HP Connect-It

ソフトウェアバージョン：3.90

動作の改善

ドキュメントリリース日：15 May 2008
ソフトウェアリリース日：May 2008

法的制限事項

限定保証条項

機密コンピュータソフトウェア。

所有、使用、または複製するには、HP からの有効なライセンスが必要です。

FAR 12.211および12.212準拠。商用コンピュータソフトウェア、コンピュータソフトウェアマニュアル、技術データは、ベンダの標準商用ライセンスに基づき、米国政府にライセンス供与されています。

保証

HP製品およびサービスに対する保証は、当該製品またはサービスに付帯する明示的保証条項でのみ規定されます。

本規定のいかなる部分も、他の保証を構成すると解釈されるものではありません。

HPは本書の技術上または編集上の誤謬、欠落についての責任は負わないものとします。

本書に含まれる内容は、予告なく変更される場合があります。

ブランド

- Adobe®, Adobe logo®, Acrobat® and Acrobat Logo® are trademarks of Adobe Systems Incorporated.
- Corel® and Corel logo® are trademarks or registered trademarks of Corel Corporation or Corel Corporation Limited.
- Java™ is a US trademark of Sun Microsystems, Inc.
- Microsoft®, Windows®, Windows NT®, Windows® XP, Windows Mobile® and Windows Vista® are U.S. registered trademarks of Microsoft Corporation.
- Oracle® is a registered trademark of Oracle Corporation and/or its affiliates.
- UNIX® is a registered trademark of The Open Group.

目次

はじめに

このガイドの対象読者

このガイドはConnect-Itを使用して統合を行うユーザすべてを対象に説明されています。

本書の使用方法

「Connect-It－プレゼンテーション」の章

この章では、Connect-Itの動作の概要が説明されています。

「最適化－ソースコネクタ」の章

この章では、シナリオの動作に影響を与えるConnect-Itパラメータについて説明しています。

「最適化－ターゲットコネクタ」の章

この章では、シナリオの動作に影響を与えるConnect-Itパラメータについて説明しています。

「シナリオの操作を改善する方法－例」の章

この章では、処理動作を改善するためにシナリオに変更を加える方法を説明します。

「他の考慮すべき点」の章

この章では、処理動作に影響を与える可能性のある外部要素について記載します。

本ガイドで使用する表記法

本ガイドで使用する表記法の一覧を以下に示します。

表記法	説明
JavaScriptコード	コードまたはコマンドの例。
Fixed width characters	DOSコマンド、関数のパラメータ、またはデータフォーマット
...	省略したコードまたはコマンド部分。
注意: 追加情報...	補足情報
重要項目: 以下の注意事項は...	ユーザにとって重要な情報
ヒント: 使用上のヒント	ヒント
警告: 警告	ユーザにとって非常に重要な情報
オブジェクト	Connect-It インタフェースオブジェクト：メニュー、メニューエントリ、タブやボタンなど。

以下の表記法も併せて使用されます。

- 番号が振られたリストにある順番に従って実行する手順です。以下に一例を挙げます。
  1. 手順1
  2. 手順2
  3. 手順3と最終手順
- すべての図と表には、その章と、章に現れる順番で番号付けられます。例えば、第2章の4番目の表には**表2-4**となります。

1 Connect-It プレゼンテーション

データ処理

Connect-Itは、コネクタ（コンポーネント）を使用して外部アプリケーションと相互に作用します。あるデータベースから別のデータベースへデータを転送する場合、

- ソースコネクタは、ドキュメントを生成します。
  各ドキュメントは、ソースアプリケーション内の1データ集合に一致します。
- マッピングボックスが、ソースコネクタにより生成されたドキュメントの構造を再構築し、ターゲットコネクタがドキュメントを取り込めるようにします
- ターゲットコネクタがドキュメントを取り込みます。
  各ドキュメントは、ターゲットアプリケーション内の1データ集合に一致します。

データベース型のコネクタでは、各ドキュメントは1データベーステーブルのレコード（または別のテーブルへのリンク）に対応しています。

他のコネクタの場合、データ集合は、Eメールメッセージ、区切り文字で区切られたテキストおよびセキュリティ情報などに当たります。

コネクタの選択、コネクタ間の関係、そしてソースドキュメントとターゲットドキュメント間のマッピングの定義が、統合シナリオを構成します。テストとデバッグを経た後、シナリオはスケジューラに関連付けられ生成段階に入ります。

どの統合過程でも起こるように、ドキュメントの一部または全体が拒否されることもあります。しかし処理レポートやドキュメントログを使用すると、統合シナ

リオ全体を定義し直すことなく、拒否されたドキュメントを再処理することができます。

特殊な用語解説

インジェクタ

インジェクタは、データをコネクタに挿入する要素のことをいいます。（例：サーバ、コネクタ）

2 最適化－ソースコネクタ

この章では、シナリオの動作に影響を与えるConnect-Itパラメータについて説明しています。

正しい要素の取得

データフォーマット

データフォーマットは、ソースデータベースとターゲットデータベースで一致している必要があります。

データフォーマットがソースとターゲットで異なる場合、マッピング上か、ターゲットデータベースに挿入する前にデータの処理が必要です。処理は、動作に直接的にまたはネガティブに影響を与えます。データ処理は、Connect-Itの処理タスクとして行うよりもデータベースのレベルで行う方が有効です。

処理を行わないと、レコードが重複するか余分なレコードがデータベースに挿入されます。

- データフォーマットに関する共通的な例は、テキスト型フィールドです。

  例えば、ソースデータベースでは名字の書式が"Abcde"と設定され、ターゲットデータベースでは"ABCDE"と設定されます。この場合、ソースアプリケーション内の処理ルールを適用し、テキスト型要素をターゲットデータベース内のフォーマットに合うように書式を設定することができます。これを行うと、マッピングまたはConnect-It内のデータ照合更新時に追加クエリを実行する必要がなくなります。

- また、文字列が適切に処理され、不要な空白スペースが追加されていないことを確認する必要があります。空白が追加されると、データ照合更新中にレコードが重複し、要求フィールドに対して照合更新が実行されない場合があります。

関連項目にフィルタを実行するためのWhere句の使用

本節では、データベース型コネクタのみを扱います。

Where句を適用すると、ソースコネクタによって生成されるドキュメントの数が減り、関連項目のみを分析することによって処理時間が改善されます。

生成対象のドキュメントタイプにフィルタを適用（WHERE句を使用）する場合、ユーザがフィルタの対象にするフィールドをインデックス化する必要があります。これを行わないと、クエリの効率が低下することがあります。

