。

CASCADE

このオプションを指定すると、権限グループが削除され、メンバーが保持するグループ権が取り消されます。

留意事項

システムカタログのメタデータは、3500 以上である必要があります。

承認要件

- 権限グループを削除するには、次のいずれかである必要があります。
  - 権限グループのオーナー
  - セキュリティ管理者
  - Super ID（Super ID がセキュリティ管理者のグループの一部である場合、またはセキュリティ管理者のグループが存在しない場合）

コマンド例

メンバーを持たない権限グループを削除するには、以下のコマンドを発行します。

```
DROP PRIVILEGE GROUP temp_users;
```

メンバーを持つ、またはメンバーを持たない権限グループを削除するには、以下のコマンドを発行します。

```
DROP PRIVILEGE GROUP temp_users CASCADE;
```

スキーマレベルの権限

スキーマレベルの権限は、スキーマのオーナーが、スキーマ全体に対する権限をユーザーまたはユーザーグループに付与または取り消しできるようにします。

スキーマレベルの変更をサポートするために、以下の文が更新されます。

- **GRANT 文**
- **REVOKE 文**

注記:

本書で後述する GRANT 文および REVOKE 文のセクションには、これらの文の完全な構文、セマンティックス、および留意事項が記されています。

CREATE SCHEMA の変更

スキーマの作成中、WITH GRANT OPTION ですべての DDL 権限とすべての DML 権限がスキーマのオーナーに付与されます。これにより、スキーマオーナーはこれらの権限を他のユーザーに付与できるようになります。

マルチテナント環境では、CREATE SCHEMA 操作は、スキーマと関連付けられており、アクセスレベル（読み取り、読み取り/書き込み、および DLL を含むフルアクセス）を表す権限グループに、関連する権限を自動的に付与します。

GIVE SCHEMA の変更

GIVE SCHEMA 操作は、あるユーザーから別のユーザーにスキーマの所有権を転送するために使用されます。権限付与対象者タイプ'O'のオーナーが保持するスキーマに対する権限は、新しいオーナーに転送されます。この転送には DDL 権限が含まれます。

権限のタイプ

スキーマレベルの権限は、SQL/MX データ定義言語（DDL）とデータ操作言語（DML）の両方の権限をサポートします。

- DDL

  スキーマレベルの DDL 権限が付与されたユーザーは、そのスキーマと、そのスキーマ内の既存または将来のオブジェクトに DDL 操作を実行することができます。DDL 権限は、スキーマレベルでのみ適用されます。個々のオブジェクトに対する DDL 権限を付与したり取り消したりすることはできません。
- DML

  スキーマレベルの DML 権限は、個々のオブジェクトに対して付与できる種類の権限を、スキーマ全体に対して付与できるようにします。このような権限は、そのスキーマ内の既存または将来のオブジェクトに関連します。たとえば、スキーマに対する SELECT 権限をあるユーザーに付与した場合、そのユーザーはそのスキーマ内のオブジェクトから SELECT を実行できます。スキーマオーナーは、他のユーザーが所有しているスキーマ内のオブジェクトに対して暗黙的な DML 権限を保持していません。

ターゲットのタイプ

DDL 権限は、スキーマレベルでのみ適用されます。これらの権限を個々のオブジェクトに対して付与したり取り消したりすることはできません。DML 権限は、スキーマレベルとオブジェクトレベルの 2 つのレベルで付与できます。

権限付与対象者のタイプ

権限付与対象者のタイプには、オーナー、ユーザー、パブリックがあります。権限付与対象者にはユーザーを指定することも、権限グループを指定することもできます。

DB_PUBLIC

SQL/MX リリース 3.5 では、特殊な authid DB_PUBLIC が導入されました。DB_PUBLIC は、マルチテナント環境では、ユーザーデータベースに接続しているユーザーおよびアプリケーションに使用できます。マルチテナント環境外では、DB_PUBLIC は PUBLIC と同じです。

ユーザー名のタイプ

Guardian ユーザー名と外部ユーザー名の 2 種類のユーザー名があります。詳しくは、**ユーザー管理**を参照してください。

ユーティリティの承認要件

ユーティリティ	承認要件
CLEANUP	ターゲットオブジェクトを含むスキーマに対する ALTER 権限。
DUP	ソーステーブルまたはそのスキーマに対する SELECT 権限。ターゲットテーブルを含むスキーマに対する CREATE および ALTER 権限。
FASTCOPY	ソーステーブルまたはそのスキーマに対する SELECT 権限。ターゲットテーブルまたはそのスキーマに対する SELECT、INSERT、UPDATE、および DELETE 権限。
FIXUP	影響を受けるオブジェクトを含むスキーマに対する ALTER 権限。
IMPORT	ターゲットテーブルまたはそのスキーマに対する INSERT 権限。
MODIFY	影響を受けるオブジェクトを含むスキーマに対する ALTER 権限。
POPULATE INDEX	影響を受けるオブジェクトを含むスキーマに対する ALTER 権限。
PURGEDATA	ターゲットテーブルまたはそのスキーマに対する SELECT および DELETE 権限。
VERIFY オブジェクト	オブジェクトまたはそのスキーマに対する SELECT および EXECUTE 権限。
VERIFY スキーマ	スキーマに対する ALTER 権限。

GRANT および REVOKE の変更

GRANT および REVOKE 文は、ユーザー管理とスキーマレベルの権限をサポートするように変更されています。

注記:

GRANT 文および REVOKE 文のセクションには、これらの文の完全な構文、セマンティックス、および留意事項が記されています。

GRANT 文

GRANT 文は、SQL/MX のテーブル、ビュー、MV、スキーマ、シーケンスジェネレーター、またはストアドプロシージャーへのアクセス権を指定されたユーザーに付与します。

次を実行する権限を付与することもできます。

- DDL 権限を使用したスキーマレベルでの DDL 操作とユーティリティ操作。
- スキーマレベルでの DML 操作。そのような権限はスキーマ内の個々のオブジェクトに適用されます。

```
GRANT {privilege [,privilege]... | { ALL | ALL_DML | ALL_DDL } [PRIVILEGES]}
    ON target
    TO grantee [,grantee ]...
    [WITH GRANT OPTION]
    [BY authid-grantor]

grantee is:
   authid    |
   PUBLIC    |
   DB_PUBLIC |
   PRIVILEGE GROUP pg-name

privilege is:
     SELECT
   | DELETE
   | INSERT
   | UPDATE [(column [,column]...)]
   | REFERENCES [(column [,column]...)]
   | USAGE
   | EXECUTE
   | CREATE
   | ALTER
   | DROP

target is:
   { object-specification }
   { schema-specification }

object-specification is:
[ TABLE | SEQUENCE | PROCEDURE ]
object-name

schema-specification is:
   { SCHEMA schema-name }
```

GRANT の構文の説明

privilege **[,*privilege*]... | ALL [PRIVILEGES] | ALL_DML [PRIVILEGES] | ALL_DDL [PRIVILEGES]**

付与する権限を指定します。

テーブルまたはビューに対しては次の権限を指定できます。

SELECT	オブジェクトに対して SELECT 文を使用できます。
DELETE	オブジェクトに対して DELETE 文を使用できます。
INSERT	オブジェクトに対して INSERT 文を使用できます。
UPDATE	オブジェクトに対して UPDATE 文を使用できます。
REFERENCES	オブジェクトを参照する制約を作成できます。

シーケンスジェネレーターに対しては次の権限を指定できます。

USAGE	疑似列 CURRVAL および NEXTVAL を使用して、シーケンスジェネレーター値にアクセスできます。

ストアドプロシージャーまたは関数に対しては次の権限を指定できます。

EXECUTE	ストアドプロシージャーまたは関数を呼び出すことができます。

すべてのタイプのオブジェクトに対して次の権限を指定できます。

ALL PRIVILEGES	オブジェクトタイプに適用されるすべての権限を持つことができます。オブジェクトがテーブルまたはビューで、ALL を指定した場合は、SELECT、DELETE、INSERT、UPDATE、および REFERENCES 権限のみが適用されます。オブジェクトがストアドプロシージャーで、ALL を指定した場合は、EXECUTE 権限のみが適用されます。オブジェクトがシーケンスジェネレーターで、ALL を指定した場合は、USAGE 権限のみが適用されます。

テーブル、シーケンスジェネレーター、またはビューに対しては次の権限を指定できます。

ALL_DML	オブジェクトタイプに適用される DML 権限を持つことができます。スキーマの場合、ALL_DML にはすべての DML 権限が含まれます。

ターゲットがスキーマである場合にのみ、次の権限を指定できます。

CREATE	テーブル、インデックス、ビュー、トリガー、SPJ、シーケンスジェネレーターなどのオブジェクトをスキーマ内に作成できます。 作成したオブジェクトのオーナーは、CQD CREATE_AS_SCHEMA_OWNER の値に応じて、スキーマオーナーかクリエーターのどちらかになります。
ALTER	スキーマ内のオブジェクトの定義を変更できます。これには、MODIFY など、オブジェクトの定義を変更するユーティリティ操作を実行する機能も含まれます。
DROP	スキーマ自体を含む、スキーマのすべてのオブジェクトを削除できます。
ALL_DDL	すべての DDL 権限を保有できます。

(*column* [,*column*]...)

