pitney bowes

Spectrum™ Technology Platform

バージョン 2019.1.0

管理ガイド

目次

8 - パフォーマンス

9 - 監視

10 - バックアップと復旧

11 - 設定

12 - 管理ユーティリティ

13 - クラスタ化

14 - Spectrum™ Technology Platform について

1 - はじめに

このセクションの構成

新規システムの設定

Spectrum™ Technology Platform を初めてインストールするときは、システムに基本レベルのセキュリティを適用すると同時に、Spectrum™ Technology Platform を通じて処理するデータにアクセスできるように、いくつかの操作を行う必要があります。

1. admin ユーザのパスワードを変更します。

   重要: システムに対して不正な管理者アクセスが行われることを回避するために、Spectrum™ Technology Platform をインストールした後すぐに admin パスワードを変更してください。

   a) Web ブラウザで次の URL を表示します。

      http://*サーバー*:*ポート*/managementconsole

      ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

   b) デフォルトの管理者資格情報でログインします。

      ユーザ名: admin
      パスワード: admin

   c) **[システム] > [セキュリティ]** を選択します。
   d) **"admin"** アカウントの横にあるボックスをチェックしてから、編集ボタン をクリックします。
   e) **[新しいパスワード]** フィールドに、新しいパスワードを入力します。**[パスワードの確認]** フィールドに、同じパスワードを再度入力します。
   f) **[保存]** をクリックします。

2. 必要に応じて、ユーザと役割を作成します。

   詳細については、「**Management Console を用いたユーザの追加**（21ページ）」を参照してください。

3. ユーザのアクセスを許可する Spectrum™ Technology Platform サーバー上のフォルダを指定します。

   詳細については、「**サーバーディレクトリアクセスの制限**（47ページ）」を参照してください。

4. Spectrum™ Technology Platform サーバーへの Web サービス要求に対して、ベーシック認証を許可するか、トークン認証を求めるかを決定します。トークン認証を求める場合は、ベー

シック認証を無効にします。詳細については、「**Web サービスのベーシック認証の無効化**（55ページ）」を参照してください。

5. 必要に応じて、データベース リソースを定義します。

   データベース リソースを定義する必要があるか確認するには、**[リソース] > [Spectrum データベース]** を選択します。**[Spectrum データベース]** メニューが表示されなければ、データベース リソースを定義する必要はありません。

6. Spectrum™ Technology Platform からアクセスするデータベース、ファイル サーバー、および他のデータソースを定義します。データ ソースを定義するには、**[リソース] > [データ ソース]** を選択します。
7. Spectrum™ Technology Platform サーバーのバックアップ スケジュールを構成して、深刻なシステム障害や災害が発生した場合もサーバーを復旧できるようにします。詳細については、「**スケジュール バックアップ情報**（266ページ）」を参照してください。

Management Console へのアクセス

Management Console は、Spectrum™ Technology Platform を管理するためのツールです。Management Console を使用すると、次のようなタスクを実行できます。

- ユーザと他のセキュリティ オプションを管理する。
- データベースや Web サービスなどのデータソースへの接続を定義する。
- リモート コンポーネント (データベース) のプロパティを設定する。
- サービスのデフォルト設定を指定します。
- ジョブ実行をスケジュールする。

Management Console にアクセスするには

1. Web ブラウザで次の URL を表示します。

   http://*サーバー*:*ポート*/`managementconsole`

   ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

2. 有効なユーザ名とパスワードを入力します。

   管理ユーザ名は "admin" で、そのデフォルトのパスワードは "admin" です。

   重要: システムに対して不正な管理者アクセスが行われることを回避するために、Spectrum™ Technology Platform をインストールした後すぐに admin パスワードを変更してください。

言語と地域の設定

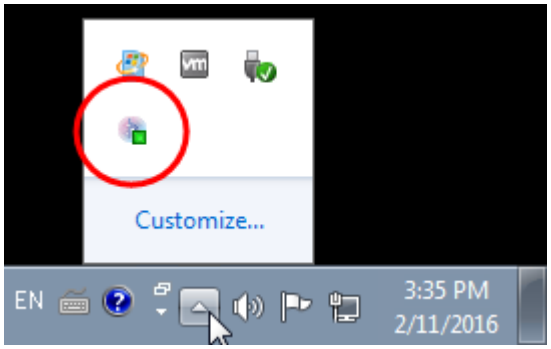
Management Console は、英語、フランス語、日本語、ドイツ語、スペイン語で表示することができます。また、ブラウザの言語がこれらの言語のいずれかである限り、その設定で表示されます。それ以外の場合は、英語で表示されます。ブラウザの言語または設定が利用できない場合は、以下の手順に従って、使用可能な言語のいずれかを強制的に設定できます。

1. Management Console にログインします。
2. 右上隅のユーザ メニューをクリックします。
3. **[プロファイル]** を選択します。
4. **[言語]** フィールドで、使用する言語を選択します。
5. **[国]** フィールドで、地域を選択します。この設定は、日付および時刻を表示する際に使う書式を制御します。
6. **[保存]** をクリックします。

サーバーの起動および停止

検索テーブルのインストールや製品アップデートの適用などのメンテナンスを実施する場合は、Spectrum™ Technology Platform サーバーを停止し、起動する必要があります。

- Windows の場合、Spectrum™ Technology Platform は、Windows の起動時に自動的に起動するように設定されます。Spectrum™ Technology Platform が起動したことを確認するには、Windows システム タスクの Spectrum™ Technology Platform アイコンを右クリックします。アイコンが緑色である場合、サーバーは起動しています。



  Spectrum™ Technology Platform を停止するには、アイコンを右クリックして **[Spectrum™ を停止する]** を選択します。
- Unix または Linux 上のサーバーを起動するには

a) 作業ディレクトリを、`bin` のインストール場所の Spectrum™ Technology Platform ディレクトリに変更します。

例:

```
cd /usr/g1/tst/server/bin
```

b) セットアップ ファイルのソースを指定します。

例:

```
. ./setup
```

c) Spectrum™ Technology Platform を起動します。

- Spectrum™ Technology Platform をバックグラウンドで起動するには、次のコマンドを入力します。

  ```
  ./server.start
  ```

- Spectrum™ Technology Platform をフォアグラウンドで起動するには、次のコマンドを入力します。

  ```
  ./server.start console
  ```

- Unix または Linux 上で Spectrum™ Technology Platform を停止するには、次のコマンドを入力します。

  ```
  ./server.stop
  ```

注： Java は、デフォルトで `/var/tmp` を一時ディレクトリとして使用します。このディレクトリに十分な空きスペースがない場合、Spectrum™ Technology Platform サーバーは起動しません。

クライアント ツールのインストール

Spectrum™ Technology Platform クライアント ツールは、サーバーの管理や、データフローの設計および実行に使用するアプリケーションです。クライアント ツールをインストールする前に、Spectrum™ Technology Platform サーバーをインストールする必要があります。

インストールする前に、リリース ノートに目を通してください。リリース ノートには、既知の問題のリスト、互換性に関する重要な情報、リリースに固有のインストール上の注意事項が記載されています。

この手順では、以下のクライアント ツールのインストール方法について説明します。

- **Enterprise Designer** — データフローの作成、変更、実行に使用します。
- **Job Executor** — コマンド ラインまたはスクリプトからジョブを実行できるコマンド ライン ツールです。ジョブは、Enterprise Designer を使用して、Spectrum™ Technology Platform で作成および保存されたものである必要があります。
- **Process Flow Executor** — コマンド ラインまたはスクリプトからプロセス フローを実行することのできるコマンド ライン ツールです。プロセス フローは、Enterprise Designer を使用して、Spectrum™ Technology Platform で作成および保存されたものである必要があります。
- **Administration Utility** — 管理ユーティリティでは、いくつかの管理機能をコマンドラインから実行できます。この機能はスクリプトで利用できるため、特定の管理タスクを自動化できます。対話式の操作で機能を実行することもできます。

クライアント ツールをインストールするには、次の操作を行います。

1. Web ブラウザを起動し、次の Spectrum™ Technology Platform の Welcome ページを開きます。

   `http://<servername>:<port>`

   例えば、Spectrum™ Technology Platform が "myspectrumplatform" という名前のコンピュータにインストールされており、デフォルトの HTTP ポート 8080 を使用している場合は、次のアドレスに移動します。

   `http://myspectrumplatform:8080`

2. **[プラットフォーム クライアント ツール]** をクリックします。
3. インストールするクライアント ツールをダウンロードします。

ネットワーク ポート

Spectrum™ Technology Platform サーバーは、通信にいくつかのネットワーク ポートを使用します。ネットワーク ポートに競合があると、モジュール コンポーネントが起動に失敗する恐れがあります。コンポーネントが Management Console に表示されない場合は、コンポーネントが起動に失敗したことを表しています。この問題をトラブルシューティングするには、Spectrum™ Technology Platform の `spectrum-server.log file` を確認してください。このログには、どのポートが問題を起こしているかが示されています。Spectrum™ Technology Platform のサーバー ログは以下で確認できます。

`server\spectrum-server.log`

spectrum-container.properties ファイルに定義されているサーバー ポート設定

次のファイルのプロパティを修正してサーバーを再起動することによって、ネットワーク ポートを変更できます。

`server\conf\spectrum-container.properties`

> **注:** クラスタ環境では、クラスタ内の*ノードごとに* `spectrum-container.properties` ファイルを変更する必要があります。

ポート	説明
5001	このポートは、Spectrum™ Technology Platform 構成データベースが使用します。 非クラスタ環境で別のポートを使用するには、repository/neo4j.template を設定します。 クラスタ環境で別のポートを使用するには: • 5001 の代わりに使用するポートを `spectrum.repository.server.coordinator.port` に指定します。 • 構成データベースのシード ノードを **spectrum.repository.server.seeds** に指定します。
5701	このポートは、クラスタ内の Spectrum™ Technology Platform サーバー間の分散処理を管理するために Hazelcast が使用します。 非クラスタ環境で別のポートを使用するには、次のプロパティを変更します。 `spectrum.cluster.port` クラスタ環境で別のポートを使用するには: • 5701 の代わりに使用するポートを `spectrum.cluster.port` で指定します。 • `spectrum.cluster.seeds` に指定したすべての IP アドレスの最後に Hazelcast ポート番号を追加します。例えば、`spectrum.cluster.port` が 5702 に設定され、シード ノードの IP アドレスが 1.2.3.4.5 である場合、`1.2.3.4.5:5702` を `spectrum.cluster.seeds` に指定します。
6362	このポートは、Spectrum™ Technology Platform 構成データベースのバックアップが有効になっている場合に使用されます。別のポートを使用するには、次のプロパティを変更します。 `spectrum.repository.backup.port`
7474	このポートは、Spectrum™ Technology Platform 構成データベースが使用します。別のポートを使用するには、neo4j.template を設定します。
7687	このポートは、Spectrum™ Technology Platform 構成データベースが使用します。別のポートを使用するには、**spectrum.repository.port** プロパティを設定します。

ポート	説明
8080	サーバーと Enterprise Designer と Management Console の間の通信に使用するポート。このポートは Web サービスも使用します。別のポートを使用するには、次のプロパティを変更します。 `spectrum.http.port`
9200	このポートは、インデックス サーバーが使用します。別のポートを使用するには、次のプロパティを変更します。 `spectrum.index.http.port`
9300	このポートは、Advanced Matching モジュールで使用される検索インデックス エンジンが使用しており、その設定は、`spectrum.index.tcp.port` で行います。
10119	このポートは、サービスに対する API 呼び出しで使用されます。別のポートを使用するには、次のプロパティを変更します。`spectrum.socketgateway.port`
32751	このポートは、Metadata Insights で作成される ODBC 接続モデル ストアで使用されます。別のポートを使用するには、次のプロパティを変更します。 `spectrum.metadata.odbc.port`

neo4j.properties ファイルに定義されている Data Hub ポート設定

次のファイルのプロパティを修正してサーバーを再起動することによって、Data Hub ポートを変更できます。

`/server/modules/hub/db/neo4j.properties`

ポート	説明
6044-6299	これらのポートは、Data Hub モジュールで使用されます。この指定は次のプロパティによって行います。 `ha.host.data.port`
6372-6627	これらのポートは、Data Hub モジュールで使用されます。この指定は次のプロパティによって行います。 `dbms.backup.address`

ポート	説明
7001	このポートは、Data Hub モジュールで使用されます。この指定は次のプロパティによって行います。 `ha.host.coordination.base_port`

Machine Learning モジュール

このセクションに記載のポートは、Machine Learning モジュールで使用する必要のあるポートです。

ポート	説明
15431	ポート 15431 は、Machine Learning モジュールで使用する必要のあるポートです。

ライセンスと有効期限の通知の設定

この手順は、有効期限通知を送信するタイミングと通知メールの受取人の指定方法を示しています。

1. Management Console を開きます。
2. **[システム] > [ライセンスと有効期限]** を選択します。
3. **[通知の設定]** をクリックします。
4. **[通知を送る]** チェック ボックスをオンにします。
5. **[期限切れまでの日数]** フィールドに、ライセンス、ソフトウェア、またはデータの有効期限をその何日前に通知するかを、有効期限までの日数で指定します。この値がデフォルトです。**[システム] > [ライセンスと有効期限]** ページで、ライセンス項目ごとに異なる通知期間を指定できます。

   例えば、項目の有効期限が切れる 30 日前の通知を希望する場合は、 `30`と指定します。

6. **[受取人]** の下で、追加ボタン をクリックし、有効期限通知メールを受け取る電子メール アドレスを入力します。必要に応じて、複数の電子メール アドレスを入力できます。
7. **[保存]** をクリックします。

以上で、通知の受取人と、有効期限の何日前に通知メールが送信されるかが指定されました。電子メールの送信に使用するメール サーバーをまだ設定していない場合は、その設定を行う必要があります。メール サーバーの設定が済むまでは、通知は送信されません。

注: デフォルトでは、有効期限が近づいているすべての項目 (ライセンス、データベース、ソフトウェア コンポーネントなど) に対して有効期限通知が送信されます。**[システム] > [ライセンスと有効期限]**を選択して、特定の項目の有効期限通知を無効にすることができます。

バージョン情報の表示

1. Web ブラウザで次の URL を表示します。

   http://*サーバー*:*ポート*/managementconsole

   ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

2. **[システム] > [バージョン]**をクリックします。

2 - 承認フロー

このセクションの構成

エンティティの承認フロー

Metadata Insights でのエンティティ作成は、定義済みの承認フローに従います。承認フローは以下を指定します。

- 必要な承認のレベル
- 各種承認レベルの役割
- 指定されたレベルの各承認者が変更または承認に関して振り返りが必要な日数

1. **Management Console** のメイン メニューで、**[リソース] > [承認フロー]** の順にクリックします。
   定義できる承認フローの 2 つのカテゴリを示す **[承認フロー]** ページが表示されます。
2. ページ左上の **[編集]** アイコンの前にある **[Metadata Insights.Business Glossary エンティティ]** のチェックボックスをオンにします。

   > **注：** **[Metadata Insights.Business 用語集エンティティ]** リンクを直接クリックすることもできます。

   [承認フローの編集] ページが表示されます。このページでは、**[エンティティ タイプ]** が **[Metadata Insights.Business 用語集エンティティ]** になっています。
3. **[承認フローを有効にする]** スライダーをクリックして、［はい］にします。
   これで、これから定義するフローが有効になります。このフローは、定義されるすべての用語集に適用できます。
4. ページ右側のテーブルの上にある **[承認レベル]** ボタンをクリックします。
   テーブルの下に 1 行追加されます。
5. 以下の承認フロー詳細情報を入力します。

フィールド	説明
レベル	承認レベルを定義します。 1 は第 1 レベルの承認、2 は第 2 レベルの承認、3 は第 3 レベルの承認を示します。

フィールド	説明
Roles	このレベルの承認者となる役割を選択します。例えば、レベルが 1 の場合は、ここで選択した役割がすべての用語集エンティティで第 1 レベルの承認者となります。 **注：** ここに表示される役割は、管理者が **[システム] > [セキュリティ] > [役割]** オプションを使用して定義したものです。
承認までの日数	承認者がエンティティの編集内容を承認または提案するまでにかけられる日数を選択します。

6. 別のレベルの承認を追加するには、**[承認レベル]** ボタンを再びクリックし、ステップ 5 の指示内容に従います。
7. **[保存]** をクリックします。
   承認フローが保存され、**[承認フロー]** ページに表示されます。

   注：[承認フロー] ページでは、ワークフローを有効または無効にすることもできます。そのためには、ページ上の **[有効]** スライダーを使用します。

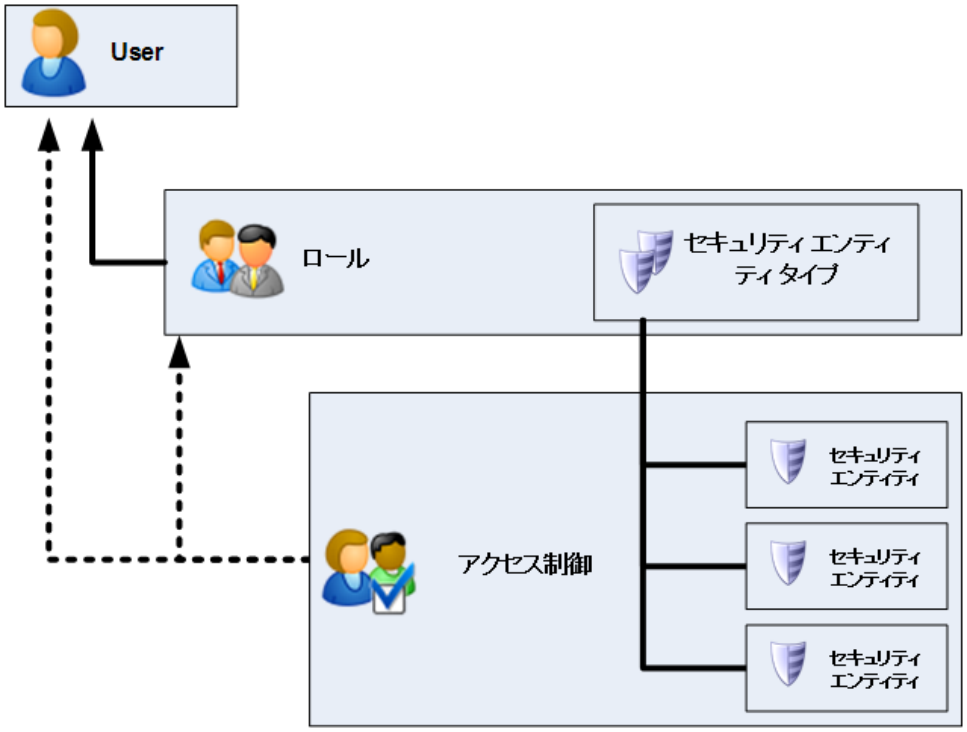
3 - セキュリティ

このセクションの構成

セキュリティ モデル

Spectrum™ Technology Platform は、役割ベースのセキュリティ モデルを使用して、システムへのアクセスを制御します。

以下の図に、Spectrum™ Technology Platform セキュリティ モデルの重要な概念を示します。



ユーザとは、各個人に割り当てられるアカウントのことで、Spectrum™ Technology Platform に対する認証、つまり Enterprise Designer や Management Console などのクライアント ツールに対する認証、または Web サービスや API を介してサービスを呼び出すときの認証で使用されます。

ユーザには 1 つ以上の役割が割り当てられます。*役割*とは、システムのさまざまな部分に対するアクセスを付与または拒否する権限の集合です。通常、役割は、特定のタイプのユーザがシステムとの間で行う対話操作の種類を表します。例えば、データフローの作成および変更アクセスを付与する、データフロー設計者向けの役割や、既存のデータフローを利用したデータ処理のみを必要とするユーザ向けの役割があります。

役割は、セキュア エンティティ タイプに権限を付与します。*セキュア エンティティ タイプ*とは、アクセスを付与または拒否するアイテムのカテゴリです。例えば、"データフロー" というセキュア エンティティ タイプは、システム上のすべてのデータフローに対するデフォルトの権限を制御します。

アクセスをきめ細かく制御する必要がある場合は、アクセス制御を設定すれば役割またはユーザの設定をオーバーライドすることができます。アクセス制御設定は、役割と連動して、ユーザの権限を定義します。役割は、すべてのデータフローやすべてのデータベース リソースなど、エンティティのカテゴリに対する権限を定義し、アクセス制御設定は、*セキュア エンティティ*と呼ばれる特定のエンティティに対する権限を定義します。セキュア エンティティの例として、特定のジョブや特定のデータベース接続などがあります。アクセス制御設定の定義は必須ではありません。アクセス制御設定を定義しない場合は、役割に定義された権限により、ユーザの権限が制御されます。

アクセス制御設定は、役割と連動して、ユーザの権限を定義します。役割は、すべてのデータフローやすべてのデータベース リソースなど、エンティティのカテゴリに対する権限を定義し、アクセス制御設定は、*セキュア エンティティ*と呼ばれる特定のエンティティに対する権限を定義します。セキュア エンティティの例として、特定のジョブや特定のデータベース接続などがあります。例えば、ある役割によって "データフロー" というセキュア エンティティ タイプに対する変更権限が付与されているが、特定のデータフローをユーザが変更できないようにしたいとします。アクセス制御を使用して変更したくない特定のデータフローへの変更権限を削除することにより、これが可能になります。アクセス制御設定は、ユーザまたは役割に対して指定できます。ユーザに対するアクセス制御設定は、ユーザの役割によって付与された特定のユーザの権限をオーバーライドします。役割に対するアクセス制御設定は、その役割を持つすべてのユーザに適用されます。

ユーザ

Spectrum™ Technology Platform ユーザ アカウントは、ユーザがシステムで実行できるアクションのタイプを制御します。

以下のアクションを行うには、ユーザ アカウントが必要です。

- Management Console、Enterprise Designer、Metadata Insights、コマンドライン ツールなどのツールを使用する。
- ジョブをスケジュールどおりに実行する。
- ジョブをコマンド ラインから実行する。
- Web サービスまたは API を介してサービスにアクセスする。

システムに付属する管理アカウントとして**admin**があります。このアカウントは、完全アクセスを持ちます。初期パスワードは "admin" です。

重要： システムに対して不正な管理者アクセスが行われることを回避するために、Spectrum™ Technology Platform をインストールした後すぐに admin パスワードを変更してください。

ユーザ アカウントは必要に応じていくつでも作成できます。

Management Console を用いたユーザの追加

この手順では、Spectrum™ Technology Platform ユーザ アカウントを作成し、そのユーザ アカウントに役割を割り当てる方法について説明します。

ユーザ アカウントを追加して役割を割り当てるには:

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. [追加] ボタン をクリックします。
4. このユーザ アカウントを使用できるようにする場合は、**[有効]** スイッチを **[オン]** のままにしておきます。
5. **[ユーザ名]** フィールドにユーザ名を入力します。

   注 : ユーザ名には ASCII 文字のみを使用できます。ユーザ名は大文字と小文字が区別されます。

6. **[電子メール アドレス]** フィールドにユーザの電子メール アドレスを入力します。電子メール アドレスは、ユーザに通知を送信するために一部のモジュールで使用されます。
7. **[説明]** フィールドにユーザの説明を入力します。
8. ユーザのパスワードを入力し、確認のためにもう一度入力します。
9. このユーザに付与する役割を選択します。

   独自の役割を作成することも、デフォルトの役割を使用することもできます。デフォルトの役割は次のとおりです。

   admin この役割は、システムのすべての部分に対する完全アクセスを持ちます。

   designer この役割は、Enterprise Designer でデータフローとプロセス フローを作成するユーザを対象としています。この役割では、データフローを設計および実行できます。

   integrator この役割は、Spectrum™ Technology Platform を介してデータを処理する必要があるが、データフローを作成または変更する必要のないユーザを対象としています。この役割では、Web サービスと API を介してサービスにアクセスし、ジョブを実行できます。

   user これはデフォルトの役割です。この役割は、システムに対するアクセスを持ちません。この役割を持つユーザは、セキュア エンティティ オーバーライドを介して権限が付与された場合にのみシステムにアクセスできます。

   役割の作成については、**Management Console での役割の作成**（27ページ）を参照してください。

10. **[保存]** をクリックします。

パスワードの変更

この手順は、ユーザのパスワードの変更方法を示しています。

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. ユーザを選択し、編集ボタン をクリックします。
4. **[パスワードの変更]** をクリックします。
5. 新しいパスワードを入力し、確認のためにもう一度入力します。
6. **[保存]** をクリックします。

最小パスワード長の設定

この最小パスワード長は、パスワードの作成または変更時に適用されます。既存のパスワードについては、この最小長よりも短いものであっても引き続き有効となります。

最小パスワード長を設定するには:

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. 管理者アカウントでログインします。
3. "Domain: com.pb.spectrum.platform.config" の下にある **[com.pb.spectrum.platform.config:manager=AccountConfigurationManager]** をクリックします。
4. **[updatePasswordPolicy]** 操作の **[enableAdvanceControl]** オプションを **[True]** に設定します。
5. **[minLength]** フィールドに最小パスワード長を入力します。
6. **[Invoke]** をクリックします。
7. **[Return to MBean View]** をクリックして、アカウント構成マネージャーの画面に戻ります。

電子メール アドレスの変更

ユーザ アカウントに関連付けられた電子メール アドレスは、一部の Spectrum™ Technology Platform モジュールによる通知を受け取ります。

電子メール アドレスを変更するには、以下の手順に従います。

1. Management Console にログインします。
2. 右上隅のユーザ メニューをクリックします。
3. **[プロファイル]** を選択します。
4. **[電子メール]** フィールドに、新しい電子メール アドレスを入力します。
5. **[保存]** をクリックします。

ユーザ アカウントの無効化

ユーザ アカウントを無効にして Spectrum™ Technology Platform にアクセスできないようにすることができます。

Spectrum™ Technology Platform では、admin アカウントを除く任意のユーザ アカウントを無効にできます。無効になったユーザ アカウントによる実行がスケジュールされているジョブも実行されなくなります。

> **注：** ユーザ アカウント "admin" は無効にできません。

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. 変更するユーザの横にあるチェックボックスをオンにし、編集ボタン をクリックします。
4. **[有効]** スイッチを **[オフ]** にします。
5. **[保存]** をクリックします。

これで、ユーザ アカウントが無効になり、Spectrum™ Technology Platform にアクセスできなくなります。

Management Console を用いたユーザの削除

ユーザ アカウントを完全に削除することができます。

この手順では、Spectrum™ Technology Platform ユーザ アカウントを完全に削除する方法について説明します。

ヒント：また、ユーザ アカウントは無効にすることもできます。ユーザ アカウントを無効にすると、アカウントを削除しなくても、システムにアクセスできなくなります。

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. 削除するユーザの横にあるチェックボックスをオンにし、削除ボタン をクリックします。

注： "admin" ユーザ アカウントは削除できません。

ユーザ アカウントのロック

セキュリティ対策として、ユーザの認証が 5 回連続で失敗した場合はユーザ アカウントが無効になります。この回数には、Enterprise Designer、Management Console、Web サービス、またはクライアント API への認証失敗も含まれます。

管理者は、Management Console にログインし、ユーザを編集して **[有効]** スイッチを **[オン]** に切り替えてユーザ アカウントを再び有効にすることができます。ユーザ アカウントは、管理ユーティリティを使って有効にすることもできます。ユーザ自身がアカウントのロックを解除することはできません。

注： 認証に LDAP または Active Directory を使っている場合は、それらのサービスのアカウント ロック ルールが適用されます。LDAP または Active Directory のルールでは、許容されるログイン失敗の回数が Spectrum™ Technology Platform よりも少ない場合があります。

admin アカウントのロック解除

ログインに何度か失敗すると、Spectrum™ Technology Platform によってユーザ アカウントがロックされます。ロックされたほとんどのユーザ アカウントは Management Console を使用してロック解除できます。ただし、admin アカウントは例外です。admin アカウントをロック解除するには、サーバー上でスクリプトを実行する必要があります。

admin アカウントをロック解除するには:

1. Spectrum™ Technology Platform を実行しているサーバーにログインします。

   クラスタで Spectrum™ Technology Platform を実行している場合は、いずれかのノードにログインします。ロック解除スクリプトは、いずれか 1 つのノードで実行するだけで済みます。

2. コマンド プロンプトを開き、*Spectrum Folder*\server\bin フォルダに移動します。
3. (Unix および Linux のみ) 次のコマンドを実行します。

```
. ./setup
```

4. enableadmin スクリプトを実行します。

   Windows の場合:

   ```
   enableadmin.bat -h HostAndPort -p AdminPassword [-s]
   ```

   Unix および Linux の場合:

   ```
   ./enableadmin.sh -h HostAndPort -p AdminPassword [-s]
   ```

   説明:

HostAndPort	Spectrum™ Technology Platform が使用するホスト名と HTTP ポートです。例: `spectrumserver:8080`
AdminPassword	admin アカウントのパスワードです。ロックされた admin アカウントのパスワードが不明な場合は、Pitney Bowes テクニカル サポートまでご連絡ください。
-s	Spectrum™ Technology Platform が HTTPS を使用するように設定されている場合は、`-s` を指定します。

非アクティブ セッションのログアウト

Enterprise Designer および Web クライアント (Management Console、Relationship Analysis Client、Business Steward Portal など) のユーザは、30 分間動作がない状態が続くと自動的にログアウトされます。システムからログアウトする前に、セッションが期限切れになるという警告メッセージが表示されます。

JMX コンソールによるパスワード サポート

JMX コンソールは、Spectrum™ Technology Platform システムのパスワードの構造を制御するための、一連の属性を提供します。

お使いのサイトのパスワードの必要条件を定義するには、以下の手順を実行します。

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. **Domain: com.pb.spectrum.platform.configuration** の下で、**com.pb.spectrum.platform.configuration:manager=AccountConfigurationManager** を選択します。
3. サイトの必要条件に従って各属性を設定します。

   注：[設定] をクリックして各定義を保存します。

   小文字 少なくとも 1 つの小文字を必須とするかどうかについて、true または false を指定します。デフォルトは false です。

   最小長 パスワードの最小長を指定します。デフォルトは 6 文字です。

   番号 パスワードに少なくとも 1 つの数字を必須とするかどうかについて、true または false を指定します。デフォルトは false です。

   特殊文字 パスワードに少なくとも 1 つの特殊文字を必須とするかどうかについて、true または false を指定します。デフォルトは false です。有効な文字は、次のとおりです。

名前	記号
アンパサンド	&
アスタリスク	*
の	@
ドル	$
パーセント	%
感嘆符	!
疑問符	?

   大文字 少なくとも 1 つの大文字を必須とするかどうかについて、true または false を指定します。デフォルトは false です。

4. **[すべての MBean]** をクリックして、メインの JMX コンソール ページに戻ります。

Spectrum™ Technology Platform のパスワードのグローバルな必要条件の定義が完了しました。

Roles

Spectrum™ Technology Platform には以下の定義済み役割が用意されています。

*役割*とは、システムのさまざまな部分に対するアクセスを付与または拒否する権限の集合です。通常、役割は、特定のタイプのユーザがシステムとの間で行う対話操作の種類を表します。例えば、データフローの作成および変更アクセスを付与する、データフロー設計者向けの役割や、既存のデータフローを利用したデータ処理のみを必要とするユーザ向けの役割があります。

Spectrum™ Technology Platform では以下の役割が事前に定義されています。

admin	この役割は、システムのすべての部分に対する完全アクセスを持ちます。
designer	この役割は、Enterprise Designer でデータフローとプロセス フローを作成するユーザを対象としています。この役割では、データフローを設計および実行できます。
integrator	この役割は、Spectrum™ Technology Platform を介してデータを処理する必要があるが、データフローを作成または変更する必要のないユーザを対象としています。この役割では、Web サービスと API を介してサービスにアクセスし、ジョブを実行できます。
user	これはデフォルトの役割です。この役割は、システムに対するアクセスを持ちません。この役割を持つユーザは、セキュア エンティティ オーバーライドを介して権限が付与された場合にのみシステムにアクセスできます。

これらの各役割に付与されている権限を表示するには、Management Console を開き、**[セキュリティ]** に移動して **[役割]** をクリックします。次に、表示する役割を選択し、**[表示]** をクリックします。

ヒント： あらかじめ定義されている役割は変更できません。ただし、あらかじめ定義されている役割を基に新しい役割をできます。

Management Console での役割の作成

役割とは、ユーザに割り当てる権限の集合です。Spectrum™ Technology Platform にあらかじめ定義されている役割が組織のニーズに合わない場合は、独自の役割を作成できます。

Management Console で役割を作成するには:

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. **[役割]** をクリックします。

4. [追加] ボタン  をクリックします。

   ヒント： 既存の役割とよく似た役割を作成する場合は、コピーしたい役割の横にあるチェックボックスをオンにしてからコピー ボタン  をクリックすると、既存の役割のコピーを作成できます。続いて、その新しい役割を編集してから以下の手順に進みます。

5. **[役割名]** フィールドに、この役割に付与する名前を入力します。任意の名前にすることができます。
6. オプション: セキュア エンティティ タイプの一覧は長くなることがあり、その場合は、セキュア エンティティ タイプの特定のグループのみを表示することもできます。同じ権限をグループ内のすべてのエンティティに適用する場合は、その特定のグループのみを表示すると便利です。例えば、すべてのデータベース リソースから変更権限を削除する場合は、フィルタリングしてデータベース リソース グループのみを表示できます。1 つのグループのみを表示して変更するには:
   a) **[グループのフィルタリングを有効にする]** ボックスをオンにします。
   b) **[グループ]** 列のヘッダにある漏斗（ろうと）アイコンをクリックし、表示するグループを選択します。
   c) 適用する権限の列ヘッダにあるボックスをオンまたはオフにします。
   d) セキュア エンティティ タイプの完全な一覧に戻るには、フィルタ アイコンをクリックし、**[(すべて)]** を選択して、**[グループのフィルタリングを有効にする]** ボックスをオフにします。
7. 各エンティティ タイプに付与する権限を選択します。権限は以下のとおりです。

   ビュー ユーザは、エンティティ タイプに含まれるエンティティを表示できます。例えば、JDBC 接続エンティティ タイプに対する表示権限を許可すると、この役割を持つユーザは Management Console でデータベース接続を表示できます。

   変更 ユーザは、エンティティ タイプに含まれるエンティティを変更できます。例えば、JDBC 接続エンティティ タイプに対する変更権限を許可すると、この役割を持つユーザは Management Console でデータベース接続を変更できます。

   作成 ユーザは、このエンティティ タイプのカテゴリに分類されるエンティティを作成できます。例えば、JDBC 接続エンティティ タイプに対する作成権限を許可すると、この役割を持つユーザは Management Console で新しいデータベース接続を作成できます。

   削除 ユーザは、エンティティ タイプに含まれるエンティティを削除できます。例えば、JDBC 接続エンティティ タイプに対する削除権限を許可すると、この役割を持つユーザは Management Console でデータベース接続を削除できます。

   実行 ユーザは、ジョブ、サービス、プロセス フローの処理を開始できます。例えば、ジョブ エンティティ タイプに対する実行権限を許可すると、この役割を持つユーザはバッチ ジョブを実行できます。例えば、サービス エンティティ タイプに対

する実行権限を許可すると、この役割を持つユーザは Spectrum™ Technology Platform 上で実行されるサービスに API または Web サービスを介してアクセスできます。

8. **[保存]** をクリックします。

これで、ユーザに役割を割り当てることができます。

Management Console での役割の削除

使用されなくなった役割は削除できます。

ユーザへの割り当てがなくなった役割を削除します。

注： 次の役割は削除できません: admin、user、designer、integrator

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. **[ユーザ]** タブで、削除したい役割がどのユーザにも割り当てられていないことを確認します。ユーザに割り当てられている役割は、削除できません。
4. **[役割]** をクリックします。
5. 削除する役割の横にあるチェックボックスをオンにし、削除ボタン をクリックします。

セキュア エンティティ タイプ - Advanced Matching モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。以下のエンティティ タイプは、Advanced Matching モジュールの各部分へのアクセスを制御します。

マッチ ルール管理	Enterprise Designer の Interflow Match ステージ、Intraflow Match ステージ、Transactional Match ステージ、およびマッチ ルール管理ツールへのアクセスを制御します。
検索インデックスの管理	Write to Search Index、Candidate Finder、および Enterprise Designer の検索インデックスの管理ツールにおける、検索インデックスへのアクセスを制御します。

セキュア エンティティ タイプ - Business Steward モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。以下のエンティティ タイプは、Business Steward モジュールの各部分へのアクセスを制御します。

例外

エンティティ タイプは、Business Steward モジュールの例外を変更、削除、管理、参照するためのアクセスを制御します。

ビュー 例外トレンド データを参照できるように、[データ品質] 画面へのアクセスを提供します。表示権限のないユーザは、[データ品質] 画面を見ることができません。

変更 [管理] ページ、[ダッシュボード] ページおよび [エディター] ページ上のすべてのユーザの例外へのアクセスを提供します。変更権限のないユーザは、[管理] ページや他のユーザの例外にはアクセスできません。

> **注：** 変更権限により、Read Exceptions ステージのユーザへのアクセスも制御できます。変更権限のないユーザは、Business Steward Portal から他のユーザの例外を読み込むことはできません。

削除 ユーザが例外を削除できるように、[管理] ページの [完全削除] セクションへのアクセスを提供します。

> **注：** ユーザが [管理] ページにアクセスするには、変更権限も必要です。

セキュア エンティティ タイプの作成に加えて、アクセス制御を使用して、特定のデータフローまたは特定のステージに対する例外レコード制限を指定するセキュア エンティティ オーバーライドを作成することもできます。これらのオーバーライドは、権限をより制限するために、ユーザ ベースおよび役割ベースのセキュリティ エンティティ タイプの設定に優先します。

例えば、ユーザ JohnSmith がセキュア エンティティ タイプに基づいた変更権限を持っていて、彼の管理者がデータフローの特定の例外をだれにも変更させたくないと考えている場合、管理者は John Smith によるそのデータフローの例外の変更を制限するアクセス制御設定を定義できます。この場合、John Smith はより厳格なアクセス制御設定が存在するデータフローを変更できなくなりますが、その他のデータフローは変更できます。

セキュア エンティティ タイプ - Data Federation モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。例えば、"Baseview" というエンティティ タイプは、Data Federation モジュール内のすべての Baseview に対する権限を制御します。

> **注：** Spectrum Data Federation エンティティ タイプは、Data Federation モジュールのみに適用されます。

Data Federation のエンティティ タイプを以下に示します。

Baseview	Data Federation モジュール内のすべての Baseview へのアクセスを制御します。
データソース - **Amazon DynamoDB**	Data Federation モジュールで作成されたすべての Amazon DynamoDB 接続へのアクセスを制御します。
データソース - **Amazon SimpleDB**	Data Federation モジュールで作成されたすべての Amazon SimpleDB 接続へのアクセスを制御します。
データソース - **Apache Cassandra**	Data Federation モジュールで作成されたすべての Apache Cassandra 接続へのアクセスを制御します。
データソース - フラット ファイル	Data Federation モジュールで作成されたすべてのフラット ファイル接続 (固定長フラット ファイルと区切り記号付きフラット ファイルの両方) へのアクセスを制御します。
データソース - **Marketo**	Data Federation モジュールで作成されたすべての Marketo 接続へのアクセスを制御します。
データソース - **MS Dynamics CRM**	Data Federation モジュールで作成されたすべての MS Dynamics CRM オンライン接続へのアクセスを制御します。
データソース - **NetSuite**	Data Federation モジュールで作成されたすべての NetSuite 接続へのアクセスを制御します。
データソース - **Salesforce**	Data Federation モジュールで作成されたすべての Salesforce 接続へのアクセスを制御します。
データソース - **SAP**	Data Federation モジュールで作成されたすべての SAP NetWeaver 接続へのアクセスを制御します。
データソース - **Splunk**	Data Federation モジュールで作成されたすべての Splunk へのアクセスを制御します。
データソース - **SugarCRM On-Demand**	Data Federation モジュールで作成されたすべての SugarCRM On-Demand 接続へのアクセスを制御します。
Metaview	Data Federation モジュールで作成されたすべての Metaview へのアクセスを制御します。

Virtual Data Source Data Federation モジュールで作成されたすべての Virtual Data Source へのアクセスを制御します。

注： Data Federation モジュールでの「実行」権限の付与は、役割にかかわらず、Virtual Data Source エンティティ タイプにアクセスする場合に限られます。

セキュア エンティティ タイプ - Data Hub モジュール

役割を定めた後、各役割に付与される権限を決定できます。以下のエンティティ タイプは、Data Hub モジュールの各部分に対する権限を制御します。

管理 Management Console の Data Hub の設定に対する以下のアクションの実行可否を制御します。

- 表示: ユーザは Data Hub の設定を表示できます。
- 変更: ユーザは Data Hub の設定を変更できます。

アルゴリズム Data Hub Visualization および Relationship Analysis Client に対する実行アルゴリズムの実行可否を制御します。

- 実行: ユーザはアルゴリズムを実行できます。

モデル管理 Data Hub Stages、Data Hub Visualization、Data Hub Browser、および Relationship Analysis Client に対する以下のアクションの実行可否を制御します。

- 表示: ユーザはモデルを表示できます。
- 作成: ユーザはモデル (エンティティ、関連性、プロパティを含む) を作成できます。
- 変更: ユーザはモデル エンティティと関連性プロパティを変更できます。
- 削除: ユーザはモデル、エンティティ、関連性、プロパティを削除できます。

モデル メタデータ Data Hub Stages、Data Hub Visualization、Data Hub Browser、および Relationship Analysis Client に対する以下のアクションの実行可否を制御します。

- 表示: ユーザはモデル メタデータを表示できます。
- 作成: ユーザはモデル メタデータ (エンティティ、関連性、プロパティを含む) を作成できます。
- 変更: ユーザはモデル メタデータ エンティティと関連性プロパティを変更できます。
- 削除: ユーザはモデル メタデータ、エンティティ、関連性、プロパティを削除できます。

注： Write to Hub ステージでは、この権限でモデルをクリアすることもできます。

モニター管理 Relationship Analysis Client に対する以下のアクションの実行可否を制御します。

- 作成: ユーザはモデル エンティティまたは関連性の変更を検出するモニターを作成できます。
- 表示: ユーザはエンティティおよび関連性モニターを表示できます。
- 変更: ユーザはモデル エンティティまたは関連性の変更を検出するモニターを変更できます。
- 削除: ユーザはエンティティまたは関連性モニターを削除できます。

クエリ管理 Data Hub Browser、Data Hub Visualization、および Relationship Analysis Client に対する以下のアクションの実行可否を制御します。

- 作成: ユーザはモデルに対するクエリを作成できます。
- 表示: ユーザはモデルに対するクエリを表示できます。
- 変更: ユーザはモデルに対するクエリを変更できます。
- 削除: ユーザはモデルに対するクエリを削除できます。

テーマ管理 Relationship Analysis Client に対する以下のアクションの実行可否を制御します。

- 作成: ユーザはモデルのテーマを作成できます。
- 表示: ユーザはモデルのテーマを表示できます。
- 変更: ユーザはモデルのテーマを変更できます。
- 削除: ユーザはモデルのテーマを削除できます。

セキュア エンティティ タイプ - Data Normalization モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

次のエンティティ タイプは、Data Normalization モジュールの各部分へのアクセスを制御します。

Advanced Transformer のテーブル	Enterprise Designer のテーブル管理ツールで各 Advanced Transformer テーブルへのアクセスを制御します。
Open Parser カルチャー	Enterprise Designer の Open Parser ドメインエディタ ツールでカルチャーへのアクセスを制御します。
Open Parser ドメイン	Enterprise Designer の Open Parser ドメインエディタ ツールでドメインへのアクセスを制御します。
Open Parser のテーブル	Enterprise Designer のテーブル管理ツールで各 Open Parser テーブルへのアクセスを制御します。
正規化のテーブル	Enterprise Designer のテーブル管理ツールで各正規化テーブルへのアクセスを制御します。

セキュア エンティティ タイプ - データベース リソース

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

使用できるデータベース リソースは、インストールされているモジュールによって異なります。以下に例を示します。

- **Centrus データベース リソース**
- **ルーティング**
- **空間データベース リソース**

セキュア エンティティ タイプ - Enterprise Data Integration モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

Enterprise Data Integration モジュールの各部分へのアクセスを制御するエンティティ タイプが 1 つあります。

キャッシュ	Write to Cache および Query Cache ステージと Management Console のキャッシュ管理ツールで使用するキャッシュへのアクセスを制御します。

セキュア エンティティ タイプ - 外部の Web サービス

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

[外部の Web サービス] カテゴリには、セキュア エンティティ タイプが 1 つあります。

接続	このセキュア エンティティ タイプは、Management Console と Enterprise Designer において外部の Web サービスへのアクセスを制御します。

セキュア エンティティ タイプ - Spatial モジュール

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

Spatial モジュールには、以下のモジュール固有のエンティティ タイプがあります。1

名前付きリソース	Spatial モジュール内のすべての名前付きリソースに対する権限を制御します。Spatial モジュール サービスのユーザには、最低限、使用するリソースとその従属リソースに対する読み込み権限が必要です。(Spectrum Spatial™ Manager、管理ユーティリティ、Named Resource サービス、WebDAV などの任意のツールを使用して) 名前付きリソースが作成されると、新しい LocationIntelligence.Named

Resource セキュア エンティティがその名前付きリソースに対して自動的に作成されます。

Dataset.DML Spatial モジュール内で使用される、名前付きテーブルと関連付けられたデータセットに対する権限を制御します。名前付きテーブルが作成またはアップロードされると (Spectrum Spatial™ Manager、管理ユーティリティ、Named Resource サービス、WebDAV などの任意のツールを使用)、新しい LocationIntelligence.Dataset セキュア エンティティがその名前付きテーブルに関連付けられたデータセットに対して自動的に作成されます。ユーザが、書き込み可能 (JDBC ベース) テーブルで DML 操作を行うには、名前付きテーブルに対する表示権限、およびデータセットに対する作成/変更/削除権限が必要です。DML 操作には挿入、更新、および削除の操作があり、Write Spatial Data ステージまたは Feature Service を使用して実行します。

セキュア エンティティ タイプ - Metadata Insights

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。Metadata Insights のセキュア エンティティ タイプは、Metadata Insights Web アプリケーションのモデリング、プロファイリング、系統および影響分析の各機能へのアクセスを制御します。

承認フロー Business Glossary エンティティおよび意味型の承認フローへのアクセスを制御します。

Business Glossary エンティティ Business Glossary エンティティの CRUD 操作を制御します。

Business Glossary 意味型 Business Glossary 意味型の CRUD 操作を制御します。

系統および影響分析 Metadata Insights の **[系統および影響分析]** ビューへのアクセスを制御します。系統および影響分析では、情報を表示することしかできないので、適用可能な権限は **[表示]** のみです。

> **注：** **[系統および影響分析]** の **[表示]** 権限をユーザに与えると、そのユーザは、論理モデル、物理モデル、データ ストアの各セキュア エンティティ タイプに対する **[表示]** 権限がなくても、それらを表示できます。

論理モデル Metadata Insights の **[モデリング]** セクションの論理モデルへのアクセスを制御します。

Model Store Metadata Insights の **[モデリング]** セクションのモデル ストアへのアクセスを制御します。

物理モデル Metadata Insights の **[モデリング]** セクションの物理モデルへのアクセスを制御します。

セキュア エンティティ タイプ - プラットフォーム

エンティティ タイプとは、アクセスを付与または拒否するアイテムのカテゴリです。

例

"データフロー" というエンティティ タイプを想定します。システム上のすべてのデータフローに対する権限は、このエンティティ タイプによって制御されます。プラットフォーム エンティティ タイプは、Spectrum™ Technology Platform のすべてのインストールに適用され、モジュール固有のエンティティ タイプは、その特定のモジュールをインストールした場合にのみ適用されます。プラットフォームレベルのエンティティ タイプを以下に示します。

監査ログ	Management Console の **[システム] > [ログ] > [監査ログ]** エリアへのアクセスを制御します。
データフロー	Enterprise Designer ですべてのデータフロー (ジョブ、サービス、およびサブフロー) へのアクセスを制御します。 **注 :** 編集権限がないユーザには、エクスポーズされたバージョンと最終保存バージョンのみが Enterprise Designer の**[バージョン]** ウィンドウに表示されます。
データフロー - エクスポーズ	データフローを実行のために使用可能にできるかどうかを制御します。 **注 :** データフローの最終保存バージョン (Enterprise Designer の **[バージョン]** ウィンドウで常に最上位にあるバージョン) をエクスポーズするためには、ユーザが **[データフロー - エクスポーズ]** セキュア エンティティ タイプに対する編集権限に加えて **[データフロー]** セキュア エンティティ タイプに対する編集権限も持っている必要があります。最終保存バージョンは、エクスポーズ前にまずバージョンとして保存する必要があり、そのためにはデータフローに対する編集権限が必要になるからです。
フローのデフォルト - データ タイプの変換	Management Console の **[フロー] > [デフォルト] > [データ タイプの変換]** エリアへのアクセスを制御します。すべてのユーザがデータ タイプの変換オプションに対する View アクセスを持ちます。View アクセスを削除することはできません。
フローのデフォルト - 形式に誤りのあるレコード	Management Console の **[フロー] > [デフォルト] > [形式に誤りのあるレコード]** エリアへのアクセスを制御します。すべてのユーザが形式に誤りのあるレコード オプションに対する View アクセスを持ちます。View アクセスを削除することはできません。

フローのデフォルト - レポート	Management Console の **[フロー] > [デフォルト] > [レポート]** エリアへのアクセスを制御します。すべてのユーザがレポート オプションに対する View アクセスを持ちます。View アクセスを削除することはできません。
フローのデフォルト - ソート パフォーマンス	Management Console の **[フロー] > [デフォルト] > [ソート パフォーマンス]** エリアへのアクセスを制御します。すべてのユーザがソート パフォーマンス オプションに対する View アクセスを持ちます。View アクセスを削除することはできません。
フロー履歴 - ジョブ	Enterprise Designer と Management Console のジョブ実行履歴への表示アクセスを制御します。
フロー履歴 - プロセス フロー	Management Console と Enterprise Designer でプロセス フロー実行履歴へのアクセスを制御します。
フロー履歴 - トランザクション	Management Console の **[フロー] > [履歴] > [トランザクション]** エリアへのアクセスを制御します。
フロー スケジューリング	Management Console の **[フロー] > [スケジュール]** エリアへのアクセスを制御します。
ジョブ	Enterprise Designer、Management Console、Job Executor、および管理ユーティリティでのジョブ実行を制御します。
通知 - ライセンス有効期限	Management Console でライセンス有効期限通知の電子メール設定へのアクセスを制御します。
通知 - SMTP 設定	Management Console の **[システム] > [メール サーバー]** エリアへのアクセスを制御します。
プロセスフロー	Enterprise Designer でプロセス フローへのアクセスを制御します。 **注 :** 編集権限がないユーザには、エクスポーズされたバージョンと最終保存バージョンのみが Enterprise Designer の**[バージョン]** ウィンドウに表示されます。
プロセス フロー - エクスポーズ	プロセス フローを実行で使用できるようにする Enterprise Designer の機能を制御します。 **注 :** プロセス フローの最終保存バージョン (Enterprise Designer の **[バージョン]** ウィンドウで常に最上位にあるバージョン) をエクスポーズするためには、ユーザが **[プロセス フロー - エクスポーズ]** セキュア エンティティ タイプに対する編集権限に加えて **[プロセス フロー]** セキュア エンティティ タイプに対する編集権限も持っている必要があります。最終保存バージョンは、エクスポーズ前にまずバージョンとして保存する必要があり、そのためにはデータフローに対する編集権限が必要になるからです。

リソース - データベース接続	Management Console でデータベース接続を設定する機能を制御します。
リソース - 外部の Web サービス	Management Console で外部の Web サービスの管理へのアクセスを制御します。
リソース - ファイル サーバー接続	Management Console でファイル サーバーを設定する機能を制御します。
リソース - JDBC ドライバ	Management Console で JDBC ドライバを設定する機能を制御します。
リソース - リモート サーバー	Management Console の **[リソース] > [リモート サーバー]** エリアへのアクセスを制御します。
セキュリティ - アクセス制御	Management Console の **[システム] > [セキュリティ] > [アクセス制御]** エリアのアクセス制御設定へのアクセスを制御します。
セキュリティ - アクセス トークン	ユーザのトークンを表示し、トークンを削除する機能を制御します。トークンは、クライアントとサーバーとの間の認証に使用されます。読み取り権限があれば、それぞれアクティブなセッションを表すアクティブなトークンのリストを表示できます。削除権限があれば、ユーザーのトークンを削除し、そのセッションを終了させることができます。
セキュリティ - ディレクトリ アクセス	Management Console の **[システム] > [セキュリティ] > [ディレクトリ アクセス]** エリアを使ってサーバー ディレクトリ リソースの制限を有効または無効にする機能を制御します。
セキュリティ - ディレクトリ パス	Management Console の **[システム] > [セキュリティ] > [ディレクトリ アクセス]** エリアでサーバー ディレクトリ リソースを設定する機能を制御します。
セキュリティ - オプション	Management Console の **[システム] > [セキュリティ] > [役割]** エリアでセキュリティを有効または無効に切り替える機能へのアクセスを制御します。
セキュリティ - 役割	Management Console の **[システム] > [セキュリティ] > [役割]** エリアの役割設定機能へのアクセスを制御します。
セキュリティ - ディレクトリ パス	Management Console の **[システム] > [セキュリティ] > [ディレクトリ アクセス]** エリアでサーバー ディレクトリ リソースを設定する機能を制御します。
セキュリティ - ユーザ	Management Console の **[システム] > [セキュリティ] > [ユーザ]** エリアのユーザ アカウント管理機能へのアクセスを制御します。
サービス	API と Web サービスを介してサービスを実行する機能を制御します。
ステージ	エクスポーズされたサブフローを Enterprise Designer のデータフローのステージとして使用できるかどうかを制御します。
システム - ライセンス	Management Console の **[システム] > [ライセンスと有効期限]** エリアに表示されるライセンス情報へのアクセスを制御します。

