



EDB

Postgres® for the AI Generation

Migration Toolkit

リリース **55.8.0**

EnterpriseDB Corporation.

2024 年 05 月 31 日

目次

第 1 章 Introduction	3
1.1 Release notes	3
1.1.1 Migration Toolkit 55.8.0 release notes	3
1.1.2 Migration Toolkit 55.7.0 release notes	4
1.1.3 Migration Toolkit 55.6.1 release notes	5
1.1.4 Migration Toolkit 55.6.0 release notes	6
1.1.5 Migration Toolkit 55.5.0 release notes	7
1.1.6 Migration Toolkit 55.4.0 release notes	7
1.1.7 Migration Toolkit 55.3.0 release notes	8
1.1.8 Migration Toolkit 55.2.3 release notes	9
1.1.9 Migration Toolkit 55.2.2 release notes	9
1.1.10 Migration Toolkit 55.2.1 release notes	10
1.1.11 Migration Toolkit 55.2.0 release notes	10
1.1.12 Migration Toolkit 55.1.0 release notes	11
1.1.13 Migration Toolkit 55.0.0 release notes	11
1.2 Supported platforms and databases	12
1.2.1 データベースバージョン	12
1.2.2 サポートされているオペレーティングシステムバージョン	13
1.3 Overview	14
1.3.1 オブジェクト移行のサポート	15
1.3.2 オンライン移行とオフライン移行	16
1.3.3 制限事項	16
第 2 章 Installing and upgrading	19
2.1 Installing Migration Toolkit	19
2.1.1 Linux x86-64 (amd64)	19
2.1.2 Linux Installing Migration Toolkit on Linux IBM Power (ppc64le)	20
2.1.3 Macintosh	20
2.1.4 Windows	20
2.2 Upgrading	40
第 3 章 Using Migration Toolkit	41
3.1 Specifying connection properties	41
3.1.1 Advanced Server URL の定義	42
3.1.2 PostgreSQL URL の定義	43
3.1.3 BigAnimal URL の定義	44
3.1.4 Oracle URL の定義	45
3.1.5 SQL Server URL の定義	46
3.1.6 MySQL URL の定義	47
3.1.7 Sybase URL の定義	48

3.2	Invoking Migration Toolkit	49
3.2.1	バイナリゼロ NULL 文字が埋め込まれた文字データのインポート	50
3.2.2	Oracle からのスキーマの移行	50
3.2.3	Oracle 以外のソースデータベースからの移行	51
第 4 章	Troubleshooting	77
4.1	Migration errors	77
4.1.1	接続エラー	77
4.1.2	移行エラー	79
4.2	Error codes	83
4.2.1	エラーコードの概要	84
4.3	FAQ	92

Migration Toolkit は、移行プロセスの詳細な制御を提供する強力なコマンドラインツールです。Migration Toolkit は、データベースオブジェクトとデータを次の場所から EDB Postgres Advanced Server または PostgreSQL データベースに移行するのに役立ちます。

- Oracle
- MySQL
- Microsoft SQL Server

Migration Toolkit を使用して、データベースオブジェクトとデータを Sybase Adaptive Server Enterprise から EDB Postgres Advanced Server、または EDB Postgres Advanced Server と PostgreSQL 間で移行することもできます。

EDB Postgres Advanced Server インストーラーまたは Stack Builder を使用して Migration Toolkit をインストールできます。Stack Builder は、EDB Postgres Advanced Server と、[EnterpriseDB website](#) から入手可能な PostgreSQL ワンクリックインストーラーの両方とともに配布されています。

注釈: このドキュメントの Postgres という用語は、EDB Postgres Advanced Server または PostgreSQL のインストールを指します。Stack Builder という用語は、StackBuilder Plus EDB Postgres Advanced Server とともに配布 または Stack Builder EnterpriseDB から PostgreSQL ワンクリックインストーラーとともに配布を指します。

第1章 Introduction

1.1 Release notes

Migration Toolkit のドキュメントでは、マイナーリリースとパッチを含む Migration Toolkit55 の最新バージョンについて説明しています。このセクションのリリースノートは、各リリースの新機能に関する情報を提供します。マイナーまたはパッチリリースで導入された新機能の場合、コンテンツ内にどのリリースで機能が導入されたかを示すインジケータもあります。

Version	Release Date
<i>Migration Toolkit 55.8.0 release notes</i>	16 May 2024
<i>Migration Toolkit 55.7.0 release notes</i>	13 Dec 2023
<i>Migration Toolkit 55.6.1 release notes</i>	06 Sep 2023
<i>Migration Toolkit 55.6.0 release notes</i>	25 May 2023
<i>Migration Toolkit 55.5.0 release notes</i>	14 Feb 2023
<i>Migration Toolkit 55.4.0 release notes</i>	29 Nov 2022
<i>Migration Toolkit 55.3.0 release notes</i>	06 Oct 2022
<i>Migration Toolkit 55.2.3 release notes</i>	16 Jun 2022
<i>Migration Toolkit 55.2.2 release notes</i>	10 Mar 2022
<i>Migration Toolkit 55.2.1 release notes</i>	13 Jan 2022
<i>Migration Toolkit 55.2.0 release notes</i>	2 Dec 2021
<i>Migration Toolkit 55.1.0 release notes</i>	20 Sep 2021
<i>Migration Toolkit 55.0.0 release notes</i>	19 Mar 2021

1.1.1 Migration Toolkit 55.8.0 release notes

リリース日 2024 年 5 月 16 日

Migration Toolkit 55.8.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description	Addresses
Enhancement	ターゲットデータベースへの再接続を自動的に試行し、接続が失われたときに移行中であったテーブルのデータ移行を再開またはリスタートする機能を追加しました。	

次のページに続く

表 2 – 前のページからの続き

Type	Description	Addresses
Enhancement	Migration Toolkit では、保存されている `procedure` オブジェクトをソース PostgreSQL からターゲット PostgreSQL データベースに移行できるようになりました。	
Enhancement	オブジェクトの除外機能移行とオプションセット	#99271 / 31314
Enhancement	Migration Toolkit によって開始された接続セッションは、アプリケーション名で識別できるようになりました。これにより、ソースまたはターゲットデータベースに関連する問題のデバッグが容易になります。	
Enhancement	Migration Toolkit は、ソースデータベースから 1 つ以上のテーブルが欠落している場合、エラーメッセージの一部として、欠落しているテーブルの名前をログに記録するようになりました。	#98134 / 30362
Bug fix	空のデフォルト句を含む数値列を含む Oracle テーブルの移行失敗の原因となった問題を修正しました。	#100987 / 32874
Bug fix	ユーザー定義の NOT NULL チェック制約が移行からスキップされる問題を修正しました。	#102630 / 35543
Bug fix	JSON タイプの列を含む MySQL テーブルの移行が失敗する問題を修正しました。	#102690 / 35550
Bug fix	テーブルレベルのパラレルロードが有効になっている <code>dataLoaderCount > 1</code> の場合、パーティションテーブルのデータの移行が重複エラーで失敗する問題を修正しました。	
Bug fix	MySQL テーブル名全体が大文字の場合、大文字と小文字を区別する識別子の処理を修正しました。	

1.1.2 Migration Toolkit 55.7.0 release notes

リリース日 2023 年 12 月 13 日

Migration Toolkit 55.7.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、分散高可用性クラスターを含む BigAnimal でサポートされているすべてのクラスタータイプで動作することが認定されています。

次のページに続く

表 3 – 前のページからの続き

Type	Description
Enhancement	1 回のネットワークラウンドトリップですべてのコンストレインを取得するようにオペレーションを更新することにより、多くのテーブルのテーブルレベルのパラレルロードのパフォーマンスが向上しました。[サポートチケット #97379]
Bug fix	ターゲットの PostgreSQL または EDB Postgres Advanced Server データベースのユーザー作成のトリガーが、データ移行フェーズ中に無効になっていた問題を修正しました。[サポートチケット #97637]
Bug fix	テーブルレベルのパラレルロードモードの無限ループと同じテーブルの再評価を発生させた問題を修正しました。[サポートチケット #96054]
Bug fix	Latin9 キャラクタセットで構成されたソース Oracle データベースからの移行中に、データのコピーが失敗した問題を修正しました。[サポートチケット #99475、#99506]
Bug fix	コピー API がデータのコピーに失敗し、その後の一括挿入モードでデータをコピーしようとする、ターゲットデータベースの重複エラーで失敗していた問題を修正しました。[サポートチケット #99506、#97784]
Bug fix	EDB Postgres Advanced サーバーターゲットデータベースで大文字と小文字が区別される列名を持つソース SQL サーバータブルの移行中に、データのコピーが失敗した問題を修正しました。
Bug fix	大文字と小文字が区別されるテーブル名が原因で、ソース MySQL テーブルの移行中にデータのコピーが失敗する問題を修正しました。
Bug fix	レンジでサブパーティション化されたパーティションテーブルの移行に失敗する原因となった `SUBPARTITION TEMPLATE` 句を使用したパーティションテーブルの問題を修正しました。
Bug fix	ソース EDB Postgres Advanced Server バージョン 16 データベースからの移行に `allgroups` オプションが指定された場合、`capture_admin` データベースシステムグループが含まれていた問題を修正しました。

1.1.3 Migration Toolkit 55.6.1 release notes

リリース日 2023 年 9 月 6 日

Migration Toolkit 55.6.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	テーブルレベルのパラレルロード機能を更新して、フェッチオフセットアプローチの代わりに個々のテーブルデータチャンクをロードするためのレンジベースの基準を使用することにより、ソースデータベースのオーバーヘッドを削減します。この最適化は、テーブルの主キー/一意制約が非複合数値型属性に基づいている場合に適用されます。[サポートチケット #93360]
Bug fix	名前に全角文字が含まれるソース Oracle テーブルを移行するときに、データのコピーが失敗する問題を修正しました。[サポートチケット #93552]

次のページに続く

表 4 – 前のページからの続き

Type	Description
Bug fix	ソース MySQL テーブル名にダッシュ文字が含まれている場合、データコピーが失敗する問題を修正しました。[サポートチケット #95794]
Bug fix	1 つ以上の一重引用符で定義された `DEFAULT` 句を持つソース MySQL テーブルのテーブル作成に失敗する問題を修正しました。[サポートチケット #95794]
Bug fix	ソース MySQL テーブルが名前として MySQL の予約語を使用している場合、データコピーが失敗する問題を修正しました。[サポートチケット #94822]
Bug fix	Microsoft SQL Server から EDB Postgres Advanced Server、Oracle から EDB Postgres Advanced Server、または EDB Postgres Advanced Server から Oracle `dataOnly` への移行中に発生した問題を修正しました。親テーブルは、指定されたテーブルのリストに含まれません。

1.1.4 Migration Toolkit 55.6.0 release notes

リリース日 2023 年 5 月 25 日

Migration Toolkit 55.6 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、移行中にオブジェクトのサブセットを除外できる 2 つの新しいコマンドオプションを提供するようになりました。`-excludeTables` オプションは、移行対象にすべてのテーブルが選択される場合、テーブルのサブセットをスキップし、`-excludeViews` オプションは、移行対象にすべてのビューが選択される場合、ビューのサブセットをスキップします。
Bug fix	Migration Toolkit がシステム生成インデックスをドロップしようとする、データの移行が失敗する問題を修正しました。システム生成されたインデックスは削除からスキップされるようになりました。[サポートチケット #90282]
Bug fix	`XMLTYPE` 列を持つ Oracle テーブルを移行するときに、「無効な LOB ロケーターが指定されました」エラーでデータの移行が失敗する問題を修正しました。[サポートチケット #92284]
Bug fix	ソーステーブルに LOB タイプの列が含まれている場合、SQL Server からデータのみの移行を実行するときに、`TRUNCATE` が常にターゲットデータベースに適用される問題を修正しました。この動作は修正され、ユーザーが `truncLoad` オプションを指定した場合にのみ `TRUNCATE` が適用されるようになりました。
Bug fix	Debian の Migration Toolkit パッケージを更新して、JDBC ドライバーが Migration Toolkit にバンドルされないようにして、Migration Toolkit でサポートされている他の OS プラットフォームのパッケージと一貫性を持たせます。

1.1.5 Migration Toolkit 55.5.0 release notes

リリース日 2023 年 2 月 14 日

Migration Toolkit 55.5 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、さまざまな移行の順列のソースおよび/またはターゲットとして EDB Postgres Advanced Server バージョン 15 をサポートするようになりました。
Enhancement	Migration Toolkit が強化され、ソースデータベースとターゲットデータベースの列の順序が異なる場合、デフォルトの `COPY` を使用したデータの移行をサポートするようになりました。[サポートチケット #88336]
Bug fix	ソース PostgreSQL / EDB Postgres Advanced Server データベースの一意のインデックス制約に基づく外部キー制約がターゲットデータベースへの移行からスキップされる問題を修正しました。[サポートチケット #89491]
Bug fix	並列データのロードモードで 1 つ以上のテーブルの移行が停止する原因になった問題を修正しました。

1.1.6 Migration Toolkit 55.4.0 release notes

リリース日 2022 年 11 月 29 日

Migration Toolkit 55.4 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、PostgreSQL バージョン 15 で動作することが認定されました。PostgreSQL 15 は、さまざまな移行の順列のソースおよび/またはターゲットの両方としてサポートされるようになりました。
Bug fix	複数の MySQL ソーステーブルが同じ制約名を共有している場合、テーブル制約の失敗を引き起こした問題を修正しました。[サポートチケット #86413]
Bug fix	バックスラッシュで終わるテキスト列値を含む MySQL テーブル行のデータの移行が失敗する問題を修正しました。[サポートチケット #86412]
Other	Migration Toolkit で使用する PostgreSQL 10 および EDB Postgres Advanced Server 10 のサポートを削除しました。PostgreSQL および EPAS のこれらのバージョンはサポートが終了し、通常のサポート契約ではサポートされなくなりました。

次のページに続く

表 7 – 前のページからの続き

Type	Description
Other	Migration Toolkit で使用する Oracle 10g のサポートを削除しました。このバージョンは、Migration Toolkit で使用することが以前にテストおよび認定されています。Oracle 10g は引き続き Migration Toolkit で動作する場合がありますが、EDB はその進行中の検証を実行していません。

1.1.7 Migration Toolkit 55.3.0 release notes

リリース日 2022 年 10 月 06 日

Migration Toolkit 55.3 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は、Microsoft が提供する SQL Server JDBC ドライバーバージョン 10.2 で動作することが認定されました。オープンソースの jTDS ドライバーは、Migration Toolkit を使用した Microsoft SQL Server への接続に引き続き使用できますが、代わりに Microsoft 提供の SQL Server JDBC ドライバーを使用することをお勧めします。より新しいバージョンの SQL Server のサポートを提供します。
Enhancement	Migration Toolkit は、Debian 11 プラットフォームで動作することが認定されました。
Security Fix	Migration Toolkit は、 CVE-2022-31197 のセキュリティ修正を含む最新の EDB JDBC ドライバーバージョン 42.5.0.1 と PostgreSQL コミュニティ JDBC ドライバーバージョン 42.5.0 で認定されています。
Security Fix	Migration Toolkit は、 CVE-2022-21363 のセキュリティ修正を含む MySQL JDBC ドライバーバージョン 8.0.30 で動作することが認定されています。Migration Toolkit ドキュメントの MySQL JDBC ドライバーへの参照が、それに応じて更新されました。
Bug Fix	仮想列が含まれる場合、EPAS への Oracle テーブルの移行が失敗する問題を修正しました。[サポートチケット #82718]
Other	これらのバージョンは Microsoft によってサポートされなくなったため、Migration Toolkit で使用する Microsoft SQL Server 2008 および SQL Server Server 2012 のサポートを削除しました。これらのバージョンは、Migration Toolkit で使用することが以前にテストおよび認定されています。これらは引き続き Migration Toolkit で動作する場合がありますが、EDB はそれらの進行中の検証を実行していません。

1.1.8 Migration Toolkit 55.2.3 release notes

リリース日 2022 年 6 月 16 日

Migration Toolkit 55.2.3 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Bug Fix	列定義にデフォルト句として NEXTVAL 呼び出しを含むシーケンスが含まれる場合、EDB Postgres Advanced Server への Oracle テーブルの移行が失敗する問題を修正しました。
Bug Fix	POSITION キーワードにちなんで名前付けられた列を持つテーブルの移行失敗の原因となった問題を修正しました。
Bug Fix	Migration Toolkit Windows インストーラーを修正して、アップグレードプロセスの一部として commons-lang ライブラリの古いバージョンを削除します。
Bug Fix	スキーマ修飾テーブル名をレポートするように一部の情報/エラーメッセージを修正しました。

1.1.9 Migration Toolkit 55.2.2 release notes

リリース日 2022 年 3 月 10 日

Migration Toolkit 55.2.2 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	SQL Server 2016、2017、2019 と Sybase SAP ASE 16.0 のサポートを提供します。
Security Fix	Windows および macOS で使用されるインストーラーが更新され、 CVE-2022-21724 にアドレスする EDB JDBC コネクタの最新バージョンが含まれます。Linux の場合、JDBC ドライバーは Migration Toolkit にパッケージされていないため、この EDB knowledge base article に文書化されている Linux の手順を使用してドライバーを別途更新する必要があります。
Bug Fix	テーブルデータのロード中に ENABLE TRIGGER が失敗した場合、Migration Toolkit は例外を適切に処理するようになりました。特定のテーブルで ENABLE TRIGGER が失敗した場合、残りのテーブルではスキップされません。
Bug Fix	Migration Toolkit は、MySQL データベースからの移行中に、TEXT フィールドに入るバックスラッシュをエスケープするようになりました。

1.1.10 Migration Toolkit 55.2.1 release notes

リリース日 2022 年 1 月 13 日

Migration Toolkit 55.2.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security Fix	Migration Toolkit のロギングは、log4j 2.17.1 に基づいています。以前は、log4j v1 に基づいていました。この変更により、log4j v1 に対して報告された脆弱性が削除され、log4j v2 の古いバージョンに対して発見および報告された RCE 脆弱性がバイパスされるようになります。
Bug Fix	カスタム列タイプのマッピングは、Oracle RAW タイプから PostgreSQL / Postgres Plus Advanced Server UUID タイプに修正されました。
Bug Fix	'-loaderCount' パラメーターは、Sybase、MySQL、および SQL Server データベースでは使用できません。
	NA
Bug Fix	MySQL から PostgreSQL への移行中の重複インデックス名の問題を修正しました。
Bug Fix	MySQL で実行され、結果セットを返さないクエリの executeQuery を executeUpdate に置き換えます。

1.1.11 Migration Toolkit 55.2.0 release notes

リリース日 2021 年 12 月 02 日

Migration Toolkit 55.2.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit は PostgreSQL 14 をサポートすることが認定されています。
Enhancement	新しい Oracle 互換性機能の一部を含む EPAS 14 のサポートを追加しました。
Enhancement	LIST BY PARTITION 句と AUTOMATIC キーワードが続くパーティションテーブルを、ターゲット EPAS 13 および 14 データベースに移行できるようになりました。
Enhancement	Migration Toolkit は、SUBPARTITION TEMPLATE 句を使用してパーティションテーブルをターゲット EPAS 14 データベースに正常に移行できるようになりました。
Bug Fix	ORA-01795 問題を修正して、1,000 を超えるユーザーの移行を成功できるようにしました。
Bug Fix	ORA-01795 問題を修正して、1,000 を超えるパーティションを正常に移行できるようにしました。

1.1.12 Migration Toolkit 55.1.0 release notes

リリース日 2021 年 9 月 20 日

Migration Toolkit 55.1.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Migration Toolkit が機能強化され、MySQL 5.7 および 8.0 からの移行をサポートします。
Bug Fix	MySQL から PostgreSQL への移行、ブールデータ型は LO モードで移行されませんでした。この問題は修正されました。
Bug Fix	問題を修正して、数字で始まるテーブル名を MySQL から PostgreSQL へ移行できるようにしました。
Bug Fix	MySQL から PostgreSQL に移行するときに、同じタイムゾーン内で TIMES-TAMP 値が変更される問題を修正しました。
Bug Fix	MySQL から PostgreSQL への移行中、高速コピーモードではターゲットデータベースに列挙型が作成されませんでした。この問題は修正されました。
Bug Fix	n > 1 の場合、MySQL から PostgreSQL への BIT(n) データ型の移行に関連する障害を修正しました。
Bug Fix	Oracle から PostgreSQL への移行中に、数値フォーマット例外が報告された問題を修正しました 72863
Bug Fix	関連する OID 値が INTEGER 範囲を超えると、PostgreSQL テーブルの移行は失敗します。この問題は修正されました 72694
Bug Fix	PostgreSQL から Oracle への JSON データ型の移行に関連する問題を修正しました。
Bug Fix	Oracle 19c から EDB Postgres Advanced Server の場合、特定のシステムユーザー/スキーマが移行のために誤って選択されました。この問題は修正されました。
Bug Fix	問題を修正して、パラレルロードモードで大文字と小文字を区別するスキーマ/テーブル名を Oracle から EDB Postgres Advanced Server に移行できるようにしました。
Bug Fix	[カスタム列タイプマッピング] オプションの使用に関連する特定のコーナーケースが修正されています。