▶ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、スケジュールの作成、シナリオの性能を最適化する

WHERE句の使用の最適化

データベース型コネクタの生成用ルールでは、WHERE句を使用できます。

クエリは、インデックス化されたフィールドに実行する必要があります。そうしないと、クエリがテーブル全体に実行されます。

▶『Connect-It コネクタガイド』「コネクタのルール（ディレクティブ）」「生成用ルール」「WHERE句とORDER BY句」

ポインタ

スケジューリングポインタは、次のように機能します。Connect-Itがポインタのステータスを保存し、処理済みと未処理の間に境界を設定します（通常は、日付-時刻フィールドにドキュメントまたはレコードが最後に処理された日付が含まれます）。

ポインタは、シナリオ内で処理するデータの数を減らすために、インデックス化が可能です。

多くの場合、ポインタのステータスは特定の日付-時刻フィールドに関係があります。ポインタによって示される情報は、指定のユーザコンテキストに従ってサードパーティアプリケーションのデータの処理を管理するために使用されます。

例：ソースコネクタは、各従業員の雇用日付をポインタとして使用することによって、新規に雇用された従業員に相当する従業員テーブルのレコードのみを処理します。

▶ Connect-It ユーザガイド - 統合シナリオの管理、スケジュールの作成

使用されるポインタ型は、サードパーティのアプリケーションと生成用ドキュメントタイプによって異なります。

ポインタのインデックス化

ポインタとして使用するフィールドをインデックス化しないと、関連ドキュメントを取得するためにテーブル全体がデータベースエンジンによって検索されます。そのため、ドキュメントの生成に必要な時間が増えます。

例えば、インデックス化されていない1千万個のドキュメントを含むテーブルから10個のドキュメントを取得するのは、1万レコードごとにインデックス化されたテーブルから10個のドキュメントを取得するよりも困難です。

ドキュメントの生成に必要な時間を削減するために、ポインタとして使用されるフィールドをインデックス化する必要があります。

例えば、シナリオのスケジューリングがスケジューリングポインタに基づく場合、シナリオが最後に実行された時間から変更されたレコードのみが生成用に選択されます。

シナリオに関係する列のみを選択する

コネクタの生成用ドキュメントタイプの場合、取得されるデータに関係するフィールドのみを選択します。例えば、部署と従業員のテーブルへのクエリの場合、従業員の識別子のみを取得し、その他の列からデータを取得しません。

ドキュメントの生成の改善

コネクタのドキュメント生成を改善するには、ソースアプリケーションからできるだけ高速でデータを取得する必要があります。

本節では、ドキュメントの生成にかかる時間を削減するさまざまな方法について説明します。

ソースデータのサーバの最適化

ドキュメントの生成は、ソースデータのサーバの使用と同様にその接続によって大きく左右されます。

Connect-Itは、生成モード時に非常に多数の変換を実行することも大量のデータを処理することもありません。

結果を改善するには、以下を検証します。

- ソースデータサーバの負荷

  プロセッサ、利用可能メモリなど、サーバの技術的な性能に応じて、負荷が決まります。
- ネットワーク接続（WAN、LAN）

ネットワークでの待ち時間

アプリケーションは、1つのサーバに配置するのが理想的ですが、そうでない場合、Connect-Itが最も多く相互に作用するサーバの近くにConnect-Itをインストールすることを推奨します。

コネクタが実行するクエリの検証

Connect-Itは、生成モード時に非常に多数の変換を実行することも大量のデータを処理することもありません。

ソースデータベースに転送されるクエリをトラッキングするには、**Edit/ Options** メニューでConnector/ Show tracking linesオプションを選択します。

このオプションを設定した後に、ドキュメントログにシナリオのソースコネクタが実行するクエリに対応するメッセージが表示されます。

例：

```
SELECT AcctCode,AssetTag,BarCode,dDispos,DisposProfit,dtListPriceCv,Field2,FullName FROM amAsset
```

この情報を使用すると、以下の作業を実行できます。

- 定義されたドキュメントタイプと設定済みの生成用ルールに従って要求が生成されたことを検証します。
- クエリのシンタックスが適切であり要求されたオブジェクトに関係することを確認します。
- サードパーティのツールでクエリを再起動して、実行にかかる時間を確認します。

▶ Connect-It ユーザガイド - 統合シナリオの管理、ドキュメント生成の改善

データセット取得用のインデックス化されたクエリの実行

データセットを取得する多くの場合、クエリ（WHERE句）内で使用される照合更新キーが1つ以上のインデックス化されたフィールドに該当すると、処理性能が向上します。

注意:

インデックスの使用と処理性能における結果の向上は、データベースエンジンと使用しているコネクタによって左右されます。

通常、インデックスの使用はすべてのソースレコードの5%未満を取得する場合に有益です。反対に、大部分のデータを処理する必要がある場合、インデックスの使用は有益ではありません。

生成するドキュメントの数の計算

このオプションを使用すると、ソースコネクタで処理するドキュメントの数を集計できます。

この値を使用すると、進行状況バーをScenario Builder内に表示できます。このバーは、ソースコネクタが処理するドキュメントの数を示します。

Asset Managerまたはデータベースコネクタの場合、このオプションはSELECT COUNT型のクエリをトリガします。

```
SELECT COUNT(AcctCode) FROM amAsset
```

例：

警告

通常、このオプションは性能に著しく悪い影響を与えます。プロダクションモードでは、このオプションを無効にすることを推奨します。

ソースコネクタで生成する必要があるドキュメントの数の計算を無効にするには：

1. **Edit/ Options**を選択します。
2. **Connector**ノードを開きます。
3. **Calculate number of documents to be processed**オプションを*No*に設定します。
4. **OK**をクリックします。

 注意:

場合によって（外部結合など）、返される結果が壊れます。

コネクタのキャッシュオプションの使用

データベース型のコネクタの場合、またはデータフォーマットが安定しているコネクタの場合、生成用ドキュメントタイプ（メタデータ）の構造にキャッシュを使用することを推奨します。

キャッシュを使用すると、生成用ドキュメントタイプ（メタデータ）の記述の読み込みがローカルで実行されるため、コネクタのオープンが速くなります。これは、テーブルまたはオブジェクトが多い場合に顕著になります。

 注意:

