Unify DataServer™

機能拡張ガイド

Release 7.0B - 7.2

Publications team Natalie Calkins
Linda Costello

Part Number:7829-01

目次

このマニュアルについて

このマニュアル（*Unify DataServer: 機能拡張ガイド Release 7.0B - 7.2*）は、Unify DataServer ﾘﾘｰｽ 7.0B, 7.1, 7.2 において追加された新機能について説明します。新機能が適応されている特定のﾘﾘｰｽﾊﾞｰｼﾞｮﾝは、機能の説明と一緒に記されています。このマニュアル内の情報は、既存の Unify DataServer ﾘﾘｰｽ 7.0A 用マニュアルからの補充です。

対象とする読者

このマニュアルは、Unify DataServer を使用する開発者のために書かれています。

表記ルール

このマニュアルでは、次のような表記方法を使っています。

通常の文章

< >	不等記号で囲まれた語は、引数、変数、数値、または提供すべき式などを表します。
Monospaced フォント	必ず入力しなければならないコードサンプルやテキストはこのフォントで表されています。
Windows	特に指定のない限り、Windows は Windows NT と Windows 2000 の両方を意味します。

特に重要な段落

ヒント	このマークのついた段落には、推奨する操作などの有用な情報が含まれています。
要注意	このマークのついた段落には、データの損失を引き起こす可能性のある操作について警告しています。
関連情報	このマークのついた段落には、そのセクションとトピックに関連した参照情報を提供します。

テクニカルサポート

ユニファイ製品に関して何か問題が起こった場合は、色々な情報が提供されている Web サイト上のテクニカルサポートをご覧ください。ユニファイのテクニカルサポートには下記のアドレスよりアクセスできます。

http://support.unify.com/supportal/Home.html

テクニカルサポートが必要な場合には、まず、ユニファイテクニカルサポート FAQ に探している答え(情報)があるかチェックしてください。ユニファイテクニカルサポート FAQ には下記のアドレスよりアクセスできます。

http://support.unify.com/supportal/FAQs.htm

FAQ を見ても問題が解決しない時には、ユニファイカスタマの皆さんによるディスカッションフォーラムであるメーリングリストへの参加をおすすめします。メーリングリストを購読するには、下記の電子メールアドレス宛てに、

listserv@unify.com

次のようなテキストをメッセージ本文に入力して、メールを送信してください。

subscribe customers

メールの受信後、確認のメールをお送りします。このメールには、メーリングリストサービスへのメールでの質問の仕方や登録の解除方法も書かれています。

Unify 製品ドキュメントの Web サイトでは正誤表やリリースされてからのドキュメントの変更点が確認できます。また、最新のバージョンのドキュメントをダウンロードすることもできます。Unify 製品ドキュメントには下記のアドレスよりアクセスできます。

http://support.unify.com/supportal/Docs.htm

1

TRUNCATE TABLE

リリース 7.2 より利用可能

SQL/A の変更

TRUNCATE TABLE という新しい SQL/A 文が Unify DataServer に追加されました。以下のセクションで、TRUNCATE TABLE 文について説明します。

TRUNCATE TABLE

対話型、または埋め込み型（Prepare 可能）

DML 文

構文

TRUNCATE TABLE *table* **;**

引数

table トランケートするテーブル名。ただし、次のテーブルは指定できません。リンクインデックスの一部（親，子）であるテーブル。および、外部ファイル（.idx）に格納される B ツリーをもつテーブル。

説明

TRUNCATE TABLE 文は、指定されたテーブルに割り当てられたセグメントを解放し、ボリュームのフリースペースにそのセグメントを戻します。テーブル内の全てのデータは、そのインデックスと共に完全に削除されます。ただし、テーブルの定義とインデックスの定義はそのまま残されます。その際、テーブル上のトリガーなどは実行されず、DIS の定義は無視されます。

必要とされていないテーブルによって使用されているセグメントを、テーブルの定義は残したままで簡単に解放したい場合に、TRUNCATE TABLE 文を使用して下さい。（DROP TABLE 文もまた、割り当てられたセグメントを解放します。しかし、この場合テーブルの定義は保持されません。）

これは、DML 操作ですが、変更を元に戻すことは不可能です。しかし、テーブルのトランケート操作は、ログに記録されますので、**irma** ユーティリティを使って再実行することは可能です。

TRUNCATE TABLE 文は、特定のトランザクション内で実行されなければなりません。ただし、そのトランザクションをロールバックすることはできません。

トランケートされているテーブルが、可変長データを含む場合、可変長データボリューム内で保持されているスペースを解放する必要があります。この方法に関する情報は、『Unify DataServer リリース 7.2： README』内の「Reclaim ユーティリティは利用できない」を参照してください。

ヒント

TRUNCATE TABLE 文は、同様の DELETE 文よりも非常に速く実行できます。DELETE 文は、テーブルに割り当てられたセグメントを解放する代わりに、そのセグメントを空にしておく行単位のオペレーションです。

機密保護

この文を使用するためには、次の条件のうち1つは満たしていなければなりません。

- DBA 特権があること
- カレントスキーマ内のテーブルであること
- 別のスキーマ内のテーブルの場合、カレントスキーマに、そのテーブルまたはそのテーブルを含んでいるスキーマに対する DELETE 特権があること

例

「company」という名前のテーブルをトランケートするには、次のように指定します。

```
sql> TRUNCATE TABLE company;
```

参照

DROP TABLE, DELETE, utrnctbl()

付録 A の変更

付録 A で記述されている予約語の一覧に TRUNCATE を追加します。

埋め込み型 SQL/A の変更

『*Unify DataServer: 埋め込み型 SQL/A の使用*』マニュアルの「埋め込み型 SQL/A の紹介」の章に次の表を追加します。

コマンド	埋め込みバージョン	Prepare 可能かEXECUTE IMMEDIATEで使用できる
TRUNCATE TABLE		×

RHLI の変更

RHLIで、テーブルをトランケートする際の操作は、**utrnctbl**関数によって実行します。この関数のリファレンスを次のセクションに記載します。

utrnctbl

テーブルをトランケートします。

構文

int utrnctbl(*txid*, *tid*, *status***)**
UTXID *txid*;
UTID *tid*;
USTATUS **status*;