1.1.13 Migration Toolkit 55.0.0 release notes

リリース日 2021 年 3 月 19 日

Migration Toolkit 55.0.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	テーブルレベルでの並列データのロードのサポート。これにより、マルチプルのスレッドを使用して、ソースデータベーステーブルから並列にデータをロードし、ターゲットデータベーステーブルに並列に適用できます。これにより、比較的大きなサイズのテーブルを処理する場合のデータのロード時間が大幅に向上します。ラボベンチマークの結果、マルチプルの並列スレッドを使用する場合、大規模なテーブルのデータのロード時間が 25～60%削減されることがわかりました。この機能は、ソース PostgreSQL、EDB Postgres Advanced Server、および Oracle データベースでサポートされています。 並列データのロードのための移行オプション を参照してください。
Enhancement	テーブルレベルの並列データのロード機能の強化の一部として、`tableLoaderLimit` および `parallelLoadRowLimit` オプションがコマンドラインオプションリストに追加されました。
Enhancement	Oracle バージョン 18c および 19c からのスキーマとデータの移行のサポート。
Enhancement	スキーマとデータ移行セッションの合計移行時間は、`Total Elapsed Migration Time` 属性を介して報告されます。
Bug Fix	テーブルのデータの移行が失敗した場合、概要レポートはステータスを正確にレポートしません。
Bug Fix	概要レポートは、冗長な無効なオブジェクトを削除しません。
Bug Fix	指数形式で報告されるテーブルのデータサイズ。
Bug Fix	重複するテーブル名が `tables` リストを介して指定された場合の障害を回避します。
Bug Fix	以前は 0.0 MB として表示されていた小さなテーブルサイズを反映します。

1.2 Supported platforms and databases

1.2.1 データベースバージョン

Migration Toolkit では、次のデータベース製品バージョンを使用できます。

- PostgreSQL バージョン 12、13、14、15、および 16
- EDB Postgres Advanced Server バージョン 12、13、14、15、および 16
- Oracle 11g
- Oracle 12c
- Oracle 18c
- Oracle 19c
- SQL Server 2014
- SQL Server 2016
- SQL Server 2017
- SQL Server 2019

- MySQL 5.5
- MySQL 5.7
- MySQL 8.0
- Sybase SAP Adaptive Server Enterprise 15.7
- Sybase SAP Adaptive Server Enterprise 16.0

注釈: BigAnimal シングルノード、プライマリ/スタンバイ高可用性、および分散高可用性クラスタータイプとして使用可能なすべての PostgreSQL および EDB Postgres Advanced Server バージョンは、Migration Toolkit での使用もサポートされています。BigAnimal の supported cluster types の詳細については、BigAnimal EDB の管理対象データベースクラウドサービス documentation を参照してください。BigAnimal で使用可能な PostgreSQL および EDB Postgres Advanced Server のバージョンについては、database version policy documentation を参照してください。

スーパーユーザーロールは、BigAnimal EDB Postgres Advanced Server 分散高可用性クラスターデータベースでは使用できません。その結果、BigAnimal 分散高可用性クラスターをターゲットとする場合、スーパーユーザーアクセスを必要とする次の Migration Toolkit 機能はサポートされません。

- db_link_ora 拡張機能を使用してデータの移行を有効にする `-copyViaDBLinkOra` オプション
- Oracle 互換 SQL コマンド `CREATE DATABASE LINK`

注釈: Migration Toolkit は、Oracle Real Application Clusters RAC および Exadata での使用をテストしておらず、公式にサポートしていません。ただし、Migration Toolkit は、単一の永続ノードに接続されている場合に動作する場合があります。詳細については、EDB 担当者に連絡してください。!!!

1.2.2 サポートされているオペレーティングシステムバージョン

Migration Toolkit は、Linux、Windows、および MacOS プラットフォームでのインストールをサポートしています。詳細は、[Product Compatibility](#) を参照してください。

注釈: ojdbc7.jar は、ResultSet からの取得中に Integer.MAX_VALUE 2147483647 を超えるレコードの制限のため、大規模なデータの移行が不完全になる場合があります。この問題を回避するには、ojdbc8.jar にアップグレードすることをお勧めします。

1.3 Overview

Migration Toolkit は、移行プロセスの詳細な制御を提供する強力なコマンドラインツールです。Migration Toolkit の使用は、2 段階のプロセスです。

1. `toolkit.properties` ファイルを編集して、ソースデータベースとターゲットデータベースを指定します。
2. コマンドラインで Migration Toolkit を呼び出し、移行オプションを指定します。

Migration Toolkit は、データベースオブジェクトとデータを次の場所から EDB Postgres Advanced Server または PostgreSQL データベースへの移行に役立ちます。

- Oracle
- MySQL
- SQL Server

Migration Toolkit を使用すると、データベースオブジェクトとデータを Sybase から EDB Postgres Advanced Server データベースに移行することもできます。Migration Toolkit を使用して、EDB Postgres Advanced Server と PostgreSQL 間で移行することもできます。Migration Toolkit には多くのオプションが含まれており、移行プロセスを詳細に制御できます。

- `-safeMode` オプションを使用して、移行時に各行をコミットします。
- `-fastCopy` オプションを使用して WAL ロギングをバイパスし、移行を最適化します。
- `-batchSize` オプションを使用して、一括挿入のバッチサイズを制御します。
- `-cpBatchSize` オプションを使用して、COPY コマンドで使用するバッチサイズを指定します。
- `-lobBatchSize` オプションを使用して、ラージオブジェクトデータ型に使用されるバッチサイズを指定します。
- `-filterProp` オプションを使用して、指定した条件を満たす行のみを移行します。
- `-customColTypeMapping` オプションを使用して、選択した列のデータ型を変更します。
- 移行前に `-dropSchema` オプションを使用して、既存のスキーマを削除し、新しいスキーマを作成します。
- EDB Postgres Advanced Server では、`-allDBLinks` オプションを使用してすべての Oracle データベースリンクを移行します。
- EDB Postgres Advanced Server では、`-copyViaDBLinkOra` オプションを使用して `dblink_ora` モジュールを有効にします。
- `-connRetryCount <connection_attempts>` オプションを使用して、データの移行中にターゲットデータベース接続が失われた場合に実行する再試行回数を指定します。
- `-connRetryInterval <seconds>` オプションを使用して、データ移行中に各ターゲットデータベースに再接続を試行する前に待機する秒数を指定します。
- `-abortConnOnFailure <true/false>` オプションを使用して、データの移行中にすべてのターゲットデータベースの再接続が失敗した場合に、移行を中止するかどうかを指定します。デフォルトは

`true` で、指定された `-connRetryCount` しきい値の後に接続が失敗した場合にセッションを中止します。

- `-pgIdleTxSessionTimeOut <seconds>` を使用して、MTK 接続セッションの PostgreSQL または EDB Postgres Advanced Server `idle_in_transaction_session_timeout` 構成オプションの値をオーバーライドします。

1.3.1 オブジェクト移行のサポート

Migration Toolkit は、オブジェクト定義 DDL、テーブルデータ、またはその両方を移行します。次の表には、Migration Toolkit できるデータベースオブジェクトのタイプのプラットフォーム固有のリストが含まれています。

Object	Oracle	Sybase	SQL Server	MySQL
Schemas	X	X	X	X
Tables	X	X	X	X
List-partitioned table	X			
Range-partitioned table	X			
Hash partitioned table	X			
Constraints	X	X	X	X
Indexes	X	X	X	X
Triggers	X			
Table data	X	X	X	X
Views	X		X	
Materialized views	X			
Packages	X			
Procedures	X			
Functions	X			
Sequences	X			
Users/Roles	X			
Profiles	X			
Object types	X			
Object type methods	X			
Database links	X			
Queues	X			

インポートされるオブジェクトの詳細な制御を提供するコマンドの詳細については、

[スキーマオブジェクト選択オプション](#) を参照してください。

1.3.2 オンライン移行とオフライン移行

Migration Toolkit は、Postgres データベースにすぐに直接移行できます**オンライン移行**。後で Postgres データベースでオブジェクト定義を再作成するために使用するスクリプトを生成することもできます**オフライン移行**。

デフォルトでは、Migration Toolkit は Postgres データベースにオブジェクトを直接作成します。または、**-offlineMigration** オプションを含めて、移行されたオブジェクトまたはデータを新しいデータベースに再現するために後で使用できる SQL スクリプトを生成します。移行ツールキットを実行する前に、Migration Toolkit によって生成された移行スクリプトをカスタマイズすることにより、移行されたオブジェクトを変更できます。**-offlineMigration** オプションを使用すると、システム負荷に最適なときに実際の移行をスケジュールできます。

-offlineMigration オプションの詳細については、
[オフライン移行オプション](#) を参照してください。

1.3.3 制限事項

EDB Postgres Advanced Server は、一部の Oracle 機能の完全なサポートと、他の機能の部分的なサポートを提供します。Migration Toolkit は、サポートされていない機能を使用するオブジェクトを移行できません。

場合によっては、Migration Toolkit は、部分的な互換性を提供する機能を使用するオブジェクトを移行できます。他の場合、EDB Postgres Advanced Server は適切な回避策をサポートします。

全文検索は、Oracle と完全な互換性のない機能の例です。EDB Postgres Advanced Server データベースには、かなり前から全文検索のサポートが含まれていますが、実装は Oracle のものとは異なります。Migration Toolkit は、この機能を使用するオブジェクトを移行できません。

EDB Postgres Advanced Server は、他の機能をまだサポートしていません。このカテゴリの機能には、自動ストレージ管理、テーブル圧縮、外部テーブルが含まれます。多くの場合、成功する回避策を実装できます。

- 自動ストレージ管理をシステム固有のボリューム管理ソフトウェアに置き換えることができます。
- 圧縮ファイルシステムにあるテーブルスペースにデータを保存することにより、テーブル圧縮を実装できます。
- 外部テーブルは EDB Postgres Advanced Server には存在しませんが、フラットテキストファイルをデータベースのステー징テーブルにロードすることができます。EDB*Loader ユーティリティを使用して、データを EDB Postgres Advanced Server データベースにすばやくロードすることをお勧めします。
- 複数のプロファイルを Oracle データベースから EDB Postgres Advanced Server に移行する場合、移行の完了時にプロファイルをユーザーに手動で割り当てる必要があります。

サポートされていない **Postgres** 機能

Migration Toolkit は、次の **Postgres** 機能の移行をサポートしていません。

- **OPERATOR CLASS**
- **OPERATOR FAMILY**

OPERATOR CLASS および **OPERATOR FAMILY** の詳細については、

[PostgreSQL core documentation](#) を参照してください。

第2章 Installing and upgrading

2.1 Installing Migration Toolkit

リンクを選択して、該当するインストール手順にアクセスします。

2.1.1 Linux x86-64 (amd64)

Red Hat Enterprise Linux RHEL とその派生

- *Installing Migration Toolkit on RHEL 9 ppc64le*、*Installing Migration Toolkit on RHEL 8 ppc64le*、*Installing Migration Toolkit on RHEL 7 or OL 7 x86_64*
- *Oracle Linux (OL) 9*、*Oracle Linux (OL) 8*、*Oracle Linux (OL) 7*
- *Installing Migration Toolkit on AlmaLinux 9 or Rocky Linux 9 x86_64*、*Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on AlmaLinux 9 or Rocky Linux 9 x86_64*、*Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on CentOS 7 x86_64*

SUSE Linux Enterprise SLES

- *Installing Migration Toolkit on SLES 15 ppc64le*、*Installing Migration Toolkit on SLES 12 ppc64le*

Debian とデリバティブ

- *Installing Migration Toolkit on Ubuntu 22.04 x86_64*、*Installing Migration Toolkit on Ubuntu 20.04 x86_64*
- *Installing Migration Toolkit on Debian 11 x86_64*、*Installing Migration Toolkit on Debian 10 x86_64*

2.1.2 Linux Installing Migration Toolkit on Linux IBM Power (ppc64le)

Red Hat Enterprise Linux RHEL とその派生

- *Installing Migration Toolkit on RHEL 9 ppc64le*、*Installing Migration Toolkit on RHEL 8 ppc64le*

SUSE Linux Enterprise SLES

- *Installing Migration Toolkit on SLES 15 ppc64le*、*Installing Migration Toolkit on SLES 12 ppc64le*

2.1.3 Macintosh

- *Installing on Mac OS X*

2.1.4 Windows

- *Windows Server 2019*
- *Windows Server 2022*
- *Windows 11*

Installing Migration Toolkit on Linux x86 (amd64)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise Linux RHEL とその派生

- *Installing Migration Toolkit on RHEL 9 ppc64le*
- *Installing Migration Toolkit on RHEL 8 ppc64le*
- *Installing Migration Toolkit on RHEL 7 or OL 7 x86_64*
- *Oracle Linux (OL) 9*
- *Oracle Linux (OL) 8*
- *Oracle Linux (OL) 7*
- *Installing Migration Toolkit on AlmaLinux 9 or Rocky Linux 9 x86_64*
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*
- *Installing Migration Toolkit on AlmaLinux 9 or Rocky Linux 9 x86_64*
- *Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64*

- *Installing Migration Toolkit on CentOS 7 x86_64*

SUSE Linux Enterprise SLES

- *Installing Migration Toolkit on SLES 15 ppc64le*
- *Installing Migration Toolkit on SLES 12 ppc64le*

Debian とデリバティブ

- *Installing Migration Toolkit on Ubuntu 22.04 x86_64*
- *Installing Migration Toolkit on Ubuntu 20.04 x86_64*
- *Installing Migration Toolkit on Debian 11 x86_64*
- *Installing Migration Toolkit on Debian 10 x86_64*

Installing Migration Toolkit on RHEL 9 or OL 9 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on RHEL 8 or OL 8 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on AlmaLinux 9 or Rocky Linux 9 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on AlmaLinux 8 or Rocky Linux 8 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on RHEL 7 or OL 7 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。
 - その他の前提条件に対処します。

```
# Install the EPEL repository:  
sudo yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.  
↪noarch.rpm
```

パッケージをインストールする

```
sudo yum -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on CentOS 7 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。

2. EDB リポジトリへのアクセスを提供するボタンを選択します。
 3. ダウンロードするプラットフォームとソフトウェアを選択します。
 4. EDB リポジトリの設定手順に従ってください。
- EPEL リポジトリをインストールします。

```
sudo yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.  
→noarch.rpm
```

パッケージをインストールする

```
sudo yum -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 15 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
 2. EDB リポジトリへのアクセスを提供するボタンを選択します。
 3. ダウンロードするプラットフォームとソフトウェアを選択します。
 4. EDB リポジトリの設定手順に従ってください。
- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/15.4/x86_64
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 12 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/12.5/x86_64  
sudo SUSEConnect -p sle-sdk/12.5/x86_64
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Ubuntu 22.04 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Ubuntu 20.04 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Debian 11 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Debian 10 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo apt-get -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on Linux IBM Power (ppc64le)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise Linux RHEL

- [Installing Migration Toolkit on RHEL 9 ppc64le](#)
- [Installing Migration Toolkit on RHEL 8 ppc64le](#)

SUSE Linux Enterprise SLES

- [Installing Migration Toolkit on SLES 15 ppc64le](#)
- [Installing Migration Toolkit on SLES 12 ppc64le](#)

Installing Migration Toolkit on RHEL 9 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on RHEL 8 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

```
sudo dnf -y install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 15 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
 2. EDB リポジトリへのアクセスを提供するボタンを選択します。
 3. ダウンロードするプラットフォームとソフトウェアを選択します。
 4. EDB リポジトリの設定手順に従ってください。
- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/15.4/ppc64le
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing Migration Toolkit on SLES 12 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは1回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. [EDB repositories](#) に移動します。
 2. EDB リポジトリへのアクセスを提供するボタンを選択します。
 3. ダウンロードするプラットフォームとソフトウェアを選択します。
 4. EDB リポジトリの設定手順に従ってください。
- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/12.5/ppc64le  
sudo SUSEConnect -p sle-sdk/12.5/ppc64le
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

```
sudo zypper -n install edb-migrationtoolkit
```

初期構成

Migration Toolkit を呼び出す前に、ソースおよびターゲットデータベースに接続するための JDBC ドライバーをダウンロードしてインストールする必要があります。詳細は、[Installing a JDBC driver](#) を参照してください。

Installing on Mac OS X

EDB は、MacOS 用のグラフィカルインタラクティブインストーラーを提供します。2つの方法でアクセスできます。

- [Downloads page](#) からグラフィカルインストーラーをダウンロードし、インストーラーを直接呼び出します。[直接インストールする](#) を参照してください。
- PostgreSQL で Stack Builder を使用して、EDB インストーラーパッケージをダウンロードし、グラフィカルインストーラーを起動します。[スタックビルダーを使用する](#) を参照してください。

前提条件

Stack Builder が Migration Toolkit のインストールを実行する前に、Java JVM バージョン 1.8.0 以降を導入する必要があります。[Adoptium](#) は、完全なオープンソースから事前にビルドされた OpenJDK バイナリを提供します。

JAVA_HOME 環境変数も設定する必要があります。次のコマンドを使用します。

```
export JAVA_HOME=$(/usr/libexec/java_home)
```

スタックビルダーを使用する

PostgreSQL を使用している場合、Stack Builder でグラフィカルインストーラーを呼び出すことができます。[Using Stack Builder](#) を参照してください。

1. Stack Builder で、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、ドロップダウンリストボックスを使用して、使用可能なサーバーのリストからターゲットサーバーインストールを選択します。ネットワークでインターネットにアクセスするためにプロキシサーバーを使用する必要がある場合は、**Proxy servers** を選択してサーバーを指定します。プロキシサーバーを使用する必要がない場合は、**Next** をクリックして続行します。

1. **Registration-required and rotationproducts** ノードを展開します。
2. **EnterpriseDB Tools** ノードを展開し、**Migration Toolkit** を選択します。
3. **Next** を選択し、[グラフィカルインストーラーを使用する](#)に進みます。

直接インストールする

1. グラフィカルインストーラーをダウンロードした後、ターミナルを使用して次のコマンドを実行します。必要に応じて、プロンプトが表示されたらパスワードを入力します。スーパーユーザーまたは管理者権限が必要です。

```
sudo ~/Downloads/edb-migrationtoolkit-55.5.0-1-osx.app/Contents/MacOS/installbuilder.  
→ sh
```

1. [グラフィカルインストーラーを使用する](#)に進みます。

グラフィカルインストーラーを使用する

1. インストール言語を選択し、**OK** を選択します。
2. Setup Migration Toolkit ページで、**Next** を選択します。
3. ライセンス契約を読みます。契約に同意する場合は、**I accept the Agreement** を選択し、**Next** を選択します。
4. Migration Toolkit をインストールするディレクトリを参照するか、インストーラーがデフォルトの場所に Migration Toolkit をインストールすることを許可します。**Next** を選択します。
5. [Ready to install] ページで、[**Next**] を選択します。

情報ボックスには、選択したコンポーネントのインストールの進行状況が表示されます。

1. インストールが完了したら、[**Finish**] を選択します。

Migration Toolkit のインストール後、移行を実行する前に適切な JDBC ドライバーをインストールする必要があります。詳細については、[Installing a JDBC driver](#) を参照してください。

注釈: EDB JDBC ドライバーは、Mac OS では使用できません。

Installing on Windows

EDB は、Windows のグラフィカル対話型インストーラーを提供します。2つの方法でアクセスできます。

- [Downloads page](#) からグラフィカルインストーラーをダウンロードし、インストーラーを直接呼び出します。[直接インストールする](#) を参照してください。
- Stack Builder PostgreSQL を使用する場合は StackBuilder Plus を使用する EDB Postgres Advanced Server を使用して、EDB インストーラーパッケージをダウンロードし、グラフィカルインストーラーを起動します。[Stack Builder または StackBuilder Plus を使用する](#) を参照してください。

前提条件

Stack Builder が Migration Toolkit のインストールを実行する前に、Java JVM バージョン 1.8.0 以降を導入する必要があります。[Adoptium](#) は、完全なオープンソースから事前にビルドされた OpenJDK バイナリを提供します。

Java 実行可能ファイルは、検索パス Windows の %PATH% にある必要があります。Windows では、次のコマンドを使用して検索パスを設定し、javadir の Java 実行可能ファイルを保持するディレクトリの名前を置き換えます。