システム - バージョン情報	Management Console の **[システム] > [バージョン]** エリアへのアクセスを制御します。
システム ログ	Management Console のシステム ログへのアクセスを制御します。

アクセス制御

アクセス制御設定は、役割と連動して、ユーザの権限を定義します。役割は、すべてのデータフローやすべてのデータベース リソースなど、エンティティのカテゴリに対する権限を定義し、アクセス制御設定は、セキュア エンティティと呼ばれる特定のエンティティに対する権限を定義します。セキュア エンティティの例として、特定のジョブや特定のデータベース接続などがあります。例えば、ある役割によって "データフロー" というセキュア エンティティ タイプに対する変更権限が付与されているが、特定のデータフローをユーザが変更できないようにしたいとします。アクセス制御を使用して変更したくない特定のデータフローへの変更権限を削除することにより、これが可能になります。アクセス制御設定は、ユーザまたは役割に対して指定できます。ユーザに対するアクセス制御設定は、ユーザの役割によって付与された特定のユーザの権限をオーバーライドします。役割に対するアクセス制御設定は、その役割を持つすべてのユーザに適用されます。

Management Console でのアクセス制御の設定

アクセス制御設定は、役割と連動して、ユーザの権限を定義します。役割は、すべてのデータフローやすべてのデータベース リソースなど、エンティティのカテゴリに対する権限を定義し、アクセス制御設定は、特定のジョブや特定のデータベース接続など、特定のエンティティに対する権限を定義します。

アクセス制御を設定するには、これらのセキュア エンティティ タイプに対する表示と変更の権限が必要です。

- セキュリティ - アクセス制御
- セキュリティ - 役割
- セキュリティ - ユーザ

アクセス制御を設定するには、次の手順を実行します。

1. Management Console で、**[システム] > [セキュリティ]** に移動します。
2. **[アクセス制御]** タブをクリックします。
3. [追加] ボタン をクリックします。
4. 次のいずれかの操作を実行します。

- 役割に対するアクセス制御を指定する場合は、**[役割]** をクリックします。指定するアクセス制御権限は、選択した役割を持つすべてのユーザに影響します。
- 単一のユーザに対するアクセス制御を指定する場合は、**[ユーザ]** をクリックします。指定するアクセス制御権限は、選択したユーザにのみ影響します。

5. アクセス制御を定義する役割またはユーザを選択します。
6. [追加] ボタン をクリックします。
7. 該当するセキュア エンティティが含まれるセキュア エンティティ タイプを選択します。例えば、データフローに対するアクセス制御を設定する場合は、Platform.Dataflows を選択します。
8. アクセス制御制御を設定するセキュア エンティティを選択して **>>** ボタンをクリックし、**[選択中のエンティティ]** リストに追加します。
9. **[追加]** をクリックします。
   選択したセキュア エンティティが表示されます。チェック ボックスにより、選択した役割またはユーザに適用されている権限が示されます。
10. 各セキュア エンティティに付与する権限を指定します。各セキュア エンティティには、次のいずれかの権限を付与できます。

	権限は、役割から継承されます。
	権限は、役割から継承され、オーバーライドできません。
	この権限は、ユーザまたは役割で指定された権限をオーバーライドする形で付与されます。
	この権限は、ユーザまたは役割で指定された権限をオーバーライドする形で拒否されます。

アクセス制御の例

以下に、役割 RetentionDepartmentDesigner のアクセス制御設定を示します。

この例では、プラットフォーム.データフローセキュアエンティティタイプに表示および変更権限が許可されていますが、削除権限は許可されていません。したがって、デフォルトでは、RetentionDepartmentDesigner の役割を持つどのユーザも、すべてのデータフローに対して表示および変更権限を持ちます。ただし、この役割を持つユーザに対し、ExampleJob1 データフローについてのみは変更を許可しないように制限するとします。対象: この場合、ExampleJob1 の [変更] 列のチェックボックスをオフにします。これで、この役割を持つユーザは、このデータフローを変更できなくなりますが、他のデータフローは引き続き変更可能です。

複数ドメインのサポート

***SpectrumDirectory*/server/conf/spring/security/spectrum-config-ldap.properties** ファイルの **spectrum.ldap.dn.format** プロパティで複数のドメインを定義することもできます。

spectrum.ldap.dn.format で複数のドメインを設定するには、個々の完全なドメイン名をカンマで区切り、ドメイン値を単一引用符で囲んで指定します。

```
spectrum.ldap.dn.format='CN=%s,CN=Users,dc=spectest,dc=pvt',
'CN=%s,ou=Services,dc=spectest,dc=pvt','CN=%s,ou=managers,dc=spectest,dc=pvt'
```

Management Console でのアクセス制御設定の削除

ユーザまたは役割のアクセス制御設定を削除すると、そのアクセス制御設定で定義されていた権限オーバーライドがユーザまたは役割から削除されます。

ユーザの場合は、ユーザの役割によって付与された権限がオーバーライドなしで効力を持つことになります。役割の場合は、その役割に定義された権限がオーバーライドなしで効力を持つことになります。

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. **[アクセス制御]** をクリックします。
4. アクセス制御を削除するユーザまたは役割の横にあるボックスをチェックし、削除ボタン をクリックします。

Spatial のセキュリティ

Spectrum Spatial では、Spectrum™ Technology Platform によって提供される、役割ベースのセキュリティを使用します。管理者は、ユーザと役割の両方を Management Console で管理でき、アクセス制御リスト (ACL) と呼ばれる権限を Spectrum Spatial™ Manager の名前付きリソースに割り当てることができます。

ユーザと役割

Spatial 管理者およびサブ管理者は、Spectrum Spatial リポジトリ内の個々の名前付きリソースおよびフォルダへのアクセスまたは編集を行うための権限を、役割およびユーザに与えます。

Spectrum™ Technology Platform Management Console には、ユーザと役割を管理するための設定があります。Spectrum Spatial に関連する役割には、次の 2 種類があります。

1. Spectrum のインストール時に用意される定義済みの役割。こうした役割に属しているユーザには、特定のデフォルト権限が付与されます。
2. 管理者 (admin) が作成するカスタム役割。カスタム役割には Spectrum Spatial™ Manager で指定されるまでは権限がありません。

定義済みの spatial の役割

Spatial モジュールのインストール後、4 つの定義済み役割が Spectrum Management Console で使用可能になります。そのうち、2 つの役割 (spatial-admin と spatial-sub-admin) は、ユーザが Spectrum Spatial 内のコンテンツを管理できるように、admin 関連の権限をユーザに付与するものであり、残り 2 つの役割 (spatial-user と spatial-dataset-editor) は、Spectrum Spatial™ Manager で通常割り当てられているリソース権限をオーバーライドするものです。

spatial-admin

spatial-admin の役割は、Spectrum Spatial リポジトリ内のすべてのコンテンツの確認および管理 (表示、作成、削除、権限の設定) を行うための完全な権限を持ちます。この役割は、Feature サービス (挿入、更新、削

除の各メソッド) を使用して、名前付きテーブルに関連付けられているデータ セットを編集できます。

この役割に割り当てられているユーザは、Spectrum Spatial™ Manager へのログイン、ACL REST API の使用、Map Uploader の使用が可能です。spatial-admin の役割と Spectrum™ Technology Platform admin の役割との主な違いは、spatial-admin ではユーザや役割を Management Console で管理できない点にあります。

spatial-sub-admin

spatial-sub-admin の役割は spatial-admin と類似していますが、Spectrum Spatial リポジトリ内のすべてのコンテンツを表示できるわけではありません。この役割には、読み取り権限があるフォルダ内のコンテンツが表示されます。spatial-sub-admin の役割が割り当てられるユーザは、少なくとも 1 つのフォルダに対する権限を持っている必要があります。

この役割を割り当てられたユーザは、この役割が書き込み権限を持っているフォルダのみを管理 (読み取り、作成、削除、変更、権限の設定) できます。ただし、ユーザが複数の役割を持っている場合があります。その場合は、それらの役割が権限を持つフォルダも管理できます。
spatial-sub-admin の役割は、追加の権限が付与されない限り、名前付きテーブルに関連付けられているデータセットを編集できません。Spectrum Spatial™ Manager へのログイン、ACL REST API の使用、Map Uploader の使用が可能ですが、権限を持つフォルダ内にあるリソースしか確認できません。

また、Spectrum Spatial™ Manager でユーザに spatial-sub-admin の役割を割り当てることもできます。

spatial-user

spatial-user の役割は、Spectrum Spatial リポジトリ内のすべての名前付きリソースに対する読み取り権限を提供し、Spectrum Spatial™ Manager 内の名前付きリソースに対する付与済みの読み取り権限をオーバーライドします。特定の権限を必要とするユーザには、この役割を割り当てないでください。

この役割を割り当てられたユーザは、Spectrum Spatial Web サービスを使用して、タイル、マップ、レイヤをレンダリングしたり、Feature サービスを使用してテーブルに対するクエリを実行したりできます。名前付きテーブルに関連付けられているデータセットを編集することはできません。フォルダ権限は持っていないので、リソースの管理は行えません。

spatial-dataset-editor

spatial-dataset-editor の役割は、名前付きテーブルに割り当てられているすべてのデータセットに対する編集 (挿入、更新、削除) の権限を提供し、Spectrum Spatial™ Manager 内の名前付きテーブルに対する付与済みの権限をオーバーライドします。特定の権限を必要とするユーザには、この役割を割り当てないでください。

この役割を割り当てられたユーザは、Spectrum Spatial Feature サービス (挿入、更新、削除の各メソッド) を使用して、テーブルを編集したり、テーブルに対するクエリを実行したりできます。フォルダ権限は持っていないので、リソースの管理は行えません。

データ フローを作成するデータフロー設計者には、designer の役割 (Management Console で設定済み) が必要です。これは、名前付きリソースにアクセスするための任意の権限に対する追加の役割であり、(すべてのリソースを確認できるように) spatial-user のメンバーにするか、Spectrum Spatial™ Manager を使用して特定の名前付きリソースに対する権限を付与することで割り当てられます。空間データフロー設計者の作成方法については、**空間データフロー設計者の作成**（45ページ）を参照してください。

カスタムの spatial の役割とアクセス制御の設定

Spectrum Spatial でのアクセス制御は、ユーザに割り当てられているカスタム役割を使用して管理されます。これにより、複数ユーザの管理が簡単になります。役割には特定の権限が設定されています。ユーザは、割り当てられている役割の権限を継承します。特定の名前付きリソースにアクセスするための権限を指定するには、Spectrum Spatial™ Manager を使用します。

Spectrum Spatial には、データの表示、編集、管理という 3 種類の権限があります。以下の状況で付与するための役割を作成することをお勧めします。

- 組織全体で使用可能なマップ、レイヤ、テーブルへの読み取り専用アクセス。

  この役割には **GeneralAcces** という名前を付けます。すべてのユーザがこの役割に属している場合、組織内のあらゆるユーザがこれらのマップやレイヤを確認できます。

- 機密扱いのマップおよびレイヤへの読み取り専用アクセス。

  この役割は特定のユーザに割り当てます。その他のユーザはこうしたデータを見ることができません。

- 名前付きテーブルへの編集アクセス。

  例えば、所有地への訪問後にデータを編集する所有用地監査人など、一部のユーザが更新する Property Site Inspections (所有用地監査) というテーブルがあるとします。この役割に編集権限を付与したうえで、用地監査人にこの役割を割り当てることができます。このテーブルを閲覧するその他すべてのユーザは、そのデータを編集できません。

- リポジトリのフォルダ内にあるリソースを管理するための書き込みアクセス。

  例えば、`SalesData` という Spectrum リポジトリ内のフォルダへの書き込み権限を持つ SalesMamagers (販売マネージャー) という役割を作成します。そのうえで、spatial-sub-admin と SalesManager の役割を販売部門の 1 人または 2 人のユーザに割り当てることができます。すると、これらのユーザは Spectrum Spatial™ Manager と Map Uploader ユーティリティを使用して、`SalesData` フォルダ内の名前付きリソースを管理できるようになります。

空間データフロー設計者の作成

designer の役割は、Spectrum™ Technology Platform セキュア エンティティ タイプ (データフローなど) へのアクセスを提供します。

データ フロー、Spectrum Spatial ステージ、サービスを作成するには、ユーザに designer の役割と、データ フロー内の名前付きリソースに対する実行権限が必要です。また、そのユーザは、すべてのリソースを確認するために spatial-user である必要があります。spatial-user の役割には、Spectrum Spatial™ Manager などのアプリケーション内のすべてのリソースを確認できる権限があります。

designer の役割をユーザに割り当てるには:

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. [編集] ボタンをクリックして既存のユーザを選択するか、[追加] ボタン をクリックして新しいユーザを作成します。
4. **[役割]** セクションで、designer と spatial-user の両方の役割をユーザ アカウントに割り当てます。
5. **[保存]** をクリックします。

これで、ユーザは名前付きリソースを表示したり、Spatial モジュールのステージやサービスでリソースを使用してデータフローを設計したりできる権限を持つことになります。

リポジトリに対する WebDAV アクセスの制限

WebDAV は、Spectrum Spatial リポジトリ内のリソースにアクセスして変更するためのプロトコルですが、名前付きリソースに関してそのアクセス制御リスト (ACL) が適切に参照されない可能性があります。

WebDAV の代わりに、Spectrum Spatial™ Manager、Named Resource サービス、および Access Control サービスを使用します。

デフォルトでは、WebDAV を使用したリポジトリへのアクセスは特定のサーバーに制限されるものではなく、リポジトリにアクセス可能なすべてのサーバーに公開されています。空間 java プロパティ ファイルを変更することによって、アクセスを特定のサーバーに制限することができます。これを行うには、WebDAV を公開するホスト名 (IP) の (カンマ区切りの) リストを含むプロパティを追加します。

> **注:** この変更を適用するには、Spectrum™ Technology Platform サーバーを再起動する必要があります。

WebDAV を使用したリポジトリ アクセスを制限するには、次の手順に従います。

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`
   説明:
   - *server* は Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。
   - *port* は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。
2. 管理者アカウントでログインします。
3. **Domain: com.pb.spectrum.platform.configuration.properties** の下で、**com.pb.spectrum.platform.configuration.properties:manager=spatial.properties** をクリックします。
4. **repository.accesscontrol.allows** の値を変更します。

   このプロパティを空のままにすると、Spectrum™ Technology Platform がインストールされているマシンを除くすべてのサーバーに対して、WebDAV を使用したすべてのアクセスが無効になります。

   他のサーバーによる WebDAV アクセスを許可するには、サーバーの IP アドレスをカンマで区切ったリストを入力します。例:

   ```
   192.168.2.1,192.168.2.2
   ```

5. **repository.accesscontrol.allows** の横にある **[設定]** ボタンをクリックします。
6. サーバーを再起動します。

この処理が終わると、リポジトリへの WebDAV アクセスが制限されます。

WebDAV を HTTPS で使用する場合

HTTPS を介してサーバーと通信し、ドライブをリポジトリにマッピングする場合、WebDAV クライアントでは TLS v1.2 プロトコルの使用が必要です。クライアント コンピュータが Windows 7 SP1、Windows Server 2008 R2 SP1、および Windows Server 2012 上で稼働している場合、このプロトコルを利用するために、セキュリティ パッチとレジストリの更新を適用する必要があります。

1. クライアント コンピュータ上で、使用しているオペレーティング システムに該当するパッチをマイクロソフト サポート技術情報 (**https://support.microsoft.com/ja-jp/kb/3140245**) から適用します。

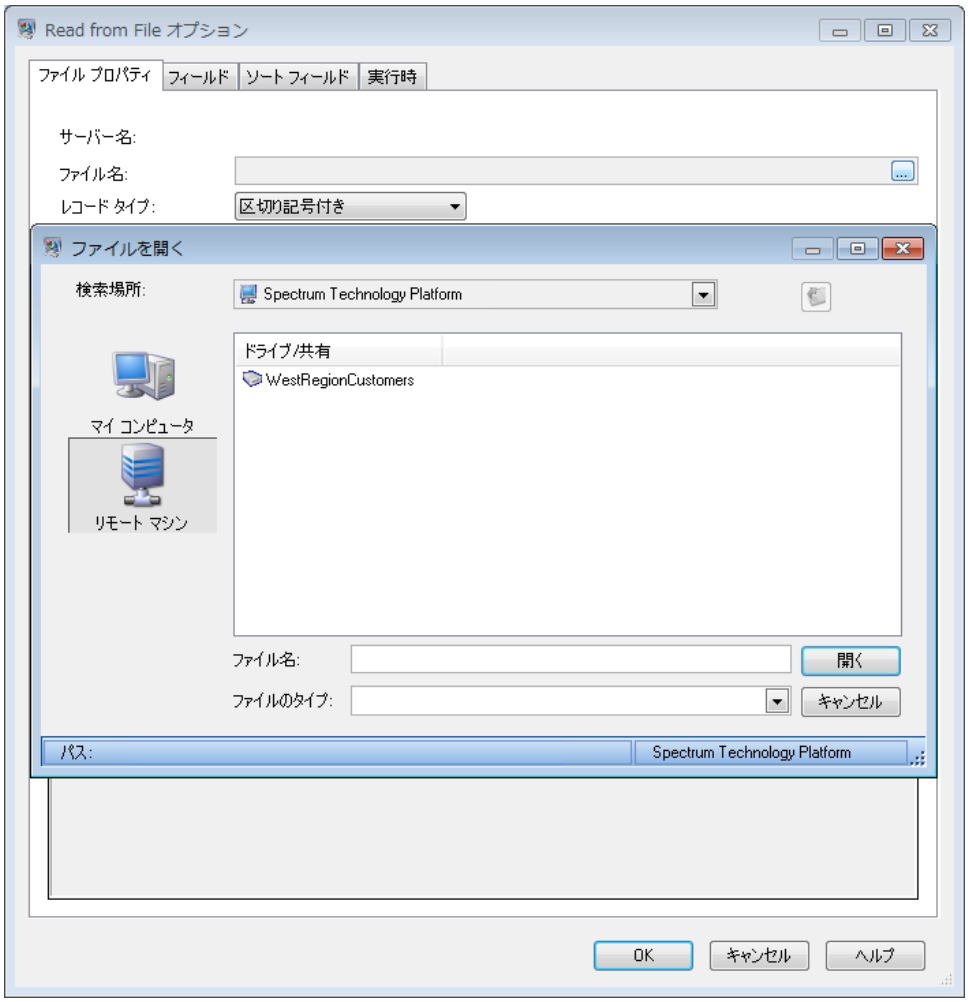
2. サポート技術情報の記事に記載されている手順に従って、レジストリを更新し、TLS v1.2 のサポートを追加します。DefaultSecureProtocols の値を少なくとも 0x00000800 にする必要があります。
3. レジストリのエントリを変更した後、クライアント コンピュータを再起動します。

サーバー ディレクトリ アクセスの制限

ファイルの選択が必要なタスクを実行する際に、Spectrum™ Technology Platform サーバーのフォルダを参照できます。例えば、ユーザは、Enterprise Designer のソースまたはシンク ステージで入力ファイルや出力ファイルを選択する際にサーバーを参照できます。

管理者としては、サーバー内の慎重な取り扱いを要する部分を参照したり変更したりできないように、アクセスを制限したいことがあります。

サーバーのファイル システムへのアクセスを防ぐための 1 つの方法は、プラットフォームのセキュリティ権限である **[セキュリティ - ディレクトリ パス]** を決してユーザに与えないことです。これにより、サーバー上のすべてのフォルダへのアクセスを防ぐことができます。サーバー上の一部のフォルダへのアクセスを禁止して、その他のフォルダへのアクセスを許可することもできます。部分的なアクセスを許可する場合、アクセスを許可したフォルダが、ユーザのファイル参照ウィンドウの最上位フォルダとして表示されます。例えば、サーバー上にある WestRegionCustomers という名前のフォルダへのアクセスのみを許可している場合、ユーザがそのサーバーを参照すると、次のように、そのフォルダのみが表示されます。



次の 2 つの場合は、部分的なアクセスしか許可されていなくても、サーバーのファイル システム全体をユーザが参照できます。

- Management Console で Spectrum データベースを作成するときに、データベース ファイルを参照する場合
- Management Console でドライバを作成するときに、JDBC ドライバ ファイルを参照する場合

サーバーのファイル システム全体をユーザが参照できないようにするには、Spectrum データベースと JDBC ドライバに対するユーザのアクセスを制限する役割を使用します。

サーバー上の一部のフォルダへのアクセスのみを許可するには、以下の手順に従います。

1. Management Console を開きます。
2. **[システム] > [セキュリティ]** を選択します。
3. **[ディレクトリ アクセス]** をクリックします。
4. **[サーバー ディレクトリへのアクセスを制限]** スイッチを **[オン]** に設定します。
5. [追加] ボタン + をクリックします。

6. **[名前]** フィールドに、アクセスを付与するフォルダの名前を指定します。

   ここで指定した名前は、ユーザがサーバーを参照する際に、ディレクトリのルート名として表示されます。このトピックの冒頭で示した例では、アクセス可能なディレクトリに付与された名前は WestRegionCustomers です。

7. **[パス]** フィールドに、アクセスを付与するフォルダを指定します。ユーザは、指定されたフォルダ内に含まれるすべてのファイルとサブフォルダにアクセスできます。
8. **[保存]** をクリックします。
9. 他のフォルダへのアクセスを追加で指定する場合は、前の手順を必要に応じて繰り返します。

これで、ユーザは指定されたフォルダにのみアクセスが許可されます。サーバーのディレクトリにアクセスするには、ユーザーがプラットフォーム セキュリティ権限 **[セキュリティ - ディレクトリ パス]** を持つ必要があることに注意してください。

> **注：** 以前はアクセスできたが、ファイル参照の制限によって現在はアクセスできないファイルがデータフローにある場合、そのデータフローは正しく実行されません。

HTTPS 通信の設定

デフォルトで Spectrum™ Technology Platform サーバーは、Enterprise Designer との通信、Management Console や Metadata Insights などのブラウザ アプリケーションとの通信、Web サービス リクエストや API 呼び出しの処理に HTTP を利用します。

これらのネットワーク通信をセキュリティ保護する場合は、HTTPS を使用するように Spectrum™ Technology Platform を設定できます。

> **注：** Spectrum™ Technology Platform では、通信の暗号化に TLS 1.2 を使用します。Spectrum™ Technology Platform の Web サービスや API にアクセスするアプリケーションは、HTTPS を介して接続するために TLS 1.2 をサポートする必要があります。

以下では、HTTPS 通信を Spectrum™ Technology Platform の単一サーバー インストール環境で有効にする手順を説明します。HTTPS を使用する必要があるが、Spectrum™ Technology Platform をクラスタで実行している場合は、この手順を行わないでください。そのようなケースでは、クライアントとの通信に HTTPS を使用するようにロード バランサーを設定します。ロード バランサーと Spectrum™ Technology Platform ノードとの通信、およびノード間の通信は、Spectrum™ Technology Platform クラスタリングが HTTPS をサポートしないことから、暗号化されます。ロード バランサーとクラスタ内の Spectrum™ Technology Platform サーバーは、セキュアな環境を提供するため、ファイアウォールの反対側にある必要があります。

HTTPS 通信を Spectrum™ Technology Platform の単一サーバー インストール環境で設定するには:

1. Spectrum™ Technology Platform サーバーを停止します。
   - Windows 上のサーバーを停止するには、Windows システム トレイの Spectrum™ Technology Platform アイコンを右クリックして、**[Spectrum™ を停止する]** を選択します。または、Windows の [コントロール パネル] の [サービス] を使用して、Pitney Bowes Spectrum™ Technology Platform サービスを停止できます。
   - Unix または Linux 上のサーバーを停止するには、*SpectrumDirectory*`/server/bin/setup` スクリプトをソースとして、*SpectrumDirectory*`/server/bin/server.stop` スクリプトを実行します。
2. 信頼できる認証局 (CA) によって署名された証明書を作成します。

   注: 証明書は、Spectrum™ Technology Platform で使用されている Java のバージョンに対応する暗号化と長さの要件を満たしている必要があります。Java のバージョンを調べるには、Management Console を開いて、**[システム] > [バージョン]** に移動します。詳細については、**java.com/en/jre-jdk-cryptoroadmap.html** を参照してください。
3. この証明書を JSSE キーストアに読み込みます。詳細については、こちらのリソースを参照してください。
4. 適当なテキスト エディタでファイル `spectrum-container.properties` を開きます。ファイルの場所は *SpectrumDirectory*`/server/conf` です。以下のプロパティを設定します。

   ```
   spectrum.http.default.protocol=https
   spectrum.https.enabled=true
   spectrum.https.port=8443
   spectrum.encryption.validateCerts=false
   spectrum.encryption.trustAllHosts=true
   spectrum.encryption.selfSignedCert=true
   ```

   重要: 署名権限を使用しない自己署名証明書の場合のみ **spectrum.encryption.selfSignedCert=true** を設定します。
5. 自己署名証明書をインポートします。例:

   ```
   keytool -importkeystore -srckeystore "C:\Pitney
   Bowes\Spectrum\server\conf\certs\keystore.p12" -destkeystore "C:\Pitney
   Bowes\Spectrum\server\conf\certs\truststore.p12" -deststoretype pkcs12
   ```

   詳細については、「自己署名証明書の実装（53ページ）」を参照してください。
6. Spectrum Spatial モジュールおよびサービスに対して HTTPS 通信を設定する場合は、追加設定を実行してから Spectrum™ Technology Platform サーバーを再起動する必要があります。

Spectrum Spatial™ Manager で、次のサービス設定の URL を、HTTPS を使用するように変更します。

- Mapping (SOAP 経由でマッピング サービスにアクセスし、RenderMap 要求の ReturnImage パラメータが False の場合にのみ必要)
- WFS
- WMS
- WMTS

手順については、『*Spectrum Spatial ガイド*』の「*Spatial の管理*」セクションにある「*Spectrum Spatial™ Manager*」を参照してください。

7. Spectrum™ Technology Platform サーバーを開始します。
   - Windows 上のサーバーを開始するには、Windows システム トレイの Spectrum™ Technology Platform アイコンを右クリックして、**[Spectrum™ を起動する]** を選択します。または、Windows の [コントロール パネル] の [サービス] を使用して、Pitney Bowes Spectrum™ Technology Platform サービスを開始できます。
   - Unix または Linux 上のサーバーを開始するには、*SpectrumDirectory*`/server/bin/server.start` スクリプトを実行します。

AIX システムでの HTTPS の設定

spectrum-container.properties ファイルの **#ADVANCED SECTION** には、IBM Java Runtime Environment (JRE) または Java Development Kit (JDK) を使用する AIX 環境で使用されるプロパティが含まれています。

これらのプロパティは、TLS プロトコルを使用するネットワークを保護する暗号スイートを確立します。この環境の設定方法は以下のとおりです。

- `spectrum.https.encryption.excludeCipherSuites` からエスケープ シーケンス `^SSL_.*$` を削除します。
- `spectrum.https.encryption.includeCipherSuites` をコメント解除します。

以下のサンプルは、**spectrum-container.properties** ファイル内のプロパティを示しています。

```
#############################################################################
# Comma seperated regex expression for the excluded protocols
# Exclude weak / insecure ciphers
# Exclude ciphers that don't support forward secrecy
# The following exclusions are present to cleanup known bad cipher suites
# that may be accidentally included via include patterns.
# Excludes Null patterns
```

```
# In case of IBM Java (AIX environment please remove ^SSL_.*$
# from the list)
# spectrum.https.encryption.excludeCipherSuites=^.*_(MD5|SHA|SHA1)$,
# ^TLS_RSA_.*$, ^.NULL.$, ^.anon.$
##########################################################################

 spectrum.https.encryption.excludeCipherSuites=^.*_(MD5|SHA|SHA1)$,
 ^TLS_RSA_.*$, ^.NULL.$, ^.anon.$, ^SSL_.*$
##########################################################################
# Only uncomment in case of IBM JRE/JDK on AIX environment
# Comma separated values for the included cipher suites only in case of
# AIX IBM JRE
# Please remove ^SSL_.*$ from the above list
#(spectrum.https.encryption.excludeCipherSuites)
##########################################################################
# spectrum.https.encryption.includeCipherSuites=^SSL_ECDHE.*$,
# ^SSL_DHE.*$,  SSL_RSA.*$, TLS_EMPTY_RENEGOTIATION_INFO_SCSV
```

WebFolders を使用して Spectrum Spatial リポジトリ リソースにアクセスする

名前付きリソースを追加または変更するには、WebDAV ツールを使用してリソースをリポジトリへ、またはリポジトリからコピーすることができます。WebFolders を使用すると、Spectrum Spatial リポジトリとそれに含まれるリソースに簡単にアクセスできます。

> **注：** リポジトリにアクセスするには、Spectrum™ Technology Platform とリポジトリがインストールされているマシンと同じマシンにログインしている必要があります。

Windows 7 で WebFolder を設定するには

1. Windows エクスプローラで、[ネットワークドライブの割り当て] を選択します。
2. ポップアップウィンドウで、[ドキュメントと画像の保存に使用できるWebサイトに接続します] のリンクをクリックして、ネットワークの場所の追加ウィザードを開きます。
3. [次へ] をクリックし、[カスタムのネットワークの場所を選択] を選択します。[次へ] をクリックします。
4. [インターネットまたはネットワークのアドレス] フィールドに、リポジトリ URL を追加します。例えば、`http://<サーバー>:<ポート>/RepositoryService/repository/default/` と入力します。[次へ] をクリックします。
5. 入力を求められた場合は、資格情報 (ユーザ名とパスワード) を入力します。
6. この接続に名前を付けます。例えば、`Spectrum Spatial Repository` とします。[次へ] をクリックします。

完了すると、指定したネットワーク プレースの下にリポジトリのコンテンツへのフォルダ接続が作成されます。

リポジトリへの WebFolder 接続は、他のすべての Windows エクスプローラと同じように使用できます。

> **注：** もしあなたが WebDAV を使って名前付きリソースやメタデータリソースのレコードを変更する場合、同じベース名を同じフォルダには配置しない様に注意してください。そうしないと、リソースに対する移動・名前変更・削除などの操作が、Spectrum Spatial™ Manager のメタデータ更新に反映されなくなります。

自己署名証明書の実装

Spectrum の SSL プロパティでは、認証局 (CA) による証明書検証をさまざまな度合いで制御できます。

CA の役割は、デジタル証明書を信頼できるエンティティに発行し、証明書を評価しようとする SSL プロトコルにその信頼を引き渡すことです。CA は、そのエンティティを確認 (信頼) できない場合、認証をブロックできます。

> **注：** 自己署名証明書は、サポートされていても、実稼働環境では使用しないことをお勧めします。当社では、これがベスト プラクティスであるとは考えていません。一部の認証セキュリティのチェックがオーバーライドされるためです。

SSL プロパティとデフォルト設定

プロパティ/デフォルト	説明
`spectrum.encryption.selfSignedCert=false`	true または false: Spectrum™ Technology Platform で自己署名証明書を実装する
`spectrum.encryption.trustAllHosts=false`	true または false: すべての証明書を暗黙的に信頼し、証明書で指定されているホスト名を検証時に無視する
`spectrum.encryption.validateCerts=true`	true または false: CA による信頼の検証をバイパスする

自己署名証明書のための SSL 処理および優先設定の設定

Spectrum Technology Platform で自己署名証明書を実装するには、最初に **spectrum-container.properties** ファイルのプロパティ `spectrum.encryption.selfSignedCert=true` を設定します。

その他の SSL プロパティを使用すると、認証局 (CA) による証明書検証をより具体的かつ詳細に制御できます。CA の役割は、デジタル証明書を信頼できるエンティティに発行し、証明書を評価しようとする SSL プロトコルにその信頼を引き渡すことです。CA は、そのエンティティを確認 (信頼) できない場合、認証をブロックできます。

• CA による信頼の検証をバイパスする場合は、次のプロパティを設定します。`spectrum.encryption.validateCerts=true`。

• 証明書を (署名の有無を問わず) 暗黙的に信頼するには、プロパティ `spectrum.encryption.validateCerts` が false に設定されている場合は次のプロパティを設定します。`spectrum.encryption.trustAllHosts`。

Web サービスの認証

Spectrum™ Technology Platform Web サービスは要求元に対し、有効なユーザ資格情報による認証を求めます。認証方法には、ベーシック認証とトークンベースの認証の 2 種類があります。

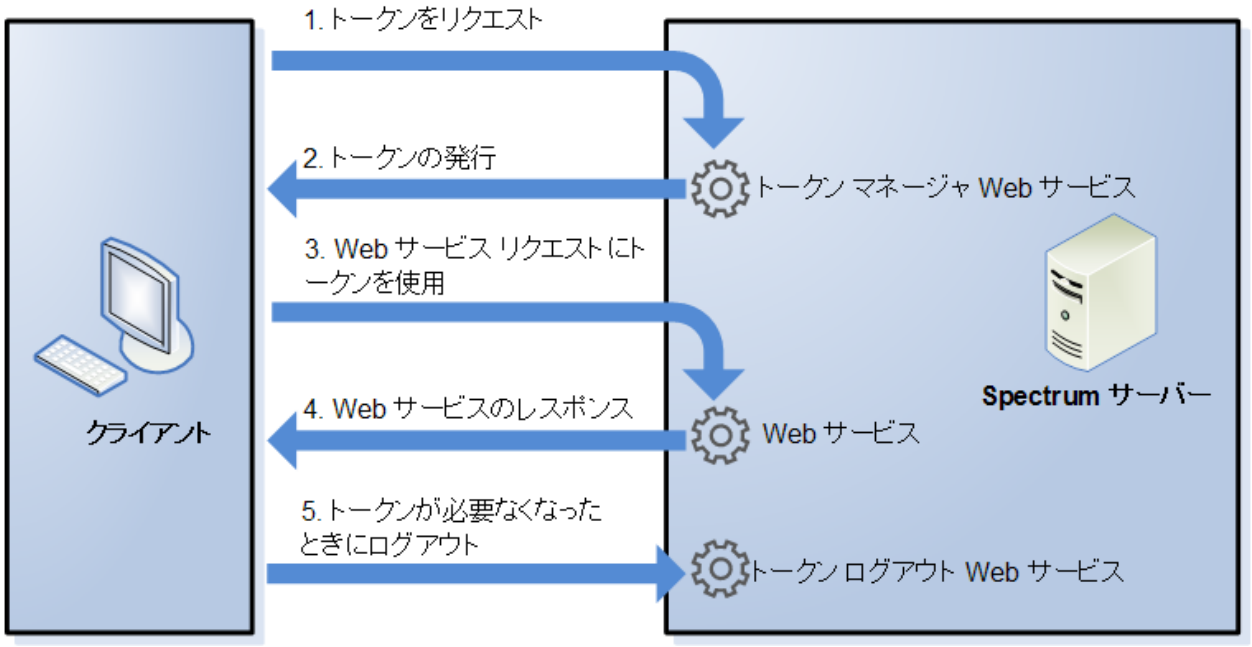
ベーシック認証

ベーシック認証では、ユーザ ID とパスワードが Web サービスへの各要求の HTTP ヘッダに付加されて Spectrum™ Technology Platform に引き渡されます。ベーシック認証はデフォルトで許可されていますが、管理者によってベーシック認証が無効化されている場合もあります。ベーシック認証が無効化されている場合は、トークンベースの認証を使用して Web サービスにアクセスする必要があります。

トークンベースの認証

トークンベースの認証では、要求元は Spectrum™ Technology Platform サーバーからトークンを取得し、そのトークンを使用して Web サービスに要求を送信します。要求ごとにユーザの資格情報を送信する代わりに、トークンがサーバーに送信され、サーバーはトークンが有効であるかどうかを確認します。

次の図は、このプロセスを表したものです。



1. トークン管理サービスに要求を送信することにより、Spectrum™ Technology Platform サーバーからトークンを取得します。
2. トークン管理サービスがトークンを発行します。セッション トークンを要求した場合は、セッション ID も発行されます。
3. HTTP ヘッダにトークンを入れて、対象の Web サービスに要求を送信します。セッション トークンの場合は、セッション ID を HTTP ヘッダに含めます。
4. Web サービスが応答を発行します。そのトークンを使用して、同じ Web サービス、または Spectrum™ Technology Platform サーバー上の他の任意の Web サービスに対して、追加の Web サービス要求を送信することができます。1 つのトークンで送信可能な Web サービス要求数に上限はありませんが、トークンに有効期限 (Time To Live とも呼ばれます) がある場合は、Time To Live の時間が経過するとトークンは無効になります。セッション トークンの場合は、操作がない状態が 30 分間続くと無効になります。
5. トークンが不要になった場合は、トークン ログアウト Web サービスに要求を送信することによって、ログアウトする必要があります。それによってトークンは、Spectrum™ Technology Platform サーバー上の有効トークンのリストから削除されます。

Web サービスのベーシック認証の無効化

Spectrum™ Technology Platform では、Web サービス要求を認証する方法としてベーシック認証とトークン認証の 2 つの方法をサポートしています。デフォルトでは、両方の認証方法が有効です。Web サービス要求に対してベーシック認証ではなくトークン認証を適用したい場合は、以下の手順でベーシック認証を無効にすることができます。

注： ベーシック認証を無効にすると、既存クライアントは失敗することに注意してください。Location Intelligence モジュールの場合、WMS、WMTS、および WFS クライアントでは、ベーシック認証または認証なしが想定されます。トークンベースの認証のみを有効にすると、これらのクライアントは失敗する可能性があります。

1. Spectrum™ Technology Platform サーバーを停止します。
2. 次のファイルをテキスト エディタで開きます。

   SpectrumLocation/server/app/conf/spectrum-container.properties

3. 次のプロパティに false を設定します。

```
spectrum.security.authentication.webservice.basicauth.enabled=false
```

4. サーバーを開始します。

Web サービスの認証の無効化

Spectrum™ Technology Platform で使用されるすべてのサービスおよびリソースへのアクセスは、デフォルトで認証がオンになるように設定されています。

すべての SOAP または REST Web サービス (あるいはその両方) のサービス レベル認証を無効にすることができます。これは例えば、Location Intelligence モジュール サービスを使用しているソリューションに独自の高レベル認証を組み込む場合に役立ちます。

Spectrum™ Technology Platform 上の Web サービスの認証を無効にするには

1. Spectrum™ Technology Platform サーバーを停止します。
2. この設定ファイルをテキスト エディタで開きます。

   SpectrumLocation\server\app\conf\spectrum-container.properties

3. 各プロパティの値を必要に応じて変更します。例えば、すべての SOAP サービスの認証を無効にする場合は、次のようにします。

```
spectrum.security.authentication.webservice.enabled.REST=true
spectrum.security.authentication.webservice.enabled.SOAP=false
```

   注： Location Intelligence モジュールの場合、REST サービスには OGC Web サービスも含まれます。

4. プロパティ ファイルを保存して閉じます。
5. Spectrum™ Technology Platform サーバーを開始します。

以上の操作を完了すると、指定したタイプの Web サービスで認証がオフになります。

CORS の有効化

Cross-Origin Resource Sharing (CORS) は、ドメイン間のデータ共有を可能にする W3C 標準仕様です。CORS により、1 つのドメインで実行している Web アプリケーションが他のドメインからのデータにアクセスできるようになります。Spectrum™ Technology Platform サーバー上で CORS を有効にすることで、他のドメインでホストされている Web アプリケーションに Spectrum™ Technology Platform Web サービスへのアクセスを許可できます。

例えば、**webapp.example.com** でホストされている Web アプリケーションがあるとします。この Web アプリケーションには、**spectrum.example.com** でホストされている Spectrum™ Technology Platform Web サービスを呼び出す JavaScript 関数が含まれています。CORS がない場合は、プロキシ サーバーを使用してこの要求を処理する必要があり、それによって実装は複雑になります。CORS があれば、プロキシ サーバーは必要ありません。**webapp.example.com** を "allowed origin" (許可された生成元) として指定することで、Spectrum™ Technology Platform がドメイン **webapp.example.com** からの Web サービス要求に応答することを許可します。

Spectrum™ Technology Platform サーバー上で CORS を有効にするには

1. Spectrum™ Technology Platform サーバーを停止します。
2. 次のファイルをテキスト エディタで開きます。

   SpectrumLocation/server/app/conf/spectrum-advanced.properties

3. 次のパラメータを編集します。

   spectrum.jetty.cors.enabled

   このプロパティに true を設定して CORS を有効にします。デフォルトは false です。

   spectrum.jetty.cors.allowedOrigins

   Spectrum™ Technology Platform サーバー上のリソースへのアクセスを許可された生成元のカンマ区切りリスト。デフォルト値は http://localhost:8080,http://localhost:443 で、デフォルトの HTTP ポート 8080 とデフォルトの HTTPS ポート 443 を使用したリソースへのアクセスを許可します。

   例えば http://*.domain.com のように、許可された生成元に 1 つ以上のアスタリスク ("*") が含まれる場合、アスタリスクは .* に変換され、ドット文字 (".") は "\." とエスケープされて、その結果の許可された生成元が正規表現として解釈されます。つまり、許可された生成元として、より複雑な表現が使用できます。例えば https?://*.domain.[a-z]{3} は、http または https、複数のサブドメイン、そして任意の 3 文字のトップレベル ドメイン (.com、.net、.org など) にマッチします。

spectrum.jetty.cors.allowedMethods

Spectrum™ Technology Platform サーバー上のリソースにアクセスするときに使用できる HTTP メソッドのカンマ区切りリスト。デフォルト値は POST,GET,OPTIONS,PUT,DELETE,HEAD です。

spectrum.jetty.cors.allowedHeaders

Spectrum™ Technology Platform サーバー上のリソースにアクセスするときに使用できる HTTP ヘッダのカンマ区切りリスト。デフォルト値は X-PINGOTHER, Origin, X-Requested-With, Content-Type, Accept です。値がアスタリスク ("*") 1 つである場合、すべてのヘッダが使用可能です。

spectrum.jetty.cors.preflightMaxAge

クライアントが preflight 要求をキャッシュできる秒数。デフォルト値は 1800 秒、つまり 30 分です。

spectrum.jetty.cors.allowCredentials

リソースに対し、資格情報付きの要求が可能かどうかを示します。デフォルト値は true です。

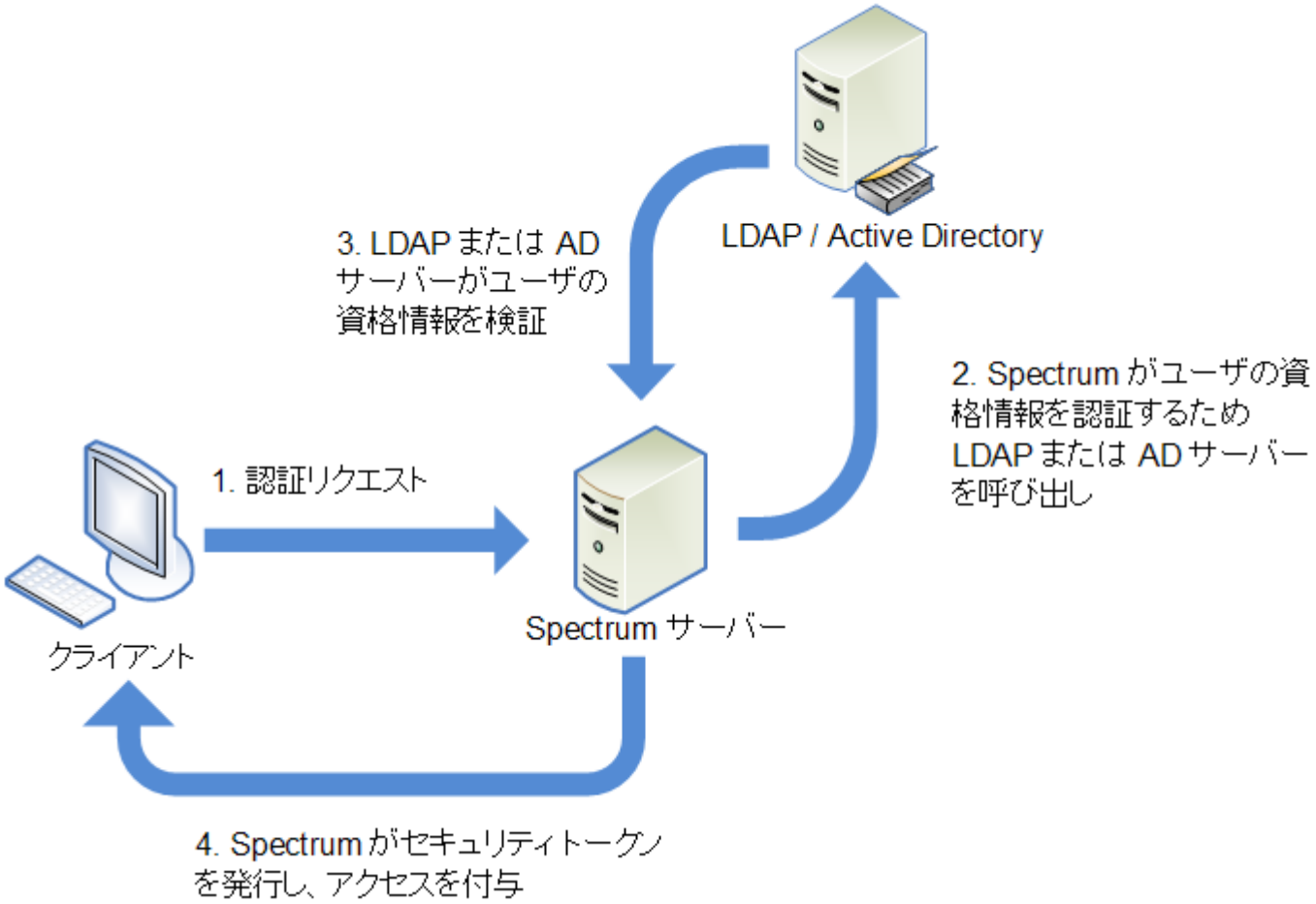
4. ファイルを保存して閉じます。
5. Spectrum™ Technology Platform サーバーを開始します。

LDAP または Active Directory による認証

Spectrum™ Technology Platform は、LDAP または Active Directory サーバーを認証に使うために設定できます。

ユーザが Spectrum™ Technology Platform にログインすると、LDAP または Active Directory を使ってユーザの資格情報が検証されます。その後で、同じ名前の Spectrum™ Technology Platform ユーザが存在するかどうかがシステムで確認されます。存在する場合は、ユーザがログインされます。存在しない場合は、そのユーザの Spectrum™ Technology Platform ユーザ アカウントが自動的に作成され、役割 `user` が付与されます。

次の図に認証プロセスを示します。



ディレクトリ サービスを使用して認証するように Spectrum™ Technology Platform を設定する前に、お使いのディレクトリ サービスが以下の要件を満たすことを確認してください。

- LDAP の場合、ディレクトリ サーバーは、LDAP バージョン 3 に準拠する必要があります。
- Active Directory サーバーに固有の要件はありません。

> **注：** この処理手順の説明については、Pitney Bowes テクニカル サポートまたはプロフェッショナル サービスにお問い合わせいただくことを推奨します。
>
> **注：** `spectrum.security.account.createNonExisting=true` を指定すると、Spectrum では、ユーザが最初にログインした際に、Active Directory ユーザが作成されます。そのプロパティに `false` を設定すると、Spectrum では、Spectrum™ Technology Platform 管理者がユーザを追加しない限り、こうしたユーザを認証できません。

1. Management Console で設定した既存のユーザが存在し、LDAP 認証または Active Directory 認証を有効にした後で、それらのユーザを使用する場合は、LDAP システムまたは Active Directory システムにそれらのユーザを作成します。Spectrum™ Technology Platform と同じユーザ名を使うようにしてください。

注： LDAP または Active Directory で "admin" ユーザを作成する必要はありません。このユーザは、LDAP または Active Directory を有効にした後も引き続き、認証に Spectrum™ Technology Platform を使用します。

2. Spectrum™ Technology Platform サーバーを停止します。
3. LDAP または Active Directory の認証を有効にします。
   a) この設定ファイルをテキスト エディターに開きます。
   `server\conf\spectrum-container.properties`
   b) プロパティ `spectrum.security.authentication.basic.authenticator` を LDAP に設定します。
   `spectrum.security.authentication.basic.authenticator=LDAP` LDAP を設定すると、LDAP だけでなく Active Directory も有効になります。
   c) ファイルを保存して閉じます。
4. 接続プロパティの設定:
   a) この設定ファイルをテキスト エディターに開きます。
   `server\conf\spring\security\spectrum-config-ldap.properties`
   b) 次のプロパティを変更します。

   spectrum.ldap.url

   LDAP サーバまたは Active Directory サーバの URL (ポートを含む)。例:
   `spectrum.ldap.url=ldap://ldapserver.example.com:389/`

   spectrum.ldap.dn.format

   LDAP または Active Directory のユーザ アカウントを検索するための書式です。ユーザ名には変数 `%s` を使用します。例を次に示します。

   LDAP:

   ```
   spectrum.ldap.dn.format=uid=%s,ou=users,dc=example,dc=com
   ```

   Active Directory:

   ```
   spectrum.ldap.dn.format=%s@example.com
   ```

   複数ドメイン:

   ドメイン名は単一引用符で囲み、各ドメイン値はカンマで区切ります。この方法は、単一ドメインの設定には適用されません。例:

   ```
   spectrum.ldap.dn.format='CN=%s,CN=Users,dc=spectest,dc=pvt',
   'CN=%s,ou=Services,dc=spectest,dc=pvt','CN=%s,ou=managers,dc=spectest,dc=pvt'
   ```

   spectrum.ldap.dn.base

LDAP または Active Directory のユーザ アカウントを検索するための識別名 (dn)。例を次に示します。

LDAP:

```
spectrum.ldap.dn.base=ou=users,dc=example,dc=com
```

Active Directory:

```
spectrum.ldap.dn.base=cn=Users,dc=example,dc=com
```

spectrum.ldap.search.filter

役割などの属性の検索に使用される検索フィルタ。検索フィルタには以下の変数を含めることができます。

- `{user}` : Spectrum™ Technology Platform にログインしているユーザ名
- `{dn}` は `spectrum.ldap.dn.base` で指定された識別名です。

LDAP の場合の例:

```
spectrum.ldap.search.filter=uid={user}
```

Active Directory の場合の例:

```
spectrum.ldap.search.filter=userPrincipalName={dn}
```

spectrum.ldap.attribute.roles

これはオプションです。ユーザの Spectrum™ Technology Platform 役割の名前が含まれる LDAP 属性または Active Directory 属性を指定します。LDAP 属性または Active Directory 属性で指定する役割名は、Spectrum™ Technology Platform で定義されている役割の名前と一致する必要があります。

この属性に `designer`という名前の役割が含まれる場合、`designer` 役割がユーザに付与されます。

指定できる属性は 1 つだけですが、属性に複数の役割が含まれることがあります。属性内の複数の役割を指定するには、それぞれの属性をカンマで区切ります。属性の各インスタンスに異なる役割が含まれる、複数値の属性を指定することもできます。この 1 つの属性で指定されている役割だけが Spectrum™ Technology Platform で使用されます。それ以外の LDAP 属性または Active Directory 属性は Spectrum™ Technology Platform の役割に影響を与えません。

Spectrum™ Technology Platform でユーザに役割が割り当てられている場合、ユーザの権限は、LDAP または Active Directory から付与される権限と Spectrum™ Technology Platform からの権限を結合したものになります。

例えば、属性 `spectrumroles` で定義されている役割を適用するには、次のように指定します。

```
spectrum.ldap.attribute.roles=spectrumroles
```

> **注：** 初めてログインするユーザが Spectrum™ Technology Platform ユーザアカウントを持っていない場合は、アカウントが自動的に作成され、役割 `user` が付与されます。このユーザの有効な権限は、`user` の役割と、`spectrum.ldap.attribute.roles` プロパティの属性リストで指定された役割に付与された権限の組み合わせです。
>
> **注：** Management Console でユーザの役割を表示した場合、`spectrum.ldap.attribute.roles` プロパティでユーザに割り当てられた役割は表示されません。

spectrum.ldap.pool.min

LDAP サーバーまたは Active Directory サーバーに接続するための接続プールの最小サイズ。

spectrum.ldap.pool.max

LDAP サーバーまたは Active Directory サーバーに同時に接続できる接続の最大数。

spectrum.ldap.timeout.connect

LDAP サーバーまたは Active Directory サーバーへの接続が確立されるまで待機する時間 (ミリ秒)。デフォルトでは、1000 ミリ秒に設定されます。

spectrum.ldap.timeout.response

接続が確立された後で、LDAP サーバーまたは Active Directory サーバーからの応答を待機する時間 (ミリ秒)。デフォルトでは、5000 ミリ秒に設定されます。

spectrum.ldap.retry.count

The number of times the Spectrum™ Technology Platform server will try connecting to the LDAP or Active Directory server if the initial connection attempt fails. Set this to 0 if you want to allow only one connection attempt.

