



Informatica® Multidomain MDM
10.4 HotFix 1

Oracle の Zero Downtime アップグレードガイド

Informatica Multidomain MDM Oracle の Zero Downtime アップグレードガイド

10.4 HotFix 1

2020 年 9 月

© 著作権 Informatica LLC 2011, 2021

本ソフトウェアおよびマニュアルは、使用および開示の制限を定めた個別の使用許諾契約のもとでのみ提供されています。本マニュアルのいかなる部分も、いかなる手段（電子的複製、写真複製、録音など）によっても、Informatica LLC の事前の承諾なしに複製または転載することは禁じられています。

米政府の権利プログラム、ソフトウェア、データベース、および関連文書や技術データは、米国政府の顧客に配信され、「商用コンピュータソフトウェア」または「商業技術データ」は、該当する連邦政府の取得規制と代理店固有の補足規定に基づきます。このように、使用、複製、開示、変更、および適応は、適用される政府の契約に規定されている制限およびライセンス条項に従うものとし、政府契約の条項によって適当な範囲において、FAR 52.227-19、商用コンピュータソフトウェアライセンスの追加権利を規定します。

Informatica および Informatica ロゴは、米国およびその他の国における Informatica LLC の商標または登録商標です。Informatica の商標の最新リストは、Web (<https://www.informatica.com/trademarks.html>) にあります。その他の企業名および製品名は、それぞれの企業の商標または登録商標です。

本ソフトウェアまたはドキュメンテーション（あるいはその両方）の一部は、第三者が保有する著作権の対象となります。必要な第三者の通知は、製品に含まれています。

本マニュアルの情報は、予告なしに変更されることがあります。このドキュメントで問題が見つかった場合は、infa_documentation@informatica.com までご報告ください。

Informatica 製品は、それらが提供される契約の条件に従って保証されます。Informatica は、商品性、特定目的への適合性、非侵害性の保証等を含めて、明示的または黙示的ないかなる種類の保証をせず、本マニュアルの情報を「現状のまま」提供するものとします。

発行日: 2021-04-21

目次

序文	4
Informatica のリソース.....	4
Informatica Network.....	4
Informatica ナレッジベース.....	4
Informatica マニュアル.....	4
Informatica 製品可用性マトリックス.....	5
Informatica Velocity.....	5
Informatica Marketplace.....	5
Informatica グローバルカスタマサポート.....	5
 第 1 章 : Zero Downtime の概要	6
Zero Downtime の概要.....	6
ZDT 用の Oracle GoldenGate プロセス.....	6
Zero Downtime の前提条件.....	7
Zero Downtime を使用したアップグレードのワークフロー.....	7
 第 2 章 : Zero Downtime を使用したアップグレード	9
以前のロードテーブルデータの転送.....	9
パッシブ環境から制御するアップグレード手順.....	10
アクティブ環境から制御するアップグレード手順.....	17
 第 3 章 : トラブルシューティング	19
バックフィルタスクが登録されている場合にバッチジョブが失敗する.....	19
ターゲットでの一致のリセット.....	19
レプリケーションが機能しない.....	19
 索引	21

序文

Zero Downtime 環境で Multidomain MDM をアップグレードするには、『Informatica® Multidomain MDM Zero Downtime のアップグレードガイド』の手順に従います。Zero Downtime はオプションのライセンス機能で、Multidomain MDM のアップグレード中のダウンタイムを最小限にすることができます。

Informatica のリソース

Informatica は、Informatica Network やその他のオンラインポータルを通じてさまざまな製品リソースを提供しています。リソースを使用して Informatica 製品とソリューションを最大限に活用し、その他の Informatica ユーザーや各分野の専門家から知見を得ることができます。

Informatica Network

Informatica Network は、Informatica ナレッジベースや Informatica グローバルカスタマサポートなど、多くのリソースへの入口です。Informatica Network を利用するには、<https://network.informatica.com> にアクセスしてください。

Informatica Network メンバーは、次のオプションを利用できます。

- ナレッジベースで製品リソースを検索できます。
- 製品の提供情報を表示できます。
- サポートケースを作成して確認できます。
- 最寄りの Informatica ユーザーグループネットワークを検索して、他のユーザーと共同作業を行えます。

Informatica ナレッジベース

Informatica ナレッジベースを使用して、ハウツー記事、ベストプラクティス、よくある質問に対する回答など、製品リソースを見つけることができます。

ナレッジベースを検索するには、<https://search.informatica.com> にアクセスしてください。ナレッジベースに関する質問、コメント、ご意見の連絡先は、Informatica ナレッジベースチーム (KB_Feedback@informatica.com) です。

Informatica マニュアル

Informatica マニュアルポータルでは、最新および最近の製品リリースに関するドキュメントの膨大なライブラリを参照できます。マニュアルポータルを利用するには、<https://docs.informatica.com> にアクセスしてください。

製品マニュアルに関する質問、コメント、ご意見については、Informatica マニュアルチーム (infa_documentation@informatica.com) までご連絡ください。