UPDATE 権限または REFERENCES 権限が適用されるテーブルまたはビューの列を指定します。UPDATE または REFERENCES を列名なしで指定した場合、権限はテーブルまたはビューのすべての列に適用されます。ターゲットがスキーマの場合、UPDATE および REFERENCES の列を指定できません。

ON [TABLE | SEQUENCE | PROCEDURE] *object*

権限を付与するテーブル、ビュー、シーケンスジェネレーター、またはストアドプロシージャーを指定します。

ON SCHEMA *schema-name*

権限を付与するスキーマの名前を指定します。

TO *grantee* [,*grantee*]...

権限を付与する 1 人以上のユーザーを指定します。grantee には、*authid* か、PUBLIC または DB_PUBLIC のいずれかの特殊な認証 ID を指定できます。

authid

認証 ID を指定します。これは次のいずれかにする必要があります。

- 有効な Guardian ユーザー名。二重引用符で囲みます。
- すでに Guardian ユーザーに関連付けられている外部ユーザー名。
- PUBLIC または DB_PUBLIC のいずれかの特殊な認証 ID。

pg-name

既存の権限グループである SQL 識別子の名前です。

WITH GRANT OPTION

権限が付与される認証 ID のユーザーが、同じ権限をほかの認証 ID に付与する権限を持つことを指定します。

BY *authid-grantor*

認証 ID *authid-grantor* を指定します。付与操作はこの認証 ID として実行されます。*authid-grantor* には SYSTEM を指定できません。

Super ID だけが *BY* 句を使用できますが、それは次のいずれかに該当する場合に限られます。

- セキュリティ管理者グループが空である。
- Super ID がセキュリティ管理者として指定されている。

BY 句を使用した場合の効果は、*authid-grantor* が GRANT を直接発行した（BY 句を使用せずに）場合と同じです。

セキュリティ管理者のグループが空の場合は、*authid-grantor* が有効な認証 ID であり、付与される権限を WITH GRANT OPTION 付きで保持している必要があります。

ただし、Super ID がセキュリティ管理者として指定されている場合は、*authid-grantor* を有効な認証 ID として設定できる強力な GRANT BY 機能を持つことになります。

GRANT に関する留意事項

- スキーマのターゲットの DML 権限は、以下に示す認証要件に従って、そのスキーマ内のすべての現在および将来のオブジェクトに適用されます。つまり、スキーマに対する権限の付与により、そのスキーマ内のすべての既存の該当するオブジェクトと、そのスキーマ内のすべての将来の該当するオブジェクトに対するアクセス権が与えられます。
- スキーマターゲットの DDL 権限を付与されたユーザーは、そのスキーマ内のオブジェクトに対する DDL 操作とユーティリティ操作を行うことができます。ただし、オブジェクトが作成されると、その存在は、そのスキーマに対する DDL 権限を保持するオーナーに依存しません。
- スキーマに付与された権限は、そのスキーマ内のオブジェクトごとに取り消すことができず、また、その逆もできません。
- ALL_DDL は、CREATE、ALTER、および DROP 権限の組み合わせです。ALL_DDL が与えられている場合は、個別に CREATE、ALTER、および DROP の権限を取り消すことができます。
- 各オブジェクトの ALL_DML は、次のようにオブジェクトの種類に応じて 1 つまたは複数の関連する権限に解決されます。
  - ターゲットがテーブルまたはビューである場合、SELECT、INSERT、UPDATE、DELETE、REFERENCES。
  - ターゲットがストアドプロシージャ—である場合、EXECUTE。
  - ターゲットが Sequence Genertor である場合、USAGE。
- スキーマ上の ALL_DML は、SELECT、INSERT、UPDATE、DELETE、REFERENCES、USAGE、および EXECUTE 権限の組み合わせに解決されます。オブジェクトの種類に応じて、スキーマ内のオブジェクトに関連する権限が適用されます。
- ALL [PRIVILEGES]は、ターゲットが個々のオブジェクトである場合は ALL_DML に相当し、ターゲットがスキーマである場合は ALL_DML と ALL_DDL の組み合わせに相当します。
- 権限グループに付与する場合、付与対象者はその特権グループ内の個々のユーザーではありません。

- 権限グループは、権限の付与者として機能できません。
- 権限グループの付与対象者は、WITH GRANT OPTION で使用できません。
- 権限グループへの付与によって、GRANT 時にメンバーであるユーザーや、グループメンバーとして追加されたメンバーを含む、権限グループのすべてのメンバーに対して、ターゲットオブジェクトへのアクセス権が与えられます。
- 権限グループ内のユーザーは、その権限グループから削除されるとき、および権限グループがドロップされるときに、その権限グループへのアクセス権を失います。

権限の要件

セキュリティ管理者または Super ID である場合を除き、オブジェクトまたはスキーマに権限を付与するには、その権限と、その権限を付与する権限の両方が必要です。つまり、権限が WITH GRANT OPTION で発行済みであり、取り消されていない必要があります。1 つ以上の特定の権限を付与する権限がない場合は、システムが警告を返します（さらに、指定された中で、付与する権限を持っている権限の付与を実行します）。指定された権限のいずれも WITH GRANT OPTION 付きでない場合、システムはエラーを返します。オブジェクトまたはスキーマのオーナーは WITH GRANT OPTION 付きのそのオブジェクトまたはスキーマに対して関連するすべての権限を自動的に保持します。これらの継承されたオーナー権限は、取り消しできません。

セキュリティ管理者である場合は、上記の制限から除外され、その権限がなくても権限を付与できます。ただし、そのような付与を PUBLIC またはセキュリティ管理者に対して行ったり、WITH GRANT OPTION を使用して行ったりすることはできません。セキュリティ管理者によるすべての付与について記録される付与者は、付与を実行するセキュリティ管理者の認証 ID です。これにより、トレーサビリティが確保され、権限を付与したセキュリティ管理者にまでさかのぼることができます。セキュリティ管理者はオーナー派生の WITH GRANT OPTION 権限を保持でき、その場合は他のユーザーのようにその権限を付与できます（PUBLIC への付与と WITH GRANT OPTION を使用した付与を含みます）。この後者のタイプの付与は、オーナー派生の付与の階層に含まれます。

Super ID である場合は、セキュリティ管理者のグループに応じて権限を付与することができます。セキュリティ管理者のグループが空の場合は、任意のオブジェクトに対して任意の権限を付与することができます。そのような付与は、 *GRANT BY authid-grantor* のように動作します。（ここで、*authid-grantor* はオブジェクトのオーナーです）。

Super ID がセキュリティ管理者として指定されている場合、Super ID は他のセキュリティ管理者と同じ権限を持つだけでなく、*GRANT BY authid-grantor* を実行できます。この場合、 *BY authid-grantor* を省略した場合の暗黙の付与者は他のセキュリティ管理者付与と同じです。つまり、暗黙の付与者はオブジェクトオーナーではなく実行しているセキュリティ管理者（Super ID）です。

スキーマレベルでの権限の GRANT および REVOKE に関する認証ルールは、オブジェクトレベルのものと同じです。セキュリティ管理者のグループが空ではなく、Super ID がセキュリティ管理者として指定されていない場合、GRANT 文に関しては通常のユーザーと同じ制限が Super ID に課されます。

スキーマに付与される DML 権限は、以下の場合に、そのスキーマの個々のオブジェクトに対するアクセス権として拡張されます。

- スキーマのオーナーがオブジェクトに対する WITH GRANT OPTION の権限を保持している。
- 付与者がセキュリティ管理者である。

REVOKE 文

REVOKE 文は、SQL/MX テーブル、ビュー、MV、スキーマ、シーケンスジェネレーター、またはストアドプロシージャーに対するアクセス権限を、指定したユーザーから取り消します。

```
REVOKE [GRANT OPTION FOR]
    {privilege [,privilege ]... | { ALL | ALL_DML | ALL_DDL } [PRIVILEGES]}
    ON target
    FROM grantee [,grantee ]... [drop-behavior ]
    [BY authid-grantor]

grantee is:
   authid | usernumber | PUBLIC

privilege is:
        SELECT
      | DELETE
      | INSERT
      | UPDATE [(column [,column ]...)]
      | REFERENCES [(column [,column ]...)]
      | USAGE
      | EXECUTE
      | CREATE
      | ALTER
      | DROP

grantee is:
   authid | usernumber | PUBLIC | DB_PUBLIC

target is:
   { object-specification }
   { schema-specification }

object-specification is:
[ TABLE | SEQUENCE | PROCEDURE ]
object-name

schema-specification is:
   { SCHEMA schema-name }

drop-behavior is:
   CASCADE | RESTRICT
```