ヒント： If you cluster your LDAP or Active Directory servers, we recommend that you set this value to 1 or more to allow the LDAP or Active Directory load balancer to redirect the connection request to a different server if the one that is initially tried is unavailable.

spectrum.ldap.retry.wait

接続試行から次の接続試行までの待ち時間 (ミリ秒)。

spectrum.ldap.retry.backoff

再試行が失敗するたびに待ち時間を増やすために使う乗算計数。例を次に示します。

```
spectrum.ldap.timeout.connect=1000 ...
spectrum.ldap.retry.count=5
spectrum.ldap.retry.wait=500
spectrum.ldap.backoff=2
```

この例では、最初の接続試行時の待ち時間は 1,000 ミリ秒で、5 回の再試行のたびに待ち時間が 2 倍ずつ増えるので、再試行ごとの待ち時間は次のようになります。

```
Retry attempt 1: 500 milliseconds
Retry attempt 2: 1,000 milliseconds
Retry attempt 3: 2,000 milliseconds
Retry attempt 4: 4,000 milliseconds
Retry attempt 5: 8,000 milliseconds
```

c) プロパティ ファイルを保存して閉じます。

5. Spectrum™ Technology Platform サーバーを開始します。

Spectrum™ Technology Platform をクラスタ内で実行している場合は、クラスタ内の各サーバー上で `spectrum-container.properties` ファイルと `spectrum-config-ldap.properties` ファイルを変更する必要があります。ファイルを変更する前にサーバーを停止し、ファイルの変更を終えてからサーバーを起動します。LDAP 属性値を役割にマッピングすると、このマッピングはクラスタ内のすべてのノードに複製されるため、このマッピング操作を JMX コンソールで繰り返す必要はありません。

LDAP 属性値を役割にマッピングする

属性からユーザの役割へのマッピングを有効にするには、プロパティ **spectrum.ldap.attribute.roles** を設定します。

この手順を実行する前に、LDAP 認証を有効にする必要があります。Spatial モジュールを使用している場合は、Jackrabbit 設定ファイルも変更する必要があります。詳細については、「**LDAP または Active Directory による認証**（58ページ）」を参照してください。

LDAP または Active Directory による認証を使用するように Spectrum™ Technology Platform を設定するには、ユーザに付与する役割を決定する値を持つ LDAP 属性を、設定プロパティの 1 つ (ファイル `spectrum-config-ldap.properties` 内の `spectrum.ldap.attribute.roles` プロパティ) で指定する必要があります。デフォルトで、この属性値は Spectrum™ Technology Platform の役割名に正確に一致する必要があります。そうでなければ、役割が付与されません。例えば、designer 役割を付与するには、属性に値 designer を設定する必要があります。

役割の割り当てに使用する LDAP 属性値が Spectrum™ Technology Platform の役割名に一致しない場合は、LDAP 属性値を役割名にマッピングすることができます。また、Spectrum™ Technology Platform の役割と同じ名前の LDAP 属性値を別の役割にマッピングすることもできます。例えば、組み込まれている役割の 1 つとして designer があります。designer という名前の LDAP 属性値があり、これを別の役割にマッピングしたい場合は、マッピングを作成することができます。

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. 次のプロパティをクリックします。

   com.pb.spectrum.platform.common.security.role:mappings=RoleMappings

   > **注：** このプロパティは、LDAP 認証を有効にし、サーバが完全に起動した後で表示されます。LDAP 認証を有効にしていない場合は、**LDAP または Active Directory による認証**（58ページ）を参照してください。

3. **addMapping** セクションの **ldapValue** フィールドに、Spectrum™ Technology Platform 役割にマッピングする LDAP 属性値を入力します。
4. **roleName** フィールドに、LDAP 属性にマッピングする Spectrum™ Technology Platform 役割を入力します。
5. **[Invoke]** をクリックします。

この LDAP 属性を持つユーザは、次回から Spectrum™ Technology Platform にログインすると、指定された役割が付与されます。

マッピングを削除するには、**removeMapping** セクションの **ldapValue** フィールドに、マッピングを解除する LDAP 属性を入力します。

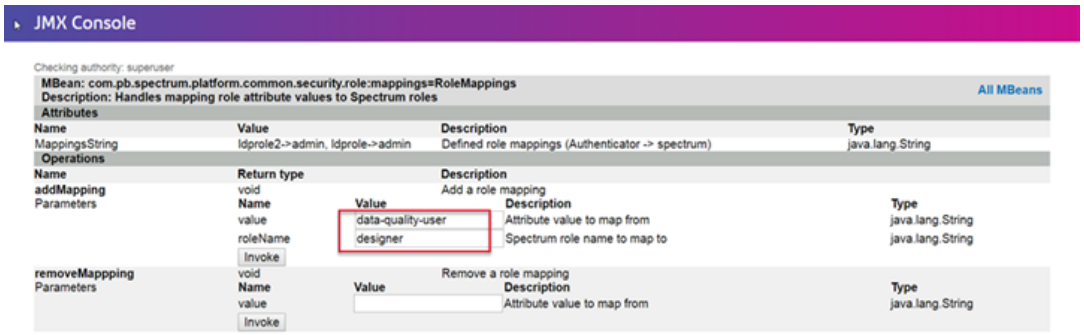
例

例えば、`gecos` 属性の値を使用して Spectrum™ Technology Platform の役割を割り当てるケースを想定します。`gecos` に値 data-quality-user が設定されている場合に、designer 役割を Spectrum™ Technology Platform へのログイン時にユーザに付与したいとします。

これを行うには、役割の割り当てに使用される属性として `gecos` 属性をファイル `spectrum-config-ldap.properties` 内で指定します。

```
spectrum.ldap.attribute.roles=gecos
```

次に、JMX コンソールで、data-quality-user 値を designer 役割にマッピングします。



これにより、`gecos` 属性の値が data-quality-user に設定されているユーザに役割 designer が付与されます。

LDAP SSO インストールでのユーザ ログイン資格情報の指定

LDAP SSO 環境で内部認証を定義できます。

LDAP SSO の `spectrum.security.authentication.internal.users` プロパティでは、外部の LDAP AD FS ユーザ アカウント リポジトリに対して認証するのではなく、Spectrum が内部で認証するユーザを定義します。**spectrum-container.properties ファイル**の `spectrum.security.authentication.internal.users` プロパティに明示的にユーザ名を追加する必要があります。例えば、"admin" ユーザを定義していない場合、このユーザは Spectrum™ Technology Platform サーバーにログインできません。

以下のように、1 人以上のユーザを追加します。
`spectrum.security.authentication.internal.users=`*`user1`*`,`*`user2`*`,`*`user3`*

"admin" ユーザがこのプロパティで定義されていない場合に admin としてログインを試みると、Spectrum は LDAP に対して認証を試みます (このユーザは存在しない可能性があります)。LDAP/AD FS を通した外部認証を強制的に適用するには、このプロパティを空白のままにします。

LDAP との SSL 通信の有効化

Spectrum™ Technology Platform と LDAP サーバーまたは Active Directory サーバーとの間の通信では、デフォルトで TCP が使用されます。

Spectrum™ Technology Platform サーバーと LDAP サーバーまたは Active Directory サーバーとの間の通信をセキュリティで保護する必要がある場合は、LDAP over SSL を使用するように Spectrum™ Technology Platform を設定できます。

以下の場合は、Spectrum™ Technology Platform で使用される Java TrustStore に証明書を追加しなければならない可能性があります。

- デフォルトの Java TrustStore に、使用している認証局のエントリが含まれていない。
- 自己署名証明書を使用している。自己署名証明書の使用は、実稼働環境では推奨されないことに注意してください。

上記のいずれかの条件にあてはまる場合は、以下の手順に従って Java TrustStore に証明書を追加します。

1. 証明書のコピーを入手します。証明書のコピーは LDAP 管理者から入手できます。また、LDAP Admin などのツールを使用して証明書を表示および保存しても入手できます。
2. JDK 付属の `keytool` ユーティリティを使用して、証明書を新規または既存の TrustStore に追加します。
   例:

   ```
   keytool -import -file X509_certificate_ldap.cer -alias
   server.example.com -keystore ldapTrustStore
   ```

   詳細については、ベンダーの Java ドキュメントを参照してください。

   > **注：** 証明書は、Spectrum™ Technology Platform で使用されている Java のバージョンに対応する暗号化と長さの要件を満たしている必要があります。Java のバージョンを調べるには、Management Console を開いて、**[システム] > [バージョン]** に移動します。詳細については、**java.com/en/jre-jdk-cryptoroadmap.html** を参照してください。

3. Spectrum™ Technology Platform サーバーを停止します。
   - Windows 上のサーバーを停止するには、Windows システム トレイの Spectrum™ Technology Platform アイコンを右クリックして、**[Spectrum™ を停止する]** を選択します。または、Windows の [コントロール パネル] の [サービス] を使用して、Pitney Bowes Spectrum™ Technology Platform サービスを停止できます。

- Unix または Linux 上のサーバーを停止するには、*SpectrumDirectory*/server/bin/setup スクリプトをソースとして、*SpectrumDirectory*/server/bin/server.stop スクリプトを実行します。

4. 次のファイルをテキスト エディタで開きます。

   SpectrumDirectory\server\conf\spring\security\spectrum-config-ldap.properties

5. 以下のプロパティを構成します。

   spectrum.ldap.url

   LDAP サーバーの URL を指定します。必ず SSL ポート番号を指定してください。番号は通常 636 です。例:

   ```
   spectrum.ldap.url=ldap://server.example.com:636
   ```

   注： URL の末尾にスラッシュ記号を追加しないでください。

   spectrum.ldap.useSSL

   true を指定して LDAP との SSL 通信を有効にします。

   spectrum.ldap.trustStore

   LDAP との SSL 通信に使用する証明書が格納されている TrustStore の場所を指定します。Windows の場合:

   ```
   spectrum.ldap.trustStore=file:D:\\Certs\\MyTrustStore
   ```

   Linux および Unix の場合:

   ```
   spectrum.ldap.trustStore=file://Certs//MyTrustStore
   ```

   spectrum.ldap.trustStore.password

   TrustStore パスワードを指定します。

重要： Spectrum™ Technology Platform をクラスタ内で実行している場合は、クラスタ内の各サーバーでこの手順を繰り返します。

LDAP との SSL 通信の無効化

LDAP や Active Directory との通信に SSL を使用するように Spectrum™ Technology Platform が設定されており、TCP を使用するように設定を戻す必要がある場合は、以下の手順に従います。

1. Spectrum™ Technology Platform サーバーを停止します。

- Windows 上のサーバーを停止するには、Windows システム トレイの Spectrum™ Technology Platform アイコンを右クリックして、**[Spectrum™ を停止する]** を選択します。または、Windows の [コントロール パネル] の [サービス] を使用して、Pitney Bowes Spectrum™ Technology Platform サービスを停止できます。
- Unix または Linux 上のサーバーを停止するには、*SpectrumDirectory*/server/bin/setup スクリプトをソースとして、*SpectrumDirectory*/server/bin/server.stop スクリプトを実行します。

2. 次のファイルをテキスト エディタで開きます。

   SpectrumDirectory\server\conf\spring\security\spectrum-config-ldap.properties

3. 以下のプロパティを構成します。

   spectrum.ldap.url

   SSL ポートの代わりに TCP ポートを使用するように LDAP サーバーの URL を変更します。デフォルト値は 389 です。例:

   ```
   spectrum.ldap.url=ldap://ldapserver.example.com:389/
   ```

   注 : URL の末尾にスラッシュを追加する必要があります。

   spectrum.ldap.useSSL

   false を指定して LDAP との SSL 通信を無効にします。

   spectrum.ldap.trustStore

   このプロパティをコメントアウトします。

   spectrum.ldap.trustStore.password

   このプロパティをコメントアウトします。

Spectrum のシングル サインオン (SSO) の実装

Spectrum™ Technology Platform では、Active Directory フェデレーション サービス (AD FS) および Ping Identity を利用したシングル サインオン (SSO) の機能のみを提供しています。

AD FS/Ping Identity IdP は、単一の Active Directory アカウントを通して、複数の Web アプリケーションに対する SSO 機能を実現します。SSO を利用することで、単一の資格情報のセットを使用して任意の Spectrum™ Technology Platform Web ベース サービスにアクセスできるようになり

ます。IdP では、Cookie ベースの認証を使用して、信頼できる当事者情報をシームレスに共有できます。

IdP 管理ツール (adfs.msc) は、Microsoft® 管理コンソール (MMC) のスナップインです。アカウントおよびリソース パートナーの追加、パートナー要求のマッピング、アカウント ストアの追加と設定、フェデレーション対応 Web アプリケーションの特定と設定に使用します。

システム要件

現在のシステム要件については、**サポート サイト**を参照してください。

Spectrum™ Technology Platform を使い慣れていない場合は、以下のトピックも確認してください。

- **HTTPS 通信の設定**（49ページ）
- **ネットワーク ポート**（10ページ）

環境設定の前提と SSO 展開のチェック

Spectrum SSO は、エンドユーザに対してシームレスに設計されています。こうしたシームレスな実装を容易にするために、SSO を実装する前に、特定のセキュリティ機能が設定されていることを確認します。

システム管理者と協力して、以下の確認を行います。

システム管理者がフェデレーション サーバーを展開している。	Microsoft® は、**フェデレーション サーバー展開**および**検証**に関するオンライン リファレンスを提供しています。
システム管理者が IdP サーバーの役割をインストールして設定している。	処理環境のために AD FS/Ping Identity のセットアップと設定が行われていることを確認します。AD FS には、この処理を支援する設定ウィザードが用意されています。
システムに認識済みのロード バランサーが含まれている。	Spectrum SSO が HTTP レベルで実装されている場合、HTTPS 通信は Spectrum クラスタ構成のロード バランサー レベルで終了する必要があります。
サーバー ホスト名を適宜変更する。	構成内の各クラスタは一意のホスト名 (コンピュータ名) を持ちます。ホスト マシンの命名に関するベスト プラクティス (名前に DNS を含めるなど) を使用して、ホスト マシンの識別と追跡を容易にします。

認証サポート用のサーバー設定

Spectrum™ Technology Platform では、IdP (Ping Identity または AD FS) によって SSO 認証をサポートするために特定のサーバー側プロパティおよびエンティティを定義する必要があります。

必要条件

このセクションで定義される設定のセットアップを行うには、Spectrum™ Technology Platform サーバーが HTTPS に対応している必要があります。詳細については、**HTTP または HTTPS の設定**（79ページ）を確認してください。

セキュリティ認証の設定

セキュリティ セットアップ プロセスでは、SSO 認証タイプの定義と JCE ポリシー ファイルの適用が必要です。

> **注：** すべての設定が適切に行われたことを確認した後に、これらのプロパティを設定します。詳細については、**HTTPS 通信の設定**（49ページ）を参照してください。

認証プロパティを設定するには、*SpectrumDirectory*`/server/conf/` の **spectrum-container.properties** ファイルを見つけて、プロパティ `spectrum.security.authentication.basic.authenticator=LDAP/SSO` を設定します。

このプロパティは、SSO に対して、ブラウザのバックグラウンド Web アプリケーションをリダイレクトするように指示します。Web サービスや CLI など、表示される Spectrum アプリケーションでは、STS が使用されます。

また、Spectrum™ Technology Platform サーバーで実行している Java のバージョンと同じバージョンの Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files をダウンロードして適用する必要があります。これは、IdP と Spectrum™ Technology Platform サーバーとの間でやり取りされるメッセージの暗号化と復号に役立ちます。ファイルを解凍し、JCE および *.jar ファイル (US_export_policy.jar、local_policy.jar) を Spectrum™ Technology Platform サーバーにインストールされている Java ファイルとともに *JavaLocation*`/jre/lib/security` の場所に配置します。

Spectrum™ Technology Platform では、Spectrum ソフトウェアをインストールする前に、現在のバージョンの Java のダウンロードおよびインストールが求められます。現在のバージョンの Java を適切なベンダーからダウンロードします。

サーバー認証プロパティを設定する

AD FS_STS または AD FS_SSO を認証で使用するように Spectrum™ Technology Platform を設定する場合は、SSO が SSL/TLS 経由で使用できることを確認してください。これは認証サーバーで必要になります。

SpectrumDirectory\conf\spring\security\ にある **spectrum-config-sso-sts.properties** を設定します。

このファイルには、LDAP/SSO-specific 特有のプロパティの設定値が含まれています。SSO に対応した認証では、SSO を有効にするためにこのプロパティを変更します。

`spectrum.security.authentication.web.sso.enabled=true`
このプロパティは、Web アプリケーション用のブラウザベース SSO と、その他のアプリケーション (クライアント サービスや Web サービスなど) 用の STS を有効にします。このプロパティを **false** に設定した場合は LDAP/SSO による認証となり、Basic STS がデフォルトの認証プロトコルになります。

キーストア設定プロパティを設定する

Spectrum™ Technology Platform サーバーは、IdP サーバーとのハンドシェイクを行う必要があり、秘密鍵と公開鍵のペアを必要とします。

信頼キーストア、キー、およびキーストア パスワードを設定します。Spectrum™ Technology Platform ホスト キーストアの場合、以下の値を設定します。

- `spectrum.sso.sp.keystorePath=`*`KeyStorePath`*例:

  ```
  c:/saml2/AD FS/AD FS-java/tomcat-ssl.keystore
  ```

セキュリティ プロパティ

- `spectrum.sso.sp.keystorePassword=`*`KeyStorePassword`*
- `spectrum.sso.sp.privateKeyPassword=`*`PrivateKeyPassword`*
- `spectrum.sso.sp.alias=`*`KeyAliasUsedWhileCreatingCert`*

IdP セッション タイムアウト管理のプロパティ

IdP には独自のセッション管理プロパティがあります。また、Spectrum™ Technology Platform のセッション タイムアウトを分離して制御することもできます。ベスト プラクティスとして、これら双方のプロパティを同じタイムアウト値によって定義することをお勧めします。

このプロパティでは、非アクティブな状態が指定時間続いた場合のセッション タイムアウトを `spectrum.sso.IdP.maximumAuthenticationLifetime=1800` のように設定します。

Spectrum サーバーのセッション タイムアウト プロパティの設定が SSO のプロパティよりも優先することに注意してください。Spectrum サーバーのセッション タイムアウトの設定は、

spectrum-container.properties ファイルのプロパティ `spectrum.security.authentication.session.timeout=`*`seconds`* にあります。ここで、*seconds* はセッションが終了するまでの秒数を示します。

SAML2 アサーションの設定

SAML2 アサーションを使用するには、サイトで使用する IdP の SAML メタデータをダウンロードし、リクエストを生成するためにローカルに保存する必要があります。

この SAML メタデータ (XML) により、Spectrum™ Technology Platform からの SAML ログインおよびログアウト リクエストが生成されます。
`spectrum.sso.IdP.identityProviderMetadataPath=`*`LocalPath`*`/ADFSv2.0-FederationMetadata.xml`

サービス プロバイダは独自の SAML2 データを生成し、これにより Spectrum™ Technology Platform がサービス プロバイダとして適切に設定されていることを確認できます。
`spectrum.sso.sp.serviceProviderMetadataPath=`*`localpath`*`/SP-FederationMetadata.xml`

IdP は、リライング パーティを必要とします。通常、これは、サービス プロバイダ情報です。Spectrum™ Technology Platform は、IdP によって認識される識別子を生成する必要があります。この情報は SAML リクエストに追加され、Spectrum Technology Platform から IdP に送信されます。IdP には既に識別子 `spectrum.sso.sp.serviceProviderEntityId=`*`YourIdentifierForRelyingParty`*`.` が設定されています。これは IdP への信頼されたリクエストの確認に役立ちます。例: **`https://US-5H19PH2-10.pbi.global.pvt/AD FS/trust`**

SSO バインディングのプロパティの設定

バインディングでは、アーティファクト解決プロトコルを使用して、SAML ID が指定されたメッセージを参照によって解決します。

クライアントのバインド/トランスポート認証を設定するには、*`SpectrumDirectory`*`\conf\spring\security` にある **spectrum-config-sso-sts.properties** を見つけます。プロパティ `spectrum.sso.idp.destinationBindingType=`*`Property for the binding type configuration`* のバインディング タイプを設定します。次に例を示します。

```
spectrum.sso.idp.destinationBindingType=urn:oasis:names:tc:SAML:
2.0:bindings:HTTP-Redirect
spectrum.saml.sts.idp.type=ADFS
```

表 1 : バインディング タイプ

バインディング タイプ	定義
REDIRECT	SAML プロトコル メッセージを URL パラメータ内で送信できるようにします。このバインディングは、SAML リクエスタとレスポンダが HTTP ユーザ エージェントを仲介として使用して通信する必要があるものの、両者の間に直接パスがない場合に使用されます。
POST	SAML プロトコル メッセージを HTML フォーム コントロールの Base64 にエンコードされたコンテンツ内で送信できるようにします。REDIRECT バインディングと同様に、POST は SAML リクエスタとレスポンダが HTTP ユーザ エージェントを仲介として使用して通信する必要があるものの、両者の間に直接パスがない場合に使用されます。さらに、このバインディングは、レスポンダがユーザとの対話 (認証など) を要求する場合も必要になることがあります。
Artifact	アーティファクトと呼ばれる小さな代用品を使用して、SAML リクエスト、SAML レスポンス、またはその両方を参照として送信できるようにします。HTTP アーティファクト バインディングは、SAML リクエスタとレスポンダが HTTP ユーザ エージェントを仲介として使用して通信する必要があるものの、メッセージ全体が通過することを仲介が受け付けないか許可できない場合に使用されます。

SAML バインディングの詳細については、こちらのリソースを参照してください。

クラスタ構成での SSO

クラスタ環境での SSO セットアップの一環として、最初にクラスタ化設定を用意する必要があります。

適切なクラスタ設定を確実に行うには

- IdP を使用するクラスタのセットアップで SSO を利用するには、ロードバランサーが HTTPS 対応でなければなりません。
- すべてのノードおよびロード バランサーのホスト ファイルでドメイン エントリを定義します。これにより、ドメインと各ノードの IP アドレスがマッピングされ、Spectrum SSO で認識されるようになります。例えば、3 つのノードがあるクラスタ構成では、次のように定義します。

```
node1IP hostname
node2IP hostname
node3IP hostname
ADFSIP hostname
loadBalancerIP hostname
```

- 詳細については、クラスタのセットアップに関する Spectrum™ Technology Platform のドキュメントを確認してください (クラスタ アーキテクチャ（542ページ）)。

役割とプロパティの管理およびマッピング

Spectrum SSO は、管理者が割り当てた資格情報にユーザ アカウントをマッピングします。

Spectrum™ Technology Platform により、LDAP/SSO 資格情報を使用してログインするユーザに、適切な共有が許可されます。役割マッピングを削除するには、JMX コンソールの **removeMapping** セクションの **value** フィールドに、マッピングを解除する LDAP 属性を入力します。

適切な資格情報と権限によってユーザが Spectrum™ Technology Platform に定義されていることを確認してください。プロパティ設定が `spectrum.security.account.createNonExisting=False` になっているユーザがいる場合、そのユーザは認識されず、SSO で認証されません。ユーザ名は、システム管理者が手動で作成する必要があります。外部認証リポジトリに存在しないユーザは、たとえ Spectrum Management Console で手動で作成されたユーザでも Spectrum にログインできません。外部認証リポジトリ内に作成されれば、ユーザは Spectrum にログインできるようになります。

管理者の役割のセットアップ

ユーザに admin の役割をマッピングできます。マップ済みの admin レベルのユーザは、Spectrum admin レベルのユーザと同じ権限を持ちますが、基本的なユーザ役割権限を持つ非 admin ユーザとして表示されます。

Management Console の [セキュリティ] ページでは、ユーザの権限を編集して、正しい権限を表示できます。Spectrum SSO 実装では、デフォルトの管理者の共有とユーザの役割が自動的に適用されることはありません。ユーザの役割の権限を適用し、表示するには、*ドメイン ユーザ グループ*にマッピングされたユーザにプロパティを設定する必要があります。

管理者 ("Admin") のアクセス プロファイルなど、システム全体のアクセス プロファイルを確立するには:

1. *`SpectrumDirectory`*`\..\server\conf\spring\security` 内にある **spectrum-config-sso-sts.properties** に移動します。
2. **spectrum-config-sso-sts.properties** ファイル内で、admin グループの権限を Spectrum サーバーの起動時に適用する動的なプロパティを設定するために、`spectrum.security.authentication.idpserver.admin.role=`***rolename*** *と指定します。ここで、****rolename*** はシステムレベルの管理者権限を継承するユーザのグループ名です。
3. JMX コンソールにログインし、プロパティ **com.pb.spectrum.platform.common.security.role:mappings=RoleMappings** を検索します。

   このプロパティは、すべてのユーザ グループの役割のマッピングを管理します。
4. 次のパラメータを定義します。

1. `addMapping` セクションの `value` フィールドに、Spectrum™ Technology Platform 役割にマッピングする SSO 役割値を入力します。
2. `roleName` フィールドに、LDAP 属性にマッピングする Spectrum™ Technology Platform 役割を入力します。
3. [`Invoke`] をクリックします。SSO 役割を持つユーザが Spectrum™ Technology Platform に 1 回でもログインすると、指定した役割がそのユーザに付与されます。
4. マッピングを削除するには、`removeMapping` セクションの **value** フィールドに、マッピングを解除する LDAP 属性を入力します。

管理者の役割の割り当て

管理者の役割をマッピングするための最後のステップでは、管理者の役割を特定のユーザまたはユーザ グループに割り当てます。

1. *`SpectrumDirectory`*`/server/conf/spring/security` にある **spectrum-config-sso-sts.properties** で次のプロパティを更新します。
   `spectrum.security.authentication.IdPserver.admin.role=`*`GroupName`*`.`
2. "Domain Users" など、Spectrum™ Technology Platform 管理者の役割を必要とする *GroupName* を指定します。
3. 割り当てたグループ名に属するユーザとしてログインし、他のユーザの役割を設定します。**[セキュリティ] > [ユーザ] > [役割]** に移動するか、「**Mapping LDAP/SSO roles to Spectrum Technology Platform roles**（75ページ）」で説明する役割マッピング プロセスを使用します。

Mapping LDAP/SSO roles to Spectrum™ Technology Platform roles

Before mapping roles, ensure that you have enabled LDAP/SSO authentication.

> **注：** We have verified identity providers AD FS and Ping Identity for Spectrum™ Technology Platform.

When you configure Spectrum™ Technology Platform to use LDAP/SSO for authentication, by default, the role values must match the Spectrum™ Technology Platform role names, exactly in order, to grant the role. For example, to grant the designer role, the role you specify must be "designer."

> **注：** If you are using the Spatial Module, you must also update the Jackrabbit configuration file. For more information see **LDAP または Active Directory による認証**（58ページ）.

You can map non-matching LDAP/SSO role values to an existing Spectrum™ Technology Platform role name. You can also map an LDAP/SSO role value with the same name as a Spectrum™ Technology Platform role to a different role. For example, one of the built-in roles is "designer." If

you have an LDAP/SSO role value that is also named "designer," but you want it to map to another role, you could create a role map.

To map an LDAP/SSO role value to an existing Spectrum role:

1. Open a Web browser and go to `http://server:port/jmx-console`, where:
   - *server* is the IP address or host name of your Spectrum™ Technology Platform server.
   - *port* is the HTTP port used by Spectrum™ Technology Platform. The default is 8080.
2. Select this property:
   com.pb.spectrum.platform.common.security.role:mappings=RoleMappings
   This property is visible only when you enable LDAP or LDAP/SSO authentication, and the Spectrum™ Technology Platform server is fully started.
3. In the **addMapping** section, configure these settings:
   a) In the **value** field, enter the LDAP/SSO role value to map to a Spectrum™ Technology Platform role.
   b) In the **roleName** field, enter the Spectrum™ Technology Platform role to map to the LDAP attribute value.
4. Click **Invoke**.

Users who have been assigned an LDAP/SSO role will now be granted the role you specified for them the next time they log in to Spectrum™ Technology Platform.

To remove a mapping, enter the LDAP attribute you want to unmap in the **value** field in the **removeMapping** section in JMX console.

暗号

証明書ベースの暗号化

Spectrum™ Technology Platform の証明書ベースの暗号化では、特定のセキュリティ、信頼、コミュニケーション ツールを設定する必要があります。

> **注:** 暗号化の設定を初めて行う前に、さまざまな**暗号化方式**（77ページ）など、Spectrum™ Technology Platform の暗号化に関するすべてのドキュメントを確認することをお勧めします。

デフォルトでは、HTTP/HTTPS、インデックス作成、キャッシュなど、サーバーのすべての部分で暗号化が無効になっています。グローバル設定によってサーバーのすべての部分で暗号化を有

効にするためのプロパティ `spectrum.encryption.enabled=true` を使用して、**spectrum-container.properties** で暗号化を設定します。

ブラウザおよび API 通信用の HTTPS など、Spectrum™ Technology Platform の一部だけで暗号化をセットアップする場合は、`spectrum.encryption.enabled=false` となっていることを確認します。

信頼証明書の定義

Spectrum では、すべてのレベルの暗号化について信頼証明書を定義し、その証明書を *SpectrumDirectory*`/server/conf/certs` に格納する必要があります。

キーストアと信頼ストアをセットアップする

信頼できる証明書が格納される信頼ストアと、信頼証明書の秘密鍵コンポーネントを格納するキーストアを定義します。

暗号化を有効にするには、信頼ストアを定義する必要があります。この信頼ストアにより、信頼できる証明書と、その秘密鍵コンポーネントを格納するためのキーストアが管理されます。デフォルトでは、Spectrum によって自己署名暗号化証明書が提供されます。

- `keystore.p12` – この pkcs12 キーストアには、通信用の証明書チェーンが格納されます。
- `truststore.p12` – この pkcs12 キーストアには、ルート認証局 (CA) の公開証明書が格納されます。

  注： これらの証明書は、実稼働用には推奨されません。

各キーストアは *SpectrumDirectory*`/server/conf/certs` フォルダ内に配置する必要があります。

暗号化方式

このセクションでは、暗号化方式と各方式の設定およびプロパティについて説明します。

ご利用のサイトで暗号化をセットアップする前に、使用可能なすべての暗号化方式を確認しておくことをお勧めします。

- **方法 1: ユーザが提供する CA 証明書を受け入れるように Spectrum を設定する**（78ページ）
- **方法 2: Spectrum に、Pitney Bowes が提供する自己署名証明書を設定する**（78ページ）
- **方法 3: Spectrum に、独自の自己署名証明書を設定する**（78ページ）
- **個別の設定**（79ページ）

方法 1: ユーザが提供する CA 証明書を受け入れるように Spectrum を設定する

この設定方法では、認証局 (CA) に登録されているユーザ指定の証明書が受け入れられます。

最高レベルのセキュリティを実現できるため、この方法を使用することをお勧めします。この設定では、以下に示すように、同じタイプ (ノードまたはクライアント) のすべてのノードが、一致する DN が指定された証明書を持っていることが必要です。

1. キーストアと信頼ストアをセットアップする（77ページ）。また、これらを *SpectrumDirectory*/server/conf/certs フォルダにコピーします。
2. サーバーのインストール場所で暗号化設定を行います。

   - spectrum.encryption.enabled=true
   - spectrum.encryption.algorithm=JASYPT

   - spectrum.encryption.selfSignedCert=false
   - spectrum.encryption.trustAllHosts=false
   - spectrum.encryption.keystoreType=pkcs12 or jks
   - spectrum.encryption.keystore=node-keystore.p12
   - spectrum.encryption.keystorePassword=*password*
   - spectrum.encryption.keystoreAlias=*keystore alias if one applies*
   - spectrum.encryption.truststoreType=pkcs12 or jks
   - spectrum.encryption.truststore=truststore.p12
   - spectrum.encryption.truststorePassword=*truststore password*

方法 2: Spectrum に、Pitney Bowes が提供する自己署名証明書を設定する

このトピックでは、Pitney Bowes からの自己署名証明書を実装する手順を示します。

> **注：** この設定を実稼働環境で使用することは**お勧めしません**。

1. Spectrum サーバーを停止します。
2. spectrum-container.properties で以下のプロパティを変更します。

   - spectrum.encryption.enabled=true
   - spectrum.encryption.algorithm=JASYPT
   - spectrum.encryption.selfSignedCert=true
   - spectrum.encryption.trustAllHosts=true

3. Spectrum サーバーを起動します。

方法 3: Spectrum に、独自の自己署名証明書を設定する

この設定を実稼働環境で使用することはお勧めしません。

1. Spectrum サーバーを停止します。

2. キーストアと信頼ストアを作成し、*SpectrumDirectory*/server/conf/certs フォルダにコピーします。必ずこの場所を使用してください。
3. 次のサーバーの場所で暗号化設定を行います。 *SpectrumDirectory*/server/conf
4. **spectrum-container.properties** で以下のプロパティを変更します。
   - spectrum.encryption.enabled=true
   - spectrum.encryption.algorithm=JASYPT
   - spectrum.encryption.selfSignedCert=true
   - spectrum.encryption.trustAllHosts=true

     注： Set spectrum.encryption.trustAllHosts to true only if a single certificate will be used across multiple hosts.

   - spectrum.encryption.keystoreType=pkcs12 or jks
   - spectrum.encryption.keystore=node-keystore.p12
   - spectrum.encryption.keystorePassword=*keystorepassword*
   - spectrum.encryption.keystoreAlias=*keystore alias, if one applies*
   - spectrum.encryption.truststoreType= pkcs12 or jks
   - spectrum.encryption.truststore=truststore.p12
   - spectrum.encryption.truststorePassword=*truststorepassword*
5. Spectrum サーバーを起動します。

個別の設定

このセクションで説明した設定により、Spectrum™ Technology Platform の各部分の暗号化プロトコル、キャッシュ、インデックス作成を個別に設定できます。

暗号化セットアップを成功させる確固たる基盤を構築するために、**証明書ベースの暗号化**（76ページ） を確認しておくことを強くお勧めします。留意事項: Spectrum™ Technology Platform の特定の部分を設定すると、グローバル プロパティ設定がオーバーライドされます。

HTTP または HTTPS の設定

spectrum-container.properties の **Spectrum http settings** セクションにある設定を使用すると、HTTP または HTTPS の設定を行うことができます。

これらの設定では、こうしたプロトコルの両方または一方を有効または無効にできます。HTTP と HTTPS の両方が有効になっている場合、**spectrum.http.default.protocol** プロパティにより、Spectrum は適切なプロトコルを適用して内部通信を使用できます。

注： HTTP はデフォルトで有効になっています。HTTPS はデフォルトでは有効になっていません。

HTTPS または HTTP 設定では、キーストアまたは信頼ストアを定義します。

キャッシュの設定

ブローカ エンティティおよびクラスタリング プロパティを細かく指定できます。

キャッシュおよびリモート ノード間呼び出しを設定するには、以下のプロパティを使用します。特定の **キャッシュ設定**（84ページ） を確認することもできます。

ブローカ設定

これらの設定は、ブローカ プロパティを制御するもので、**spectrum-container.properties** の "Spectrum broker settings" (Spectrum ブローカ設定) セクションにあります。

プロパティ	説明
`spectrum.broker.name`	ブローカ エンティティの名前 (例: SpectrumCluster-Broker)
`spectrum.broker.port`	ブローカが使用するポート (デフォルト値は 5710)。
`spectrum.broker.password`	Hazelcast ブローカのパスワード。詳細については、 • 「**暗号化文字列の生成**（88ページ）」を参照してください。 • 「**パスワードの暗号化または暗号化文字列のマスク**（89ページ）」を参照してください。
`spectrum.broker.addresses`	クラスタのすべてのメンバーの IP アドレス。コメントアウトされている場合は、デフォルトで `spectrum.cluster.seeds` の値に設定されます。
`spectrum.broker.seeds`	暗号化の共有ソースとして使用される IP アドレス。通常、この値は `spectrum.broker.addresses` の値に一致します。 **注:** この値をコメントアウトすると、デフォルトで `spectrum.broker.address` の値に設定されます。`spectrum.broker.addresses` がコメントアウトされている場合は、デフォルトで `spectrum.cluster.seeds` の値に設定されます。

クラスタ設定

これらの設定は、キャッシュ クラスタ設定を定義するもので、**spectrum-container.properties** の "Spectrum cluster settings" (Spectrum クラスタ設定) セクションにあります。

プロパティ	説明
`spectrum.cluster.enabled`	クラスタ キャッシュを有効化/無効化。true は有効、false (デフォルト) は無効を示す
`spectrum.cluster.password`	Hazelcast キャッシュ クラスタの名前。詳細については、 • 「**暗号化文字列の生成**（88ページ）」を参照してください。 • 「**パスワードの暗号化または暗号化文字列のマスク**（89ページ）」を参照してください。
`spectrum.cluster.seeds`	クラスタの各メンバーの IP アドレスを示すカンマ区切りリスト
`spectrum.cluster.port`	キャッシュ クラスタが使用するポート (デフォルト値は 5701)。
`spectrum.cluster.nodeId`	クラスタ内の各ノードについて、一意のノード ID を定義する

インデックスの設定 - Elasticsearch

Elasticsearch のインデックス作成プロパティを具体的に設定できます。

Elasticsearch のプロパティを設定するには、**インデックスの設定 - Elasticsearch**（81ページ）で定義されているプロパティを使用します。

CLI 暗号化の設定 - Windows クライアントのみ

以下の手順は、暗号化定義を適用できるテンプレートとなります。

以下のテンプレート手順を pflowexecutor、enableadmin ユーティリティ、および管理ユーティリティの暗号化定義に適用します。その場合、プロパティ ファイルの名前はそれぞれ **pflowexecutor.properties**、**enableadmin.properties**、**cli.properties** となります。

CLI プロパティ ファイルは、CLI コンポーネントの実行可能ファイルと同じディレクトリにあります。例えば、jobexecutor.jar が `C:\Users\myUser\cliClients\jobexecutor` にある場合は、プロパティ ファイルを `jobexecutor` フォルダに配置します。

jobexecutor

jobexecutor の場合は、**jobexecutor.properties** というプロパティ ファイルを作成します。この例では、サーバーの certs フォルダにある Spectrum 自己署名証明書 (`node-keystore.p12` および `node-keystore/truststore.p12`) のコピーが必要になります。この 2 つのファイルをローカル ディレクトリ (`C:\myKeys` など) にコピーします。

```
# sample properties when using  a Spectrum self-signed cert
spectrum.encryption.algorithm=JASYPT
spectrum.encryption.keystoreType=pkcs12
spectrum.encryption.keystore=C:\\myKeys\\node-keystore.p12
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.keystoreAlias=spectrum
spectrum.encryption.truststoreType=pkcs12
spectrum.encryption.truststore=C:\\myKeys\\truststore.p12
spectrum.encryption.truststorePassword=p*********s
spectrum.encryption.truststoreAlias=spectrum
spectrum.encryption.trustAllHosts=true
spectrum.encryption.trustSelfSigned=true
```

enableadmin

SSL を有効にして enableadmin を使用するには、jobexecutor で使用したのと同様のプロパティ ファイル、**enableadmin.properties** を作成する必要があります。Pitney Bowes では、`server/conf/certs` フォルダにある証明書を参照するこのファイルを `server/bin` で提供しています。

プロパティは以下のとおりです。

```
# enable admin account properties
spectrum.encryption.algorithm=JASYPT
spectrum.encryption.keystoreType=pkcs12
spectrum.encryption.keystore=../conf/certs/client-keystore.p12
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.keystoreAlias=spectrum-client
spectrum.encryption.truststoreType=pkcs12
spectrum.encryption.truststore=../conf/certs/truststore.p12
spectrum.encryption.truststorePassword=p*********s
```

```
spectrum.encryption.trustAllHosts=true
spectrum.encryption.trustSelfSigned=true
```

暗号化のプロパティ

このリファレンスでは、**spectrum-container.properties** にある、グローバル暗号化プロパティとサーバーの特定の部分の暗号化プロパティの一覧および説明を示します。

グローバル暗号化設定

グローバル暗号化設定は、HTTP、HTTPS、キャッシュ、インデックスのすべてのレベルに適用されます。レベル固有のプロパティを使用すると、特定のレベルの優先設定を定義できます。

プロパティ	説明
`spectrum.encryption.enabled`	Basic HTTP の有効または無効: true は有効、false (デフォルト) は無効を示す **注：** Spectrum 暗号化では、このプロパティが false に設定されていてもグローバル暗号化設定が評価および適用され、**インデックス作成設定**（86ページ） が明示的に適用されない限り Elasticsearch インデックス作成を使用できません。
`spectrum.encryption.algorithm`	リソース パスワードに使用する暗号化アルゴリズム: JASYPT (デフォルト) または AES
`spectrum.encryption.keystoreAlias`	証明書のエイリアス (該当する場合)、または見つかった最初のキーを使用 (`spectrum` など)
`spectrum.encryption.keystoreType`	キーストアのタイプ: pkcs12 (デフォルト) または jks
`spectrum.encryption.keystore`	場所におけるキーストアのファイル名 *`SpectrumDirectory`*`/conf/certs`
`spectrum.encryption.keystorePassword`	キーストア パスワード。詳細については、 • 「**暗号化文字列の生成**（88ページ）」を参照 • 「**パスワードの暗号化または暗号化文字列のマスク**（89ページ）」を参照
`spectrum.encryption.selfSignedCert`	証明書は自己署名か? true または false

プロパティ	説明
spectrum.encryption.truststoreType	信頼ストアのタイプ: pkcs12 または jks
spectrum.encryption.truststore	場所における信頼ストアのファイル名 *SpectrumDirectory*/server/conf/certs
spectrum.encryption.truststorePassword	信頼ストアのパスワード。詳細については、 • 「暗号化文字列の生成（88ページ）」を参照 • 「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を参照
spectrum.encryption.validateCerts	証明書の検証が必要か? true (デフォルト) または false
spectrum.encryption.trustAllHosts	検証時、証明書で指定されているホスト名を無視します。

キャッシュ設定

これらの定義は、キャッシュ設定を制御し、**spectrum-container.properties** の **Cache settings (Hazelcast)** セクションにあります。

プロパティ	説明
spectrum.cache.encryption.keystoreType	キーストアのタイプ: pkcs12 または jks
spectrum.cache.encryption.keystore	キーストアのファイル名: *SpectrumDirectory*/server/conf/certs
spectrum.cache.encryption.keystorePassword	キーストア パスワード。詳細については、 • 暗号化文字列の生成については、「暗号化文字列の生成（88ページ）」を参照してください。 • 文字列の暗号化およびマスキングの詳細については、「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を確認してください。
spectrum.cache.encryption.truststoreType	信頼ストアのタイプ: pkcs12 または jks
spectrum.cache.encryption.truststore	信頼ストアのファイル名: *SpectrumDirectory*/server/conf/certs

プロパティ	説明
spectrum.cache.encryption.truststorePassword	信頼ストアのパスワード。詳細については、 • 暗号化文字列の生成については、「暗号化文字列の生成（88ページ）」を参照してください。 • 文字列の暗号化およびマスキングについては、「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を確認してください。

HTTPS および HTTP 設定

これらの定義は、HTTP および HTTPS プロパティの設定を制御し、**spectrum-container.properties** の "Spectrum http settings" セクションにあります。

プロパティ	説明
spectrum.http.enabled	Basic HTTP の有効/無効
spectrum.http.port	HTTP ポート
spectrum.https.enabled	Basic HTTPS の有効/無効: true または false
spectrum.https.port	HTTPS ポート
spectrum.https.encryption.validateCerts	証明書の検証が必要か?
spectrum.https.encryption.trustAllHosts	キーストアも信頼ストアも提供されていない場合に、すべての証明書を信頼するか?
spectrum.https.encryption.selfSignedCert	証明書は自己署名か?
spectrum.https.encryption.trustAllHosts	ホスト名の検証は無効か?
spectrum.https.encryption.keystoreType	キーストアのタイプ: pkcs12 または jks
spectrum.https.encryption.keystore	キーストアのファイル名: *SpectrumDirectory*/server/conf/certs

プロパティ	説明
spectrum.https.encryption.keystorePassword	キーストア パスワード。詳細についての参照先: • 「暗号化文字列の生成（88ページ）」を参照してください。 • 「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を参照してください。
spectrum.https.encryption.keystoreAlias	証明書のエイリアス(該当する場合)、または見つかった最初のキーを使用
spectrum.https.encryption.truststoreType	信頼ストアのタイプ: pkcs12 または jks
spectrum.https.encryption.truststore	信頼ストアのファイル名: *SpectrumDirectory*/server/conf/certs
spectrum.https.encryption.truststorePassword	信頼ストアのパスワード。詳細については、 • 「暗号化文字列の生成（88ページ）」を参照してください。 • 「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を参照してください。

インデックス作成設定

これらの定義は、インデックス作成設定を制御し、**spectrum-container.properties** の "Index settings (Elasticsearch)" セクションにあります。

プロパティ	説明
spectrum.index.encryption.enabled	インデックス作成の暗号化の有効/無効: true または false
spectrum.index.encryption.trustAllHosts	ホスト名の検証は無効か?
spectrum.index.encryption.keystoreType	キーストアのタイプ: pkcs12 または jks
spectrum.index.encryption.keystoreAlias	証明書のエイリアス(該当する場合)、または見つかった最初のキーを使用

プロパティ	説明
spectrum.index.encryption.keystore	インデックス キーストア名 *SpectrumDirectory*/server/conf/certs
spectrum.index.encryption.keystorePassword	*SpectrumDirectory*/server/conf/certs におけるインデックス キーストア パスワード。詳細についての参照先: • 「暗号化文字列の生成（88ページ）」を参照してください。 • 「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を参照してください。
spectrum.index.encryption.truststoreType	インデックス信頼ストアのタイプ: pkcs12 または jks
spectrum.index.encryption.truststore	インデックス キーストア名 *SpectrumDirectory*/server/conf/certs
spectrum.index.encryption.truststorePassword	インデックス信頼ストアのパスワード。詳細についての参照先: • 「暗号化文字列の生成（88ページ）」を参照してください。 • 「パスワードの暗号化または暗号化文字列のマスク（89ページ）」を参照してください。

パスワード アルゴリズム設定

この定義は、パスワードレベルの暗号化設定を制御し、**spectrum-container.properties** にあります。

プロパティ	説明
spectrum.password.decryption.algorithm	パスワードの復号化に使用する暗号化アルゴリズム: JASYPT (デフォルト) または AES

暗号化文字列の生成

暗号化文字列の設定方法は 2 つあります。

方法 1

`*.jar` ファイル ユーティリティの *`SpectrumDirectory`*`/server/bin/password-utility.jar` を使用して、Spectrum デフォルト パスワードの暗号化文字列を生成します。

> **注：** 同じパスワードの暗号化を複数回実行すると、そのたびに異なる文字列が生成されます。これにより、暗号化の強度とセキュリティが高まります。

デフォルトのパスワード p****s に対する暗号化文字列を生成します。

コマンドを指定します。

```
java -jar password-utility.jar -p password -a algorithm
```

ここで、

- *password* はご利用のサイトのパスワードです
- *algorithm* は暗号化方式 (`AES` または `PBEWithMD5ANdDeS`) です。

サンプル出力:

```
############################################################
############ Encrypted String for the password ############
############################################################
            9yOYoZ9W2aAF2Baapa5wIxCMNQ/9TZFP
```

方式 2

`encryptString` プロパティを使用して JMX コンソールから暗号化文字列を生成することもできます。このリクエストは `MBean operation: invoke method encryptString on MBean` *`servername`*`:manager=EncryptTextManager` という形式をとります。

パスワードの暗号化または暗号化文字列のマスク

機密情報がログ ファイルや表示でエクスポーズされないように、パスワードを暗号化して暗号化文字列をマスクすることができます。

パスワードの暗号化または暗号化文字列のマスクを行うには、次のプロパティ ファイルを更新し、*p*********s* を暗号化文字列に置き換えます。

```
SpectrumDirectory..\Spectrum\server\conf\spectrum-container.properties
spectrum.encryption.algorithm=JASYPT
spectrum.broker.password=p*********s
spectrum.cluster.password=p*********s
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.truststorePassword= p*********s
#spectrum.https.encryption.keystorePassword=p*********s
#spectrum.https.encryption.truststorePassword=p*********s
#spectrum.cache.encryption.keystorePassword=p*********s
#spectrum.cache.encryption.truststorePassword=p*********s
#spectrum.index.password=p*********s
#spectrum.index.encryption.keystorePassword=p*********s
#spectrum.index.encryption.truststorePassword=p*********s


SpectrumDirectory\Program Files\Pitney
Bowes\Spectrum\server\bin\enableadmin.properties
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.truststorePassword=p*********s

/jobexecutor.properties (available on client side where CLI utilities
are downloaded)
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.truststorePassword=p*********s

/pflowexecutor.properties (available on client side where CLI utilities
 are downloaded)
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.truststorePassword=p*********s

/cli.properties (available on client side where CLI utilities are
downloaded)
spectrum.encryption.keystorePassword=p*********s
spectrum.encryption.truststorePassword=p*********s
```

4 - データ ソース

このセクションの構成

データ ソース接続

データ ソースとは、データベース、ファイル サーバー、クラウド サービスなど、Spectrum™ Technology Platform から処理したいデータが入っているさまざまなソースを指します。Spectrum™ Technology Platform では 20 種類を超えるデータ ソースに接続できます。

Spectrum™ Technology Platform をデータ ソースに接続するには、入力 XML を定義する前に、まず接続を定義する必要があります。同様に、データフローの出力をデータベースに書き込む場合は、最初に、データベースを外部リソースとして定義する必要があります。

例えば、組織のデータが、Salesforce、Apache Cassandra、Hadoop、Dynamo DB、SQL サーバー、CSV ファイルなど、さまざまなソースに存在しているとします。

データ セットにアクセスするには:

1. 最初にこれらのデータ ソースすべてに接続する必要があります。Spectrum™ Technology Platform では、これらのすべてと、さらに多くのデータ ソース (後述のサブセクションを参照) に接続できます。
2. これらの接続の確立に成功したら、以下からアクセスできます。
   - **Enterprise Designer** の各種ステージ。例:
     - **Read From DB**
     - **Read From File**
     - **Read from Hadoop Sequence File**
     - **Read from Hive File**
     - **Read From HL7 File**
     - **Read from NoSQL DB**
     - **Read from SAP**
     - **Read from Spreadsheet**
     - **Read from Variable Format File**
     - **Read From XML**
   - **Metadata Insights** の Discovery、Modeling、Profiling モジュール

接続の定義

Spectrum™ Technology Platform で新しい接続を定義するには、以下のいずれかのモジュールを使用します。

- Management Console
- Metadata Insights の **[接続]** メニュー オプション

> **注：** Spectrum™ Technology Platform サーバー上のローカルファイルのデータに対する読み書き操作を行う場合は、接続を定義する必要はありません。

Amazon への接続

Amazon DynamoDB への接続

Spectrum™ Technology Platform で Amazon DynamoDB のデータにアクセスするには、Management Console を使って Amazon DynamoDB への接続を定義する必要があります。接続を定義した後は、Amazon DynamoDB に対してデータの読み書きを行うデータフローを Enterprise Designer で作成できます。

> **注：** この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注：** デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Amazon DynamoDB]** を選択します。
5. **[アクセス キー ID]** フィールドに、Amazon AWS アカウントにアクセスするために与えられている 20 文字の英数字列を入力します。
6. **[シークレット アクセス キー]** フィールドに、接続を認証するために必要な 40 文字のキーを入力します。
7. **[リージョン]** フィールドで、Amazon AWS アカウントのリージョンを選択します。
8. 接続をテストするには、**[テスト]** をクリックします。
9. **[保存]** をクリックします。

Amazon DynamoDB の制限事項

1. リスト、セット、マップなどの階層構造のデータ タイプは、String データ タイプとして解釈されます。これらのデータ タイプはサポートされていないためです。
2. DynamoDB データ ソースの null 値は、空の列値として解釈されます。
3. `count` 集約関数は Model Store に対するクエリではサポートされません。

Amazon S3 への接続

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[クラウド]** を選択します。
5. **[クラウド サービス]** フィールドで、**[AmazonS3]** を選択します。
6. **[バケット名]** フィールドに、お使いの Amazon S3 クラウド サービスで定義されているバケット名を入力します。Spectrum™ Technology Platform はこのバケットにファイルを読み書きします。
7. Amazon によって割り当てられたアクセス キーと秘密鍵を入力します。
8. **[ストレージ タイプ]** フィールドで、データ ストレージに対して許容する冗長性レベルを選択します。

標準	Amazon S3 で提供されるデフォルトの冗長性レベルです。
低冗長化	重要性が低く、簡単に再作成可能なデータを、低いレベルの冗長性で保存します。このオプションを使用すると、適度に信頼できるストレージが低いコストで利用できます。

9. **[暗号]** セクションで、データ暗号化方式を選択します。サーバー側の暗号化、クライアント側の暗号化、または両方を選択できます。

   サーバー側のキー

   データはサーバー側で暗号化および復号化されます。データはプレーン テキストで Amazon クラウド サービスに送信され、そこで暗号化および格納されます。取得時には、Amazon クラウド サービスによって復号化されたデータが、プレーン テキストでユーザのシステムに送信されます。