引数

txid	トランザクションID
tid	トランケートするテーブルのテーブルID
status	関数の実行ステータス

解説

utrnctbl 関数は、指定されたテーブルに割り当てられたセグメントを解放し、ボリュームのフリースペースにそのセグメントを戻します。テーブル内の全てのデータは、そのインデックスと共に完全に削除されます。ただし、テーブルの定義とインデックスの定義はそのまま残されます。その際、テーブル上のトリガーなどは実行されず、DIS の定義は無視されます。

必要とされていないテーブルによって使用されているセグメントを、テーブルの定義は残したままで簡単に解放したい場合に、**utrnctbl** 関数を使用して下さい。(**udrptbl** 関数もまた、割り当てられたセグメントを解放します。しかし、この場合テーブルの定義は保持されません。)

これは、DML 操作ですが、変更を元に戻すことは不可能です。しかし、テーブルのトランケート操作は、ログに記録されますので、**irma** ユーティリティを使って再実行することは可能です。

utrnctbl 関数は、特定のトランザクション内で実行されなければなりません。ただし、そのトランザクションをロールバックすることはできません。

トランケートされているテーブルが、可変長データを含む場合、可変長データボリューム内で保持されているスペースを解放する必要があります。この方法に関する情報は、『Unify DataServer リリース 7.2: README』内の「Reclaim ユーティリティは利用できない」を参照してください。

utrnctbl 関数は、リモートアクセスによる利用はできません。

機密保護

この文を使用するために、次の条件のうち1つは満たしていなければなりません。

- DBA 特権があること

- カレントスキーマ内のテーブルであること
- 別のスキーマ内のテーブルの場合、カレントスキーマに、そのテーブルまたはそのテーブルを含んでいるスキーマに対する DELETE 特権があること

例

以下の例はテーブルをトランケートするためにutrnctbl()関数を使用しています。

```
/* define required parameters */
UTXID txid;
USTATUS status;
...
/* start a new transaction */
...
/* xlock the table to be truncated */
...
/* truncate table scratchTbl */
if ( utrnctbl(txid, $scratchTbl.$, &status) != SUCCESS )
{
/* handle error */
fprintf(stderr, “Error truncating scratchTbl: status = %ld¥n”, status);
...
}
/* commit the transaction */
...
```

TRUNCATE TABLE に関連した新しいエラーメッセージ

-1586 UETTLINK リンクインデックスの一部のテーブルをトランケートできません
-1587 UETTDML TRUNCATE は他の DML や DDL の操作と一緒にできません
-1588 UETTRLBK このリリースでは、トランケート操作をロールバックできません
-1589 UETTBIDX 旧式な B ツリーファイル(.idx)を持つテーブルをトランケートできません

2

サードパーティ backup/restore ユーティリティ

リリース 7.2 より利用可能

budb と **redb** ユーティリティでは、データベースファイルの物理的なバックアップやリストアを実行するためにサードパーティのユーティリティを利用できるようになりました。サードパーティのユーティリティが実際のバックアップの読み書きを制御し、**budb**, **redb**, **irma** ユーティリティがバックアップ／リストアプロセスの論理的な解釈を制御します。例えば、budb ユーティリティは、物理的なバックアップが進行している間、データベースボリュームが安定していることを保証します。

この機能によって、システム全体で動作するバックアップツールを選ぶことができるようになります。すなわち、DataServer のデータベースファイルだけでなく、他のデータベース以外のファイルもバックアップを取ることができるツールを選べるのです。例えば、データの圧縮, データの検証, エラーの検出と補正, ハイスピードのような機能を提供するサードパーティツールを探してください。

budb ユーティリティの変更

budb ユーティリティは、その構文内で新しい引数(-Obudb_util)を利用できます。以下のセクションでは、**budb** ユーティリティでの変更を記述します。

構文

budb [-d dbname **] [-Ojrnlwait=***number_of_seconds* **]**
[-Obudb_util=*string* **]**

新しい引数

-Obudb_util バックアップを実行するスクリプトの名前を含む文字列。文字列には、そのスクリプトへの引数を含めることができます。

文字列は、その実行のためにオペレーティングシステムの system()関数に渡されます。**budb** ユーティリティは、$DBPATH ディレクトリにおいてスクリプトを実行します。

解説

サードパーティのバックアップユーティリティの使用

サードパーティのバックアップユーティリティを使用するには、一般的に以下の項目を実行するスクリプトを作成します。

1. どのファイルをバックアップする必要があるかを決定するために、バックアップリストファイル(.bul)を読み込みます。
2. これらのファイルに対して、バックアップユーティリティを実行します。
3. エラーのチェックをします。

バックアップリストファイルは、**budb** ユーティリティが、-Obudb_util 引数付きで実行される際に作成する ASCII テキストファイルです。バックアップリストファイルは$DBPATH ディレクトリ内に作成されます。どのファイルをサードパーティのバックアップユーティリティでバックアップする必要があるかを識別するために、このファイルを使用します。**redb** ユーティリティも、リストアする必要のあるファイルを識別するために、このファイルを使用します。

バックアップリストファイルには、バックアップするファイルのリスト、および各ファイルのタイプ, オフセット, 長さ, パス名のカラムが含まれます。リスト中の相対ファイルパスは、$DBPATH ディレクトリからの相対パスとみなされます。以下に、サンプルのバックアップリストファイルを示します。

```
67
I       1052672 0           TxLogmark
L       0       0           /MyDB/file.bul
f       0       4098048     /MyDB/file.lg
f       0       112         /MyDB/file.dis
f       0       3145728     /MyDB/file.dbv
f       0       62910464    /MyDB/file.db
b       0       20971520    vols/vol_1gb
```

一行目は、シングルフィールドで構成されます。それは、バックアップのバージョン番号で、この事例では、「67」です。バックアップのバージョン番号は、通常 file.bu のヘッダ内に格納されているもの、また以降のジャーナルファイル全てに格納されるものと同じ番号です。サードパーティのバックアップユーティリティは、複数のバックアップとジャーナルのグループの順序を管理するために、この番号を使用することができます。

二行目の「I 1052672 TxLogmark」は、サードパーティのバックアップユーティリティからは無視できます。これは、バックアップをリストアする際に**redb**ユーティリティによってのみ使用されます。

残りの行はそれぞれ、サードパーティのバックアップユーティリティでバックアップされるべきファイルを表します。バックアップリストファイルそれ自身もバックアップする必要があります。各行には、4つのフィールドがあり、それぞれ空白文字で区切られています。フィールドは、以下の情報を含んでいます。(左から右)