```
SET PATH=javadir;%PATH%
```

注釈: インストールプロセスは、この場所 `C:\Program Files\edb\mktcsysconfig\edbmtk-<version>.config` に構成ファイルを作成します。この構成ファイルは、Java 実行可能ファイルの場所を指定します。システムに複数のバージョンの Java がインストールされている場合は、`edbmtk-<version>.config` の内容を変更して、Migration Toolkit が正しいバージョンを使用していることを確認できます。!!!

直接インストールする

グラフィカルインストーラーをダウンロードした後、インストールウィザードを開始するには、十分な特権スーパーユーザーまたは管理者を引き受け、インストーラーアイコンをダブルクリックします。プロンプトが表示されたら、パスワードを入力します。

Windows の一部のバージョンでは、管理者権限でインストーラーを起動するには、インストーラーアイコンを右クリックし、コンテキストメニューから **Run as Administrator** を選択する必要があります。

このトピックの [グラフィカルインストーラーを使用する](#) セクションに進みます。

Stack Builder または StackBuilder Plus を使用する

PostgreSQL を使用している場合、Stack Builder でグラフィカルインストーラーを呼び出すことができます。[Using Stack Builder](#) を参照してください。

1. Stack Builder で、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、ドロップダウンリストボックスを使用して、使用可能なサーバーのリストからターゲットサーバーインストールを選択します。ネットワークでインターネットにアクセスするためにプロキシサーバーを使用する必要がある場合は、**Proxy servers** を選択してサーバーを指定します。プロキシサーバーを使用する必要がある場合は、**Next** をクリックして続行します。

1. **Registration-required and rotationproducts** ノードを展開します。
2. **EnterpriseDB Tools** ノードを展開し、**Migration Toolkit** を選択します。
3. [グラフィカルインストーラーを使用する](#) に進みます。

EDB Postgres Advanced Server を使用している場合、StackBuilder Plus でグラフィカルインストーラーを起動できます。[Using StackBuilder Plus](#) を参照してください。

1. StackBuilder Plus では、モジュール選択ページが表示されるまでプロンプトに従います。

[Welcome] ページで、ドロップダウンリストボックスを使用して、使用可能なサーバーのリストからターゲットサーバーインストールを選択します。ネットワークでインターネットにアクセスするためにプロキシサーバーを使用する必要がある場合は、**Proxy servers** を選択してサーバーを指定します。プロキシサーバーを使用する必要がある場合は、**Next** をクリックして続行します。

1. **アドオン、ツール、およびユーティリティ** ノードを展開し、**Enterprise DB Migration Toolkit** を選択します。
2. [グラフィカルインストーラーを使用する](#) に進みます。

グラフィカルインストーラーを使用する

1. インストール言語を選択し、**OK** を選択します。
2. Setup Migration Toolkit ページで、**Next** を選択します。
3. ライセンス契約を読みます。契約に同意する場合は、**I accept the Agreement** オプションを選択し、**Next** を選択します。
4. Migration Toolkit をインストールするディレクトリを参照するか、インストーラーがデフォルトの場所に Migration Toolkit をインストールすることを許可します。**Next** を選択します。
5. [Ready to install] ページで、[**Next**] を選択します。

情報ボックスには、選択したコンポーネントのインストールの進行状況が表示されます。これには数分かかる場合があります。

1. インストールが完了したら、[**Finish**] を選択します。

Migration Toolkit のインストール後、移行を実行する前に適切な JDBC ドライバーをインストールする必要があります。詳細については、[Installing a JDBC driver](#) を参照してください。

Installing a JDBC driver

Migration Toolkit には、Java バージョン 1.8.0 以降が必要です。

ドライバーの選択

どの JDBC ドライバーを使用するかは、移行元または移行先のデータベースによって異なります。

- EDB Postgres Advanced Server との間で移行する場合は、EDB JDBC ドライバーを使用してください。最新のドライバーをダウンロードするには、EDB Downloads ページの [EDB Connectors](#) を参照してください。インストール手順については、Installing and configuring EDB JDBC Connector を参照してください。

注釈: EDB JDBC ドライバーは、Mac OS では使用できません。

- PostgreSQL へ、または PostgreSQL から移行する場合は、PostgreSQL JDBC ドライバーを使用します。サポートされている最新のドライバーをダウンロードするには、PostgreSQL のダウンロードページの [JDBC drivers section](#) を参照してください。
- Oracle、MySQL、Microsoft SQL Server または Sybase から移行する場合は、無料で入手可能なソース固有の JDBC ドライバーを使用してください。
- [Oracle JDBC](#)
- [MySQL JDBC](#)
- [Microsoft SQL Server JDBC](#)
- [Sybase ASE \(jTDS\) JDBC](#)

注釈: オープンソース jTDS ドライバーは、古いバージョンの Microsoft SQL Server バージョン 2012 以前もサポートし、Migration Toolkit で使用できます。Microsoft が提供する SQL Server 用の JDBC ドライバーは、Migration Toolkit でサポートされている SQL Server の新しいバージョンに推奨されます。

lib へのドライバーの追加

ドライバーをダウンロードした後、ドライバーファイルを<mtk_install_dir>/lib ディレクトリに移動します。

!!!note “EDB JDBC ドライバーの場合” Migration Toolkit をインストールする前に EDB JDBC ドライバーがインストールされている場合、そのコピーを<mtk_install_dir>/lib ディレクトリに配置する必要はありません。edb-jdbc18.jar ファイルへのシンボリックリンクは、Migration Toolkit のインストール中に作成されます。

Migration Toolkit のインストール後、次のコマンドを実行して、シンボリックリンクが存在するかどうかを確認できます。

```
ls -l /usr/edb/migrationtoolkit/lib/edb-jdbc18.jar
```

2.2 Upgrading

最初に現在の EnterpriseDB リポジトリへのアクセスを有効にする `edb.repo` ファイルをアップグレードし、次に Migration Toolkit をアップグレードすることにより、既存の Linux インストールをアップグレードできます。

`edb.repo` ファイルをアップグレードするには

```
# Update your repository configuration file

sudo <package-manager> -y upgrade edb-repo

# Upgrade the installed product

sudo <package-manager> -y edb-migrationtoolkit
```

`<package-manager>` は、オペレーティングシステムで使用するパッケージマネージャーです。

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
yum	RHEL 7 and derivatives, CentOS 7
zypper	SLES
apt-get	Debian and Ubuntu

第3章 Using Migration Toolkit

3.1 Specifying connection properties

Migration Toolkit は、移行プロセス中に `toolkit.properties` ファイルに保存されている構成と接続情報を使用して、ソースデータベースとターゲットデータベースを識別して接続します。

Linux では、`toolkit.properties` ファイルは `/usr/edb/migrationtoolkit/etc` にあります。

Windows では、ファイルは `C:\Program Files\edb\mtk\etc` にあります。

Mac OS では、ファイルは `/Library/edb/mtk/etc` にあります。

この図は、サンプル `toolkit.properties` ファイルを示しています。

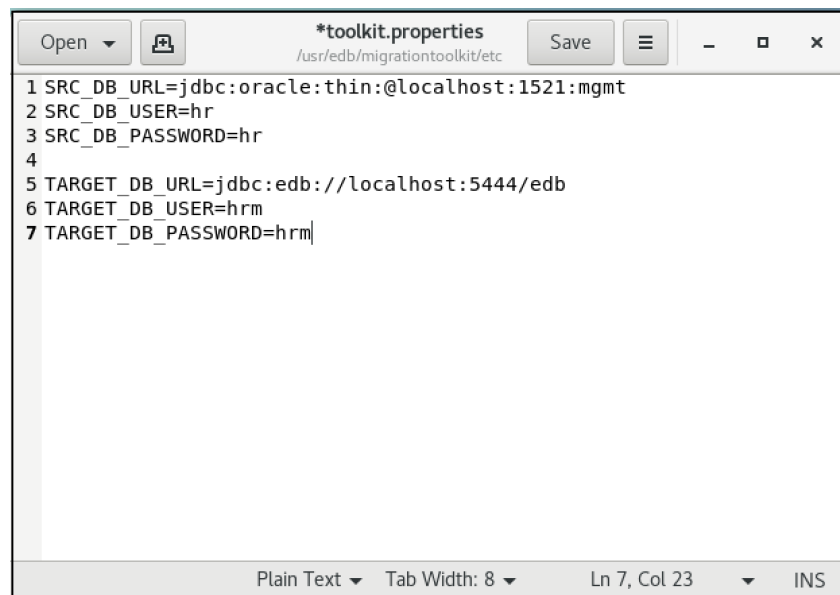


図 1: A typical toolkit.properties file.

Migration Toolkit コマンドを実行する前に、選択したエディターで `toolkit.properties` ファイルを変更します。ファイルを更新して、次の情報を含めます。

- `SRC_DB_URL` は、Migration Toolkit がソースデータベースに接続する方法を指定します。各ソースデータベースの URL の形成の詳細は次のとおりです。
- `SRC_DB_USER` は、ソースデータベース内の十分な特権を持つユーザー名を指定します。
- `SRC_DB_PASSWORD` は、ソースデータベースユーザーのパスワードを指定します。
- `TARGET_DB_URL` は、ターゲットデータベースの JDBC URL を指定します。

- TARGET_DB_USER は、特権ターゲットデータベースユーザーの名前を指定します。
- TARGET_DB_PASSWORD は、ターゲットデータベースユーザーのパスワードを指定します。

注釈: コマンドラインで指定しない限り、Migration Toolkit はソースデータベースが Oracle であり、ターゲットデータベースが EDB Postgres Advanced Server であると想定します。他のソースまたはターゲットデータベースの場合、[Oracle 以外のソースデータベースからの移行](#)の説明に従って `-sourcedbtype` または `-targetdbtype`` オプションを指定します。

BigAnimal でターゲットデータベースを指定する場合は、
[BigAnimal URL の定義](#) を参照してください。

3.1.1 Advanced Server URL の定義

Migration Toolkit は、次のプラットフォームから Advanced Server への移行に役立ちます。

- Oracle
- PostgreSQL
- MySQL
- Sybase
- SQL Server

各データベースタイプから移行されたオブジェクトの最終的なリストについては、
[Functionality overview](#) を参照してください。

Migration Toolkit は、`toolkit.properties` ファイルからソースデータベースとターゲットデータベースの接続仕様を読み取ります。それぞれの接続情報には次のものが含まれる必要があります。

- データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

URL は JDBC 標準に準拠し、次のフォームです。

`{TARGET_DB_URL|SRC_DB_URL}=jdbc:edb://<host>:<port><database_id>`

Advanced Server URL には、次の情報が含まれています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `edb` — Advanced Server を使用している場合、サブプロトコル値に `edb` を指定します。
- `<host>` — Postgres インスタンスが実行されているホストの名前または IP アドレス。
- `<port>` — Advanced Server データベースリスナーが監視しているポート番号。デフォルトのポート番号は 5444 です。

- `<database_id>` — ソースまたはターゲットデータベースの名前。
- `{TARGET_DB_USER|SRC_DB_USER}` — 移行する各種別のオブジェクトを作成する権限を持つユーザーを指定します。データをテーブルに移行する場合、指定されたユーザーは、各ターゲットテーブルの挿入、切り捨て、および参照特権も必要とする場合があります。
- `{TARGET_DB_PASSWORD|SRC_DB_PASSWORD}` — Advanced Server 特権ユーザーのパスワードに設定します。

3.1.2 PostgreSQL URL の定義

Migration Toolkit は、次のプラットフォームから PostgreSQL への移行に役立ちます。

- Oracle
- Advanced Server
- MySQL
- SQL Server

各データベースタイプから移行されたオブジェクトの最終的なリストについては、

[Functionality overview](#) を参照してください。

Migration Toolkit は、`toolkit.properties` ファイルからソースデータベースとターゲットデータベースの接続仕様を読み取ります。それぞれの接続情報には次のものが含まれる必要があります。

- データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

PostgreSQL URL は JDBC 標準に準拠し、次のフォームです。

```
{SRC_DB_URL|TARGET_DB_URL}=jdbc:postgresql://<host>:<port>/<database_id>
```

URL には次の情報が含まれています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `postgresql` — PostgreSQL を使用している場合、サブプロトコル値に `postgresql` を指定します。
- `<host>` — Postgres インスタンスが実行されているホストの名前または IP アドレス。
- `<port>` — Postgres データベースリスナーが監視しているポート番号。デフォルトのポート番号は 5432 です。
- `<database_id>` — ソースまたはターゲットデータベースの名前。
- `{SRC_DB_USER|TARGET_DB_USER}` — 移行する各種別のオブジェクトを作成する権限を持つユーザーを指定します。データをテーブルに移行する場合、指定されたユーザーは、各ターゲットテーブルの挿入、切り捨て、および参照特権も必要になる場合があります。
- `{SRC_DB_PASSWORD|TARGET_DB_PASSWORD}` — PostgreSQL 特権ユーザーのパスワードに設定します。

3.1.3 BigAnimal URL の定義

Migration Toolkit は、次のプラットフォームから BigAnimal への移行に役立ちます。

- Oracle
- Advanced Server
- PostgreSQL
- MySQL
- SQL Server

Migration Toolkit は、`toolkit.properties` ファイルからソースデータベースとターゲットデータベースの接続仕様を読み取ります。それぞれの接続情報には次のものが含まれる必要があります。

- データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

BigAnimal に移行する場合、`TARGET_DB_URL` は JDBC URL のフォームを取ります。例

```
jdbc:<postgres_type>://<host_name>[:<port>]/<database_id>?sslmode=<sslmode>
```

注釈: ターゲットデータベース URL に必要な値の多くは、BigAnimal ポータルから入手できます。BigAnimal で、クラスターを選択し、[**Connect**] タブに移動して値を見つけます。 !!!

URL には次の情報が含まれています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `postgres_type` — サブプロトコルは `Postgres` タイプです。Advanced Server を使用している場合は `edb` を指定し、PostgreSQL を使用している場合は `postgresql` を指定します。
- `<host_name>` — クラスターのホスト名。BigAnimal ポータルの **Connect** タブの **Host** フィールドからコピーできます。
- `<port>` — データベースリスナーが監視しているポート番号。BigAnimal ポータルの **Connect** タブの **Port** フィールドからコピーできます。
- `<database_id>` — ターゲットデータベースの名前。これを、移行ターゲットデータベースとして使用するクラスター内のデータベースの名前に設定します。クラスターのデフォルトデータベースの名前は、BigAnimal ポータルの **Connect** タブの **Dbname** フィールドに表示されます。多くの場合、移行ターゲットとして使用するために別のデータベースが作成されます。
- `TARGET_DB_USER` — 特権データベースユーザーの名前を指定します。BigAnimal ポータルの **Connect** タブの **User** フィールドからコピーできます。
- `TARGET_DB_PASSWORD` — 指定されたユーザーのパスワードが含まれます。

- `sslmode` — “require” または “verify-full” のいずれか。Recommended settings for SSL mode を参照してください。BigAnimal ポータルの **Connect** タブの **Service URI** 値の最後にリストされています。

3.1.4 Oracle URL の定義

Migration Toolkit は、Oracle データベースから PostgreSQL または Advanced Server データベースへの移行に役立ちます。Oracle から移行する場合、`toolkit.properties` ファイルで Oracle ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- Oracle データベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

Oracle データベースから移行する場合、`SRC_DB_URL` には、2 つの形式のいずれかで指定される JDBC URL が含まれる必要があります。最初のフォームは次のとおりです。

```
jdbc:oracle:thin:@<host_name>:<port>:<database_id>
```

2 番目のフォームは次のとおりです。

```
jdbc:oracle:thin:@//<host_name>:<port>{<database_id|service_name>}
```

Oracle URL には、次の情報が含まれています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `oracle` — サブプロトコルは常に `oracle` です。
- `thin` — ドライバーのタイプ。 `thin` のドライバータイプを指定します。
- `<host_name>` — Oracle サーバーが実行されているホストの名前または IP アドレス。
- `<port>` — Oracle データベースリスナーが監視しているポート番号。
- `<database_id>` — Oracle データベースのデータベース SID。
- `<service_name>` — Oracle サービスの名前。
- `SRC_DB_USER` — 特権 Oracle ユーザーの名前を指定します。Oracle ユーザーが Oracle から Advanced Server にオブジェクトを移行するには DBA 権限が必要です。Oracle `GRANT DBA TO user` コマンドを使用して Oracle ユーザーに DBA 特権を付与して、目的のデータベースオブジェクトをすべて移行します。
- `SRC_DB_PASSWORD` — 指定されたユーザーのパスワードが含まれます。

3.1.5 SQL Server URL の定義

Migration Toolkit は、SQL Server データベースから PostgreSQL または Advanced Server データベースへの移行に役立ちます。Migration Toolkit は、次のオブジェクト定義の移行をサポートしています。

- スキーマ
- テーブル
- テーブルデータ
- コンストレイン
- インデックス

Migration Toolkit は、`toolkit.properties` ファイルからソースデータベースの接続仕様を読み取ります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

SQL Server データベースに接続する場合、`SRC_DB_URL` は、

Microsoft JDBC URL 推奨または *JTDS URL* のフォームを取ります。

Microsoft JDBC URL

デフォルトでは、SQL Server 用の Microsoft JDBC Driver は、クライアントと SQL Server 間のすべての通信に TLS 暗号化を使用します。TLS 暗号化を無効にする場合は、`encrypt` を `false` に設定します。JDBC タイプ 4 ドライバーを使用して Microsoft SQL Server への接続の詳細については、[Building the connection URL](#) を参照してください。

`jdbc:sqlserver://<server_name>:<port>[;databaseName=<database_id>][;encrypt=false]`

SQL サーバー URL には、次の情報が含まれています。

- `jdbc` — プロトコルは常に `jdbc` です。
- `sqlserver` — サーバータイプは常に `sqlserver` です。
- `<server_name>` — ソースサーバーが実行されているホストの名前または IP アドレス。
- `<port>` — ソースデータベースリスナーが監視しているポート番号。
- `<database_id>` — ソースデータベースの名前。
- `SRC_DB_USER` — 特権 SQL Server ユーザーの名前を指定します。
- `SRC_DB_PASSWORD` — 指定されたユーザーのパスワードが含まれます。

JTDS URL

```
jdbc:jtds:sqlserver://<server_name>:<port>/<database_id>
```

Tip: JTDS ドライバーは、古いバージョンの Microsoft SQL Server をサポートするオープンソースの JDBC 3.0 タイプ 4 ドライバーです。 <http://jtds.sourceforge.net/> を参照してください。新しいバージョンの Microsoft SQL Server を Migration Toolkit に接続する場合、SQL Server 用の Microsoft JDBC Driver をお勧めします。

SQL サーバー URL には、次の情報が含まれています。

- jdbc — プロトコルは常に jdbc です。
- jtds — ドライバー名は常に jtds です。
- sqlserver — サーバータイプは常に sqlserver です。
- <server_name> — ソースサーバーが実行されているホストの名前または IP アドレス。
- <port> — ソースデータベースリスナーが監視しているポート番号。
- <database_id> — ソースデータベースの名前。
- SRC_DB_USER — 特権 SQL Server ユーザーの名前を指定します。
- SRC_DB_PASSWORD — 指定されたユーザーのパスワードが含まれます。

3.1.6 MySQL URL の定義

Migration Toolkit は、MySQL データベースから Advanced Server または PostgreSQL データベースへの移行に役立ちます。MySQL から移行する場合、`toolkit.properties` ファイルで MySQL ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

MySQL から移行する場合、SRC_DB_URL は JDBC URL のフォームを取ります。例