Informatica 製品可用性マトリックス

製品可用性マトリックス (PAM) には、製品リリースでサポートされるオペレーティングシステム、データベースなどのデータソースおよびターゲットが示されています。Informatica PAM は、<https://network.informatica.com/community/informatica-network/product-availability-matrices> で参照できます。

Informatica Velocity

Informatica Velocity は、Informatica プロフェッショナルサービスが開発したヒントとベストプラクティスのコレクションで、多数のデータ管理プロジェクトから得た実体験に基づいています。Informatica Velocity には、世界中の組織と連携してデータ管理ソリューションを計画、開発、デプロイ、管理する Informatica コンサルタントによる集合知を表しています。

Informatica Velocity リソースには、<http://velocity.informatica.com> からアクセスしてください。Informatica Velocity についての質問、コメント、またはアイデアがある場合は、ips@informatica.com から Informatica プロフェッショナルサービスにお問い合わせください。

Informatica Marketplace

Informatica Marketplace は、お使いの Informatica 製品を拡張したり強化したりするソリューションを検索できるフォーラムです。Marketplace で、Informatica デベロッパーやパートナーからの多数のソリューションを活用すれば、生産性を向上したり、プロジェクトでの実装時間を短縮したりできます。Informatica Marketplace は、<https://marketplace.informatica.com> からアクセスしてください。

Informatica グローバルカスタマサポート

電話または Informatica Network からグローバルサポートセンターに連絡できます。

各地域の Informatica グローバルカスタマサポートの電話番号は、Informatica Web サイト (<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>) を参照してください。

Informatica Network でオンラインサポートリソースを見つけるには、<https://network.informatica.com> にアクセスし、eSupport オプションを選択します。

第 1 章

Zero Downtime の概要

この章では、以下の項目について説明します。

- [Zero Downtime の概要, 6 ページ](#)
- [ZDT 用の Oracle GoldenGate プロセス, 6 ページ](#)
- [Zero Downtime の前提条件, 7 ページ](#)
- [Zero Downtime を使用したアップグレードのワークフロー, 7 ページ](#)

Zero Downtime の概要

MDM Hub へのアクセスを中断せずに、Zero Downtime (ZDT) を使用して Multidomain MDM をアップグレードします。バッチおよびサービス統合フレームワーク (SIF) のユーザープロセスは、ZDT アップグレードプロセス中に実行できます。

ZDT を使用するには、パッシブ環境とアクティブ環境の 2 つの環境があります。Oracle GoldenGate を使用して、環境間のデータレプリケーションとメッセージストリームを管理します。パッシブ環境をアップグレードしている間、MDM ユーザーはアクティブ環境の MDM Hub にアクセスできます。アップグレード中に MDM Hub に加えられた変更は、アクティブ環境から変更を複製する機能には影響しません。バックフィルメカニズムにより、アップグレードによって引き起こされる MDM Hub メタデータへの影響が処理されます。

ZDT 用の Oracle GoldenGate プロセス

ZDT には、一連の Oracle GoldenGate プロセスが必要です。一部のプロセスでは、アクティブ環境からパッシブ環境にデータを抽出してレプリケートします。

次の表は、プロセスグループ名で使用するプレフィックスを示しています。

プレフィックス	プロセスタイプ	目的
E_、P_	EXTRACT	アクティブ環境にある MDM Hub からデータを抽出します。
R_	REPLICAT	パッシブ環境にある MDM Hub へデータをレプリケートします。

たとえば、次のリストは、ENVA および ENVB のプロセスを示しています。ここで、ENVA はアクティブ環境です。

```
GGSCI (hostname) 13> info all
```

Program	Status	Group
MANAGER	RUNNING	
EXTRACT	RUNNING	E_ENVA
REPLICAT	RUNNING	R_ENVB
REPLICAT	RUNNING	R_ENVB

Zero Downtime の前提条件

ソースシステムとターゲットシステムを特定し、データベースソフトウェアが両方のシステムにあることを確認し、両方のシステムに Oracle GoldenGate をインストールして、レプリケーション用に MDM Hub ストアのデータベースを設定する必要があります。ZDT のインストールの詳細については、*Oracle 用の Multidomain MDM Zero Downtime のインストールガイド*を参照してください。

Zero Downtime を使用したアップグレードのワークフロー

Zero Downtime を使用してアップグレードする場合、次の一般的なアクティビティを実行します。

1. パッシブ環境をアップグレードします。
2. パッシブ環境とアクティブ環境を切り替えます。
3. 以前のアクティブ環境を削除します。
4. 新しいアクティブ環境のコピーから新しいパッシブ環境を作成します。
5. アップグレード中に発生するすべてのデータ変更をレプリケートして、アップグレードプロセスの終了時に環境が同じになるようにします。

このガイドでは、一部の手順は条件付きです。これらの手順は、特定のタイプの更新またはアップグレードを行う場合にのみ実行してください。

条件付きの手順には、次のプレフィックスが 1 つ以上付いています。

- **スキーマの更新。** ビジネス要件に基づいて、オペレーショナルリファレンスストアスキーマを変更した場合。
- **データ変更によるスキーマの更新。** データベース内のオペレーショナルリファレンスストアスキーマおよび一部のデータを変更した場合。
- **MDM のアップグレード。** Multidomain MDM を、新しいメジャー、マイナー、または HotFix のリリースにアップグレードする場合、または緊急のバグ修正を適用する場合。
- **インフラストラクチャのアップグレード。** Multidomain MDM が実行されているのと同じ環境で、他のソフトウェアまたはハードウェアをアップグレードする場合。