   キーの指定方法は、2 つあります。

   - **AWS 管理**: キーは、Amazon S3 クラウド サービスによって自動的に生成されます。
   - **ユーザ提供**: Amazon S3 クラウド サービスがサーバー側でデータを暗号化/復号化するために使用するキーを入力します。

   クライアント側のキー

   データはクライアント側で暗号化および復号化されます。データは、ユーザのクライアント システム上でローカルに暗号化されてから、Amazon S3 クラウド ストレージに送信されます。取得時には、暗号化形式で送り返されたデータが、クライアント システム上で復号化されます。

クライアント側のキー: クライアント システムがデータを暗号化/復号化するために使用するキーを入力します。

[サーバー側のキー] と **[クライアント側のキー]** の両方を選択した場合、暗号化と復号化は、サーバー側とクライアント側の両方で行われます。データはまず、クライアント側のキーで暗号化されて暗号化形式で Amazon に送信され、そこでサーバー側のキーによって再度暗号化されて格納されます。取得時には、Amazon がまずサーバー側のキーによってデータを復号化してから、暗号化形式でユーザのシステムに送信し、そこで最後に、クライアント側のキーによる復号化が行われます。

注 : Amazon S3 クラウドの暗号化機能を利用するには、Amazon S3 Security JAR ファイルをインストールする必要があります。詳細については、「**Amazon S3 クラウド暗号**（95ページ）」を参照してください。

Amazon S3 暗号化機能の詳細については、以下を参照してください。

docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html

10. アクセス権限を設定する場合は、**[権限]** セクションで + をクリックします。

    被付与者は次の 3 種類です。

Everyone	Authenticated Users と Log Delivery グループ以外のすべてのユーザ。
AuthenticatedUsers	Amazon にログインしたユーザ。
LogDelivery	Bucket Logging が有効である場合に、ユーザ指定のバケットにアクティビティ ログを書き込むユーザ。

    それぞれの被付与者に対し、必要な権限を次の中から選択します。

開く/ダウンロード	ファイルのダウンロードが可能です。
ビュー	ファイルに対する現在の権限を表示できます。
編集	ファイルに対する権限を変更および設定できます。

11. 接続をテストするには、**[テスト]** をクリックします。
12. **[保存]** をクリックします。

Amazon S3 クラウド暗号

Amazon S3 クラウド サービスの暗号セキュリティ機能を使うには、セキュリティ JAR ファイルをダウンロードし、Spectrum™ Technology Platform サーバーに配置する必要があります。暗号の使用は任意です。

1. ダウンロード サイトに移動します。

Java 7 を使用する Windows または Linux プラットフォームの場合、JAR ファイルは次の場所からダウンロードできます。

http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html

Java 7 を使用する AIX プラットフォームの場合、JAR ファイルは次の場所からダウンロードできます。

https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=jcesdk

2. 次の 2 つの JAR ファイルをダウンロードします。
   - `local_policy.jar`
   - `US_export_policy.jar`
3. JAR ファイルを次の場所に配置します。

```
%JAVA_HOME%\jre\lib\security
```

4. サーバーを再起動します。

Amazon SimpleDB への接続

Spectrum™ Technology Platform で Amazon SimpleDB のデータにアクセスするには、Management Console を使って Amazon SimpleDB への接続を定義する必要があります。接続を定義した後は、Amazon SimpleDB に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Amazon SimpleDB]** を選択します。
5. **[アクセス キー ID]** フィールドに、Amazon AWS アカウントにアクセスするために与えられている 20 文字の英数字列を入力します。
6. **[シークレット アクセス キー]** フィールドに、接続を認証するために必要な 40 文字のキーを入力します。
7. 接続をテストするには、**[テスト]** をクリックします。
8. **[保存]** をクリックします。

Amazon SimpleDB の制限事項

書き込みの制限事項

Write to DB ステージで、Amazon SimpleDB テーブルに書き込む場合は **[更新]** の書き込みモードは使用できません。**[挿入]** オプションで、挿入と更新の両方の操作が処理されます。挿入と更新の区別は、すべての Amazon SimpleDB テーブルに存在する `ItemName` 列の一意の値を使用して行われます。

理由: 更新クエリでは更新対象のテーブルのレコードごとにプライマリ キーが必要になりますが、これは Amazon SimpleDB データベースではサポートされていません。

読み取りの制限事項

集約関数 `SUM` および `AVG` は、Model Store に対するクエリの実行中はサポートされません。

Apache Cassandra への接続

Spectrum™ Technology Platform で Cassandra データベースのデータにアクセスするには、Management Console を使って Cassandra データベースへの接続を定義する必要があります。接続を定義した後は、Cassandra データベースに対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注 : デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注 : デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注 : 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Apache Cassandra]** を選択します。
5. **[ホスト]** フィールドに、Apache Cassandra データベースがインストールされているマシン名または IP を入力します。
6. **[キースペース]** フィールドに、アクセスするデータ センターのキースペース名を入力します。
7. **[ポート]** フィールドに、Apache Cassandra データベースが設定されているポートを入力します。
8. Cassandra データベースの認証に使用するユーザ名とパスワードを入力します。
9. **[一貫性レベル]** フィールドで、データ トランザクションを正常に実行するために、複製ノードにおいてデータ行がどれだけ一致する必要があるかを選択します。使用可能なノードの少なくとも 1 つ、すべて、または組み合わせとすることができます。
10. **[フェッチ サイズ]** に、各読み取りトランザクションで取得する結果セット行の数を入力します。
11. 接続をテストするには、**[テスト]** をクリックします。
12. **[保存]** をクリックします。

Apache Cassandra の制限事項

`count` 集約関数は Model Store に対するクエリではサポートされません。

Azure クラウドへの接続

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   > **注 :** 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[クラウド]** を選択します。
5. **[クラウド サービス]** フィールドで、**[AzureBlobStorage]** を選択します。
6. **[プロトコル]** フィールドで、Azure と Spectrum™ Technology Platform の間の接続に HTTP と HTTPS のどちらを使用するかを選択します。
7. **[アカウント名]** フィールドに、Azure ストレージ アカウントの名前を入力します。
8. **[アクセス キー]** フィールドに、Azure アカウントへのアクセス キーを入力します。
9. クラウド接続をテストするには、**[テスト]** をクリックします。
10. **[保存]** をクリックします。

Data Hub への接続

次の手順は、Data Hub モデルをデータ ソースとして使用する方法を示しています。Data Hub コネクタを使用して、Metadata Insights で使用されるエンティティ モデル データを読み取ることができます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   > **注 :** 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Data Hub]** を選択します。
5. **[モデル名]** フィールドに Data Hub モデル名を入力します。
6. 接続をテストするには、**[テスト]** をクリックします。
7. **[保存]** をクリックします。

Data Hub コネクタの制限事項

- Data Hub コネクタは、モデル内のエンティティからしか読み取ることができません。関連性からは読み取れません。
- `count` 集約関数内の列名を `count(column_name)` と指定する必要があります。`count(*)` としてワイルドカードを使用することはできません。

フラット ファイルへの接続

区切り記号付きフラット ファイルへの接続

新しい区切り記号付きフラット ファイル接続を追加するには、**[接続] > [フラット ファイル]** に移動して、**[レコード タイプ]** として **[区切り記号付き]** を選択します。ファイルのアクセスとコンテンツ タイプの詳細情報を入力して、Data Federation モジュールがファイルを正しく読み取れるようにします。

> **注：** この接続は、Metadata Insights モジュールで使用されます。

1. **[接続] > [フラット ファイル]** に移動します。
2. デフォルトで、画面が作成モードで開きます。あるいは、＋をクリックして新しいフラット ファイル接続を追加します。
3. フラット ファイル データ接続の **[接続名]** を入力します。
4. **[参照]** をクリックしてファイルのディレクトリを選択することにより、**[ファイル パス]** を入力します。
5. フラット ファイルの **[文字エンコーディング]** をドロップダウン リストから選択します。
6. **[レコード タイプ]** として **[区切り記号付き]** を選択します。
7. **[フィールド区切り文字]** で、ファイル レコードの任意の 2 つのフィールドの間の区切り文字を選択します。
8. ファイル レコードのフィールド値を囲む **[テキスト修飾子 (オプション)]** を必要に応じて選択します。
9. **[行区切り文字]** では［デフォルト］が選択されており、Spectrum™ Technology Platform が Unix と Windows のどちらのシステム上で稼働しているかによって行区切り文字が異なることを表します。
10. ファイルの先頭行がヘッダ行であるかどうかを指定するには、**[最初の行はヘッダ レコード]** スライダを **[はい]** または **[いいえ]** のいずれかに移動します。
11. ファイルの任意のレコードの多様なフィールドのデータ タイプを自動的に検出するかどうかを指定するには、**[ファイルからデータ タイプを検出]** スライダを **[はい]** または **[いいえ]** のいずれかに移動します。
12. ファイルのパーシング時に形式に誤りのあるレコードを飛ばすには、**[形式に誤りのあるレコードをスキップ]** スライダを **[オン]** に移動します。
13. **[テスト]** をクリックします。
    接続のテストが正常に終了したことを示すメッセージが表示されます。
14. **[保存]** をクリックします。

接続が正常に作成されたことを示すメッセージが表示されます。

作成された区切り記号付きフラット ファイル接続を使用して取得されたサンプル レコードを表示するには、ヘッダ バーの **[プレビュー]** をクリックします。ファイル レコードが取得され、ユーザが指定した設定に基づいてフィールドがソートされます。

固定長フラット ファイルへの接続

新しい固定長フラット ファイル接続を追加するには、**[接続] > [フラット ファイル]** に移動して、**[レコード タイプ]** として **[固定長]** を選択します。ファイルのアクセスとコンテンツ タイプの詳細情報を入力して、Data Federation モジュールがファイルを正しく読み取れるようにします。

注： この接続は、Metadata Insights モジュールで使用されます。

1. **[接続] > [フラット ファイル]** に移動します。
2. デフォルトで、画面が作成モードで開きます。あるいは、 をクリックして新しいフラット ファイル接続を追加します。
3. フラット ファイル データ接続の **[接続名]** を入力します。
4. **[参照]** をクリックしてファイルのディレクトリを選択することにより、**[ファイル パス]** を入力します。
5. フラット ファイルの **[文字エンコーディング]** をドロップダウン リストから選択します。
6. **[レコード タイプ]** として **[固定長]** を選択します。
7. **[レコード長]** フィールドに、ファイル レコード内の文字総数を入力します。

ステップ 8 ～ 13 を繰り返して、ファイル レコード内で想定されるすべてのフィールドに対して情報を入力します。

8. **[フィールドの追加]** をクリックして、ファイル レコード内のフィールド用の行を追加します。
9. **[名前]** 列に、フィールド値の名前を入力します。
10. **[タイプ]** 列で、フィールド値のデータ タイプを選択します。
11. **[開始位置]** 列に、ファイル レコード内におけるそのフィールド値の開始位置を入力します。
    ファイル レコードの最初のフィールドから順に **[開始位置]** は 1 から開始します。
12. **[長さ]** フィールドに、**[開始位置]** の文字を含むそのフィールドの文字総数を入力します。
    どのフィールドについても、**[開始位置]** と **[長さ]** の値の合計は、**[レコード長]** を超えてはいけません。

    次のファイル レコードがあるとします。

```
01234Rob Smith29PitneyBowes
```

    レコード長 = 27

    フィールド 'Name' の各列の値は次のとおりです。

開始位置 = 6

長さ = 9

```
Name = Rob Smith
```

13. フィールド値の先頭または末尾の空白を削除する場合は、**[トリム]** チェックボックスをオンにします。
14. **[テスト]** をクリックします。
    接続のテストが正常に終了したことを示すメッセージが表示されます。
15. **[保存]** をクリックします。
    接続が正常に作成されたことを示すメッセージが表示されます。

作成された固定長フラット ファイル接続を使用して取得されたサンプル レコードを表示するには、ヘッダ バーの **[プレビュー]** をクリックします。ファイル レコードが取得され、ユーザが指定した設定に基づいてフィールドがソートされます。

ファイル接続における日付/時刻形式

Spectrum™ Technology Platform でファイル接続を用いてファイルから日付および時刻値を読み込む場合、それらの値はある特定の日付/時刻形式に従っている必要があります

許容される日付/時刻形式

- Date: "yyyy-mm-dd"
- Datetime: "yyyy-mm-dd HH:mm:ss"
- Time: "HH:mm:ss"

これらの形式は、標準の日付/時刻表記に基づきます。

区切り記号付きファイル

区切り記号付きファイルの設定時に **[検出タイプ]** 機能をオンにすると、上記の形式に従うファイル レコード内の日付値と時刻値が自動的に Date タイプとして検出されます。

許容される形式のいずれにも従わない日付/時刻値は、Date タイプではなく String タイプの値として読み込まれます。

固定長ファイル

固定長ファイルの場合、固定長ファイル接続を作成する際に date タイプの値が設定されます。そのため、これらの値は、許容形式に従っているかどうかにかかわらず Date タイプ値として読み込まれます。

固定長ファイルの中の日付/時刻値が許容形式に従っていない場合は、Logical Model 作成ステージにおいて**変換**を使用してそれを処理する必要があります。これを行うには、以下の *[変換]* カテゴリの関数を値に適用します。

```
parsedate(String date, String format)
```

ここで、*date* はファイルから受け取った値で、*format* はファイルから受け取った値の日付/時刻形式です。これによって、この日付/時刻値を正しくパースできるようになります。

例えば、date = `23-Feb-2008` の場合、format = `dd-MMM-yyyy` となります。

結果の値形式

Model Store でデータをプレビューする場合:

- 日付/時刻値として読み込まれている値は、許容されるいずれかの日付/時刻形式でプレビューに表示されます。
- String 値として読み込まれている値は、そのままプレビューに表示されます。

FTP サーバーへの接続

Spectrum™ Technology Platform から FTP サーバー上のファイルにアクセスするには、Management Console を使って FTP サーバーへの接続を定義する必要があります。接続を定義した後は、FTP サーバー上のファイルに対してデータの読み書きを行うデータフローを Enterprise Designer で作成できます。

FTP サーバーに接続する前に、FTP サーバー上のタイムアウトの設定が、この接続を使うジョブに適しているか確認します。ジョブの設計によっては、接続がアイドルになる時間があり、それが接続のタイムアウトを引き起こす可能性があります。例えば、2 つの Read from File ステージが 1 つの Import To Hub ステージに接続されているデータフローがあるとします。Import To Hub ステージが一方の Read from File ステージからレコードを読み込んでいる間にもう一方がアイドルとなり、FTP サーバーへの接続がタイムアウトになる可能性があります。接続がタイムアウトにならないように、FTP サーバー上のタイムアウト値に 0 を設定することを検討してください。

> **注：** FTP サーバーは、能動的接続モードで実行されている必要があります。受動的接続モードはサポートされていません。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**FTP** を選択します。
5. **[ユーザ名]** と **[パスワード]** のフィールドに、FTP サーバーを認証するために使用する資格情報を入力します。これは、FTP サーバーがユーザ名を要求する場合にのみ必要です。
6. **[ホスト]** フィールドに、FTP サーバーのホスト名または IP アドレスを入力します。
7. **[ポート]** フィールドに、 サーバーで FTP に使用されるネットワーク ポートを入力します。
8. **[テスト]** をクリックして、Spectrum™ Technology Platform サーバーが FTP サーバーに接続できることを確認します。
9. **[保存]** をクリックします。

SFTP サーバーへの接続

Spectrum™ Technology Platform から SFTP サーバー上のファイルにアクセスするには、Management Console を使って SFTP サーバーへの接続を定義する必要があります。

SFTP サーバーに接続する前に、SFTP サーバー上のタイムアウトの設定が、この接続を使うジョブに適しているか確認します。ジョブの設計によっては、接続がアイドルになる時間があり、それが接続のタイムアウトを引き起こす可能性があります。例えば、2 つの Read from File ステージが 1 つの Import To Hub ステージに接続されているデータフローがあるとします。Import To Hub ステージが一方の **Read from File** ステージからレコードを読み込んでいる間に、もう一方がアイドル状態になり、SFTP サーバーへの接続がタイムアウトになることがあります。接続がタイムアウトにならないように、SFTP サーバー上のタイムアウト値に 0 を設定することを検討してください。

注： SFTP サーバーは、能動的接続モードで実行されている必要があります。受動的接続モードはサポートされていません。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[SFTP]** を選択します。
5. **[ホスト]** フィールドに、SFTP サーバーのホスト名または IP アドレスを入力します。
6. **[ポート]** フィールドに、 サーバーで SFTP に使用されるネットワーク ポート番号を入力します。デフォルトは 22 です。
7. 厳密なホスト キー チェックを有効にする場合は、**[厳密なホスト キー チェック]** ボタンを **[はい]** に切り替えます。デフォルトは **[いいえ]** です。厳密なホスト キー チェックを有効にすると、*ssh* はホスト キーを既知のホスト ファイルに自動的に追加しません。これは追加のセキュリティ機能です。
8. **[既知のホスト ファイル]** フィールドで、既知のホストの詳細を管理しているファイルの場所を参照し、ファイルを選択します。

   注： **[厳密なホスト キー チェック]** を無効にした場合、このフィールドは表示されません。逆に、厳密なホスト キー チェックを有効にしている場合は、この詳細が必須です。

9. **ユーザ名**を入力します。
10. 優先する**認証タイプ**を選択します。選択できる認証タイプは、［パスワード］または［キーベース］です。デフォルトは［パスワード］です。
    - **パスワード**: これを認証タイプとして選択した場合は、**[認証タイプ]** フィールドの下に **[パスワード]** フィールドが表示されます。必要なパスワードをこのフィールドに入力します。
    - **キーベース**: キーベース認証を選択した場合は、**秘密鍵ファイル**を参照して選択し、ホストサーバー管理者によって共有されている**パスフレーズ**を入力します。パスフレーズは、このオプションを使用してキーが設定されている場合のみ必要です。
11. **[テスト]** をクリックして、Spectrum™ Technology Platform サーバーから SFTP サーバーに接続できることを確認します。
12. **[保存]** をクリックします。

Google Cloud Storage への接続

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

Management Console:	http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。 **注：** デフォルトの HTTP ポートは 8080 です。 **[リソース] > [データ ソース]** に移動します。
Metadata Insights:	http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。 **注：** デフォルトの HTTP ポートは 8080 です。 **[接続]** に移動します。

2. [接続を追加] ボタン ＋ をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[クラウド]** を選択します。

5. **[クラウド サービス]** フィールドで、**[GoogleCloudStorage]** を選択します。
6. **[バケット名]** フィールドに、お使いの Google クラウド サービスで定義されているバケット名を入力します。Spectrum™ Technology Platform はこのバケットにファイルを読み書きします。
7. アプリケーション名、サービス アカウント、Google によって提供された秘密鍵ファイルを入力します。

   注： 秘密鍵ファイルが Spectrum™ Technology Platform サーバー上に存在することを確認してください。

8. アクセス権限は、**[権限]** セクションで設定できます。

データと権限の管理	ユーザは、データと権限を管理できます。
データを表示	ユーザは、データを表示できます。
データを管理	ユーザは、データを管理できます。

9. 接続をテストするには、**[テスト]** をクリックします。
10. **[保存]** をクリックします。

詳細については、Google の**サービス アカウント認証情報**を参照してください。

Hadoop への接続

Read from Hadoop Sequence File、**Write to Hadoop Sequence File**、**Read From File**、**Write to File**、**Read From XML**、**Write to XML**、**Read from Hive File**、**Write to Hive File**、**Read From HL7 File** などのステージを **Enterprise Designer** で使用するには、Hadoop システムに接続します。

重要： Spectrum™ Technology Platform は、Windows プラットフォーム上の Kerberos 認証に対して *Hadoop 2.x* をサポートしません。

Hadoop システムに接続するには、次の手順を実行します。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[HDFS]** を選択します。
5. **[ホスト]** フィールドに、HDFS クラスタ内の NameNode のホスト名または IP アドレスを入力します。
6. **[ポート]** フィールドに、ネットワーク ポート番号を入力します。
7. **[ユーザ]** で、次のいずれかのオプションを選択します。

   サーバー ユーザ HDFS クラスタで認証が有効になっている場合は、このオプションを選択します。このオプションでは、Spectrum™ Technology Platform サーバーを実行するユーザ資格情報を使用して HDFS を認証します。

   ユーザ名 HDFS クラスタで認証が無効になっている場合は、このオプションを選択します。

8. この HDFS ファイル サーバー接続に対して Kerberos 認証機能を有効にする場合は、**[Kerberos]** チェックボックスをオンにします。
9. **[Kerberos]** 認証を有効にした場合は、**[Keytab ファイル パス]** フィールドに Keytab ファイルのパスを入力します。

   注： Keytab ファイルが Spectrum™ Technology Platform サーバー上に存在することを確認してください。

10. **[プロトコル]** フィールドで、次のいずれかを選択します。

    WEBHDFS HDFS クラスタで HDFS 1.0 以降を実行している場合は、このオプションを選択します。このプロトコルは、読み込みと書き込みの両方の操作をサポートしています。

HFTP HDFS クラスタで HDFS 1.0 よりも古いバージョンを実行している場合、または組織で WEBHDFS プロトコルが許可されていない場合は、このオプションを選択します。このプロトコルは、読み込み操作のみをサポートしています。

HAR Hadoop アーカイブ ファイルにアクセスする場合は、このオプションを選択します。このオプションを選択する場合は、アーカイブ ファイルへのパスを **[パス]** フィールドに指定します。このプロトコルは、読み込み操作のみをサポートしています。

11. **[詳細オプション]** を展開します。
12. WEBHDFS プロトコルを選択した場合は、必要に応じて次の詳細オプションを指定できます。

複製係数 各ブロックを複製するデータ ノードの数を指定します。例えば、デフォルト設定の 3 は、各ブロックをクラスタ内の異なる 3 つのノードに複製します。最大複製係数は 1024 です。

ブロック サイズ 各ブロックのサイズを指定します。HDFS は、ここで指定するサイズのブロックにファイルを分割します。例えば、デフォルトの 64 MB を指定した場合、各ファイルは 64 MB ブロックに分割されます。その後、各ブロックは、**[複製係数]** フィールドに指定された、クラスタ内のノード数に複製されます。

ファイル権限 Spectrum™ Technology Platform によって HDFS クラスタに書き込まれるファイルに対するアクセス レベルを指定します。次の各オプションに対して、読み取り権限および書き込み権限を指定できます。

注： *実行*権限は Spectrum™ Technology Platform に適用されません。

ユーザ これは前の手順で指定した、**[サーバー ユーザ]** のユーザか、**[ユーザ名]** フィールドに指定したユーザのいずれかです。

グループ これは、ユーザがメンバーとして所属する任意のグループを指します。例えば、ユーザが john123 の場合、グループ権限は john123 がメンバーとして所属するグループにすべて適用されます。

その他 これは、他のすべてのユーザと、指定されたユーザがメンバーとして所属しないグループを指します。

13. 以下の**ファイル権限**についての説明を参照し、ステージやアクティビティで接続が使用される際にソートとフィルタリングの機能が正しく動作するよう、Hadoop の [サーバー プロパティ] を定義します。プロパティを追加するには、次のいずれかの手順を実行します。
    -  をクリックし、プロパティとその値をそれぞれ **[プロパティ]** および **[値]** フィールドに追加します。

- をクリックし、設定 XML ファイルをアップロードします。この XML ファイルは `hdfs-site.xml`、`yarn-site.xml`、または `core-site.xml` のようになっているはずです。

  注： サーバーに設定ファイルを配置します。

ファイル権限とパラメータ - Hadoop 1.x

このセクションの説明は、次のステージおよびアクティビティに適用されます。

- ステージ - **Read from Sequence File**
- アクティビティ - **Run Hadoop Pig**

fs.default.name

Hadoop が実行するノードとポートを指定します。例えば、`hdfs://152.144.226.224:9000` とします。

mapred.job.tracker

MapReduce ジョブ トラッカーを実行するホスト名または IP アドレスと、ポートを指定します。ホスト名をローカルとして入力した場合は、ジョブは単一のマップとして実行され、タスクが少なくなります。例えば、`152.144.226.224:9001` とします。

dfs.namenode.name.dir

DFS 名前ノードが名前テーブルを格納する、ローカル ファイルシステム上の場所を指定します。ディレクトリのカンマ区切りリストである場合、名前テーブルは冗長性のためにすべてのディレクトリに複製されます。例えば、`file:/home/hduser/Data/namenode` とします。

hadoop.tmp.dir

他の一時ディレクトリのベース ディレクトリを指定します。例えば、`/home/hduser/Data/tmp` とします。

ファイル権限とパラメータ - Hadoop 2.x

このセクションの説明は、次のステージおよびアクティビティに適用されます。

- ステージ - **Read from Sequence File**
- アクティビティ - **Run Hadoop Pig**

fs.defaultFS

Hadoop が実行するノードとポートを指定します。例えば、`hdfs://152.144.226.224:9000` とします。

注意: Spectrum バージョン 11.0 以前では、パラメータ名 `fs.defaultfs` を使用する必要があります。大文字と小文字の違いに注意してください。バージョン 11

SP1 以降では、`fs.defaultfs` と `fs.defaultFS` のどちらの名前も有効です。11.0 SP1 以降のリリースでは、パラメータ名 `fs.defaultFS` を使用することをお勧めします。

yarn.resourcemanager.resource-tracker.address

Resource Manager のホスト名または IP アドレスを指定します。例えば、`152.144.226.224:8025` とします。

yarn.resourcemanager.scheduler.address

Scheduler Interface のアドレスを指定します。例えば、`152.144.226.224:8030` とします。

yarn.resourcemanager.address

Resource Manager に含まれる Applications Manager インターフェイスのアドレスを指定します。例えば、`152.144.226.224:8041` とします。

mapreduce.jobhistory.address

MapReduce Job History Server が実行するホスト名または IP アドレスと、ポートを指定します。例えば、`152.144.226.224:10020` とします。

mapreduce.application.classpath

Map Reduce アプリケーション用の CLASSPATH を指定します。この CLASSPATH は、Map Reduce アプリケーションに関連するクラスが存在する場所を表します。エントリをカンマで区切って指定する必要があります。

例:

```
$HADOOP_CONF_DIR, $HADOOP_COMMON_HOME/share/hadoop/common/*,
$HADOOP_COMMON_HOME/share/hadoop/common/lib/*,
$HADOOP_HDFS_HOME/share/hadoop/hdfs/*,
$HADOOP_HDFS_HOME/share/hadoop/hdfs/lib/*,
$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*,
$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/lib/*,
$HADOOP_YARN_HOME/share/hadoop/yarn/*,
$HADOOP_YARN_HOME/share/hadoop/yarn/lib/*
```

mapreduce.app-submission.cross-platform

Spectrum サーバーが Windows コンピュータ上で実行しており、そこに Cloudera をインストールする場合に生じる、さまざまなプラットフォームの問題を処理します。Spectrum サーバーと Cloudera が異なるオペレーティング システム上で実行している場合は、このパラメータの値として `true` を入力します。それ以外の場合は、`false` にします。

> **注:** Cloudera は Windows クライアントをサポートしません。このパラメータを設定することは回避策であり、結果として生じるすべてのプラットフォームの問題を解決するものではありません。

ファイル権限とパラメータ - Kerberos

このセクションの説明は、次のステージおよびアクティビティに適用されます。

- ステージ - **Read from Sequence File**
- アクティビティ - **Run Hadoop Pig**

[Kerberos] チェック ボックスをオンにした場合は、以下の Kerberos 設定プロパティを追加します。

hadoop.security.authentication

使用される認証セキュリティの種類。`kerberos` という値を入力します。

yarn.resourcemanager.principal

Hadoop YARN リソース ネゴシエータ用のリソース マネージャに対して使用される Kerberos プリンシパル。例えば、`yarn/_HOST@HADOOP.COM`。

dfs.namenode.kerberos.principal

Hadoop 分散ファイル システム (HDFS) の NameNode に対して使用される Kerberos プリンシパル。例えば、`hdfs/_HOST@HADOOP.COM`。

dfs.datanode.kerberos.principal

Hadoop 分散ファイル システム (HDFS) のデータ ノードに対して使用される Kerberos プリンシパル。例えば、`hdfs/_HOST@HADOOP.COM`。

ファイル権限とパラメータ - Hadoop 1.x

このセクションの説明は、次のステージに適用されます。

- ステージ **Read from File**
- ステージ **Write to File**
- ステージ **Read from Hive ORC File**
- ステージ **Write to Hive ORC File**

fs.default.name

Hadoop が実行するノードとポートを指定します。例えば、`hdfs://152.144.226.224:9000` とします。

ファイル権限とパラメータ - Hadoop 2.x

このセクションの説明は、次のステージに適用されます。

- ステージ **Read or write from File**
- ステージ **Read or write from Hive ORC File**

fs.defaultFS

Hadoop が実行するノードとポートを指定します。例えば、`hdfs://152.144.226.224:9000` とします。

> **注意**: Spectrum バージョン 11.0 以前では、パラメータ名 `fs.defaultfs` を使用する必要があります。大文字と小文字の違いに注意してください。バージョン 11 SP1 以降では、`fs.defaultfs` と `fs.defaultFS` のどちらの名前も有効です。11.0 SP1 以降のリリースでは、パラメータ名 `fs.defaultFS` を使用することをお勧めします。

14. 接続をテストするには、**[テスト]** をクリックします。
15. **[保存]** をクリックします。

HDFS クラスタへの接続を定義した後は、Enterprise Designer のソース ステージとシンク ステージ (Read from File、Write to File など) でその接続を使用できるようになります。ソースまたはシンク ステージでファイルを定義するときに **[リモート マシン]** をクリックすると、HDFS クラスタを選択できます。

Hadoop に対する圧縮サポート

Spectrum™ Technology Platform は、Hadoop 上の圧縮形式として `gzip (.gz)` と `bzip2 (.bz2)` をサポートします。HDFS 接続で **Read from File** および **Write to File** ステージを使用している場合は、**[ファイル名]** フィールドで、必要な圧縮形式に対応する拡張子 (`.gz` または `.bz2`) を指定する必要があります。ファイルは、指定された圧縮拡張子に基づいて解凍または圧縮されます。Spectrum™ Technology Platform は、ファイルの圧縮と解凍を処理します。

Hive への接続

Spectrum™ Technology Platform が提供するドライバを通して、または Apache JDBC ドライバを追加することで、 Hive データベースに接続できます。Spectrum™ Technology Platform が提供するドライバは (*hive-jdbc-1.2.2-batch-18.2.jar*)、Apache Hive ドライバの拡張バージョンで、バッチ処理をサポートしています。次の場所にあります。

`SpectrumDirectory``\server\modules\bigdata\drivers\hive`ここで、*SpectrumDirectory* は Spectrum™ Technology Platform サーバーをインストールしたフォルダです。

サーバーへの JDBC ドライバ ファイルの追加および接続の定義については、**JDBC データベースへの接続**（115ページ）を参照してください。

> **注：** Hive JDBC ドライバのクラス パスは `com.pb.spectrum.hive.jdbc.batch.HiveBatchDriver` です。

JDBC データベースへの接続

[データ ソース] ページを使用して接続を定義します。このページには、**Management Console** または **Metadata Insights** モジュールから移動できます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注 : デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注 : デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン  をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注 : 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、接続したいデータベースのタイプを選択します。

   Spectrum™ Technology Platform Data Integration モジュールには、SQL Server、Oracle、および PostgreSQL データベース用の JDBC ドライバが含まれています。別のデータベース タイプに接続する場合は、接続を定義する前に JDBC ドライバを追加する必要があります。

5. **[URL]** フィールドに、JDBC 接続の URL を入力します。この URL はデータベース管理者から提供されます。

   例えば、サーバー "MyServer" でホストされている MySQL データベース "SampleDatabase" に接続するには、次のように入力します。

   ```
   jdbc:mysql://MyServer/SampleDatabase
   ```

6. JDBC ドライバによっては、他のフィールドにも入力する必要があります。それらのフィールドは、**[タイプ]** フィールドで選択した JDBC ドライバの接続文字列のさまざまなプロパティを表します。接続の種類によって異なる接続のプロパティと値の詳細については、JDBC ドライバ提供業者のドキュメントを参照するか、データベース管理者に問い合わせてください。
7. **[保存]** をクリックします。
8. 新しい接続の横にあるチェック ボックスをオンにして、[テスト] ボタン  をクリックすることによって、接続をテストします。

JDBC ドライバのインポート

Spectrum™ Technology Platform では、JDBC ドライバを使用して任意のデータベース内のデータにアクセスできます。Spectrum™ Technology Platform Data Integration Module には SQL、Oracle、および PostgreSQL 用のドライバが用意されていますが、他の種類のデータベース用のドライバも含まれています。必要とするデータベース タイプのドライバが Spectrum™ Technology Platform で提供されていない場合は、JDBC ドライバを追加します。

この手順では、ドライバ ファイルを Spectrum™ Technology Platform サーバーにコピーすることにより、JDBC ドライバをインポートします。この手順を実行した後、Management Console で JDBC データベース接続を定義すると、ドライバが利用できるようになります。

> **注：** この手順は *JDBC 4.x* ドライバに対して有効です。以前のバージョンの JDBC を使うドライバを追加する場合は、Management Console でドライバを手動で追加する必要があります。詳細については、**JDBC ドライバの手動による追加**（117ページ）

1. 目的のデータベース用のすべての JDBC ドライバ ファイルを適当なフォルダに入れます。

   `Name.jdbc`

   ここで、フォルダの*名前*は適当に決めてかまいません。ただし、名前の末尾に `.jdbc` を付けてください。

2. Spectrum™ Technology Platform を実行しているサーバーにログインします。
3. ドライバが入っているフォルダを、このフォルダにコピーします。

   SpectrumDirectory`\server\drivers`

   ドライバが自動的にインポートされます。

4. ドライバが正常にインポートされたことを確認するには、Management Console にログインし、**[システム] > [ドライバ]** に移動します。ドライバがリストされているはずです。

   ドライバがリストにない場合は、Management Console でシステム ログを開き、JDBC ドライバの展開関連のエラーが発生していないか確認します。

JDBC ドライバの手動による追加

Spectrum™ Technology Platform では、JDBC ドライバを使用して任意のデータベース内のデータにアクセスできます。Spectrum™ Technology Platform Data Integration Module には SQL、Oracle、および PostgreSQL 用のドライバが用意されていますが、他の種類のデータベース用のドライバも含まれています。必要とするデータベース タイプのドライバが Spectrum™ Technology Platform で提供されていない場合は、JDBC ドライバを追加します。

この手順では、JDBC ドライバのファイルをサーバーに追加し、接続文字列と接続のプロパティを手動で定義します。作業を開始する前に、ドライバで必要とされている接続文字列の形式とプロパティについて十分に理解してください。これらを正確に定義しないと、ドライバーは正常に機能しません。通常、ドライバの接続文字列とプロパティに関する情報は、ドライバ提供業者の Web サイトで入手できます。

> **注：** *JDBC 1.x*、*2.x*、または *3.x.* を使う JDBC ドライバを追加するときにだけ、この手順を使用することをお勧めします。*JDBC 4.x* を使うドライバの場合は、import メソッドを使用してドライバを追加することをお勧めします。詳細については、「**JDBC ドライバのインポート**（116ページ）」を参照してください。

1. Management Console を開きます。
2. **[システム] > [ドライバ]** に移動します。
3. [追加] ボタン  をクリックします。
4. **[名前]** フィールドに、ドライバの名前を入力します。任意の名前にすることができます。
5. **[JDBC ドライバ クラス名]** フィールドにドライバの Java クラス名を入力します。通常は、JDBC ドライバのドキュメントにクラス名が記載されています。

   例えば、Microsoft JDBC ドライバを使用するには、次のように入力します。

   ```
   com.microsoft.sqlserver.jdbc.SQLServerDriver
   ```

6. **[接続文字列テンプレート]** フィールドに、データベースへの接続に使用する JDBC 接続 URL を、接続文字列に設定するプロパティがあればそれらを含めて入力します。データベース ベンダーによって接続文字列は異なるため、接続文字列の詳細については、お使いのデータベースのドキュメントを確認してください。

   ドライバを複数のデータベース接続で使用する場合は、各接続によって異なる可能性があるプロパティ値をハードコーディングする代わりに、接続文字列にプロパティ トークンを使用することを検討してください。例えば、暗号化を使用する接続と使用しない接続が存在する場合は、暗号化プロパティ用のプロパティ トークンを定義することができます。

   接続文字列にプロパティ トークンを使用するには、次の構文を使用します。

   ```
   ${PropertyToken}
   ```

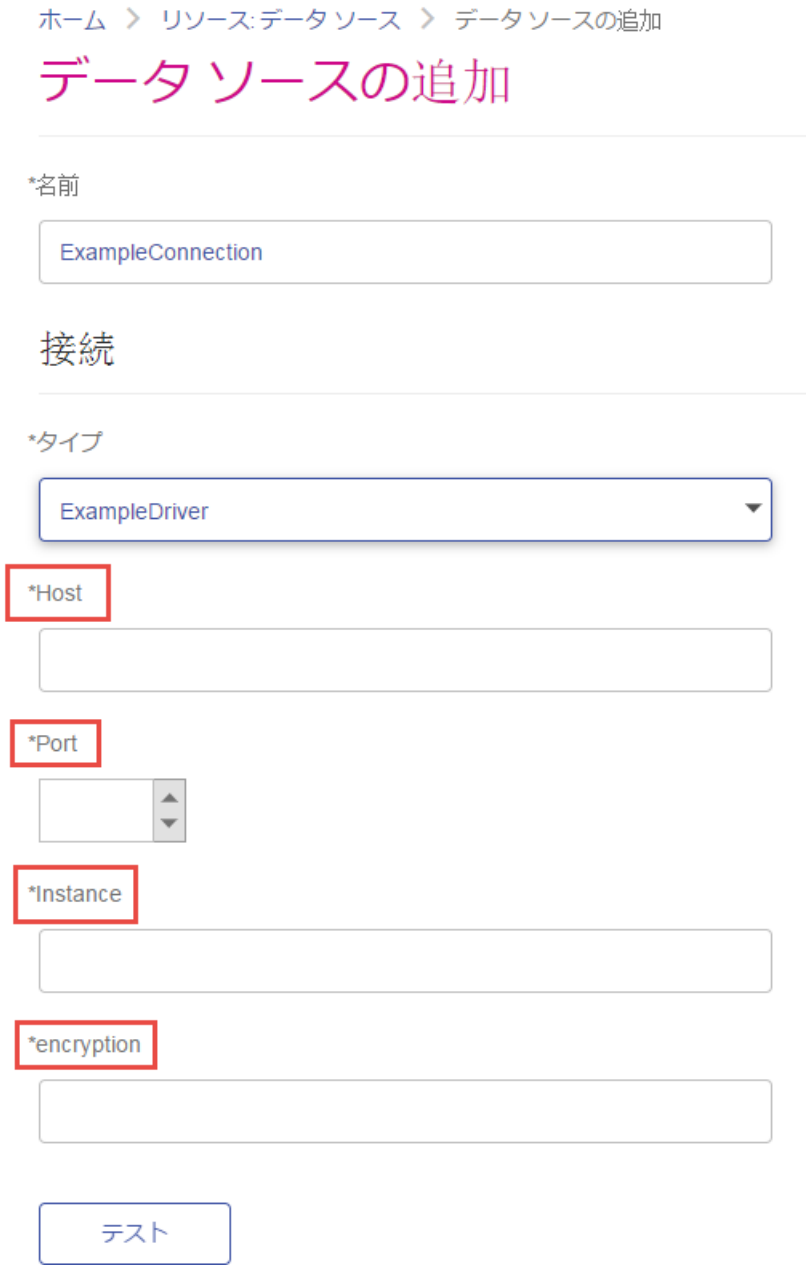
接続文字列テンプレートに含めるすべてのプロパティ トークンが、データベース接続を定義する際の必須フィールドになります。

> **注：** データベース パスワードが格納されるプロパティには、プロパティ トークン名 `${password}` を使用します。このトークン名を使用することで、パスワードは Management Console のフィールドでマスク表示され、データベース内では暗号化されます。

例えば、次の SQL の接続文字列には、ホスト、ポート、インスタンス、暗号化用のプロパティ トークンが含まれています。

```
jdbc:sqlserver://${host}:${port};databaseName=${instance};encrypt=${encryption};
TrustServerCertificate=true
```

これらのトークンは、このドライバを使用するデータベース接続を定義する際の必須フィールドです。



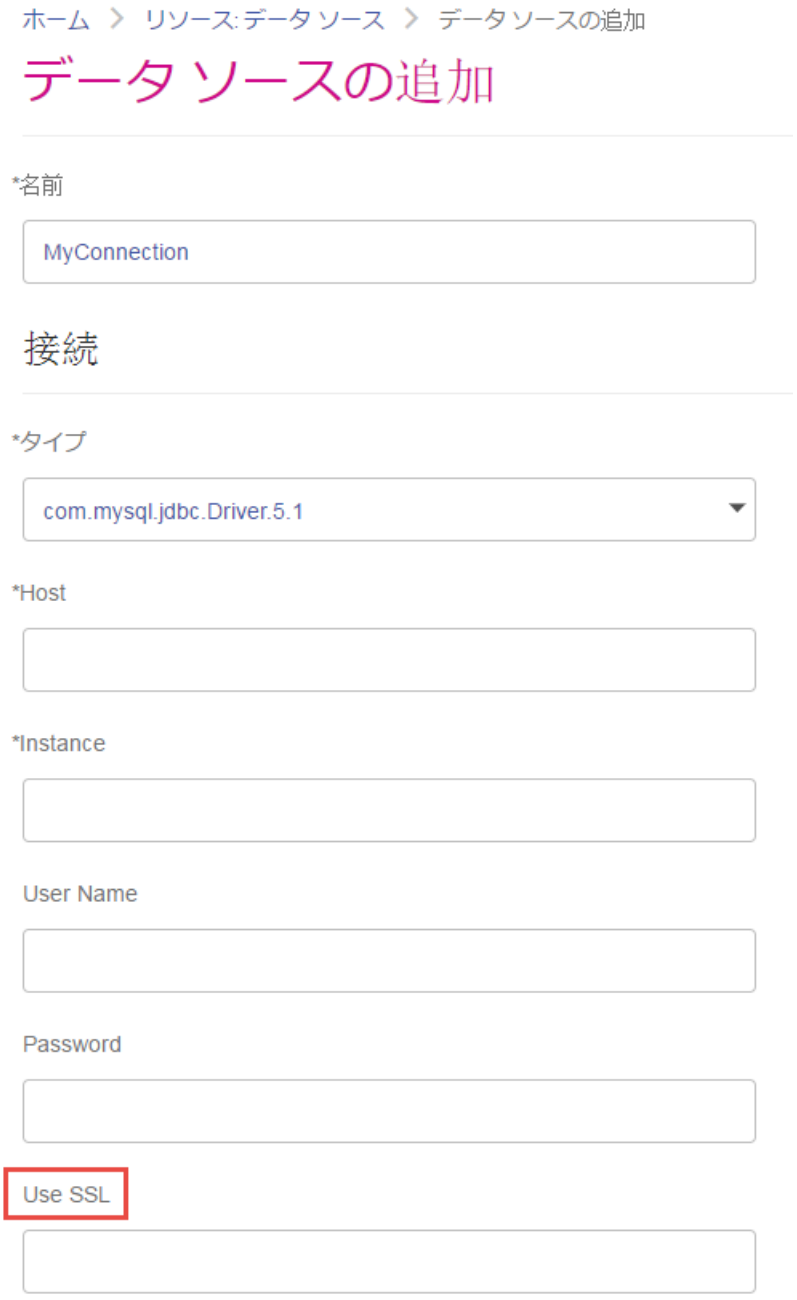
7. データベース接続のオプションにするプロパティは、**[接続プロパティ]** セクションで定義します。
   a) **[接続プロパティ]** セクションで、[追加] ボタン をクリックします。
   b) **[ラベル]** フィールドに、プロパティをわかりやすく説明するラベルを入力します。ここに入力したラベルが、このドライバを使用して接続を作成する際に、接続ウィンドウのフィールド ラベルとして使用されます。

c) **[プロパティ トークン]**フィールドに、オプションのプロパティのトークンを入力します。データベース ドライバでサポートされているプロパティについては、そのドライバのドキュメントを参照してください。

注： データベース パスワードが格納されるプロパティには、プロパティ トークン名 `password`を使用します。このトークン名を使用することで、パスワードは Management Console のフィールドでマスク表示され、データベース内では暗号化されます。

例えば、暗号化をこのドライバを使用するデータベース接続のオプションにする場合は、暗号化プロパティを次のように定義できます。

データベース接続がこのドライバを使用する際に、暗号化プロパティは、データベース接続におけるオプションのプロパティとして表示されます。



8. Spectrum™ Technology Platform を実行しているサーバーにログインし、データベース ドライバ ファイルをサーバー上の適当なフォルダに入れます。フォルダの位置は特に重要でありません。
9. **[ドライバ ファイル]** セクションで、[追加] ボタン をクリックします。

10. **[ファイル パス]** フィールドに、サーバー上のデータベース ドライバ ファイルへのパスを入力します。
11. **[保存]** をクリックします。

インポートした JDBC ドライバの削除

JDBC ドライバを Spectrum™ Technology Platform にインポートするときに Management Console から手動で追加しなかった場合、そのドライバを Management Console で削除することはできません。その場合は、次の手順でドライバを削除します。

重要： ドライバを削除する前に、そのドライバを使用しているデータベース接続が存在しないことを確認します。

1. Spectrum™ Technology Platform サーバーを停止します。
2. 次のフォルダに移動します。

   ```
   SpectrumDirectory\server\drivers
   ```

3. `drivers` フォルダで、ドライバが入っているフォルダを削除します。
4. Spectrum™ Technology Platform サーバーを開始します。
5. ドライバが削除されたことを確認するには、Management Console にログインし、**[システム] > [ドライバ]** に移動し、ドライバがもうリストにないことを確認します。

サポートされるデータベースのデータ タイプ

Spectrum™ Technology Platform は、データベースで一般的に使用されるこれらのデータ タイプをサポートしています。

bigdecimal	小数点以下 38 桁の精度をサポートする数値データ タイプ。高い精度が必要な算術計算で使用されるデータ (特に金融データ) には、このデータ タイプを使用してください。bigdecimal データ タイプは、double データ タイプより正確な計算をサポートします。
boolean	true と false の 2 つの値を持つ論理タイプ。
date	月、日、年を含むデータ タイプ。例: 2012-01-30、January 30, 2012。デフォルトの日付の形式は Management Console で指定できます。
datetime	月、日、年、時、分、秒を含むデータ タイプ。例: 2012/01/30 6:15 PM。
double	正と負の倍精度数を含む数値データ タイプ。値の範囲は、2^{-1074} ～ $(2-2^{-52})\times2^{1023}$。指数表記すると、値の範囲は、-1.79769313486232E+308 ～ 1.79769313486232E+308 となります。
float	正と負の単精度数を含む数値データ タイプ。値の範囲は、2^{-149} ～ $(2-2^{23})\times2^{127}$。指数表記すると、値の範囲は、-3.402823E+38 ～ 3.402823E+38 となります。

integer	正と負の整数を含む数値データ タイプ。値の範囲は、-2^{31} (-2,147,483,648) ～ 2^{31}-1 (2,147,483,647)。
long	正と負の整数を含む数値データ タイプ。値の範囲は、-2^{63} (-9,223,372,036,854,775,808) ～ 2^{63}-1 (9,223,372,036,854,775,807)。
string	文字シーケンス。
time	時刻を含むデータ タイプ。例: 21:15:59 or 9:15:59 PM。
Raw	可変長のバイナリ データを格納するための Oracle データタイプ。最大サイズは 2000 バイト (Oracle 7 では最大長は 255 バイトでした)

他のデータベースのデータ タイプは、次のように、サポートされるデータ タイプのいずれかに自動的に対応付けられます。

データベースのデータ タイプ	サポートされるデータ タイプ
日付/時間タイプ	
TIMESTAMP	datetime
文字列タイプ	
char	string
CLOB	string
LONGVARCHAR	string
NCHAR	string
NVARCHAR	string
VARCHAR	string
数値タイプ	
bigint	long
DECIMAL	double
FLOAT	double

データベースのデータ タイプ	サポートされるデータ タイプ
NUMERIC	bigdecimal
real	float
SMALLINT	integer
tinyint	integer
Boolean タイプ	
BIT	boolean

Location Intelligence モジュールでサポートされるデータベースのデータ タイプ

これらのデータベースのデータ タイプは、Location Intelligence モジュールでサポートされるデータ タイプのいずれかに自動的に対応付けられます。

データベースのデータ タイプ	サポートされるデータ タイプ
SQL Server	
tinyint	SHORT_INTEGER
smallint	SHORT_INTEGER
int	INTEGER
bigint	LONG_INTEGER
float	DOUBLE
real	DOUBLE
decimal(10, 5)	DOUBLE
numeric(10, 5)	DOUBLE
date	DATE

データベースのデータ タイプ	サポートされるデータ タイプ
time	TIME
datetime	DATE_TIME
smalldatetime	DATE_TIME
char(10)	STRING
varchar(10)	STRING
nchar(10)	STRING
nvarchar(10)G	STRING
binary(10)	BINARY
varbinary(10)	BINARY
PostGIS	
smallint	SHORT_INTEGER
integer	INTEGER
bigint	LONG_INTEGER
numeric(10, 5)	DOUBLE
real	DOUBLE
double precision	DOUBLE
serial	INTEGER
bigserial	LONG_INTEGER
bytea	BINARY

データベースのデータ タイプ	サポートされるデータ タイプ
date	DATE
time	TIME
timestamp	DATE_TIME
character(10)	STRING
character varying(10)	STRING
nchar(10)	STRING
Oracle	
NUMBER	DOUBLE
CHAR(10)	STRING
VARCHAR(10)	STRING
VARCHAR2(10)	STRING
NCHAR(10)	STRING
NVARCHAR2(10)	STRING
DATE	DATE_TIME
TIMESTAMP	DATE_TIME
BLOB	BINARY
SAP HANA	
tinyint	SHORT_INTEGER

データベースのデータ タイプ	サポートされるデータ タイプ
smallint	SHORT_INTEGER
integer	INTEGER
bigint	LONG_INTEGER
smalldecimal	DOUBLE
decimal(10, 5)	DOUBLE
real	DOUBLE
double	DOUBLE
float(30)	DOUBLE
varchar(30)	STRING
nchar(10)	STRING
nvarchar(30)	STRING
alphanum(30)	STRING
date	DATE
time	TIME
seconddate	DATE_TIM
timestamp	DATE_TIM
varbinary(30)	BINARY

JDBC データベース コネクタの制限事項

- Metadata Insights では、PrestoDB を介した MongoDB/Cassandra コネクタはサポートされていません。MongoDB および Cassandra 用に別途コネクタが用意されています。

- `Write to Any DB` を Presto を介して使用することは Presto DB で推奨されていないため、Presto JDBC コネクタではサポートされていません。

Knox への接続

Apache Knox Gateway を使用すると、Knox セキュリティ レイヤを経由して Hadoop サービスにアクセスできます。

この接続により、Enterprise Big Data モジュールのステージを使用して Enterprise Designer にフローを作成し、Knox 経由で Hadoop との間でデータを読み書きすることができます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注 : デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注 : デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注 : 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、［ゲートウェイ］を選択します。
5. **[ゲートウェイ タイプ]** フィールドで、**[Knox]** を選択します。
6. **[ホスト]** フィールドに、このゲートウェイを実行している HDFS クラスタ内ノードのホスト名または IP アドレスを入力します。
7. **[ポート]** フィールドに、Knox ゲートウェイのポート番号を入力します。
8. **[ユーザ名]** フィールドに、Knox ゲートウェイのユーザ名を入力します。

9. **[パスワード]** フィールドに、Knox ゲートウェイへのアクセスを認証するパスワードを入力します。
10. **[ゲートウェイ名]** フィールドに、アクセスする Knox ゲートウェイの名前を入力します。
11. **[クラスタ名]** フィールドに、アクセスする Hadoop クラスタの名前を入力します。
12. **[プロトコル]** フィールドで、`webhdfs` を選択します。
13. **[サービス名]** フィールドに、アクセスする Hadoop サービスの名前を入力します。
14. 接続をテストするには、**[テスト]** をクリックします。
15. **[保存]** をクリックします。

HDFS クラスタへの Knox 接続を定義した後で、この接続を Enterprise Designer において、**Read from File** ステージと **Write to File** ステージで使用できます。ソースまたはシンク ステージでファイルを定義するときに **[リモート マシン]** をクリックすると、HDFS クラスタを選択できます。

Windows のマッピングされたドライブへの接続

Spectrum™ Technology Platform が Windows サーバーで実行されている場合は、サーバーのマッピングされたドライブ上のデータにアクセスできます。Spectrum™ Technology Platform サーバーは、特定のユーザ アカウント (通常はローカル システム アカウント) によって Windows サービスとして実行されるため、サーバーのスタートアップ プロセスでマッピングされたドライブを定義して、そのマッピングされたドライブを Enterprise Designer と Management Console に表示する必要があります。

1. Spectrum™ Technology Platform サーバーを停止します。
2. Spectrum™ Technology Platform サーバーがインストールされているフォルダの `server\bin\wrapper` に移動します。例えば、`C:\Program Files\Pitney Bowes\Spectrum\server\bin\wrapper` です。
3. ファイル `wrapper.conf` をテキスト エディタで開きます。