テストまたは開発フェーズで、交換するデータのフォーマットが変わることがあります。この場合、キャッシュオプションを無効にすることを推奨します。

▶ Connect-It コネクタガイド - 「コネクタの設定」「Configure the cache」

キャッシュを使用しない場合、コネクタは、オープンするたびにソースデータベースの記述に対応するデータセットを取得する必要があります。

データベース型コネクタでは、この記述データは以下の通りです。

- テーブルのリスト
- フィールドとフィールドタイプのリスト
- 書式のリスト
- インデックスのリスト
- リンクのリスト
- 結合のリスト
- その他

コネクタを開くために必要な時間は、データ量の増加に伴って多くなります。

注意:

コネクタを開くのにかかる時間は、使用可能な帯域幅が限られている場合も長くなります。これは、ネットワーク接続（WAN、LAN）の性能に関係します。

警告

シナリオを生成する場合には、キャッシュの使用を推奨します。

しかし、ソースデータベースの記述データが変わった場合は、コネクタの変更を同期化する必要があります。

コネクタのキャッシュの同期をとるには：

1. Scenario Builderを起動します。
2. シナリオを開きます。
3. キャッシュの同期をとる必要があるコネクタを選択します。
4. コネクタを開きます（F4）。
5. **Tools/Cache/Synchronize the cache**メニューを選択します。

キャッシュの同期をとった後、新しいソースデータベース記述に存在しない項目がマッピングに含まれていないことを確認します。

自動再接続

データベース型コネクタ（ServiceCenter、Asset Manager、データベースなど）の場合、Connect-Itの内部ポリシーは、処理性能の向上を図るためにサーバとの接続を開いておくことです。これらのコネクタは、接続が失われた場合と事前に設定された時間差の後にサーバに自動的に再接続するために再構成できます。

ネットワークの安定が維持されない場合、再接続に著しく時間がかかることがあります。ネットワークのファイアウォールと計画または計画外の停止について、ネットワークに適したポリシーを確認してください。

データソースがリモートサーバであるコネクタの使用を計画する場合、自動サーバ接続オプションを選択する必要があります。

自動再接続のオプションは、データパケットの伝送が増えるため、ネットワークでの待ち時間に多大な影響を与えます。

▶『Connect-It コネクタガイド』「コネクタの設定」「再接続のパラメータを設定する」

再接続に関する情報は、Connect-Itログ内に表示されます。

Asset Managerコネクタの再接続の試行例：

```
Opening session for connector 'Asset Manager (TPRIM)'
Connecting to the server...
'Asset Management' API error: 'Not connected to the Oracle server
Unable to connect to database 'STRONTIUM' (UserId=TPRIM) ; Oracle code=
12154
Unable to connect to this database engine.'
The next attempt to connect to the server will be performed in 1 s
Attempting to reconnect ...
'Asset Management' API error: 'Not connected to the Oracle server
Unable to connect to database 'STRONTIUM' (UserId=TPRIM) ; Oracle code=
12154
Unable to connect to this database engine.'
The next attempt to connect to the server will be performed in 2 s
[...]
```

セッション毎の再接続（Asset Managerコネクタ用）

Asset Managerコネクタ専用の設定オプションを使用すると、シナリオの各セッションの終了時に接続を切断し、新規セッションの開始時のみに接続を再開するように設定できます。

これにより、Connect-Itサーバが使用するリソースが少なくなり、サーバへの再接続が迅速になります。

▶『Connect-It コネクタガイド』「Hewlett-Packardコネクタ」「Asset Managerコネクタ」「Asset Managerコネクタの設定」

警告

シナリオの実行頻度が高い場合（5分おきなど）は、このオプションを無効にする必要があります。

▶Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、スケジュールの作成

3 最適化 - ターゲットコネクタ

この章では、シナリオの動作に影響を与えるConnect-Itパラメータについて説明しています。

ターゲットシステムのカスタマイズ

Connect-Itの動作の最適化に加えて、ターゲットシステムをカスタマイズすることもできます。以下の点に注意してください。

- データベースにインデックスを作成します。
- ディスクをリモートでアクセスする場合は特に、ファイルをディスク間で転送しないでください（データは同一ディスク上で使用してください）。

初期読み込みオプション

このオプションは、データベース型コネクタに使用します。

このオプションを使用する前に、以下の前提条件が満たされていることを確認してください。

- 空のデータベースで開始する
- インポート対象のドキュメントが一意である

初期読み込みオプションを選択すると、クエリを挿入（INSERT）のみに限定することによって、クエリの数を減らすことができます。照合更新選択クエリ（ドキュメントの有無を確認する事前チェック）が実行されません。

コネクタのキャッシュオプションの使用

このオプションの動作については、「最適化－ソースコネクタ [献 9]」の章に記載されています。

▶ コネクタのキャッシュオプションの使用 [献 13].

自動再接続

このオプションの動作については、「最適化－ソースコネクタ [献 9]」の章に記載されています。

▶ 自動再接続 [献 14].

照合更新キー

照合更新は、コネクタが作成または更新するテーブル内のレコードを一意に識別できるフィールドの定義で構成されます。

データ照合更新に使用するフィールドは、照合更新キーという名前のキーを付けることによって識別します。複雑マッピング要素は、場合によって複数の更新キーがあります。各キーはキーセットに属します。

照合更新キーとして使用するフィールドは、ターゲットコネクタによるドキュメントの取り込みに大きく影響します。

▶『Connect-It コネクタガイド』「コネクタのルール（ディレクティブ）」「照合更新」

シナリオの実行時に、以下の操作を実行します。

1 ソースコネクタの生成用ドキュメントタイプを作成します。
2 マッピングからドキュメントタイプを作成します。
3 ターゲットコネクタに対して、照合更新が行われます。マッピングのドキュメントの要素をデータベース内の任意のレコードに存在するフィールドと比較します。
4 照合更新スクリプトをターゲットコネクタに適用します。

▶ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、ドキュメントタイプのマッピングの定義

照合更新キーの機能

照合更新は、コネクタが作成または更新するテーブル内のレコードを一意に識別できるフィールドの定義で構成されます。

ドキュメントを取り込むには、以下に示す2つの操作が必要です。

1 ターゲットアプリケーション内のレコードの有無を知るために、照合更新キーとして選択されたフィールドを使用するクエリを送信します。

2 レコードを挿入または更新します。照合更新キーとして選択されたフィールドをインデックス化しない場合は、データベースの検索とドキュメントの取り込みかかる時間が長くなります。