REVOKE の構文の説明

GRANT OPTION FOR

WITH GRANT OPTION または権限を取り消すことを指定します。権限自体は取り消されません。

***privilege* [,*privilege*]... | ALL_DML [PRIVILEGES] | ALL_DDL [PRIVILEGES] | ALL [PRIVILEGES]**

取り消す権限を指定します。

テーブルまたはビューに対しては次の権限を指定できます。

SELECT	オブジェクトに対して SELECT 文を使用できません。
DELETE	オブジェクトに対して DELETE 文を使用できません。
INSERT	オブジェクトに対して INSERT 文を使用できません。
UPDATE	オブジェクトに対して UPDATE 文を使用できません。
REFERENCES	オブジェクトを参照する制約を作成できません。

シーケンスジェネレーターに対しては次の権限を指定できます。

USAGE	疑似列 CURRVAL および NEXTVAL を使用して、シーケンスジェネレーター値にアクセスできません。

ストアドプロシージャーまたは関数に対しては次の権限を指定できます。

EXECUTE	ストアドプロシージャまたは関数を呼び出すことはできません。

すべてのタイプのオブジェクトに対して次の権限を指定できます。

ALL PRIVILEGES	オブジェクトタイプに適用されるすべての権限を持つことはできません。 オブジェクトがテーブルまたはビューで、ALL を指定した場合は、SELECT、DELETE、INSERT、UPDATE、および REFERENCES 権限を意味します。オブジェクトがストアドプロシージャーで、ALL を指定した場合は、EXECUTE 権限のみを意味します。オブジェクトがシーケンスジェネレーターで、ALL を指定した場合は、USAGE 権限のみを意味します。

テーブルまたはビューに対しては次の権限を指定できます。

ALL_DML PRIVILEGES	すべての DML 権限を持つことはできません。

スキーマのオブジェクトに対しては次の権限を指定できます。

CREATE	スキーマのオブジェクトを作成する権限を権限付与対象者に付与できません。
ALTER	スキーマのオブジェクトの定義を変更する権限を権限付与対象者に付与できません。これには、MODIFY など、オブジェクトの定義を変更するユーティリティ操作を実行する機能も含まれます。
DROP	スキーマ自体を含む、スキーマのすべてのオブジェクトを削除する権限を権限付与対象者に付与できません。
ALL_DDL	オブジェクトのすべての DDL 権限を取り消す権限を権限付与対象者に付与できません。

(*column* [,*column*]...)

UPDATE 権限または REFERENCES 権限を取り消すテーブルまたはビューの列を指定します。UPDATE または REFERENCES を列名なしで指定した場合、テーブルまたはビューのすべての列の権限が取り消されます。

ON [TABLE | SEQUENCE | PROCEDURE] ***object***

権限を取り消すテーブル、ビュー、シーケンスジェネレーター、またはストアドプロシージャーを指定します。

ON SCHEMA ***schema-name***

権限を取り消すスキーマの名前を指定します。

FROM ***grantee*** **[,*grantee*]...**

権限を取り消す 1 人以上のユーザーを指定します。*grantee* には、*authid*、*usernumber*、または PUBLIC と DB_PUBLIC のいずれかの特殊な認証 ID を指定できます。

authid には、権限を取り消す認証 ID を指定します。認証 ID によって、SQL 文の処理中にユーザーが識別されます。認証 ID は、二重引用符で囲んだ有効な Guardian ユーザー名である必要があります（たとえば、"PAYROLL.HANS"）。*authid* は大文字と小文字を区別しません。

SQL:1999 の仕様では、PUBLIC と SYSTEM の 2 つの特別な認証 ID が定められています。

- PUBLIC は現在および将来のすべての認証 ID を指定します。
- SYSTEM は権限の暗黙的な付与者にオブジェクトの作成者を指定します。

REVOKE 文の *grantee* として SYSTEM を指定することはできません。

usernumber には、権限を取り消すユーザーを指定します。*usernumber* は、二重引用符で囲んだ正しい Guardian ユーザー番号（グループ,ユーザー）です（例："255,255"）。*usernumber* オプションは、USERID ファイルから削除されたユーザーから権限を取り消すための手段として提供されています。

drop-behavior

RESTRICT を指定し、指定された権限が削除された後に有効ではなくなる権限記述子またはオブジェクトがある場合、REVOKE 操作は失敗します。

CASCADE を指定した場合は、そのような従属する権限記述子およびオブジェクトが、REVOKE 操作の一部として削除されます。

デフォルトは RESTRICT です。

BY ***authid-grantor***

認証 ID *authid-grantor* を指定します。取消操作はこの認証 ID として実行されます。*authid-grantor* には SYSTEM を指定できません。*authid-grantor* のユーザー番号（例："151,18"）は許可されません。セキュリティ管理者または Super ID は *BY* 句を使用できますが、SUPER ID が *BY* 句を使用できるのは、次のいずれかに該当する場合に限られます。

- セキュリティ管理者グループが空である。
- Super ID がセキュリティ管理者として指定されている。

BY 句を使用した場合の効果は、*authid-grantor* が REVOKE を直接発行した（*BY* 句を使用せずに）場合と同じです。セキュリティ管理者のグループが空の場合は、*authid-grantor* が有効な認証 ID であり、取り消される 1 つ以上の権限を WITH GRANT OPTION 付きで保持している必要があります。セキュリティ管理者には強力な *REVOKE BY* 機能があり、以前にターゲットオブジェクトに権限を付与した有効な認証 ID を *authid-grantor* に指定できます。

authid

認証 ID を指定します。これは次のいずれかにする必要があります。

- 有効な Guardian ユーザー名。二重引用符で囲みます。
- すでにいずれかの Guardian ユーザーに関連付けられている外部ユーザー名。
- 特殊な *authid*、PUBLIC、または DB_PUBLIC。

pg-name

既存の権限グループである SQL 識別子の名前です。

REVOKE に関する留意事項

- オブジェクトのオーナーが保持している固有の権限（ALL WITH GRANT OPTION）は取り消すことはできません。特定のユーザーが所有するオブジェクトへのアクセスを拒否するには、そのオブジェクトの所有権を別のユーザーに付与する必要があります。
- 取り消している権限が 1 つ以上存在しない場合、システムは警告を返します。
- 権限グループのメンバーシップによりアクセス権を持つユーザーは、権限グループから削除されるとき、または権限グループがドロップされるときに、これらのアクセス権を失います。
- 個々のメンバーは、権限グループに付与された権限を取り消すことはできず、逆に権限グループは個々のメンバーの権限を取り消すことができません。
- スキーマのターゲットの DML 権限は、そのスキーマ内のすべての現在および将来のオブジェクトに適用されます。スキーマに対する権限が取り消されると、スキーマレベルの付与により付与対象者が保持している、そのスキーマ内のオブジェクトに対するこれらの対応するアクセス権が削除されます。
- 個別に付与される DDL 特権は、REVOKE の ALL_DDL を使用して取り消すことができ、逆も可能です。
- ALL [PRIVILEGES]は、ターゲットが個々のオブジェクトである場合は ALL_DML に相当し、ターゲットがスキーマである場合は ALL_DML と ALL_DDL の組み合わせに相当します。
- スキーマに付与される権限は、そのスキーマ内のオブジェクトごとに取り消すことはできず、その逆もできません。

権限の要件

セキュリティ管理者または Super ID ではない場合は、以前にユーザーに付与した権限のみを取り消すことができます。セキュリティ管理者である場合は、上記の制限から除外されます。また、権限を正常に取り消すと、セキュリティ管理者によって付与されたその権限のすべてのインスタンスが取り消されます。これは、オーナーに由来するその権限の付与には影響しません（つまり、オーナーに由来する付与の階層には影響しません）。ただし、セキュリティ管理者は *REVOKE BY authid-grantor* を使用してオーナーに由来する付与を取り消すことができます。 セキュリティ管理者はオーナーに由来する付与のターゲットである場合があるため、権限に由来する WITH GRANT OPTION を保持していることがあります。その場合は、通常のユーザーのようにその権限を取り消すことができます。

Super ID である場合は、セキュリティ管理者のグループに応じて権限を取り消すことができます。セキュリティ管理者のグループが空の場合は、任意のオブジェクトの任意の権限を取り消すことができます。そのような取り消しは、*REVOKE BY authid-grantor* のように動作します（ここで、*authid-grantor* はオブジェクトのオーナーです）。