   重要： 以下の手順では、このファイルに新しいプロパティを追加します。これらの手順に正確に従って、記載されるプロパティの追加と変更のみを行うことが重要です。このファイルに含まれるそれ以外のプロパティを変更しないでください。

4. 以下の行を追加します。

```
wrapper.share.1.location
wrapper.share.1.target
wrapper.share.1.type
wrapper.share.1.account
wrapper.share.1.password
```

5. `wrapper.share.1.location` プロパティで、マッピングされたドライブの場所を UNC 形式で指定します。

   注： UNC に後続バックスラッシュを含めないでください。

   例を次に示します。

```
wrapper.share.1.location=\\myserver\share
```

6. `wrapper.share.1.target` プロパティで、このマッピングされたドライブに割り当てるドライブ文字を指定します。

   例を次に示します。

```
wrapper.share.1.target=Y:
```

7. `type` プロパティで、`DISK` を指定します。

   例を次に示します。

```
wrapper.share.1.type=DISK
```

8. 接続先の共有がユーザ名とパスワードを要求する場合は、`wrapper.share.1.account` プロパティにユーザ名を指定し、`wrapper.share.1.password` プロパティにパスワードを指定します。

   例を次に示します。

```
wrapper.share.1.account=domain\user123
wrapper.share.1.password=mypassword1
```

   注： Spectrum™ Technology Platform サーバー サービスがローカル システム ユーザによって実行されている場合は、ユーザ名とパスワードを指定できません。共有がユーザ名とパスワードを要求する場合は、別のアカウントで実行するようにサービスを変更する必要があります。

例

以下の例は、`wrapper.conf` ファイルに定義される 2 つのマッピングされたドライブを示しています。

```
wrapper.share.1.location=\\myserver\data
wrapper.share.1.target=Y:
wrapper.share.1.type=DISK
wrapper.share.1.account=sample\user
```

```
wrapper.share.1.password=samplepass
wrapper.share.2.location=\\myserver\moredata
wrapper.share.2.target=Z:
wrapper.share.2.type=DISK
wrapper.share.2.account=sample\user
wrapper.share.2.password=samplepass
```

Marketo への接続

Spectrum™ Technology Platform で Marketo のデータにアクセスするには、Management Console を使って Marketo への接続を定義する必要があります。接続を定義した後は、Marketo に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**Marketo** を選択します。

5. **[エンドポイント URL]** フィールドに、Marketo アカウントのエンドポイント URL を入力します。

   エンドポイント URL を確認するには、Marketo アカウントにログインして **[管理]** > **[統合]** > **[Web サービス]** に移動します。エンドポイント URLは、**[REST API]** という見出しの下に、次の形式で記載されています。

   `https://AccountID.mktorest.com/rest`

   URL の `/rest` の前の部分をコピーします。例えば、`https://AccountID.mktorest.com` です。

6. Marketo アカウントのクライアント ID と秘密鍵を入力します。

   クライアント ID と秘密鍵を確認するには、Marketo アカウントにログインして **[管理]** > **[統合]** > **[LaunchPoint]** > **[API Rest]** > **[詳細の表示]** に移動します。ポップアップ ウィンドウに詳細情報が表示されます。

7. 接続をテストするには、**[テスト]** をクリックします。
8. **[保存]** をクリックします。

Marketo の制限事項

1. 以下のクエリは、`List` エンティティと `Activity_type` エンティティのみに適用されます。それ以外に対しては、フィルタ タイプを指定してください。

   ```
   Select * from Marketo_Table
   ```

   。

2. Marketo は、`Lead` エンティティと `Lead_List` エンティティの結合を除き、結合操作をサポートしていません。`Lead` と `Lead_List` を List_Id で結合するクエリは、次のように記述します。

   ```
   Select Lead.* from Lead Inner Join Lead_List
   On Lead.ID = Lead_List.Lead_ID
   And Lead_List.List_ID = <List ID>
   ```

サポートされているエンティティと操作

以下のタイプのエンティティがあります。

1. エンティティ
2. エンティティ更新: これは、Lead エンティティの更新に使用される仮想テーブルです。例えば、**Merge_Leads** は異なる Marketo Lead の結合に使用します。

Microsoft Dynamics 365 への接続

Microsoft Dynamics 365 Online への接続

Spectrum™ Technology Platform で Microsoft Dynamics 365 Online のデータにアクセスするには、Management Console を使って Microsoft Dynamics 365 Online への接続を定義する必要があります。接続を定義した後は、Microsoft Dynamics 365 Online に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

> **注：** この接続は、Metadata Insights モジュールで使用されます。

Microsoft Dynamics 365 Online への接続を定義するには、以下の手順に従います。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注：** デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注：** デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン  をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   > **注：** 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Microsoft Dynamics 365]** を選択します。
5. **[開発タイプ]** フィールドで、**[オンライン]** を選択します。
6. **[ユーザ名]** フィールドに、Microsoft Dynamics ユーザ名を入力します。
7. **[パスワード]** フィールドに、Microsoft Dynamics パスワードを入力します。

8. **[組織名]** フィールドに、CRM インスタンスの識別に使われる、組織の一意の名前を入力します。

   組織の一意の名前を確認するには、Microsoft Dynamics にログインして **[設定] > [カスタマイズ] > [カスタマイズ] > [開発者リソース]** に移動します。組織の一意の名前が表示されます。

9. **[地域]** フィールドで、Microsoft Dynamics アカウントの地理的な地域を選択します。
10. 接続をテストするには、**[テスト]** をクリックします。
11. **[保存]** をクリックします。

Microsoft Dynamics 365 On Premises への接続

現在、Spectrum は Microsoft Dynamics 365 On Premises のクレームベース認証をサポートしています。

必要条件

証明書をキーストア ファイルにインポートする: Dynamics CRM Server の証明書を Spectrum Java ディストリビューション キーストアにインポートするには、次の操作を行います。

1. サーバーの証明書をローカル フォルダにコピーします。
2. 次のパスを参照して Spectrum JAVA ディストリビューションに移動します: `<SPECTRUM_HOME>\java\jre\lib\security`
3. 次のコマンドで証明書をインポートします: <codeph>keytool -importcert -alias <証明書のエイリアス名> -file " <証明書のパス>\<証明書の名前>" -keystore keystore.jks</codeph> (Windows の場合) または <codeph>keytool -import -alias <証明書のエイリアス名> -file "<証明書のパス>/<証明書の名前>" -keystore keystore.jks</codeph> (Unix の場合)

Microsoft Dynamics 365 On Premises 接続の設定

Spectrum™ Technology Platform を有効化して Microsoft Dynamics 365 On Premise のデータにアクセスするには、Management Console で Microsoft Dynamics 365 OnPremise への接続を設定します。接続を設定した後、Enterprise Designer 内でフローを作成して Microsoft Dynamics 365 On Premise に対するデータの読み書きを行うことができます。

> **注 :** この接続は、Metadata Insights モジュールで使用されます。

Microsoft Dynamics 365 On Premises 接続を設定する手順は次のとおりです。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology

Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注 : デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights:

http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注 : デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注 : 接続をいったん保存すると、名前の変更は不可能になります。

4. **[Microsoft Dynamics 365]** (**[タイプ]**) をクリックします。
5. **[On Premise]** (**[開発タイプ]**) をクリックします。
6. Microsoft Dynamics ユーザ名を **[ユーザ名]** に入力します。
7. Microsoft Dynamics パスワードを **[パスワード]** に入力します。
8. ホストの名前を **[ホスト名]** に入力します。
9. ポートの名前を **[ポート名]** に入力します。
10. STS の URL を**[STS URL]** に入力します。
11. **[テスト]** をクリックして、接続をテストします。
12. **[保存]** をクリックします。

制限事項

以下に制限事項を示します。

1. **作成/更新:** エンティティ内の列が複数のリファレンス エンティティにマッピングされている場合、作成/更新は失敗します。例えば、顧客の 'ParentCustomerId' はアカウント、潜在顧客などに関連付けることができます。これを解決するには、この列のデータの形式を 'GUID' の代わりに 'ReferenceEntityName:GUID' にする必要があります。

サポートされているエンティティと操作

以下のタイプのエンティティがあります。

- ユーザ所有
- 組織所有
- ビジネス所有
- なし

モデル ストアへの接続

データベース、ファイル サーバー、クラウド サービスなど、さまざまなソースから連携したデータを使用するには、モデル ストアに接続します。接続を定義すると、Enterprise Designer の **Read from DB** および **Write to DB** ステージで、モデル ストアの論理モデルと物理モデルのデータ (Metadata Insights で作成および展開) を使用できます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   > **注 :** デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   > **注 :** 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Model Store]** を選択します。

5. **[Model Store]** フィールドに、接続を確立するモデル ストアの名前を入力します。

   使用可能なモデル ストアの名前を検索するには、Metadata Insights を開いて [モデリング] に移動し、**[Model Store]** タブをクリックします。

6. 接続をテストするには、**[テスト]** をクリックします。
7. **[保存]** をクリックします。

   注 : Write to DB ステージをモデル ストア接続で使用すると、*[テーブルの作成]*、*[データを挿入する前にテーブルを切り捨てる]*、*[テーブルが既に存在する場合は破棄して作成し直す]* がサポートされないなど一定の制限があります。

NetSuite への接続

Spectrum™ Technology Platform で NetSuite のデータにアクセスするには、Management Console を使って NetSuite への接続を定義する必要があります。接続を定義した後は、NetSuite に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。NetSuite 接続に対する読み込みと書き出しに対し、インタラクティブ モードとバッチ モードの両方がサポートされています。

注 : この接続は、Metadata Insights モジュールで使用されます。

Spectrum™ Technology Platform では、以下の NetSuite エンティティ タイプがサポートされています。

- 標準レコード
- カスタム レコード
- 保存済み検索
- 標準レコード間の結合

NetSuite に接続するには

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注 : デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[NetSuite]** を選択します。
5. **[電子メール]** フィールドに、接続に使用する NetSuite アカウントにリンクされた電子メールを入力します。
6. **[パスワード]** フィールドに、NetSuite アカウントのパスワードを入力します。
7. **[アカウント]** フィールドに、NetSuite アカウントのユーザ名を入力します。
8. **[役割]** フィールドで、特定の NetSuite ユーザ アカウントにマッピングされた複数の役割から、この接続に対する適切な役割を選択します。

   [役割] フィールドはオプションです。**[役割]** フィールドを空白のままにした場合は、デフォルトの役割が接続を介したログインに使用されます。

   重要： 標準の役割のみがサポートされています。カスタム役割はサポートされていません。

9. 接続をテストするには、**[テスト]** をクリックします。
10. **[保存]** をクリックします。

    注： NetSuite 接続を使用してレコードを `INSERT` するには、プライマリ キー (`internalId`) を空白にして `UPSERT` クエリを使用します。

NetSuite の制限事項

1. 結合を使用してクエリを実行する場合は、具体的な列を指定する必要があります。例えば、以下のクエリはサポートされていません。

```
select * from CUSTOMER_M
```

2. NetSuite への同時接続はサポートされていません。NetSuite では、1 つのアカウントにつき単一のログインしか許可されないためです。

3. Standard (標準) と Custom (カスタム) のレコードしか書き込むことはできません。
4. `UPDATE`クエリと `UPSERT` クエリの双方では、`UPSERT` 操作が実行されます。
5. Write to DB ステージで許容される最大バッチ サイズは、`insert`操作で 200、`update` 操作で 100 です。
6.

サポートされているエンティティと操作

以下のタイプのエンティティがあります。

- 標準レコード
- カスタム レコード
- 結合
- 保存済み検索

注： NetSuite 接続テーブルでは、プライマリ キー列は `internalId` です。

NoSQL への接続

以下の種類の NoSQL データベースがサポートされています。

- Couchbase
- MongoDB

Read from Hadoop Sequence File、**Write to Hadoop Sequence File**、**Read From File**、**Write to File**、**Read From XML**、**Write to XML**、**Read from Hive File**、**Write to Hive File**、**Read From HL7 File** などのステージを **Enterprise Designer** で使用するには、Hadoop システムに接続します。

重要： Spectrum™ Technology Platform は、Windows プラットフォーム上の Kerberos 認証に対して *Hadoop 2.x* をサポートしません。

- Query NoSQL DB
- Read from NoSQL DB
- Write to NoSQL DB

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、次のいずれかを選択します。
   - Couchbase
   - MongoDB
5. アクセスする特定の NoSQL データベースの **[ホスト]**、**[ポート]**、**[データベース]**、**[ユーザ名]**、および **[パスワード]** を指定します。
6. **[テスト]** をクリックして、データベースに正しく接続されていることを確認します。
7. **[OK]** をクリックします。

Salesforce への接続

Spectrum™ Technology Platform で Salesforce のデータにアクセスするには、Management Console を使って Salesforce への接続を定義する必要があります。接続を定義した後は、Salesforce に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[リソース] > [データ ソース] に移動します。

Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注： デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Salesforce]** を選択します。
5. **[ユーザ名]** フィールドに、Salesforce データ ストアに登録されている電子メール ID を入力します。
6. **[パスワード]** フィールドに、Salesforce ポータルのパスワードと、Salesforce ポータルによって生成されたセキュリティ トークンの組み合わせを入力します。

   例えば、パスワードが Sales@Test で、Salesforce によって与えられたセキュリティ トークンが 56709367 である場合、この Salesforce 接続を認証するためのパスワードは Sales@Test56709367 となります。

7. 接続をテストするには、**[テスト]** をクリックします。
8. **[保存]** をクリックします。

   注： 監査フィールドは、デフォルトですべてのテーブルに対して有効です。Salesforce には、次の監査フィールドがあります。

   - 作成日
   - 最終更新日
   - 作成者
   - 最終更新者

重要： Salesforce 接続を使用して Spectrum™ Technology Platform バージョン 10 以前で作成された Physical Model のテーブルに対して、監査フィールドを有効にするには、モデルを開いて保存し直す必要があります。

Salesforce の制限事項

集約関数は Model Store に対するクエリの実行中はサポートされません。

SAP NetWeaver への接続

Management Console で OData サービスを使用して SAP NetWeaver 接続を作成すると、CRM データや ERP データの読み込み、書き出し、同期が可能です。SAP 接続に対する読み込みと書き出しに対し、インタラクティブ モードとバッチ モードの両方がサポートされています。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[SAP]** を選択します。
5. **[ユーザ名]** フィールドに、SAP Web サービスにアクセスするユーザ名を入力します。
6. **[パスワード]** フィールドに、SAP Web サービスのパスワードを入力します。

7. [OdataURL] フィールドに、この接続に対して使用する Odata Web サービスのアドレスを入力します。
8. **[テスト]** をクリックします。
   接続のテストが正常に終了したことを示すメッセージが表示されます。
9. **[保存]** をクリックします。
   接続が正常に作成されたことを示すメッセージが表示されます。

   注： 取得操作を実行するには、OData サービスが `$skip` 操作と `$top` 操作をサポートしている必要があります。サービスがこれらの操作をサポートしない場合、取得されたレコードは Model Store のプレビューにおいて矛盾を示します。

SAP NetWeaver の制限事項

`UPDATE` と `UPSERT` の両方の操作に対し、`UPDATE` 操作が実行されます。

サポートされているエンティティと操作

次の 2 タイプのエンティティがあります。

- ネイティブ: ネイティブのデータ タイプを持つ列は、それぞれのデータ タイプで表示されます。
- カスタム定義: カスタム定義のデータ タイプを持つ列は、空白のデータ タイプで表示されます。

SAP 接続に基づくモデル ストアを展開するには、その論理モデルと物理モデルに、ネイティブなデータ タイプの列を持つエンティティしか含まれないようにしてください。モデルにカスタム定義のデータ タイプを持つエンティティがあると、モデル ストアは展開できません。

SharePoint への接続

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サー

バーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

注 : デフォルトの HTTP ポートは 8080 です。

[接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注 : 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[クラウド]** を選択します。
5. **[クラウド サービス]** フィールドで、**[Sharepoint]** を選択します。
6. **[バージョン]** フィールドで、**v2010** を選択します。Spectrum™ Technology Platform は現在、Sharepoint バージョン 2010 をサポートしています。
7. **[プロトコル]** フィールドで、Sharepoint の接続に必要なプロトコルを選択します。
8. **[サーバー アドレス]** フィールドに、接続する SharePoint サーバーのホスト名または IP アドレスを入力します。
9. SharePoint の認証に使用するユーザ名とパスワードを入力します。
10. **[プロジェクト]** フィールドに、アクセスする Sharepoint ロケーションを含む特定のプロジェクトを入力します。
11. 接続をテストするには、**[テスト]** をクリックします。
12. **[保存]** をクリックします。

例

例えば、次の SharePoint URL への接続を作成するとします。

```
https://sharepoint.example.com/sites/myportal
```

[プロトコル]、**[サーバー アドレス]**、**[プロジェクト]** の各フィールドを次のように設定します。

- **プロトコル**: https
- **サーバー アドレス**: `sharepoint.example.com`
- **プロジェクト**: myportal

Splunk への接続

Spectrum™ Technology Platform で Splunk のデータにアクセスするには、Management Console を使って Splunk への接続を定義する必要があります。接続を定義した後は、Splunk に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Splunk]** を選択します。
5. **[ユーザ名]** フィールドに、Splunk インスタンスを認証するための Splunk アカウント ユーザ名を入力します。
6. **[パスワード]** フィールドに、Splunk アカウントのパスワードを入力します。
7. **[ホスト]** フィールドに、Splunk データ ソースがホストされているサーバーのアドレスまたはホスト名を入力します。
8. **[ポート]** フィールドに、Splunk データ ソースのポート番号を入力します。
9. 接続をテストするには、**[テスト]** をクリックします。

10. **[保存]** をクリックします。

Splunk の制限事項

以下のクエリはサポートされていません。

```
select count(*) from SplunkTable
```

サポートされているエンティティと操作

サポートされている操作

`LIKE`、`ORDER BY`、`LIMIT`、`IN`、`BETWEEN`、`!=`、`<=`、 `>=`、`<`、`>`、複数の AND/OR 演算子

サポートされている関数

- 文字列関数: `upper`、`lower`、`length`、`len`、`ltrim`、`rtrim`、`substring`、`max`、`min`
- 数学関数: `abs`、`ceil`、`exp`、`floor`、`sqrt`、`round`

> **注 :** その他すべてのクエリ操作については、以下で説明するように Splunk `search` 列を使用します。

Spectrum™ Technology Platform では、Splunk テーブル内に列 `search` を提供します。これによって、Splunk 接続で必要なデータを検索することができます。

`SplunkTable` に対して `select` クエリを実行する際に、次のどちらの目的にも `search` 列を `where` 句で使用できます。

1. ANSI SQL 構文では指定できない検索条件を含める。
2. メインの SQL クエリの一部としては含められない Splunk 固有の検索条件を含める。

例えば、以下のクエリは、値が `ACC` であるキー `opp` を含む `_raw` 値を検索します。

```
select "_raw" from SplunkTable where "search"='search opp=ACC'
```

SugarCRM への接続

Spectrum™ Technology Platform で SugarCRM のデータにアクセスするには、Management Console を使って SugarCRM への接続を定義する必要があります。接続を定義した後は、SugarCRM に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。SugarCRM のオンライン版とオンプレミス版の両方がサポートされています。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン + をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[SugarCRM]** を選択します。
5. SugarCRM のユーザ名とパスワードを入力します。
6. **[URL]** フィールドに、この接続で使用する SugarCRM アカウントの URL を入力します。
7. SugarCRM アカウントの **[クライアント ID]** と **[クライアント シークレット]** を入力します。
8. 接続をテストするには、**[テスト]** をクリックします。
9. **[保存]** をクリックします。

SugarCRM の制限事項

1. `UPDATE` クエリと `UPSERT` クエリの双方では、`UPSERT` 操作が実行されます。
2. 接続の **[物理モデル スキーマ]** に表示されるテーブル プロパティの **[Null 可]** 列と **[更新可能]** 列は、正しい操作を表していない場合があります。例えば、更新可能となっていない列を更新しようとしてもシステム例外が発生しなかったり、逆に、Null 可とマークされている列に Null を設定すると例外が発生したりすることがあります。
3. 結合を使用してクエリを実行する場合は、エイリアスを使用する必要があります。

サポートされているエンティティと操作

サポートされている操作

`LIKE` (その操作は指定された値で始まる取得オプションに制限されています。例えば、ステートメント `WHERE name LIKE 's%'` はアルファベット S で始まるすべての名前を取得します)、`ISNULL`、`IS NOT NULL`、`IN`、`NOT IN`、`>`、`>=`、`<`、`<=`、`=`、`<>`、`AND`、`OR`

Oracle Eloqua への接続

Spectrum™ Technology Platform で Oracle Eloqua のデータにアクセスするには、Management Console を使って Oracle Eloqua への接続を定義する必要があります。

。接続を定義した後は、Eloqua に対してデータの読み書きを行うフローを Enterprise Designer で作成できます。

注： この接続は、Metadata Insights モジュールで使用されます。

1. 次のいずれかのモジュールを使用して **[データ ソース]** ページにアクセスします。

   Management Console: http://*server*:*port*/managementconsole という URL を使用して Management Console にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [リソース] > [データ ソース] に移動します。

   Metadata Insights: http://*server*:*port*/metadata-insights という URL を使用して Metadata Insights にアクセスします。ここで、*server* は Spectrum™ Technology Platform サーバーのサーバー名または IP アドレス、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。

   注： デフォルトの HTTP ポートは 8080 です。

   [接続] に移動します。

2. [接続を追加] ボタン をクリックします。
3. **[名前]** フィールドに、接続の名前を入力します。任意の名前にすることができます。

   注： 接続をいったん保存すると、名前の変更は不可能になります。

4. **[タイプ]** フィールドで、**[Oracle Eloqua]** を選択します。
5. **[サイト名]** フィールドに会社名と同じ名前を入力します。
6. **[ユーザ名]** フィールドにユーザ名を入力します。
7. **[パスワード]** フィールドにパスワードを入力します。
8. **[テスト]** をクリックして、接続をテストします。
9. **[保存]** をクリックします。

特殊な操作

1. 連絡先リスト内の連絡先を取得するには、次の結合クエリを使用します。

```
select * from Contacts inner join ContactListMembers on
Contacts.Eloqua_Contact_ID = ContactListMembers.Contact_Id where
ContactListMembers.ContactList_Id = '<id>'
```

   連絡先セグメント内の連絡先を取得するには、次の結合クエリを使用します。

```
select * from Contacts inner join ContactSegmentMembers on
Contacts.Eloqua_Contact_ID = ContactSegmentMembers.Contact_Id where
ContactSegmentMembers.Contactlist_Id = '<id>'
```

2. 連絡先リストに連絡先を挿入するには、次のステートメントを使用します。

```
insert into ContactListMembers (ContactList_ID,Contact_ID) values
('<contactlist_id>','<contact_id>')
```

3. 連絡先リストから連絡先を削除するには、次のステートメントを使用します。

```
delete from ContactListMembers where ContactList_ID =
'<contactlist_id>' and Contact_ID = '<contact_id>'
```

制限事項

以下に制限事項を示します。

1. **作成/更新:**
   a. Null でない列が空欄または存在しない場合、`Insert/Upsert` (挿入/アップサート) に失敗します。
   b. 特定のバッチで `Unique` (ユニーク) 列の値が一意でない場合、`Insert/Upsert` (挿入/アップサート) に失敗します。
   c. ロールバックの例外を開扉するためには、**[コミットするバッチ数]** の値を 1 のままにしておきます。

2. **読み込み:**
    a. カスタム エンティティでは、`Select` (選択) の操作が連絡先エンティティとの結合に対してのみ適用されます。
3. **Filter:**
    a. サポートされているフィルタは =、!=、>、<、>=、<= です。
    b. 複数の値を指定した場合の `IN` および `NOT IN` 条件演算子は一切サポートされていません。
    c. エンティティ間の `Joins` (結合) は一切サポートされていません。
    d. `OR` 条件演算子は、アカウントと連絡先のエンティティでのみサポートされます。
    e. **AND** 条件演算子は、2 つの条件の間でのみ使用できます。
    f. = フィルタは、`timestamp` データ タイプを持つフィールドに対して常に機能するわけではありません。

サポートされているエンティティと操作

以下のエンティティがサポートされています。

- **エンティティ:** ビジネス エンティティを表すテーブルを示します。
- **アクティビティ:** 何らかのアクティビティに基づいてデータが生成されるビジネス エンティティを表すテーブルを示します。
- **カスタム エンティティ:** コネクタで提供されている特殊な操作の一部として使用されるエンティティを示します。

このテーブルには、エンティティと、それらに対してサポートされている操作がリストされています。

エンティティ名	作成	読み込み	更新	削除	バッチのサポート	最大バッチ サイズ
アカウント	X	X	X	X	挿入/更新*	1000
アカウント グループ		X				
キャンペーン		X				
連絡先	X	X	X	X	挿入/更新*	1000
連絡先リスト	X	X	X	X		

エンティティ名	作成	読み込み	更新	削除	バッチのサポート	最大バッチ サイズ
連絡先セグメント	X	X	X	X		
電子メール		X				
電子メール フォルダ		X				
電子メール グループ		X				
マイクロサイト		X				
ユーザ		X				
訪問者		X				
アクティビティ						
電子メール オープン		X				
電子メール クリックスルー		X				
電子メール送信		X				
購読		X				
購読解除		X				
バウンスバック		X				
Web 訪問		X				
ページビュー		X				
フォーム送信		X				
カスタム エンティティ						

エンティティ名	作成	読み込み	更新	削除	バッチのサポート	最大バッチ サイズ
連絡先リスト メンバー	X	X		X	挿入/削除	1000
連絡先セグメント メンバー		X				

* 更新操作は挿入として機能します。

クラウド ファイル サーバーの圧縮のサポート

Amazon S3、Google クラウド ストレージ、MS Azure Blobstore の各ファイル サーバーは、`gzip (.gz)` と `zip (.zip)` の圧縮形式をサポートしています。

Spectrum™ Technology Platform は、ファイル サーバーに対して読み書きするファイルの圧縮と解凍を処理します。

> **注：** 同じファイル サーバーを、ファイルの通常の読み書きと、ファイルの圧縮および解凍の両方に使用できます。

圧縮形式ファイルの読み取り

サーバーからファイルを読み取るとき、その圧縮形式は、サーバーから受け取ったメタデータ キー プロパティ `Content-Encoding` から得られます。

圧縮形式ファイルの書き込み

サーバーにファイルを書き込むときには、必要な圧縮形式として `.gz` または `.zip` を指定します。ファイルは、指定された圧縮拡張子に基づいて圧縮されます。

メタデータ キー プロパティ `Content-Encoding` も、選択された圧縮形式に基づいて設定されます。このプロパティ値は、ファイルの書き込み時にクラウド ファイル サーバーに引き渡されます。

接続の削除

以下の任意のモジュールを使用して接続を削除できます。

- Management Console
- Metadata Insights

1. 必要なモジュールの **[データ ソース]** ページにアクセスします。
   - Management Console で **[リソース]** > **[データ ソース]** をクリックします。
   - Metadata Insights で、**[接続]** をクリックします。
2. 削除する接続の横にあるチェック ボックスをオンにして、**[削除]** ボタン  をクリックします。

操作方法ビデオ - 接続の設定

このビデオでは、さまざまなタイプのデータ ソースに接続し、それらを Spectrum Technology Platform で使用する方法をご紹介します。ビデオをご覧ください。

5 - Spectrum のデータベース

このセクションの構成

Spectrum データベースの概要

Spectrum データベースには、信頼できるデータ プロバイダから提供され、データの拡張と検証に使用される参照データが格納されています。例えば、Spectrum™ Technology Platform の住所検証では、郵便当局の正式な住所データを使用して、ユーザの住所を、レコードの住所と照合します。Spectrum データベースを使用した処理タイプの例として、ジオコーディング、ルーティングのほか、特定の住所に対する税務管轄区域の割り当てなどが挙げられます。

弊社では Spectrum データベースを定期的に更新し、サードパーティ データ プロバイダからの最新データをユーザに提供します。データベースの更新は、ソフトウェアのアップデートとは独立して行われ、四半期ごとの場合もあれば、毎月行われる場合もあります。データベースが更新された場合は、更新されたデータベースをダウンロードするためのリンクを含む電子メールによって、ユーザへの通知が行われます。提供されている最も正確なデータを使用するために、できる限り早くこれをインストールすることをお勧めします。

Spectrum データベースを使用するのは一部のモジュールのみです。Spectrum データベースを使用するモジュールには、Enterprise Tax モジュール、Enterprise Geocoding モジュール、Global Sentry モジュール、Universal Addressing モジュール、ルーティング (Spatial and Routing モジュールの機能) などがあります。Spectrum データベースを使用するモジュールがインストールされているかどうかを確認するには、Management Console を開いて **[リソース] > [Spectrum データベース]** に移動し、[追加] ボタン  をクリックします。Spectrum データベースを使用するモジュールがインストールされていれば、**[モジュール]** フィールドに表示されます。

Spectrum データベースのインストール

Spectrum データベースには、信頼できるデータ プロバイダから提供され、データの拡張と検証に使用される参照データが格納されています。例えば、Spectrum™ Technology Platform の住所検証では、郵便当局の正式な住所データを使用して、ユーザの住所を、レコードの住所と照合します。Spectrum データベースを使用した処理タイプの例として、ジオコーディング、ルーティングのほか、特定の住所に対する税務管轄区域の割り当てなどが挙げられます。

弊社では Spectrum データベースを定期的に更新し、サードパーティ データ プロバイダからの最新データをユーザに提供します。データベースの更新は、ソフトウェアのアップデートとは独立して行われ、四半期ごとの場合もあれば、毎月行われる場合もあります。データベースが更新された場合は、更新されたデータベースをダウンロードするためのリンクを含む電子メールによっ

て、ユーザへの通知が行われます。提供されている最も正確なデータを使用するために、できる限り早くこれをインストールすることをお勧めします。

1. リリースの通知またはウェルカム メールに記載されているリンクを使用して、Pitney Bowes からライセンス済みの SPD ファイルをダウンロードします。
2. `.spd` ファイルを次の場所に配置します。

   SpectrumLocation`/server/app/dataimport`

SPD ファイルは次の場所に自動的に抽出されます。

SpectrumLocation`/server/app/repository/datastorage`

`datastorage` フォルダに抽出されデータベースは、インストールされ、Management Console を使用してデータベース リソースとして定義できます。Management Console の詳細については、『管理ガイド』を参照してください。

必要に応じて、Spectrum データベースのインストール処理を次のように変更できます。

- データをインポートするディレクトリの場所を変更するには、`<SpectrumLocation>/server/app/conf/dataimportdirectories.properties` ファイルの **platform** プロパティを変更します。
- データを格納するフォルダの場所を変更するには、`<SpectrumLocation>/server/app/conf/spectrum-container.properties` ファイルの Data Manager 設定セクションにある **spectrum.data.manager.storage.directory** プロパティを変更します。ストレージ フォルダは、Spectrum のアンインストール時に削除されないよう、Spectrum の外側に作成することを検討してください。
- デフォルトでは、SPD ファイルの展開後や Spectrum のアンインストール時に SPD ファイルは削除されます。ただし、`<SpectrumLocation>/server/app/conf/spectrum-container.properties` ファイルの Data Manager 設定セクションにある **spectrum.data.manager.archive.data** プロパティを "true" に設定することによって、SPD ファイルをアーカイブすることができます。
- プロパティ ファイルを変更した場合は、Spectrum サーバーを停止して再起動する必要があります。

Spectrum データベースの追加

Spectrum データベースには、住所の検証で使用する住所データや、ジオコーディングで使用する空間データなどの参照データが含まれます。Spectrum データベース リソースを追加する場合は、使用している特定のモジュール用の手順を参照してください。

データベースのプール サイズと実行時インスタンス数

ほとんどの Spectrum™ Technology Platform 環境において、バッチ ジョブであるか、Web サービスまたは API 要求に応答するサービスであるかの違いはあれ、複数のフローが同時に実行しています。同時処理を最適化するために、データベースのプール サイズ設定が使用できます。これによって、Spectrum のデータベースが処理する同時要求数と、同時に実行するフロー ステージのインスタンス数を制御する実行時インスタンス数が制限されます。最適なパフォーマンスを得るには、これら 2 つの設定を同時にチューニングする必要があります。

データベースのプール サイズ

Spectrum データベースには、住所の検証で使用する郵便データや、住所のジオコードで使用するジオコーディング データなど、特定のステージで使用するリファレンス データが含まれます。これらのデータベースは、それを使用するデータフロー ステージやサービスから同時に行われる複数の要求を受け付け、データフローの要求やサービスの要求のパフォーマンスを高めるように設定することが可能です。データベースのプール サイズは、Spectrum のデータベースが処理する同時要求の最大数を設定します。デフォルトでは、Spectrum データベースのプール サイズは 4 で、データベースは 4 つの要求を同時に処理できることを意味します。

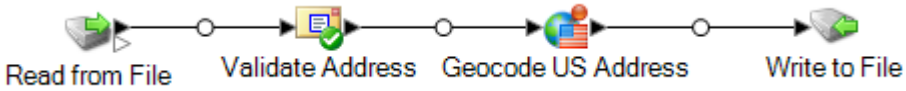
最適なプール サイズはモジュールによって異なります。一般的には、サーバーが搭載する CPU の数の半分から 2 倍のプール サイズを設定すると、最適な結果が得られます。ほとんどのモジュールに最適なプール サイズは CPU 数と同数です。例えば、サーバーが 4 つの CPU を搭載している場合は、プール サイズを 2 (CPU 数の半分) ～ 8 (CPU 数の 2 倍) の間で試すことができ、多くの場合、最適なサイズは 4 (CPU 数と同数) です。

プール サイズを変更するときは、データベースにアクセスするステージ用としてデータフローに指定されている実行時インスタンスの数を考慮する必要もあります。例えば、1 つの実行時インスタンスを使用するように設定された Geocode US Address ステージを持つデータフローがあるとします。この場合、米国ジオコーディング データベースのプール サイズを 4 に設定しても、パフォーマンスは向上しません。実行時インスタンスが 1 つしかないので、データベースへの要求

は一度に 1 つになります。ただし、Geocode US Address の実行時インスタンスの数を 4 つに増やすと、パフォーマンスが向上します。データベースに同時にアクセスする Geocode US Address のインスタンスが 4 つあるので、プール全体を使用できます。

実行時インスタンス

データフローの各ステージはそれぞれのスレッドで非同期に動作し、他のステージから独立しています。このため、データフロー内の複数ステージが並列処理され、1 つのステージに対して複数の実行時インスタンスを利用することができます。これは、データの処理時間が異なるステージで構成されるデータフローで役に立ちます。その結果、スレッド間での作業の分配のバランスが悪くなる可能性があります。例えば、次の 2 つのステージで構成されるデータフローを考えてみましょう。



ステージの設定によって、Validate Address ステージが Geocode US Address ステージより速くレコードを処理することがあります。その場合、データフロー実行のある時点で、Validate Address はすべてのレコードを処理していますが、Geocode US Address にはまだ未処理のレコードがあります。このデータフローのパフォーマンスを向上させるには、最も速度の遅いステージ(この場合は Geocode US Address) のパフォーマンスを向上させる必要があります。その方法の 1 つは、ステージの実行時インスタンスを複数指定することです。例えば、実行時インスタンス数を 2 に設定すると、そのステージのインスタンスが2つになります。各インスタンスはそれぞれのスレッド内で動作し、レコードの処理に使用することができます。

一般的なルールとして、実行時インスタンス数は、少なくともリモート コンポーネントのインスタンス数と等しくなければなりません。リモート コンポーネントの詳細については、『*管理ガイド*』を参照してください。複数の実行時インスタンスを指定するとパフォーマンスが向上しますが、この値を高くしすぎると、システムのリソースに負荷がかかり、パフォーマンスが低下する可能性があります。

> **注：** 複数の実行時インスタンスを使用してパフォーマンスが向上するのは、複数のレコードを使用するジョブまたはサービス要求を実行する場合のみです。

チューニング手順

データベースのプール サイズと実行時インスタンス数に適切な値を設定するには、さまざまな設定を試して、リソースに過度に負荷をかけたりパフォーマンスを低下させたりすることなく、使用可能なサーバー リソースを最大限に活用できる設定を見つける必要があります。

> **注：** データベースのプール サイズを調整する前にデータフローのプール サイズを最適化する必要があります。データフローのプール サイズの最適化については、「データフローのプール サイズ（232ページ）」を参照してください。

1. まず、さまざまな設定を試す際に使用するサンプル データを見つけます。実行時間の測定や一貫性の確認ができるように、サンプル データセットは十分に大きい必要があります。またサンプル データは、実際に処理するデータを代表するものでなければなりません。例えば、ジオコーディングのパフォーマンス テストを行う場合は、ジオコードする予定のすべての国に対して同数のレコードがテスト データに含まれるようにします。
2. 郵便データベースやジオコーディング データベースなど、データベース リソースの使用が必要なサービスまたはデータフローをテストする場合は、データベースの最新版がインストールされていることを確認してください。
3. サンプル データを用意し、最新のデータベース リソースをインストールしたら、ファイルからデータを読み込み、最適化したいステージでそれを処理し、ファイルに書き出す、簡単なデータフローを作成します。例えば、Validate Address のパフォーマンス設定をテストする場合は、Read from File、Validate Address、Write to File で構成されるデータフローを作成します。
4. データベース リソースのプール サイズを 1 に設定します。
   a. Management Console を開きます。
   b. **[リソース] > [Spectrum データベース]** に移動します。
   c. 最適化するデータベース リソースを選択し、変更ボタン  をクリックします。
   d. **[プール サイズ]** フィールドに、1 と入力します。
   e. **[OK]** をクリックします。
5. ステージの実行時インスタンスを 1 に設定します。
   a. Enterprise Designer でデータフローを開きます。
   b. 複数の実行時インスタンスを使用するように設定するステージをダブルクリックします。
   c. **[実行時]** をクリックします。

      注 : すべてのステージで複数の実行時インスタンスを使用できるわけではありません。ステージのウィンドウの下部に **[実行時]** ボタンがない場合、そのステージでは複数の実行時インスタンスを使用できません。

   d. **[ローカル]** を選択して 1 を指定します。
   e. **[OK]** をクリックして **[実行時パフォーマンス]** ウィンドウを閉じます。さらに **[OK]** をクリックしてステージを閉じます。
6. データフローを複数回実行し、次の各項目の平均値を記録することによって、ベースライン パフォーマンスを算出します。
   - 経過時間
   - CPU 使用率
   - メモリ使用率

ヒント： JMX コンソールを使用してパフォーマンスをモニタリングできます。詳細については、**JMX コンソールによるパフォーマンスのモニタリング**（249ページ）を参照してください。

7. ジョブの複数インスタンスの同時実行をサポートする必要がある場合は、それを実行します。それぞれの場合に対して、経過時間、CPU 使用率、メモリ使用率を記録します。

   ヒント： ファイル モニターを使用して、ジョブの複数インスタンスを同時に実行できます。詳細については、**コントロール ファイルによるフローのトリガー**（201ページ）を参照してください。

8. データベース リソースのプール サイズと、ステージ実行時インスタンスの設定値を増加させます。
9. サーバーを再起動します。
10. データフローを再度実行し、経過時間、CPU 使用率、メモリ使用率を記録します。
11. パフォーマンスが低下し始めるまで、データベース リソースのプール サイズとステージ実行時インスタンスの設定値を増加していきます。
12. ジオコーディング パフォーマンスをテストする場合は、単一国と複数国の入力を使用してこの手順を繰り返します。

データベース プロパティの設定

Spectrum™ Technology Platform は、リモート / 接続済みデータベースにあるデータにアクセスし、そのデータを処理するためにメモリに読み込むことができます。Management Console の **[データベースの追加]** ウィンドウでは、UI を使ってデータベースを設定できます。

データベース接続を設定するには:

1. `http://server.port/managementconsole` という URL を使用して Management Console にアクセスします。ここで、*server* はサーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform で使用される HTTP または HTTPS ポートです。デフォルトのポート番号は 8080 です。
2. **[リソース] > [Spectrum データベース]** に移動します。
3. **[データベースの追加]** ボタンを選択して、**[データベースの追加]** ウィンドウを表示します。
4. [名前] フィールドに、わかりやすい名前 (**Q1 Addressing Job Database** など) を入力して、データベースに名前を付けます。
5. データベースの **[プール サイズ]**、**[最小メモリ]**、および **[最大メモリ]** を設定します。
   - **[最大メモリ]** は 0 より大きい値を指定する必要がありますが、65336 MB (64GB) を超えることはできません。

- **[最小メモリ]** は **[最大メモリ]** 以下の値でなければなりません。
- **注：** 詳細については、「データベースのプール サイズと実行時インスタンス数」を参照してください。

  注：[プール サイズ]、[最小メモリ]、および [最大メモリ] の値を定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。

6. **[モジュール]** で、このデータベースを適用する Spectrum™ Technology Platform モジュールを選択します。使用可能なデータベース タイプが **[タイプ]** 選択リストに表示されます。
7. 設定するデータベースを選択して、**[必須データベース]** および **[オプションのデータベース]** ドロップダウン リストに選択肢が設定されるようにします。オプションのデータベースが 1 つもない場合は、[オプションのデータベース] ドロップダウン リストに選択肢は表示されません。
8. 必要に応じて、**[詳細設定をオーバーライド]** チェックボックスをオンにして、Java プロパティや環境変数を調整するか、コマンド ライン処理引数を指定します。

詳細設定をオーバーライド

この機能は、**[Spectrum データベース] > [データベースの追加]** ウィンドウで使用できます。この機能を使用すると、処理を参照するためのプロパティ ファイルを設定しなくても、Management Console でその他のコマンド ライン引数を定義できます。この方法は、特にクラスタ環境において、データベースの設定を Spectrum™ Technology Platform 全体について定義し、維持することを容易にします。

注： これらの設定を変更する際は十分に注意してください。影響が広い範囲に及ぶことがあります。

新しいプロパティまたは変数の追加

Java プロパティまたは環境変数を追加するには:

1. **[Java プロパティ]** または **[環境変数]** を選択して、既存のプロパティのリストを展開します。
2. **[追加]** ボタン ✚ を選択して、空白の入力行を表示します。
3. 新しいエンティティの **[名前]** と **[値]** を追加します。
4. **Enter** キーを押して新しいエンティティを保存します。

プロパティまたは変数の削除

Java プロパティまたは環境変数を削除するには:

1. **[Java プロパティ]** または **[環境変数]** を選択して、既存の定義のリストを展開します。
2. 削除するエンティティの横にあるチェックボックスをオンにします。
3. エンティティのリストの上にある **[削除]** ボタン をクリックして設定を削除します。

注： 削除ボタンを選択しても、確認メッセージは表示されません。

コマンド ライン プロパティの指定

このフィールドを使用して、メモリに関連しない、Java プロパティでは表現できない設定であるコマンド ライン プロパティを定義します。

注： これらの設定を変更する際は十分に注意してください。影響が広い範囲に及ぶことがあります。

Spectrum データベースの削除

Spectrum データベースには、住所の検証で使用する住所データや、ジオコーディングで使用する空間データなどの参照データが含まれます。システムで使用されていない Spectrum データベースは、削除できます。例えば、データベースの更新バージョンをインストールした後には、Spectrum データベースを削除したいと考える可能性があります。

重要： リソースを削除する前に、それを使用するジョブまたはサービスが存在しないことを確認してください。ジョブまたはサービスによって参照されるリソースを削除すると、それらのジョブまたはサービスは機能しなくなります。

1. Management Console を開きます。
2. **[リソース] > [Spectrum データベース]** に移動します。
3. 削除する Spectrum データベースの横にあるチェックボックスをオンにして、削除ボタン をクリックします。

Spectrum データベース リソースを削除しても、データベース ファイルそのものは削除されません。システム上の領域を解放するには、リソースを削除した後にデータベース ファイルを削除する必要があります。

6 - サービス

このセクションの構成

Spectrum サービス

サービスとは、REST や SOAP Web サービス インターフェイス、または Spectrum™ Technology Platform API を介してアクセスする処理機能です。1 つ以上のレコードをサービスに渡し、レコードを処理する際に使用するオプションを任意で指定できます。サービスはデータを処理し、そのデータを返します。

一部のサービスは、モジュールをインストールしたときに使用可能になります。例えば、Universal Addressing モジュールをインストールすると、お使いのシステム上で ValidateAddress サービスが使用可能になります。Enterprise Designer でサービスを作成し、そのサービスをシステム上でユーザ定義サービスとしてエクスポーズしなければならない場合もあります。例えば、Spatial モジュールのステージは、このモジュールのステージを使用するサービスをまず作成しなければ、サービスとして使用できません。

デフォルト サービス オプションの指定

デフォルト サービス オプションは、システム上の各サービスのデフォルトの動作を制御します。サービス内の各オプションのデフォルト値を指定できます。デフォルト オプション設定は、API 呼び出しまたは Web サービス リクエストで、オプションの値が明示的に定義されていない場合に有効です。また、デフォルト サービス オプションは、Enterprise Designer でこのサービスを使用してフローを作成する際にデフォルトとして使用される設定です。

> **注：** サービスの場合、デフォルト値はサービスを最初にエクスポーズする前にしか変更できません。サービスをエクスポーズした後は、Enterprise Designer を使用してデフォルト値を変更することはできません。代わりに、Management Console を使用する必要があります。

1. Management Console を開きます。
2. **[サービス]** をクリックします。
3. 必要なサービスの横にあるチェックボックスをオンにして、編集ボタン  をクリックします。
4. サービスのオプションを設定します。サービスのオプションの詳細については、そのサービスのモジュールのソリューション ガイドを参照してください。
5. **[保存]** をクリックします。

サービスのプレビュー

Management Console でサービスの [プレビュー] タブを使うと、サービスの結果をプレビューできます。プレビューは、どのオプションを指定するかを決める際に便利です。異なるオプションが、サービスから返されるデータに与える影響を直ちに確認できるためです。

1. Management Console を開きます。
2. **[リソース]** メニューを開き、プレビューするサービスを選択します。
3. **[プレビュー]** タブをクリックします。
4. テスト用のデータを各フィールドに入力します。

   プレビューを使用する際には、次の点に注意してください。

   - すべてのフィールドにデータを入力する必要はありません。フィールドを空のままにすると、空の文字列がプレビューに使用されます。
   - フィールドに null 値を引き渡した場合の影響をプレビューするには、そのフィールドの横の無効アイコンをクリックします。

   - 複数のレコードを同時にプレビューできます。レコードを追加するには、追加ボタン  をクリックします。
   - テスト用のデータはファイルからインポートできます。データをインポートするには、インポート ボタン  をクリックします。次のことに注意してください。

     - ファイルの最初の行は、ヘッダー レコードでなければなりません。ヘッダー内のフィールド名は、サービスで必要とされるフィールド名と一致する必要があります。
     - フィールドの区切り文字がスペースの場合、フィールド値を引用符で囲む必要があります。スペースをフィールドの区切り文字に使うファイルの例を以下に示します。

```
AddressLine1 AddressLine2 City StateProvince PostalCode
"One Global View" "" "Troy" "NY" "12180"
"3001 Summer St" "" "Stamford" "CT" "06926"
"224 N Michigan Ave" "Suite 300" "Chicago" "IL" ""
```

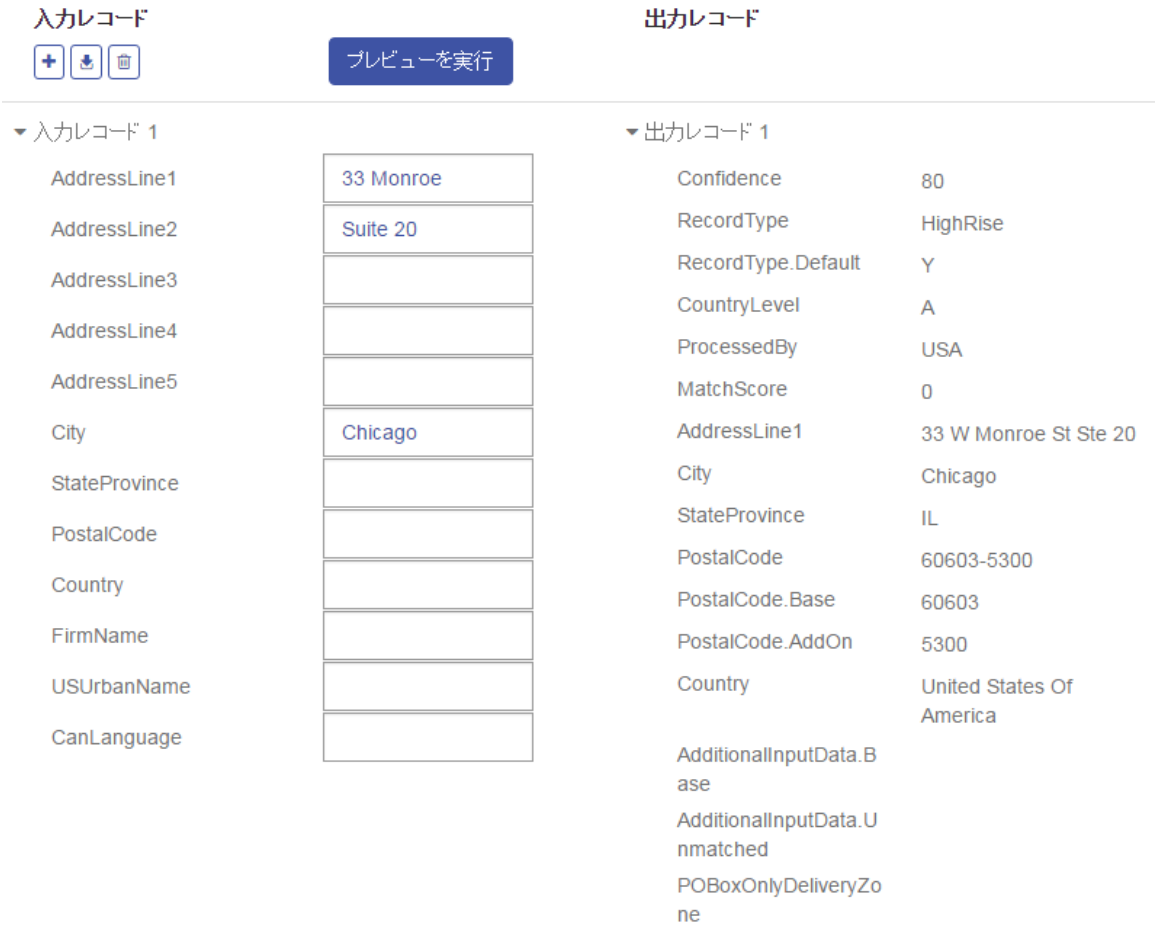
- すべてのレコードを削除するには、プレビュー エリアの上部にある [削除] ボタン

  をクリックします。
- 特定のレコードを削除するには、入力レコード名 ("Input Record 1" など) の上にカーソルを合わせてレコード名の横の [削除] ボタン

  をクリックします。
- サービスが階層化されたデータを入力として受け取る場合
  - 子レコードを追加するには、親レコードの上にカーソルを合わせて追加ボタンをクリックします。
  - 親からすべての子レコードを削除するには、親レコードの上にカーソルを合わせて削除ボタンをクリックします。
  - 特定の子レコードを削除するには、そのレコードの上にカーソルを合わせて削除ボタンをクリックします。

5. **[プレビューを実行]** をクリックします。

   サービスは入力レコードを処理し、結果を表示します。



6. 出力データをプレビューして、適切な結果が得られることを確認します。必要に応じて、サービスの設定に変更を加えて、**[プレビューを実行]** を再度クリックすることができます(データを再度入力する必要はありません)。

サービスの最適化

Spectrum™ Technology Platform サービスを呼び出す方法は多数あり、そのいくつかは他に比べてパフォーマンスに優れています。Spectrum™ Technology Platform サービスを呼び出すさまざまな方法を最も速いものから順に並べると、おおよそ次のようになります。

- ソケットを使用したクライアント API
- HTTP を使用したクライアント API
- HTTPS を使用したクライアント API
- HTTP を使用した XML
- Web サービス - HTTP を使用した SOAP および REST

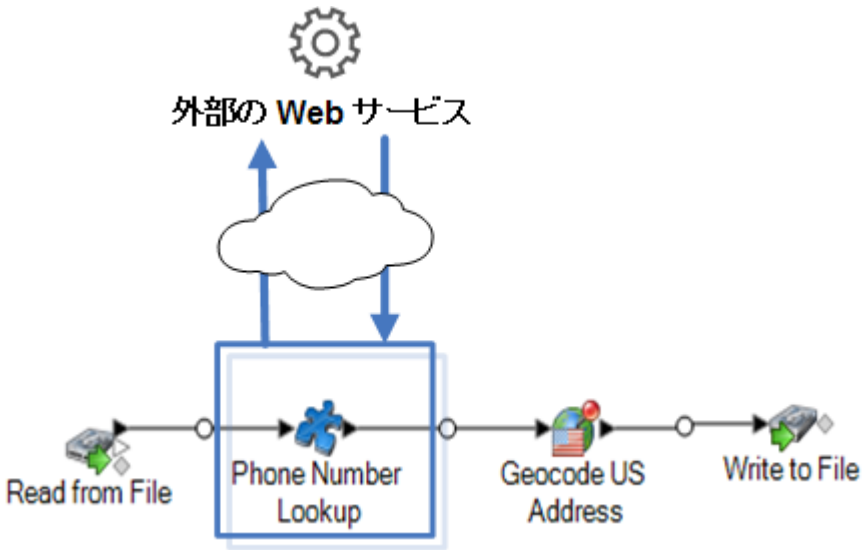
一般的に、クライアント API によるサービスの起動は、Web サービスの呼び出しより速くなります。ネットワーク プロトコルは、サービス呼び出しのラウンドトリップ タイムに大きな影響を与えることがあります。例えば、HTTP の代わりに持続的なソケット接続を使用すると、応答時間を 30% ～ 50% 向上させることができます。