```
jdbc:mysql://<host_name>[:<port>]/<database_id>
```

URL には次の情報が含まれています。

- jdbc — プロトコルは常に jdbc です。
- mysql — サブプロトコルは常に mysql です。
- <host_name> — ソースサーバーが実行されているホストの名前または IP アドレス。
- [<port>] — MySQL データベースリスナーが監視しているポート番号。

- <database_id> —ソースデータベースの名前。
- SRC_DB_USER —特権 MySQL ユーザーの名前を指定します。
- SRC_DB_PASSWORD —指定されたユーザーのパスワードが含まれます。

注釈:

- データ型 `tinyInt(1)` を使用して MySQL ソースデータベースに 0 と 1 以外のバイト値を保存する場合、MySQL Connector/J JDBC ドライバーの URL にオプションのパラメーター `?tinyInt1isBit=false` を追加してください。

-
- MySQL Connector/J JDBC ドライバー 8.0.26 のバグのため、特定の場合に外部キーコンストレインの移行が失敗します。EDB では、Migration Toolkit を使用したデータの移行にこのドライバを使用しないことをお勧めします。代わりに、MySQL Connector/J JDBC Driver 8.0.30 以降を使用してください。

このバグの詳細については、[MySQL bug report](#) を参照してください。

TINYINT(1) は、PostgreSQL/ EDB Postgres Advanced Server で BIT(1) にマッピングされますこれは場合によっては予期されない場合があります。しかし、MySQL JDBC ドライバーはそれを BIT(1) として報告するため、Migration Toolkit は PostgreSQL/ EDB Postgres Advanced Server の BIT(1) にマッピングします。

値-128～127 を保持できる TINYINT(1) とは異なり、BIT(1) は 0 と 1 のみを保持できます。したがって、このフィールドに 0 と 1 以外の値を保存する場合、MySQL JDBC ドライバーのデフォルト動作を変更する必要がある場合がありますドライバがそれを TINYINT として報告するようにします。

この動作を変更するには、接続文字列で次の URL オプションを設定します。

```
jdbc:mysql://localhost:3306/world?tinyInt1isBit=false
```

この場合、JDBC ドライバーは TINYINT(1) を TINYINT として報告し、ターゲット PostgreSQL/ EDB Postgres Advanced Server の SMALLINT にマッピングされます。

3.1.7 Sybase URL の定義

Migration Toolkit は、Sybase データベースから Advanced Server データベースへの移行に役立ちます。Sybase から移行する場合、`toolkit.properties` ファイルで Sybase ソースデータベースの接続仕様を指定する必要があります。接続情報には以下を含める必要があります。

- ソースデータベースの URL
- 特権ユーザーの名前
- 指定されたユーザーに関連付けられたパスワード

Sybase から移行する場合、SRC_DB_URL は JTDS URL のフォームを取ります。例

```
jdbc:jtds:sybase://<host_name>[:<port>]/<database_id>
```

Tip: Sybase ASE および古いバージョンの Microsoft SQL Server 用のオープンソース JDBC 3.0 タイプ 4 ドライバーについては、<http://jtds.sourceforge.net/> を参照してください。

Sybase URL には、次の情報が含まれています。

- jdbc — プロトコルは常に jdbc です。
- jtds — ドライバー名は常に jtds です。
- sybase — サーバータイプは常に sybase です。
- <host_name> — ソースサーバーが実行されているホストの名前または IP アドレス。
- <port> — Sybase データベースリスナーが監視しているポート番号。
- <database_id> — ソースデータベースの名前。
- SRC_DB_USER — 特権 Sybase ユーザの名前を指定します。
- SRC_DB_PASSWORD — 指定されたユーザーのパスワードが含まれます。

3.2 Invoking Migration Toolkit

Migration Toolkit をインストールし、*toolkit.properties file* でソースデータベースとターゲットデータベースの接続プロパティを指定すると、Migration Toolkit は移行を実行する準備が整います。

Migration Toolkit 実行可能ファイルの名前は、Linux システムでは `runMTK.sh`、Windows システムでは `runMTK.bat` です。Linux システムでは、実行可能ファイルは次の場所にあります。

```
/usr/edb/migrationtoolkit/bin
```

Windows では、実行可能ファイルは次の場所にあります。

```
C:\Program Files\edb\mtk\bin
```

移行の詳細の制御については、*Migration Toolkit command options* を参照してください。

注釈: Migration Toolkit の呼び出し時に次のエラーが表示された場合は、*toolkit.properties* ファイルのファイル許可を確認してください。

```
MTK-11015: The connection credentials file ../etc/toolkit.properties is
not secure and accessible to group/others users. This file contains
plain passwords and should be restricted to Migration Toolkit owner user
only.
```

Migration Toolkit を実行するオペレーティングシステムのユーザーアカウントは、ファイルに対する最小限の読み取り権限を持つ *toolkit.properties* ファイルの所有者である必要があります。さらに、グループおよび他のユーザーにはいかなる種類の権限も存在しない必要があります。次に、推奨されるファイル許可の例を示します。ここで、ユーザー `enterprisedb` は Migration Toolkit を実行しています。

```
-rw----- 1 enterprisedb enterprisedb 191 Aug 1 09:59 toolkit.properties
```

3.2.1 バイナリゼロ NULL 文字が埋め込まれた文字データのインポート

Migration Toolkit は、値が NULL の列のインポートをサポートしています。ただし、Migration Toolkit は、JDBC 接続プロトコルを使用した NULL 文字値埋め込みバイナリゼロ 0x00 のインポートをサポートしていません。NULL 文字を含むデータをインポートする場合は、`-replaceNullChar` オプションを使用して、NULL 文字を単一の NULL 以外の置換文字に置き換えます。

注釈:

- MTK は、NULL 文字を空の文字列に暗黙的に置き換えます。 `-replaceNullChar` オプションは、`-copyViaDBLinkOra` オプションとは併用できません。
-

データが移行されたら、SQL ステートメントを使用して、`-replaceNullChar` で指定された文字をバイナリゼロに置き換えます。

3.2.2 Oracle からのスキーマの移行

コマンドラインで指定しない限り、Migration Toolkit はソースデータベースが Oracle であり、ターゲットデータベースが EDB Postgres Advanced Server であることを想定しています。Linux で完全なスキーマを移行するには、実行可能ファイルに移動し、次のコマンドを呼び出します。

```
$ ./runMTK.sh <schema_name>
```

Windows で完全なスキーマを移行するには、実行可能ファイルに移動し、次のコマンドを呼び出します。

```
> .\runMTK.bat <schema_name>
```

そこで

`<schema_name>` は、移行する `toolkit.properties` ファイルで指定されたソースデータベースのスキーマの名前です。少なくとも 1 つの `schema_name` 値を含める必要があります。

注釈:

- 移行されたスキーマのデフォルトのデータベースユーザーが自動的に移行されると、カスタムプロファイルが存在する場合、デフォルトのデータベースユーザーのカスタムプロファイルも移行されます。カスタムプロファイルは、ユーザーが作成するプロファイルです。たとえば、カスタムプロファイルには、Oracle プロファイル `DEFAULT` および `MONITORING_PROFILE` が除外されます。
- PostgreSQL のデフォルト行のサイズは 8KB に制限されています。これは、テーブルの各行が単一の 8 KB ブロックに収まる必要があることを意味します。それ以外の場合、エラーが発生して、たとえば、1600 列の INT でテーブルを作成し、すべての列にデータを挿入できることを示します。ただし、INT は 4 バイトとして保存されるが、各 BIGINT はより多くの領域 8 バイトを必要とするため、BIGINT 列で同じことはできません。詳細については、PostgreSQL ドキュメントの [PostgreSQL Limits](#) を参照してください。

コマンド名の後にスキーマ名のコンマ区切りリストを続けることにより、複数のスキーマを移行できます。

Linux では、次のコマンドを実行します。

```
$ ./runMTK.sh <schema_name1>,<schema_name2>,<schema_name3>
```

Windows では、次のコマンドを実行します。

```
> .\runMTK.bat <schema_name1>,<schema_name2>,<schema_name3>
```

3.2.3 Oracle 以外のソースデータベースからの移行

ソースデータベースタイプとターゲットデータベースタイプを指定しない場合、Postgres はソースデータベースが Oracle であり、ターゲットデータベースが EDB Postgres Advanced Server であると想定します。

Migration Toolkit を呼び出すには、コマンドウィンドウを開き、実行可能ファイルに移動して、次のコマンドを呼び出します。

```
$ ./runMTK.sh -sourcedbtype <source_type> -targetdbtype <target_type> [options, ...]
<schema_name>;
```

```
-sourcedbtype <source_type>
```

<source_type> は、ソースデータベースのサーバータイプを指定します。<source_type> は大文字と小文字を区別しません。デフォルトでは、<source_type> は oracle です。source_type は、次のいずれかの値です。

To migrate from:	Specify:
Oracle	oracle (the default value)
MySQL	mysql
SQL Server	sqlserver
Sybase	sybase
PostgreSQL	postgres or postgresql
EDB Postgres Advanced Server	enterprisedb

```
-targetdbtype <target_type>
```

<target_type> は、ターゲットデータベースのサーバータイプを指定します。<target_type> は大文字と小文字を区別しません。デフォルトでは、<target_type> は enterprisedb です。<target_type> は、次のいずれかの値です。

To migrate to:	Specify:
EDB Postgres Advanced Server	enterprisedb

次のページに続く

表 2 – 前のページからの続き

To migrate to:	Specify:
PostgreSQL	postgres or postgresql

<schema_name>

<schema_name> は、移行する toolkit.properties ファイルで指定されたソースデータベースのスキーマの名前です。少なくとも 1 つの<schema_name> 値を含める必要があります。

次の例では、HR という名前のスキーマテーブル定義とテーブルコンテンツを、Linux システム上の MySQL データベースから EDB Postgres Advanced Server ホストに移行します。コマンドには、- sourcedbtype および -targetdbtype オプションが含まれます。

Linux では、次のコマンドを使用します。

```
$ ./runMTK.sh -sourcedbtype mysql -targetdbtype enterprisedb HR
```

Windows では、次のコマンドを使用します。

```
> .\runMTK.bat -sourcedbtype mysql -targetdbtype enterprisedb HR
```

Migration Toolkit コマンドの最後にスキーマのコンマ区切りリストを含めることにより、ソースデータベースから複数のスキーマを移行できます。次の例では、複数のスキーマ HR および ACCTG と呼ばれますを MySQL データベースから PostgreSQL データベースに移行します。

Linux では、次のコマンドを使用します。

```
$ ./runMTK.sh -sourcedbtype mysql -targetdbtype postgres HR,ACCTG
```

Windows では、次のコマンドを使用します。

```
> .\runMTK.bat -sourcedbtype mysql -targetdbtype postgres HR,ACCTG
```

Migration Toolkit command options

移行の詳細を制御するには、Migration Toolkit を実行するときに移行オプションを追加します。たとえば、データベース内のすべてのスキーマを移行するには、コマンドに -allSchemas オプションを追加します。

```
$ ./runMTK.sh -allSchemas
```

注釈:

- -allSchemas パラメーターは、Oracle、EDB Postgres Advanced Server、および PostgreSQL ソースデータベースでのみサポートされています。 Sybase、MS SQL Server、および MySQL ソースデータベースではサポートされていません。 - Migration Toolkit は、データの移行中にユーザー作成のトリガーを無効にします。ただし、PostgreSQL/ EDB Postgres Advanced Server は、ALTER TABLE の `DISABLE TRIGGER USER` 句を使用して継承された子パーティショントリガーを無効にすることを許可していないため、トリガーはパーティションテーブルのデータの移行中にアクティブになり、予期しない結果が発生する可能性があります。

表に示すように、Migration Toolkit で動作するコマンドオプションは動作によってグループ化されています。

Feature	Relevant options
オフライン移行オプション	-offlineMigration
インポートオプション	-sourcedbtype, -targetdbtype, -schemaOnly, -dataOnly
スキーマ作成オプション	-dropSchema, -targetSchema
スキーマオブジェクト選択オプション	-allTables, -tables, -excludeTables, -constraints, -ignoreCheckConstFilter, -skipCKConst, -skipFKConst, -skipColDefaultClause, -indexes, -triggers, -allViews, -views, -excludeViews, -allSequences, -sequences, -excludeSequences, -allProcs, -procs, -excludeProcs, -allFuncs, -funcs, -excludeFuncs, -checkFunctionBodies, -allPackages, -packages, -excludePackages, -allDomains, -allQueues, -queues, -excludeQueues, -allRules, -allgroups, -groups
オフライン移行オプション	-truncLoad, -enableConstBeforeDataLoad, -retryCount, -safeMode, -fastCopy, -analyze, vacuumAnalyze, -replaceNullChar, -copyDelimiter, -batchSize, -cpBatchSize, -lobBatchSize, -fetchSize, -filterProp, -customColTypeMapping, -customColTypeMappingFile
接続再試行オプション	-connRetryCount, -connRetryInterval, -abortOnConnFailure
Oracle 固有のオプション	-allUsers, -users, -allProfiles, -profiles, -importPartitionAsTable, -objectTypes, -copyViaDBLinkOra, -allDBLinks, -allSynonyms, -allPublicSynonyms, -excludeSynonyms, -allPrivateSynonyms, -useOraCase, -skipUserSchemaCreation
その他のオプション	-help, -logDir, -logFileCount, -logFileSize, -logBadSQL -verbose, -version
並列データのロードのための移行オプション	-loaderCount, -parallelLoadRowLimit, -tableLoaderLimit

オフライン移行オプション

コマンドラインで `-offlineMigration` オプションを指定すると、Migration Toolkit はオフライン移行を実行します。オフライン移行中に、Migration Toolkit は選択された各オブジェクトの定義を読み取り、後で実行時に Postgres の各オブジェクトをレプリケートする SQL スクリプトを作成します。

注釈: 次の例は、Linux で Migration Toolkit を呼び出します。Windows で Migration Toolkit を呼び出すには、`runMTK.sh` コマンドの代わりに `runMTK.bat` コマンドを使用します。

スキーマとデータの両方のオフライン移行を実行するには、`- offlineMigration` キーワードに続いてスキーマ名を指定します。

```
$ ./runMTK.sh -offlineMigration <schema_name>
```

各データベースオブジェクト定義は、ホームフォルダーのスキーマ名とオブジェクトタイプから派生した名前での別のファイルに保存されます。代替のファイルの宛先を指定するには、`- offlineMigration` オプションの後にディレクトリ名を含めます。

```
$ ./runMTK.sh -offlineMigration <file_dest> <schema_name>
```

スキーマオブジェクトのみのオフライン移行空のテーブルの作成を実行するには、Migration Toolkit を呼び出すときに `- offlineMigration` キーワードに加えて `- schemaOnly` キーワードを指定します。

```
$ ./runMTK.sh -offlineMigration -schemaOnly <schema_name>
```

スキーマオブジェクト定義を省略してデータのみのオフライン移行を実行するには、Migration Toolkit を呼び出すときに `- dataOnly` キーワードと `- offlineMigration` キーワードを指定します。

```
$ ./runMTK.sh -offlineMigration -dataOnly <schema_name>
```

デフォルトでは、データは COPY 形式で書き込まれます。データをプレーン SQL 形式で書き込むには、`- safeMode` キーワードを含めます。

```
$ ./runMTK.sh -offlineMigration -dataOnly -safeMode <schema_name>
```

デフォルトでは、テーブルデータを含むオフライン移行を実行すると、テーブルごとに別個のファイルが作成されます。複数のテーブルのデータを含む単一のファイルを作成するには、`- singleDataFile` キーワードを指定します。

```
$ ./runMTK.sh -offlineMigration -dataOnly -singleDataFile -safeMode <schema_name>
```

注釈: `-singleDataFile` オプションは、プレーン SQL 形式でデータを移行する場合にのみ使用できます。`- singleDataFile` オプションを含める場合、`-safeMode` キーワードを含める必要があります。

オフライン移行スクリプトの実行

edb-psql または psql コマンドラインを使用して、オフライン移行中に生成されたスクリプトを実行できます。次の例では、EDB Postgres Advanced Server に保存されている新しいデータベース `acctg` という名前にスキーマ `hr` という名前を復元する方法を説明します。

1. `createdb` コマンドを使用して、`acctg` データベースを作成します。ここに、移行したデータベースオブジェクトを復元します。

```
$ createdb -U enterprisedb acctg
```

2. `edb-psql` を使用して新しいデータベースに接続します。

```
$ edb-psql -U enterprisedb acctg
```

3. \i メタコマンドを使用して、オブジェクト定義を作成する移行スクリプトを呼び出します。

```
$ acctg=# \i ./mtk_hr_ddl.sql
```

4. `-offlineMigration` コマンドに `-singleDataFile` キーワードが含まれている場合、`mtk_hr_data.sql` スクリプトには、新しいターゲットデータベースのすべてのオブジェクトを再作成するために必要なコマンドが含まれています。次のコマンドを使用してデータベースにデータを設定します。