Super ID がセキュリティ管理者として指定されている場合、Super ID は他のセキュリティ管理者と同じ REVOKE 権限を持ちます。ただし、この場合、BY *authid-grantor* を省略すると、暗黙の付与者はオブジェクトオーナーではなく Super ID であり、権限を正常に取り消すと、セキュリティ管理者によって付与されたその権限のすべてのインスタンスが取り消されます（つまり、Super ID は他のセキュリティ管理者 ID のように機能します）。これは、オーナーに由来するその権限の付与には影響しません（つまり、オーナーに由来する付与の階層には影響しません）。セキュリティ管理者のグループが空ではなく、Super ID がセキュリティ管理者として指定されていない場合、REVOKE 文に関しては通常のユーザーと同じ制限が Super ID に課されます。

スキーマへの権限を取り付けすための認証権は、オブジェクトに対する権限付与および権限取り消しの認証権と同じです。セキュリティ管理者のグループが空ではなく、Super ID がセキュリティ管理者として指定されていない場合、REVOKE 文に関しては通常のユーザーと同じ制限が Super ID に課されます。

スキーマで取り消される DML 権限は、以下の場合に、そのスキーマの個々のオブジェクトに対するアクセス権を取り消します。

- スキーマのオーナーがオブジェクトに対する WITH GRANT OPTION の権限を保持している。
- 権限を取り消すユーザーがセキュリティ管理者である。

リッスンする IP アドレスを構成可能とした MXOAS 機能強化

MXOAS は、指定された IP アドレスでリッスンするように機能強化されています。MXOAS の複数のインスタンスは、同じまたは異なるポート上の異なる IP アドレスで、同じシステム上で実行できます。

管理者は、MXOAS の起動中に`-ip` タグを使用して IP アドレスを指定できます。例、

```
run  MXOAS /name $L35, nowait /-pn 12222 -ip 15.213.91.4
```

IP アドレスが明示的に指定されている場合、MXOAS は、入力された IP アドレスでリッスンし、それ以外の場合は、システム上のすべての IP アドレスでリッスンします。入力する IP アドレスは、システム上に存在する必要があり、有効な IPv4 または IPv6 アドレスでなければなりません。MXOAS のインスタンスが指定された IP アドレスのポートでリッスンしている場合は、IP アドレスを構成しなければ同じポートでリッスンする別の MXOAS インスタンスを起動できません。

次のコマンドは、IP アドレスとともに MXOAS サービスの詳細を表示します。

```
Info service <service name>, detail;
```

```
Info service *, detail;
```

指定された IP アドレスでリッスンしていない MXOAS サービスについては、IP アドレスのタグは mxci では表示されません。MXDM は、指定された IP アドレスでリッスンしている MXOAS サービスへの接続時に、ポート番号とともに IP アドレスを表示します。

AF_UNIX プロトコルのサポート

ODBC/MX は、CLIM 障害がデータベース接続に影響しないように、OSS ODBC/MX ドライバーと ODBC/MX サーバー間の通信に対応した AF_UNIX プロトコルをサポートしています。

AF_UNIX ソケットを使用するには、データソースの構成に次の EVAR を追加します。

```
MXCS>ADD evar <DS Name>.AF_UNIX,type SET, value 'TRUE';
```

次に例を示します。

```
CS>add evar C1.AF_UNIX,type SET, value 'TRUE';
-- ADD EVAR \NBSTS07.$JVN.C1.AF_UNIX Successful
```

EVAR を設定し、データソースを再起動すると、/usr/tmp の下にプロセス ID の名前で Unix ソケットファイルが作成されます。Unix ソケットファイルを作成するには、/usr/tmp ディレクトリでの適切なアクセス許可（読み取り、書き込みなど）が必要です。ODBC/MX サーバーが終了すると、ソケットファイルは削除されます。AF_UNIX ソケットを使用するように構成されているデータソースは、同じシステム上の OSS ODBC/MX ドライバーでのみ使う必要があります。

注記:

データソースは、AF_UNIX EVAR を設定、変更、または削除する ADD/ALTER/DELETE EVAR の後に再起動される必要があります。

ODBC/MX サーバーが異常終了した場合は、ソケットファイルを手動で削除する必要があります。

テーブルサイズの仕様

デフォルトでは SQL テーブルは、作成時の最大サイズがパーティションあたりほぼ 100MB 程度と小さいです。実稼働環境では、大きなテーブルサイズが必要です。「エクステントサイズ」および「エクステントの最大数」と呼ばれる物理ファイル属性を通じて、作成時にテーブルの初期サイズと最大サイズを設定できます。テーブルのエクステントサイズを決定する新しい属性が追加されました。属性について詳しくは、**属性**を参照してください。

CREATE TABLE 文は、次の 2 つの新しい構文オプションを含むように拡張されました。

```
CREATE TABLE table
                { (table-element [,table-element]...) | like-spec }
                [file-option ]
file-option is:
STORE BY store-option
| LOCATION [\node.]$volume[.subvolume.file-name]
[NAME partition-name]
| partn-file-option
| ATTRIBUTE[S] attribute [,attribute]...
| INITIAL TABLE SIZE table_size
| MAX TABLE SIZE size
```

注記:

ここでの CREATE TABLE の構文は完全ではなく、新しい構文オプションを表示するスニペットに過ぎません。

- `INITIAL TABLE SIZE` *`initial_table_size`*

  プライマリエクステントのサイズを指定します。*initial_table_size* はバイト単位です。
- `MAX TABLE SIZE` *`max_table_size`*

  テーブルの最大サイズを指定します。*max_table_size* はバイト単位です。

メタデータのテーブル

この項では、SQL/MX メタデータテーブルの更新内容を一覧にしています。

メタデータの新たなテーブルと変更されたテーブル

- **SYSTEM_SECURITY_SCHEMA** スキーマに、以下の新しいテーブルが追加されました。
  - **DATABASE_USERS**
  - **DATABASE_USERS_EXT**
  - **PRIVILEGE_GROUPS**
  - **PRIVILEGE_GROUP_GRANTS**
  - **PRIVILEGE_GROUP_MEMBERSHIP**
- **SYSTEM_DBS_SCHEMA スキーマ**が、新しく追加され、新しいテーブルが追加されました。
- **DEFINITION_SCHEMA_VERSION_3500 スキーマ**に、以下の新しいテーブルが追加されました。
  - **TBL_GROUP_PRIVILEGES**
  - **COL_GROUP_PRIVILEGES**
  - **SCH_PRIVILEGES**

  ROUTINES テーブルは増強されて、新しい関数の属性が加えられました。
- 新しい属性が追加されました。

以下のテーブルは、新しいスキーマバージョン 3500 をサポートするために更新されました。

- SCHEMATA テーブルでは、SCHEMA_VERSION、SOURCE_VERSION、および TARGET_VERSION の列が新しいスキーマバージョン 3500 をサポートするために更新されました。
- OBJECTS テーブルでは、バージョン 3500 機能を使用するオブジェクトの OBJECT_FEATURE_VERSION は 3500 です。

SYSTEM_SCHEMA スキーマのテーブル

SCHEMATA テーブルでは、SCHEMA_VERSION、SOURCE_VERSION、および TARGET_VERSION 列が新しいスキーマバージョン 3500 をサポートするように更新されています。

DATABASE_USERS テーブル

DATABASE_USERS テーブルは、システムが認識するすべてのユーザーを表します。ユーザーが、オーナー、付与者、または付与対象者である場合、DATABASE_USERS にそのユーザーの行が存在します。プライマリキーは USERID 列です。テーブルは、GUARDIAN_USER_NAME 列に一意のインデックス DB_USER_BY_NAME を含みます。

列番号	列の見出し	データタイプ	説明
1*	USERID	INT	Guardian ユーザー ID システムに付与者、付与対象者、またはオーナーとして認識されているユーザー。 ユーザーが PUBLIC の場合、-1 を返します。 ユーザーが SYSTEM の場合は、-2 を返します。

表は続く

2	GUARDIAN_USER_NAME	CHAR(32)	Guardian ユーザー名。また、特殊なユーザーの PUBLIC および SYSTEM を含んでいます。
* プライマリキーを示す			

DATABASE_USERS_EXT テーブル

DATABASE_USERS_EXT は、DATABASE_USERS テーブル内のユーザーの外部ユーザー名を表します。外部ユーザー名を持つことはユーザーの任意になります。プライマリキーは EXTERNAL_USER_NAME 列です。テーブルは、USERID 列に一意のインデックスである EXT_USER_BY_USERID を含みます。

列番号	列の見出し	データタイプ	説明
1	USERID	INT	値は、DATABASE_USERS テーブルの USERID 列に対応します。
2*	EXTERNAL_USER_NAME	CHAR(128)	外部ユーザー名
* プライマリキーを示す			

PRIVILEGE_GROUPS テーブル

PRIVILEGE_GROUPS テーブルは、権限グループを表します。権限グループは、GRANT コマンドまたは REVOKE コマンドのターゲットにすることができます。権限グループに付与された権限は、そのグループ内のすべての既存ユーザーと将来のユーザーに影響します。