1 ファイルタイプを示す単一の文字で、選択できるのは次の通りです。

'f' 通常ファイル
'b' ブロック型特殊ファイル
'c' 文字型特殊ファイル
'L' .bul ファイル

2 ファイルへのオフセット。

ファイルタイプが、'b'または'c'の場合、オフセットと長さ(次のフィールド)は、バックアップされるローデバイスの一部を記述します。ファイルタイプが、'f'または'L'である場合、オフセットは適応されません。

file.bul において、オフセットと長さはバイトで表されます。

3 バックアップされる長さ。

ファイルタイプが、'b'または'c'の場合、オフセットと長さ(次のフィールド)は、バックアップされるローデバイスの一部を記述します。ファイルタイプが、'f'または'L'である場合、オフセットは適応されません。

file.bul において、オフセットと長さはバイトで表されます。

4 ファイルへのパス名。

パス名は、絶対パスあるいは相対パスで記述することができます。相対パスは、$DBPATH からの相対パスになります。スクリプトにて最も簡単に処理させるには、ファイルパスに空白文字を含んではいけません。ファイルパスに空白文字がある場合でも、スクリプトは正しくそれらを処理しなければなりません。Unix のシェルスクリプトは、空白文字を含むファイルパスを扱うために、「read」コマンドを使用して変数セット内の文字列を結び付けることができます。

スクリプトは、バックアップが成功したことを示す値「0」で終了しなければなりません。その他の値を返すと、**budb** ユーティリティは操作を失敗した旨のメッセージを表示します。そのメッセージは、file.ral に記録されます。

スクリプトは、**budb** ユーティリティの stdin/stdout/stderr を受け継ぎます。したがって、追加の情報をユーザに対して表示することや、あるいはサードパーティのバックアップ処理を実行している間、ユーザの入力に応ずることができます。

サードパーティユーティリティを使用してバックアップを実行する際、ファイルのパーミッションとオーナーシップが処理の間変更されないように注意してください。

バックアップの間、**budb** ユーティリティはデータベースの同期を取ります。そして、トランザクションログに「backup record」、ログデーモンに「suspends」を書き込み、現在のバックアップバージョンの最終ジャーナルをクローズします。この時点で、スクリプトが実行されます。スクリプトが完了した後、ログデーモンは新しいジャーナルを使用する処理の開始が許可され、再度、データベースの同期が実行されます。

スクリプトを実行する前に、**budb** ユーティリティは以下のような形式のメッセージを標準出力に表示し、file.ral 内にそれを記録します。

```
"Invoking user backup utility '<cmd>'", where <cmd> is the value of the
-Obudb_util option.
```

ユーザユーティリティが終了した後、以下の形式のメッセージが表示され、記録されます。

```
"User backup utility returned N ([error | no error]).
```

例

以下のスクリプトは、BudTool プロダクトを使用するバックアップの例を示します。以下のスクリプトは、サンプルの **budb** コマンド、およびバックアップのテンプレートとして扱えます。

```
#!/bin/sh BudToolBackup.sh

# Sample script to backup files using BudTool 'btbu' utility

build_backup_information_file() {
    # btbu uses backup information file to determine systems, tape, etc

    # create backup information file, starting from template
    cp -f backup_information_template.txt backup_information_file

    # filter out backup number
    read BU_NUM
    echo Backup Number: $BU_NUM
```

```
    ALL_BU_FILES=""

    # read values from $DBPATH/file.bul, one line at a time
    while read FTYP OFFS FLEN BU_FILE; do
        # transfer file names to backup_information_file,
        # except files of type "I"
        if [ ${FTYP} != "I" ]; then
            ALL_BU_FILES="$ALL_BU_FILES $BU_FILE"
        fi
    done

    echo $ALL_BU_FILES >> backup_information_file

    # ensure file ends with a blank line
    echo >> backup_information_file
}

# prepare for the backup

build_backup_information_file < $DBPATH/file.bul

# perform the backup

btbu -i backup_information_file > backup.out 2>&1

# check return value and report result

retvalRsh=$?
if [ ${retvalRsh} -eq 0 ]; then
    echo Backup ok!
    echo
    exit 0
else
    echo Problem backing up files - see file 'btbu.out' for details!
    echo " Return value: " ${retvalRsh}
    echo
    exit 1
fi
```

以下のバックアップコマンドが使用されます。

```
budb -d/space/db/file.db -Obudb_util=BudToolBackup.sh
```

以下のバックアップテンプレートが使用されます。スクリプトは最終行で「¥」のためにテンプレートへファイル名を追加します。

```
merc_dlt2,3|backup_dbsys_db
@BTADMIN:a,
dbsys|root|dbsys:budb|/space|root|1|dump.Solaris|3|0|1|0| ¥
```

redb ユーティリティの変更

redb ユーティリティは、その構文内で新しい引数を利用できます。以下のセクションにおいて、**redb** ユーティリティの変更点を説明します。

構文

redb [-d dbname **] [-Ojrnlwait=***number_of_seconds* **]**
[-Oredb_util=*string* **] [-Obu_num]**

新しい引数

-Oredb_util リストアを実行するスクリプトの名前を含む文字列。文字列には、そのスクリプトへの引数を含めることができます。

文字列は、その実行のためにオペレーティングシステムの system()関数に渡されます。スクリプトは、$DBPATH ディレクトリで実行されます。

-Obu_num スクリプトを実行する前に、-Oredb_util 文字列へ追加されるバックアップバージョン番号を示す整数値。

解説

サードパーティのリストアユーティリティの使用

サードパーティのリストアユーティリティを使用するには、一般的に以下の項目を実行するスクリプトを作成します。

1 バックアップに対して、リストアユーティリティを実行します。

2 エラーのチェックをします。

スクリプトは、リストアが成功したことを示す値「0」で終了しなければなりません。その他の値を返すと、**redb** ユーティリティは操作を失敗した旨のメッセージを表示します。そのメッセージは、file.rel に記録されます。

スクリプトは、**redb** ユーティリティの stdin/stdout/stderr を受け継ぎます。したがって、追加の情報をユーザに対して表示することや、あるいはサードパーティのリストア処理が実行している間、ユーザの入力に応ずることができます。

サードパーティユーティリティを使用してバックアップを実行する際、ファイルのパーミッションとオーナーシップが処理の間変更されないように注意してください。

リストアの間、**redb** ユーティリティは構成ファイル (file.cf) を読み込み、環境を初期化した直後にスクリプトを呼び出します。スクリプトの戻り値が「0」以外の場合 (失敗を示す)、それ以降の処理は実行されません。スクリプトの戻り値が「0」の場合、file.bul にリストされた全てのファイルが存在し、期待されるサイズであることを確認 (デバイスのチェックは含まない) した後に、**redb** ユーティリティは、通常の処理を行います。