ZDT アップグレードの手順は、シェルスクリプトなどのコマンドラインインターフェイス、またはコマンドラインジョブスケジューラから実行できます。アップグレードは単一の制御フローから実行されるため、アップグレードプロセスをほぼ完全に自動化できます。ZDT のアップグレード手順には、アクティブ環境とパッシブ環境の間のメッセージング、レプリケーション制御の保持、およびバックフィルの統合の手順が含まれます。

バージョン 9.5.0 以前からの Multidomain MDM のアップグレード

Multidomain MDM バージョン 9.5.1 では、マスタデータベーススキーマのデータ構造が変更されました。バージョン 9.5.0 以前からアップグレードする場合は、アップグレードサイクルを開始する前に、準備サイクルを実行する必要があります。準備サイクルでは、アップグレード前に解決が必要なデータの問題を特定します。準備サイクルの後、ZDT を使用してアップグレードすると、アップグレードサイクルによってスキーマのデータ構造が更新されます。

このガイドの、バージョン 9.5.0 以前からのアップグレード手順を含むバージョンを要求するには、Informatica グローバルカスタマサポートにお問い合わせください。

第 2 章

Zero Downtime を使用したアップグレード

この章では、以下の項目について説明します。

- [以前のロードテーブルデータの転送, 9 ページ](#)
- [パッシブ環境から制御するアップグレード手順, 10 ページ](#)
- [アクティブ環境から制御するアップグレード手順, 17 ページ](#)

以前のロードテーブルデータの転送

アクティブ環境の以前のロードテーブルからパッシブ環境の以前のロードテーブルにデータをコピーできます。以前のロードテーブルのデータは、レプリケーションプロセス中に転送されません。以前のロードテーブルをレプリケートしない場合、パッシブ環境がアクティブになった後に最初に行うステージバッチジョブでは、ランディングテーブル内のすべてのデータが処理される可能性があります。以前のロードテーブルの名前は `_PRL` で終わります。

注: ステージバッチジョブを実行する前に、アップグレードが完了していることを確認してください。アップグレードが完了する前にステージジョブを実行すると、ステージバッチジョブによってソースの以前のロードテーブルに追加されるデータは、ターゲットの以前のロードテーブルに追加されません。

1. アクティブ環境で、以前のロードテーブルのパラメータファイル `prl_exdpd.prm` および `prl_impdp.prm` を生成します。

アクティブ環境の SQL*Plus から、次のコマンドを実行します。

`prl_exdpd.prm:`

```
directory=OUTPUT_DIRECTORY
logfile=PRL_TABLES_EXDPD_LOG.log
dumpfile=PRL.dmp
include=table:"LIKE '%PRL'"
CONTENT=DATA_ONLY
```

`prl_impdp.prm:`

```
directory=INPUT_DIRECTORY
logfile=PRL_TABLES_IMPDP_LOG.log
dumpfile=PRL.dmp
TABLE_EXISTS_ACTION=APPEND
CONTENT=DATA_ONLY
```

パラメータファイルは GGS/dirprm ディレクトリに生成されます。

2. コマンドプロンプトを開き、GGS/dirprm ディレクトリに移動します。

3. アクティブ環境で、以前のロードテーブルのデータをエクスポートします。
`expdp <active Operational Reference Store name>/<TNS password>@<TNS name> parfile=prl_expdp.prm`
`prl.dmp` ファイルが GGS/dirprm に生成されます。
4. PRL.dmp ファイルおよび prl_impdp.prm ファイルを、アクティブ環境の GGS/dirprm ディレクトリからパッシブ環境の GGS/dirprm ディレクトリにコピーします。
5. アクティブ環境とパッシブ環境をスワップするのが初めてでない場合は、パッシブ環境で以前のロードテーブルを切り捨てます。アクティブ環境の以前のロードテーブルからデータをインポートできるように、パッシブ環境で以前のロードテーブルを切り捨てる必要があります。
 パッシブ環境で、SQL*Plus にログインし、次のコマンドを実行します。
`TRUNCATE TABLE <Previous Load Table Name>;`
6. パッシブ環境で、コマンドプロンプトを開き、GGS/dirprm に移動します。
7. 次のコマンドを実行して、パッシブ環境の以前のロードテーブルにデータをインポートします。
`impdp <passive Operational Reference Store name>/<TNS password>@<TNS name> remap_schema=<active Operational Reference Store name>:<passive Operational Reference Store name> parfile=prl_impdp.prm`