Guardian ユーザーグループとは、そのグループが作成されると自動的にメンバーになるタイプの権限グループです。

テーブルには、一意のインデックス PRIV_GROUP_BY_ID が、PRIV_GROUP_UID 列にあります。

列番号	列の見出し	データ タイプ	説明
1*	PRIV_GROUP_NAME	CHAR（128）	権限グループの名前。
2	PRIV_GROUP_UID	INT	権限グループの一意の ID。
3	PRIV_GROUP_OWNER	INT	グループ所有者の Guardian ユーザー ID。これは、DATABASE_USERS の一部でなければなりません。

表は続く

4	PRIV_GROUP_TYPE	CHAR(2)	• **G** - Guardian ユーザーグループ（UID が間隔[0:255]）。システムはこの権限グループを所有します。 • **E** - 明示的な権限グループ。CREATE PRIVILEGE GROUP コマンド（UID >= 65536）により作成されます。権限グループを作成するユーザーはその権限グループを所有します。 • **MD** - マルチテナンシーデフォルト権限グループです。テナントデータベースを作成すると作成されます（UID が間隔[256:65535]）。ユーザーデータベースを作成したグローバル DBS 管理者ユーザーは、権限グループを所有します。 • **TR** - マルチテナンシー読み取りアクセス権限グループです。テナントデータベースを作成すると作成されます。ユーザーデータベースを作成したグローバル DBS 管理者ユーザーは、権限グループを所有します。 • **TW** - マルチテナンシー書込みアクセス権限グループです。テナントデータベースを作成すると作成されます。ユーザーデータベースを作成したグローバル DBS 管理者ユーザーは、権限グループを所有します。 • **TC** - マルチテナンシー作成アクセス権限グループです。テナントデータベースを作成すると作成されます。ユーザーデータベースを作成したグローバル DBS 管理者ユーザーは、権限グループを所有します。 暗黙的権限グループのタイプは MD、TR、TW、または TC です
* プライマリキーを示します。			

PRIVILEGE_GROUP_GRANTS テーブル

PRIVILEGE_GROUP_GRANTS テーブルには、ユーザースキーマの権限グループに対する付与に関する情報が含まれています。

クラスタリングキーは PRIV_GROUP_ID、CAT_UID、SCHEMA_UID、および OBJECT_UID の列です。

テーブルにはインデックス PG_GRANT_BY_TARGET が、CAT_UID, SCHEMA_UID および OBJECT_UID の列にあります。

列番号	列の見出し	データ タイプ	説明
1*	PRIV_GROUP_UID	INT	PRIVILEGE_GROUPS テーブルの PRIV_GROUP_UID へのリンク。

表は続く

2*	CAT_UID	LARGEINT	CATSYS テーブルの CAT_UID へのリンク。
3*	SCHEMA_UID	LARGEINT	SCHEMATA テーブルの SCHEMA_UID へのリンク。
4*	OBJECT_UID	LARGEINT	OBJECTS テーブルの OBJECT_UID へのリンク。
* プライマリキーを示します。			

PRIVILEGE_GROUP_MEMBERSHIP テーブル

PRIVILEGE_GROUP_MEMBERSHIP テーブルは、ユーザーと権限グループの間の N:N 関係を表します。クラスタリングキーは PRIV_GROUP_UID および USERID の列です。

テーブルには、インデックス PG_MEMBERSHIP_BY_USERID が USERID 列にあります。

列番号	列の見出し	データ タイプ	説明
1*	PRIV_GROUP_UID	INT	PRIVILEGE_GROUPS テーブルの PRIV_GROUP_UID へのリンク。
2*	USERID	LARGEINT	DATABASE_USERS テーブルの場合は、USERID へのリンクです。
* プライマリキーを示します。			

SYSTEM_DBS_SCHEMA スキーマのテーブル

SYSTEM_DBS_SCHEMA には、DBS 構成テーブルが含まれています。

テーブル名はその使用方法を示します。「DBS_」で始まる名前は、通常、構成済みのリソースを表し、「DATABASE_」で始まる名前は、所定のユーザーデータベースに関連付けられたリソースを表します。

ALL_DATABASES テーブル

ALL_DATABASES テーブルは、すべてのテナントユーザーのデータベースを表します。

データベースにはそのカタログ名である名前が付いていますが、そのカタログ名は CATSYS システムのスキーマテーブルから直接取得できるため、データベースを直接説明するものではありません。クラスター化キーは、対応するカタログ UID を含む DB_UID 列です。

列番号	列名	データタイプ	説明
1*	DB_UID	LARGEINT	データベースのカタログ UID。CATSYS へのリンク。
2	SERVICE_UID	LARGEINT	データベースのサービスへの参照。DBS_SERVICES へのリンク。

表は続く

3	LOCK_UID	LARGEINT	データベースのアクティブなユーティリティ操作の DDL ロックの UID。アクティブなユーティリティ操作がない場合は-1 です。 OBJECTS テーブルと DDL_LOCKS テーブルへのリンク。
4	DB_TYPE	INT	今後使用するために予約されています。
5	SUM_OF_MAX_SRVRS	INT	データベースに関連付けられているすべての MXCS DS の最大サーバー属性の合計。
6	IS_SHARED	CHAR(2)	データベースが共有されているかどうかを示す Y/N。
* プライマリキーを示す			

DATABASE_CPUS テーブル

DATABASE_CPUS テーブルは、ALL_DATABASES と DBS_CPUS との間の N:N の関係を表します。

列番号	列の見出し	データ タイプ	説明
1*	DB_UID	LARGEINT	データベースのカタログ UID。ALL_DATABASES へのリンク。
2	CPU_NUMBER	INT	CPU の数。
3	WEIGHT	INT	この CPU 割り当てにおいて、対応する DBS_CPUS 行で重視される重み。
* プライマリキーを示します。			

DATABASE_DS テーブル

DATABASE_DS テーブルは、データベースでどの MXCS データ ソースが使用されているかを示します。プライマリキーは DS_NAME 列です。

テーブルは、一意でないインデックスである DS_BY_DB_UID を DB_UID 列に含みます。

列番号	列の見出し	データタイプ	説明
1*	DS_NAME	CHAR(128)	DS の名前。MXCS メタデータにリンクします。
2	DB_UID	LARGEINT	データベースのカタログ UID。ALL_DATABASES にリンクします。
* プライマリキーを示す			

DATABASE_PRIVILEGE_GROUPS テーブル

DATABASE_PRIVILEGE_GROUPS テーブルは、どの権限グループがどのデータベースに関連付けられているかを表します。プライマリキーは、DB_UID 列と PRIV_GROUP_ID 列の組み合わせです。

ユーザーデータベースには、データベースと共に作成された 4 つの暗黙の権限グループがあります。

- デフォルトの権限グループは、データベースに関連付けられたすべてのユーザーが対象です。
- データベースへの読み取りアクセス権を持つユーザーを対象とした権限グループ
- データベースへの書き込みアクセス権を持つユーザーを対象とした権限グループ
- データベースへの作成アクセス権を持つユーザーを対象とした権限グループ

データベースには、追加で明示的な権限グループを作成できます。

列番号	列の見出し	データタイプ	説明
1*	DB_UID	LARGEINT	データベースのカタログ UID。
2*	PRIV_GROUP_UID	INT	ALL_DATABASES にリンクします。
3	PRIV_GROUP_TYPE	CHAR(2)	• MD • TR • TW • TC • E 詳しくは、**PRIVILEGE_GROUPS テーブル**を参照してください。
* プライマリキーを示す			

DATABASE_VOLUMES テーブル

DATABASE_VOLUMES テーブルは、どのボリュームがどのデータベースによって使用されるかを表します。プライマリキーは、DB_UID、NODE_NAME、および VOLUME_NAME 列の組み合わせです。

列番号	列の見出し	データタイプ	説明
1*	DB_UID	LARGEINT	データベースのカタログ UID。ALL_DATABASES にリンクします。
2*	NODE_NAME	CHAR(8)	ボリュームが常駐するシステムの Expand ノート名。DBS_VOLUMES にリンクします。
3*	VOLUME_NAME	CHAR(8)	ボリューム名。DBS_VOLUMES にリンクします。
4	RESERVED_CAPACITY	LARGEINT	関連付けられているデータベース用に予約されているボリュームのディスク領域の容量。

表は続く

5	REQUESTED_CAPACITY	LARGEINT	関連付けられているデータベース用に予約されているボリュームに割り当てられているディスク領域の量。ただし、データボリュームが割り当てられた容量よりも多く使用されている場合、SQL/MX は制限を一切課しません。
6	IS_EXCLUSIVE	CHAR(2)	ボリュームがデータベースによる排他的使用に予約されているかどうかを示す Y/N。
* プライマリキーを示す			

DBS_SERVICES テーブル

DBS_SERVICES テーブルは、DBS 環境専用の MXOAS プロセスを表します。プライマリキーは GENERIC_PROCESS_NAME 列です。テーブルは、一意のインデックスである SERVICE_BY_UID を SERVICE_UID 列に含みます。