リアルタイム アプリケーションによる Spectrum™ Technology Platform サービス呼び出しのパフォーマンスは、アプリケーションがシングルスレッドかマルチスレッドか、またサーバーがサービス要求を満たすだけのリソースを使用できるかどうかにも依存します。シングルスレッドのクライアント アプリケーションでは、ステージの実行時インスタンスを追加で指定しても、応答時間に及ぼす影響は最小限です。マルチスレッドのクライアント アプリケーションは、通常、実行時インスタンスを複数にする (最大で同時並行スレッド数) ことによって恩恵を受けます。

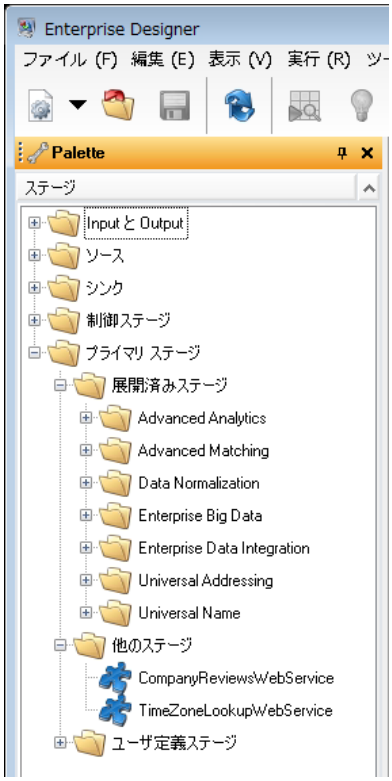
外部の Web サービス

外部の Web サービスとは、サードパーティがインターネット上で提供しているデータ処理サービスのことです。外部の Web サービスを Management Console で定義し、それをデータフローのステージとして使用することができます。したがって、インターネット上には多種多様な Web サービスが公開されているので、ほとんどすべての種類の処理を Spectrum™ Technology Platform 環境に組み込むことができます。

外部の Web サービスの概念を次の図に示します。ここでは、Phone Number Lookup という名前の外部の Web サービスがデータフローに追加されています。データフローの実行時、Spectrum™ Technology Platform は外部の Web サービスに各 レコードを送信します。外部の Web サービスはそのレコードを処理してステージに返します。電話番号を追加して更新されたレコードは、データフローの次のステージ (この例では Geocode US Addres) で引き続き使用されます。



外部の Web サービスは Enterprise Designer のパレットに表示され、他のステージを実行しながら、それらのサービスを操作できます。以下に外部の Web サービスの例として、CompanyReviewsWebService と TimeZoneLookupWebService の 2 つを示します。



要件と制約

Spectrum™ Technology Platform は、REST、SOAP 1.1、または SOAP 1.2 メッセージングを使用する外部の Web サービスをサポートしますが、次の制約があります。

- 複数の表現を持つ WADL 要求および応答はサポートされていません。
- 再帰的なスキーマはサポートされていません。

外部 Web サービスの追加

外部の Web サービスとは、サードパーティがインターネット上で提供しているデータ処理サービスのことです。外部 Web サービスをフローのステージとして使用し、Spectrum™ Technology Platform サーバーの機能を拡張することができます。

この手順では、Spectrum™ Technology Platform サーバーとサードパーティ Web サービスとの間の接続を定義します。手順が完了すると、Enterprise Designer に、外部 Web サービスを表す新

しいステージが追加されます。この外部Webサービスは、フローの他のステージと同様に使用できます。

1. Management Console を開きます。
2. **[リソース] > [外部の Web サービス]** に移動します。
3. [追加] ボタン をクリックします。
4. **[記述子]** ステップで、次の手順を実行します。
   a) 次のいずれかの方法で、Web サービスの WADL または WADL を指定します。

URL からの Web サービス記述子のロード	**[記述子をロード]** フィールドで、**[URL によって]** を選択し、**[URL]** フィールドで WADL または WSDL の URL を指定します。Web サービスの資格情報を求められた場合は、資格情報を入力します。
ファイルからの Web サービス記述子のロード	**[記述子をロード]** フィールドで、**[アップロード]** を選択し、**[ファイルのアップロード]** フィールドでファイルを選択します。Web サービス ベンダーによっては、WSDL または WADL の提供方法として、URL 経由ではなく、ファイルを指定している場合があります。
WADL を持たない REST Web サービスの場合	**[記述子をロード]** フィールドで、**[なし]** を選択します。

   b) **[次へ]** をクリックします。
5. **[設定]** ステップで、次の手順を実行します。
   a) **[名前]** フィールドに、外部の Web サービスが Spectrum™ Technology Platform で公開される時の名前を入力します。この名前は、Enterprise Designer に表示されるステージ名となります。どのような名前を付けてもかまいませんが、システム上の他の Web サービスで使用されている名前と重複してはなりません。
   b) **[タイムアウト]** フィールドに、Web サービスに送信したリクエストがタイムアウトするまでの経過時間を秒数で入力します。

      注： ここで指定したタイムアウト値は、Web サービスに対するすべてのリクエストに適用されます。これには、公開された Web サービスに対するトランザクションだけでなく、Webサービスの設定の過程で送信されるリクエストも含まれます。このような設定のリクエストは、**[リクエスト]** ページで新しい項目を選択した時や、プレビューを実行したときに送信されます。タイムアウトは、これらのどのアクションを実行する場合にも生じる可能性があります。タイムアウトが生じる場合、Web サービスが実際に稼働中であるならば、**[タイムアウト]** の値を大きくすることによってそれを回避できる可能性があります。

c) 外部のWebサービスでユーザ名とパスワードが必要とされる場合は、**[セキュリティ設定]**の**[タイプ]**フィールドで、ユーザ名とパスワードをSpectrum™ Technology Platformサーバーから外部のWebサービスに転送する方法を選択します。

なし	外部のWebサービスの使用にユーザ名とパスワードの入力が必要とされない場合は、このオプションを選択します。
ベーシック認証	ユーザ名およびパスワード情報をHTTPヘッダで外部のWebサービスに渡す場合は、このオプションを選択します。
WS-Security	(SOAPサービスのみ) ユーザ名およびパスワード情報をSOAPメッセージのヘッダーを通じて外部のWebサービスに渡す場合は、このオプションを選択します。

d) 外部Webサービスへのアクセスに必要ならば、ユーザ名とパスワードを入力します。

e) **[次へ]**をクリックします。

6. **[リクエスト]**ステップで、外部のWebサービスへのリクエストに含めるパラメータを設定します。

REST Webサービスの場合:

URL **[記述子]**ステップでWADLを指定しなかった場合は、パス パラメータ (あれば)と、要求に含めるクエリ パラメータを含む、サンプル要求URLを入力します。例:

http://example.com/rest/customers/{state}?age=31

ここには、1つのパス パラメータ `{state}` と1つのクエリ パラメータ `age` があります。

[記述子]ステップでWADLを指定した場合は、指定したWADLに基づくエンドポイントが**[URL]**フィールドに表示されます。このエンドポイントを編集することはできません。

Resource この設定は、**[記述子]**ステップでWADLを指定した場合のみ表示されます。Spectrum™ Technology Platform上に公開するWebサービスのリソースを選択します。

> **注:** 2つ以上のリソースを公開したい場合は、各リソースに対して個別の外部Webサービスを定義する必要があります。

方法 外部のWebサービスへの要求に使用するHTTPメソッドを選択します。

[記述子]ステップでWADLを指定した場合は、その外部WebサービスによってサポートされているHTTPメソッドのみがリストに表示されます。

パス パラメータ

[パス パラメータ] セクションには、その外部 Web サービスがパス パラメータを使用する場合は、URL パスに含まれるパラメータが一覧表示されます。例えば、この URL にはパス パラメータ `{state}` が含まれます。

http://example.com/rest/customers/{state}?age=31

[記述子] ステップで WADL を指定した場合は、Web サービスのパス パラメータがリストに表示されます。**[記述子]** ステップで WADL を指定しなかった場合は、**[URL]** フィールドに入力したサンプル要求 URL からパス パラメータのリストが生成されます。パス パラメータを追加または削除するには、URL からそれを追加または削除します。URL の中の波括弧で囲まれた部分はすべて、パス パラメータとして解釈されます。

クエリ パラメータ

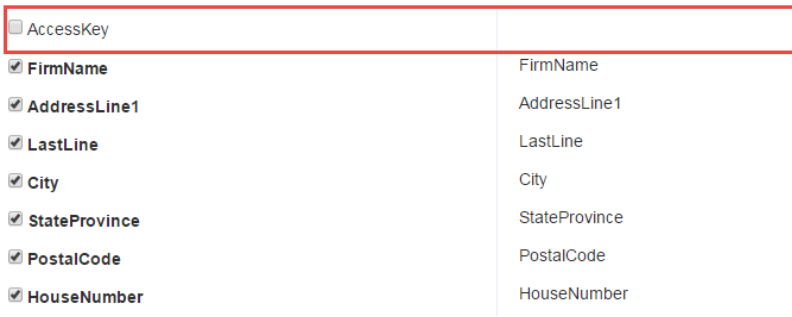
[クエリ パラメータ] セクションには、要求 URL 内で "?" の後に記述されるパラメータのリストが表示されます。例えば、この URL にはクエリ パラメータ `age` が含まれます。

http://example.com/rest/customers/{state}?age=31

[記述子] ステップで WADL を指定した場合は、Web サービスのクエリ パラメータがリストに表示されます。**[記述子]** ステップで WADL を指定しなかった場合は、**[URL]** フィールドに入力したサンプル要求 URL からクエリ パラメータのリストが生成されます。クエリ パラメータを追加または削除するには、URL からそれを追加または削除します。

表 2 : REST パス パラメータとクエリ パラメータの設定

設定	説明
エクスポーズ	Spectrum™ Technology Platform ステージでパラメータを使用可能にするには、この列のチェックボックスをオンにします。
要求	この列には、外部 Web サービスへの要求に使用されるパラメータ名がリスト表示されます。
入力	この列には、フローで表示される時の入力フィールドの名前がリストされます。サードパーティの Web サービスで使用されていたものと同じ名前を使用することも、変更することもできます。名前を変更するには、名前の上にカーソルを合わせて編集ボタン をクリックします。

設定	説明
デフォルト値	フィールドのデフォルト値を指定する場合は、このチェックボックスをオンにします。チェックボックスをオンにした後に表示されるフィールドにデフォルト値を入力します。 フローからオーバーライドできないデフォルト値を指定するには、**[デフォルト値]**列のチェックボックスをオンにして、**[要求]**列の対応するボックスをオフにします。これによって、外部の Web サービスへの要求でデフォルト値を指定する場合に、フローにおいてそのフィールドは非表示となります。要求ごとに指定する必要のあるアクセス キーがある場合は、これが便利です。例: 

SOAP Web サービスの場合:

URL このフィールドには、**[記述子]** ステップで指定した WADL に基づくエンドポイントが表示されます。このフィールドは編集できません。

操作 実行する Web サービスの操作を選択します。

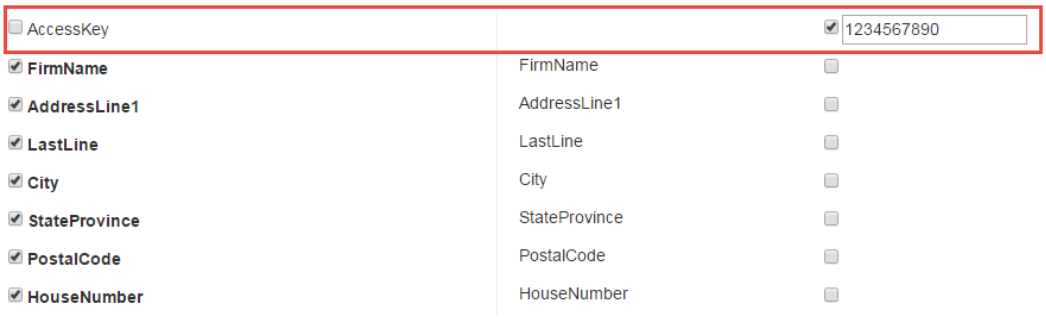
> **注：** 2 つ以上の操作を公開したい場合は、各操作に対して個別の外部 Web サービスを定義する必要があります。

要求 この列では、Spectrum™ Technology Platform を通じて利用可能にするフィールドとオプションを選択します。

入力 この列には、フローで表示される時の入力フィールドの名前がリストされます。サードパーティの Web サービスで使用されていたものと同じ名前を使用することも、変更することもできます。名前を変更するには、名前の上にカーソルを合わせて編集ボタン をクリックします。

デフォルト値 フィールドのデフォルト値を指定する場合は、このチェックボックスをオンにします。チェックボックスをオンにした後に表示されるフィールドにデフォルト値を入力します。

フローからオーバーライドできないデフォルト値を指定するには、**[デフォルト値]** 列のチェックボックスをオンにして、**[要求]** 列の対応するボックスをオフにします。これによって、外部の Web サービスへの要求でデフォルト値を指定する場合に、フローにおいてそのフィールドは非表示となります。要求ごとに指定する必要のあるアクセス キーがある場合は、これが便利です。例:



7. **[メソッド]** フィールドで POST または PUT を選択した場合は、POST または PUT 操作で Web サービスに送信するデータの構造を定義します。これを定義するには、**[フォーマット]** ボタンをクリックし、次のオプションのいずれかを選択します。

   スキーマのアップロード POST または PUT 操作で Web サービスに送信するデータの構造を定義する XML スキーマを用意している場合は、このオプションを選択します。このオプションを選択した後、スキーマ ファイルの場所を参照して指定します。

   サンプルの提供 POST または PUT 操作で Web サービスに送信するデータのサンプルを用意している場合は、このオプションを選択します。このオプションを選択した後、手動でサンプルを入力するか、ウィンドウにサンプルをペーストします。

   スキーマかサンプルを提供したら、フロー ステージで利用可能にする各データ要素の隣にあるボックスをチェックします。

8. **[次へ]** をクリックします。
9. **[ヘッダー]** ステップで、次の手順を実行します。
   a) **[HTTP ヘッダ]** で、各ヘッダについて、外部の Web サービスに渡す値を指定します。ここで指定した値は、Spectrum™ Technology Platform から外部の Web サービスに送信されるすべてのリクエストで使用されます。ヘッダが表示されていない場合、その外部の Web サービスでは HTTP ヘッダの使用が必須ではありません。

b) SOAP Web サービスで、その外部の Web サービスが SOAP ヘッダをサポートしている場合は、**[SOAP ヘッダ]** でヘッダを選択することが可能です。各 SOAP ヘッダにデフォルト値を指定できます。このデフォルト値は、Spectrum™ Technology Platform への各リクエストにおいて上書きすることができます。**[入力]** 列には、Spectrum™ Technology Platform へのリクエストで使用されるヘッダの名前が表示されます。名前を変更する場合は、名前の上にカーソルを合わせて、編集ボタン をクリックします。

チェック ボックスがオンで、グレー表示されている場合は、そのヘッダは必須で、無効にできないことを意味します。

c) **[次へ]** をクリックします。

10. **[応答]** ステップで、次の手順を実行します。

a) 外部の Web サービスから、単一のフィールドでレスポンスが返されるようにしたい場合は、**Return payload as field** のボックスをチェックします。これにより、すべてのレスポンス要素が個別フィールドではなく、単一のフィールドに配置されます。フィールド名は、REST Web サービスの場合は RestReponse で、SOAP Web サービスの場合は SoapResponse です。

b) REST Web サービスを設定している場合は、**[フォーマット]** ボタンが表示されます。このボタンをクリックすると、Spectrum™ Technology Platform によって返される Web サービス レスポンスの構造を定義する方法を選択できます。

スキーマのアップロード	Spectrum™ Technology Platform が返すレスポンスの構造を定義する XML スキーマを用意している場合は、このオプションを選択します。このオプションを選択した後、スキーマ ファイルの場所を参照して指定します。
サンプルの提供	Spectrum™ Technology Platform からのレスポンスを定義するための、外部の Web サービスからのレスポンスのサンプルを用意している場合は、このオプションを選択します。このオプションを選択した後、手動でサンプルを入力するか、ウィンドウにサンプルをペーストします。

c) **[レスポンス]** 列で、Spectrum™ Technology Platform で利用可能にするフィールドを選択します。

このアイコン は、フィールドがリスト フィールドとして返されることを意味します。リストは、反復可能な階層データを含む、Spectrum™ Technology Platform のデータ タイプです。例えば、[PhoneNumbers] フィールドには、複数の [Phone] フィールドが含まれる可能性があります。

```
<PhoneNumbers>
   <Phone>
 <Type>Cell</Type>
 <Number>312-123-4567</Number>
```

```
    </Phone>
    <Phone>
  <Type>Home</Type>
  <Number>773-123-4567</Number>
    </Phone>
</PhoneNumbers>
```

この場合、[PhoneNumbers] フィールドは、[Phone] 要素のリストを含むリスト フィールドとなります。

d) **[出力]** 列には、フローで表示される出力フィールドの名前がリストされます。サードパーティの Web サービスで使用されていたものと同じ名前を使用することも、変更することもできます。名前を変更するには、名前の上にカーソルを合わせて編集ボタンをクリックします。

e) **[次へ]** をクリックします。

11. **[プレビュー]** ステップでは、サンプル データを入力して **[プレビューを実行]** をクリックすることで、外部の Web サービスをテストできます。この手順は省略可能です。

サンプル データを入力する際は、次の点に注意してください。

- すべてのフィールドにデータを入力する必要はありません。フィールドを空のままにすると、空の文字列がプレビューに使用されます。
- フィールドにデフォルト値または null 値を引き渡した場合の影響をプレビューしたい場合は、フィールドの横にある無効アイコン

をクリックします。**[要求]** タブでフィールドのデフォルト値を定義した場合は、デフォルト値が使用されます。デフォルト値を定義しなかった場合は、null 値が使用されます。

- 複数のレコードを同時にプレビューできます。レコードを追加するには、追加ボタンをクリックします。
- テスト用のデータはファイルからインポートできます。データをインポートするには、インポート ボタンをクリックします。次のことに注意してください。
  - ファイルの最初の行は、ヘッダー レコードでなければなりません。ヘッダー内のフィールド名は、サービスで必要とされるフィールド名と一致する必要があります。
  - インポートできるレコードは、最大 5 個です。

- フィールドの区切り文字がスペースの場合、フィールド値を引用符で囲む必要があります。スペースをフィールドの区切り文字に使うファイルの例を以下に示します。

```
AddressLine1 AddressLine2 City StateProvince PostalCode
"One Global View" "" "Troy" "NY" "12180"
"3001 Summer St" "" "Stamford" "CT" "06926"
"224 N Michigan Ave" "Suite 300" "Chicago" "IL" ""
```

- すべてのレコードを削除するには、プレビュー エリアの上部にある [削除] ボタン

をクリックします。

- 特定のレコードを削除するには、入力レコード名 ("Input Record 1" など) の上にカーソルを合わせてレコード名の横の [削除] ボタン

をクリックします。

12. 外部の Web サービスをフローで利用可能にするには、**[有効]** スイッチを **[On]** に切り替えます。
13. **[保存]** をクリックします。

以上で外部の Web サービスが定義され、Enterprise Designer のフロー内のステージとして使用できるようになります。

外部 Web サービスのプレビュー

テスト要求を送信することによって、外部の Web サービスからの応答をプレビューすることができます。[Web サービスの追加] ウィザードの最後のステップで、新しい外部 Web サービスを追加するときにこの操作を実行できます。Spectrum™ Technology Platform サーバーに既に追加されている外部 Web サービスからの応答をプレビューすることもできます。以下では、既に追加されている外部の Web サービスをプレビューする方法について説明します。

> **注 :** 外部の Web サービスをプレビューできるようにするには、**[外部の Web サービス - 接続]** の表示および変更権限に加えて、**[プラットフォーム - サービス]** の表示および実行権限が必要です。

1. Management Console で、**[サービス] > [外部の Web サービス]** を選択します。
2. プレビューする外部 Web サービスをクリックします。
3. 外部 Web サービスへのテスト要求に使用するサンプル データを入力します。

   サンプル データを入力する際は、次の点に注意してください。

   - すべてのフィールドにデータを入力する必要はありません。フィールドを空のままにすると、空の文字列がプレビューに使用されます。
   - フィールドにデフォルト値または null 値を引き渡した場合の影響をプレビューしたい場合は、フィールドの横にある無効アイコン

     をクリックします。**[要求]** タブでフィールドのデフォルト値を定義した場合は、デフォルト値が使用されます。デフォルト値を定義しなかった場合は、null 値が使用されます。
   - 複数のレコードを同時にプレビューできます。レコードを追加するには、追加ボタン をクリックします。
   - テスト用のデータはファイルからインポートできます。データをインポートするには、インポート ボタン をクリックします。次のことに注意してください。
     - ファイルの最初の行は、ヘッダー レコードでなければなりません。ヘッダー内のフィールド名は、サービスで必要とされるフィールド名と一致する必要があります。

- インポートできるレコードは、最大 5 個です。
- フィールドの区切り文字がスペースの場合、フィールド値を引用符で囲む必要があります。スペースをフィールドの区切り文字に使うファイルの例を以下に示します。

```
AddressLine1 AddressLine2 City StateProvince PostalCode
"One Global View" "" "Troy" "NY" "12180"
"3001 Summer St" "" "Stamford" "CT" "06926"
"224 N Michigan Ave" "Suite 300" "Chicago" "IL" ""
```

- すべてのレコードを削除するには、プレビュー エリアの上部にある [削除] ボタン

をクリックします。

- 特定のレコードを削除するには、入力レコード名 ("Input Record 1" など) の上にカーソルを合わせてレコード名の横の [削除] ボタン

をクリックします。

4. **[プレビューを実行]** をクリックします。

外部 Web サービスからの結果が入力データの右側に表示されます。

外部の Web サービスの定義のエクスポート

外部の Web サービスの定義には、Spectrum™ Technology Platform からインターネットを経由して第三者の Web サービスにアクセスするための接続プロパティが格納されます。接続プロパティとは、外部の Web サービスの URLやユーザの資格情報などです。この情報はファイルに保存できるので、外部の Web サービスを別の Spectrum™ Technology Platform サーバーに簡単に追加できます。

1. Management Console を開きます。
2. **[リソース] > [外部の Web サービス]** に移動します。
3. エクスポートする外部の Web サービスの定義の横にあるボックスをチェックし、エクスポート ボタン をクリックします。
4. ファイルを保存する場所を選択します。

   外部の Web サービスの定義は、`.ews` ファイル拡張子を付けてファイルに保存されます。

外部の Web サービスの定義のインポート

外部の Web サービスの定義には、Spectrum™ Technology Platform からインターネットを経由して第三者の Web サービスにアクセスするための接続プロパティが格納されます。接続プロパティとは、外部の Web サービスの URLやユーザの資格情報などです。 外部の Web サービスの定義ファイルをインポートすると、Spectrum™ Technology Platform サーバー間で外部の Web サービスの定義を受け渡し、別のサーバー上にそのサービスを実装することができます。

1. Web サービスの定義をインポートするサーバーで Management Console にログインします。
2. **[リソース] > [外部の Web サービス]** に移動します。
3. インポート ボタン をクリックします。
4. インポートする外部の Web サービスの定義ファイルを選択します。外部の Web サービスの定義ファイルは、`.ews` ファイル拡張子が付いています。
5. **[OK]** をクリックします。

外部 Web サービスの削除

外部 Web サービスを削除すると、その外部 Web サービスを参照する既存のフローは正常に実行できなくなります。外部 Web サービスを参照するフローがなければ、サービスは差し支えなく削除できます。

1. Management Console で、**[リソース] > [外部 Web サービス]** を選択します。
2. 削除する Web サービスの横にあるボックスをチェックして、削除ボタン  をクリックします。
3. **[OK]** をクリックして確認します。

7 - フロー

このセクションの構成

フロー デフォルトの構成

データ タイプ変換のデフォルト値の設定

システムに対するデフォルトのデータ タイプ変換は、Management Console で設定できます。Enterprise Designer では、デフォルト フォーマットを個別のデータフローについてオーバーライドできます。

システムでデフォルトのデータ タイプ変換オプションを設定するには、次の手順に従います。

1. Management Console を開きます。
2. **[フロー] > [デフォルト]** を選択します。
3. **[データ タイプの変換]** をクリックします。
4. 文字列に変換する日付および時間データで使用する形式を指定します。日付または時間を文字列に変換する場合、文字列にはここで指定する形式が使用されます。
   a) **[ロケール]** フィールドで、文字列に変換する日付で使用する形式を持つ国を選択します。**[日付]**、**[時間]**、および **[日付/時刻]** フィールドのデフォルト値は、ここでの選択によって決まります。また、月のスペルを表記するときの言語も、ここでの選択によって決まります。例えば、英語を指定した場合、1 年の最初の月は "January" ですが、フランス語を指定した場合は "Janvier" になります。
   b) **[日付]** フィールドで、日付データを文字列に変換するときに使用する形式を指定します。選択したロケールで最も一般的に使用される形式の一覧が表示されます。

      例えば、形式として **[M/D/YY]** を選択し、日付フィールドに 2020-3-2 が含まれている場合、日付データは文字列`3/2/20`.

   c) **[時間]** フィールドで、時間データを文字列に変換するときに使用する形式を指定します。選択したロケールで最も一般的に使用される形式の一覧が表示されます。

      例えば、形式として **[h:mm a]** を選択し、時間フィールドに 23:00 が含まれている場合、時間データは文字列 `11:00 PM`.

   d) **[日付/時刻]** フィールドで、DateTime データ タイプを含むフィールドを文字列に変換するときに使用する形式を選択します。選択したロケールで最も一般的に使用される形式の一覧が表示されます。

      例えば、形式として **[M/d/yy h:mm a]** を選択し、DateTime フィールドに 2020-3-2 23:00 が含まれている場合、DateTime データは文字列 `3/2/20 11:00 PM`.

e) **[整数]** フィールドで、数値型 (Float および Double データ タイプ) で使用する書式設定を選択します。

例えば、形式 **[#,###]** を選択すると、数字 4324 は`4,324`.

> **注 :** このフィールドを空白にしておくと、数字には Spectrum™ Technology Platform 8.0 以降と同じ形式が使用されます。特に、桁区切り文字は使用されません。小数点記号としてドット (".") が使用されます。10^{-3} 未満または 10^{7} 以上の数字は指数表示されます。負の数には先頭にマイナス記号 ("-") が付きます。また、このフィールドを空白にしておくと、BigDecimal データ タイプを使用する数字には常に形式 #,###.000 が使用されます。

f) **[小数]** フィールドで、Decimal (Integer および Long データ タイプ) を含む数字で使用する書式設定を選択します。

例えば、形式 **[#,##0.0#]** を選択すると、数字 4324.25 は`4,324.25`.

> **注 :** このフィールドを空白にしておくと、数字には Spectrum™ Technology Platform 8.0 以降と同じ形式が使用されます。特に、桁区切り文字は使用されません。小数点記号としてドット (".") が使用されます。10^{-3} 未満または 10^{7} 以上の数字は指数表示されます。負の数には先頭にマイナス記号 ("-") が付きます。また、このフィールドを空白にしておくと、BigDecimal データ タイプを使用する数字には常に形式 #,###.000 が使用されます。

また、ニーズを満たす形式がない場合は、独自の日付、時間、および数値形式を指定することもできます。独自の日付または時間形式を指定するには、**日付および時間パターン**（185ページ） に記載される表記を使用して、その形式をフィールドに入力します。独自の数値形式を指定するには、**数字パターン**（188ページ） に記載される表記を使用して、形式をファイルに入力します。

5. **[Null の処理]** で、フィールドに NULL 値が含まれている場合にタイプ変換を実行するかどうかを選択します。次のいずれかのオプションを選択すると、NULL 値を含むデータフローまたはレコードが、**[エラー ハンドリング]** フィールドの選択に基づきエラーを返します。

文字列型の Null のエラー	NULL 値を含む文字列フィールドでタイプ変換が必要な場合に、フローまたはレコードでエラーを返します。
Boolean 型の Null のエラー	NULL 値を含む Boolean フィールドでタイプ変換が必要な場合に、フローまたはレコードでエラーを返します。
数値型の Null のエラー	NULL 値を含む数値フィールドでタイプ変換が必要な場合に、フローまたはレコードでエラーを返します。数値フィールドには、double、float、long、integer、および Big Decimal フィールドが含まれます。

日付型の Null のエラー NULL 値を含む日付フィールドでタイプ変換が必要な場合に、フローまたはレコードでエラーを返します。これには、date、time、および DateTime フィールドが含まれます。

6. **[エラー ハンドリング]** フィールドで、フィールドの値をステージで必要とされるデータ タイプに自動変換できない場合の対処方法を指定します。

   フローのエラー フィールドを変換できない場合、フローは失敗します。

   レコードのエラー フィールドを変換できない場合、レコードは失敗しますが、フローの実行は続行されます。

   フィールドをデフォルト値で初期化する フィールドを変換できない場合、フィールドの値はここで指定する値に置き換えられます。一部のレコードに不正なデータが含まれることがわかっていて、不正なデータをデフォルト値に置き換える場合、このオプションが役に立ちます。各データ タイプの値を指定します。

日付および時間パターン

日付および時間データのデータ タイプ オプションを定義する場合、あらかじめ定義されているオプションでニーズを満たすことができないときは、独自の日付または時間パターンを作成できます。日付または時間パターンを作成するには、次の表に記載される表記法を使用します。例えば、次のパターンを使用します。

dd MMMM yyyy

これは、次のような日付を生成します。

14 December 2020

文字	説明	例
G	年代指示子	AD
yy	年数の 2 桁表記	96
yyyy	年数の 4 桁表記	1996
M	月の数字表記	7
MM	月の数字表記: 数字が 10 未満の場合は、2 桁の数字にするためにゼロが付加されます。	07

文字	説明	例
MMM	月の名前の短縮表記	Jul
MMMM	月の名前の完全表記	7月
w	年の週数	27
ww	年の週数の 2 桁表記: 週が 10 未満の場合はゼロが付加されます。	06
W	月の週数	2
D	年の日数	189
DDD	年の日数の 3 桁表記: 数字が 3 桁未満の場合はゼロが付加されます。	006
d	月の日数	10
dd	月の日数の 2 桁表記: 数字が 10 未満の場合はゼロが付加されます。	09
F	週の日数	2
E	曜日の短縮表記	Tue
EEEE	曜日の完全表記	火曜日
a	AM PM マーカー	PM
H	1 日の時間。最初の時間は 0 で表記し、最後の時間は 23 で表記	0
HH	1 日の時間の 2 桁表記。最初の時間は 0 で表記し、最後の時間は 23 で表記します。: 数字が 10 未満の場合はゼロが付加されます。	08
k	1 日の時間。最初の時間は 1 で表記し、最後の時間は 24 で表記	24

文字	説明	例
kk	1 日の時間の 2 桁表記。最初の時間は 1 で表記し、最後の時間は 24 で表記します。: 数字が 10 未満の場合はゼロが付加されます。	02
k	午前中 (AM) または午後 (PM) の時間。最初の時間は 0 で表記し、最後の時間は 11 で表記	0
KK	1 日の時間の 2 桁表記。最初の時間は 1 で表記し、最後の時間は 24 で表記します。: 数字が 10 未満の場合はゼロが付加されます。	02
h	午前中 (AM) または午後 (PM) の時間。最初の時間は 1 で表記し、最後の時間は 12 で表記します。	12
hh	午前中 (AM) または午後 (PM) の時間の 2 桁表記。最初の時間は 1 で表記し、最後の時間は 12 で表記します。: 数字が 10 未満の場合はゼロが付加されます。	09
m	1 時間の分数	30
mm	1 時間の分数の 2 桁表記: 数字が 10 未満の場合はゼロが付加されます。	05
s	1 分の秒数	55
ss	1 分の秒数の 2 桁表記: 数字が 10 未満の場合はゼロが付加されます。	02
S	1 秒のミリ秒数	978
SSS	1 秒のミリ秒数の 3 桁表記。数字が 3 桁未満の場合は、3 桁にするために 1 つまたは 2 つのゼロが付加されます。	978 078 008
z	時間帯の名前の省略形。時間帯に名前がない場合は GMT オフセットを使用。	PST GMT-08:00

文字	説明	例
zzzz	時間帯の名前の完全表記。時間帯に名前がない場合はGMTオフセットを使用	太平洋標準時 GMT-08:00
Z	RFC 822 時間帯	-0800
X	ISO 8601 時間帯	-08Z
XX	分を付加した ISO 8601 時間帯	-0800Z
XXX	分、およびコロン区切り文字を時間と分の間に付加した ISO 8601 時間帯	-08:00Z

数字パターン

数値データのデータ タイプ オプションを定義する場合、あらかじめ定義されているオプションでニーズを満たすことができないときは、独自の数字パターンを作成できます。基本的な数字パターンは、次の要素で構成されます。

- 通貨記号などの接頭辞 (オプション)
- オプションのグループ文字を含む数字のパターン (例えば、桁区切り文字として使用するカンマ)
- 接尾辞 (オプション)

例えば、次のパターンを使用します。

$ ###,###.00

次のような形式の数字が生成されます (最初の 3 桁の数字の後に桁区切り文字を使用していることに注意)。

$232,998.60

負の数のパターン

デフォルトでは、負の数は、正の数と同じ形式で表記されますが、数字の先頭に負記号が追加されます。数字記号に使用される文字はロケールに基づきます。マイナス記号 "-" は、ほとんどのロケールで使用されます。例えば、次の数字パターンを指定するとします。

```
0.00
```

マイナス 10 は、ほとんどのロケールで次の形式で表記されます。

```
-10.00
```

ただし、負の数に使用する別の接頭辞または接尾辞を定義する場合は、2 つ目のパターンを指定し、そのパターンと 1 つ目のパターンをセミコロン (";") で区切ります。例:

```
0.00;(0.00)
```

このパターンでは、負の数は、次のように括弧で囲んで表記されます。

```
(10.00)
```

指数表記

数字を指数表示する場合は、文字 `E` の後に続けて、指数に含める最小桁数を指定します。例えば、次のパターンを使用するとします。

```
0.###E0
```

数字 `1234` は、次のような形式で表記されます。

```
1.234E3
```

つまり、これは 1.234 x 10^3 を表します。

注:

- 指数文字の後に続く桁数は、指数の最小桁数を表します。最大桁数はありません。
- 負の指数は、このパターンの接頭辞や接尾辞ではなく、ローカライズされた負の記号を使用した形式で表記されます。
- 指数表記パターンにグループ区切り文字 (桁区切り文字など) を含めることはできません。

特殊な数字パターン文字

次の文字は、その文字自体では別の文字を表しますが、実際には結果の数字で再現されます。次の特殊文字を数字パターンの接頭辞または接尾辞のリテラル文字として使用する場合は、特殊文字を引用符で囲みます。

記号	説明

`0`	必要な位置にゼロを含むパターン内の桁を表します。例えば、数字 27 に次のパターンを適用するとします。
	`0000`
	次のように表されます。
	`0027`

記号	説明

#	桁を表しますが、ゼロは省略されます。例えば、数字 27 に次のパターンを適用するとします。
	`####`
	次のように表されます。
	`27`

.	選択したロケールで使用される小数点記号または通貨区切り文字。例えば、米国では小数点記号にはドット (.) が使用されますが、フランスの場合、小数点記号にはカンマ (,) が使用されます。
-	選択したロケールで使用される負の記号。ほとんどのロケールでは、マイナス記号 (-) が使用されます。

,	選択されたロケールで使用されるグループ文字。選択されたロケールの適切な文字が使用されます。例えば、米国の場合、区切り文字にはカンマ (,) が使用されます。
	一般に、グループ区切り文字は千の位を区切るのに使用されますが、一部の国では一万の位を区切るのに使用されます。グループのサイズは、グループ区切り文字間の桁数を指定する定数です。例えば、3 は 100,000,000、4 は 1,0000,0000 になります。パターンに複数のグループ区切り文字を指定すると、最後のグループ区切り文字と数値の末尾との間隔が、使用される間隔です。例えば、次のパターンはすべて同じ結果を生成します。
	`#,##,###,####`
	`######,####`
	`##,####,####`

E	指数表記の仮数と指数を区切ります。パターン内で E を引用符で囲む必要はありません。**指数表記**（189ページ） を参照してください。
;	正のサブパターンと負のサブパターンを区切ります。**負の数のパターン**（188ページ） を参照してください。

記号	説明

%	数字を 100 で乗算し、その数字をパーセンテージで表示します。例えば、数字 .35 に次のパターンを適用するとします。
	`##%`
	次のような結果が生成されます。
	`35%`

¤	選択したロケールの通貨記号。double の場合、国際通貨記号が使用されます。パターンで表す場合、小数点記号ではなく、通貨区切り文字が使用されます。

'	接頭辞または接尾辞の特殊文字を引用符で囲むのに使用されます。例を次に示します。
	`"'#'#"`
	123 は次のように表記されます。
	`"#123"`
	単一引用符自体を作成するには、2 つの単一引用符を続けて使用します。
	`"# o''clock"`

形式に誤りのあるレコード数のデフォルト値の設定

形式に誤りのあるレコードとは、Spectrum™ Technology Platform が解析できない入力レコードです。デフォルトでは、形式に誤りのあるレコードが 1 つでもジョブの入力データに含まれていると、ジョブは停止します。この設定を変更すると、形式に誤りのあるレコードを複数許容したり、そうしたレコードを無制限に許容したりできます。この手順では、システム上のジョブに対して形式に誤りのあるレコード数の最大値のデフォルトを設定する方法について説明します。

> **注：** Enterprise Designer でジョブを開き、**[編集] > [ジョブ オプション]** に移動して、形式に誤りのあるレコード数の最大数のデフォルトをオーバーライドできます。

1. Management Console を開きます。
2. **[フロー] > [デフォルト]** を選択します。
3. **[形式に誤りのあるレコード]** をクリックします。
4. 次のいずれかを選択します。

形式に誤りのあるレコードがこれだけ多く含まれているジョブを停止	形式に誤りのあるレコードが入力データに 1 つ以上含まれている場合は、このオプションを選択します。ジョブ停止のトリガとして設定する、形式に誤りのあるレコードの数を入力します。デフォルトは 1 です。
形式に誤りのあるレコードによってフローを終了しない	入力データに含まれる、形式に誤りのあるレコードの数を無制限に許容する場合は、このオプションを選択します。

レポートに関するデフォルト値の設定

レポートは、レポート ステージが含まれているジョブによって生成されます。レポートには、処理の概要 (ジョブによって処理されたレコード数など) や郵便フォーム (USPS CASS 3553 フォームなど) を含めることができます。一部のモジュールには、定義済みレポートが付属しています。カスタム レポートを作成することもできます。レポートに関するデフォルト値を設定すると、レポートを保温するためのデフォルト設定が定められます。こうしたデフォルト設定は、Enterprise Designer を使用すると、特定のジョブやあるジョブ内の特定のレポートについてオーバーライドできます。

ここでは、システムに対するデフォルト レポート オプションを設定する方法を示します。

1. Management Console を開きます。
2. **[フロー] > [デフォルト]** を選択します。
3. **[レポート]** をクリックします。
4. レポートの保存に使用する形式を選択します。レポートは HTML、PDF、またはテキストとして保存できます。
5. レポートの保存場所を選択します。

レポートをジョブ履歴に保存	レポートをジョブ履歴の一部としてサーバーに保存します。こうすると、Management Console および Enterprise Designer ユーザはレポートを実行履歴で参照できるようになって便利です。
レポートをファイルに保存	指定した場所にあるファイルにレポートを保存します。これは、Spectrum™ Technology Platform ユーザではない人とレポートを共有したい場合に便利です。また、レポートのアーカイブを別の場所に作成したい場合にも便利です。この方法で保存されたレポートを表示する場合は、そのレポートの形式を開くことができる任意のツール (PDF レポートの場合は PDF ビューア、HTML レポートの場合は Web ブラウザ) を使用できます。

6. **[Save reports to a file (レポートをファイルに保存)]** を選択したは、次のフィールドに入力します。

レポートの場所 レポートの保存先フォルダです。

レポート名に追加 ファイル名に含める変数の情報を指定します。次のオプションから1つ以上を選択できます。

- **ジョブ ID** ジョブの実行に対して割り当てられる一意の ID です。システムで初めて実行したジョブの ID は 1 になります。2 回目にジョブを実行したときには、それが前回と同じジョブでも異なるジョブでも、ジョブ ID は 2 になります (3 回目以降も同様)。
- **ステージ** レポートにデータを提供したステージの名前であり、Enterprise Designer 内のレポート ステージで指定されているものです。
- **Date** レポートが作成された年月日です

既存のレポートを上書きする 以前のレポートを、同じファイル名を持つ新しいレポートで置き換えます。新しいレポートと同じ名前の既存レポートがあるのにこのオプションを選択していない場合、ジョブは正常に完了しますが、新しいレポートは保存されません。その場合は、レポートが保存されなかったことを示すコメントが実行履歴に表示されます。

フローのスケジュール

フローのスケジュール

フローのスケジュールを行うと、ジョブまたはデータフローを指定した時間に自動的に実行することができます。

> **注:** 日付と時間の繰り返しスケジューリング: すべてのフローを各月の初日に開始し、設定した繰り返しスケジュールに従って繰り返します。繰り返しスケジューリングでは、フローが実行される開始時刻と終了時刻、および間隔が定義されます。例えば、フローを 6 日間隔で午前 2 時に実行するようにスケジュールした場合、初日、6 日、12 日、24 日など、月末まで同じ時刻に実行されます。

- 日付と時間を指定して実行
- 日付と時間を指定して周期的に実行

注： スケジュールの作成、編集、表示を行うには、スケジュールするセキュア エンティティ タイプである**データフロー**または**プロセス フロー**への表示権限が必要です。

1. フローをエクスポーズしていない場合は、エクスポーズします。

   フローをエクスポーズするには、Enterprise Designer でフローを開き、**[ファイル] > [エクスポーズ/アンエクスポーズして保存]** を選択します。

2. Management Console を開きます。
3. **[フロー] > [スケジュール]** に移動します。
4. [追加] ボタン をクリックします。
5. **[名前]** フィールドに、このスケジュールに付ける名前を入力します。これがスケジュールの一覧に表示される名前になります。
6. **[フロー]** フィールドに、実行するジョブまたはプロセス フローを入力します。保存およびエクスポーズされたジョブとプロセス フローだけが、ここで使用可能です。
7. フローを指定した後で、追加のフィールドが**[フロー]**フィールドの下に表示されます。フローの各ソース ステージ (Read from File など) と各シンク ステージ (Write to File など) に対応するフィールドが提供されます。これらのフィールドは、このスケジュールに従ってフローが実行されるときに使用するファイルを示します。デフォルトで、フローのソースとシンクで指定されたファイルが使用されます。スケジュールで使われるファイルを変えるには、ファイル パスを別のファイルへのパスに置き換えます。例えば、フローの Read from File ステージでデータが `C:\FlowInput\Customers.csv` から読み込まれるが、このスケジュールの実行時に `C:\FlowInput\UpdatedCustomers.csv` のデータを使いたい場合は、`C:\FlowInput\UpdatedCustomers.csv` を [Read from File] フィールドに指定します。

   注： ソース ステージやシンク ステージで使われるファイルを変更するには、**リソース - ファイル サーバー** セキュア エンティティ タイプの読み取り権限が必要です。

フローがスケジュールによって実行される場合、フローで使われるファイルは Management Console で外部リソースとして定義された Spectrum™ Technology Platform サーバーまたはファイル サーバーに存在する必要があります。これはジョブと、プロセス フロー内のジョブアクティビティの両方に適用されます。ソース ステージまたはシンク ステージがクライアント コンピュータ上のファイルを参照している場合は、次の手順のいずれかを実行します。

オプション	説明
オプション 1: データフローを変更する	ファイルを Spectrum™ Technology Platform サーバーまたはファイル サーバーに移動し、データフローを変更します。 1. Enterprise Designer でデータフローを開きます。 2. ソースまたはシンク ステージをダブルクリックします。 3. **[ファイル名]** フィールドで、参照ボタンをクリックします。

オプション	説明
	4. **[リモート マシン]** をクリックして、必要なファイルを選択します。 **注：** Spectrum™ Technology Platform サーバーと同じコンピュータ上で Enterprise Designer を実行している場合、[リモート マシン] のクリックは、[マイ コンピュータ]をクリックするのと同じであるように見えます。しかし、ファイルが Spectrum™ Technology Platform サーバー上に存在することをシステムに認識させるためには、[リモート マシン] を使用してファイルを選択する必要があります。
オプション 2: このスケジュールの実行時にデータフローのファイルの場所をオーバーライドする	このスケジュールを実行するときに、フロー内のファイル参照をオーバーライドします。これを行うには、ソース フィールドとシンク フィールドに表示されているデフォルト ファイルを Management Console で定義された Spectrum™ Technology Platform サーバー上のファイルまたはファイル サーバー リソースへのパスで置き換えます。

8. **[トリガー]** フィールドで、次のいずれかを選択します。

Date/Time	日付と時刻を指定してフローを実行します。
Recurring Date/Time	繰り返しのパターンを使用して、複数の日付と時刻にフローを実行します。
Control File	指定のディレクトリにファイルが存在する場合にフローを実行します。コントロール ファイルの使用方法については「コントロール ファイルによるフローのトリガー（201ページ）」を参照してください。

9. フローを実行する日付と時刻または繰り返す間隔を指定します。

   注： **[トリガー]** フィールドで **[繰り返し]** を選択した場合は、必ず、繰り返しのパターンに一致する開始日を選択するようにしてください。例えば、月の最初の月曜日にフローを実行することを選択した場合は、月の最初の月曜日にあたる日付を選択する必要があります。繰り返しのパターンと一致しない日付を選択すると、フローが予期しない時間に実行される可能性があります。また、開始日に過去の日付を選択した場合も、フローが予期しない時間に実行される可能性があります。

10. フローが電子メール通知用に設定されている場合は、スケジュールの実行時に通知が送信される受信者を追加で指定できます。ここで指定した受信者は、フローの通知設定で指定された受信者に追加される形で通知を受け取ります。通知を送信するフローを設定するには、Enterprise Designer でフローを開き、**[編集] > [通知]** を選択します。
11. **[保存]** をクリックします。

スケジュールの表示

フロー スケジュールは、ジョブまたはプロセス フローをいつ実行するかを定義します。システム上のフロー スケジュールとフロー実行結果を表示できます。

1. Management Console を開きます。
2. **[フロー] > [スケジュール]** に移動します。

フロー スケジュールのリストが表示されます。スケジュールのリストを並べ替えるには、列の見出しをクリックします。リストは、**[フィルタ]** フィールドにキーワードを入力することでフィルタリングできます。このフィルタ フィールドによるフィルタリングは、**[スケジュール名]**、**[ユーザ]**、**[次回実行予定日時]**、および**[前回実行日時]** 列に対してのみ実行できることに注意してください。

スケジュールの削除

スケジュールが設定されたフローは、設定された時間または繰り返しのスケジュールに従って自動的に実行されます。フローをスケジュールに従って実行する必要がなくなった場合は、スケジュールを削除できます。スケジュールを削除しても、フローは削除されません。

スケジュールを削除するには

1. Management Console を開きます。
2. **[フロー] > [スケジュール]** に移動します。
3. 削除するスケジュールの横にあるボックスをチェックし、削除ボタン  をクリックします。

フロー ステータスと履歴の表示

ジョブ、プロセス フロー、およびサービス実行の履歴は、Management Console と Enterprise Designerで表示できます。

Management Console の場合

Management Console でフロー ステータスや履歴を表示するには、**[フロー] > [履歴]** を選択します。**[フロー]** タブにはジョブとプロセス フローの履歴が表示され、**[トランザクション]** タブにはサービス履歴が表示されます。

> **注：** フローの履歴の場合、**[結果]** 列の上にカーソルを置くと表示されるレコード数は、フローのすべてのシンクによって出力として書き出されるレコードの総数です。フローにおいてレコードが結合または分割されたり、新しいレコードが作成されたりする場合は、この数は入力レコード数とは異なる可能性があります。

デフォルトで、トランザクションの履歴は無効になっています。トランザクションの履歴を有効にすると、パフォーマンスに悪影響が出るためです。トランザクションの履歴を表示する場合は、**[トランザクションのログ記録]** スイッチをクリックしてトランザクション履歴のログへの記録を有効にする必要があります。ユーザの操作履歴を表示するには、**[システム] > [ログ]** でアクセスできる監査ログを使うことをお勧めします。

フローの履歴リストは、30 秒ごとに自動的に更新されます。それよりも先に更新するには、更新ボタン をクリックします。

Enterprise Designer の場合

Enterprise Designerでフロー ステータスや履歴を表示するには、**[表示] > [実行履歴]** を選択します。

フローの履歴リストは、30 秒ごとに自動的に更新されます。実行履歴の表示が遅いと感じる場合は、**[自動更新]** チェックボックスをオフにします。

[ジョブ] タブでは、ジョブ ステータスの監視、実行中のジョブの一時停止、再開、またはキャンセル、完了したジョブの削除を行います。

> **注：** **[ジョブ]** タブに表示されるレコード数は、フローのすべてのシンクによって出力として書き出されるレコードの総数です。フローにおいてレコードが結合または分割されたり、新しいレコードが作成されたりする場合は、この数は入力レコード数とは異なる可能性があります。

- **[Succeeded]** 列には、フローのすべてのシンクによって出力として書き出されるレコードで、[ステータス] フィールドの値が空であるものの総数が表示されます。
- **[Failed]** 列には、フローのシンクによって出力として書き出されるレコードで、[ステータス] フィールドの値が F であるものの総数が表示されます。
- **[Malformed]** 列には、ソース ステージのすべてのエラー ポートからのレコードの総数が表示されます。

[プロセス フロー] タブでは、プロセス フロー ステータスの監視、実行中のプロセス フローのキャンセル、完了したプロセス フローの削除を行います。各プロセス フローの横にある [+] 記号をク

リックすると、そのプロセス フローのアクティビティ ステータス情報が表示されます。このエリアには、次の情報が表示されます。

アクティビティ名 成功アクティビティを含む、プロセス フローを構成するすべてのアクティビティの名前が含まれます

州 アクティビティのステータスです (失敗、成功、実行中、キャンセル)

リターン コード プロセス フローの結果を示すコード

- **1** プロセス フローは失敗しました。
- **0** プロセス フローは正常終了しました。
- **-1** プロセス フローはキャンセルされました。
- **その他の数値** プロセス フローに「プログラムの実行」アクティビティが含まれる場合は、外部プログラムが独自のコードを返す可能性があります。ReturnCode 列の値が 1、0、-1 以外である場合、それは外部プログラムによるものです。外部プログラムのリターンコードの説明については、そのプログラムのドキュメントを参照してください。

開始 アクティビティが開始した日時です

終了 アクティビティが終了した日時です

コメント アクティビティに関するコメントです

フロー履歴のダウンロード

Management Console の [履歴] 画面に表示される情報を Microsoft Excelファイルにダウンロードできます。

1. Management Console を開きます。
2. **[フロー] > [履歴]** に移動します。
3. サービスの履歴情報をダウンロードするには、**[トランザクション履歴]** をクリックします。ジョブとプロセス フローの履歴をダウンロードするには、**[フロー]** タブを選択します。
4. ダウンロード ボタン をクリックします。

   ヒント： 履歴リストから特定の項目だけをダウンロードするには、フィルタ設定を変更してダウンロードしたい履歴だけを表示します。

実行履歴の完全削除

たくさんのフローや、頻繁に使用されるサービスがある場合、Management Console の実行履歴がかなりの大きさになることがあります。ここでは、古いレコードを実行履歴から削除する手順について説明します。構成データベースのサイズを縮小するために、古いレコードを完全に削除することができます。新しいバージョンにアップグレードする前にレコードを完全に削除すると、Spectrum™ Technology Platform のアップグレードにかかる時間を短縮できます。

完全削除には次の 2 つの方法があります。

- レコードの完全削除: com.pb.spectrum.platform.configuration:manager=ArchiveTransactionManager
- レコードの完全削除とアーカイブ ステータスの指定: com.pb.spectrum.platform.transaction:manager=archiveTransactionManager

以下の手順は、"レコードの完全削除" (アーカイブ ステータスなし) リクエストを示しています。

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. **Domain: com.pb.spectrum.platform.configuration** の下で、**com.pb.spectrum.platform.configuration:manager=ArchiveConfigurationManager** をクリックします。
3. (オプション) 完全削除する履歴のアーカイブを保存する場合は、アーカイブを保存するパスを **ArchiveDirectory** フィールドに指定し、**set** をクリックします。続いて、**ArchiveEnabled** を **true** に設定して、**set** をクリックします。
4. **ArchiveRetain** フィールドに、何日分のレコードを履歴に残すかを指定し、**set** をクリックします。例えば、45 と入力すると、45 日間の履歴レコードが保持され、46 日以上前のレコードは完全に削除されます。何日分のレコードが残せるかを判断するには、Enterprise Designer でジョブとプロセス フローの履歴を表示して、レコード数が 10 万件を超える時点を確認します。
5. (オプション) 完全削除を定期的に実行する場合は、Cron 式を使って **CronExpression** フィールドにスケジュールを入力します。

   cron 式は、スペースで区切られた 6 つの値で構成されます。オプションとして、第 7 の値を指定できます。

   秒

分
時間
月の日数
月
曜日
年 (オプション)