パッシブ環境から制御するアップグレード手順

システムを確実に同期するには、アクティブ環境からパッシブ環境へのレプリケーションを実行しておく必要があります。

重要: すべてのコマンドを順番に実行してください。特に指定されていないかぎり、各プロセスが終了してから次のコマンドを実行するようにしてください。

1. **MDM のアップグレード。** アプリケーションサーバーを停止します。
2. パッシブ環境に対する読み取りサービスを無効にします。
3. コマンドプロンプトで、アップグレードプロセスを開始するには、次の swing コマンドを入力します。
`sip_ant swing_start`
 swing コマンドは、ACTIVE_UPGRADE_IND を 1 に設定し、レプリケーションの再生をオフにし、パッシブ環境のスキーマ内の古いオブジェクトをすべて削除します。
4. **MDM のアップグレード。** Multidomain MDM をアップグレードします。
 1. Hub ストアをアップグレードします。
 2. Hub サーバーをアップグレードします。
 3. プロセスサーバーをアップグレードします。
 Multidomain MDM へのアップグレードの詳細については、*Multidomain MDM のアップグレードガイド* を参照してください。
重要: アップグレードが完了したら、アプリケーションサーバーが実行されていることを確認します。
5. **スキーマの更新。** Metadata Manager コマンドラインユーティリティを実行して、変更リストを適用します。
 このユーティリティは、変更リストファイル内の変更をスキーマに適用します。例えば、変更リストを使用して、ベースオブジェクトテーブル内のカラムを追加または削除したり、カラムに信頼値を設定したりできます。
6. 一致トークンの生成バッチジョブを実行して、_STRP をサフィックスとして使用するテーブル内のすべての一致トークンを更新します。

一致トークンの生成バッチジョブは、Hub コンソールから、またはサービス統合フレームワーク（SIF）API を使用して実行できます。

実行方法	手順
Hub コン ソー ル	<ol style="list-style-type: none"> 1. MDM Hub コンソールでバッチビューアツールを開きます。 2. バッチビューアナビゲーションペインで、すべての一致トークンを再生成するベースオブジェクトを展開します。 3. 【一致トークンの生成】を展開します。 4. 一致トークンの生成に使用するバッチジョブを選択します。 5. 【すべての一致トークンを再生成】を選択します。 6. 【バッチの実行】をクリックします。
API	<p>すべてのレコードに対して一致トークンの生成バッチジョブを実行するには、fullRestripInd 属性を 1 に設定して ExecuteBatchGenerateMatchTokens 要求を使用します。</p> <p>次のサンプルコードは、C_PARTY ベースオブジェクト内のすべてのレコードに対して一致トークンを作成する ExecuteBatchGenerateMatchTokens 要求を示しています。</p> <pre> <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:siperian.api"> <soapenv:Header/> <soapenv:Body> <urn:executeBatchGenerateMatchTokens> <urn:username>admin</urn:username> <urn:password> <urn:password>admin</urn:password> <urn:encrypted>false</urn:encrypted> </urn:password> <urn:orsId>localhost-orclsnl-UTSOURCE</urn:orsId> <urn:asynchronousOptions> <urn:isAsynchronous>false</urn:isAsynchronous> <urn:jmsReplyTo></urn:jmsReplyTo> <urn:jmsCorrelationId></urn:jmsCorrelationId> </urn:asynchronousOptions> <urn:tableName>C_PARTY</urn:tableName> <urn:fullRestripInd>1</urn:fullRestripInd> </urn:executeBatchGenerateMatchTokens> </soapenv:Body> </soapenv:Envelope> </pre>

7. **MDM のアップグレードとスキーマの更新。**バックフィルテーブル C_REPOS_ZDT_BACKFILL_TASK に入力して、ベースオブジェクトテーブルで信頼バックフィルが必要なことを指定します。

sip_ant add_backfill_task メソッドを使用します。

```
sip_ant -Dnoprompt=true -noinput add_backfill_task -DprocedureName=<backfill type> -DtableName=<base object name> -DusageType=<api> -Dsequence=1
```

ここで、

- *backfill type* はバックフィルのタイプです。次のいずれかのバックフィルタイプを使用します。
 - TRUST_BACKFILL。推奨。信頼できる新しいカラムを追加する場合に使用します。このオプションは、REVALIDATE タイプおよび RECALCULATE タイプと同じプロセスを実行します。
 - REVALIDATE。検証ルールを変更または追加する場合に使用します。
 - RECALCULATE。信頼ルールを変更する場合に使用します。
 - TOKENIZE。未処理レコードにトークン化プロセスを実行する必要があるがバッチジョブを実行できない場合に使用します。

- *base object name* は、ベースオブジェクトのテーブル名です。Best Version of the Truth (BVT) を再計算するすべてのベースオブジェクトテーブルでコマンドを実行します。スキーマ更新の影響を受けたテーブルが不明な場合は、スキーマ内のすべてのベースオブジェクトテーブルに対してコマンドを実行してください。
- *api* は、バックフィルタスクを実行する API を指定します。R は Read API、W は Write API、B は Read API と Write API を表します。B を使用してください。
- *sequence* は、他のタスクとの関連でバックフィルタスクを実行する順番です。不明な場合は、1 を指定してバックフィルタスクを最初に実行します。