1. budb ユーティリティで書きこまれた「buckup record」のポイントまで、リストアされたトランザクションログ (file.rc) に含まれる、コミットされていない全てのトランザクションを取り消します。
2. バックアップ以後に作成されたジャーナルをリプレイします。
3. (利用可能な場合) クラッシュの時点から、完了した全てのトランザクション (トランザクションログの内容から) をやり直します。

スクリプトを実行する前に、**redb** ユーティリティは以下のような形式のメッセージを標準出力に表示し、file.ral 内にそれを記録します。

```
"Invoking user restore utility '<cmd>'", where <cmd> is the value of the
-Oredb_util option.
```

ユーザユーティリティが終了した後、以下の形式のメッセージが表示され、記録されます。

```
"User restore utility returned N ([error | no error]).
```

例

以下のスクリプトは、BudTool プロダクトを使用しているリストアの例を示します。以下のスクリプトは、サンプルの **redb** コマンドとして扱えます。

```
#!/bin/sh BudToolRestore.sh

# Sample script to restore backup files using BudTool 'btr' utility

prepare_restore_list() {
```

```
    cp /dev/null restore_list
    read BU_NUM
    while read FTYP OFFS FLEN BU_FILE; do
        if [ ${FTYP} != “I” ]; then
            # prepend the system name to the file paths
            echo dbsys:$BU_FILE >> restore_list
        fi
    done
}

# restore ‘file.bul’ - redb requires ‘file.bul’ in order to validate restore

# specify the media server and full path to file (including system name)

btr -m merc_dlt2,3 dbsys:/space/db/file.bul > restore.out 2>&1

# check return value and report result

retvalBtr=$?
if [ ${retvalBtr} -eq 0 ]; then
    echo
    echo Restore of ‘file.bul’ ok!
    echo
else
    echo
    echo Unable to restore ‘file.bul’ - see file ‘restore.out’ for details.
    exit 1
fi

# prepare for the restore

prepare_restore_list < $DBPATH/file.bul

# perform the restore - specify the media server and files to restore

btr -m merc_dlt2,3 `cat restore_list` >> restore.out 2>&1

# check return value and report result

retvalBtr=$?
if [ ${retvalBtr} -eq 0 ]; then
    echo Restore ok!
    echo
    exit 0
else
    echo Problem restoring files - see file ‘restore.out’ for details.
    echo
    exit 1
fi
```

以下のリストアコマンドが使用されます。

```
redb -Oredb_util=BudToolRestore.sh
```

新しいエラーメッセージ

以下のエラーメッセージが追加されました。

-11761 redb: Error accessing a transaction log file.

説明: file.lg, file.rlg, file.rc のいずれかをリネーム、オープン、読み込み、書き込みしている際にエラーが発生しました。

訂正: ファイルの存在とパーミッションをチェックしてください。

-11762 budb: Invoking user backup utility <cmdline>

説明: 情報のみ: ネイティブ system()関数に渡される完璧なコマンドラインを示します。

-11763 budb: User backup utility returned N (error | no error).

説明: 情報のみ: ユーザバックアップユーティリティを呼び出した後、ネイティブ system()関数の戻り値を表示します。

-11764 redb: Invoking user restore utility <cmdline>

説明: 情報のみ: ネイティブ system()関数に渡される完璧なコマンドラインを示します。

-11765 redb: User restore utility returned N (error | no error).

説明: 情報のみ: ユーザリストアユーティリティを呼び出した後、ネイティブ system()関数の戻り値を表示します。

-11766 redb: The .bul file cannot be accessed or is corrupt.

説明: file.bul にアクセスしている時にエラーが発生しました。ファイルの存在とパーミッションをチェックしてください。

訂正: file.bul の内容を調べて、問題の場所を見つけてください。

3

SQL/A の新しい演算子

リリース 7.2 より利用可能

SQL/A 文で、「等しくない」条件指定する場合に新しく 「!=」演算子を利用できるようになりました。また、今まで通り「<>」演算子も利用できます。

4

新しいバージョンへの更新

リリース 7.1 より利用可能

remkview を使用してデータベースのビューをコンバート

構文

remkview

引数

なし

解説

remkview ユーティリティは、DBPATH、DBNAME コンフィグレーション変数（または、それらのデフォルト）で指定されたデータベース内の全てのビューを再作成します。データベース内の全てのビューをドロップ, 再作成が必要となるプラットフォームおよびデータベースバージョンに対してのみ、このユーティリティを実行してください。**remkview** を実行する必要がある場合、**dbcnv** ユーティリティは、その旨を通知します。**dbcnv** が **remkview** を実行するよう通知しない場合、**remkview** を実行する必要はありません。

remkview に関連したエラーメッセージ

データベースコンバージョン（**dbcnv** による）を実行した後にデータベースを起動しようとする場合、そのデータベースに対し **remkview** が実行されていないと（**remkview** が必要な場合のみ）、**startdb** は失敗し、以下のエラーを受けます。

The software version does not match the database version. (-6)

remkview 実行時の出力例

以下は、remkview を実行している時に表示される出力の例です。

```
$ $UNIFY/../diag/remkview
Phase I...
Phase II...
Phase III...
Phase IV...
Views successfully re-created.
$
```

Phase Ⅰで、**remkview** はパーミッションのチェックを行い、以降の Phase で処理を実行できるか検証するために、データベースの情報を収集します。PhaseⅡでは、データベース内のビューを実際にドロップしたり、再作成時に実行する SQL 文を作成します。PhaseⅢでは、権限を付与する GRANT 文が作成され、PhaseⅣで、最終的にそれまでに作成した全ての SQL 文が実行されます。

dbcnv を使用してデータベースをコンバート

dbcnv ユーティリティは、データベースを新バージョンにコンバートします。

構文

dbcnv

引数

なし

解説

dbcnv ユーティリティは、DBPATH、DBNAME コンフィグレーション変数（または、それらのデフォルト）で指定されたデータベースを、現在の Unify DataServer リリースに一致するバージョンにコンバートします。