列番号	列の見出し	データタイプ	説明
1*	GENERIC_PROCESS_NAME	CHAR(32)	MXOAS プロセスが構成されている NSK 内の汎用プロセスの名前です。
2	SERVICE_NAME	CHAR(64)	この MXOAS 経由の接続に使用する DNS または IP アドレス。
3	SERVICE_UID	LARGEINT	サービスの一意の識別子。
4	PROCESS_NAME	CHAR(8)	MXOAS プロセスの構成済み NSK プロセスの名前。
5	INITIAL_PORT	INT	この MXOAS 経由の接続に使用するポート。
6	PORT_RANGE	INT	この MXOAS 用のポートの範囲です。
7	SUM_OF_MAX_SRVRS	INT	MXOAS に関連付けられているすべての MXCS DS の最大サーバー属性の合計。
8	DB_ENABLED	CHAR(2)	このサービスで MT データベースアクセスが有効になっているかどうかを示す Y/N
9	CONNECTION_DS	VARCHAR(128)	関連付けられた MXCS データソースの名前。この手順はオプションです。
* プライマリキーを示す			

DBS_GLOBALS テーブル

このテーブルは、DBS 環境全体のグローバル情報を表します。プライマリキーは、グローバル行が 1 行しかないため、0 値を含む DBS_GLOBAL_UID 列です。このテーブルのプライマリキー以外の列は、SYSTEM_SECURITY_SCHEMA 内のテーブルにリンクします。

列番号	列の見出し	データタイプ	説明
1*	DBS_GLOBAL_UID	LARGEINT	定数値 0。
2	TENANT_ADMIN_GROUP	INT	DBS 管理者グループの Guardian グループ番号。この列は PRIVILEGE_GROUPS テーブルにリンクされます。
3	TENANT_ADMIN_INITIAL_USER	INT	明示的に構成されているグローバル DBS 管理者ユーザーの Guardian ユーザー ID。この列は PRIVILEGE_GROUPS テーブルにリンクされます。
4	TENANT_ADMIN_INTERN_USER	INT	暗黙的に構成されているグローバル DBS 内部ユーザーの Guardian ユーザー ID。この列は PRIVILEGE_GROUPS テーブルにリンクされます。
* プライマリキーを示す			

DBS_PLATFORM_USERS テーブル

DBS_PLATFORM_USERS テーブルは、テナント使用のために Safeguard に予約されているユーザーを表します。テーブルのプライマリキーは USERID 列です。テーブルの GUARDIAN_GROUP_UID 列には PLATFORM_USER_BY_GROUP インデックスがあります。

列番号	列の見出し	データタイプ	説明
1*	USERID	INT	ユーザーの Guardian ユーザー ID。
2	GUARDIAN_USER_NAME	CHAR(32)	ユーザーの対応する Guardian ユーザー名。
3	GUARDIAN_GROUP_UID	LARGEINT	Guardian ユーザーのグループ番号。
4	LOCK_UID	LARGEINT	このユーザーで動作するアクティブなユーティリティ操作の DDL ロックの UID。ユーティリティ操作がアクティブではない場合は-1 です。OBJECTS テーブルと DDL_LOCKS テーブルにリンクします。
5	IN_USE	CHAR(2)	ユーザーが 1 つ以上のテナントに関連付けられているかどうかを示す Y/N。
* プライマリキーを示す			

DBS_SFG_GROUPS テーブル

DBS_SFG_GROUPS テーブルは、テナント使用のために Safeguard に予約されているファイル共有グループを表します。テーブルの FS_GROUP_NAME 列には一意のインデックスである SFG_GROUP_BY_NAME があります。

列番号	列の見出し	データタイプ	説明
1*	FS_GROUP_UID	INT	予約されているファイル共有グループの Safeguard ファイル共有グループ。PRIVILEGE_GROUPS テーブルにリンクします。
2	FS_GROUP_NAME	CHAR(32)	対応するファイルの共有グループの名前。
3	DB_UID	LARGEINT	関連付けられているデータベースのカタログ UID。ファイル共有グループがデータベースに関連付けられていない場合は-1 です。 ALL_DATABASES にリンクします。
4	LOCK_UID	LARGEINT	このファイル共有グループで動作するアクティブなユーティリティ操作の DDL ロックの UID。ユーティリティ操作がアクティブではない場合は-1 です。 OBJECTS テーブルと DDL_LOCKS テーブルへにリンクします。
* プライマリキーを示す			

DBS_VOLUMES テーブル

DBS_VOLUMES テーブルは、DBS に構成されているすべてのデータボリュームを表します。プライマリキーは、NODE_NAME 列および VOLUME_NAME 列の組み合わせです。FREE_CAPACITY および FULL_CAPACITY 列の値が同じである場合、データボリュームは割り当てられません。

列番号	列の見出し	データタイプ	説明
1*	NODE_NAME	CHAR(8)	ボリュームが存在するシステムの Expand ノード名。
2*	VOLUME_NAME	CHAR(8)	ボリュームの名前。
3	FULL_CAPACITY	LARGEINT	ボリューム上のディスク領域の量。
4	FREE_CAPACITY	LARGEINT	ボリューム上の予約されていないディスク領域の量。
5	PRIMARY_CPU	INT	構成時のボリュームのプライマリ CPU。DBS_CPUS テーブルにリンクします。
6	BACKUP_CPU	INT	構成時のボリュームのバックアップ CPU。DBS_CPUS テーブルにリンクします。

表は続く

7	LOCK_UID	LARGEINT	このボリュームで動作するアクティブなユーティリティ操作の DDL ロックの UID。ユーティリティ操作がアクティブではない場合は-1 です。OBJECTS テーブルと DDL_LOCKS テーブルにリンクします。
* プライマリキーを示す			

DEFINITION_SCHEMA_VERSION_3500 スキーマのテーブル

OBJECTS テーブルでは、バージョン 3500 機能を使用するオブジェクトの OBJECT_FEATURE_VERSION は 3500 となります。

次のテーブルが追加されます。

COL_GROUP_PRIVILEGES テーブル

COL_GROUP_PRIVILEGES テーブルは、付与対象者が特権であるカタログの列に付与情報を格納します。テーブルに対応する権利付与情報は、TBL_GROUP_PRIVILEGES テーブル内に別々に格納されます。

列番号	列の見出し	データタイプ	説明
1*	TABLE_UID	LARGEINT	テーブルの UID。
2*	COLUMN_NUMBER	INT	テーブル内の位置。最初の列は 0 です。
3*	GRANTOR	INT	付与者がオーナーとして動作する Super ID である場合、オーナーの付与者の権限 ID。
4*	GRANTOR_TYPE	CHAR(2)	ユーザーが付与する場合、値は U です。スキーマのオーナーとして付与した場合、値は O です。システムは、グループの権限付与を実行しません。
5*	GRANTEE	INT	付与対象者のグループ権限 UID。SYSTEM_SECURITY_SCHEMA の PRIVILEGE_GROUPS へのリンクです。
6*	GRANTEE_TYPE	CHAR(2)	値は U です ― 権限グループはオーナーと PUBLIC のいずれにすることもできません。
7*	PRIVILEGE_TYPE	CHAR(2)	権限タイプは次のとおりです。 • S SELECT • U UPDATE • R REFERENCES

表は続く

8	IS_GRANTABLE	CHAR(2)	N – WITH GRANT OPTION 値は、グループの権限付与には適用されません。
* プライマリキーを示す			

ROUTINES テーブル

ROUTINES は、このカタログですべてのタイプのストアドプロシージャーとファンクションの共通属性を含む DEFINITION_SCHEMA_VERSION_*vernum* のメタデータテーブルです。

VARCHAR 列は、文字をそのまま格納します（大文字に変換されません）。

注記:

このテーブルでは、LINK_TYPE、SECURITY_TYPE、STORED が新しい列として SQL/MX 3.5 で追加されています。

列の見出し	データの種類	説明
*1 UDR_UID	LARGEINT	プロシージャーオブジェクトの UID
2 UDR_TYPE	CHAR(2)	プロシージャーの場合は P
3 LANGUAGE_TYPE	CHAR(2)	Java の場合は J
4 DETERMINISTIC_BOOL	CHAR(2)	確定的な場合は Y それ以外の場合は N
5 SQL_ACCESS	CHAR(2)	M=MODIFIES SQL DATA N=NO SQL C=CONTAINS SQL R=READS SQL DATA
6 CALL_ON_NULL	CHAR(2)	Y（渡されたパラメーターがヌルの場合は SPJ を呼び出す）
7 ISOLATE_BOOL	CHAR(2)	Y（別プロセスで実行）
8 PARAM_STYLE	CHAR(2)	Java の場合は J
9 EXTRA_CALL	CHAR(2)	N（追加の呼び出しなし）
10 TRANSACTION_ATTRIBUTES	CHAR(2)	常に RQ（今後使用するために予約されています）
11 MAX_RESULTS	INT	SPJ 結果セットとともに 0～255 の範囲の正の値が表示されます
12 STATE_AREA_SIZE	INT	将来使用のため確保