以下の場合、性能の処理に時間がかかります。

- インデックス化されたクエリをレコード取得用に使用する場合
- キー識別を最適化するために一意の制約を使用する場合（キーセットの重み）

  例えば、部署と従業員のテーブルからレコードを取得するには、従業員の名字、名前、および電話番号ではなく従業員のIDを使用できます。

▶ Connect-It ガイド - HP Connect-It データベース統合ソリューション、マッピング - 例

ドキュメントの生成の改善

コネクタのドキュメント取り込みを改善するには、ターゲットアプリケーションに送信するデータをできるだけ高速で処理する必要があります。

本節では、ドキュメントの生成にかかる時間を削減するさまざまな方法について説明します。

トランザクション管理

デフォルトで、データベース型タイプのコネクタは、ドキュメントを取り込むたびに'commit'を実行します。

ドキュメントのグループごとに'commit'を実行できます。

コミットするために取り込むドキュメントの数を設定するには：

1 Scenario Builderを起動します。

2 シナリオを開きます。

3 トランザクションパラメータを変更するコネクタを選択します。

4 コネクタ設定ウィザードを起動します（F2）。

  警告：コネクタの高度な設定が有効になっている（ツールバーで アイコンが選択されている）ことを確認します。

5 **Next**を数回押して、**Manage transactions**ページを開きます。

6 **Commit by group of documents**オプションを選択します。

7 'commit'をトリガする前に取り込む必要があるドキュメントの数を指定します。

多数のドキュメントを取り込む場合、200以上のドキュメントに対して'commit by group'機能を起動することを推奨します。

例えば、任意のシナリオの場合、処理性能が以下のようになります。

1 ドキュメントごとのコミット（デフォルトのオプション）

性能：6000ドキュメント用に4分（25ドキュメント／秒）

2 500ドキュメントのグループごとのコミット

性能：6000ドキュメント用に3分（33ドキュメント／秒）

コミット

調査に基づき、さらに各シナリオに従って、バッチ（20～100レコード）でコミットを設定すると、直接コミットを実行することと比較して30%の改善が実現することが分かりました。性能の改善は、トランザクションのオブジェクトとトランザクションロックに直接関連付けられています。

この説明は、固有インジェクタのみに当てはまります。同じデータにアクセスする複数のインジェクタを使用すると、サーバロックの数が増加し、コミットの遅延が大きくなる可能性があります。これにより、データのアクセス前にロックが開放されるまで他のインジェクタが待機する必要があるため、処理性能に悪い影響を及ぼします。

また、これにより、永続的なデッドロックが残る状態が発生する可能性があります。この場合、DBMSがいずれかのアプリケーションを選択およびキャンセルします。

バッチ処理

バッチ処理を選択するときにドキュメントでエラーが発生すると、ドキュメントのセット全体が1つずつ再試行されます。この場合、バッチに多数のドキュメントが含まれると、ドキュメント単位の処理よりもバッチ処理に時間がかかることがあります。

このエラーを防ぐには、バッチの単位を10または20ドキュメントにすることを推奨します。

並列化

シナリオ内への並列化の組み込みは、以下の質問に回答を提示するために行われます。

- 複数のプロセッサを最大限に活用するには？

  目的：処理時間を増やすために、複数のコネクタインスタンスを実行します。

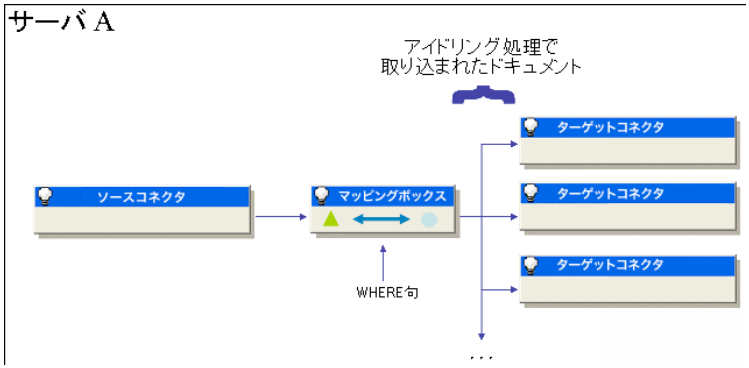
シナリオに並列化を使用するとは、以下の作業を行うことです。

- シナリオを実行するためにプロセッサの数を割り当てる

複数のプロセスをインシデントなしで同時に実行できることを検証するために、テストを実行できます。

- Connect-It ユーザガイド - Connect-It エンジンのドキュメント処理時間の改善

並列化 - 機能概要



並列化を使用すると、複数のプロセスを同時に実行できます。親プロセスは、子プロセスを制御して、可能であればそれらをトリガします。

例：3つのプロセスが並行して実行中です。プロセスは、1つの親プロセスと2つの子プロセスです。親プロセスはプロセッサの50%を使用します。親プロセスは、いずれかの子プロセスをトリガし、その子プロセスは残りのプロセッサ能力（この例では50%）を使用して実行されます。その結果、待機している3番目のプロセスが起動されると、使用可能なプロセッサ時間が再割り当てされ、各コネクタの性能が3分の1に減らされます。この例で、3番目のプロセスの動作が処理向上の妨げになるため、3番目のプロセスを起動する必要はありません。

注意:

並列時の処理性能は、ドキュメントの取り込みおよび生成方法に直接関係します。

専用マシンの場合、理想的なプロセッサ能力の消費最適化は90～95%です。

また、仮想メモリの割り当ては、ディスクアクセス時間が増加する原因になるため、できるだけ回避する必要があります。

並列シナリオの起動

1つのシナリオで多数のデータを移行しなくてはならない場合、シナリオを複数のシナリオに分割して、それぞれを別々のサービス（Windows）またはデーモン

（UNIX）で起動することが必要になる場合があります。また、サービスを別々のサーバで起動することも役に立ちます。

▶ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、Connect-It サービスの定義

例

データベースに記録された従業員のリストを別のデータベースにインポートするシナリオが必要です。

このデータの移行にかかる時間を削減するために、以下に示す2つのシナリオを作成できます。

- 1番目のシナリオは、名前がA～Jの文字で始まる従業員のリストを移行します。
- 2番目のシナリオは、名前がK～Zの文字で始まる従業員のリストを移行します。

従業員は、2つのシナリオのソースコネクタに生成用ルール（WHERE句）を入力するときに選択します。

▶『Connect-It コネクタガイド』「コネクタのルール（ディレクティブ）」「生成用ルール」

ロック

本節では、データベース型コネクタを説明します。

目的は、サーバロックと処理に要する時間の適正なバランスを見つけることです。

データの並列による処理の主な不利益は、レコード処理時にサーバによってレコードがロックされる場合に生じます。ロックは処理性能の低下につながります。

各Connect-Itプロセスの役割はドキュメントタイプを取り込むことであるため、場合によっては、任意のシナリオの同一テーブルに複数の同時アクセスが行われます。

レコードは、作成時または更新時にサーバによってロックされます。別のプロセスがレコードへのアクセスを試行すると、そのレコードにロックが存在することがプロセスに通知されます。これにより、レコードへのアクセスを取得するためにプロセスが試行を繰り返すので、性能が低下します。