要注意

データベースコンバージョンが必要であるか否かに関わらず、異なるリリースでこのユーティリティを使用する場合は、必ず事前にバックアップを取っておいてください。このユーティリティが失敗すると、バックアップからデータベースをリストアしなければならない場合があります。

dbcnv を使用してデータベースをコンバートしておかなければ、同じプラットフォームにインストールした Unify DataServer の新リリースを使用して既存のデータベースを操作することができないことがあります。データベースは、別々にコンバートしてください。また、**dbcnv** ユーティリティはデータベースが存在するシステム上で実行してください。

環境を新リリースの設定に変更する前に、データベースにマッチする Unify DataServer リリース含まれる **shutdb** を使用して、そのデータベースを停止してください。**dbcnv** を使用する場合、データベースは必ず停止状態にしてください。

dbcnv 実行中、メッセージが表示され、進行状況が通知されます。最後の方に出力されるメッセージは、時々コンバージョンが完了するまでに必要になる追加の作業を記述するとても重要な指示が与えられることがあります。

要注意

これらの処理ステップが完了せず途中終了した場合、例えば、パフォーマンスの劣化, データベースが起動できない, アクセスメソッドコラプションなどの重大な問題を引き起こす可能性があります。

dbcnv に関連したエラーメッセージ

dbcnv を使用してコンバージョンを行っていないデータベースを、新リリースで起動しようとすると、以下のエラーが表示され、新しいデータベースへのコンバージョンが必要であることを通知します。

The software version does not match the database version. (-6)

Unify DataServer ファイル「magic」のエントリにアクセス可能な場合、「file」コマンドを使用して特定のデータベースのバージョン番号を参照することができます。

以下の例では、データベースのバージョンは 37 です。

```
$ file -m $UNIFY/../install/magic $DBPATH/file.db/db/file.db
Unify data base Root volume Version 37 Native Machine
```

データベースのバージョン番号を参照する例

ほとんどの Unify DataServer 実行モジュールで、-version 引数を付けて実行すると、そのプロダクトリリースに対応するデータベースのバージョン番号を参照することができます。

以下の例は、リリースが必要とするデータベースのバージョン番号が 38 であると示しています。

```
$ SQL -version
Database Version: 38
Revision: 7.0B
...
```

dbcnv ユーティリティが必要でない場合であっても、新しい Unify DataServer リリースに切り替えるときは、リリースのライブラリを使用（リンス）する実行形式ファイルがあれば、全て再リンクしてください。

dbcnv ユーティリティは、コンバージョン実行中の主要ステップを順次に表示し、現在のデータベースにどんな変更も加える前に、コンバージョンを続行するか中止するかをユーザに選択させます。

dbcnv を使用してデータベースをコンバートする例

ロケール（locale）に「de」を使用して作成されたバージョン 37 のデータベースを、現在のバージョン 38 にコンバートする場合の **dbcnv** の出力例を以下に示します。

```
$ dbcnv
dbcnv: Checking the data base version.
Making sure the database is not running...
Converting database /my/db/file.db

Conversion from database version 37 to 38 involves the following steps:

        - correct the internal representation of string data in
          access methods. This step requires a rebuild of certain access
          methods for this db locale (de)

Would you like to continue and perform the conversion now? (Y/N) y

Opening the database...
Updating the database version to 38...
There are 2 access methods incompatible with this version of
DataServer. These access methods will be dropped.
A SQL script to recreate them has been written to $DBPATH/remkam.sql
('/my/db/remkam.sql'),
and should be run after the conversion.
Syncing the database...
/ab/ds7/unstable/opus/bin/dbcnv conversion successfully completed.
$
```

dbcnv を使用してデータベースをコンバートする例 (2)

```
$ dbcnv
dbcnv: Checking the data base version.
Making sure the database is not running...
Converting database /myother/db/file.db

Conversion from database version 35 to 38 involves the following steps:

        - add a locale identifier in the root volume of the DB.
        - prepare the database to support multiple volumes for variable
          length data. This could take a while depending on the amount
          of vdata in the database.
        - update database header for this database locale (C).

        To complete the conversion of this database you must shut down
        the database, run diag/remkview to recreate all views, and then
        drop and recreate all of your B-tree indexes.

Would you like to continue and perform the conversion now? (Y/N) y

Opening the database...
Updating the database version to 36...
Updating the database version to 37...
Updating the database version to 38...
Syncing the database...
/prods/ds/bin/dbcnv conversion successfully completed.

NOTE: To complete the conversion of this database
you must shut down the database, run diag/remkview,
and then drop and recreate all of your B-tree indexes.
You can ignore the 'unable to access required file'
error log entries that are written when you shut down.

The B-tree indexes will reside in database volumes
when they are recreated. If they are not already in
database volumes, you may need to add or resize
volumes before recreating your B-trees.
```

WITH CHECK OPTION 述部

CREATE VIEW ... WITH CHECK OPTION によって作成された更新可能ビューの全てについて、今後 Unify DataServer の INSERT/UPDATE/DELETE の各操作は、CREATE VIEW 文で指定された WITH CHECK OPTION 述部に制約されません。

5

ストアードプロシージャとトリガについてのヒント

リリース 7.1 より利用可能

ストアードプロシージャやトリガは、以下の要領にしたがって定義してください。

1. 別のストアードプロシージャを呼び出す場合は、EXTERN 文を使って、そのストアードプロシージャを STORED 型関数として宣言してください。
2. Unify DataServer ACCELL/SQL の CREATE PROCEDURE または CREATE TRIGGER を使用してください。
3. 関数の戻り値の型を VOID と指定するか、または NUMERIC のように Unify DataServer RDBMS のデータ型と互換性のある ACCELL/SQL データ型を指定してください。
4. ストアードプロシージャの関数パラメータはリスト変数やマトリクス変数ではなく、スカラ変数として宣言してください。
5. 全てのストアードプロシージャ宣言において指定されているパラメータの数が、ストアードプロシージャ定義（EXTERN 文または CREATE PROCEDURE 文）と同じ数であることを確かめてください。ストアードプロシージャが、プロシージャ（VOID）として宣言されているものか、それとも指定データ型で値を戻す関数として宣言されているものかを定めるには EXTERN 宣言が必要です。

関連情報

互換データ型のリストは、『*ACCELL/SQL: RDBMS Integration*』の「Using RDBMS Data Types With ACCELL/SQL Applications」を参照してください。

ストアードプロシージャ呼び出し

ストアードプロシージャを 4GL 内の他の関数のように呼び出すことができます。例えば、SET 文を使用してストアードプロシージャから変数への戻り値を割り当てることが可能です。