```
$ acctg=# \i ./mtk_hr_data.sql
```

インポートオプション

デフォルトでは、Migration Toolkit はソースデータベースが Oracle であり、ターゲットデータベースが EDB Postgres Advanced Server であると想定しています。 `-sourcedbtype` および `-targetdbtype` キーワードを含めて、デフォルト以外のソースまたはターゲットデータベースを指定します。

デフォルトでは、Migration Toolkit はスキーマの移行時にデータとオブジェクト定義の両方をインポートします。または、データまたはオブジェクト定義のインポートを選択できます。

`-sourcedbtype <source_type>`

`-sourcedbtype` オプションは、ソースデータベースタイプを指定します。 `source_type` の場合、次の値のいずれかを使用します。 `mysql`、`oracle`、`sqlserver`、`sybase`、`postgresql`、または `enterprisedb`。 `source_type` は大文字と小文字を区別しません。デフォルトでは、`source_type` は `oracle` です。

`-targetdbtype <target_type>`

`-targetdbtype` オプションは、ターゲットデータベースタイプを指定します。 `target_type` の場合、`enterprisedb`、`postgres`、または `postgresql` のいずれかの値を使用します。 `target_type` は大文字と小文字を区別しません。デフォルトでは、`target_type` は `enterprisedb` です。

`-schemaOnly`

このオプションはスキーマ定義をインポートし、選択したすべてのスキーマオブジェクトをターゲットデータベースに作成します。このオプションは `-dataOnly` オプションとともに使用できません。

`-dataOnly`

このオプションはデータのみをコピーします。 `-tables` オプションとともに使用する場合、Migration Toolkit は選択されたテーブルのデータのみをインポートします。このオプションは `-schemaOnly` オプションとともに使用できません。

スキーマ作成オプション

デフォルトでは、Migration Toolkit はソーススキーマオブジェクトまたはデータを同じ名前のスキーマにインポートします。ターゲットスキーマが存在しない場合、Migration Toolkit はスキーマを作成します。または、`-targetSchema` オプションを使用してカスタムスキーマ名を指定できます。次のオプションを使用して、既存のスキーマを削除し、新しいスキーマを作成することを選択できます。

`-dropSchema [true|false]`

このオプションを `true` に設定すると、Migration Toolkit は既存のスキーマとそのスキーマ内のオブジェクトを削除し、新しいスキーマを作成します。デフォルトでは、`-dropSchema` は `false` です。

`-targetSchema <schema_name>`

`-targetSchema` オプションを使用して、移行したスキーマの名前を指定します。複数のスキーマを移行する場合は、スペース文字を介さないコンマ区切りのリストで各スキーマの名前を指定します。`-targetSchema` オプションを使用しない場合、新しいスキーマの名前はソーススキーマの名前と同じになります。

`information-schema`、`dbo`、`sys`、または `pg_catalog` はターゲットスキーマ名として指定できません。これらのスキーマ名は、EDB Postgres Advanced Server のメタデータストレージ用に予約されています。

スキーマオブジェクト選択オプション

次のオプションを使用して、移行する特定のスキーマオブジェクトを選択します。

`-allTables`

ソーススキーマからすべてのテーブルをインポートします。

`-tables <table_list>`

選択したテーブルをソーススキーマからインポートします。`table_list` は、スペース文字を含まないテーブル名のコンマ区切りリストたとえば `-tables emp,dept,acctg`。

`-excludeTables <table_list>`

選択したテーブルを移行から除外します。`table_list` は、スペース文字を含まないテーブル名のコンマ区切りリストたとえば `-excludeTables emp,jobhist`。このオプションは、スキーマのすべてのテーブルが `-allTables` オプションを使用して移行のために選択された場合に適用されます。

`-constraints`

テーブルコンストレインをインポートします。このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションを指定する場合にのみ有効です。

`-ignoreCheckConstFilter`

デフォルトでは、Migration Toolkit は Sybase データベースからのチェック制約とデフォルト句の移行を実装しません。制約とデフォルト句を Sybase データベースから移行するには、`-constraints` パラメーターを指定するときに `-ignoreCheckConstFilter` パラメーターを含めます。

`-skipCKConst`

チェックコンストレインの移行を省略します。このオプションは、ターゲットデータベースでサポートされていないソースデータベースのビルトインファンクションに基づいたチェック制約を移行する場合に役立ちます。

このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションが指定された場合にのみ有効です。

`-skipFKConst`

外部キーコンストレインの移行を省略します。このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションが指定された場合にのみ有効です。

`-skipColDefaultClause`

列の `DEFAULT` 句の移行を省略します。

`-indexes`

テーブルインデックスをインポートします。このオプションは、スキーマ全体をインポートする場合、または `-allTables` または `-tables <table_list>` オプションが指定された場合に有効です。

`-triggers`

テーブルトリガーをインポートします。このオプションは、スキーマ全体をインポートする場合、または `allTables` または `-tables <table_list>` オプションが指定された場合に有効です。

`-allViews`

ソーススキーマからビューをインポートします。このオプションは、動的ビューとマテリアライズドビューをソースから移行します。Oracle および Postgres のマテリアライズドビューがサポートされています。

`-views <view_list>`

指定されたマテリアライズドビューまたはダイナミックビューをソーススキーマからインポートします。Oracle および Postgres のマテリアライズドビューがサポートされています。`view_list` は、スペース文字を含まないビュー名のコンマ区切りリストたとえば `-views all_emp,mgmt_list,acct_list`。

`-excludeViews <view_list>`

選択したビューを移行から除外します。`view_list` は、スペース文字を含まないビュー名のコンマ区切りリストたとえば `-excludeViews all_emp,acct_list`。このオプションは、スキーマのすべてのビューが `-allViews` オプションを使用して移行のために選択された場合に適用されます。

`-allSequences`

ソーススキーマからすべてのシーケンスをインポートします。

`-sequences <sequence_list>`

選択したシーケンスをソーススキーマからインポートします。`<sequence_list>` は、スペース文字を含まないシーケンス名のコンマ区切りリストです。

`-excludeSequences <sequence_list>`

選択した配列を移行から除外します。`sequence_list` は、スペース文字を含まないシーケンス名のコンマ区切りリストです。このオプションは、スキーマのすべてのシーケンスが `-allSequences` オプションを使用して移行のために選択される場合に適用されます。

例 `-allSequences -excludeSequences my_sequence,my_sequence_with_increment`

`-allProcs`

ソーススキーマからすべてのストアードプロシージャをインポートします。

`-procs <procedures_list>`

選択したストアードプロシージャをソーススキーマからインポートします。 `procedures_list` は、スペース文字を含まないプロシージャ名のコンマ区切りリストです。

`-excludeProcs <procedures_list>`

選択したプロシージャを移行から除外します。 `procedures_list` は、スペース文字を含まないプロシージャ名のコンマ区切りのリストです。このオプションは、スキーマのすべてのプロシージャが `-allProcs` オプションを使用して移行のために選択された場合に適用されます。

例 `-allProcs -excludeProcs show_notice,show_custom_notice`

`-allFuncs`

ソーススキーマからすべてのファンクションをインポートします。

`-funcs <function_list>`

選択したファンクションをソーススキーマからインポートします。 `function_list` は、スペース文字を含まないファンクション名のコンマ区切りリストです。

`-excludeFuncs <function_list>`

選択した機能を移行から除外します。 `function_list` は、スペース文字を含まないファンクション名のコンマ区切りリストです。このオプションは、`-allFuncs` オプションを使用して移行のためにスキーマのすべてのファンクションが選択された場合に適用されます。

例 `-allFuncs -excludeFuncs calculate_average_salary,add_two_numbers`

`-checkFunctionBodies [true/false]`

`false` の場合、ファンクション作成中にファンクション本体の検証を無効にします。この検証を無効にすると、ファンクションに前方参照が含まれる場合、エラーが回避されます。デフォルト値は `true` です。

`-allPackages`

ソーススキーマからすべてのパッケージをインポートします。

`-packages <package_list>`

選択したパッケージをソーススキーマからインポートします。 `package_list` は、スペース文字を含まないパッケージ名のコンマ区切りリストです。

`-excludePackages <package_list>`

選択したパッケージを移行から除外します。 `package_list` は、スペース文字を含まないパッケージ名のコンマ区切りリストです。このオプションは、スキーマのすべてのパッケージが `-allPackages` オプションを使用して移行のために選択された場合に適用されます。

例 `-allPackages -excludePackages my_package1, mypackage2`

`-allDomains`

すべてのドメイン、列挙、および複合タイプをソースデータベースからインポートします。このオプションは、ソースとターゲットの両方が Postgres ホストに保存されている場合にのみ有効です。

-allQueues

ソーススキーマからすべてのキューをインポートします。これらは、DBMS_AQ および DBMS_AQADM ビルトインパッケージによって作成および管理されるキューです。Oracle がソースデータベースの場合、**-objectTypes** オプションも指定する必要があります。EDB Postgres Advanced Server がソースデータベースの場合、**-allDomains** および **-allTables** オプションも指定する必要があります。Oracle および EDB Postgres Advanced Server キューがサポートされています。

-queues <queue_list>

選択したキューをソーススキーマからインポートします。**queue_list** は、スペース文字を含まないキュー名のコンマ区切りリストです。これらは、DBMS_AQ および DBMS_AQADM ビルトインパッケージによって作成および管理されるキューです。Oracle がソースデータベースの場合、**-objectTypes** も指定する必要があります。EDB Postgres Advanced Server がソースデータベースの場合、**-allDomains** および **-allTables** も指定する必要があります。Oracle および EDB Postgres Advanced Server キューがサポートされています。

-excludeQueues <queue_list>

選択したキューを移行から除外します。**queue_list** は、スペース文字を含まないキュー名のコンマ区切りリストです。このオプションは、スキーマのすべてのキューが **-allQueues** オプションを使用して移行のために選択される場合に適用されます。

例 **-allQueues -excludeQueues EMP_QUEUE1,EMP_QUEUE2**

-allRules

ソースデータベースからすべてのルールをインポートします。このオプションは、ソースとターゲットの両方が Postgres ホストに保存されている場合にのみ有効です。

-allGroups

ソースデータベースからすべてのグループをインポートします。

-groups <group_list>

ソースデータベースから選択されたグループ。**<group_list>** は、グループ名のコンマ区切りリストです。

例 **-groups acct_emp,mkt_emp** .

移行オプション

移行オプションを使用して、移行プロセスの詳細を制御します。

-truncLoad

新しいデータをインポートする前に、テーブルからデータを切り捨てます。このオプションは **-dataOnly** オプションとともに使用します。

-enableConstBeforeDataLoad

非パーティションソーステーブルがパーティションテーブルにマップされる場合、**-enableConstBeforeDataLoad** オプションを含めます。このオプションは、データの移行の前に、

ターゲットテーブルのすべてのトリガーを有効にします。データを個々のパーティションにリダイレクトするトリガーを含みます。`-enableConstBeforeDataLoad` は、`-truncLoad` パラメーターも指定した場合にのみ有効です。

`-retryCount` [`<value>`]

複数のスキーマの移行を実行している場合、スキーマ間の依存関係のため、最初の移行試行中に移行に失敗したオブジェクトは、後の移行中に正常に移行される場合があります。`-retryCount` オプションを使用して、Migration Toolkit が最初の移行試行中に失敗したオブジェクトの移行を試行する回数を指定します。0 より大きい値を指定します。デフォルト値は 2 です。

`-safeMode`

`-safeMode` オプションを含める場合、Migration Toolkit は移行として各行をコミットします。移行ですべてのレコードの転送に失敗した場合、障害ポイントより前に挿入された行はターゲットデータベースに残ります。

`-fastCopy`

`-fastCopy` オプションを含めると、Migration Toolkit が WAL ロギングをバイパスして、最適化された方法で COPY 操作を実行するように指定します。デフォルトでは無効になっています。`-fastCopy` オプションの使用を選択した場合、移行が中断された場合、ターゲットデータベースで移行されたデータをリカバリできない場合があります。

`-replaceNullChar` `<value>`

Migration Toolkit は、値が NULL の列のインポートをサポートしています。ただし、Migration Toolkit は、JDBC 接続プロトコルでの NULL 文字値埋め込みバイナリゼロ 0x00 のインポートをサポートしていません。NULL 文字を含むデータをインポートする場合は、`-replaceNullChar` オプションを使用して、NULL 文字を単一の非 NULL 置換文字に置き換えます。置換文字を引用符またはアポストロフィで囲まないでください。

データが移行されたら、SQL ステートメントを使用して、`-replaceNullChar` で指定された文字をバイナリゼロに置き換えます。

`-analyze`

`-analyze` オプションを含めて、ターゲットデータベースに対して Postgres ANALYZE オペレーションを呼び出します。オプティマイザーは、ANALYZE オペレーションによって収集された統計を参照し、その情報を使用して効率的なクエリプランを構成します。

`-vacuumAnalyze`

`-vacuumAnalyze` オプションを含めて、ターゲットデータベースに対して VACUUM および ANALYZE オペレーションの両方を呼び出します。オプティマイザーは、ANALYZE オペレーションによって収集された統計を参照し、その情報を使用して効率的なクエリプランを構成します。VACUUM オペレーションは、ターゲットデータベースでデッドタプルが占有している記憶領域を再利用します。

`-copyDelimiter`

テーブルデータをロードするときに COPY コマンドで区切り文字として使用する単一の文字を指定します。デフォルト値は `'\t'` タブです。

`-batchSize`

一括挿入のバッチサイズを指定します。有効な値は 1～1000 です。デフォルトのバッチサイズは 1000 です。Out of Memory 例外が発生した場合は、`-batchSize` の値を減らします。

`-cpBatchSize`

COPY コマンドで使用するバッチサイズを MB 単位で指定します。0 より大きい値は有効です。デフォルトのバッチサイズは 8MB です。

`-lobBatchSize`

LOB データ型の場合、バッチでロードする行数を指定します。BYTEA、BLOB、または CLOB などのラージオブジェクトタイプ LOB 列を含むテーブルのデータ移行は、デフォルトで一度に 1 行ずつ実行されます。これは、個々の LOB 列が数百メガバイトのデータを保持する場合のヒープ領域不足エラーを回避するためです。LOB 列の平均データサイズが下限にある場合、各バッチの行数を 0 より大きい値で指定することにより、LOB バッチサイズをカスタマイズできます。

`-fetchSize`

`-fetchSize` オプションを使用して、結果セットで取得される行数を指定します。指定された `-fetchSize` が大きすぎる場合、Out of Memory 例外が発生する可能性があります。大規模なテーブルを移行するときこの落とし穴を回避するには、`-fetchSize` オプションを含めます。デフォルトのフェッチサイズは JDBC ドライバー実装に固有であり、データベースによって異なります。

MySQL ユーザーの注 デフォルトでは、MySQL JDBC ドライバーは、テーブルのすべての行を 1 回のネットワークラウンドトリップで Migration Toolkit に取得します。この動作は、大規模なテーブルの使用可能なメモリを簡単に超過する場合があります。ヒープ領域不足エラーが発生した場合は、コマンドライン引数として `-fetchSize 1` を指定して、Migration Toolkit に一度に 1 行ずつテーブルデータをロードするように強制します。

`-filterProp <file_name>`

`<file_name>` は、`key=value` ペアのコンストレインを含むファイルの名前を指定します。データベースから読み取られた各レコードは、制約に対して評価されます。制約を満たすものが移行されます。ペアの左側にはテーブル名がリストされます。

- テーブル名はスキーマ修飾されていない必要があります。
- Oracle テーブル名は大文字と小文字を区別しませんが、Oracle はデフォルトで大文字です。一方、Postgres はデフォルトで小文字です。Oracle データを移行するときは、使用している大文字と小文字が Oracle の大文字と小文字が一致することを確認します。それ以外の場合、Migration Toolkit はそれと一致せず、コンストレインを無視します。右側は、移行される各行に `true` が必要な条件を指定します。

たとえば、このコードは、`AR` と等しくない `COUNTRY_ID` 値を持つ国のみを移行します。

```
COUNTRIES=COUNTRY_ID<>'AR'
```

この制約は、`COUNTRIES` テーブルに適用されます。

複数のテーブルに条件を指定することもできます。ただし、各テーブルの条件はプロパティファイルの新しい行にある必要があります。

例

プロパティファイルの次のエントリは、EMPLOYEES および DEPARTMENTS テーブルから関連するデータのみを移行します。

```
EMPLOYEES=(LAST_NAME IN ('Grant','Weiss') AND PHONE_NUMBER LIKE '650%')
```

```
DEPARTMENTS=(DEPARTMENT_ID BETWEEN 10 AND 30)
```

```
-customColTypeMapping <column_list>
```

カスタムタイプマッピングを使用して、移行された列のデータ型を変更します。各ペアの左側は、正規表現で列を指定します。各ペアの右側は、その列が想定するデータ型を名前付けます。<column_list> のセミコロン区切りリストに複数のペアを含めることができます。たとえば、名前が ID で終わる列を INTEGER 型にマップするには、次のカスタムマッピングエントリを使用します。

```
.*ID=INTEGER
```

カスタムマッピングは、列がテーブル修飾されていない限り、基準に一致するすべてのテーブル列に適用されます。

'\\' 文字はエスケープ文字列として機能します。 '.' は正規表現の予約文字であるため、Linux では '\\.' を使用して '.' 文字を表します。たとえば、カスタムマッピングを使用して EMP テーブルの EMP_ID 列から行を選択するには、次のカスタムマッピングエントリを指定します。

```
EMP\\.*EMP_ID=INTEGER
```

Windows、'\\.' を使用して '.' 文字を表します。

```
EMP\\.*EMP_ID=INTEGER
```

または、複数のカスタムタイプマッピングを<property_file> に含めることができます。

```
-customColTypeMappingFile <property_file>
```

ファイルの各エントリを key=value のペアで個別の行に指定します。各ペアの左側は、正規表現を使用して列を選択します。各ペアの右側は、その列が想定するデータ型を名前付けます。<property_file> で使用される場合、'\\' 文字は、オペレーティングシステムでエスケープ文字列として機能します。

接続再試行オプション

データの移行中にターゲットデータベースとの接続障害が発生するたび、移行トウキットはターゲットデータベースに再接続を試行して、テーブルをスキップせずに移行を完了します。接続が再確立されると、Migration Toolkit は、接続が失われたときに移行中であったテーブルのデータコピーを再起動または再開します。次に、残りのテーブルのデータコピーを実行します。

Migration Toolkit は、データの移行に使用していたモードに基づいて、移行を再開します。

- Migration Toolkit がコピー API を使用してデータを移行していた場合、最後に失敗した行からデータのコピーを続行します。Copy API は、ターゲットデータベースとして PostgreSQL および EDB Postgres Advanced Server と互換性があります。
- 他のデータ移行モードでは、Migration Toolkit はターゲットテーブルのデータを切り捨て、障害が発生したテーブル全体を再コピーします。

!!!note スコープと制限 データベーススコープ接続再試行機能により、Migration Toolkit はターゲットデータベースに再接続できます。ソースデータベースの問題の再試行はサポートされていません。

移行スコープ この機能により、Migration Toolkit はデータの移行を再試行できます。スキーマの移行に関する問題の再試行はサポートされていません。

モダリティスコープ この再接続機能は、データ移行モード `-dataOnly` で利用できます。 `-dataOnly` または `-schemaOnly` オプションを指定せずに移行を実行する場合にも使用できます。

`-connRetryCount` [`<connection_attempts>`]

`-connRetryCount` オプションを使用して、ターゲットデータベース接続が失敗した場合に実行する再試行回数を指定します。 [`<connection_attempts>`] 値は 0～50 の数値である必要があります。デフォルトは 3 回の再試行です。

再試行はデータの移行に適用されるため、 `-schemaOnly` オプションとは互換性がありません。

例

```
$ ./runMTK.sh -connRetryCount 2 -dataOnly -tables dept,emp,jobhist public
```

`-connRetryInterval` [`<seconds>`]

`-connRetryInterval` オプションを使用して、ターゲットデータベース接続が失敗した場合に、後続の各再接続を試行する前に待機する秒数を指定します。 [`<seconds>`] 値は、0～900 の数値である必要があります。デフォルトは 30 秒です。

再試行はデータの移行に適用されるため、 `-schemaOnly` オプションとは互換性がありません。

例

```
$ ./runMTK.sh -connRetryCount 2 -connRetryInterval 50 -dataOnly -tables dept,emp,
→jobhist public
```

`-abortConnOnFailure` [`true/false`]

すべての再接続試行が失敗した場合に、移行を中止するかどうかを指定します。

障害が発生したテーブルの移行をスキップして、次のテーブルに進むには、 `-abortConnOnFailure` を `false` に設定します。デフォルトは `true` で、指定された `-connRetryCount` しきい値の後に接続が失敗した場合にセッションを中止します。

再試行はデータの移行に適用されるため、 `-schemaOnly` オプションとは互換性がありません。

例

```
$ ./runMTK.sh -connRetryCount 2 -connRetryInterval 50 -abortConnOnFailure false -
→dataOnly -tables dept,emp,jobhist public
```

`-pgIdleTxSessionTimeout` [`<seconds>`]

PostgreSQL または EDB Postgres Advanced Server `idle_in_transaction_session_timeout` を指定します。これは、移行トランザクションがアイドル状態にあるときにデータベースがセッションを終了するまでの時間を定義します。 [`<seconds>`] は、0 より大きい値です。デフォルトは 180 秒です。

例

```
runMTK.sh -connRetryCount 2 -connRetryInterval 50 -pgIdleTxSessionTimeOut 90 -  
→dataOnly -tables dept,emp,jobhist public
```

並列データのロードのための移行オプション

比較的多数のテーブルまたは大規模なテーブルを扱う場合、通常のデータの移行には時間がかかります。この問題の解決策として、Migration Toolkit には、スキーマレベルおよびテーブルレベルで並列にデータを移行できる並列データのロードアルゴリズムがあります。

スキーマレベルでの並列データの移行では、マルチプルのスレッドを使用して複数のテーブルを同時に移行します。テーブルレベルでの並列データのロードでは、マルチプルのスレッドを使用して、単一のテーブルをソースデータベースからターゲットデータベースにより効率的かつ短時間で移行します。

注釈:

- パラレルロードは、大規模なテーブルの場合にのみ役立ちます。小さなテーブルの場合、並列スレッドはさらに時間がかかる場合があります。したがって、比較的大規模なテーブルの場合、テーブルレベルでの並列データのロードを使用することをお勧めします。
-
- 並列データのロードは、主キー/一意キーのないテーブルには適用されません。
 - 並列データのロードでは `fastCopy` パラメーターを使用できません。
 - パラレルロード機能は、ソース Oracle、EDB Postgres Advanced Server、および PostgreSQL データベースにのみ適用されます。

次のオプションは、MTK がデータを並列して移行する方法を制御します。

`-loaderCount` [`<value>`]

`-loaderCount` オプションを使用して、データのロードを並列実行するためにプールで使用可能な並列スレッドの数を指定します。このオプションは、Migration Toolkit を実行しているホストシステムにハイエンドの CPU および RAM リソースがある場合に特に役立ちます。`value` にはゼロ以外の正の数を指定できますが、その値が CPU コア数を超えないことをお勧めします。たとえば、デュアルコア CPU には、2 の最適値があります。デフォルトは 1 です。

テーブルレベルのスレッドの数は、テーブルサイズと `shared_buffers` や `work_mem` などの特定のメモリ構成に応じて、ソースデータベースサーバーにオーバーヘッドを発生させる可能性があります。メモリ構成はデータベースサーバーに最適な値に設定する必要があります。非常に大規模なテーブルの場合、マシンで使用可能な合計コアに従って、より低い `-loaderCount` オプション値を試して、最適な `-loaderCount` 値を計算することをお勧めします。

注釈: マルチプルのスレッドが使用される場合、実際のスレッド数とテーブルデータのサイズによっては、Migration Toolkit のメモリヒープサイズを調整する必要がある場合があります。

`-parallelLoadRowLimit` [`<value>`]

`-parallelLoadRowLimit` オプションを使用して、テーブルに必要な最小行数を指定して、テーブルレベルでデータのロードを並列実行します。`<value>` パラメーターは 0 より大きい必要があります。デフォルトは 100,000 です。

たとえば、`-parallelLoadRowLimit` の値が 10,000 の場合、行数 10,000 以上のテーブルのみがテーブルレベルで並列にロードされます。他のテーブルは、シークエンシャルモードまたはスキーマレベルで移行します。

`-tableLoaderLimit [<value>]`

`-tableLoaderLimit` オプションを使用して、スレッドプールから割り当てるスレッドの最大数を指定して、テーブルレベルで並列チャンクで単一のテーブルデータをロードします。`<value>` パラメーターは、`-loaderCount` の値を超えてはなりません。デフォルト値は `-loaderCount` オプションの値と同じです。このオプションは、`-loaderCount` のデフォルト以外のカスタム値に基づいて暗黙的に変更されます。

たとえば、`-loaderCount` が 4 に設定され、`-tableLoaderLimit` が指定されていない場合、`-tableLoaderLimit` も 4 に設定されます。したがって、すべてのスレッドが単一のテーブルのロードに使用されない場合は、`-tableLoaderLimit` をより低い値に減らします。

並列データのロードアルゴリズムの高レベルの概要

アルゴリズムは 3 つのステップで動作します。

1. テーブルとテーブルチャンクのリストは、テーブル名でソートされた、移行するテーブルの順序付けリストから作成されます。リストは、`-parallelLoadRowLimit` および `-tableLoaderLimit` オプション値に基づいて作成されます。
2. スレッドプールは、`-loaderCount` オプション値に基づいて作成されます。
3. 並列データのロードアルゴリズムは、テーブルとテーブルチャンクのリストからテーブル/チャンクごとにタスクを作成し、スレッドプールで実行します。スレッドの数と同じ数のタスクが並列実行されます。タスクの実行が完了すると、そのスレッドはリストから次のタスクの実行を開始します。パラレルデータのロードは、リストの最後のタスクが完了するとすぐに完了します。

例

このシナリオでは、並列データのロードを使用して 4 つのテーブル `tab1`, `tab2`, `tab3`, `tab4` を移行します。