これにより、シナリオ実行が遅くなり、取り込みが低下します。

ロック解除の待機

レコードがサーバによってロックされると、そのレコードへのアクセスを試行しているプロセスが待機するよう指示されます。プロセスは、待機する状態に移行してから、一定間隔でレコードへのアクセス取得を試みます。

この動作モードには、レコードのロックが解除された後であってもプロセスが待機モードから移らないという欠点があります。この例で、親プロセス（または別のプロセス）は同じレコードにアクセスして再度ロックします。続いて、他のプ

ロセスがレコードへのアクセスを試行し、待機するよう再び指示されます。プロセスが待機しなければならない時間が長過ぎる場合は、エラーを返します。

デッドロックの回避

レコードロックを回避するには、いくつかの方法があります。

- レコードは、プロセスが書き込みモードの場合のみロックできます。任意のシナリオで、データを挿入および更新するプロセスと、データを読み込むことのみが必要なプロセスを区別できます。
- 書き込みモードではなく読み込みモードでプロセスがレコードを検索するように、シナリオを構成します。

  例えば、関連レコードを作成した後でプロセスがそれらのレコードを検索します。HP Asset Managerアプリケーションの場合、モデルテーブルは、それを参照するデータセットの前に挿入されます。

▶ Connect-It ガイド - Asset Managerデータベース統合ソリューション、Asset Managerデータベースのマッピング、デッドロック

4 最適化 - エンジン

この章では、シナリオの動作に影響を与えるConnect-Itパラメータについて説明しています。

Connect-Itの使用

Connect-Itエンジンのドキュメント処理時間の改善

本節では、Connect-Itエンジンによるドキュメント処理時間の削減のために行うことを説明します。

非グラフィカルインタフェースの使用

シナリオの実行中にScenario Builderを表示する場合、コマンド（更新、ドキュメントログの閲覧、新規オプションの入力など）を選択するたびに、Connect-Itエンジンの反応速度が遅くなります。

スケジュールの設定

Connect-Itのスケジューリングは、シナリオの生成用ドキュメントタイプに1つまたは複数のスケジューラを関連付けるために使用します。

例：スケジューラがAsset Managerコネクタを1時間ごとに起動します。起動するたびに、コネクタはAsset Managerアプリケーションのレコードに対応するドキュメントを生成します。

コネクタが起動するたびに、Connect-Itは処理するデータの量を計算し、データ全体が処理されるまで停止しません。処理するデータ量に応じて、シナリオのスケジューラを調整する必要があります。大規模なデータベースのマイグレーションの場合は、稼動率が低い時期（夜間のスケジュールなど）にソースコネクタの起動をスケジュールすることが推奨されています。少量のデータを処理する統合シナリオの場合は、既製の**Synchronous**スケジューラの使用が推奨されています(ソースコネクタが毎秒起動）。

▸ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、スケジュールの作成

シナリオの作成

.cfgファイルのカスタマイズ

.cfgファイルは、設定ファイルエディタを使用してカスタマイズできます。

.cfgファイルをカスタマイズすることによって、生成用ドキュメントタイプの平面図の上にビューを作成できます（テーブルとフィールド）。

1つのドキュメントで複数のテーブルに関連する情報を表示し、新しい結合またはサブクエリを作成することもできます。この方法の欠点は、一意で総合的なドキュメントを作成するために、多数のクエリの作成と負荷のかかるデータベース操作が必要になり、性能に悪い影響を与える点です。例えば、データベースコネクタの読み込み能力が毎秒2,000ドキュメントであり、IntelLanDescコネクタ（メタデータ記述が追加されたデータベースコネクタを使用して開発済み）の読み込み能力が毎秒7ドキュメントです。