例えば、次のコマンドは C_CUSTOMER ベースオブジェクトに信頼バックフィルを適用します。

```
sip_ant -Dnoprompt=true -noinput add_backfill_task -DprocedureName=TRUST_BACKFILL -DtableName=C_CUSTOMER -DusageType=B -Dsequence=1
```

8. ベースオブジェクトごとにバックフィルバッチジョブを実行します。

バックフィルバッチジョブは、Hub コンソールから、またはサービス統合フレームワーク (SIF) API を使用して実行できます。

実行方法	手順
Hub コンソール	<ol style="list-style-type: none"> 1. MDM Hub コンソールでバッチビューアツールを開きます。 2. バッチビューアナのビゲーシオンペインで、バックフィルを実行するベースオブジェクトを選択します。 バックフィルバッチジョブがベースオブジェクトのバッチビューアに表示されない場合は、【バッチビューア】 > 【更新】 の順に選択します。 3. バックフィルバッチジョブを実行します。

実行方法	手順
API	<p>1. Hub サーバーが実行されていることを確認します。</p> <p>2. 以下のいずれかの API を実行します。</p> <ul style="list-style-type: none"> すべてのベースオブジェクトをバックフィルするには、ExecuteBatchBackfillAll API を使用します。 <p>注: すべてのレコードに対してバックフィルを実行するには、dirtyOnlyInd パラメータを false に設定します。</p> <ul style="list-style-type: none"> 指定したベースオブジェクトをバックフィルするには、ExecuteBatchBackfill API を使用します。 <p>重要: 要求内の rowidObjectTable 要素をコメントアウトします。</p> <p>次のサンプルコードは、C_BO_TRUST ベースオブジェクト内のレコードにバックフィルを実行する ExecuteBatchBackfill 要求を示しています。</p> <pre><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:siperian.api"> <soapenv:Header/> <soapenv:Body> <urn:ExecuteBatchBackfill> <urn:username><user name>/urn:username> <urn:password> <urn:password><password>/urn:password> </urn:password> <urn:orsId>localhost-orclsnl-UTSOURCE</urn:orsId> <urn:asynchronousOptions> <urn:isAsynchronous>false</urn:isAsynchronous> </urn:asynchronousOptions> <urn:tableName>C_BO_TRUST</urn:tableName> <!--urn:rowidObjectTable?</urn:rowidObjectTable--> <urn:dirtyOnlyInd>false</urn:dirtyOnlyInd> </urn:ExecuteBatchBackfill> </soapenv:Body> </soapenv:Envelope></pre>

9. **データ変更によるスキーマの更新。** C_AGREEMENT テーブルの Oracle GoldenGate マッピングを無効にして、マッピングを C_AGREEMENT_XREF_NEW_FROM_A テーブルに変更します。
 - a. sip_ant disable_replication コマンドを使用して、C_AGREEMENT ベースオブジェクト全体および関連するすべてのテーブルのマッピングをオフにします。


```
sip_ant -Dnoprompt=true -noinput disable_replication -DtableName=C_AGREEMENT
```
 - b. sip_ant remap コマンドを使用して、C_AGREEMENT_XREF から C_AGREEMENT_XREF_NEW_FROM_A へのマッピングを作成します。ソーステーブルとターゲットテーブルのデータは同じである必要があります。sip_ant remap コマンドは、テーブル C_AGREEMENT_XREF_NEW_FROM_A を自動的に作成します。このテーブルが存在する場合、コマンドは失敗します。次のコマンドを実行します。


```
sip_ant -Dnoprompt=true -noinput remap -DtableName=C_AGREEMENT_XREF -DmapTableName=C_AGREEMENT_XREF_A
```
10. **データ変更によるスキーマの更新。** C_AGREEMENT_XREF から C_AGREEMENT へのデータの再ロードを開始します。データが再ロードされるまで待たずに次の手順に進むことができます。
11. ユーザーの受け入れ検証を実施するには、次の手順を実行します。
 - a. システム変更番号 (SCN) を書き留めます。
 - b. ユーザーの受け入れ検証を実行します。
 - c. 書き留めた SCN にフラッシュバックします。
12. アップグレードプロセスを続行するには、次の swing コマンドを入力します。


```
sip_ant swing_continue
```

swing コマンドは次のプロセスを実行します。

1. レプリケーションの再生を開始して、パッシブ環境からデータ変更を処理します。
 2. パッシブ環境でレプリケーションキャッチアップの完了を検出します。
 3. パッシブ環境からアクティブ環境に、アクティブ環境でのバッチプロセスを禁止するメッセージを送信します。
 4. ターゲットシステムで書き込み可能な API サービスを有効にします。
 5. シーケンスを同期します。ターゲットシステム上の新しいシーケンス値は、ソースシステムよりも大きくなっています。
 6. アクティブ環境で書き込み可能な SIF サービスを無効にします。
 7. パッシブ環境で処理中のレプリケーションを完了します。
13. パッシブ環境でアプリケーションサーバーが実行されていることを確認したうえで、アクティブ環境からパッシブ環境にサービスをリダイレクトします。
- パッシブ環境で読み取りおよび書き込みサービスが有効になります。
14. **データ変更によるスキーマの更新。** C_AGREEMENT テーブルの差分を処理します。
- 実装リソースは、パッシブ環境を再ロードした後にアクティブ環境から取得されたデータを処理するために、差分を書き込みます。
15. **MDM のアップグレードとスキーマの更新。** ベースオブジェクトごとに未処理レコードに対してバックフィルジョブを実行します。