```
# ./runMTK.sh -sourcedbtype enterprisedb -targetdbtype enterprisedb -loaderCount 4 -
→parallelLoadRowLimit 100000 -tableLoaderLimit 2 -tables tab1,tab2,tab3,tab4 public
```

テーブル `tab1` には 50,000 行があり、テーブル `tab2` には 9,500 行があり、テーブル `tab3` には 50,001 行があり、テーブル `tab4` には 9,999 行があります。ステップ 1 のテーブルのリストは `tab1`, `tab2`, `tab3`, および `tab4` です。

表 `tab1` および `tab3` のみが、テーブルレベル `-parallelLoadRowLimit 100000` のパラレルロードの対象となります。

-tableLoaderLimit オプションの値は 2 であるため、テーブル tab1 は 2 つのチャンク、tab1_chunk1 および tab1_chunk2 にロードされます。各チャンクの行数は、totalRowCount/tableLoaderLimit に基づいて決定されます。この場合、両方のチャンクに 25,000 行 50,000/2 があります。

テーブル tab3 は、2 つのチャンク、tab3_chunk1 および tab3_chunk2 にもロードされます。テーブル tab3 の合計行数が奇数 50,001 であるため、最初のチャンクは追加行を収容するため、tab3_chunk1 には 25,001 行、tab3_chunk2 には 25,000 行があります。

次に、作成されたテーブルとテーブルチャンクのリストを示します。

tab1_chunk1	(25,000 rows)
tab1_chunk2	(25,000 rows)
tab2	(9500 rows)
tab3_chunk1	(25,001 rows)
tab3_chunk2	(25,000 rows)
tab4	(9999 rows)

4 つのスレッド T1, T2, T3, T4 からなるスレッドプールが作成されます。手順 1 で作成したリストからテーブル/チャンクを移行するタスクは、スレッドプールで実行されます。

T1 → tab1_chunk1	(25,000 rows)
T2 → tab1_chunk2	(25,000 rows)
T3 → tab2	(9500 rows)
T4 → tab3_chunk1	(25,001 rows)

スレッド T3 が最初に終了し、リストの次のチャンクである tab3_chunk2 の実行を開始します。

T1 → tab1_chunk1	(25,000 rows)
T2 → tab1_chunk2	(25,000 rows)
T3 → tab3_chunk2	(25,000 rows)
T4 → tab3_chunk1	(25,001 rows)

スレッド T1 は 2 番目に終了し、tab4 の実行を開始します。

T1 → tab4	(9999 rows)
T2 → tab1_chunk2	(25,000 rows)
T3 → tab3_chunk2	(25,000 rows)
T4 → tab3_chunk1	(25,001 rows)

この方法では、4 つのスレッドを使用して、混合スキーマとテーブルレベルで 4 つのテーブルを並列に移行します。コマンドオプションを変更して -tableLoaderLimit を 1 に設定すると、純粋なスキーマレベルの並列処理ができます。

```
# ./runMTK.sh -sourcedbtype enterprisedb -targetdbtype enterprisedb -loaderCount 4 -
→parallelLoadRowLimit 10000 -tableLoaderLimit 1 -tables tab1,tab2,tab3,tab4 public
```

各テーブルは 1 つのスレッドで同時に処理されます。

```
T1 → tab1    (50,000 rows)
T2 → tab2    (9500 rows)
T3 → tab3    (50,001 rows)
T4 → tab4    (9999 rows)
```

システムリソースの推奨事項と制約

Migration Toolkit を実行しているホストシステムで利用可能な CPU および RAM リソースに応じて、スレッド数を選択します。ソースデータベースには CPU、IO、ディスクの使用に関して特定のオーバーヘッドが存在するため、並列スレッド数が特定のしきい値を超えて超過すると、Migration Toolkit のパフォーマンスが低下します。

Oracle 固有のオプション

次のオプションは、ソースデータベースが Oracle の場合にのみ適用されます。

-objectTypes

runMTK.sh コマンドの最後に指定されたスキーマリストからユーザー定義オブジェクトタイプをインポートします。

-allUsers

すべてのユーザーとロールをソースデータベースからインポートします。- allUsers オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-users <user_list>

選択したユーザーまたはロールをソース Oracle データベースからインポートします。<user_list> は、スペース文字を含まないユーザー/ロール名のコンマ区切りリストです -users MTK, SAMPLE, acctg など。-users オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-allProfiles

すべてのカスタムつまりユーザー作成のプロファイルをソースデータベースからインポートします。DEFAULT や MONITORING_PROFILE などの他の Oracle 非カスタムプロファイルはインポートされません。

インポートされたプロファイルの場合、プロファイルに関連付けられている次のパスワードパラメーターのみがインポートされます。

FAILED_LOGIN_ATTEMPTS

PASSWORD_LIFE_TIME

PASSWORD_REUSE_TIME

PASSWORD_REUSE_MAX

PASSWORD_LOCK_TIME

PASSWORD_GRACE_TIME

PASSWORD_VERIFY_FUNCTION

Oracle リソースパラメーターなど、他のすべてのプロファイルパラメーターはインポートされません。SRC_DB_USER で指定された Oracle データベースユーザーは、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 特権を持っている必要があります。

注釈: - *allProfiles* オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-profiles <profile_list>

選択したカスタムつまりユーザー作成のプロファイルをソース Oracle データベースからインポートします。profile_list は、スペース文字を含まないプロファイル名のコンマ区切りリストたとえば **-profiles ADMIN_PROFILE,USER_PROFILE**。DEFAULT や MONITORING_PROFILE などの Oracle の非カスタムプロファイルはインポートされません。

-allProfiles オプションと同様に、パスワードパラメーターのみがインポートされます。SRC_DB_USER で指定された Oracle データベースユーザーは、Oracle データディクショナリビュー DBA_PROFILES に対する SELECT 特権を持っている必要があります。

注釈: -profiles オプションは、Oracle データベースから EDB Postgres Advanced Server データベースに移行する場合にのみサポートされます。

-importPartitionAsTable <table_list>

-importPartitionAsTable パラメーターを含めて、Oracle ホストにあるパーティションテーブルの内容を単一の非パーティションテーブルにインポートします。table_list は、スペース文字を含まないテーブル名のコンマ区切りリストたとえば **-importPartitionAsTable emp,dept,acctg**。

-copyViaDBLinkOra

dblink_ora モジュールは、SQL レベルでの EDB Postgres Advanced Server と Oracle の接続を提供します。dblink_ora は、EDB Postgres Advanced Server データベースインストールの一部としてバンドルおよびインストールされます。dblink_ora は、COPY API メソッドを使用してデータベース間でデータを転送します。このメソッドは、JDBC COPY メソッドよりもかなり高速です。

この例では、dblink_ora COPY API を使用して、HR スキーマからすべてのテーブルを移行します。

```
./runMTK.sh -copyViaDBLinkOra -allTables HR
```

ターゲット EDB Postgres Advanced Server データベースには、dblink_ora をインストールおよび構成する必要があります。dblink_ora を参照してください。

-allDBLinks [link_Name_1=password_1,link_Name_2=password_2,...]

Oracle データベースリンクを移行するには、このオプションを選択します。ソースデータベースの各リンク接続のパスワード情報は暗号化されるため、指定しない限り、ダミーパスワード edb に置き換えられます。

接続したユーザーのダミーパスワードとして edb を使用して、すべてのデータベースリンクを移行するには

```
$/runMTK.sh -allDBLinks HR
```

または、スペース文字を介さない `name=value` ペアのコンマ区切りリストを介して、各データベースリンクのパスワードを指定できます。ペアの左側にリンク名を指定し、右側にパスワード値を指定します。

コマンドラインで指定された実際のパスワードを使用してすべてのデータベースリンクを移行するには

```
$/runMTK.sh -allDBLinks LINK_NAME1=abc,LINK_NAME2=xyz HR
```

Migration Toolkit は、EnterpriseDB で現在サポートされているデータベースリンクタイプのみを移行します。これらのタイプには、パブリックおよびプライベートタイプの固定ユーザーリンクが含まれます。

`-allSynonyms`

`-allSynonyms` オプションを含めて、すべてのパブリックおよびプライベートシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行したバージョンに置き換えられます。

`-allPublicSynonyms`

`-allPublicSynonyms` オプションを含めて、すべてのパブリックシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行したバージョンに置き換えられます。

`-excludeSynonyms <synonym_list>`

選択した同義語を移行から除外します。 `synonym_list` は、スペース文字を含まないシノニム名のコンマ区切りリストです。このオプションは、スキーマのすべてのシノニムが `-allSynonyms` オプションを使用して移行のために選択される場合に適用されます。

例 `-allSynonyms -excludeSynonyms SYNEMP1,SYNEMP2`

`-allPrivateSynonyms`

`-allPrivateSynonyms` オプションを含めて、すべてのプライベートシノニムを Oracle データベースから EDB Postgres Advanced Server データベースに移行します。同じ名前のシノニムがターゲットデータベースに既に存在する場合、既存のシノニムは移行したバージョンに置き換えられます。

`-useOraCase`

`-useOraCase` オプションを含めて、Oracle データベースから EDB Postgres Advanced Server データベースに移行するときに、すべてのデータベースオブジェクトの Oracle のデフォルトの大文字の命名規則を保持します。

大文字の命名規則は、テーブル、ビュー、シーケンス、プロシージャ、ファンクション、トリガー、パッケージなどで保持されます。これらのデータベースオブジェクトの場合、大文字の命名規則は以下に適用されます。

- データベースオブジェクトの名前
- テーブル、ビューのカラム名、キー名、インデックス名、制約名など
- ビューの `SELECT` 列リスト
- プロシージャまたはファンクションヘッダーの一部であるパラメーター名

注釈: プロシージャ、ファンクション、トリガー、またはパッケージのプロシージャルコード本体では、エラーなしで実行するにはプログラムの識別子参照を手動で編集する必要がある場合があります。このような修正は、発生した可能性のある識別子参照の適切な大文字と小文字の変換に関するものです。

注釈: `-useOraCase` オプションを指定する場合、`-skipUserSchemaCreation` オプションも指定する必要があります。詳細については、`-skipUserSchemaCreation` オプションの説明を参照してください。

`-useOraCase` オプションを使用しない Migration Toolkit のデフォルト動作は、データベースオブジェクトが引用符で囲まれて Oracle で明示的に作成されていない限り、引用符で囲まらずに Oracle からデータベースオブジェクト名が抽出されることです。以下は、Migration Toolkit で `-offlineMigration` オプションを使用して生成されたテーブルコマンドの一部です。

```
CREATE TABLE DEPT (
    DEPTNO NUMBER(2) NOT NULL,
    DNAME VARCHAR2(14),
    LOC VARCHAR2(13)
);
ALTER TABLE DEPT ADD CONSTRAINT DEPT_PK PRIMARY KEY (DEPTNO);
ALTER TABLE DEPT ADD CONSTRAINT DEPT_DNAME_UQ UNIQUE (DNAME);
```

次に、このテーブルを移行して EDB Postgres Advanced Server で作成すると、引用符で囲まれていないオブジェクト名はすべて小文字に変換されるため、テーブルは次のように EDB Postgres Advanced Server に表示されます。

```
Table "edb.dept"
Column | Type | Modifiers
-----+-----+-----
deptno | numeric(2,0) | not null
dname | character varying(14) |
loc | character varying(13) |
Indexes:
    "dept_pk" PRIMARY KEY, btree (deptno)
    "dept_dname_uq" UNIQUE CONSTRAINT, btree (dname)
```

EDB Postgres Advanced Server アプリケーションが、引用符で囲まれた大文字の識別子を使用して移行されたデータベースオブジェクトを参照している場合、データベースオブジェクト名が小文字になっているため、アプリケーションは失敗します。

```
usepostcase=# SELECT * FROM "DEPT";
ERROR: relation "DEPT" does not exist
LINE 1: SELECT * FROM "DEPT";
```

アプリケーションが引用符で囲まれた大文字の識別子を使用している場合、`-useOraCase` オプションを使用して移行を実行します。DDL は、すべてのデータベースオブジェクト名を引用符で囲みます。

```
CREATE TABLE "DEPT" (
  "DEPTNO" NUMBER(2) NOT NULL,
  "DNAME" VARCHAR2(14),
  "LOC" VARCHAR2(13)
);
ALTER TABLE "DEPT" ADD CONSTRAINT "DEPT_PK" PRIMARY KEY ("DEPTNO");
ALTER TABLE "DEPT" ADD CONSTRAINT "DEPT_DNAME_UQ" UNIQUE ("DNAME");
```

次に、このテーブルを移行して EDB Postgres Advanced Server で作成すると、すべてのオブジェクト名が大文字で維持されるため、テーブルは次のように EDB Postgres Advanced Server に表示されます。

```
Table "EDB.DEPT"
Column | Type | Modifiers
-----+-----+-----
DEPTNO | numeric(2,0) | not null
DNAME | character varying(14) |
LOC | character varying(13) |
Indexes:
  "DEPT_PK" PRIMARY KEY, btree ("DEPTNO")
  "DEPT_DNAME_UQ" UNIQUE CONSTRAINT, btree ("DNAME")
```

アプリケーションは、引用符で囲まれた大文字の名前を使用してオブジェクトにアクセスできます。

```
useoracase=# SELECT * FROM "DEPT";
DEPTNO | DNAME | LOC
-----+-----+-----
10 | ACCOUNTING | NEW YORK
20 | RESEARCH | DALLAS
30 | SALES | CHICAGO
40 | OPERATIONS | BOSTON
(4 rows)
```

-skipUserSchemaCreation

Oracle ユーザーが移行されると、ロールがまだ存在しない場合、Oracle ユーザーのターゲットデータベースサーバーにロールつまりユーザー名が作成されます。ロール名は小文字で作成されます。新しいロールが作成されると、同じ名前のスキーマも小文字で作成されます。

-skipUserSchemaCreation オプションを指定すると、移行された Oracle ユーザー名の自動スキーマ作成が防止されます。このオプションは、-useOraCase オプションが指定されている場合に特に役立ち、大文字と小文字のみが異なる同じ名前でも 2 つのスキーマを作成するのを防ぎます。-useOraCase オプションを指定すると、Migration Toolkit が呼び出されるときにオプションリストに続いて指定されたソーススキーマの大文字の Oracle 命名規則でスキーマが作成されます。

したがって、-skipUserSchemaCreation オプションを使用せずに -useOraCase オプションを指定すると、ターゲットデータベースには、1 つは小文字、もう 1 つは大文字の 2 つの同じ名前のスキーマがあります。-useOraCase オプションが -skipUserSchemaCreation オプションとともに指定された場合、ターゲットデータベースには大文字のスキーマのみが含まれます。

その他のオプション

これらの移行オプションを使用して、Migration Toolkit のヘルプとバージョン情報を表示します。これらのオプションを使用して、Migration Toolkit のフィードバックとログオプションを制御することもできます。

-help

アプリケーションのコマンドラインの使用情報を表示します。

-logDir <log_path>

このオプションを含めて、ログファイルを書き込む場所を指定します。<log_path> は、ログファイルを保存する場所です。デフォルトでは、Linux ではログファイルは次の場所に書き込まれます。

`$HOME/.enterprisedb/migration-toolkit/logs`

Windows では、ログファイルは次の場所に保存されます。

`%HOMEDRIVE%%HOMEPATH%\enterprisedb\migration-toolkit\logs`

-logFileCount <file_count>

このオプションを含めて、ログファイルのローテーションで使用するファイルの数を指定します。0 の値を指定して、ログファイルのローテーションを無効にし、単一のログファイルを作成します。このファイルは、logFileSize オプションを使用して指定された値に達すると、切り捨てられます。<file_count> は 0 以上である必要があります。デフォルトは 20 です。

-logFileSize <file_size>

新しいログファイルにローテーションする前に、このオプションを含めて最大ファイルサイズ制限を MB 単位で指定します。file_size は 0 より大きい必要があります。デフォルトは 50 です。

-logBadSQL

失敗したオブジェクトのスキーマ定義 DDL スクリプトをファイルに保存するには、このオプションを含めます。ファイルはエラーログに使用されるのと同じパスの下に保存され、次の形式で名前付けられます。

`mtk_bad_sql_<schema_name_timestamp>.sql`

schema_name はスキーマの名前であり、timestamp は Migration Toolkit 実行のタイムスタンプです。

-verbose [on|off]

アプリケーションログメッセージを標準出力に表示します。デフォルトでは、verbose はオンになっています。

-version

Migration Toolkit のバージョンを表示します。

例

この例では、Oracle から EDB Postgres Advanced Server への移行を実行します。

以下は、`toolkit.properties` ファイルの内容です。

```
SRC_DB_URL=jdbc:oracle:thin:@192.168.2.6:1521:xe
SRC_DB_USER=edb
SRC_DB_PASSWORD=password

TARGET_DB_URL=jdbc:edb://localhost:5444/edb
TARGET_DB_USER=enterprisedb
TARGET_DB_PASSWORD=password
```

次のコマンドは、Migration Toolkit を呼び出します。

```
$ ./runMTK.sh EDB
__OUTPUT__
Running EnterpriseDB Migration Toolkit (Build 48.0.0) ...
Source database connectivity info...
conn =jdbc:oracle:thin:@192.168.2.6:1521:xe
user =edb
password= **** *\\*
Target database connectivity info...
conn =jdbc:edb://localhost:5444/edb
user =enterprisedb
password= **** *\\*
Connecting with source Oracle database server...
Connected to Oracle, version Oracle Database 10g Express Edition
Release 10.2.0.1.0 - Production
Connecting with target EnterpriseDB database server...
Connected to EnterpriseDB, version 9.4.0.0
Importing redwood schema EDB...
Creating Schema...edb
Creating Sequence: NEXT_EMPNO
Creating Tables...
Creating Table: BAD_TABLE
MTK-15013: Error Creating Table BAD_TABLE
DB-42704: ERROR: type "binary_double" does not exist at position 58
- - CREATE TABLE BAD_TABLE (
- - F1 NUMBER NOT NULL,
- - Line 3: F2 BINARY_DOUBLE
- - ^
Creating Table: DEPT
Creating Table: EMP
Creating Table: JOBHIST
Creating Table: "MixedCase"
```

(次のページに続く)

(前のページからの続き)