▸ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、設定ファイルの編集

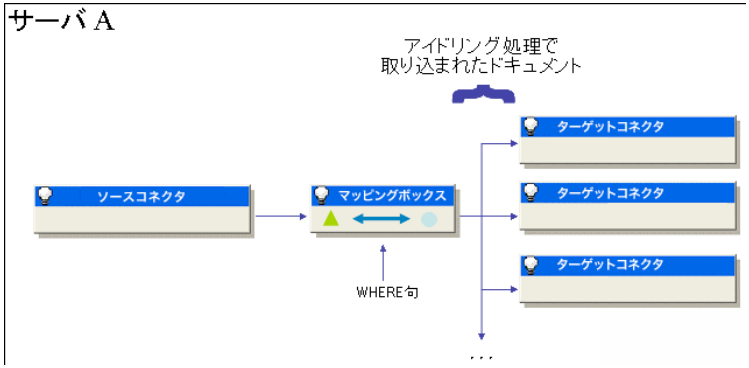
インジェクタ

インジェクタは、データを挿入するオブジェクト（シナリオ、コネクタ）です。

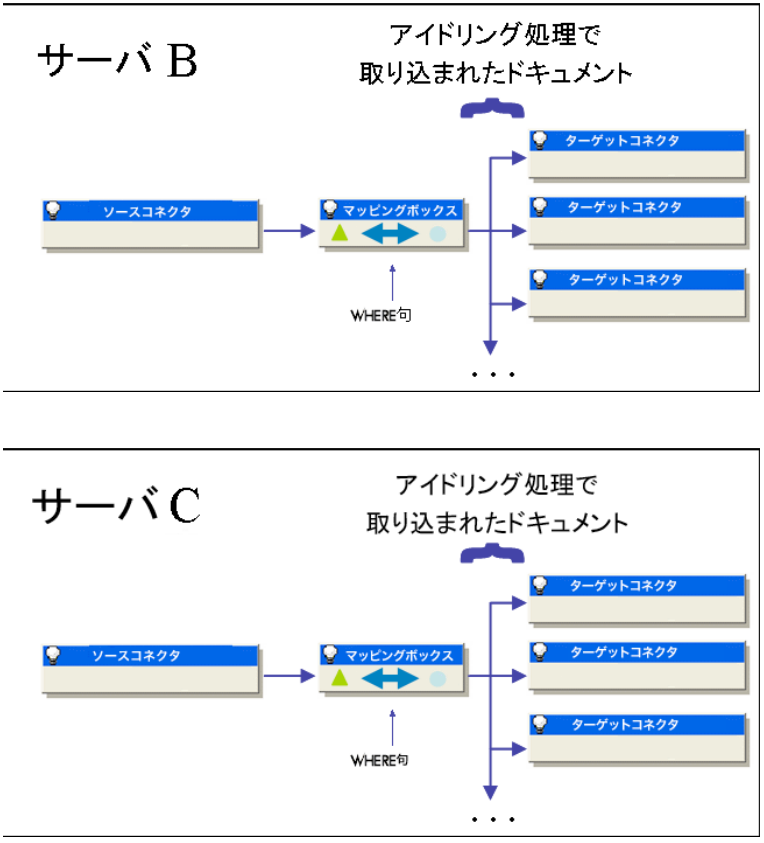
シナリオの処理性能を上げるために、以下のオプションを使用できます。

- 同一シナリオの複数インスタンスを起動する
- 各インジェクタを並列化モードで使用できる

例：3つのインジェクタが並列で3つのプロセスを実行するので、同時に9個のデータソースを挿入しています。



3つのインジェクタを実行するシナリオでは、同時にデータを処理しないでください。WHERE句に基づくフィルタを各インジェクタに使用し、データが1回だけ読み込まれて処理されることを確認する必要があります。



レコードキャッシュ

キャッシュを使用すると、以前のクエリで取得された情報を再利用できます。レコードIDを要求するクエリが送信済みである場合、同じ情報を要求する後続のクエリはデータソースに送信されません。レコードのIDは、Connect-Itキャッシュデータベースから取得されます。この動作は、照合更新キャッシュメニュー内のOptions/ Connector/ Maximum number of documentsによって設定できます。

シナリオを次回使用するときに、Connect-Itは、データベースでオブジェクトの有無を確認するのではなく、キャッシュ内でその識別子を代わりに使用します。これによって、クエリをデータベースに送信する必要がなくなるので、時間を削

減できます。このオプションは、部署と従業員、製品、カタログ参照など頻繁に使用されるテーブルを分析する一般的な例に役立ちます。

キャッシュのデフォルトサイズは、ドキュメントタイプ内のノードに関連付けられたテーブルに対して1,000レコードです。

キャッシュは、基本テスト時にコネクタへのSQL Select型クエリについて30%の改善をもたらします。つまり、データベースサーバとのトランザクションが30%削減されます。

更新照合キャッシュに保存されるドキュメントの最大数のオプションを設定すると、処理性能に影響します。

キャッシュの設定にはメリットがありますが、特定の状況下でデメリットになる場合があります。

- メリット：コネクタとターゲットデータベース間のラウンドトリップの数を制限します。
- デメリット：キャッシュが大き過ぎる場合に、サーバ上で使用可能なリソースが制限されます。

注意:

レコードキャッシュとコネクタキャッシュ（コネクタを設定ウィザードで構成するときに定義済み）を混同しないでください。コネクタキャッシュは、任意の時点でコネクタが検索するデータベース構造のスナップショットに相当します。

- Connect-It ユーザガイド - キャッシュファイルの使用
- 『Connect-It コネクタガイド』「Configure the cache (advanced mode)」

スクリプト

- スクリプト内の変数宣言の数を制限します。
  例えば、以下のスクリプト

```
dim test1,test2,test3 as string
test1="abcd"
test2="efg"
test3=test1+test2
```

  は、このスクリプトよりも与える影響が大きくなります。

```
dim test as string
test="abcdefg"
```

  また、他のグローバル変数を含む特定ファイル内で変数を宣言すると良くなります。
- プログラミングのループからデータをできるだけ取り除きます。

ログ

ドキュメントログを使用すると、ソースコネクタの生成用ドキュメントを処理している間に問題が発生したかどうかを確認できます。

以下の対策により、処理性能が向上します。

- 処理上の問題を生じるドキュメントのみを記録するようにドキュメントログを設定することを推奨します。
- シナリオを管理しているサーバ上で使用可能なリソースに応じて、メモリに読み込む行数を定義する必要があります。

  大量のドキュメントをメモリに保存すると、使用可能なすべての物理メモリがいっぱいになります。その結果、サーバが仮想メモリを作成する必要が生じ、処理時間が増加します。

▸ Connect-It ユーザガイド - トラッキング項目の説明メッセージ

▸ Connect-It ユーザガイド - 管理モニタの定義

設定

ドキュメントログの設定

Connect-Itのドキュメントログを設定すると、以下の項目を表示できます。

- ドキュメントログ内に取得するエラータイプ（**Filter**フィールド）
- Connect-Itがメモリに保存するトラッキング項目の最大数
- ドキュメントログのメッセージを保存するテキストファイル

ドキュメントの処理にかかる時間を短縮するには：

1. Scenario Builderを起動します。
2. **Monitors/ Configure monitors...**メニューを選択し、Session backupsタブをクリックして、以下を無効にします。
   - Database
   - Files
3. **Save documents**オプションのサブオプションを選択することによって、ドキュメントログに保存するエラータイプを制限します。

   例：Reject
4. 保存するトラッキング項目の数を制限します（**Display sessions**タブ）。

モニタの設定

モニタを使用すると、シナリオに関連するモニタ機能を定義できます。

性能は、適用するフィルタによって異なります。特に、以下の項目について注意する必要があります。

- 割り当てられたメモリの最大量

- ドキュメントログに保存するトラッキング項目の数
- ドキュメントで使用するフィルタ（reject、field rejectなど）

▸ Connect-It ユーザガイド - 統合シナリオのインプリメンテーション、モニタの定義

実行結果

実行結果の使用によるドキュメント処理時間の評価

Connect-Itでは、シナリオを構成する各種要素を基準として実行結果が管理されます。

これらの実行結果は、データの処理に必要な時間を表示して、ボトルネック（特に、無関係のドキュメントの生成および取り込み時に発生するボトルネック）を特定するために使用されます。

Connect-Itログは、Scenario Builderのグラフィカルインタフェースまたはログファイルで参照可能であり、以下に関する実行結果を示しています。

- ソースコネクタがドキュメントの生成に費やした時間
- マッピングボックスがドキュメントの変換に費やした時間
- ターゲットコネクタがドキュメントの取り込みに費やした時間
- 拒否されたドキュメントの数