実行方法	手順
Hub コンソール	<ol style="list-style-type: none">1. MDM Hub コンソールでバッチビューアツールを開きます。2. バッチビューアナのビゲーションペインで、バックフィルを実行するベースオブジェクトを選択します。 バックフィルバッチジョブがベースオブジェクトのバッチビューアに表示されない場合は、[バッチビューア] > [更新] の順に選択します。3. 未処理のレコードのみをバックフィルするには、[未処理レコードのみ] を選択します。4. バックフィルバッチジョブを実行します。

実行方法	手順
API	<p>1. MDM Hub サーバーが実行されていることを確認します。</p> <p>2. 以下のいずれかの API を実行します。</p> <ul style="list-style-type: none"> すべてのベースオブジェクトをバックフィルするには、ExecuteBatchBackfillAll API を使用します。 <p>注: 未処理のレコードに対してのみバックフィルを実行するには、dirtyOnlyInd パラメータを true に設定します。</p> <ul style="list-style-type: none"> 指定したベースオブジェクトをバックフィルするには、ExecuteBatchBackfill API を使用します。 <p>重要: 要求内の rowidObjectTable 要素をコメントアウトします。</p> <p>次のサンプルコードは、C_BO_TRUST ベースオブジェクト内の未処理のレコードに対してバックフィルを実行する ExecuteBatchBackfill 要求を示しています。</p> <pre><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:siperian.api"> <soapenv:Header/> <soapenv:Body> <urn:ExecuteBatchBackfill> <urn:username><user name>/urn:username> <urn:password> <urn:password><password>/urn:password> </urn:password> <urn:orsId>localhost-orclsnl-UTSOURCE</urn:orsId> <urn:asynchronousOptions> <urn:isAsynchronous>false</urn:isAsynchronous> </urn:asynchronousOptions> <urn:tableName>C_BO_TRUST</urn:tableName> <!--urn:rowidObjectTable?</urn:rowidObjectTable--> <urn:dirtyOnlyInd>true</urn:dirtyOnlyInd> </urn:ExecuteBatchBackfill> </soapenv:Body> </soapenv:Envelope></pre>

16. C_REPOS_BACKFILL_TASK テーブルからバックフィルタスクを削除します。他のバッチジョブを実行できるよう、このテーブルは空にする必要があります。

```
Delete from c_repos_zdt_backfill_task;
COMMIT;
```

17. **MDM のアップグレードとスキーマの更新。** ベースオブジェクトごとに未処理レコードに対してトークン化バッチジョブを実行します。

実行方法	手順
Hub コン ソール	<p>1. MDM Hub コンソールでバッチビューアツールを開きます。</p> <p>2. バッチビューアナビゲーションペインで、すべての一致トークンを再生成するベースオブジェクトを展開します。</p> <p>3. [一致トークンの生成] を展開します。</p> <p>4. 一致トークンの生成に使用するバッチジョブを選択します。</p> <p>5. [すべての一致トークンを再生成] をクリアします。</p> <p>6. [バッチの実行] をクリックします。</p>

実行方法	手順
API	<p>未処理のレコードに対してのみ一致トークンの生成バッチジョブを実行するには、fullRestripInd 属性を 0 に設定して ExecuteBatchGenerateMatchTokens 要求を使用します。次のサンプルコードは、C_PARTY ベースオブジェクト内の未処理のレコードに対して一致トークンを作成する ExecuteBatchGenerateMatchTokens 要求を示しています。</p> <pre> <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:siperian.api"> <soapenv:Header/> <soapenv:Body> <urn:executeBatchGenerateMatchTokens> <urn:username>admin</urn:username> <urn:password> <urn:password>admin</urn:password> <urn:encrypted>false</urn:encrypted> </urn:password> <urn:orsId>localhost-orclsnl-UTSOURCE</urn:orsId> <urn:asynchronousOptions> <urn:isAsynchronous>false</urn:isAsynchronous> <urn:jmsReplyTo></urn:jmsReplyTo> <urn:jmsCorrelationId></urn:jmsCorrelationId> </urn:asynchronousOptions> <urn:tableName>C_PARTY</urn:tableName> <urn:fullRestripInd>0</urn:fullRestripInd> </urn:executeBatchGenerateMatchTokens> </soapenv:Body> </soapenv:Envelope> </pre>