```
Creating Table: "lowercase"
Created 5 tables.
Loading Table Data in 8 MB batches...
Loading Table: DEPT ...
[DEPT] Migrated 4 rows.
[DEPT] Table Data Load Summary: Total Time(s): 0.147 Total Rows: 4
Loading Table: EMP ...
[EMP] Migrated 14 rows.
[EMP] Table Data Load Summary: Total Time(s): 0.077 Total Rows: 14
Loading Table: JOBHIST ...
[JOBHIST] Migrated 17 rows.
[JOBHIST] Table Data Load Summary: Total Time(s): 0.042 Total Rows: 17
Total Size(MB): 9.765625E-4
Loading Table: "MixedCase" ...
["MixedCase"] Table Data Load Summary: Total Time(s): 0.098 Total Rows:0
Loading Table: "lowercase" ...
["lowercase"] Table Data Load Summary: Total Time(s): 0.066 Total Rows:0
Data Load Summary: Total Time (sec): 0.806 Total Rows: 35 Total
Size(MB): 0.001
Creating Constraint: DEPT_PK
Creating Constraint: DEPT_DNAME_UQ
Creating Constraint: EMP_PK
Creating Constraint: JOBHIST_PK
Creating Constraint: SYS_C008958
MTK-15001: Error Creating Constraint SYS_C008958
DB-42P01: com.edb.util.PSQLException: ERROR: relation "bad_table" does
not exist
Creating Constraint: EMP_REF_DEPT_FK
Creating Constraint: EMP_SAL_CK
Creating Constraint: JOBHIST_REF_DEPT_FK
Creating Constraint: JOBHIST_REF_EMP_FK
Creating Constraint: JOBHIST_DATE_CHK
Creating Trigger: USER_AUDIT_TRIG
Creating Trigger: EMP_SAL_TRIG
MTK-13009:Warning! Skipping migration of trigger DROP_TRIGGER, currently
non-table triggers are not supported in target database.
Creating View: SALESEMP
Creating Function: EMP_COMP
Creating Package: EMP_ADMIN
MTK-16005:Package Body is Invalid, Skipping...
Schema EDB imported with errors.
MTK-12001: The user/role migration failed due to insufficient
privileges.
Grant the user SELECT privilege on the following Oracle catalogs:
DBA_ROLES
```

(次のページに続く)

(前のページからの続き)

```

DBA_USERS
DBA_TAB_PRIVS
DBA_PROFILES
DBA_ROLE_PRIVS
ROLE_ROLE_PRIVS
DBA_SYS_PRIVS
One or more schema objects could not be imported during the migration
process. Please review the migration output for more details.
Migration logs have been saved to
/home/user/.enterprisedb/migration-toolkit/logs
****      ****      ****      ****      ****      Migration Summary      ****      ****      ****      ****      ****
Sequences: 1 out of 1
Tables: 5 out of 6
Constraints: 9 out of 10
Triggers: 2 out of 3 (skipped 1)
Views: 1 out of 1
Functions: 1 out of 1
Packages: 1 out of 1
Total objects: 30
Successful count: 20
Failed count: 2
Skipped count: 1
Invalid count: 7
List of failed objects
=====
Tables
- -----

1. EDB.BAD_TABLE
Constraints
- -----

1. EDB.BAD_TABLE.SYS_C008958
List of invalid objects
=====

1. EDB.HIRE_CLERK (FUNCTION)

2. EDB.NEW_EMPNO (FUNCTION)

3. EDB.EMP_ADMIN (PACKAGE BODY)

4. EDB.EMP_QUERY (PROCEDURE)

5. EDB.EMP_QUERY_CALLER (PROCEDURE)

```

(次のページに続く)

(前のページからの続き)

6. EDB.LIST_EMP (PROCEDURE)

7. EDB.SELECT_EMP (PROCEDURE)

スキップされたサポートされていないデータベースオブジェクトは省略されます。移行情報は、実行の最後にある移行の概要に要約されます。

第4章 Troubleshooting

4.1 Migration errors

移行プロセス中に、移行の概要に移行の進行状況が表示されます。エラーが発生した場合、概要にエラーに関する情報が表示されます。移行の概要はログファイルにも書き込まれます。

Linux では、ログファイルのデフォルトの場所は次のとおりです。

```
$HOME/.enterprisedb/migration-toolkit/logs
```

Windows では、ログファイルは次の場所に保存されます。

```
%HOMEDRIVE%%HOMEPATH%\enterprisedb\migration-toolkit\logs
```

Migration Toolkit の `-logdir <log_path>` オプションを使用して、代替ログファイルの場所を指定できます。

4.1.1 接続エラー

Migration Toolkit は、

toolkit.properties file からの情報を使用して、ソースデータベースとターゲットデータベースに接続します。

Migration Toolkit の使用時に発生する接続エラーのほとんどは、`toolkit.properties` ファイルで指定された情報に関連しています。

無効なユーザー名/パスワード

- Migration Toolkit を使用して Oracle データベースから移行を実行しようとする、次のエラーが発生します。

```
MTK-11009: Error Connecting Database "Oracle"
DB-1017: java.sql.SQLException: ORA-01017: invalid username/password;
logon denied
The user name or password specified in the `toolkit.properties` file is
not valid to use to connect to the Oracle source database.
```

このエラーを解決するには、`toolkit.properties` ファイルを編集し、`SRC_DB_USER` および `SRC_DB_PASSWORD` プロパティで移行を実行する権限を持つ有効なユーザーの名前とパスワードを指定します。

接続が拒否されました。致命的な: パスワード

- Migration Toolkit を使用して移行を実行しようとする、次のエラーが発生します。

```
MTK-11009: Error Connecting Database "EDB Postgres EDB Postgres Advanced Server"
DB-28P01: com.edb.util.PSQLException: FATAL: password authentication
failed for user "name"
```

toolkit.properties ファイルで指定されたユーザー名またはパスワードは、Postgres データベースへの接続に使用するには無効です。

このエラーを解決するには、 toolkit.properties ファイルを編集し、TARGET_DB_USER および TARGET_DB_PASSWORD プロパティで移行を実行する権限を持つ有効なユーザーの名前とパスワードを指定します。

例外 ORA-28000 アカウントはロックされています

- Migration Toolkit を使用して Oracle データベースから移行を実行しようとする、次のエラーメッセージが表示されます。

```
MTK-11009: Error Connecting Database "Oracle"
DB-28000: java.sql.SQLException: ORA-28000: the account is locked
The Oracle account associated with the user name specified in the
toolkit.properties file is locked.
```

このエラーを解決するには、Oracle サーバーでユーザーアカウントのロックを解除するか、 toolkit.properties ファイルを編集して、 SRC_DB_USER および SRC_DB_PASSWORD パラメーターで移行を実行する権限を持つ有効なユーザーの名前とパスワードを指定します。

例外 oracle.jdbc.driver.OracleDriver

- Migration Toolkit を使用して移行を実行しようとする、移行は失敗し、次のエラーメッセージが表示されます。

```
MTK-11009: Error Connecting Database "Oracle"
java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver
```

Migration Toolkit を使用する前に、移行元のデータベースの適切な JDBC ドライバーをダウンロードしてインストールする必要があります。完全な手順については、 [Installing a JDBC driver](#) を参照してください。

I/O 例外ネットワークアダプターは接続を確立できませんでした。

- Migration Toolkit を使用して移行を実行しようとすると、次のエラーが発生します。

```
MTK-11009: Error Connecting Database "Oracle"
DB-17002: java.sql.SQLException: Io exception: The
Network Adapter could not establish the connection
```

`toolkit.properties` ファイルで指定されたソースデータベースの JDBC URL には、無効な接続プロパティが含まれています。

このエラーを解決するには、[toolkit.properties file](#) を編集し、`SRC_DB_URL` プロパティでソースデータベースの有効な接続情報を指定します。`SRC_DB_URL` 値はデータベース固有です。ソースデータベースタイプの詳細については、[Building the toolkit-properties file](#) を参照してください。

例外「ターゲット」データベースに指定された URL は無効です

- Migration Toolkit を使用して移行を実行しようとすると、次のエラーが発生します。

```
MTK-10045: The URL specified for the "target" database is invalid.
Check the connectivity credentials.
```

`toolkit.properties` ファイルで指定されたターゲットデータベース EDB Postgres Advanced Server の JDBC URL には、無効な接続プロパティが含まれています。

このエラーを解決するには、`toolkit.properties` ファイルを編集し、`TARGET_DB_URL` プロパティでターゲットデータベースの有効な接続情報を指定します。EDB Postgres Advanced Server の JDBC URL の形成については、[Defining an EDB Postgres Advanced Server URL](#) を参照してください。

4.1.2 移行エラー

次のエラーは、Migration Toolkit がターゲットおよびソースデータベースサーバーに正常に接続した後に発生する可能性があります。

エラー最後の予想される列の後に余分なデータ

- *テーブルをオンラインで移行すると、次のエラーメッセージが表示されます。

```
MTK-17001: Error Loading Data into Table: table_name
DB-22P04: com.edb.util.PSQLException: ERROR: extra data after last
expected column
Where: COPY table_name, line 5: "50|HR|LOS|ANGELES"
```

このエラーは、`table_name` の列のデータに区切り文字が含まれる場合に発生します。このエラーを修正するには、区切り文字をテーブルコンテンツにない文字に変更します。

注釈: この例では、パイプ文字|が最後の列に挿入するためのテキスト ``LOS|ANGELES`` にあり、`-copyDelimiter` ' | ' オプションを使用して Migration Toolkit が実行されますが、エラーが発生します。

テーブル `TABLE_NAME` へのデータのロードエラー

データのみ移行を実行すると、次のエラーが発生します。

```
MTK-17001: Error Loading Data into Table: TABLE_NAME
DB-42P01: com.edb.util.PSQLException: ERROR: relation
"schema.\ table_name" does not exist
*I also get the error:*
Trying to reload table: TABLE_NAME through bulk inserts with a batch
size of 100
MTK-17001: Error Loading Data into Table: TABLE_NAME
DB-42P01: com.edb.util.PSQLException: ERROR: relation
"schema.\ table_name" does not exist
```

データを移行する前に、ターゲットデータベースでデータを受信するテーブルを作成する必要があります。`TABLE_NAME` の名前を持つテーブルがターゲットデータベースに存在することを確認します。必要に応じてテーブルを作成し、データの移行を再試行します。

コンストレイン `CONS_NAME_FK` の作成中にエラーが発生しました

- インデックスと制約を含むテーブルの移行を実行すると、次のエラーメッセージが表示されます。

```
MTK-15001: Error Creating Constraint EMP_DEPT_FK
DB-42P01: com.edb.util.PSQLException: ERROR: relation "hr.departments"
does not exist
Creating Constraint: EMP_JOB_FK
MTK-15001: Error Creating Constraint EMP_JOB_FK
DB-42P01: com.edb.util.PSQLException: ERROR: relation "hr.jobs" does not exist
Creating Constraint: EMP_MANAGER_FK
Schema HR imported with errors.
One or more schema objects could not be imported during the migration
process. Please review the migration output for more details.
Migration logs have been saved to
/home/user/.enterprisedb/migration-toolkit/logs
****      ****      ****      ****      ****      Migration Summary      ****      ****      ****      ****      ****
Tables: 1 out of 1
Constraints: 4 out of 6
```

(次のページに続く)

(前のページからの続き)

```

Total objects: 7
Successful count: 5
Failed count: 2
Invalid count: 0
List of failed objects
=====
Constraints
- - - - -

1. HR.EMPLOYEES.EMP_DEPT_FK

2. HR.EMPLOYEES.EMP_JOB_FK

```

移行しているテーブルには、ターゲットデータベースに存在しないテーブルの外部キー制約が含まれています。Migration Toolkit は、外部キー制約を省略してテーブルを作成します。

Migration Toolkit コマンドに `-skipFKConst` オプションを含めることにより、エラーメッセージの生成を回避できます。

テーブルへのデータのロードエラー

テーブル定義は既に移行しています。データをテーブルに移行しようとすると、エラーが発生します。

```

MTK-17001: Error Loading Data into Table: DEPARTMENTS
DB-22P04: com.edb.util.PSQLException: ERROR: extra data after last
expected column
Where: COPY departments, line 1: "10 Administration 200 1700"
Trying to reload table: DEPARTMENTS through bulk inserts with a batch
size of 100
MTK-17000: Batch entry 0 INSERT INTO hr.DEPARTMENTS VALUES (10,
Administration, 200, 1700); was aborted. Call getNextException to see
the cause.
DB-42601: java.sql.BatchUpdateException: Batch entry 0 INSERT INTO
hr.DEPARTMENTS VALUES (10, Administration, 200, 1700); was aborted.
Call getNextException to see the cause., Skipping Batch
MTK-17000:ERROR: INSERT has more expressions than target columns
Position: 48
[DEPARTMENTS] Table Data Load Summary: Total Time(s): 0.037 Total Rows:0
Data Load Summary: Total Time (sec): 0.168 Total Rows: 0 Total Size(MB):0.0
Schema HR imported with errors.

```

ターゲットデータベースのテーブル定義は、移行されたデータと一致しません。ターゲットまたはソーステーブルを変更した場合は、テーブル定義とデータに互換性があることを確認します。

エラー: 値がタイプに対して長すぎます

テーブル定義は既に移行しています。データをテーブルに移行しようとすると、次のエラーが発生します。

```
MTK-17001: Error Loading Data into Table: DEPARTMENTS
DB-22001: com.edb.util.PSQLException: ERROR: value too long for type
character(1)
Where: COPY departments, line 1, column location_id: "1700"
Trying to reload table: DEPARTMENTS through bulk inserts with a batch
size of 100
MTK-17000: Batch entry 0 INSERT INTO hr.DEPARTMENTS VALUES (10,
Administration, 200, 1700); was aborted. Call getNextException to see
the cause.
DB-22001: java.sql.BatchUpdateException: Batch entry 0 INSERT INTO
hr.DEPARTMENTS VALUES (10, Administration, 200, 1700); was aborted.
Call getNextException to see the cause., Skipping Batch
MTK-17000:ERROR: value too long for type character(1)
```

ターゲットデータベースの列は、移行されたデータを受信するのに十分な大きさではありません。この問題は、移行後にテーブル定義が変更された場合に発生する可能性があります。列名この例では location_id は、Where: で始まる行で指定されます。

COPY departments, line 1, column location_id: "1700"

問題を修正するには、列サイズを調整し、移行を再試行します。

エラー: スレッドの例外:OutOfMemoryError

- MySQL データベースから移行するときに、次のエラーが発生しました。

```
Loading Table: big_range ...
Exception in thread "dataload_job_1" java.lang.OutOfMemoryError: Java
heap space
at com.mysql.jdbc.MysqlIO.nextRow(MysqlIO.java:1370)
at com.mysql.jdbc.MysqlIO.readSingleRowSet(MysqlIO.java:2369)
at com.mysql.jdbc.MysqlIO.getResultSet(MysqlIO.java:451)
at com.mysql.jdbc.MysqlIO.readResultsForQueryOrUpdate(MysqlIO.java:2076)
at com.mysql.jdbc.MysqlIO.readAllResults(MysqlIO.java:1451)
at com.mysql.jdbc.MysqlIO.sqlQueryDirect(MysqlIO.java:1787)
at com.mysql.jdbc.Connection.execSQL(Connection.java:3277)
at com.mysql.jdbc.Connection.execSQL(Connection.java:3206)
at com.mysql.jdbc.Statement.executeQuery(Statement.java:1232)
at com.edb.dbhandler.mysql.Data.getTableData(Data.java:90)
at com.edb.DataLoader.loadDataInFastMode(DataLoader.java:594)
at com.edb.DataLoader.run(DataLoader.java:186)
at java.lang.Thread.run(Thread.java:722)
```

デフォルトでは、MySQL JDBC ドライバーは、テーブル内のすべての行を 1 回のネットワークラウンドトリップで Migration Toolkit に取得します。この動作は、大規模なテーブルの使用可能なメモリを簡単に超過する場合があります。

この問題を修正するには、移行を再試行するときにコマンドライン引数として `-fetchSize 1` を指定します。

不正なタイムスタンプと夏時間

MSSQL JDBC ドライバーを使用して SQL Server から PostgreSQL に移行する場合、エラーが観察されない場合があります。ソースデータベースに、夏時間が有効であった時の時間範囲内のタイムスタンプ値を含む行が含まれている場合、それらは間違ったタイムスタンプでターゲットデータベースに移行されます。

この問題を解決するには、`runMTK.sh` ファイルを更新し、オプション `-Duser.timezone=GMT` を提供できます。これはツールキットの実行時に動作します。

例、元の行

```
runJREApplication $JAVA_HEAP_SIZE -Dprop=$base/etc/toolkit.properties -cp $base/bin/
↳ edb-migrationtoolkit.jar:$base/lib/* com.edb.MigrationToolkit "$@"
```

これは、`-cp` オプションの前に挿入された `-Duser.timezone=GMT` で更新する必要があります。