表は続く

列の見出し	データの種類	説明
13 UDR_ATTRIBUTES	VARCHAR(128)	将来使用のため確保
14 EXTERNAL_PATH	VARCHAR(256)	指定された EXTERNAL PATH の値
15 EXTERNAL_FILE	VARCHAR(256)	Java クラスの名前で、前にパッケージ名が付加されている可能性があります
16 EXTERNAL_NAME	VARCHAR(128)	Java メソッドの単純名
17 LINK_TYPE	CHAR(2)	ユーザー定義関数（UDF）を静的（S）リンクで解決するか動的（D）リンクで解決するかを示します。 • S - 静的リンク • D - 動的リンク
18 SECURITY_TYPE	CHAR(2)	UDF を定義元（D）と呼び出し元（I）どちらの権限で実行するかを示します。 • D - 定義元 • I - 呼び出し元
19 STORED	CHAR(2)	UDF が保存されているのはデータベース内か、データベースの外部かを示します。 • Y - UDF がデータベース内 • N - UDF がデータベースの外部
* プライマリキーを示します。		

TBL_GROUP_PRIVILEGES テーブル

TBL_GROUP_PRIVILEGES テーブルは、付与対象者が権限グループであるカタログに、テーブル、ビュー、シーケンスジェネレーター、およびストアドプロシージャの付与情報を格納します。個々の列の対応する付与情報は、COL_GROUP_PRIVILEGES テーブルに格納されます。

クラスタリングキーは、TABLE_UID、GRANTOR、GRANTOR_TYPE、GRANTEE、GRANTEE_TYPE、PRIVILEGE_TYPE で構成されます。TBL_GROUP_PRIVILEGES テーブルのレイアウトは SCH_GROUP_PRIVILEGES、SCH_PRIVILEGES、および TBL_PRIVILEGES テーブルと同じです。

列番号	列の見出し	データタイプ	説明
1*	GRANTOR	INT	付与者のセキュリティ ID、または付与者がオーナーとして動作する Super ID の場合はオーナーのセキュリティ ID。

表は続く

2*	GRANTOR_TYPE	CHAR(2)	値は以下のとおりです。 ユーザーが付与する場合は U スキーマのオーナーとして付与した場合は O システムは、権限グループ付与を実行しません。
3*	GRANTEE	INT	付与対象者の権限グループ UID。SYSTEM_SECURITY_SCHEMA 内の PRIVILEGE_GROUPS にリンクします。
4*	GRANTEE_TYPE	CHAR(2)	値は U で、権限グループはオーナーでも PUBLIC でもないことを示します。
5*	TABLE_UID	LARGEINT	ターゲットスキーマのスキーマラベルオブジェクトの UID。
6*	PRIVILEGE_TYPE	CHAR(2)	権限タイプは次のとおりです。 D - DELETE E - EXECUTE I - INSERT R - REFERENCES S - SELECT U - UPDATE
7	IS_GRANTABLE	CHAR(2)	値は N であり、WITH GRANT OPTION が権限グループ付与に適用されないことを表します
* プライマリキーを示す			

SCH_PRIVILEGES テーブル

SCH_PRIVILEGES テーブルで説明する権限では、付与対象者は個々のユーザーであり、ターゲットはスキーマ全体です。

クラスタリングキーは、TABLE_UID、GRANTOR、GRANTOR_TYPE、GRANTEE、GRANTEE_TYPE、PRIVILEGE_TYPE で構成されます。SCH_ PRIVILEGES テーブルのレイアウトは、SCH_GROUP_PRIVILEGES、TBL_GROUP_PRIVILEGES、および TBL_PRIVILEGES テーブルのレイアウトと同じです。

このテーブルは SQL/MX 3.5 以降からサポートされています。

列番号	列の見出し	データ タイプ	説明
1*	GRANTOR	INT	GRANTOR_TYPE が U または O の場合、付与者の承認 ID です。付与者が所有者を代理しているスーパー ID の場合は、所有者の承認 ID です。他の値には意味がありません。

表は続く

2*	GRANTOR_TYPE	CHAR(2)	値は以下のとおりです。 S - システム付与 U - ユーザー付与 O - スキーマ所有者として付与
3*	GRANTEE	INT	GRANTEE_TYPE が U または O の場合は、付与対象者の承認 ID。この列には、それ以外の意味はありません。
4*	GRANTEE_TYPE	CHAR(2)	値は以下のとおりです。 P - パブリックが付与 U - ユーザー付与 O - 付与対象者がスキーマ所有者
5*	TABLE_UID	LARGEINT	ターゲットスキーマのスキーマラベルオブジェクトの UID。
6*	PRIVILEGE_TYPE	CHAR(2)	権限タイプは次のとおりです。 A - ALTER C - CREATE D - DELETE D - R DROP E - EXECUTE I - INSERT R - REFERENCES S - SELECT SU - USAGE U - UPDATE
7	IS_GRANTABLE	CHAR(2)	値は、付与オプションありで付与された場合は Y で、そうでない場合は N です。
* プライマリキーを示します。			

SCH_GROUP_PRIVILEGES テーブル

SCH_GROUP_PRIVILEGES テーブルで説明する権限では、付与対象者は権限グループであり、ターゲットはスキーマ全体です。

クラスタリングキーは、TABLE_UID、GRANTOR、GRANTOR_TYPE、GRANTEE、GRANTEE_TYPE、PRIVILEGE_TYPE で構成されます。SCH_GROUP_PRIVILEGES テーブルのレイアウトは SCH_PRIVILEGES、TBL_GROUP_PRIVILEGES、および TBL_PRIVILEGES テーブルのレイアウトと同じです。

列番号	列の見出し	データ タイプ	説明
1*	GRANTOR	INT	付与者の承認 ID です。付与者が所有者を代理しているスーパー ID の場合は、所有者の承認 ID です。
2*	GRANTOR_TYPE	CHAR(2)	値は、 • U - ユーザーが付与する場合 • O - スキーマ所有者として付与される場合 システムは、権限グループ付与を行ないません。
3*	GRANTEE	INT	付与対象者の権限グループ UID です。SYSTEM_SECURITY_SCHEMA 内の PRIVILEGE_GROUPS へのリンク。
4*	GRANTEE_TYPE	CHAR(2)	値は U で、権限グループが所有者でも PUBLIC でもないことを示します。
5*	TABLE_UID	LARGEINT	ターゲットスキーマのスキーマラベルオブジェクトの UID。
6*	PRIVILEGE_TYPE	CHAR(2)	権限タイプは次のとおりです。 A - ALTER C - CREATE D - DELETE D - R DROP E - EXECUTE I - INSERT R - REFERENCES S - SELECT SU - USAGE U - UPDATE
7	IS_GRANTABLE	CHAR(2)	値は N であり、WITH GRANT OPTION が権限グループ付与に適用されないことを表します。
* プライマリキーを示します。			

属性

SQL/MX 3.5 では、以下のデータ型の属性が追加されます。

属性	設定
VARCHAR_PARAM_DEFAULT_SIZE	コンテキストに応じて、タイプが設定されていないパラメーターはクエリのコンパイル時に VARCHAR タイプに変換されます。この CQD によって、デフォルトの長さを変更できます。 使用可能な値：1～32768。 デフォルトは 255 文字です。
NUMBER_DEFAULT_PRECISION	この CQD によって、NUMBER データ型のデフォルトの精度を変更できます。 使用可能な値：1～128。 デフォルト値は 18 です。
NUMBER_DEFAULT_SCALE	この CQD によって、NUMBER データ型のデフォルトの位取りを変更できます。 使用可能な値：0～128。 デフォルト値は 6 です。

SQL/MX 3.5 では、次のテーブルサイズの属性が追加されます。

属性	設定
POS_PRI_EXT_SIZE	主なエクステントのサイズを指定します。SQL/MX リリース 3.5 以降では、テーブルのサイズから、またはこの CQD のみからプライマリエクステントのサイズが決定されます。 デフォルト値は 16 です。
POS_SEC_EXT_SIZE	セカンダリエクステントのサイズを指定します。SQL/MX リリース 3.5 以降では、テーブルのサイズや、この CQD からのみセカンダリエクステントのサイズが決定されます。 デフォルト値は 64 です。
POS_MAX_EXT	デフォルトの MAXEXTENTS 値を指定します。SQL/MX リリース 3.5 以降では、テーブルのサイズや、この CQD からのみ MAXEXTENTS サイズが決定されます。 デフォルト値は 160 です。