18. 次のコマンドを実行して、アップグレードプロセスをファイナライズします。
sip_ant swing_finalize
swing コマンドは次のプロセスを実行します。
 1. ターゲットシステムでバッチサービスを有効にします。
 2. 次のテーブルから ZDT イベントキューを削除します。
 - C_REPOS_ZDT_EVENT_QUEUE
 - C_REPOS_ZDT_REPLICAT_EXCEPTION
 3. 次のテーブルを更新し、それらの値を設定します。
 - a. C_REPOS_ZDT_ENV_STATE; set state = NULL, state_ts = NULL, state_desc = NULL, updated_by=NULL, update_date=NULL
 - b. C_REPOS_ZDT_STATUS; set REPLICATION_TARGET_IND = 0
 - c. C_REPOS_ZDT_STATUS; set ACTIVE_UPGRADE_IND = 0
 4. アクティブ環境上のシーケンスを偶数に設定します。
 5. ZDT をデプロイ解除します。
 6. アクティブ環境（新しいソース）で抽出を設定して開始します。
19. **データ変更によるスキーマの更新。**更新中にユーザーデータの外部キーが変更された場合は、外部キーの検証を実行します。ベースオブジェクトごとに ExecuteBatchValidateFKRelationships SIF API を実行します。

ルックアップデータが更新された後にアクティブ環境で発生した違反がすべて検出されます。
20. **スキーマの更新。**違反が検出された場合は、違反を修正します。

ヒント: 重大でない違反は、アップグレードの完了後に修正できます。違反を修正できない場合は、Informatica グローバルカスタマサポートにお問い合わせください。

21. パッシブ環境から現在の SCN を取得します。

```
SQL-CMX_ORS_B> select current_scn from v$database;
```

```
CURRENT_SCN
-----
2880593
```

22. この SCN のデータ取得を使用して、パッシブ環境をエクスポートします。

```
c:> <ors_username>/<password>@<tns entry name>
directory=<DATA_PUMP_DIR_OBJECT>
dumpfile=<mrn_backup_envb.dmp>
logfile=<mrn_backup_after_upgrade.log>
parallel=8
job_name=<EXPORT_AFTER_UPGRADE>
flashback_scn=<CURRENT_SCN from the previous step>
```

23. パッシブ環境でバッチジョブを再開します。

アクティブ環境から制御するアップグレード手順

パッシブ環境でのアップグレード手順を完了後、アクティブ環境を準備する必要があります。アクティブ環境から手順を実行します。

重要: アップグレード時には、ソーススキーマを削除したうえでターゲットスキーマから再作成して、その後データベースダンプファイルをインポートする必要があります。変更リストを適用することでこのプロセスを省略しないでください。レプリケーションが機能するためには、両方のデータベースでスキーマが完全に同じである必要があります。不注意による変更を回避するために、ソースデータベースおよびターゲットデータベースでプロダクションモードを有効にします。Hub コンソールにログインし、データベースツールを選択し、データベースを選択して、プロダクションモードを有効にします。後日ターゲットデータベースに変更リストを適用する必要がある場合は、プロダクションモードを無効にして変更リストを適用することができます。

開始する前に、次のリポジトリテーブルを開き、カラムの値を書き留めます。

C_REPOS_ZDT_STATUS

すべてのカラムの値を記録します。これらの値は手順 6 で必要になります。

C_REPOS_DB_RELEASE

次のカラムの値を記録します。これらの値は手順 7 で必要になります。

- db_password, tns_name
- connection_port
- oracle_sid
- database_host
- connect_url
- database_id
- connection_type
- proxy_ind
- db_proxy_username
- db_proxy_password

- db_replication_username
- db_replication_password
- debug_ind
- debug_level
- debug_file_name
- debug_file_pat

1. **インフラストラクチャのアップグレード**。アクティブ環境でハードウェアとサードパーティソフトウェアをアップグレードします。
2. アプリケーションサーバーを停止し、接続を閉じてから、アクティブ環境でスキーマを削除して再作成します。
 - a. アクティブ環境でアプリケーションサーバーを停止します。
 - b. アクティブ環境へのすべての接続（SQL*Plus、Toad、アプリケーションサーバーなど）を閉じます。
 - c. アクティブ環境でスキーマを削除してから作成します。

```
Drop schema A - using system user (sqlplus system/password@tnsname)
SQL> drop user envA cascade;
```

```
Create schema A - using system user (sqlplus system/password@tnsname)
SQL> <hub_server_install>/resources/database/custom_scripts/oracle/import/@mk_cmx_ors_user; --
supply the schema name as A
```

3. パッシブ環境から生成されたダンプファイルをインポートして、オペレーショナルリファレンストアを再作成します。

```
C:\> impdp <dba_username> /<dba_password>@<tns_entry_name>
directory=<DATA_PUMP_DIR_OBJECT>
dumpfile==<mrn_backup_envb.dmp>
logfile=<mrn_restore_after_upgrade.log>
content=all
remap_schema=<from_user>:<to_user>
parallel=8
job_name=<RESTORE_ENVB>
```

スキーマの作成中は、次のメッセージを無視しても問題ありません。

```
ORA-39083: Object type TYPE failed to create with error:
ORA-02304: invalid object identifier literal
The type already exists, and therefore it is not re-created
```

4. イベントキューを削除し、テーブルを削除します。


```
/* Repository tables for ZDT */
delete from C_REPOS_ZDT_EVENT_QUEUE;
delete from C_REPOS_ZDT_REPLICAT_EXCEPTION;
update C_REPOS_ZDT_ENV_STATE set state = NULL, state_ts = NULL,
state_desc = NULL, updated_by=NULL, update_date=NULL;
```
5. 次のコマンドを実行して、Java ユーティリティファイルをインストールします。

```
sip_ant install_utility
```