例えば、次の式は毎週日曜日の午前 0 時にフローとトランザクションの履歴を完全に削除します。

```
0 0 0 ? * SUN
```

cron 式の詳細については、**quartz-scheduler.org/documentation** を参照してください。

cron 式を指定した後、**CronExpression** フィールドの横の **set** ボタンをクリックし、**PurgeEnabled** を **true** に設定し、**PurgeEnabled** フィールドの横の **set** ボタンをクリックします。

> **注：** アップグレード処理にかかる時間を短縮するために履歴を一度だけ完全に削除する場合は、完全削除をスケジュールする必要はありません。

6. (オプション) 完全削除後の履歴に残すレコードの最大数を設定したければ、**MaxHistoryRecordCount** フィールドで最大レコード数を指定します。これが役に立つのは、毎日大量の履歴レコードが発生し、以前のレコードを **ArchiveRetain** フィールドの値に基づいて完全削除しても思ったほど実行履歴のサイズが減らないときです。以前のレコードを **ArchiveRetain** フィールドの値に基づいて完全削除した後、レコード数が **MaxHistoryRecordCount** フィールドの値と等しくなるまで残りのレコードが追加的に削除されます。履歴レコードの最大数を指定したくなければ、`-1` を指定します。

   > **注：** **MaxHistoryRecordCount** で指定する上限はプロセス フローおよびジョブの上限としてそれぞれ別々に設定されます。例えば、5000 を指定した場合、プロセス フローの履歴レコードの最大数が 5,000、ジョブの履歴レコードの最大数が 5,000 と設定され、合計の最大レコード数は 10,000 になります。

7. **PurgeOperation** フィールドの値は、`ALL` のままにします。
8. **[すべての MBean]** を選択して、メインの JMX コンソール画面に戻ります。
9. **Domain: com.pb.spectrum.platform.configuration** の下で、**com.pb.spectrum.platform.transaction:manager=ArchiveTransactionManager** を選択します。
10. 完全削除を実行するには、**Invoke** をクリックします。

これでフローと実行履歴が完全に削除され、構成データベースのサイズが小さくなります。

コントロール ファイルによるフローのトリガー

モニタリング対象のディレクトリ内でコントロール ファイルが検出されたときに、フローを自動的に実行することができます。この機能は、フローの実行の前に別のプロセスの完了が必要という状況において便利です。例えば、別のビジネス プロセスによって生成される入力ファイルを必要とするフローが考えられます。コントロール ファイルをフォルダに配置するように別のプロセスを設定し、そのコントロール ファイルを検出したらフローを実行するように Spectrum™ Technology Platform を設定することができます。

> **注:** 必ず、フローで必要なすべてのファイルの準備が整い、処理できる状態になった後に、コントロール ファイルをモニタリング対象のディレクトリに配置するようにしてください。

1. フローをエクスポーズしていない場合は、エクスポーズします。

   フローをエクスポーズするには、Enterprise Designer でフローを開き、**[ファイル] > [エクスポーズ/アンエクスポーズして保存]** を選択します。

2. Management Console を開きます。
3. **[フロー] > [スケジュール]** に移動します。
4. [追加] ボタン  をクリックします。
5. **[名前]** フィールドに、このスケジュールに付ける名前を入力します。これがスケジュールの一覧に表示される名前になります。
6. **[フロー]** フィールドに、実行するジョブまたはプロセス フローを入力します。保存およびエクスポーズされたジョブとプロセス フローだけが、ここで使用可能です。
7. フローを指定した後で、追加のフィールドが**[フロー]**フィールドの下に表示されます。フローの各ソース ステージ (Read from File など) と各シンク ステージ (Write to File など) に対応するフィールドが提供されます。これらのフィールドは、このスケジュールに従ってフローが実行されるときに使用するファイルを示します。デフォルトで、フローのソースとシンクで指定されたファイルが使用されます。スケジュールで使われるファイルを変えるには、ファイル パスを別のファイルへのパスに置き換えます。例えば、フローの Read from File ステージでデータが `C:\FlowInput\Customers.csv` から読み込まれるが、このスケジュールの実行時に `C:\FlowInput\UpdatedCustomers.csv` のデータを使いたい場合は、`C:\FlowInput\UpdatedCustomers.csv` を [Read from File] フィールドに指定します。

   > **注:** ソース ステージやシンク ステージで使われるファイルを変更するには、**リソース - ファイル サーバー** セキュア エンティティ タイプの読み取り権限が必要です。

   フローがスケジュールによって実行される場合、フローで使われるファイルは Management Console で外部リソースとして定義された Spectrum™ Technology Platform サーバーまたは

ファイル サーバーに存在する必要があります。これはジョブと、プロセス フロー内のジョブ アクティビティの両方に適用されます。ソース ステージまたはシンク ステージがクライアント コンピュータ上のファイルを参照している場合は、次の手順のいずれかを実行します。

オプション	説明
オプション 1: データフローを変更する	ファイルを Spectrum™ Technology Platform サーバーまたはファイル サーバーに移動し、データフローを変更します。 1. Enterprise Designer でデータフローを開きます。 2. ソースまたはシンク ステージをダブルクリックします。 3. **[ファイル名]** フィールドで、参照ボタンをクリックします。 4. **[リモート マシン]** をクリックして、必要なファイルを選択します。 **注：** Spectrum™ Technology Platform サーバーと同じコンピュータ上で Enterprise Designer を実行している場合、[リモート マシン] のクリックは、[マイ コンピュータ] をクリックするのと同じであるように見えます。しかし、ファイルが Spectrum™ Technology Platform サーバー上に存在することをシステムに認識させるためには、[リモート マシン] を使用してファイルを選択する必要があります。
オプション 2: このスケジュールの実行時にデータフローのファイルの場所をオーバーライドする	このスケジュールを実行するときに、フロー内のファイル参照をオーバーライドします。これを行うには、ソース フィールドとシンク フィールドに表示されているデフォルト ファイルを Management Console で定義された Spectrum™ Technology Platform サーバー上のファイルまたはファイル サーバー リソースへのパスで置き換えます。

8. **[トリガー]** フィールドで、**[コントロール ファイル]** を選択します。
9. **[コントロール ファイル]** フィールドで、フローをトリガするコントロール ファイルのフル パスと名前を指定します。ファイル名は、そのまま指定するか、ワイルドカードとしてアスタリスク (*) を使用できます。たとえば、`*.trg` と指定すると、フォルダに拡張子 `.trg` の任意のファイルが現れたときフローがトリガされます。

   コントロール ファイルの存在は、フローで必要とされるファイルがすべて揃っていて、フローで使用される準備が整っていることを意味します。

コントロール ファイルは空白のファイルでもかまいません。ジョブの場合、コントロール ファイルは、Write to File ステージまたは Read from File ステージで設定されるファイル パスのオーバーライドを指定できます。コントロール ファイルを使用してファイル パスをオーバーライドするには、Read from File ステージまたは Write from File ステージの名前と、その入力ファイルまたは出力ファイルを最後の引数として指定します。

```
stagename=filename
```

例:

```
Read\ from\ File=file:C:/myfile_input.txt
Write\ to\ File=file:C:/myfile_output.txt
```

コントロール ファイルで指定するステージ名は、フロー内のステージ アイコンの下に表示されるステージ ラベルと一致する必要があります。例えば、入力ステージのラベルが "Read From File" である場合は、次のように指定します。

```
Read\ From\ File=file:C:/inputfile.txt
```

入力ステージのラベルが "Illinois Customers" である場合は、次のように指定します。

```
Illinois\ Customers=file:C:/inputfile.txt
```

Read from File または Write to File の場所をオーバーライドする場合は、次のガイドラインに従ってください。

- パスの先頭に "file:" プロトコルを付けます。例えば、Windows では、"`file:C:/myfile.txt`" と指定し、Unix または Linux では、"`"file:/testfiles/myfile.txt"`" と指定します。
- ファイルの内容には、ASCII 形式の ISO-8559-1 (Latin-1) 互換文字エンコーディングを使用する必要があります。
- ファイルのパスには、バックスラッシュではなくスラッシュを使用する必要があります。
- ステージ名に含まれるスペースは、バックスラッシュでエスケープする必要があります。
- ステージ名は大文字と小文字を区別します。

> **注:** コントロール ファイルによるトリガーを使うスケジュールが複数ある場合は、それぞれで異なるコントロール ファイルをモニタリングする必要があります。そうしなければ、同じコントロール ファイルで複数のジョブまたはプロセス フローがトリガされ、予期せぬ動作を引き起こす恐れがあります。構造上の理由から、すべての必要なファイルとコントロールファイルを専用ディレクトリに配置することをお勧めします。

10. **[ポーリング間隔]** フィールドに、コントロール ファイルの有無をチェックする間隔を指定します。例えば、10 を指定するとモニタリング対象フォルダ内にコントロール ファイルがないか 10 秒間隔でチェックされます。

デフォルトでは、60 秒に設定されています。

11. **[作業フォルダ]** フィールドに、フローの実行中にコントロール ファイルを一時的に配置するフォルダを指定します。Spectrum™ Technology Platform は、フローの実行前に、モニタリング対象フォルダから作業フォルダにファイルをコピーします。これによりモニタリング対象フォルダが空になるため、同じコントロール ファイルによって再びフローが開始されることがなくなります。
12. **[作業フォルダのオプション]** フィールドで、データフローの実行終了時に作業フォルダ内のファイルをどう処理するかを指定します。

保持	ファイルを現在の場所に現在の名前で保持します。このオプションを選択した場合、作業フォルダ内のファイルはこのスケジュールが実行されるたびに上書きされます。
Move to	ファイルを作業フォルダから指定のフォルダに移動します。これにより、作業フォルダにあったファイルを別の場所に移動し、次回ファイル モニターが実行される際に上書きされないよう、これらのファイルを保持することができます。また、このオプションを使用して、別のフローなどの下流プロセスをトリガするように、ファイルを別のモニタリング対象フォルダに移動することもできます。
Rename with time stamp	作業フォルダ内のファイル名にタイムスタンプを付加します。これにより、作業フォルダ内のファイルのコピーを保持できます。フォルダの名前を変更すると、一意の名前が与えられるため、ファイル モニターがフローを次回実行してもファイルが上書きされなくなるからです。
削除	フローの実行終了時に作業フォルダ内のファイルを削除します。

13. フローが電子メール通知用に設定されている場合は、スケジュールの実行時に通知が送信される受信者を追加で指定できます。ここで指定した受信者は、フローの通知設定で指定された受信者に追加される形で通知を受け取ります。通知を送信するフローを設定するには、Enterprise Designer でフローを開き、**[編集] > [通知]** を選択します。
14. **[保存]** をクリックします。

例: モニタリング対象フォルダと作業フォルダ

例えば、あなたは自動車修理店を営んでいるとします。あなたは、毎日、前日に訪れた顧客に、今後のサービスの割引クーポンを送付したいと考えています。これを達成するために、この日の顧客のリストを取得し、顧客名の大文字と小文字の区別が正しいことを確認し、住所を検証するフローを用意しているとします。また、別のシステムによって、その日の顧客のリストが毎日夕方に生成されます。あなたは、このシステムで生成された、顧客リストのファイルを、フローへの入力に使用しようと考えています。

顧客リストを生成するシステムにより、顧客リストが `DailyCustomerReport` という名前のフォルダに格納されます。さらに、完了すると、空白のトリガ ファイルがこのフォルダに格納されます。このフォルダを監視するように Spectrum™ Technology Platform を設定するには、トリガ ファイルを次のように指定します。

```
C:\DailyCustomerReport\*.trg
```

これにより、拡張子が `.trg` のファイルがこのフォルダ内に生成されるたびに、Spectrum™ Technology Platform によってフローが実行されます。特定のファイル名を指定することも可能ですが、この例ではワイルド カードを使用しています。

`DailyCustomerReport` フォルダで `.trg` ファイルが検出された場合、Spectrum™ Technology Platform は、フローを実行する前にこのファイルを別のフォルダに移動する必要があります。ファイルが移動されない場合、次回のポーリング間隔でこのファイルが再度検出され、このフローが再度実行されてしまいます。そのため、ファイルを「作業フォルダ」に移動します。フローを実行する間、ファイルはそのフォルダに格納されています。作業フォルダとして、`C:\SpectrumWorkingFolder` を選択します。

顧客リストを処理するフローが完了した後に、トリガ ファイルを別の場所に移動して、そこから、請求プロセスをトリガするとします。そのため、**[フォルダへ移動]** オプションを選択し、`C:\DailyBilling` という名前のフォルダを選択します。

この例では、トリガ ファイルは最初 `C:\DailyCustomerReport` に配置され、次に作業フォルダ `C:\SpectrumWorkingFolder` に移動されます。フローの完了後、トリガ ファイルは、請求プロセスを開始するために、`C:\DailyBilling` に移動されます。

コマンド ライン実行

コマンド ラインからのジョブの実行

コマンド ラインからジョブを実行するには、その前に、ジョブをエクスポーズする必要があります。ジョブをエクスポーズするには、Enterprise Designer でジョブを開き、**[ファイル] > [エクスポーズ/アンエクスポーズして保存]** を選択します。

コマンドラインからジョブを実行するには、ジョブを実行するシステムに Job Executor ユーティリティをインストールする必要があります。Job Executor は Spectrum™ Technology Platform サーバー上の Spectrum™ Technology Platform Welcome ページ (例えば、http://myserver:8080) にあります。

使用方法

`java -jar jobexecutor.jar -u`*UserID* `-p` *Password*`-j` *Job* `[`オプションの引数`]`

必須	引数	説明
いいえ	`-?`	使用方法を出力します。
いいえ	`-d` *delimiter*	インスタンス/ステータス区切り文字を設定します。これは、同期出力にのみ表示されます。
いいえ	`-e`	Spectrum™ Technology Platform サーバーとの通信にセキュアな HTTPS 接続を使用します。
いいえ	`-f` *property file*	ジョブ プロパティ ファイルへのパスを指定します。ジョブ プロパティ ファイルには Job Executor の引数が含まれています。ジョブ プロパティ ファイルの詳細については、ジョブ プロパティ ファイルの使用（214ページ）を参照してください。
いいえ	`-h` *host name*	Spectrum™ Technology Platform サーバーの名前または IP アドレスを指定します。
いいえ	`-i` *poll interval*	完了したジョブを確認する間隔を秒数で指定します。これは、同期モードにのみ適用されます。
はい	`-j` *job name*	実行するジョブをカンマで区切って指定します。ジョブ名は大文字と小文字が区別されます。ジョブは、列挙された順序で開始されます。
いいえ	`-n` *email list*	ジョブ通知設定に対する追加の電子メール アドレスをカンマで区切って指定します。
いいえ	`-o` *property file*	フロー オプション プロパティ ファイルのパスを指定します。フロー オプション プロパティ ファイルを使用すると、フローのステージ オプションを設定できます。プロパティ ファイルを使用してフロー オプションを設定

必須	引数	説明

		するには、実行時にステージオプションをエクスポーズするようにフローを構成する必要があります。詳細については、「**フロー実行時オプションの追加**（225ページ）」を参照してください。
		
		以下に、Assign GeoTAX Info ステージを含むフローのフロー オプション プロパティ ファイルの例を示します。
		
		`OutputCasing=U`
		`UseStreetLevelMatching=N`
		`TaxKey=T`
		`Database.GTX=gsl`

はい	`-p` *password*	ユーザのパスワードです。
いいえ	`-r`	ジョブについての詳細レポートを返すには、この引数を指定します。このオプションは、`-w` も指定している場合にのみ機能します。レポートには次の情報が含まれます。 • **位置 1** - ジョブの名前 • **位置 2** - ジョブ プロセス ID • **位置 3** - ステータス • **位置 4** - 開始日時 (MM/DD/YYYY HH:MM:SS) • **位置 5** - 終了日時 (MM/DD/YYYY HH:MM:SS) • **位置 6** - 成功したレコードの数 • **位置 7** - 失敗したレコードの数 • **位置 8** - 形式に誤りのあるレコードの数 • **位置 9** - 現在使用されていない 例: `MySimpleJob|4|succeeded|04/09/2019 14:50:47|04/09/2019 14:50:47|100|0|0|` 情報は、`-d` 引数で指定された区切り文字で区切られて出力されます。

必須	引数	説明
いいえ	`-s` *port*	Spectrum™ Technology Platform サーバーが実行されているソケット (ポート)。デフォルト値は 8080 です。
いいえ	`-t` *timeout*	同期モードの場合のタイムアウト (秒単位) を設定します。デフォルト値は 3600 です。最大数は 2147483 です。これは、すべてを合わせたグローバルなタイムアウト値で、spawn されたすべてのジョブが完了するまでの最大待機時間を表します。
はい	`-u` *user name*	ユーザのログイン名を指定します。
いいえ	`-v`	詳細な出力を返します。
いいえ	`-w`	Job Executor を同期モードで実行します。この場合、Job Executor はジョブが完了するまで実行されたままになります。 `-w` を指定しない場合は、Job Executor はジョブの開始後に終了します。ただし、ジョブがサーバー上のファイルを読み書きする場合は、Job Executor はローカル ファイルがすべて処理されるまで実行され、その後で終了します。
いいえ	*StageName=Protocol:FileName*	Read from File または Write to File で指定された入力または出力ファイルをオーバーライドします。詳細については、「ジョブ ファイルの場所のオーバーライド（209ページ）」を参照してください。
いいえ	*StageName*:schema=*Protocol*:*SchemaFile*	Read from File または Write to File で指定されたファイル レイアウト定義を、スキーマ ファイルで定義されたファイル レイアウト定義でオーバーライドします。詳細については、「コマンド ラインでのファイル フォーマットのオーバーライド（211ページ）」を参照してください。

Job Executor の使用例

この例では、コマンド ラインでの起動と出力を示します。

```
D:\spectrum\job-executor>java -jar jobexecutor.jar -u user123
 -p "mypassword" -j validateAddressJob1 -h spectrum.example.com
 -s 8888 -w -d "%" -i 1 -t 9999

validateAddressJob1%105%succeeded
```

この例の出力からは、'validateAddressJob1' という名前のジョブ (識別子 105) がエラーを生じることなく実行したことがわかります。その他の実行結果としては、"失敗" と "実行中" があります。

ジョブ ファイルの場所のオーバーライド

Job Executor または管理ユーティリティを使用してコマンド ラインでジョブを実行する際に、フローのソース ステージ (Read from File など) で指定された入力ファイル、およびフローのシンク ステージ (Write to File など) で指定された出力ファイルをオーバーライドできます。

Job Executor でこれを行うには、Job Executor コマンドの終わりに次のコマンドを指定します。

`StageName=Protocol:FileName`

管理ユーティリティでは、`job execute` コマンドで次のように `--l` 引数を使用します。

`--l` *`StageName=Protocol:FileName`*

説明:

StageName

Enterprise Designer で、フローのステージのアイコンの下に表示されるステージ ラベル。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read from File` と指定します。

埋め込まれたフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたフローまたはサブフローの名前とピリオドを付けて、次のようにします。

`EmbeddedOrSubflowName.StageName`

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write to File
```

別の埋め込まれたフローの中にある埋め込まれたフロー内のステージを指定するには、親フローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded Dataflow 1.Embedded Dataflow 2.Write to File
```

Protocol

通信プロトコルには、次のいずれかのタイプを指定できます。

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```
と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注：** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー/ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

FileName

入力または出力として使用するファイルへのフル パス。

注: ファイルのパスには、スラッシュを使用する必要があります。バックスラッシュは使用できません。

複数のオーバーライドを指定する場合は、カンマで区切ってください。

ファイルのオーバーライドの例

必要な Job Executor コマンドでは、ファイル `C:/myfile_input.txt` を Read from File ステージの入力ファイルとして使用し、ファイル `C:/myfile_output.txt` を Write to File ステージの出力ファイルとして使用します。

```
java -jar jobexecutor.jar -j Job1 -u Bob1234 -p "" "Read from
File"="file:C:/myfile_input.txt" "Write to
File"="file:C:/myfile_output.txt"
```

コマンド ラインでのファイル フォーマットのオーバーライド

Job Executor または管理ユーティリティを使用してジョブを実行する際に、フローの Read from File ステージおよび Write to File ステージで指定されたファイルのファイル レイアウト (またはスキーマ) をオーバーライドできます。

Job Executor でこれを行うには、Job Executor コマンド ライン コマンドの終わりに以下を指定します。

StageName`:schema=`*Protocol*`:`*SchemaFile*

管理ユーティリティでは、`job execute` コマンドで次のように `--l` 引数を使用します。

`--l` *StageName*`:schema=`*Protocol*`:`*SchemaFile*

説明:

StageName

Enterprise Designer で、フローのステージのアイコンの下に表示されるステージラベル。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read from File` と指定します。

埋め込まれたフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたフローまたはサブフローの名前とピリオドを付けて、次のようにします。

`EmbeddedOrSubflowName.StageName`

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write to File
```

別の埋め込まれたフローの中にある埋め込まれたフロー内のステージを指定するには、親フローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded Dataflow 1.Embedded Dataflow 2.Write to File
```

Protocol

通信プロトコル:

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```

と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注：** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生し

ました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

SchemaFile

使用するレイアウトを定義したファイルへのフル パス。

> **注:** ファイルのパスには、スラッシュを使用する必要があります。バックスラッシュは使用できません。

スキーマ ファイルを作成するには、目的のレイアウトを Read from File または Write to File で定義し、**[エクスポート]** ボタンをクリックして、レイアウトを定義する XML ファイルを作成します。

> **注：** Job Executor を使用する場合は、スキーマ ファイル内のフィールドのデータ タイプをオーバーライドできません。FieldSchema 要素の子である <Type> 要素の値は、フローの Read from File ステージまたは Write to File ステージで指定されたフィールドのタイプと一致している必要があります。

ファイル フォーマットのオーバーライドの例

以下の Job Executor コマンドでは、ファイル `C:/myschema.xml` を Read from File ステージによって読み込まれるファイルのレイアウト定義として使用します。

```
java -jar jobexecutor.jar -j Job1 -u Bob1234 -p "" "Read from File":schema="file:C:/myschema.xml"
```

ジョブ プロパティ ファイルの使用

ジョブ プロパティ ファイルには、ジョブの実行を制御する引数が含まれており、Job Executor または管理ユーティリティを使用したジョブの実行を制御できます。引数を再使用する場合に、各引数をコマンド ラインで個別に指定するのではなく、単一の引数 (`-f`) を指定するには、ジョブ プロパティ ファイルを使用します。

プロパティ ファイルを作成するには、各行に 1 つの引数を含むテキスト ファイルを作成します。

```
d %
h spectrum.mydomain.com
i 30
j validateAddressJob1
u user
p password
s 8888
t 9999
w true
```

ジョブ プロパティ ファイルには次の引数を含めることができます。

必須	引数	説明
いいえ	`?`	使用方法を出力します。
いいえ	`d` *delimiter*	インスタンス/ステータス区切り文字を設定します。これは、同期出力にのみ表示されます。
いいえ	`e`	Spectrum™ Technology Platform サーバーとの通信にセキュアな HTTPS 接続を使用します。
いいえ	`h` *hostname*	Spectrum™ Technology Platform サーバーの名前または IP アドレスを指定します。
いいえ	`i` *pollinterval*	完了したジョブを確認する間隔を秒数で指定します。これは、同期モードにのみ適用されます。
はい	`j` *jobname*	実行するジョブをカンマで区切って指定します。ジョブ名は大文字と小文字が区別されます。ジョブは、列挙された順序で開始されます。

必須	引数	説明
いいえ	`n` *emaillist*	ジョブ通知設定に対する追加の電子メール アドレスをカンマで区切って指定します。
はい	`p` *password*	ユーザのパスワードです。
いいえ	`r`	標準出力に出力されるジョブに関する以下の情報を、区切り文字で区切って返します。 • **位置 1** - ジョブの名前 • **位置 2** - ジョブ プロセス ID • **位置 3** - ステータス • **位置 4** - 開始日時 (MM/DD/YYYY HH:MM:SS) • **位置 5** - 終了日時 (MM/DD/YYYY HH:MM:SS) • **位置 6** - 成功したレコードの数 • **位置 7** - 失敗したレコードの数 • **位置 8** - 形式に誤りのあるレコードの数 • **位置 9** - 現在使用されていない 情報は、-d 引数で指定された区切り文字で区切られて出力されます。例: `MySimpleJob|4|succeeded|04/09/2019 14:50:47|04/09/2019 14:50:47|100|0|0|`
いいえ	`s` *port*	Spectrum™ Technology Platform サーバーが実行されているソケット (ポート)。デフォルト値は 8080 です。
いいえ	`t` *timeout*	同期モードの場合のタイムアウト (秒単位) を設定します。デフォルト値は 3600 です。最大数は 2147483 です。これは、すべてを合わせたグローバルなタイムアウト値で、spawn されたすべてのジョブが完了するまでの最大待機時間を表します。
はい	`u` *username*	ユーザのログイン名を指定します。
いいえ	`v`	詳細な出力を返します。
いいえ	`w`	同期モードにおいてジョブの完了を待つように指定します。

コマンドライン引数とプロパティ ファイルの両方の使用

コマンド ライン入力とプロパティ ファイル入力を組み合わせて使用することもできます。例:

```
java -jar jobexecutor.jar -f /dcg/job.properties -j job1
```

この場合、コマンドライン引数の方が、プロパティ ファイルで指定された引数よりも優先されます。上の例では、ジョブ job1 の方が、プロパティ ファイルで指定されたジョブよりも優先されることになります。

ジョブ プロパティ ファイルを使用した入出力ファイルのオーバーライド

データフローのソース ステージ (Read from File など) で指定された入力ファイルや、データフローのシンク ステージ (Write to File など) で指定された出力ファイルを、Job Executor のプロパティ ファイルでオーバーライドできます。これを行うには、プロパティ ファイルで次のように指定します。

```
StageName\:file=Protocol:FileName
```

説明:

StageName

Enterprise Designer で、データフローのステージのアイコンの下に表示されるステージ ラベル。ステージ名に空白、コロン、または等号が含まれる場合は、その前にバックスラッシュを挿入します。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read\ from\ File` と指定します。

```
Embedded\ Dataflow\ 1.Embedded\ Dataflow\ 2.Write\ to\ File
```

埋め込まれたデータフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたデータフローまたはサブフローの名前とピリオドを付けて、次のようにします。

```
EmbeddedOrSubflowName.StageName
```

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write\ to\ File
```

別の埋め込まれたデータフローの中にある埋め込まれたデータフロー内のステージを指定するには、親データフローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded\ Dataflow\ 1.Embedded\ Dataflow\ 2.Write\ to\ File
```

> **注:** ステージ名の後に `:file` を入れる必要があります。例えば、`Read\ from\ File:file` とします。これは、コマンド ラインでファイルをオーバーライドする場合の構文とは異なります。コマンド ラインではステージ名の後に `:file` を指定しません。

Protocol

通信プロトコル。次のいずれかです。

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```
と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注：** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー/ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー/ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

例

以下のプロパティ ファイルでは、最後の 2 行で、Read from File ステージと Write to File ステージ用のファイルを指定しています。

```
j=testJob
h=myspectrumserver.example.com
s=8080
u=david1234
p=mypassword1234
Read\ from\ File\:file=file:C:/myfile_input.txt
Write\ to\ File\:file=file:C:/myfile_output.txt
```

ジョブ プロパティ ファイルを使用したファイル形式のオーバーライド

プロパティ ファイルを使用して、データフローの Read from File ステージおよび Write to File ステージで指定されたファイルのファイル レイアウト (またはスキーマ) をオーバーライドできます。これを行うには、プロパティ ファイルで次のように指定します。

```
StageName\:schema=Protocol:SchemaFile
```

説明:

StageName

Enterprise Designer で、データフローのステージのアイコンの下に表示されるステージ ラベル。ステージ名に空白、コロン、または等号が含まれる場合は、その前にバックスラッシュを挿入します。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read\ from\ File` と指定します。

```
Embedded\ Dataflow\ 1.Embedded\ Dataflow\ 2.Write\ to\ File
```

埋め込まれたデータフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたデータフローまたはサブフローの名前とピリオドを付けて、次のようにします。

```
EmbeddedOrSubflowName.StageName
```

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write\ to\ File
```

別の埋め込まれたデータフローの中にある埋め込まれたデータフロー内のステージを指定するには、親データフローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded\ Dataflow\ 1.Embedded\ Dataflow\ 2.Write\ to\ File
```

> **注:** ステージ名の後に `:file` を入れる必要があります。例えば、`Read\ from\ File:file` とします。これは、コマンド ラインでファイルをオーバーライドする場合の構文とは異なります。コマンド ラインではステージ名の後に `:file` を指定しません。

Protocol

通信プロトコル。次のいずれかです。

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```

と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注:** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

SchemaFile

使用するレイアウトを定義したファイルへのフル パス。

> **注:** ファイルのパスには、スラッシュを使用する必要があります。バックスラッシュは使用できません。

スキーマ ファイルを作成するには、目的のレイアウトを Read from File または Write to File で定義し、**[エクスポート]** ボタンをクリックして、レイアウトを定義する XML ファイルを作成します。

> **注:** Job Executor を使用する場合は、スキーマ ファイル内のフィールドのデータ タイプをオーバーライドできません。FieldSchema 要素の子である <Type> 要素の値は、フローの Read from File ステージまたは Write to File ステージで指定されたフィールドのタイプと一致している必要があります。

例

以下のプロパティ ファイル例では、最終行において、Read from File ステージで定義されたファイル レイアウトをファイル `inputSchema.xml` で定義されたレイア

ウトでオーバーライドします。ステージ名の中の空白の前にはバックスラッシュを挿入します。

```
j=testJob
h=myspectrumserver.example.com
s=8080
u=david1234
p=mypassword1234
Read\ from\ File\:file=esclient:c:/MyData/testInput.txt
Read\ from\ File\:schema=esclient:c:/MyData/inputSchema.xml
```

コマンド ラインからのプロセス フローの実行

コマンド ラインからプロセス フローを実行するには、Process Flow Executor を使用します。Process Flow Executor は、Spectrum™ Technology Platform Welcome ページ (例えば、http://myserver:8080 など) からインストールできます。

注: 管理ユーティリティを使用して、コマンド ラインからプロセス フローを実行することもできます。

使用方法

`java -jar pflowexecutor.jar -r`*ProcessFlowName* `-u` *UserID* `-p` *Password* `[`オプションの引数`]`

必須	引数	説明
いいえ	`-?`	使用方法を出力します。
いいえ	`-d` *DelimiterCharacter*	コマンド実行時にコマンド ラインに表示されるステータス情報を区切るのに使用する区切り文字を設定します。デフォルトは "|" です。例えば、デフォルト文字を使用して、"MyProcessflow" というプロセス フローを実行した場合、コマンド ラインには次のメッセージが表示されます。 `MyProcessflow|1|Succeeded`
いいえ	`-e`	Spectrum™ Technology Platform サーバーとの通信に HTTPS 接続を使用します。 **注:** ファイル オーバーライドを指定する場合は、この引数を最後の引数として指定しないでください。

必須	引数	説明
いいえ	`-f` *PropertyFile*	プロパティ ファイルへのパスを指定します。プロパティ ファイルの詳細については、プロセス フローのプロパティ ファイルの使用（223ページ） を参照してください。
いいえ	`-h` *HostName*	Spectrum™ Technology Platform サーバーの名前または IP アドレスを指定します。
いいえ	`-i` *PollInterval*	完了したジョブを確認する間隔を秒数で指定します。デフォルト値は "5" です。
はい	`-p` *Password*	ユーザのパスワードです。必須
はい	`-r` *ProcessFlowNames*	実行するプロセス フローのリストをカンマで区切って指定します。必須 **注：** ファイル オーバーライドを指定する場合は、この引数を最後の引数として指定しないでください。
いいえ	`-s` *Port*	Spectrum™ Technology Platform サーバーが実行されているソケット (ポート)。デフォルト値は 8080 です。
いいえ	`-t` *Timeout*	このオプションは非推奨で、将来は無視されます。
はい	`-u` *UserName*	ユーザのログイン名を指定します。必須
いいえ	`-v` *Verbose*	詳細な出力を返します。ここで、*Verbose* は次のいずれかです。 **true** 詳細な出力を返します。 **false** 詳細な出力を返しません。 **注：** ファイル オーバーライドを指定する場合は、この引数を最後の引数として指定しないでください。
いいえ	`-w` *WaitToComplete*	このオプションは非推奨で、将来は無視されます。
いいえ	*StageName=FileName*	ジョブで指定されている入力または出力ファイルをオーバーライドします。詳細については、「プロセス フロー ファイルの場所のオーバーライド（224ページ）」を参照してください。

例

プロセス フロー名、ユーザ ID、パスワードを指定した基本的なコマンド ライン エントリを次に示します。

```
java -jar pflowexecutor.jar -r MyFlow1 -u Bob1234 -p
"mypassword1"
```

次の例では、前の例と同じ情報に加えて引数を指定しています。

```
java -jar pflowexecutor.jar -r Flow1 -u Bob1234 -p
"mypassword1" -h spectrum.example.com -s 8080 -w -d "%" -i
1
```

この例では、コマンド ラインでの起動と出力を示します。

```
D:\spectrum\pflow-executor>java -jar pflowexecutor.jar -u
Bob1234 -p "mypassword1" -r
validateAddressFlow1 -h spectrum.example.com -s 8080 -w -d
"%" -i
1 -t 9999
validateAddressJob1%111%succeeded
```

この例では、validateAddressFlow1 という名前のプロセス フローが実行しました (ID は 111)。エラーは発生していません。その他の実行結果としては、"失敗" と "実行中" があります。

プロセス フローのプロパティ ファイルの使用

プロパティ ファイルには引数が含まれ、Process Flow Executor で `-f` 引数を使ってプロパティ ファイルのパスを指定することにより、引数を再利用できます。プロパティ ファイルには、少なくともプロセス フロー (`r`)、ユーザ ID (`u`)、およびパスワード (`p`) が含まれる必要があります。

1. テキスト エディタを起動します。
2. 次の例で示すように、1 行で 1 つの引数を指定します。引数の一覧については、コマンド ラインからのプロセス フローの実行（221ページ）を参照してください。

   注： プロパティ ファイルを使用して、入力ファイルまたは出力ファイルをオーバーライドすることはできません。入力ファイルまたは出力ファイルのオーバーライドは、コマンド ライン引数を使用することによってのみ行うことができます。

```
d=%
h=myserver.mydomain.com
i=30
u=user
p=password
```

```
r=MyFlow1
s=8888
```

3. `*.properties` という拡張子を付けてこのファイルを保存します (`example.properties` など)。
4. Process Flow Executor を実行するときに、`-f` 引数を使用してプロパティ ファイルのパスを指定します。コマンド ライン入力とプロパティ ファイル入力を組み合わせて使用することもできます。コマンド ラインでの引数の方が、プロパティ ファイルで指定されている引数より優先されます。
   ```
   java -jar pflowexecutor.jar -f /dcg/flow.properties -r MyFlow2
   ```
   この例では、プロセス フロー MyFlow2 が、プロパティ ファイルで指定されているプロセス フローより優先されます。

プロセス フロー ファイルの場所のオーバーライド

Process Flow Executor コマンド ライン ツールを使ってプロセス フローを実行する場合、プロセス フローから参照されるジョブに指定された入力/出力ファイルとは別のファイルを使うよう指定できます。これを行うには、Read from File ステージ名または Write from File ステージ名とともに、最後の引数として入力ファイルまたは出力ファイルを次のように指定します。

```
"<jobname>|<stagename>"="<filename>"
```

説明:

JobName

は、プロセス フロー内で参照されるジョブの名前です。

StageName

データフロー内のステージのアイコンの下にあるステージ ラベルに表示される、ジョブの Read from File ステージまたは Write to File ステージの名前。例えば、入力ステージのラベルが "Read From File" である場合は、次のように指定します。

```
"Job1|Read From File"="file:C:/inputfile.txt"
```

入力ステージのラベルが "Illinois Customers" である場合は、次のように指定します。

```
"Job1|Illinois Customers"="file:C:/inputfile.txt"
```

File

ファイルのプロトコルとフル パス。ファイルのパスには、バックスラッシュではなくスラッシュを使用する必要があります。プロトコルは次のいずれかを指定します。
file:

ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータに存在する場合は、パスの先頭に "file:" プロトコルを付けます。例えば、Windows では、"file:C:/myfile.txt" と指定し、Unix または Linux では、"file:/testfiles/myfile.txt" と指定します。

> **注:** クライアントとサーバーが同じコンピュータ上で稼働している場合は、"file:" プロトコルを使用しても "esclient:" プロトコルを使用しても構いませんが、"file:" プロトコルを使用した方がパフォーマンスが高くなる可能性があります

esclient:

ファイルが Process Flow Executor と同じコンピュータに存在する場合は、パスの先頭に "esclient:" プロトコルを付けます。例えば、Windows では、"esclient:C:/myfile.txt" と指定し、Unix または Linux では、"esclient:/testfiles/myfile.txt" と指定します。

> **注:** Process Flow Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*/server/conf/spectrum-container.properties。プロパティ spectrum.runtime.hostname にサーバーの IP アドレスを設定します。

ftp:

Management Console で定義されたファイル サーバーを使用するには、ftp:*NameOfFileServer*/*PathToFile* という形式を使用します。例えば、ftp://FS/testfiles/myfile.txt と指定します。ここで FS は、Management Console で定義されたファイル サーバー リソースです。

例を次に示します。

```
java -jar pflowexecutor.jar -r Flow1 -u Bob1234 -p "mypassword1" -h
spectrum.example.com -s 8080 -w -d "%" -i 1 "Job1|Read from
File"="file:C:/myfile_input.txt" "Job1|Write to
File"="file:C:/myfile_output.txt"
```

フロー実行時オプションの追加

フロー実行時オプションを使用すると、フロー実行時のステージの動作を制御できます。これは、実行時にフローの動作を変更したい場合に役立ちます。例えば、フローの Read from DB ステー

ジ内で指定したデータベースの代わりに、Read from DB ステージ用の特定のソース データベースを、フローの実行時に指定できます。

以下では、実行時に設定できるオプションをエクスポーズする手順について説明します。この手順を実行すると、実行時に次のような手法でフロー オプションを設定できるようになります。

- ジョブの場合、フロー オプション プロパティ ファイルと Job Executor の -o 引数を使用して実行時オプションを指定できます。
- サービスの場合、実行時オプションを API オプションとして指定できます。
- Web サービスとしてエクスポーズされたサービスの場合、実行時オプションをリクエスト内のパラメータとして指定できます。
- サブフローの場合、親フロー タイプ (ジョブ、サービス、Web サービスとしてエクスポーズされたサービス) に応じて、実行時オプションは親フローから継承され、上記のいずれかの方法によってエクスポーズされます。

フローに実行時オプションを追加するには、以下の手順に従います。

1. Enterprise Designer でフローを開きます。
2. 埋め込まれたフロー内のステージに対して実行時オプションを設定する場合は、埋め込まれたフローを開きます。
3. ツールバー上のデータフロー オプション アイコンをクリックするか、**[編集] > [データフロー オプション]** をクリックします。**[データフロー オプション]** ダイアログ ボックスが表示されます。
4. **[追加]** をクリックします。**[データフロー オプションの定義]** ダイアログ ボックスが表示されます。
5. **[オプション名]** フィールドに、このオプションに付ける名前を入力します。このオプションを設定するには、実行時にこのオプション名を指定する必要があります。
6. **[ラベル]** フィールドで、別のラベルを指定できます。また、オプション名と同じラベルを使用することもできます。
7. **[説明]** フィールドにオプションの説明を入力します。
8. **[ターゲット]** フィールドで、このオプションをフローのすべてのステージに適用するか、特定のステージにのみ適用するかを選択します。

   選択ステージ

   指定するステージにのみオプションを適用する場合は、このオプションを選択します。

   すべてのステージ

   フローのすべてのステージにオプションを適用する場合は、このオプションを選択します。

   トランスフォームを含む

フローの Transformer ステージの Custom Transform で実行時オプションを使用できるようにする場合は、このオプションを選択します。このオプションを選択すると、Groovy スクリプトでトランスフォームを実行する際に、指定した値にアクセスできます。この場合、次の構文を使用します。

```
options.get("optionName")
```

例えば、`casing` という名前のオプションにアクセスするには、Custom Transform スクリプトに次の式を含めます。

```
options.get("casing")
```

9. **[ターゲット]** フィールドで **[選択ステージ]** を選択した場合は、**[データフロー オプションをステージにマッピングします]** テーブルにフローのステージの一覧が表示されます。フロー オプションとしてエクスポーズするオプションを選択します。最初の項目を選択すると、**[デフォルト値]** フィールドと **[有効値]** フィールドにデータが表示されます。

   注： 複数のオプションを選択すると、フロー オプションによって複数のステージ オプションを制御できます。この場合、選択するどのステージ オプションも有効値を共有する必要があります。例えば、あるオプションの値が「Y」と「N」である場合、追加されるどのオプションも値のセットに「Y」または「N」のいずれかを含む必要があり、実行時に使用できる値も共通する値に限られます。つまり、値が「Y」と「N」であるオプションを選択した場合、値が「E」、「T」、「M」、および「L」であるオプションを選択することはできませんが、値が「P」、「S」、および「N」であるオプションは、「N」という値が共通であるため、選択できます。ただし、このオプションに対して使用可能な値は「N」のみで、「Y」、「P」、または「S」は使用できません。

10. 実行時に指定できる値を制限するには、**[有効値]** フィールドのすぐ右にあるアイコンをクリックしてオプションを編集します。
11. デフォルト値を変更する場合は、**[デフォルト値]** フィールドに別の値を指定します。

    注： サービスの場合、デフォルト値はサービスを最初にエクスポーズする前にしか変更できません。サービスをエクスポーズした後は、Enterprise Designer を使用してデフォルト値を変更することはできません。代わりに、Management Console を使用する必要があります。詳細については、デフォルト サービス オプションの指定（164ページ）を参照してください。

12. **[OK]** をクリックします。
13. 必要に応じて、オプションの追加を続けます。
14. オプションの追加を終えたら、**[データフロー オプション]** ダイアログ ボックスの **[OK]** をクリックします。

15. 埋め込まれたフローに実行時オプションを追加した場合は、実行時にそのオプションを使用可能にするために、その実行時オプションを親フローとすべての先祖フローで定義する必要があります。これを行うには、埋め込まれたフローを含むフローを開き、作成したオプションをエクスポーズします。必要に応じて、そのフローの親を開き、そこでこのオプションを定義します。すべての先祖でこのフロー オプションが定義されるまで、これを続けます。

    例えば、"A" という名前のフローに、"B" という名前の埋め込まれたフローが含まれており、その中に "C" という名前の埋め込まれたフローが含まれているとします。つまり、埋め込まれたフローは A > B > C という階層構造を持ちます。Casing という名前のオプションを埋め込まれたフロー "C" 内のステージでエクスポーズしたい場合、埋め込まれたフロー "C" を開いてこれを定義します。続いて、埋め込まれたフロー "B" を開いて、このオプションを定義します。最後に、フロー "A" を開いてオプションを定義することで、このオプションが実行時に使用可能になります。

これで、実行時にオプションを指定できるようにフローが設定されました。

8 - パフォーマンス

このセクションの構成

パフォーマンス チューニングのチェックリスト

このチェックリストでは、Spectrum™ Technology Platform 環境で最適なパフォーマンスを得るために推奨される方法について説明します。各手法は、パフォーマンスに対する影響が最も高いものから最も低いものの順に記載されています。

パフォーマンス設定	説明	追加情報
データフロー設計	パフォーマンスに対する影響が最も高いのは、データフローの設計です。いくつかのベスト プラクティスに従うことによって、データフローのパフォーマンスを確実に高めることができます。	**パフォーマンス最適化のための設計ガイドライン**（231ページ）
データフローのプール サイズ	データフローのプール サイズは、サービス データフローが1度に実行できるインスタンス数を制御します。データフローのプール サイズのデフォルト値は 8 です。	**データフローのプール サイズ**（232ページ）
データベースのプール サイズと、ステージ実行時インスタンス数	データベースのプール サイズとステージ実行時インスタンス数は、複数の同時要求を処理するシステムの能力を制御します。両者を連動させて調整する必要があります。	**データベースのプール サイズと実行時インスタンス数**（157ページ）
ソート パフォーマンス	大きなデータ セットのソートは、バッチ処理中に実行される操作で最も時間がかかるものの1つになることがあります。ソート パフォーマンス オプションによってメモリとディスクの使用率を制御すると、使用可能なメモリとディスクの容量を十分に活用できます。	**デフォルトのソート パフォーマンス オプションの設定**（237ページ）
リモート コンポーネントのオプション	これらの設定は、住所検証やジオコーディングなどの特定のステージのメモリ使用率を制御します。	**リモート コンポーネントのオプションの設定**（238ページ）
個々のステージ	特定の種類の処理を最適化するために採用できる、いくつかのアクションがあります。パフォーマンスのベスト プラクティスを参照して、最適なパフォーマンスが得られるようにデータフローが設定されているかどうかを確認してください。	**個々のステージの最適化**（239ページ）
JVM 設定	特定のハードウェアにおいてパフォーマンスを向上させることのできる JVM オプションがあります。	**JVM パフォーマンス チューニング**（245ページ）

パフォーマンス設定	説明	追加情報
マイクロバッチ処理	マイクロバッチ処理は、サーバーに対する1回のリクエストに複数のレコードを含める手法です。	**マイクロバッチ処理**（245ページ）
Elasticsearch のヒープ サイズ設定	Metadata Insights のプロファイリング ツールや、フローのいずれかのマッチング ステージで、メモリを多用する操作を実行している場合は、Elasticsearch のヒープ サイズを大きくします。	**Elasticsearch のヒープ サイズ設定**（239ページ）

パフォーマンス最適化のための設計ガイドライン

パフォーマンスが最適化されるようデータフローを慎重に設計することは、Spectrum™ Technology Platform のパフォーマンス向上のために実行できる最も重要な作業です。これらのガイドラインでは、データフローのパフォーマンス最適化に使用できる手法について説明します。

ステージ数の最小化

Spectrum™ Technology Platform は、並列処理によって高パフォーマンスを実現します。1 つのフロー内の各ステージは、それぞれのスレッドで非同期に実行します。ただし、ある種のデータフローを実行するときプロセッサがオーバースレッドになる可能性があります。これは、"実際の仕事" を行う以上の時間がスレッドの管理に費やされることを意味します。130 の個別ステージを持つデータフローの場合、1 つまたは 2 つのプロセッサを搭載した小規模なサーバーでのパフォーマンスが非常に悪いことが確認されています。

したがって、パフォーマンスに優れたデータフローを設計するためにまず検討すべきは、必要なだけのステージを使用し、それ以上は使用しないということです。必要以上のステージを使用した例としては、次のものがあります。

- 1 つで十分にもかかわらず複数のconditional routerを使用する
- 複数のトランスフォームを 1 つのステージに結合する代わりに複数の Transformer ステージを定義する

幸いなことに、通常はこのようなデータフローを設計し直して、冗長または不要なステージを削除し、パフォーマンスを向上させることができます。

フローが複雑な場合は、埋め込みタイプのフローまたはサブフローを使用してキャンバス上が乱雑にならないように整理することを検討してください。それによって、フローが見やすくなり、操作しやすくなります。埋め込みタイプのフローを使用しても実行時のパフォーマンス上のメリットはありませんが、Enterprise Designer でのフローの操作が容易になります。サブフローを使用して複雑なフローを簡素化すれば、フロー編集時の Enterprise Designer のパフォーマンスが向上します。

レコード長の短縮

同時並行で実行されるステージ間でデータが渡されるため、入力レコードの長さについても検討する必要があります。一般的に、長いレコードの入力では、短いレコードの入力に比べ、処理に時間がかかります。これは、読み込み、書き出し、ソートするデータが多いためです。複数のソート操作を行うデータフローは、レコード長を短くすることで特に恩恵を受けます。非常に長いレコードを処理する場合は、Spectrum™ Technology Platform ジョブを実行する前に入力から不要なフィールドを削除し、その後、生成された出力ファイルにフィールドを戻すことで、処理速度を向上させることができます。

ソートの適切な使用

ソート操作の最小化についても検討する必要があります。ソートは他の操作よりも時間がかかることが多いため、入力レコードの数およびサイズが増加すると問題となる場合があります。しかし、多くの Spectrum™ Technology Platform ステージが、ソートされた入力データを必要としたり、優先したりします。例えば、Universal Addressing モジュールと Enterprise Geocoding モジュールは、入力が国および郵便番号によってソートされている場合にパフォーマンスが最適化されます。Intraflow Match や Interflow Match などのステージでは、入力が [グループ化方法] フィールドによってソートされている必要があります。外部ソート アプリケーションを使用して、入力データを事前にソートできる場合もあります。この場合、Spectrum™ Technology Platform データフロー内でのソートより速く処理できます。

データフローのプール サイズ

データフローのプール サイズは、各サービス データフローが 1 度に実行できるインスタンス数を制御します。プール サイズを大きくすることによってある程度までパフォーマンスを上げることができますが、同時に実行している各サービス データフローの複数のインスタンスを処理するためのプロセッサやメモリ リソースがサーバーにない場合は、プール サイズを大きくするとパフォーマンスが低下する恐れがあります。プロセッサやメモリ リソースが限界まで使用されている場合は、各サービス データフローの同時インスタンス数を制限するデータフローのプール サイズを小さくした方が、全体的にはより好ましいパフォーマンスが得られる可能性があります。

システムに対する適切なプール サイズを見つける際には、データフローのプール サイズによって制限されるのが、同時に実行するサービス データフローの総数ではなく、各サービス データフローのインスタンス数であることに注意してください。例えば、デフォルト設定である 8 は、各サービス データフローが 8 個のインスタンスを 1 度に実行可能であることを意味します。2 つのサービス データフローがそれぞれ最大数である 8 個の同時インスタンスを利用する場合、システム上で同時に実行するサービス データフローのインスタンス総数は 16 となります。

> **注：** データフローのプール サイズは、ジョブではなく、サービスのパフォーマンスのみに影響を与えます。

データフローのプール サイズを設定するには

1. JMX コンソールのパフォーマンス モニターを有効にします。詳細については、「**JMX コンソールによるパフォーマンスのモニタリング**（249ページ）」を参照してください。
2. Web サービスまたは API を介して、サービスに対する複数の同時要求を送信します。
3. JMX コンソールで、**ServiceRuntimeManager.borrow.*DataflowName*** に表示される時間に注目してください。これは、サービス要求が実行前に、プール内の空きを待つ時間を示します。時間はミリ秒単位で表示されます。
4. 次のファイルを開きます。

   ```
   SpectrumLocation\server\app\conf\dataflowpool-pool-sizes.properties
   ```

5. パフォーマンスが向上すると思う値に、プール サイズを増減します。デフォルトのプール サイズは 8 です。
   - **ServiceRuntimeManager.borrow.*DataflowName*** に表示される待ち時間が長い場合は、データフローのプール サイズを大きくして、サービス要求が実行を待機する時間を短くすることを検討してください。あるいは、要求を抑えてサーバー リソースに過度に負荷をかけないようにするために、待ち時間を長くとることもできます。
   - **ServiceRuntimeManager.borrow.*DataflowName*** に表示される待ち時間が長くなく、パフォーマンスが低い場合は、多数の同時サービス データフローを処理するためのプロセッサやメモリ リソースがサーバーにないことを表している可能性があります。データフローのプール サイズを小さくして、要求を抑えることを検討してください。
6. ファイルを保存して閉じます。
7. 複数の同時要求を再度送信し、JMX コンソールで **ServiceRuntimeManager.borrow.*DataflowName*** に表示される待ち時間を確認します。
8. 最適な設定が得られるまで、他のプール サイズも試します。

データベースのプール サイズと実行時インスタンス数

ほとんどの Spectrum™ Technology Platform 環境において、バッチ ジョブであるか、Web サービスまたは API 要求に応答するサービスであるかの違いはあれ、複数のフローが同時に実行しています。同時処理を最適化するために、データベースのプール サイズ設定が使用できます。これによって、Spectrum のデータベースが処理する同時要求数と、同時に実行するフロー ステージのインスタンス数を制御する実行時インスタンス数が制限されます。最適なパフォーマンスを得るには、これら 2 つの設定を同時にチューニングする必要があります。

データベースのプール サイズ

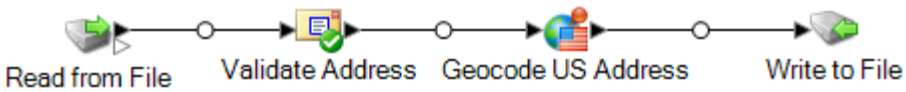
Spectrum データベースには、住所の検証で使用する郵便データや、住所のジオコードで使用するジオコーディング データなど、特定のステージで使用するリファレンス データが含まれます。これらのデータベースは、それを使用するデータフロー ステージやサービスから同時に行われる複数の要求を受け付け、データフローの要求やサービスの要求のパフォーマンスを高めるように設定することが可能です。データベースのプール サイズは、Spectrum のデータベースが処理する同時要求の最大数を設定します。デフォルトでは、Spectrum データベースのプール サイズは 4 で、データベースは 4 つの要求を同時に処理できることを意味します。