ストアードプロシージャを呼び出すと、そのストアードプロシージャはカレントトランザクションの下で動作します。あるストアードプロシージャに別のストアードプロシージャが含まれている場合、コンパイラは呼び出されたストアードプロシージャがコンパイル時に使用可能であろうとなかろうと、そのストアードプロシージャをコンパイルします。

ストアド・プロシージャ呼び出しのネスト

ストアードプロシージャやトリガは、ネストすることが可能です。EXECUTINGブロックで複数のストアードプロシージャ呼び出しがネストされている場合、各ストアードプロシージャ呼び出しにEXECUTINGブロックを付けることができ、それぞれの中にシステム関数呼び出しを1つ以上含めることができます。

最大ネストレベルを制御するには、以下のコンフィグレーション変数のどちらかを設定してください。

- ストアードプロシージャの場合は、SPMAXNESTを設定します。
- トリガの場合は、TRIGGERMAXNESTを設定します。

どちらのコンフィグレーション変数においても、ゼロ以外の正の整数が値として有効で、デフォルト値は42です。これらの変数は、コンフィグレーションファイルまたはOSのコマンドレベルから設定することができます。

ストアードプロシージャの実行許可

セキュリティ上の観点から、スキーマに対してではなくスキーマ中の全てのストアードプロシージャに対して GRANT EXECUTE を明示的に実行してください。スキーマに対して GRANT EXECUTE を実行すると、スキーマ中の全てのテーブルに対しても SELECT を許可することになるからです。

トリガの作成についての変更

トリガが失敗（戻り値が 0 以外の場合）すると、返された値に関連したメッセージが表示されます。このため戻り値は正しいエラーコード（負数）でなければなりません。

また、正の戻り値は、Unify DBMS エラーコードとして処理される前に負数に変換されます。

ドキュメントの例では、トリガはエラーの場合に 1 を返します。その結果、-1 のメッセージ（Internal inconsistency - fatal.）が表示されることになります。

6

SELECT 句と UNION 演算子のヒント

リリース 7.1 より利用可能

UNION 演算子で結合された各 SELECT 文の項目は、同じ数であり、同じデータ型でなければなりません。SELECT 句中の CHAR または TEXT 項目の結果列の長さ（ヘッダのような）は、最初の SELECT 文の対応する項目からとられます。残りの SELECT 文の対応する項目の長さは、この結果列の長さ以下でなければなりません。

最初の SELECT 文の項目が文字列定数の場合、結果列の長さは、この文字列定数の長さになります。システム定数 USER や LOGNAME の結果列の長さは、それぞれ 18, 44 になります。

7

CREATE LINK INDEX の失敗を避けるために

リリース 7.0B より利用可能

親テーブル上に参照する行を持たない行が子テーブルに存在すると、CREATE LINK INDEX は失敗します。失敗だと分かるのは、非常に多くのプロセスが実行された後であるため、大きなテーブルを操作している場合は、失敗は作業に少なからずロスを被らせます。

子テーブルの行が参照する親テーブルの行を持たず、あるいは親テーブルのどの行にも一致していない場合、NULL 列でなければその子テーブルの行は、「不適切な親のない子レコード」とみなされます。NULL 列であれば「不適切な行」にはなりません。CREATE LINK INDEX が失敗するのは、「不適切な親のない子レコード」が存在する場合のみです。

計画的に作業を進めることによって、CREATE LINK INDEX の失敗による作業ロスを回避できます。以下に 2 つの手順を紹介します。これらの手順を行うためにデータベースをダウンさせる必要は、ほとんどありません。

手順 1) 親テーブルとその子テーブルを更新するときは、「不適切な親のない子レコード」を新たに作らないでください。すでに「親のない子レコード」が存在しているならば、その位置を確認し、修正してください。(ここではリンクインデックスを作成する前であっても、便宜上、親テーブル、子テーブルという呼び方をしています)

新たな「不適切な親のない子レコード」を作らないための確実な方法は、トリガ, DIS (Data Integrity Subsystem) の利用です。列グループがリンクインデックス内で親の複数列を参照している場合は、通常の環境ではトリガを使用するべきです。リンクインデックスを単独列にのみ作成している場合は、トリガに代わって DIS も使用できます。

何を使うかに関係なく、子により参照されている親の行が削除され、新たな「親のない子レコード」が追加されていないことを確認してください。リンクインデックスを作成することによって生じる制限事項は、リンクインデックスを使用する前にあらかじめ設定してください。

親子間の関係が複数の列をベースに成り立っているときは、トリガを使うべきですが、アプリケーション内で列値に対し制限を加えている場合は、DIS を使う方が適切かもしれません。

親の行を更新する際に親の値を保護する例

以下に示すﾄﾘｶﾞの例は、親に対し更新や削除が行われたときに、親子関係を保つために必要な親の値を保護する方法を示しています。ﾄﾘｶﾞで子の行を検索しているため、子テーブル内で参照されている列に B ツリーインデックスを作成しておくことは重要です。

```
-- Trigger to ensure needed parents will not go away.
--
create trigger on ITEM for before delete, before update as
    (begin
        /* Do children exist that require this parent as it is? */

        set $resrid to select rowid from ITMDETAIL
            where IDtlNAME = old:$INAME
                and IDtlCODE = old:$ICODE;

        if ( $resrid is not undefined ) then
                return(-202);       /* The column cannot be changed due */
                                    /* to references to it. (-202). */
            return (0);             /* The operation is allowed. */
    end);
```

子の行に適切な親の割り当て保証する例

以下の 2 つの例は、この行が「不適切な親のない子レコード」にならないよう保護する方法を示しています。単独列のﾘﾚｰｼｮﾝの場合は、DIS がもっとも簡単な方法です。

DIS ソースコード例:

```
table ITMDETAIL
    IDtlNAME: checkreference (ITEM.INAME)
```

ﾄﾘｶﾞ例:

```
-- Trigger to ensure no improper children are introduced.
--
create trigger on ITMDETAIL for before insert, before update as
    (begin

        /* Is there a parent for the new child values? */
```

```
        set $resrid to select rowid from ITEM
            where INAME = new:$IDtlNAME
                and ICODE = new:$IDtlCODE;

        if ( $resrid is undefined ) then
                return (-192);      /* Attempt to reference nonexistent row */
                                    /* through use of a link (or trigger). */

        return (0);                 /* The operation is allowed. */
    end);
```

上記の DIS を適用すると、子の行の列が不適切な値に変更されようとしたときに、次のエラーメッセージが表示されます。

```
sql> update ITMDETAIL set IDtlNAME = 'NoPar' where ROWID = 5;
recognized update!
Partial success; see the individual status list. (-8)
```