6. swing インストールプロセスを完了するには、次のコマンドを実行します。

```
sip_ant swing_finish
```

レプリケーションパラメータファイルがインストールされます。

7. C_REPOS_DB_RELEASE テーブルで、アクティブ環境の環境固有の設定を確認します。

テーブル内の値はすべてローカルである必要があり、データベースエントリはローカルデータベースを指している必要があります。必要に応じて、スキーマを削除する前に C_REPOS_DB_RELEASE テーブルにあった値と一致するように値を更新します。

8. **MDM のアップグレード**。新しいパッシブ環境で、Multidomain MDM をアップグレードしてプロセスサーバーを設定します。

第 3 章

トラブルシューティング

この章では、以下の項目について説明します。

- [バックフィルタスクが登録されている場合にバッチジョブが失敗する, 19 ページ](#)
- [ターゲットでの一致のリセット, 19 ページ](#)
- [レプリケーションが機能しない, 19 ページ](#)

バックフィルタスクが登録されている場合にバッチジョブが失敗する

バックフィルタスクを有効にしてバッチジョブを実行すると、エラーが発生し、バッチジョブは失敗します。

MDM Hub では、バックフィルタスクが登録されている間は、ターゲットレプリケーション環境でバッチジョブを実行できません。バックフィルバッチジョブが通常のバッチジョブと同時に実行されている場合、ロックの競合が発生する可能性が高いため、MDM Hub ではバッチジョブの実行が許可されません。バックフィルタスクが完了し、バックフィルタスクの登録が解除されたら、バッチジョブを実行できます。

ターゲットでの一致のリセット

一致のリセットロジックは、ターゲット環境にレプリケートされません。MET の移行では、一致テーブルはリセットされません。これを行うにはスクリプトを使用する必要があります。トークン化バックフィルは一致テーブルに影響しません。一致関連の変更リストを適用後、一致のリセットをトリガするために、ターゲットの一致ルールに変更を加える必要があります。

レプリケーションが機能しない

deploy_zdt 呼び出しが完了しない場合、ソースデータベースとターゲットデータベースの間で ZDT レプリケーションが機能しない可能性があります。

1. すべての Oracle GoldenGate プロセスが実行されていることを確認します。RUNNING 状態でないプロセスをすべて再起動します。

この例では、ENVA にソースデータベースが、ENVB にターゲットデータベースが格納されています。

```
EXTRACT RUNNING E_ENVA  
REPLICAT ABENDED R_ENVB  
REPLICAT RUNNING R_ENVB
```

この例では、R_ENVB プロセスが ABENDED 状態です。このプロセスを再起動します。

2. ソースデータベースの C_REPOS_ZDT_EVENT_QUEUE テーブルにイベントを直接挿入します。ターゲットデータベース内の同じテーブルを開きます。ターゲットデータベーステーブルにイベントが表示されている場合は、この方向のレプリケーションは動作しています。ターゲットデータベースから検証プロセスを繰り返して、反対方向でもレプリケーションが動作していることを確認します。

例えば、次のコードを実行すると、ENVA のテーブルにイベントが追加されます。

```
insert into C_REPOS_ZDT_EVENT_QUEUE ( 'enva', -1, 'test', '', 'envb', 'test', CURRENT_TIMESTAMP,  
'EVENT_TOKEN');
```

3. Oracle GoldenGate プロセスがエラーなしで実行されているにもかかわらず、メッセージキューのレプリケーションが動作していない場合は、環境のトラブルシューティングを行う必要があります。Oracle GoldenGate ディレクトリ dirrpt に移動して、.rpt ファイルで潜在的な問題の情報を確認します。レプリケーション問題の詳細については、メタリンクに関する次の Oracle の記事を参照してください。
 1. Main Note - Oracle GoldenGate - Troubleshooting (Doc ID 1306476.1)
 2. Master Note - Oracle GoldenGate: Initial Load Techniques and References (Doc ID 1311707.1)
 3. DB Transactions Missing from Oracle GoldenGate Trail Files (Doc ID 1364852.1)
 4. GoldenGate の POC

索引

G

GoldenGate

サポートされるバージョン [7](#)

レプリケーションツール [6](#)

M

MDM Hub

サポートされるバージョン [7](#)

Z

ZDT

概要 [6](#)

ZDT (続く)

前提条件 [7](#)

ZDT アップグレードサイクル

アクティブ環境から制御する手順 [17](#)

パッシブ環境から制御する手順 [10](#)

パッシブ環境をアクティブ環境にコピー [17](#)

Zero Downtime、[参照項目 ZDT](#)

Zero Downtime を使用した MDM Hub のアップグレード
概要 [7](#)

と

トラブルシューティング

ターゲットでの一致のリセット [19](#)

バックフィルタスクを有効にしてバッチジョブを実行 [19](#)

登録されたバックフィルタスク [19](#)