SQL/MX 3.5 では、外部ユーザー名に以下の属性が追加されます。

属性	設定
EXTERNAL_NAME_CASESENSITIVE	外部ユーザー名の大文字小文字を区別するかどうかを指定します。これは読み取り専用 CQD です。 使用可能な値：ON、OFF、および NONE です。 デフォルト値は NONE です。 ON に設定すると、外部ユーザー名は他の SQL 識別子のように扱われます。通常の識別子は大文字小文字は区別されず、入力時に大文字に変換され、区切られた識別子は大文字小文字が区別され大文字に変換されません。 OFF に設定すると、外部ユーザー名は常に入力時に大文字に変換されます。 NONE に設定すると、外部ユーザー名を作成することはできません。
EXTERNAL_NAME_DISPLAY	ユーザー名を表示する方法を指定します。マルチテナント環境では、外部ユーザー名が常に表示されます。 使用可能な値：ON および OFF デフォルト値は OFF です。 ON に設定すると、エラーメッセージ、SHOWDDL、および SHOWLABE では外部ユーザー名が表示されます（あれば）。それ以外の場合、Guardian ユーザー名が表示されます。 OFF に設定すると、Guardian ユーザー名のみが表示されます。

Web サイト

全般的な Web サイト

Hewlett Packard Enterprise Information Library

http://www.hpe.com/info/EIL

Hewlett Packard Enterprise サポートセンター

http://www.hpe.com/support/hpesc

Contact Hewlett Packard Enterprise Worldwide

http://www.hpe.com/assistance

サブスクリプションサービス/サポートのアラート

http://www.hpe.com/support/e-updates-ja

Software Depot

http://www.hpe.com/support/softwaredepot

カスタマーセルフリペア

http://www.hpe.com/support/selfrepair

L シリーズのマニュアル

http://www.hpe.com/info/nonstop-ldocs

J シリーズのマニュアル

http://www.hpe.com/info/nonstop-jdocs

上記以外の Web サイトについては、**サポートと他のリソース**を参照してください。

サポートと他のリソース

Hewlett Packard Enterprise サポートへのアクセス

- ライブアシスタンスについては、Contact Hewlett Packard Enterprise Worldwide の Web サイトにアクセスします。

  http://www.hpe.com/assistance
- ドキュメントとサポートサービスにアクセスするには、Hewlett Packard Enterprise サポートセンターの Web サイトにアクセスします。

  http://www.hpe.com/support/hpesc

ご用意いただく情報

- テクニカルサポート登録番号（該当する場合）
- 製品名、モデルまたはバージョン、シリアル番号
- オペレーティングシステム名およびバージョン
- ファームウェアバージョン
- エラーメッセージ
- 製品固有のレポートおよびログ
- アドオン製品またはコンポーネント
- 他社製品またはコンポーネント

アップデートへのアクセス

- 一部のソフトウェア製品では、その製品のインターフェイスを介してソフトウェアアップデートにアクセスするためのメカニズムが提供されます。製品のドキュメントを確認し、推奨されるソフトウェアアップデートの方法を特定します。
- 製品のアップデートをダウンロードするには、以下のいずれかに移動します。

  Hewlett Packard Enterprise サポートセンター

  http://www.hpe.com/support/hpesc

  Hewlett Packard Enterprise サポートセンター：ソフトウェアのダウンロード

  http://www.hpe.com/support/downloads

  Software Depot

  http://www.hpe.com/support/softwaredepot
- eNewsletters およびアラートにサブスクライブするには、以下の Web サイトにアクセスします。

  http://www.hpe.com/support/e-updates-ja
- お客様の権利の表示や更新を行ったり、契約と保証をプロフィールとリンクさせたりするには、Hewlett Packard Enterprise サポートセンターの **More Information on Access to Support Materials** ページをご覧ください。

  http://www.hpe.com/support/AccessToSupportMaterials

重要:

一部のアップデートにアクセスするには、Hewlett Packard Enterprise サポートセンターからアクセスするときに製品の権利付与情報が必要になる場合があります。関連する権利付与情報を使って HP パスポートをセットアップしておく必要があります。

カスタマーセルフリペア（CSR）

Hewlett Packard Enterprise カスタマーセルフリペア（CSR）プログラムでは、ご使用の製品をお客様ご自身で修理することができます。CSR 部品を交換する必要がある場合、お客様のご都合のよいときに交換できるよう直接配送されます。一部の部品は CSR の対象になりません。Hewlett Packard Enterprise もしくはその正規保守代理店が、CSR によって修理可能かどうかを判断します。

リモートサポート（HPE 通報サービス）

リモートサポートは、保証またはサポート契約の一部としてサポートデバイスでご利用いただけます。リモートサポートは、インテリジェントなイベント診断を提供し、ハードウェアイベントを Hewlett Packard Enterprise に安全な方法で自動通知します。これにより、ご使用の製品のサービスレベルに基づいて、迅速かつ正確な解決が行われます。ご使用のデバイスをリモートサポートに登録することを強くおすすめします。

ご使用の製品にリモートサポートの追加詳細情報が含まれる場合は、検索を使用してその情報を見つけてください。

リモートサポートおよびプロアクティブケア情報

HPE 通報サービス

http://www.hpe.com/jp/hpalert

HPE プロアクティブ ケアサービス

http://www.hpe.com/services/proactivecare-ja

HPE プロアクティブケアサービス：サポートされている製品のリスト

http://www.hpe.com/services/proactivecaresupportedproducts（英語）

HPE プロアクティブケアアドバンストサービス：サポートされている製品のリスト

http://www.hpe.com/services/proactivecareadvancedsupportedproducts

保証情報

ご使用の製品の保証を確認するには、Hewlett Packard Enterprise サポートセンターで入手できる *サーバー、ストレージ、電源、ネットワーク、およびラック製品の安全と準拠に関する情報*を参照します。

http://www.hpe.com/support/Safety-Compliance-EnterpriseProducts

追加保証情報

HPE ProLiant と x86 サーバーおよびオプション

http://www.hpe.com/support/ProLiantServers-Warranties

HPE エンタープライズサーバー

http://www.hpe.com/support/EnterpriseServers-Warranties

HPE ストレージ製品

http://www.hpe.com/support/Storage-Warranties

HPE ネットワーク製品

http://www.hpe.com/support/Networking-Warranties

規定に関する情報

安全、環境、および規定に関する情報については、Hewlett Packard Enterprise サポートセンターから *サーバー、ストレージ、電源、ネットワーク、およびラック製品の安全と準拠に関する情報*を参照してください。

http://www.hpe.com/support/Safety-Compliance-EnterpriseProducts

規定に関する追加情報

Hewlett Packard Enterprise は、REACH（欧州議会と欧州理事会の規則 EC No 1907/2006）のような法的な要求事項に準拠する必要に応じて、弊社製品の含有化学物質に関する情報をお客様に提供することに全力で取り組んでいます。この製品の含有化学物質情報レポートは、次を参照してください。

http://www.hpe.com/info/reach

RoHS、REACH を含む Hewlett Packard Enterprise 製品の環境と安全に関する情報と準拠のデータについては、次を参照してください。

http://www.hpe.com/info/ecodata

社内プログラム、製品のリサイクル、エネルギー効率などの Hewlett Packard Enterprise の環境に関する情報については、次を参照してください。

http://www.hpe.com/info/environment

ドキュメントに関するご意見、ご指摘

Hewlett Packard Enterprise では、お客様により良いドキュメントを提供するように努めています。ドキュメントを改善するために役立てさせていただきますので、何らかの誤り、提案、コメントなどがございましたら、ドキュメントフィードバック担当（**docsfeedback@hpe.com**）へお寄せください。この電子メールには、ドキュメントのタイトル、部品番号、版数、およびドキュメントの表紙に記載されている刊行日をご記載ください。オンラインヘルプの内容に関するフィードバックの場合は、製品名、製品のバージョン、ヘルプの版数、およびご利用規約ページに記載されている刊行日もお知らせください。

Oracle と互換性のある SQL/MX の関数と式

以下の SQL/MX の関数および式は、Oracle と互換性があります。

集約関数

- AVG
- COUNT
- MAX
- MIN
- STDDEV
- SUM
- VARIANCE

文字列関数

- ASCII
- COALESCE
- CONCAT
- CONVERT
- CURRENT_USER
- DECODE
- LOWER
- LTRIM
- LNNVL
- NULLIF
- NVL
- NVL2
- RTRIM
- TO_CHAR
- TRANSLATE
- TRIM
- USER

日付時刻関数

- CURRENT_TIMESTAMP
- EXTRACT
- LAST_DAY
- MONTHS_BETWEEN
- TO_DATE

数値演算関数

- ABS
- ACOS
- ASIN
- ATAN
- ATAN2
- AVG
- COS
- COSH

- EXP
- FLOOR
- MOD
- POWER
- ROUND
- SIGN
- SQRT
- SIN
- SINH
- TAN
- TANH

式

- CAST