手順 2) 手順 1 が完了したら、手順 2 を始めてください。「不適切な親のない子レコード」の探査と修正です。「不適切な親のない子レコード」のデータ修正方法は、特に指定しません。

以下の例は、子テーブル内の全ての「不適切な親のない子レコード」を探査するための SQL 文です。

```
lines 0;
select IDtlNAME, IDtlCODE, count(*) from ITMDETAIL
    where IDtlNAME is not null
        and IDtlCODE is not null
        and not
            (
            <IDtlNAME, IDtlCODE> in
                (
                select INAME, ICODE from ITEM
                    where INAME is not null and
                        ICODE is not null
                )
            )
group by IDtlNAME, IDtlCODE;
```

複数列のうち 1 個か 2 個の列に有効な値を設定することによって、多数の子の行を修正する場合は、以下の RPT スクリプトが有用です。この RPT スクリプトは、上記の SQL 文の出力結果を入力情報として使っています。

```
input
    IDtlNAME    [string 8],
    IDtlCODE    [string 4],
```

```
    cnt             [numeric 4]

sort   IDtlNAME

length              1
left margin         0
bottom margin       0
top margin          0




before report
    set cwcntr to 0          /* commit work counter */

after IDtlNAME
    print 'update ITMDETAIL set IDtlNAME =',
        1['] + ('0' /+ 1['] ),                  /* '0' */
        'where IDtlNAME =',
        1['] + (IDtlNAME /+ 1['] ),             /* id val in quotes */
        ';'
    set cwcntr to cwcntr + 1

detail                                  /* commit every 100 lines */
    if cwcntr > 100 then
        begin
            set cwcntr to 0
            print 'commit work;'
        end

after report
    print 'commit work;'

end
```

上記 SQL の出力結果を使って、この RPT スクリプトを実行すると、SQL コマンドの実行ファイルが生成されます。そのファイルから数行抜粋します。

```
update ITMDETAIL set IDtlNAME = '0' where IDtlNAME = 'Pebble' ;
update ITMDETAIL set IDtlNAME = '0' where IDtlNAME = 'Rock' ;
update ITMDETAIL set IDtlNAME = '0' where IDtlNAME = 'Stone' ;
commit work;
```

8

新しいコンフィグレーション変数

リリース 7.1 より利用可能

BUCHECKSUM コンフィグレーション変数

バックアップファイルおよびジャーナルファイルのチェックサム機能を有効または無効にするフラグです。BUCHECKSUM が TRUE の場合、チェックサムがバックアップまたはジャーナルファイルに書き込まれ、ファイルがデータベースに読み込まれる際の認証に使用されます。BUCHECKSUM が FALSE の場合、チェックサム機能は無効になります。データベースを復元する際には、「バックアップまたはジャーナルは、チェックサムなしで作成され、チェックサムの認証ができません」旨のメッセージが表示されます。

BUCHECKSUM を設定すると、次のコマンドの実行速度が遅くなる可能性があります：**lgdmn**, **budb**, **redb**, **irma**, **chkbu**, **chkjrn**

遅延割合は、プラットフォームやシステムの負荷に依存します。あるシステムでは、チェックサムを無効にすると、50%以上高速化される場合があります。

BUCHECKSUM は、**chkbu** および **chkjrn** ユーティリティの終了ステータスに対しても変更を及ぼします。

- BUCHECKSUM=FALSE で実行すると、終了ステータス 4（使用エラー）で直ちに終了します。
- BUCHECKSUM=FALSE で作成されたバックアップ／ジャーナルに対して実行すると、終了ステータス 1（チェックサムが見つかりません）で終了します。

有効な値：

TRUE	バックアップ／ジャーナルファイルのチェックサム機能を有効にします。
FALSE	バックアップ／ジャーナルファイルのチェックサム機能を無効にします。
デフォルト	TRUE

RHLIGLOBCOMPAT コンフィグレーション変数

SQL/A オペレータ LIKE および SHLIKE が以下の条件において、RHLI の同様のオペレータ ULIKE および UGLOB と同じ動きになるかどうかを制御するフラグです。

- 問合せが、RHLI スキャンを通して実行される
- 検索パターンの最後に空白が含まれる
- 検索パターンにメタ文字が含まれていない

ヒント

通常、RHLI スキャン問合せは、同様の SQL 問合せよりも高速です。このため Unify DataServer は、問合せを実行するにあたり、RHLI で十分であれば RHLI を使用し実行効率を向上させています。そうでない場合は、問合せは SQL/A バックエンドを通して実行されます。

RHLIGLOBCOMPAT が TRUE の場合、LIKE および SHLIKE は UGLOB または ULIKE を使用した同一の問合せと違った結果を返します。RHLIGLOBCOMPAT が FALSE の場合、LIKE（または SHLIKE）は、RHLI ULIKE（または UGLOB）と同じ結果を返します。

多くの問合せでは、「JOHN」,「JOHNS」,「JOHNSON」,「JOHNSTON」などの名前を検索するために、「JOHN*」のようなパターンを検索で使用します。しかし、RHLIGLOBCOMPAT が TRUE の場合、検索パターン「JOHN 」（メタ文字を含まず、最後に空白文字を含む）は、「JOHN」に一致します。SQL による検索で「JOHN」に一致するのは、問合せが RHLI を通して実行される場合のみです。逆に問い合わせの処理に RHLI が使用されなかった場合には、どの文字列に対しても一致しません。

RHLIGLOBCOMPATがFALSEの場合、SQL, RHLIの双方で、どの文字列もパターンに一致しません。これはデータがテーブルに入力される際、後ろの空白文字が削除されるからです。

有効な値:

TRUE	RHLI ULIKE および UGLOB オペレータの動きは、そのままです。
FALSE	RHLI ULIKE および UGLOB オペレータが、SQL/A オペレータ SHLIKE および LIKE と同じ動きになるようにします。
デフォルト	TRUE

FLTFMT31 コンフィグレーション変数

FLTFMT31 を TRUE に設定することにより、SQL と SQLBE の動作をリリース 3.1 に復元します。

- FLTFMT が定義されている場合（デフォルト「%g」）、FLOAT 値は左揃えになります。
- FLTFMT が定義されている場合、テーブル作成時に設定された DISPLAY コンフィグレーション定義を上書きします。