```
runJREApplication $JAVA_HEAP_SIZE -Dprop=$base/etc/toolkit.properties -Duser.
↳ timezone=GMT -cp $base/bin/edb-migrationtoolkit.jar:$base/lib/* com.edb.
↳ MigrationToolkit "$@"
```

4.2 Error codes

Migration Toolkit は問題が発生すると、エラーコードとメッセージの形式で通知を発行します。

各エラーコードは、プレフィックス MTK-で始まり、5 つの数字が続きます。最初の 2 桁は、エラーの一般的な分類であるエラークラスを示します。最後の 3 桁は、特定のエラー条件を表します。

表には、エラークラスがリストされています。

Error Class	Description
00	正常に完了しました
01	インフォメーション
02	警告
03	一般的な恐怖
0x	予約済み
10	無効なユーザー入力
11	構成エラー
12	十分な権限
13	サポートされていない機能

次のページに続く

表 1 – 前のページからの続き

Error Class	Description
14	見つからないオブジェクト
15	スキーマの移行
16	プロシージャル言語の移行
17	データの読み込み

特定のデータベースサーバーから報告されたエラーがある場合、このエラーメッセージには **DB-** という接頭辞が付きます。たとえば、Postgres データベースサーバーの既存のテーブルが原因でテーブルの作成が失敗した場合、エラーコード **42P07** がデータベースサーバーから返されます。Migration Toolkit ログの特定のエラーは **DB-42P07** として表示されます。

4.2.1 エラーコードの概要

次の表の **Error code** 列には、Migration Toolkit のエラーコードがリストされています。**メッセージと解決策**列には、エラーコードとともに表示されるメッセージが含まれています。メッセージは、エラーの原因とその解決方法を説明します。

メッセージと解像度 列では、**\$NAME** は、実行時に適切な値に置き換えられる情報のプレースホルダーです。

クラス 02-警告

このクラスは警告メッセージを表します。

Error code	Message and resolution
MTK-02000	All the warnings that relate to this class and don't have a specific error code binding use this error code.
MTK-02001	Run 'runMTK -help' to see the usage details.
MTK-02002	Warning! The offline migration path <i>\$OFFLINE_PATH</i> does not exist, the scripts will be created under the user home folder.

クラス 10-無効なユーザー入力

このクラスは、無効なユーザー入力値を表します。

Error Code	Message and Resolution
MTK-10000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-10001	You cannot select information_schema, dbo, sys, or pg_catalog as target schemas. These are used to store metadata information in <i>\$DATABASE</i> .

次のページに続く

表 3 – 前のページからの続き

Error Code	Message and Resolution
MTK-10002	The '-schemaOnly' and '-dataOnly' options cannot be specified at the same time.
MTK-10003	The "\t" is required as a copyDelimiter, when using escapeTabDelimiter option.
MTK-10004	The -truncLoad option can only be given with the -dataOnly option.
MTK-10005	The '-dataOnly' option is applicable only for -allTables/-tables option. Schema DDL cannot be copied when this option is in place.
MTK-10006	The -constraints, -indexes and -triggers options are applicable only in the context of -allTables/-tables option.
MTK-10007	The '-vacuumAnalyze' and '-analyze' options cannot be specified at the same time.
MTK-10008	The -skipFKConst option can only be given with -constraints option.
MTK-10009	The -skipCKConst option can only be given with -constraints option.
MTK-10010	The -fastCopy option cannot be specified with -schemaOnly option.
MTK-10011	The -skipColDefaultClause cannot be specified with -dataOnly option.
MTK-10012	The '-customColTypeMapping' and '-customColTypeMappingFile' options cannot be specified at the same time.
MTK-10013	Provided default date time must be in following format 'yyyy-MM-dd_HH:mm:ss'. Time portion is optional, to specify time, the underscore symbol '_' is necessary.
MTK-10014	You cannot select MySQL and Sybase as the target database for migration.
MTK-10015	Copy delimiter should be a single character.
MTK-10016	Batch size should be between 1 - 50000.
MTK-10017	Invalid number for batch size, use a number from 1-50000.
MTK-10018	Copy Batch Size should be greater than 0.
MTK-10019	Invalid number for Copy Batch Size, use a number > 0.
MTK-10020	Fetch Size should be greater than 0.
MTK-10021	Invalid number for Fetch Size, use a number > 0.
MTK-10022	One or more of the command-line arguments is invalid.
MTK-10023	The -targetDBVersion value should be specified using <i>major.*minor*</i> pattern.
MTK-10024	The -targetDBVersion can only be used with -offlineMigration mode.
MTK-10025	Options (-constraints I -indexes I -triggers I -tables I -views I -sequences I -procs I -funcs I -packages I -synonyms) cannot be used with multiple schemas option.
MTK-10026	The -allSchemas option should be specified as the last option in the command-line options list.
MTK-10027	You have specified invalid command-line arguments. Run 'runMTK -help' to see the usage details.
MTK-10028	The -replaceNullChar cannot be specified with -schemaOnly option.
MTK-10029	The -nullReplacementChar can only be specified with -replaceNullChar option.
MTK-10030	The -ignoreCheckConstFilter can only be given with -constraints option.
MTK-10031	The -enableConstBeforeDataLoad can only be given with -truncLoad option.
MTK-10032	The retryCount value should be greater than 0.
MTK-10033	Invalid number for retryCount, use a number > 0.
MTK-10034	The loaderCount value should be greater than 0.
MTK-10035	Invalid number for loaderCount, use a number > 0.
MTK-10036	Cannot use singleDataFile option for offline data migration in COPY format.

次のページに続く

表 3 – 前のページからの続き

Error Code	Message and Resolution
MTK-10037	The offline data migration in COPY format is supported only when PostgreSQL or EnterpriseDB (EDB Postgres Advanced Server) is the target database.
MTK-10038	The <i>\$SCHEMA</i> cannot be used as schema name in <i>\$DATABASE</i> . Choose a different schema name via -targetSchema option.
MTK-10039	Log file size should be greater than 0.
MTK-10040	Log file count should be greater than or equal to 0.
MTK-10041	Offline migration can only be used with schema only option.
MTK-10042	The copyViaDBLinkOra option can only be used for copying data from Oracle to EnterpriseDB.
MTK-10043	The -recreateConst option can only be given with the -dataOnly option.
MTK-10044	The <i>\$DATABASE</i> database type is not supported by Migration Toolkit. Specify a valid database type string (i.e., EnterpriseDB, Postgres, Oracle, SQLServer, Sybase, or MySQL).
MTK-10045	The URL specified for the <i>\$DATABASE</i> database is invalid. Check the connectivity credentials.
MTK-10046	The -escapeKeywords value may be true or false.
MTK-10047	The -useOraCase option can only be used for migration from Oracle.
MTK-10048	The Postgres exported snapshot id is invalid.
MTK-10049	The URL specified for the Oracle database is not supported by dblink_oracle. Check the connectivity credentials and provide a valid URL.
MTK-10050	The schema <i>\$SCHEMA</i> not found on source database.
MTK-10051	The skipUserSchemaCreation option can only be used for user migration from Oracle to EnterpriseDB.
MTK-10052	The -truncLoad option cannot be used with offline data migration.
MTK-10053	The -parallelLoadRowLimit should be greater than 0.
MTK-10054	Invalid number for -parallelLoadRowLimit, use an integer in the range of 1 to \$MAX_LIMIT.
MTK-10055	The -parallelLoadRowLimit cannot be specified with -schemaOnly option.
MTK-10056	The -loaderCount cannot be specified with -schemaOnly option.
MTK-10057	The -tableLoaderLimit should be greater than 0.
MTK-10058	Invalid number for -tableLoaderLimit, use an integer in the range of 1 to \$MAX_LIMIT.
MTK-10059	The -tableLoaderLimit cannot be specified with -schemaOnly option.
MTK-10060	The parallel loading options (-loaderCount, -parallelLoadRowLimit, -tableLoaderLimit) are only supported for Oracle, EnterpriseDB and PostgreSQL source databases.
MTK-10061	The -fastCopy cannot be specified with table-level parallel loading context (-tableLoaderLimit > 1).
MTK-10062	The '-copyViaDBLinkOra' option cannot be used with offline data migration.
MTK-10063	The permutation of source database type <i>\$DATABASE</i> and target database type <i>\$DATABASE</i> is not supported.
MTK-10064	The '-tables' and '-excludeTables' options cannot be specified at the same time. Please specify one of the options.

次のページに続く

表 3 – 前のページからの続き

Error Code	Message and Resolution
MTK-10065	The '-views' and '-excludeViews' options cannot be specified at the same time. Please specify one of the options.
MTK-10066	The table name should be schema qualified when used in multiple schemas context.
MTK-10067	The '-sequences' and '-excludeSequences' options cannot be specified at the same time. Please specify one of the options.
MTK-10068	The '-procs' and '-excludeProcs' options cannot be specified at the same time. Please specify one of the options.
MTK-10069	The '-funcs' and '-excludeFuncs' options cannot be specified at the same time. Please specify one of the options.
MTK-10070	The '-packages' and '-excludePackages' options cannot be specified at the same time. Please specify one of the options.
MTK-10071	The '-synonyms' and '-excludeSynonyms' options cannot be specified at the same time. Please specify one of the options.
MTK-10072	The '-queues' and '-excludeQueues' options cannot be specified at the same time. Please specify one of the options.
MTK-10073	The connRetryCount value should be between 0 and 50.
MTK-10074	Invalid value for connRetryCount, use a number between 0 and 50.
MTK-10075	The connRetryInterval value should be between 0 and 900.
MTK-10076	Invalid value for 'connRetryInterval', use a number between 0 and 900.
MTK-10077	The '-pgIdleTxSessionTimeout' value should be greater than 0.

クラス 11-構成の問題

このクラスは、`toolkit.properties` ファイルまたは Migration Toolkit で使用されるその他の構成ファイルの無効な構成設定を表します。

Error code	Message and resolution
MTK-11000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-11001	The properties file containing table filter clause cannot be loaded.
MTK-11002	The custom type mapping file couldn't be loaded. Reason: <i>\$REASON</i> .
MTK-11003	The custom type mapping file contains an invalid mapping entry.
MTK-11004	The custom type mapping file doesn't contain any mapping entry.
MTK-11005	Connectivity information for source database is not available.
MTK-11006	Connectivity information for target database is not available.
MTK-11007	Unable to create the logs directory. <i>\$PATH</i> .
MTK-11008	The properties file containing table columns filter list cannot be loaded.
MTK-11009	Error Connecting Database <i>\$DATABASE</i> .
MTK-11010	Error loading <i>\$DBLINKORA_FILE</i> file.

[次のページに続く](#)

表 4 – 前のページからの続き

Error code	Message and resolution
MTK-11011	Error while loading DBLink Ora module. <i>\$DBLINKORA_MODULE</i> . Verify that dblink_ora is installed/configured on target EnterpriseDB server. Please see the Database Compatibility for Oracle Developer's Guide for more information about installing and configuring the dblink_ora module.
MTK-11012	Error while loading given DBLink_Ora module. <i>\$DBLINKORA_MODULE</i> . Verify that dblink_ora is installed/configured on target EnterpriseDB server. Please see the Database Compatibility for Oracle Developer's Guide for more information about installing and configuring the dblink_ora module.
MTK-11013	Could not load DBLinkOra Module.
MTK-11014	Error connecting to DBLinkOra.
MTK-11015	The connection credentials file <i>\$TOOLKIT_PROP_FILE</i> is not secure and accessible to group/others users. This file contains plain passwords and should be restricted to Migration Toolkit owner user only.
MTK-11016	Data loader task failed to initialize. Reason: \$REASON

クラス 12-十分な権限

このクラスは、データベースのユーザー/ロール情報をロードするための不十分な特権エラーを表します。

Error code	Message and resolution
MTK-12000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-12001	The user/role migration failed due to insufficient privileges. Grant the user SELECT privilege on the following Oracle catalogs: DBA_ROLES, DBA_USERS, DBA_TAB_PRIVS, DBA_PROFILES, DBA_ROLE_PRIVS, ROLE_ROLE_PRIVS, DBA_SYS_PRIVS.
MTK-12002	The migration of privileges failed due to insufficient privileges. Grant the user SELECT privilege on the following Oracle catalog: dba_tab_privs.

クラス 13-サポートされていない機能

このクラスは、サポートされていないオブジェクトと句の移行に関連するエラーを表します。ターゲットデータベースが指定されたオブジェクトを実装していないか、Migration Toolkit がその移行を処理しません。

Error code	Message and resolution
MTK-13000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-13001	Skipping index as Oracle does not allow multiple indexes against same column list.

次のページに続く

表 6 – 前のページからの続き

Error code	Message and resolution
MTK-13002	For <i>\$DATABASE</i> views migration is not supported.
MTK-13003	For <i>\$DATABASE</i> roles migration is not supported.
MTK-13004	You cannot migrate triggers, sequences, procedures, functions, packages and synonyms for MySQL, SQL Server, and Sybase databases.
MTK-13005	You cannot migrate sequences and packages for SQL Server database.
MTK-13006	Given trigger is not migrated, the trigger is owned by a different user schema and cross-schema triggers are not supported by EnterpriseDB.
MTK-13007	The given trigger is not migrated, the trigger has WHEN clause which is not supported by EnterpriseDB.
MTK-13008	Skipping Database Link <i>\$DATABASE_LINK</i> . EnterpriseDB currently does not support this type of Database Link.
MTK-13009	Warning! Skipping migration of trigger <i>\$TRIGGER</i> , currently non-table triggers are not supported in target database.
MTK-13010	You cannot migrate procedures, packages, synonyms and database links to PostgreSQL database.
MTK-13011	Domain objects are not supported in target database.
MTK-13012	Rules are not supported in <i>\$DATABASE</i> database.
MTK-13013	The database type is not supported.
MTK-13015	<i>\$TYPE</i> is Not Supported by COPY.
MTK-13016	The migration to PostgreSQL is supported only when Oracle or PostgreSQL is the source database.
MTK-13017	Groups are not supported in <i>\$DATABASE</i> database.
MTK-13018	Profile migration is not supported in <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> permutation.
MTK-13019	Cannot migrate unknown Profile <i>\$PROFILE</i> .
MTK-13020	Profiles cannot be migrated to database version <i>\$VERSION</i> .
MTK-13021	Password Profile verify function <i>\$MY_VERIFICATION_FUNCTION</i> must be explicitly migrated.
MTK-13022	Advanced Queues migration is not supported from <i>\$SRC_DB</i> database to <i>\$TARGET_DB</i> database.
MTK-13023	Advanced Queues cannot be migrated to database version <i>\$VERSION</i> .
MTK-13024	The INTERVAL partition is not supported in <i>\$DATABASE</i> , the table will be migrated without INTERVAL definition.
MTK-13025	Warning! User profile migration is not supported in target database version <i>\$VERSION</i> . The profile " <i>\$PROFILE</i> " for user " <i>\$USER</i> " will be skipped.
MTK-13026	Object Type migration is not supported from <i>\$SRC_DB</i> to <i>\$TARGET_DB</i> .
MTK-13027	Migration is not supported from <i>\$DATABASE</i> database for PDB Admin schemas/users with PDB_DBA role.
MTK-13028	PostgreSQL 10 and earlier versions do not support stored procedures.

クラス 14-不足しているオブジェクト

このクラスは、ソースデータベースでのメタデータ情報の検出の失敗を表します。

Error code	Message and resolution
MTK-14000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-14001	One or more tables are missing from the source <i>\$DATABASE</i> database.
MTK-14002	One or more tables couldn't be found in the source <i>\$DATABASE</i> database. With -tables mode, the table name should be in uppercase unless it is case-sensitive.
MTK-14003	One or more users couldn't be found in the source <i>\$DATABASE</i> database. With -users mode, the user name should be in uppercase unless it is case-sensitive.

クラス 15-スキーマの移行

このクラスは、テーブル、制約、インデックス、シノニム、ビュー、ユーザー、ロールなどの非プロシージャデータベースオブジェクトに関連する移行の問題を表します。

Error code	Message and resolution
MTK-15000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-15001	Error creating constraint <i>\$CONSTRAINT</i> .
MTK-15002	Error creating role <i>\$ROLE</i> .
MTK-15003	Error granting given privilege <i>\$PRIVILEGE</i> on <i>\$OBJECT</i> to <i>\$USER</i> .
MTK-15004	Error granting <i>\$ROLE</i> to <i>\$USER</i> .
MTK-15005	Error granting privilege <i>\$REASON</i> .
MTK-15006	Error creating user <i>\$USER</i> .
MTK-15007	Error creating index <i>\$INDEX</i> .
MTK-15008	Error creating view <i>\$VIEW</i> .
MTK-15009	Error creating materialized view <i>\$MVIEW</i> .
MTK-15010	Error creating public synonym <i>\$SYNONYM</i> .
MTK-15011	Error creating private synonym <i>\$SYNONYM</i> .
MTK-15012	Error creating sequence <i>\$SEQUENCE</i> .
MTK-15013	Error creating table <i>\$TABLE</i> .
MTK-15014	Error creating database link <i>\$DATABASE_LINK</i> .
MTK-15015	Error creating given object type <i>\$OBJECT_TYPE</i> .
MTK-15016	The table <i>\$TABLE</i> could not be created in <i>\$DATABASE</i> database.
MTK-15017	The linked schema <i>\$LINKED_SCHEMA</i> doesn't exist in the target database. Create the schema and then retry.
MTK-15018	Error creating domain <i>\$DOMAIN</i> .
MTK-15019	Error creating custom data type <i>\$DATA_TYPE</i> .
MTK-15020	Error creating membership for group <i>\$GROUP</i> .

[次のページに続く](#)

表 8 – 前のページからの続き

Error code	Message and resolution
MTK-15021	Table name <i>\$TABLE</i> does not have a schema qualifier. With multiple schema migration context, each table should be schema qualified.
MTK-15022	Schema qualifier <i>\$SCHEMA</i> does not match the schema list.
MTK-15023	The table metadata information is not available.
MTK-15024	Tables list is not initialized yet.
MTK-15025	Error while getting database metadata information.
MTK-15026	Error creating group <i>\$GROUP</i> .
MTK-15027	Error creating Profile <i>\$PROFILE</i> .

クラス 16-プロシージャル言語の移行

このクラスは、プロシージャ、ファンクション、パッケージ、無名ブロック、トリガー、ルールなどのプロシージャル言語に基づいたデータベースオブジェクトに関連する移行の問題を表します。

Error code	Message and resolution
MTK-16000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-16001	Error Creating Trigger <i>\$TRIGGER</i> .
MTK-16002	Error Creating Rule <i>\$RULE</i> .
MTK-16003	Error Creating Package Spec <i>\$PACKAGE</i> .
MTK-16004	Error Creating Package Body <i>\$PACKAGE</i> .
MTK-16005	Package Body is invalid, skipping... Note: This error message also appears when a package specification is successfully migrated, but there is no corresponding package body in the source database. In this case, the package specification should function properly in the target database despite the appearance of the error message.
MTK-16006	Error Creating Procedure <i>\$PROCEDURE</i> .
MTK-16007	Error Creating Function <i>\$FUNCTION</i> .
MTK-16008	Error Creating Anonymous Block <i>\$BLOCK</i> .
MTK-16009	Error creating Queue <i>\$QUEUE</i> .

クラス 17-データのロード

このクラスは、ソースからターゲットデータベースにデータをコピーするときに発生する可能性のあるエラーを表します。

Error code	Message and resolution
MTK-17000	All the errors that relate to this class and don't have a specific error code binding use this error code.
MTK-17001	Error Loading Data into Table: <i>\$TABLE</i>
MTK-17002	Encoding Conversion encountered some characters that will be loaded using Bulk Inserts instead.
MTK-17003	Error in copy tables <i>\$REASON</i> .
MTK-17004	Invalid DataType.
MTK-17005	This Table Contains CLOB data, Marked for Bulk Insert Loading.
MTK-17006	One of the table loader threads failed while copying data for table <i>\$TABLE</i> .

4.3 FAQ

Migration Toolkit はパッケージの移行をサポートしていますか？

Migration Toolkit は、Oracle データベースから EDB Postgres Advanced Server へのパッケージの移行をサポートしています。EDB Postgres Advanced Server が提供する移行サポートについては、[Functionality overview](#) を参照してください。

ソースデータベースからデータのみを転送する方法はありますか？

はい。データ転送は、オンラインまたはオフラインの移行の一部としてサポートされています。

Migration Toolkit は、CLOB データ型のデータを含むテーブルの移行をサポートしていますか？

Migration Toolkit は、CLOB タイプのデータを含むテーブルの移行をサポートしています。

EDB Postgres Advanced Server は列挙データ型をサポートしていますか？

EDB Postgres Advanced Server は、現在列挙データ型をサポートしていませんが、将来のリリースでサポートされる予定です。それまでは、チェック制約を使用して、EDB Postgres Advanced Server データベースに追加されるデータを制限できます。チェック制約は、列が取ることができる有効な値のリストを定義します。

次のコードサンプルには、列の値を 3 つの dept タイプのいずれかに制限するチェック制約の簡単な例が含まれています。sales、admin または technical。

```
CREATE TABLE emp (
  emp_id INT NOT NULL PRIMARY KEY,
  dept VARCHAR(255) NOT NULL,

  CHECK (dept IN (sales, admin, technical))
);
```

有効な dept タイプを入力してチェック制約をテストする場合、INSERT ステートメントはエラーなしで機能します。

```
test=# INSERT INTO emp VALUES (7324, sales);

INSERT 0 1
```

制約 `support` に含まれない値を挿入しようとすると、EDB Postgres Advanced Server はエラーを返します。

```
test=# INSERT INTO emp VALUES (7325, support);

ERROR: new row for relation "emp" violates check constraint "emp_dept_check"
```

EDB Postgres Advanced Server はマテリアライズドビューをサポートしていますか？

Postgres は、Oracle データベースと互換性のあるマテリアライズドビューをサポートしていません。Postgres でマテリアライズドビュー/サマリーテーブルを設定するには、サマリーテーブルを維持するトリガーを手動で作成する必要があります。自動クエリリライトは現在サポートされていません。サマリーテーブルの存在をアプリケーションに認識させる必要があります。

TIME データ型を含む MySQL データベースから移行しようとすると、次のエラーが発生します。テーブルへのデータのロード中にエラーが発生しました。Time のフォーマットが正しくありません。Postgres は MySQL ``TIME`` データ型をサポートしていますか？

Postgres は、時間コンポーネントの値が 24 を超えない限り、TIME データ型を保存することに問題はありません。

Postgres とは異なり、MySQL TIME データ型では、TIME または INTERVAL 値を表す値を保存できます。INTERVAL 値を表す MySQL TIME カラムに格納されている値は、有効な Postgres TIMESTAMP 値の許容範囲外である可能性があります。移行プロセス中に、Postgres は TIME データ列に格納されている値を範囲外と認識する場合、エラーを結果ます。

接続再試行機能はどのようなユースケースをサポートしていますか？

データベーススコープ 接続再試行機能により、Migration Toolkit はターゲットデータベースに再接続できます。ソースデータベースの問題の再試行は、現在サポートされていません。

移行スコープ この機能により、Migration Toolkit はデータの移行を再試行できます。スキーマの移行に関する問題の再試行は、現在サポートされていません。

モダリティスコープ この再接続機能は、データ移行モード `-dataOnly` で利用できます。 `-dataOnly` および `-schemaOnly` オプションを指定せずに移行を実行する場合、データ移行ステップでも使用できます。

データの移行中にエラーが発生し、1 つ以上のテーブルのデータが完全に移行されない場合はどうすればよいですか？

Migration Toolkit が正常に再接続できず、1 つ以上のテーブルが完全に移行されていない場合は、データ移行全体を `-dataOnly` オプションを使用して再起動するか、以前に完全に移行されなかったテーブルのみを移行するように Migration Toolkit を構成します。 `connRetryCount`、`connRetryInterval`、および `abortConnOnFailure` を使用して、再試行構成オプションを変更できます。

前回の Migration Toolkit のデータの移行の処理中に、ソースデータベースが書き込みトランザクションを受け入れていた場合、完全なすべてのテーブルのデータの移行を再度実行して、宛先データベースに移行されたデータが一貫した状態であることを確認します。

前回の移行の実行中にソースデータベースで書き込みトランザクションが実行されなかったことが確認できた場合、完全に移行されていないテーブルのみを移行しても安全です。

スキーマの移行中にエラーが発生し、すべてのスキーマオブジェクトが完全に移行されない場合はどうすればよいですか？

残念ながら、スキーマ移行エラーの再接続は、現時点ではサポートされていません。Migration Toolkit ですべてのスキーマを移行できない場合は、スキーマ全体の移行を `-schemaOnly` オプションを使用して再度リスタートするか、以前の実行で完全に移行されなかったスキーマのみを移行するように Migration Toolkit を構成します。