  例：Asset Managerコネクタの実行結果は、製品のテーブルのレコードに対応するドキュメントが処理されなかったことを示します。

  ▸『Connect-It ユーザガイド』「統合シナリオのインプリメンテーション」「管理モニタの定義」

ツールのデータ処理の実行結果を取得する

1. シナリオを開きます。
2. モニタを設定します（**Monitors/ Configure monitors**メニュー）。
3. シナリオコネクタを開きます（**Ctrl + F4**）。
4. シナリオのソースコネクタを選択します。
5. ▸をクリックします（**F5**）。
6. ソースコネクタ、ターゲットコネクタ、またはマッピングボックス上にマウスのポインタを置きます。

   Statisticsセクションを含む状況依存ウィンドウが表示されます。

注意:

このデータは、scenario diagramの下にあるConnect-It logタブ内にも表示されます。

実行結果の分析

Connect-Itログ内に表示される以下のデータは、Action Request Systemソースコネクタ、マッピングボックス、LDAPターゲットコネクタを含むシナリオの実行結果です。

```
Statistics for the 'Action Request System (fdcitsrv01)' connector  (session: 26m
in  57.310s / API: 26min  04.550s)
Document(s) consumed: 7143
Document(s) rejected: 1
Records(s) inserted: 1199
Records(s) updated: 5943
Statistics for the 'LDAP (mail-sd.Hewlett-Packard.com)' connector (session: 4
7.628s / API: 35.765s)
Document(s) produced: 7143
Statistics for the 'Mapping (Basic engine)' connector (Session: 01.404s)
Script(s) analyzed: 14286
Document(s) consumed: 7143
Document(s) produced: 7143
```

作成された実行結果は以下のように分析されます。

- Action Request Systemコネクタによるドキュメントの処理 = 27分、または1秒につき*4.4*ドキュメント。

  処理時間のうち、Action Request Systemコネクタに関連するAPIにかかった時間が26分で、Connect-Itでの処理にかかった時間は1分です。

  APIは、ネットワークの応答時間、'commits'の実行、データベース整合性ルールの遵守などです。
- LDAPコネクタによるドキュメント処理の処理 = 48秒、または1秒につき*150*ドキュメント。

  処理時間のうち、LDAP APIでかかった時間が36秒で、Connect-Itでの処理にかかった時間は12秒です。
- マッピングボックスによるドキュメントの処理：2秒、または1秒につき*5100*ドキュメント。

  この時間は、Connect-Itでのドキュメント処理の時間に相当します。

詳細な実行結果

詳細な実行結果は、データベース型コネクタに使用できます。これを使用すると、以下の作業が可能です。

- Select、Insert、Update、およびDelete型クエリに使用する時間を表示します。
- 最小、平均、および最大時間に加えて、発行されたSelect、Insert、Update、およびDeleteクエリの合計数を表示します。

ドキュメント処理速度の例

ドキュメント処理の速度の例は以下の通りです。

- データベースコネクタ
  1500～2000ドキュメント／秒
- ServiceCenterコネクタ
  450ドキュメント／秒
- Asset Managerコネクタ
  400ドキュメント／秒

データベースコネクタ

データベース型コネクタをSybaseで使用するシナリオの性能を改善する

必要に応じてSybaseネイティブ接続を使用するデータベース型コネクタの場合、詳細オプションの PostConnectSql=set forceplan onオプションを入力する と、SQLクエリ実行の性能が向上する可能性があります。

Sybase 12および12.5の場合、一意のインデックスが使用され、一意でないインデックスは使用されません。

▶ 『Connect-It コネクタガイド』 「高度な設定」「詳細オプション」

データベースのカスタマイズ

処理速度を上げるためにいくつかの手法を採り入れることができます。

- インデックス
- 実行結果
- SQLクエリ構造体

インデックス

インデックスが各フィールドを別々に扱うのではなく、検索対象の複数のフィールドを扱うと有益です。この場合、インデックスは行ではなく列を扱います。

例えば、Oracleの場合、複数の列へのSelect型クエリは、以下の構造体です。

```
SELECT W1.lWorkItemId, W2.lWfInstanceId FROM amWfWorkItem W1, amWfInstance W2
WHERE W1.lDocRecordId = 28836763 AND W1.lActivId = 1417 AND W1.seStatus = 0 AND W2.seStatus = 0 AND W1.lWfInstanceId=W2.lWfInstanceId;
```

処理時間：00:00:02.02

すべてのドキュメントを処理するためにこのクエリを10,000回実行すると、合計処理時間が20,200秒または5.6時間になります。

```
Statistics
----------------------------------------------------------
12400  consistent gets
12389  physical reads
185  bytes sent via SQL*Net to client
234  bytes received via SQL*Net from client
```

以下のインデックスを追加することによって、以下のようになります。

```
create unique index workwfitem_20051124 on
amwfworkitem(ldocrecordid, lactivid, sestatus, lwfinstanceid, lworkitemid);
```

処理時間：00:00:00.07

```
Statistics
----------------------------------------------------------
3  consistent gets
2  physical reads
185  bytes sent via SQL*Net to client
234  bytes received via SQL*Net from client
```

DBMSの削減は、3,000に及びます。

合計で、複数列のインデックスを作成することにより、19,300秒または5.3時間の削減になりました。

実行結果

ほとんどの場合、DBMSによってテーブルの実行結果を更新する必要があります。

例えば、Oracleを使用し、SQLクエリオプティマイザモード（optimizer_mode）がALL_ROWS（またはFIRST_ROWS）である場合、テーブルごとに実行結果を更新する必要があります。

実行結果は、オプティマイザモードがRULEに設定された場合、必要ありません。

実行結果を算出するには、OracleでSQL PLUSを使用して以下を行います。

```
ANALYZE TABLE xxx COMPUTE STATISTICS;
```

SQLクエリ構造体

Oracle全般では、サブクエリを使用する場合、INではなくEXISTSまたはNOT EXIST命令を使用していることを確認します。

EXISTS命令の動作は、次のようになります。クエリから返る値がTRUEの場合、値TRUEが操作全体に設定されます。これにより、他に時間がかかるクエリを実行する必要がなくなります。

注意:

この動作は、他の最適化手法が有効なDB2およびMSSQL Serverには該当しません。

Asset Managerコネクタ

Asset Managerコネクタを使用するシナリオの性能を改善する

本節では、Asset Managerコネクタを使用するシナリオ内でドキュメントの処理にかかる時間を削減するさまざまな方法について説明します。

dtLastModifフィールドのインデックス化

スケジュールモードでAsset Managerコネクタを使用するシナリオでは、シナリオで扱うすべてのAsset Managerテーブル内の**dtLastModif**フィールドがまだない場合は、このフィールドにインデックスを追加する必要があります。また、このフィールドは、最後のセッションから作成または変更されたレコードを検証するために、Connect-Itによって体系的に使用されます。