9

shutdb の新しいオプション

リリース 7.2 より利用可能

shutdb のオプションで-Owait=n は、ユーザプロセスとデータベースデーモンの処理を待つ時間を指定できます。時には、**shutdb** がデーモンプロセスの待ち時間とは別にユーザプロセスを待つ時間をコントロールすることが必要です。例えば、データベース管理者はできるだけ速くデータベースからユーザがでていくことを望む場合、シャットダウンする時間を十分に取るよう、デーモンに許可します。**shutdb** は新しく、-Ousr_wait=n と-Odmn_wait=n という2つのコマンドラインオプションを指定できるようになりました。

-Owait=n と同様に、値は秒数で指定します。-Ousr_wait=n が指定されている場合、-Owait=n オプション（指定されている場合）から得た値が、ユーザプロセスの待ち時間を上書きします。デフォルト値は、60 です。

-Odmn_wait=n を設定することで、lgdmn や cldmn が完了するまでの待ち時間をコントロールします。（これらは、実際にデータベースのログを取る唯一のデーモンです）このオプションの指定がない場合、値は-Owait=n オプションあるいはデフォルト（60）から得られます。

10

2GB を超えるファイルのバックアップ

リリース 7.0A より利用可能

この Unify DataServer に関する機能拡張は、サイズが 2GB を超える単独の通常ファイル（デフォルトは file.bu）にバックアップを作成することが可能になりました。これは、以下のオペレーティングシステムに対応する Unify DataServer について有効です。

- HP-UX 10.20 以降
- IBM RS/6000 AIX 4.3 以降
- Linux 6.1 以降
- SCO UNIXWARE 7.1
- Sequent DYNIX/ptx 4.4.4
- Solaris 2.6 以降
- WindowsNT

2GB を超えるサイズのファイルを実際に作成するには、HP および Solaris では、そのサイズのファイルをサポートできるようにファイルシステムを設定する必要があります。/etc/mount を実行した場合、「largefiles」が表示されない場合はそのバックアップファイルは 2GB を超えることができません。他のオペレーティングシステムでは、異なったキーワードを持っている場合があります。

HP-UX を使用している場合、詳しい情報は「HP-UX Large Files White Page」を参照してください。また、以下のファイルを参照するか、HP-UX ベンダに問い合わせてください。

/usr/share/doc/lg_files.*

バックアップデバイス（BUDEV コンフィグレーション変数を参照）およびジャーナルデバイス（JOURNAL コンフィグレーション変数を参照）には、共に共通する Unify DataServer 内部インターフェースが適用されるため、両者とも 2GB の制限を超えるファイルの作成が可能です。テープデバイスを使用している場合には、2GB を超える内容を書き出すことは以前でも可能でした。

バックアップは頻繁に作成するようにしてください。バックアップの間隔が開きますと、リストア時に非常に多くのトランザクションをロールフォワードしなくてはならず、時間と資源を過度に浪費することに繋がります。例を上げますと、物理ログファイルのサイズは 2GB を超えることができません。物理ログファイルには、更新されたデータベースファイルページが含まれるため、irma がジャーナルからトランザクションをロールフォワードする間に、非常に大きなサイズに膨れ上がります。

ご使用のシステムで、**ulimit** 文およびシステムコールを実行した時に実際設定されているファイルサイズの制限値に関係なく、4194304（2GB）の値が返されることがありますのでご注意ください。実際に 4194304 という制限値に設定されている場合と識別できませんので、**ulimit** が 4194304 を返した時は、2GB を超えるサイズのファイルが本当に作成可能か詳しく調査してください。

11

dbld のパフォーマンスに関するヒント

ﾘﾘｰｽ 7.1 より利用可能

dbld のパフォーマンスは、対象のテーブルに対する BEFORE UPDATE および REFORE INSERT ﾄﾘｶﾞの有無によって大きく変わります。

ﾄﾘｶﾞによる処理が、ロードに際して重要でないのであれば、ロードの前にこのﾄﾘｶﾞをドロップし、**dbld** 終了後に再作成することにより、**dbld** のパフォーマンスを向上させることができます。

12

RPT - 数値入力フィールドでの警告メッセージ

リリース 7.1 より利用可能

数値入力フィールドに 9 よりも大きな長さが指定された場合、以下の警告メッセージが stderr に書き込まれます。（インプットセクションでスクリプトファイルの 2 行目に、MyField [numeric 11] が指定された場合）

```
input
      foo [numeric 11],
      berf [numeric 10]
defail
      print 'hi'
end
```

“RPT scriptname”を実行した時、以下のメッセージが stderr に書き込まれます。

```
RPT - Report Processor
Copyright Unify Corporation 1983, 1984, 1985, 1988.

-----------------------------------------------
Warning: [ numeric 11 ] exceeds the maximum length of this type (9)
on or about line 2
Use a float type if you expect more than 9 digits in this field.
-----------------------------------------------
-----------------------------------------------
Warning: [ numeric 10 ] exceeds the maximum length of this type (9)
on or about line 3
Use a float type if you expect more than 9 digits in this field.
-----------------------------------------------
-----------------------------------------------
syntax error
on or about line 4
```

```
-----------------------------------------------
-----------------------------------------------
There is an invalid input item (-10209)
on or about line 4
-----------------------------------------------
Syntax error(s) were found in the report script.
RPT テーブル使用状況

テーブル名              Used          最大値

式ノード                  0            400
変数                      2            150
定数                      0            250
コマンド                  0            256
プリント命令文            0            125
プリント項目              0            256
ソート項目                0             15
入力項目                  2            100
コマンドグループ          0             25
セット 命令文             0            100
If 命令文                 0             50
総計                      0             50
関数コール                0             50
引数                      0            100
```

入力データがデータファイルで提供されないとき、RPT テーブル使用状況が出力されます。入力データがデータファイルで提供される時、テーブル使用状況は出力されず、以下のメッセージが提供されます。

```
$ RPT rptscpt mydata
-----------------------------------------------
Warning: [ numeric 11 ] exceeds the maximum length of this type (9)
on or about line 2
Use a float type if you expect more than 9 digits in this field.
-----------------------------------------------
-----------------------------------------------
Warning: [ numeric 10 ] exceeds the maximum length of this type (9)
on or about line 3
Use a float type if you expect more than 9 digits in this field.
-----------------------------------------------
-----------------------------------------------
syntax error
on or about line 4
-----------------------------------------------
-----------------------------------------------
There is an invalid input item (-10209)
on or about line 4
-----------------------------------------------
```

```
Syntax error(s) were found in the report script.
```