資産のテーブル（amAsset）、製品のテーブル（amProduct）、または従業員と部署のテーブル（amEmplDept）でインデックスを作成するには、以下のコマンドを実行します。

```
CREATE INDEX Ast_dtLastModif ON amAsset (dtLastModif)
GO
CREATE INDEX Prod_dtLastModif ON amProduct (dtLastModif)
GO
CREATE INDEX EmplDept_dtLastModif ON amEmplDept (dtLastModif)
GO
```

dtLastModifフィールドがすべてインデックス化されていることを検証するには、実行可能ファイルadblogを使用します。このファイルを使用すると、WHERE句で**dtLastModif**フィールドにフィルタを適用するSQLクエリの実行を検証できます。

Asset Managerデータベース用の設定 - Sybase ASEエンジン

特定のクエリの実行に時間がかかることと、クエリのFROM部分に多数のテーブルがあることがLOGファイルに示される場合は、処理時間を短縮するために以下の行を追加することを推奨します。

```
PostConnectSql= set forceplan on
```

amdb.iniファイルの変更

SQLクエリの処理にかかる時間を短縮するには、Asset Managerアプリケーションのamdb.iniファイルに以下の行を追加することも必要です。

```
PostConnectSql= set forceplan on
```

以下の行は、*DB ASE COPPER*データベース用のamdb.iniファイルの設定です。

```
[DB ASE COPPER]
PostConnectSql=set forceplan on
stmtcache=500
LongDesc=
Engine=Sybase
Location=COPPER
EngineLogin=itam
EnginePassword=78C6143D43925F46F924205FBB42F0FED21594428DDCA
C641ED76CDAA17050EA1A124254200200
ReadOnly=0
CacheDir=
CacheSize=5120000
Base=EDS
Owner=
TableSpace=
TableSpaceIndex=
AmApiDll=aamapi35.dll
UseNTSecurity=0
```

これらのパラメータは、Asset Managementデータベースサーバのamdb.iniファイル内で設定する必要があります。

Asset Managerアプリケーションのクライアント部分をConnect-Itサーバにインストールする場合、同じSybaseデータベースに関連付けられた異なる2つの接続を確立できます。

- 1番目の接続は、オプションPostConnectSql=set forceplan onとstmtcache=500を使用します。

```
[DB ASE ConnectIt]
PostConnectSql=set forceplan on
stmtcache=500
LongDesc=
Engine=Sybase
Location=COPPER
EngineLogin=itam
EnginePassword=78C6143D43925F46F924205FBB42F0FED21594428DD
CAC641ED76CDAA17050EA1A124254200200
ReadOnly=0
CacheDir=
```

```
CacheSize=5120000
Base=EDS
Owner=
TableSpace=
TableSpaceIndex=
AmApiDll=aamapi35.dll
UseNTSecurity=0
```

- 2番目の接続は、これらのパラメータを使用しません。

```
[DB ASE COPPER ACGUI]
LongDesc=
Engine=Sybase
Location=COPPER
EngineLogin=itam
EnginePassword=78C6143D43925F46F924205FBB42F0FED21594428DD
CAC641ED76CDAA17050EA1A124254200200
ReadOnly=0
CacheDir=
CacheSize=5120000
Base=EDS
Owner=
TableSpace=
TableSpaceIndex=
AmApiDll=aamapi35.dll
UseNTSecurity=0
```

従来のAsset Managerクライアント（Sybase ASEクライアント以外）の場合、オプションstmtcache=500を使用しないでください。

従来のAsset Managerクライアントでの処理性能に問題がある場合は、以下のオプションのいずれかを試みることができます。

- PostConnectSql=set forceplan on
- PostConnectSql=set table count 3
- PostConnectSql=set table count 2

5　シナリオの操作を改善する方法－例

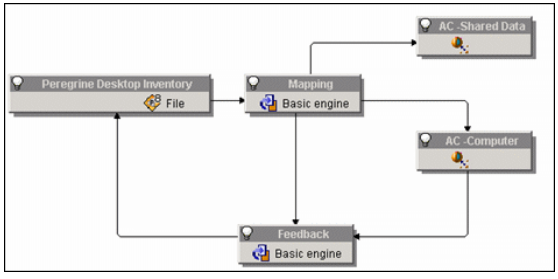
この章では、処理性能を上げるためにシナリオを変更する方法を説明します。

例

Connect-Itシナリオは、Asset Managerデータベースにインベントリデータを入力します。

シナリオで管理する項目を以下に示します。

- あるマッピング内のモデル（ハードウェアおよびソフトウェア）。1つのインジェクタが使用されます。
- 2番目のマッピング内のハードウェアおよびソフトウェアのインストール。サーバロックを削減するために、複数のインジェクタがインストールのモデルへの関連付けに使用されます。

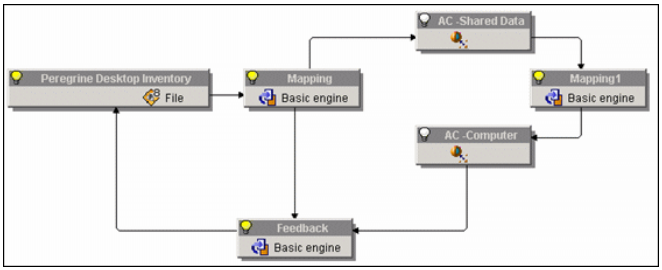


テスト結果：10,000ハードウェアレコードと23,783合計レコードが54分でインポートされました。

テスト結果（第2テスト）：10,000ハードウェアレコードと807,378合計レコードが8時間40分でインポートされました。

性能向上のために、以下の変更を行うことができます。

- ドキュメントをソースコネクタで生成するのは、最初に計画したように2回ではなく1回だけです。
- 最初にモデルを作成し、作成後にデータを挿入するという新しい動きを利用します。



テスト結果：10,000ハードウェアレコードと23,783合計レコードが47分でインポートされました。

改善：13%

テスト結果（第2テスト）：10,000ハードウェアレコードと807,378合計レコードが6時間でインポートされました。

31%の改善

結論は、以下のようになります。データを複数のマッピング内で数回処理する必要がある場合は、データを数回生成するのではなく1回だけ生成してからそれを改良することが望ましいです。

▸ Connect-It ガイド - Asset Managerデータベース統合ソリューション

インデックス

W