最適なプール サイズはモジュールによって異なります。一般的には、サーバーが搭載する CPU の数の半分から 2 倍のプール サイズを設定すると、最適な結果が得られます。ほとんどのモジュールに最適なプール サイズは CPU 数と同数です。例えば、サーバーが 4 つの CPU を搭載している場合は、プール サイズを 2 (CPU 数の半分) ～ 8 (CPU 数の 2 倍) の間で試すことができ、多くの場合、最適なサイズは 4 (CPU 数と同数) です。

プール サイズを変更するときは、データベースにアクセスするステージ用としてデータフローに指定されている実行時インスタンスの数を考慮する必要もあります。例えば、1 つの実行時インスタンスを使用するように設定された Geocode US Address ステージを持つデータフローがあるとします。この場合、米国ジオコーディング データベースのプール サイズを 4 に設定しても、パフォーマンスは向上しません。実行時インスタンスが 1 つしかないので、データベースへの要求は一度に 1 つになります。ただし、Geocode US Address の実行時インスタンスの数を 4 つに増やすと、パフォーマンスが向上します。データベースに同時にアクセスする Geocode US Address のインスタンスが 4 つあるので、プール全体を使用できます。

実行時インスタンス

データフローの各ステージはそれぞれのスレッドで非同期に動作し、他のステージから独立しています。このため、データフロー内の複数ステージが並列処理され、1 つのステージに対して複数の実行時インスタンスを利用することができます。これは、データの処理時間が異なるステージで構成されるデータフローで役に立ちます。その結果、スレッド間での作業の分配のバランスが悪くなる可能性があります。例えば、次の 2 つのステージで構成されるデータフローを考えてみましょう。



ステージの設定によって、Validate Address ステージが Geocode US Address ステージより速くレコードを処理することがあります。その場合、データフロー実行のある時点で、Validate Address はすべてのレコードを処理していますが、Geocode US Address にはまだ未処理のレコードがあります。このデータフローのパフォーマンスを向上させるには、最も速度の遅いステージ (この場合は Geocode US Address) のパフォーマンスを向上させる必要があります。その方法の 1 つは、ステージの実行時インスタンスを複数指定することです。例えば、実行時インスタンス数を 2 に

設定すると、そのステージのインスタンスが2つになります。各インスタンスはそれぞれのスレッド内で動作し、レコードの処理に使用することができます。

一般的なルールとして、実行時インスタンス数は、少なくともリモート コンポーネントのインスタンス数と等しくなければなりません。リモート コンポーネントの詳細については、『*管理ガイド*』を参照してください。複数の実行時インスタンスを指定するとパフォーマンスが向上しますが、この値を高くしすぎると、システムのリソースに負荷がかかり、パフォーマンスが低下する可能性があります。

> **注：** 複数の実行時インスタンスを使用してパフォーマンスが向上するのは、複数のレコードを使用するジョブまたはサービス要求を実行する場合のみです。

チューニング手順

データベースのプール サイズと実行時インスタンス数に適切な値を設定するには、さまざまな設定を試して、リソースに過度に負荷をかけたりパフォーマンスを低下させたりすることなく、使用可能なサーバー リソースを最大限に活用できる設定を見つける必要があります。

> **注：** データベースのプール サイズを調整する前にデータフローのプール サイズを最適化する必要があります。データフローのプール サイズの最適化については、「**データフローのプール サイズ**（232ページ）」を参照してください。

1. まず、さまざまな設定を試す際に使用するサンプル データを見つけます。実行時間の測定や一貫性の確認ができるように、サンプル データセットは十分に大きい必要があります。またサンプル データは、実際に処理するデータを代表するものでなければなりません。例えば、ジオコーディングのパフォーマンス テストを行う場合は、ジオコードする予定のすべての国に対して同数のレコードがテスト データに含まれるようにします。
2. 郵便データベースやジオコーディング データベースなど、データベース リソースの使用が必要なサービスまたはデータフローをテストする場合は、データベースの最新版がインストールされていることを確認してください。
3. サンプル データを用意し、最新のデータベース リソースをインストールしたら、ファイルからデータを読み込み、最適化したいステージでそれを処理し、ファイルに書き出す、簡単なデータフローを作成します。例えば、Validate Address のパフォーマンス設定をテストする場合は、Read from File、Validate Address、Write to File で構成されるデータフローを作成します。
4. データベース リソースのプール サイズを 1 に設定します。
   a. Management Console を開きます。
   b. **[リソース] > [Spectrum データベース]** に移動します。
   c. 最適化するデータベース リソースを選択し、変更ボタン をクリックします。
   d. **[プール サイズ]** フィールドに、1 と入力します。
   e. **[OK]** をクリックします。

5. ステージの実行時インスタンスを 1 に設定します。
   a. Enterprise Designer でデータフローを開きます。
   b. 複数の実行時インスタンスを使用するように設定するステージをダブルクリックします。
   c. **[実行時]** をクリックします。

      注 : すべてのステージで複数の実行時インスタンスを使用できるわけではありません。ステージのウィンドウの下部に **[実行時]** ボタンがない場合、そのステージでは複数の実行時インスタンスを使用できません。

   d. **[ローカル]** を選択して 1 を指定します。
   e. **[OK]** をクリックして **[実行時パフォーマンス]** ウィンドウを閉じます。さらに **[OK]** をクリックしてステージを閉じます。
6. データフローを複数回実行し、次の各項目の平均値を記録することによって、ベースラインパフォーマンスを算出します。
   - 経過時間
   - CPU 使用率
   - メモリ使用率

   ヒント： JMX コンソールを使用してパフォーマンスをモニタリングできます。詳細については、**JMX コンソールによるパフォーマンスのモニタリング**（249ページ）を参照してください。
7. ジョブの複数インスタンスの同時実行をサポートする必要がある場合は、それを実行します。それぞれの場合に対して、経過時間、CPU 使用率、メモリ使用率を記録します。

   ヒント： ファイル モニターを使用して、ジョブの複数インスタンスを同時に実行できます。詳細については、**コントロール ファイルによるフローのトリガー**（201ページ）を参照してください。
8. データベース リソースのプール サイズと、ステージ実行時インスタンスの設定値を増加させます。
9. サーバーを再起動します。
10. データフローを再度実行し、経過時間、CPU 使用率、メモリ使用率を記録します。
11. パフォーマンスが低下し始めるまで、データベース リソースのプール サイズとステージ実行時インスタンスの設定値を増加していきます。
12. ジオコーディング パフォーマンスをテストする場合は、単一国と複数国の入力を使用してこの手順を繰り返します。

デフォルトのソート パフォーマンス オプションの設定

大きなデータ セットのソートは、バッチ処理中に実行される操作のうちで最も時間がかかるものの 1 つになることがあります。そのため、適切なソート パフォーマンス オプションの設定は、ジョブのパフォーマンスに大きな影響を与える可能性があります。ソート パフォーマンス オプションによってメモリとディスクの使用率を制御すると、使用可能なメモリとディスクの容量を十分に活用できます。

ソート パフォーマンスの設定は、2 つの場所で行うことができます。1 つは Management Console です。ここでは、システムで使用されるデフォルトのソート パフォーマンス オプションを指定できます。もう 1 つは、ソートを実行するデータフロー ステージです。Sorter、Read from File、Write to File をはじめ、ソート オプションを含むその他すべてのステージには、ソート パフォーマンス オプションが含まれています。ソート パフォーマンス オプションをステージ内で指定する際には、デフォルトのソート パフォーマンス オプションをオーバーライドすることで、データフロー内の個々のステージに異なる設定を選択して適用できます。

ここでは、Spectrum™ Technology Platform サーバー上で実行されるジョブ用のデフォルトのソート パフォーマンス オプションの設定方法を示します。

1. Management Console を開きます。
2. **[フロー] > [デフォルト]** を選択します。
3. 以下の設定によって、ソート パフォーマンスを制御します。

メモリ内レコードの上限値
ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。

> **注：** 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。

一時ファイルの最大数
ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する一時ファイル数の増減がパフォーマンスに与える影響を確認する必要が

あります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。

```
(NumberOfRecords × 2) ÷ InMemoryRecordLimit =
NumberOfTempFiles
```

一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。

圧縮を有効にする 一時ファイルをディスクに書き込むときに圧縮します。

注： 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。次の式を一般的なガイドラインとして使用することで、妥当なソート パフォーマンスが得られます。*(InMemoryRecordLimit* × *MaxNumberOfTempFiles* ÷ 2) >= *TotalNumberOfRecords*

リモート コンポーネントのオプションの設定

リモート コンポーネントは、住所検証、ジオコーディング、税務管轄区域の割り当てなど特定の処理機能を実行する基盤となるエンジンです。一部のリモート コンポーネントは、パフォーマンスを最大化するよう設定できます。例えば、メモリ内にキャッシュする参照データの量や、入力データを参照データとマッチングする方法を制御するオプションを持つリモート コンポーネントがあります。

各リモート コンポーネントは独自の JVM 内に展開されます。つまり、JVM の構成は、サーバー自体とは独立して、リモート コンポーネントごとに行うことができ、メモリの柔軟な割り当てや、リモート コンポーネントの特性に基づいたパフォーマンスの調整が可能です。

リモート コンポーネントのオプションは、そのコンポーネントを使用するステージだけでなく、コンポーネントのすべてのインスタンスに影響を与えます。これは、設計時および実行時に変更可能なステージのオプションと異なる点です。

Universal Addressing モジュール コンポーネントの設定

米国住所に対しては、メモリにキャッシュする参照データを制御する、いくつかのオプションがあります。これらのオプションは、設定ファイル `server/modules/c1p/java.properties` で設定されます。

- DpvMemoryModel: メモリにキャッシュする DPV ファイルを制御します
- LacsLinkMemoryModel: メモリにキャッシュする LACSLink ファイルを制御します
- SuiteLinkMemoryModel: メモリにキャッシュする SuiteLink ファイルを制御します

これらのオプションのさまざまな値の詳細については、`java.properties` 設定ファイルを参照してください。

Enterprise Geocoding モジュール コンポーネントの設定

Enterprise Geocoding モジュールには、米国ジオコーディングのパフォーマンスに影響を与えるいくつかのオプションがあります。これらのオプションは、設定ファイル `server/modules/geostan/java.properties` にあります。その中で、特に興味深いオプションは次のとおりです。

- egm.us.multimatch.max.records: 返されるマッチ結果 (一致する可能性のあるもの) の最大数を指定します。この数値を小さくするとパフォーマンスは向上しますが、マッチ数は少なくなります。
- egm.us.multimatch.max.processing: 実行する検索の数を指定します。この数値を小さくするとパフォーマンスは向上しますが、マッチ数は少なくなります。
- FileMemoryLimit: 最初にメモリにロードする参照データの量を制御します。

Elasticsearch のヒープ サイズ設定

Elasticsearch は、Metadata Insights でデータ プロファイリングを実行する場合と、Enterprise Designer でマッチング ステージを実行する場合に使用される基盤の検索技術です。以下の場合は、Elasticsearch ヒープ サイズを大きくすることを検討してください。

- Metadata Insights で、複数のプロファイルを同時に実行するか、複数のテーブルを含むプロファイルを実行する場合
- 複数の検索インデックス クエリを並行して実行するフローがあり、各クエリが 1,000 件以上の候補を返す場合

Elasticsearch ヒープ サイズを大きくするには、次のファイルをテキスト エディタで開きます。

SpectrumLocation`\index\spectrum.vmargs`

`-Xmx` プロパティの値を大きくします。デフォルトのヒープ サイズは `-Xmx2048m` です。

個々のステージの最適化

マッチングの最適化

通常、マッチングはあらゆるデータ品質実装の中で最も時間のかかる操作の 1 つであり、できる限り効率的にマッチングを実行することが重要になります。マッチング結果とパフォーマンスは、

常にバランスが保たれています。ファイル内の各レコードをそれ以外のすべてのレコードと比較する場合、すべてのマッチを確実に特定できます。しかし、データ量が増大すると、このアプローチは持続できなくなります。例えば、100万レコードからなる入力ファイルがある場合、各レコードをそれ以外のすべてのレコードとマッチングすると、各マッチ ルールを評価するのにほぼ1兆回の比較が必要になります。

ファイル内の大部分のレコードがマッチしない場合、この問題を解決する一般的なアプローチは、マッチ キーを定義して、同じマッチ キーを持つレコードとのみ比較することです。適切なマッチ キーを定義することが、マッチング エンジンのパフォーマンスに影響を与える最も重要な変数です。適切なマッチ キーを定義するには、マッチング エンジンがレコードをどのように処理するか、および使用可能なオプションについて理解する必要があります。

デフォルトのマッチング方法では、マッチ キュー内のレコードをすべて比較して、最大数のマッチを特定します。そのため、この方法はしばしば最も時間のかかるマッチング方法になります。デフォルトのマッチング方法では、マッチ キュー内の先頭のレコードがサスペクト レコードになります。次のレコードが比較され、マッチした場合は重複として書き出されます。マッチしない場合はサスペクトとして追加され、次のレコードが2つのアクティブなサスペクトと比較されます。次のマッチ キューを考えてみましょう。

ユニーク ID	マッチ キー
1	123A
2	123A
3	123A
4	123A
5	123A
6	123A
7	123A
8	123A
9	123A
10	123A

まず、レコード 2 がレコード 1 と比較されます。マッチしない場合、レコード 2 はサスペクトとして追加されます。次に、レコード 3 がレコード 1 および 2 と比較されます。以降も同様です。マッチング レコードが存在しない場合、比較の総回数は 45 回になります。マッチするレコードがある場合、比較回数はそれより少なくります。サイズ N のマッチ キューの場合、比較の最大回数は N×(N-1)÷2 です。キュー サイズが小さい場合は目立ちませんが、キュー サイズが増大すると、影響が大きくなります。例えば、キュー サイズ 100 の場合の比較回数は 4,450 回で、キュー サイズが 500 の場合の比較回数は 124,750 回です。

適切なマッチ キーの定義

適切なマッチ キーを定義するには、以下の点を考慮してください。

- 覚えておくべき最も重要な点は、大部分のレコードはマッチしないということです。したがって、マッチする可能性のあるレコードのみを比較したいと考えます。
- 同じマッチ キーを持つレコードのみを比較します。
- パフォーマンスは重要な検討事項です。
  - マッチ キーにより、マッチ キューのサイズが決定されます。
  - 一定のレコード数では、マッチ キュー サイズが倍になると、実行時間も倍になります。
  - "厳格" なマッチ キーによって、パフォーマンスが向上します。"厳格" なマッチ キーは限定的なマッチ キーで、多数のフィールドからの多数の文字で構成されます。
  - "あいまい" なマッチ キーによって、マッチ数が増加する場合があります。"あいまい" なマッチ キーはそれほど限定的ではないマッチ キーで、少数のフィールドからの少数の文字で構成されます。

パフォーマンスとマッチ結果のバランスを見出す

パフォーマンスと結果のバランスをうまく保つには、マッチ ルールとデータの密度について検討してください。

- マッチ ルールに関して、以下の点を考慮してください。
  - 完全一致を必要とするフィールドをマッチ キーに含めることができます。
  - マッチ ルールに適したキーを作成してください。例えば、発音表記に関するマッチ ルールでは、発音表記に関するマッチ キーが適していると考えられます。
  - 多くの場合、マッチ キーはマッチングするすべてのフィールドの一部で構成されます。
  - 欠落しているデータが及ぼす影響に注意してください。
- データ密度に関して、以下のことを考慮してください。
  - 例えば、住所マッチングにおいて、すべてのレコードがある国のデータセットではなくある 1 つの町にある場合、マッチ キーはより厳格になる可能性があります。
  - 平均ではなく最大のマッチ キューを検討してください。Match Summary Report を精査して、最大マッチ キューを判断します。

- Transactional Match を使用する場合、同じ考慮事項が Candidate Finder の SELECT 文にも当てはまります。

Express マッチ キー

一般的なファイルでは、大部分の重複レコードが完全に、またはほぼ完全に一致します。Express マッチ キーを定義すると、マッチング エンジンが Experss マッチ キーを最初に比較して、2 つのレコードが重複かどうかを判断できます。これにより、フィールド レベルのマッチ ルールをすべて評価する必要はなくなるので、パフォーマンスを大幅に向上させることができます。

Intraflow Match の方法

デフォルトの Intraflow Match マッチング方法では、同じマッチ キーを持つすべてのレコードが比較されます。マッチ キューのサイズが N の場合、デフォルトの方法では N−1 から N×(N−1) までのいずれかの回数の比較が実行されます。すべてのレコードがマッチした場合、比較回数は N-1 です。レコードがまったくマッチしなかった場合、比較回数は N×(N−1) です。たいていの場合、比較回数はこの範囲の後半部分のどこかの値になります。

パフォーマンスが優先事項である場合は、デフォルトの方法ではなく、スライディング ウィンドウ マッチング方法の使用を検討してください。スライディング ウィンドウ マッチング方法では、各レコードを次の W レコード (W はウィンドウ サイズ) と比較します。ファイル サイズが N の場合、スライディング ウィンドウ方法で実行される比較回数は最大で N×W です。この方法によってパフォーマンスは向上しますが、一部のマッチが検出されなくなることがあります。

Candidate Finder の最適化

Candidate Finder は、Transactional Match で比較する候補レコードをデータベースから選択します。サスペクト レコードを Candidate Finder が返すすべての候補レコードと比較するため、Transactional Match のパフォーマンスは比較回数に比例します。

しかし、Candidate Finder のパフォーマンス向上のためにできることがいくつかあります。Candidate Finder のパフォーマンスを最大化するには、データベース管理者またはデータベース スキーマおよびインデックスについて豊富な知識を持つ開発者が、Candidate Finder の SQL SELECT 文をチューニングする必要があります。パフォーマンスに関する最も一般的な問題の 1 つは、テーブル全体のスキャンが必要となる JOIN を含んだクエリです。この場合は、インデックスの追加や、JOIN の代わりに UNION を使用することを検討してください。原則として、適任者が SQL クエリを確認し、最適化してください。

トランスフォームの最適化

Transformer ステージには、入力データに対して実行できる定義済みの操作セットがあります。これらの定義済みトランスフォームは既にコンパイルされているため、通常はカスタム トランスフォームより実行速度が速くなります。ただし、多数のトランスフォームを定義すると、カスタム トランスフォームの方がしばしば実行速度が速くなります。例えば、多数のフィールドをトリ

ムする場合、通常は、9つの個別のトリム トランスフォームよりも、次のカスタム トランスフォームの方が実行速度が速くなります。

```
data['AddressLine1'] = (data['AddressLine1'] != null) ?
data['AddressLine1'].trim() : null;
data['AddressLine2'] = (data['AddressLine2'] != null) ?
data['AddressLine2'].trim() : null;
data['AddressLine3'] = (data['AddressLine3'] != null) ?
data['AddressLine3'].trim() : null;
data['AddressLine4'] = (data['AddressLine4'] != null) ?
data['AddressLine4'].trim() : null;
data['City'] = (data['City'] != null) ? data['City'].trim() : null;
data['StateProvince'] = (data['StateProvince'] != null) ?
data['StateProvince'].trim() : null;
data['PostalCode'] = (data['PostalCode'] != null) ?
data['PostalCode'].trim() : null;
data['LastName'] = (data['LastName'] != null) ? data['LastName'].trim()
 : null;
data['FirstName'] = (data['FirstName'] != null) ?
data['FirstName'].trim() : null;
```

Write to DB の最適化

デフォルトでは、Write to DB ステージは各行をテーブルに挿入した後で確定されます。ただし、パフォーマンスを向上するためには、**[一括確定]** オプションを有効にします。このオプションが有効になっている場合は、指定された数のレコードを処理した後に確定が行われます。データベースによっては、これによって書き出しパフォーマンスを大幅に向上させることができます。

バッチ サイズの選択時には、以下の点を考慮してください。

- **Write To DB ステージへのデータ到着速度:** データの到着速度がデータベースの処理速度よりも遅い場合は、バッチ サイズを変更してもデータフローの全般的なパフォーマンスは改善しません。例えば、データフローで住所検証またはジオコーディングが行われている場合は、バッチ サイズを大きくしても効果が得られないことがあります。
- **ネットワーク トラフィック:** 低速のネットワークでは、バッチ サイズを中程度 (1,000 ～ 10,000) まで大きくすると、パフォーマンスが改善します。
- **データベースの負荷や処理速度:** データベースの処理能力が高い場合は、バッチ サイズを大きくすると、パフォーマンスが改善します。
- **複数の実行時インスタンス:** Write to DB の複数の実行時インスタンスを使用している場合、バッチ サイズを大きくすると大量のメモリが消費されるので、小または中程度のバッチ サイズ (100 ～ 10,000) を使用してください。
- **データベース ロールバック:** いずれかの文が失敗すると、バッチ全体がロールバックされます。バッチ サイズを大きくすると、ロールバックの実行時間が長くなります。

住所検証の最適化

Validate Address は、入力レコードが郵便番号でソートされている場合にパフォーマンスが最高になります。これは、参照データが郵便番号でソートされてメモリ内にロードされているためです。入力がソートされていると、ソートされていない入力に比べ、実行速度が数倍速くなることもあります。郵便番号フィールドにデータが入っていないレコードもあるため、次のソート順を推奨します。

1. Country (複数の国のレコードを処理する場合のみ必要)
2. PostalCode
3. StateProvince
4. City

ジオコーディングの最適化

ジオコーディング ステージでは、入力レコードが郵便番号でソートされている場合に最大のパフォーマンスが得られます。これは、参照データが郵便番号でソートされてメモリ内にロードされているためです。入力がソートされていると、ソートされていない入力に比べ、実行速度が数倍速くなることもあります。郵便番号フィールドにデータが入っていないレコードもあるため、次のソート順を推奨します。

1. PostalCode
2. StateProvince
3. City

異なるマッチ モードを試してみることによっても、ジオコーディング ステージを最適化できます。マッチ モードは、ジオコーディング結果が近似一致であるかどうかを、ジオコーディング ステージが判定する方法を制御します。マッチ モードを **緩和** に設定して、要件を満たす結果が得られるかどうか確認してみてください。一般的に **緩和** モードは、他のマッチ モードよりもパフォーマンスが向上します。

Geocode US Address の最適化

Geocode US Address ステージには、パフォーマンスに影響を与える複数のオプションがあります。以下のオプションがこのファイル内にあります。

`SpectrumLocation\server\modules\geostan\java.properties`

egm.us.multimatch.max.records	返されるマッチの最大数を指定します。この数値を小さくするとパフォーマンスは向上しますが、マッチ数は少なくなります。
egm.us.multimatch.max.processing	実行する検索の数を指定します。この数値を小さくするとパフォーマンスは向上しますが、マッチ数は少なくなります。
FileMemoryLimit	最初にメモリにロードする参照データの量を制御します。

JVM パフォーマンス チューニング

特定のハードウェアにおいてパフォーマンスを向上させることのできる JVM オプションがあります。これらは、高度なオプションであり、誤って使用すると予期せぬ動作を引き起こす恐れがあります (JVM がクラッシュする可能性もあります)。JVM のチューニングによってパフォーマンスを上げたいという場合は、テクニカル サポートにお問い合わせいただくことを推奨します。

- マルチ CPU のコンピュータでは、-XX:+UseParallelGC オプションを追加することによって、GC 処理を向上させることができます。
<codeph>-XX:+UseParallelGC</codeph><codeph>-XX:+UseParallelGC</codeph>`-XX:+UseParallelGC`
- 以下のオプションの追加によっても、パフォーマンスが向上することが確認されていますが、一部のハードウェア上では JVM がクラッシュすることもわかっています。
  - `-Xmn512m`
  - `-XX:+AggressiveOpts`

マイクロバッチ処理

マイクロバッチ処理は、サーバーに対する 1 回のリクエストに複数のレコードを含める手法です。各レコードを個別にリクエストする代わりに、リクエストの中に複数のレコードを含めることにより、サービスによってレコードの大規模コレクションを処理する場合のパフォーマンスを大幅に向上させることができます。Spectrum™ Technology Platform では、REST および SOAP Web サービスと、Client SDK に対するマイクロバッチ処理がサポートされています。

マイクロバッチ サイズ

1 回のリクエストに含めることのできるレコード数に制限はありませんが、一般的に 1 回のマイクロバッチで送信するレコード数を 50 ～ 100 にすると最良のパフォーマンスが得られます。さまざまなサイズのマイクロバッチをテストして、お使いの環境における最適なマイクロバッチ サイズを確認することをお勧めします。各入力レコードのレスポンスで、複数のレコードが得られる場合もあることに注意してください。例えば、マイクロバッチに 10 件の住所を含めて住所検証を実行する場合、各住所が 2 件の検証済み住所候補に一致したとすると、レスポンスでは 10 件ではなく 20 件のレコードが得られます。

Spectrum™ Technology Platform に対するリクエストで、マイクロバッチと複数スレッドの両方を使用する場合は注意が必要です。各スレッドのマイクロバッチ サイズが大きすぎると、システムは複数スレッドに対応できない可能性があります。

レコード ID の使用

マイクロバッチの各レコードに ID を割り当てると、リクエスト内のレコードとレスポンスで返されるレコードを対応付けることができて便利かもしれません。これを行うには、ユーザ フィールドを使用します。

パフォーマンス モニター

データフローのパフォーマンスをモニタリングすることにより、パフォーマンスのボトルネックを特定してパフォーマンスを調整できます。パフォーマンスは、管理ユーティリティと JMX コンソールの 2 つの方法でモニタリングできます。

管理ユーティリティは、パフォーマンス モニターを含む多数の管理機能へのアクセスが可能なコマンド ライン ツールです。パフォーマンス モニターを有効にすると、データフローが実行される度にパフォーマンス データがログ ファイルに書き込まれます。これには、データフロー内の各ステージのパフォーマンス データが含まれます。

JMX コンソールはブラウザベースのツールで、データフロー内の各ステージのパフォーマンス統計を記録するパフォーマンス モニタリング ツールを提供します。

管理ユーティリティによるパフォーマンスのモニタリング

管理ユーティリティは、パフォーマンス モニターを含む多数の管理機能へのアクセスが可能なコマンド ライン ツールです。パフォーマンス モニターを有効にすると、データフローが実行される度にパフォーマンス データがログ ファイルに書き込まれます。これには、データフロー内の各ステージのパフォーマンス データが含まれます。

1. 管理ユーティリティを開きます。
2. 次のコマンドを入力します。

   ```
   performancemonitor enabled set --e True --d DataflowName
   ```

   ここで *DataflowName* は、モニタリングするジョブまたはサービスの名前です。

これによって、指定したデータフローに対するパフォーマンス モニタリングが有効になります。データフローを実行すると、パフォーマンス情報がパフォーマンス ログに記録されます。

パフォーマンス ログ

パフォーマンス ログには、ジョブまたはサービスの実行にかかった時間の詳細が記録されます。ジョブまたはサービスの全般的なパフォーマンス情報に加えて、ジョブまたはサービス データフロー内の各ステージのパフォーマンス情報が含まれます。この情報を使用して、各ステージの実行時間と処理時間を確認し、ボトルネックを特定できます。実行時間と処理時間が大きく異なる場合、そのステージでは上流ステージからのデータを待つ時間が長いことを意味します。この場合は、上流ステージがデータフローにおけるボトルネックである可能性があります。シンクについては、実行時間と処理時間が大きく異なることが、必ずしもパフォーマンスの問題を意味するとは限らないことに注意してください。シンクは一般的に、データフローのその他の部分からの最初のレコードを待たなければならないためです。

ジョブまたはサービスのパフォーマンス モニタリングを有効にするには、管理ユーティリティでコマンド `performancemonitor enabled set` を使用します。

パフォーマンス ログは、お使いの Spectrum™ Technology Platform サーバーの次の場所にあります。

SpectrumDirectory\server\logs\performance.log

パフォーマンス ログは、モニタリング対象のジョブまたはサービスの各実行に対して 1 行で書き出されます。循環式のログで、最大 5 ファイルで構成されます。各ファイル サイズの上限は 10 MB です。この上限に達すると、最も古いパフォーマンス データが削除され、新しいパフォーマンス データが記録されます。

パフォーマンス ログの各エントリには、次の情報が含まれます。

> **注：** 以下では読みやすいように、改行とインデントを挿入しています。実際のログでは、エントリは 1 行で書き出されます。

```
Date Time [performance]
{
      "username":"UserName",
      "dataflowId":"DataflowName",
      "runMode":"BatchOrRealTime",
      "elapsedTime":Nanoseconds,
      "stageInfo":[
            {
                  "stageName":"Name",
                  "stageLabel":"Label",
                  "options":{
                        OptionsList
                  },
                  "recordsRead":Count,
                  "recordsWritten":Count,
                  "executionTime":Nanoseconds,
                  "processingTime":Nanoseconds
                  "readBlockingTime":Nanoseconds
```

```
                "writeBlockingTime":Nanoseconds
                "readBlockingPercent":Percentage
                "writeBlockingPercent":Percentage
            }
        ]
}
```

説明:

username

ジョブまたはサービスを実行したユーザ。

dataflowID

Enterprise Designer で定義されているサービスまたはジョブの名前。

runMode

ジョブとサービスのどちらのログ エントリであるかを示します。次のいずれかです。

Batch	ジョブに対するログ エントリです。
RealTime	サービスに対するログ エントリです。

elapsedTime

ジョブまたはサービスのリクエストを実行するのにかかった時間 (単位: ナノ秒)。

stageInfo

データフローの各ステージにおける実行時間の情報を一覧表示します。各ステージに対して、次の情報が表示されます。

stageName

ステージの永続的な名前。

stageLabel

ステージのユーザ定義の名前。ステージ ラベルは、Enterprise Designer のキャンバス上に表示されます。

options

実行時に指定されたオプションがあれば、その設定内容がここに表示されます。

recordsRead

ステージの全入力ポートを通してこのステージに引き渡されたレコードの総数。

recordsWritten

ステージの全出力ポートに書き出されたレコードの総数。

executiontime

ステージで最初のレコードが処理されてから最後のレコードが処理されるまでの時間。これには、データフローの他のステージからのデータを待つ間、ステージがアイドル状態だった時間が含まれます。

processingtime

ステージがレコードを実際に処理していた時間。データフローの他のステージからのデータ待ちでアイドル状態だった時間を除きます。

readBlockingTime

次のレコードの読み取り中にブロックする時間の長さ。読み取りのブロック時間を長くすると、前のプロセスにかかる時間がこのステージよりも長くなり、追加のチューニングが必要になる場合があります。

writeBlockingTime

次のレコードの書き込み中にブロックする時間の長さ。書き込みのブロック時間を長くすると、前のプロセスにかかる時間がこのステージよりも長くなり、追加のチューニングが必要になる場合があります。

readBlockingPercent

レコードの読み取り時にステージがブロックしていた合計実行時間の割合。

writeBlockingPercent

レコードの書き込み時にステージがブロックしていた合計実行時間の割合。

JMX コンソールによるパフォーマンスのモニタリング

JMX コンソールはブラウザベースのツールで、データフロー内の各ステージのパフォーマンス統計を記録するパフォーマンス モニタリング ツールを提供します。

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. 管理者アカウントでログインします。
3. "Domain: com.pb.spectrum.platform.performance" の下にある **com.pb.spectrum.platform.performance:service=PerformanceMonitorManager** をクリックします。
4. **[enable]** の横にある **[Invoke]** ボタンをクリックします。
5. **[Return to MBean View]** をクリックして、[PerformanceMonitorManager] 画面に戻ります。

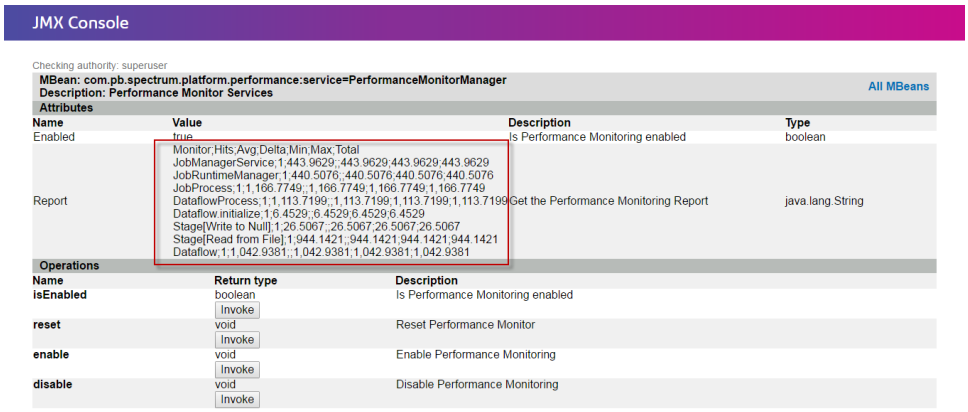
これでパフォーマンス モニタリングが有効になります。データフローを実行すると、パフォーマンス統計が [PerformanceMonitorManager] 画面の上部に表示されます。次のことに注意してください。

- 更新を確認するには、画面を再表示する必要があります。
- カウンタをリセットするには、**[reset]** の横にある **[Invoke]** ボタンをクリックします。

- Spectrum™ Technology Platform サーバーを停止すると、パフォーマンス モニタリングはオフになります。サーバーを再起動したら、パフォーマンス モニタリングを再度オンにする必要があります。

JMX パフォーマンス モニター統計

JMX コンソールのパフォーマンス モニター マネージャは、総実行時間、スループット、個々のステージの実行時間など、データフロー実行の様々な部分のパフォーマンスに関する統計を表示します。統計はセミコロン区切り形式でレポートされます。



ヒント： データをスプレッドシートに配置すると、見やすくなります。

先頭行は、次の列で構成されるヘッダ レコードです。

Monitor	パフォーマンスを測定する項目。
Hits	項目の実行回数。
Avg	項目が要求の処理に費やした時間の平均値 (単位: ミリ秒)。
Delta	この統計は未使用です。
Min	項目が要求の処理に費やした時間の最小値 (単位: ミリ秒)。
Max	項目が要求の処理に費やした時間の最大値 (単位: ミリ秒)。
Total	合計処理時間 (単位: ミリ秒)。

以下は、確認すべき最も重要な項目です。

Dataflow	データフローの総応答時間とスループット。
ServiceRuntimeManager.borrow.*DataflowName*	同時要求処理の応答時間 (単位: ミリ秒)。この応答時間は、データフローのプール サイズを変更することによって短縮できる可能性があります。

RemoteComponent.borrow.*RemoteComponentName*	データベース リソースのプール サイズによって直接影響を受けるデータベース リソースのパフォーマンス。この応答時間は、データフローのプール サイズを増加させると長くなる可能性があります。この数値が ½ ミリ秒も増加する場合は、データベース リソースのプール サイズを増加します。
Stage[*StageName*]	各ステージの応答時間。これにより、他のステージよりも処理にかなり長い時間がかかり、ボトルネックとなっているステージを特定できます。ボトルネックとなっているステージは、既存ステージの機能の実装方法を変更したり、そのステージの実行時インスタンスを増やしたりすることによって解消できる可能性があります。

9 - 監視

このセクションの構成

電子メール通知

メール サーバーの設定

Spectrum™ Technology Platform では、電子メール アラートを送信して重要なイベントを通知できます。電子メール通知は、データフローおよびプロセス フロー内の状況に従って送信したり、ライセンスやデータベースなどのアイテムが有効期限切れになりそうなタイミングで送信したりできます。

Spectrum™ Technology Platform にはメール サーバーが組み込まれていないので、電子メール通知を有効にするためには、外部の SMTP サーバーを使用できるように設定を行う必要があります。

1. Management Console を開きます。
2. **[システム] > [メール サーバー]** を選択します。
3. **[ホスト]** フィールドに、電子メール通知の送信に使用する SMTP サーバーのホスト名または IP アドレスを入力します。
4. **[ポート]** フィールドに、Spectrum™ Technology Platform サーバーと SMTP サーバーとの間のネットワーク通信で使用するポートの番号または範囲を入力します。

   デフォルトのポート番号は 25 です。
5. **[ユーザ名]** および **[パスワード]** フィールドに、Spectrum™ Technology Platform サーバーが SMTP サーバーでの認証に使用する資格情報を入力します。
6. **[送信者アドレス]** フィールドに、通知の送信元電子メール アドレスを入力します。
7. メール サーバーが正しく設定されていることを確認するためには、テスト用の電子メールを送信できます。テスト メールの送信先電子メール アドレスを **[テスト アドレス]** フィールドに入力し、**[テスト]** をクリックします。
8. **[保存]** をクリックします。

これで、Spectrum™ Technology Platform サーバーが SMTP サーバーに接続され、SMTP サーバーを使用して通知メールを送信できます。

例: メール サーバーの設定

`mail.example.com` という SMTP サーバーがあるとします。このメール サーバーを使用して、Spectrum™ Technology Platform サーバーから送信される電子メール通知を処理したいとします。この SMTP サーバー上には Spectrum123 というアカ

ウントを Example123 というパスワードで作成してあります。このアカウントの電子メール アドレスは `spectrum.notification@example.com` です。

以上の情報に従って通知を設定するには、各フィールドに次のように入力します。

ホスト	`mail.example.com`
送信者アドレス	`spectrum.notification@example.com`
ユーザ名	`Spectrum123`
パスワード	`Example123`

ライセンスと有効期限の通知の設定

この手順は、有効期限通知を送信するタイミングと通知メールの受取人の指定方法を示しています。

1. Management Console を開きます。
2. **[システム] > [ライセンスと有効期限]** を選択します。
3. **[通知の設定]** をクリックします。
4. **[通知を送る]** チェック ボックスをオンにします。
5. **[期限切れまでの日数]** フィールドに、ライセンス、ソフトウェア、またはデータの有効期限をその何日前に通知するかを、有効期限までの日数で指定します。この値がデフォルトです。**[システム] > [ライセンスと有効期限]** ページで、ライセンス項目ごとに異なる通知期間を指定できます。

   例えば、項目の有効期限が切れる 30 日前の通知を希望する場合は、 `30`と指定します。

6. **[受取人]** の下で、追加ボタン をクリックし、有効期限通知メールを受け取る電子メール アドレスを入力します。必要に応じて、複数の電子メール アドレスを入力できます。
7. **[保存]** をクリックします。

以上で、通知の受取人と、有効期限の何日前に通知メールが送信されるかが指定されました。電子メールの送信に使用するメール サーバーをまだ設定していない場合は、その設定を行う必要があります。メール サーバーの設定が済むまでは、通知は送信されません。

> **注 :** デフォルトでは、有効期限が近づいているすべての項目 (ライセンス、データベース、ソフトウェア コンポーネントなど) に対して有効期限通知が送信されます。**[システム] > [ライセンスと有効期限]**を選択して、特定の項目の有効期限通知を無効にすることができます。

バージョン情報の表示

1. Web ブラウザで次の URL を表示します。

   http://サーバー:*ポート*/managementconsole

   ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

2. **[システム] > [バージョン]**をクリックします。

有効期限通知を受信する項目の選択

通知を希望する項目が選択できるため、気になる項目についてのみ通知を受け取ることができます。

Spectrum™ Technology Platform は、ライセンス、データベース、ソフトウェア コンポーネントの有効期限が切れる前に電子メール通知を送付できます。これによって、期限切れによってビジネス プロセスに支障をきたすことなく、必要な措置を講じることができます。有効期限があるコンポーネントには、以下のようなものがあります。

- ライセンス
  - 電子メール通知は、トランザクションベースのライセンスに対しては提供されていません。ライセンスのトランザクション数が上限に近づいている場合は、Management Console のシステム ログにメッセージが表示されます。
  - Spectrum Spatial™ Manager に管理者としてログインした際に、ライセンス失効日までの期間が Management Console で設定されたライセンス有効期限の範囲内の場合、"`Spatial` ライセンスは `<n>` 日後に失効します。" という警告メッセージが表示されます。
- データベース (CASS 処理に使用される米国郵便データベースなど)
- 一部のソフトウェア コンポーネント (Universal Addressing モジュールにおいて米国住所の検証に用いられるエンジンなど)

ヒント： 有効期限がある項目を参照するには、Management Console を開き、**[システム] > [ライセンスと有効期限]** を選択します。

1. Management Console を開きます。
2. **[システム] > [ライセンスと有効期限]** を選択します。
3. ある項目に対する有効期限通知メールを受信するには、 **[通知を送る]** 列のチェックボックスをオンにします。

デフォルトよりも早く、または遅く通知を受け取りたい場合は、有効期限の何日前に通知を希望するかを、有効期限までの日数で指定します。

監査ログ

監査ログの表示

監査ログには、ユーザのアクティビティが記録されます。ユーザがシステム上でオブジェクトの作成や変更を行う際に発生するイベントと、ユーザが API や Web サービスを介してジョブを実行したりサービスにアクセスしたりする際に発生するイベントを記録します。監査ログに記録されるイベントの例としては、データフローの作成、データベース接続の変更、ジョブの実行などがあります。

1. Management Console を開きます。
2. **[システム] > [ログ]** を選択します。
3. **[監査ログ]**をクリックします。

   監査ログには、以下の情報が一覧表示されます。

列	説明
重大度	**エラー** エラーとは、システムの一部が使用不可能になる単発的な問題を指します。例えば、1 つのサービスが機能しなくなる問題によってエラーが生成されます。 **警告** 警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。 **情報** 通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。

列	説明
Date/Time	Spectrum™ Technology Platform サーバーのタイム ゾーンでのイベントの日付と時刻です。
ユーザ	アクションを実行したユーザ アカウントです。
ソース	イベントを生成したソフトウェア コンポーネントです。モジュールの名前または "Platform" となります。
イベント	発生したアクションです。プラットフォーム イベントは以下のように表示されます。インストールされているモジュールによっては、これらのイベントに加えて、その他のイベントも監査ログに表示されることがあります。 **作成** オブジェクトは作成され、サーバーに保存されました。 **バージョンの作成** Enterprise Designer で新しいバージョンのデータフローが作成されました。 **削除** オブジェクトはサーバーから削除されました。 **バージョンの削除** このデータフロー バージョンは削除されました。ほかのバージョンはまだ存在している可能性があります。 **エクスポーズ** データフローはエクスポーズされ、実行のために使用できる状態になりました。 **追加された項目** オブジェクトはサーバー上のフォルダに追加されました。 **移動された項目** オブジェクトはサーバー上の異なるフォルダに移動されました。 **名前の変更** オブジェクトの名前が変更され、そのオブジェクトは保存されました。 **アンエクスポーズ** データフローは実行のために使用できなくなりました。まだ Enterprise Designer での編集は行うことができます。 **更新** オブジェクトは変更され、保存されました。

列	説明
タイプ	システムの一部がイベントによって変更されました。タイプの例としては、データフロー タイプ (ジョブ、サービス、サブフロー、プロセス フロー) ファイル サーバーとアクセス制御設定があります。 状況によっては、**タイプ** 列に同じ名前のオブジェクトが何度か、異なる値で表示されることがあります。これは、1 つのユーザ アクションがシステム内で複数のイベントを生成する場合があるからです。例えば、Enterprise Designer でジョブを作成すると、監査ログにそのジョブの "作成" イベントが表示され、ジョブとして同じ名前を持つ FolderItem タイプの "追加された項目" イベントも表示されます。これは、ジョブが保存されてシステム上のあるフォルダ内に配置されたことを示しています。ジョブの保存とフォルダ内へのジョブの配置は、別々の 2 つのシステム イベントとして扱われます。
オブジェクト名	ログ エントリを生成した項目の名前です。例えば、データフローの名前などです。オブジェクト名には、ユーザによって定義されるもの (データフローの名前など) とシステムによって定義されるものがあります。

監査ログのアーカイブ

監査ログのイベントは、監査ログのサイズが大きくなりすぎないように、月に 1 度のペースでアーカイブされます。6 か月以上前のイベントは毎月、次のフォルダ内の圧縮ファイルに移行されます。

SpectrumDirectory\server\repository\store\archive

この圧縮ファイルは、必要に応じて、永続的なアーカイブのために別の場所に移すことができます。

監査ログによるシステム イベントの監視

セキュリティ管理者向けに、特定の期間にアクティビティをエクスポートして確認するためのオプションが 2 つ用意されています。こうした追跡は容易にロギングに組み込むことができます。

auditlog export 次のコマンドは、すべての監査ログ ファイルにアクティビティ ログを追加します。エクスポート対象ファイルは JSON または CSV 形式です。

```
auditlog export --s start_time --e end_time
```

時間の形式は `yyyyMMddHHmmss` です。特定の期間を指定しなかった場合、開始日時はデフォルトで現在の日付とコマンドを実行した時刻になります。

`auditlog info`

このコマンドにより、監査ログファイルにカウント情報ファイルが追加されます。エクスポート対象ファイルは JSON 形式です。

`auditlog info --s` *`start_time`* `--e` *`end_time`*

時間の形式は `yyyyMMddHHmmss` です。特定の期間を指定しなかった場合、開始日時はデフォルトで現在の日付とコマンドを実行した時刻になります。

システム ログ

システム イベントの表示

システム ログには、Spectrum™ Technology Platform サーバーの `spectrum-server.log` からのメッセージが表示されます。これらのメッセージには、サーバー操作や、API および Web サービスからサービスに対して作成されたリクエストに関する情報が含まれます。トラブルが発生し、考えられる原因についての情報を探す場合、システム ログを表示します。

Spectrum™ Technology Platform をクラスタで実行している場合、使用できるシステム ログはたまたま接続先となったノードで生成されたものです。特定のノードのシステム ログを表示するには、そのノードでファイル *`ServerLocation`*`\server\logs\spectrum-server.log` をテキスト エディタに開きます。

1. Management Console を開きます。
2. **[システム] > [ログ]** を選択します。
3. ダウンロード アイコンをクリックして システム ログ ファイルをダウンロードします。
4. ダウンロードしたファイルをテキスト エディタで開きます。

サービスのログ レベルの設定

デフォルトのログ レベルや、システム上の各サービスのログ レベルを指定することができます。ログ レベルを変更しても、変更前に作成されたログ エントリにはその変更は反映されません。

注： サービスに対して指定するログ レベルは、監査ログには影響しません。Management Console で表示できるイベント ログのログ レベルのみを制御します。現時点では、Web 版の Management Console ではイベント ログを表示できません。

1. Management Console を開きます。
2. **[システム] > [ログ]** を選択します。
3. **[システム デフォルト ログ レベル]** フィールドで、システムのサービスに対するデフォルトのイベント ログ レベルを選択します。

   無効 すべてのイベント ログを無効にします。

   致命的 最小限のログです。致命的なエラーのみがログに記録されます。致命的なエラーとは、システムを使用不可能にするエラーのことです。

   エラー エラーと致命的なエラーがログに記録されます。エラーとは、システムの一部が使用不可能になる単発的な問題を指します。例えば、1 つのサービスが機能しなくなる問題によってエラーが生成されます。

   警告 イベント警告、エラー、致命的なエラーがログに記録されます。警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。

   情報 抽象度の高いシステム情報がログに記録されます。これは、実稼働に適した最も詳細なログ レベルです。通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。

   デバッグ システムの問題のデバッグ時に適した、非常に詳細なログ レベルです。

   トレース プログラムの実行 (メソッドの開始と終了) をトレースする、最も詳細なログ レベルです。デバッグに使用する詳細なプログラム フロー情報を提供します。

   各ログ レベルは、1 つ上のログ レベルの内容を含みます。つまり、警告ログ レベルが選択されている場合、エラーと致命的なエラーも記録されます。情報ログ レベルが選択されている場合は、情報メッセージ、警告、エラー、および致命的なエラーが記録されます。

   注： 最も詳細なログ レベルを選択すると、システムのパフォーマンスに影響が生じる恐れがあります。したがって、必要なログ要件を満たす最小レベルの設定を選択する必要があります。

4. サービスごとに異なるログ レベルを指定することもできます。

フロー失敗の原因となったレコードのログ記録

フローが失敗した原因のトラブルシューティングを行う際には、失敗を引き起こしたレコードを調べることが有効です。フローの失敗を引き起こしたレコードは、Spectrum™ Technology Platform サーバー上のログ ファイルに書き出されます。ログ ファイルには、フロー内のステージの失敗につながったレコードが含まれます。入力レコードの形式誤りや、ライセンスの期限切れなど、他の原因でフローが失敗した場合のレコードは、このログに記録されません。

フローの失敗を引き起こしたレコードのログ記録を有効にするには

1. Web ブラウザを開いて に移動します。 `http://server:port/jmx-console`

   説明:

   server は、Spectrum™ Technology Platform サーバーの IP アドレスまたはホスト名です。

   port は、Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルト値は 8080 です。

2. 管理者アカウントでログインします。
3. 下方向にスクロールして次のエントリを探し、クリックします。

   com.pb.spectrum.platform.config:manager=LoggingConfigurationManager

4. 属性 **LogLastRecordReadOnError** に **true** を設定して **[set]** をクリックします。

これで、フローの失敗を引き起こしたレコードがサーバー上の新しいログ ファイルに記録されます。

`SpectrumDirectory/server/logs/error_records.log`

> **注：** このログには機密データが含まれる場合があるため、トラブルシューティングを終えたらログ ファイルを削除することを検討してください。

トランザクション上限に関する警告

トランザクションベースのライセンスでは、実行可能なトランザクション数に上限が設けられており、上限に達する前にライセンスを更新する必要があります。トランザクション上限までの残数がおよそ 10% になると、Management Console のイベント ログに警告メッセージが表示されます。例えば、Universal Addressing モジュールの Validate Address サービスに対してトランザ

クションを 100 万回実行可能なライセンスを保有している場合、トランザクションを 90 万回実行した時点で、次のようなメッセージがイベント ログに表示されるようになります。

```
WARN  [ValidateAddress] license for feature(s): UNC/USA/RealTime has
100,000 transactions remaining
```

上限に達すると、機能が無効になり、次のようなメッセージがイベント ログに表示されます。

```
ERROR [ValidateAddress] Usage limit exceeded for feature(s):
UNC/USA/RealTime
```

> **注：** システムは、数分ごとにトランザクション残数を計算し、必要に応じて警告メッセージをログに出力します。一度に最後の 10% 分のトランザクションを実行する、ジョブや大量のトランザクションが発生した場合は、警告メッセージが表示される前に、残りのトランザクションが使い切られてしまう可能性があります。このような場合には、警告メッセージが表示されることなく機能が無効になります。

ライセンスに対するトランザクション残数を確認するには、Management Console を開いて **[システム]**を展開し、**[ライセンスと有効期限]**をクリックしてから、**[ライセンス情報]** タブをクリックします。

ライセンスを更新する場合は、 Pitney Bowes 営業担当者までご連絡ください。

バージョン情報の表示

1. Web ブラウザで次の URL を表示します。

   http://サーバー:ポート/managementconsole

   ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

2. **[システム] > [バージョン]**をクリックします。

サーバー ステータスの表示

URL ベースのサーバー ステータス チェックにより、メモリ使用量や CPU 使用状況など、全体的なステータスをすばやく表示できます。この URL ベースのチェックでは、低オーバーヘッドかつパスワード保護のない形で、サーバーが稼働していて利用可能かどうかをチェックできます。

1. Web ブラウザで、以下の URL のいずれかを指定します。

   `https//`*`server`*`:`*`port`*`/dcg/status` または `http//`*`server`*`:`*`port`*`/dcg/status`

   説明:

   - *server* は Spectrum™ Technology Platform サーバーの名前または IP アドレスです。
   - *port* は、Spectrum™ Technology Platform が使用する HTTP または HTTPS ポートです。

     注 : デフォルトのサーバー ポートは、HTTP の場合は 8080、HTTPS の場合は 8443 です。

2. 現在のステータスを表示します。

住所	表示するサーバーの URL
cpuUsagePercentage	全体の CPU 割り当てのうち現在使用されている割合 (小数形式)
physicalMemoryUseagePercentage	現在使用されているメモリの割合 (小数形式)
heapMemoryUsagePercentage	現在使用されている実行時メモリの割合 (小数形式)
diskUsagePercentage	ディスク領域全体に対する使用領域の割合 (小数形式)

ライセンス情報の表示とエクスポート

ライセンスに関する情報を XML ファイルにエクスポートできます。この機能は、テクニカル サポートとともにライセンスの問題を解決する際に必要となる場合があります。

1. Web ブラウザで次の URL を表示します。

   http://*サーバー*:*ポート*/`managementconsole`

ここで *server* は、Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* は Spectrum™ Technology Platform が使用する HTTP ポートです。デフォルトでは、HTTP ポートが 8080、HTTPS ポートが 8443 になっています。

2. **[システム] > [ライセンスと有効期限]**をクリックします。
3. エクスポート アイコンをクリックします。

ライセンス情報は、拡張子が `.lic` である XML ファイルに保存されます。

10 - バックアップと復旧

このセクションの構成

スケジュール バックアップ情報

Spectrum™ Technology Platform サーバーをバックアップするには、サーバーの構成データベースのバックアップ コピーを作成する必要があります。構成データベースには、セキュリティ設定、データフロー、サービス オプション、データ リソース定義、スナップショットのほか、さまざまな構成情報が含まれています。サーバーのシステム障害やその他の災害でサーバーが稼働できなくなった場合、構成データベースのバックアップを使用してサーバー構成を別の Spectrum™ Technology Platform サーバーに復元できます。

重要： スケジュールに基づくバックアップは、Spectrum™ Technology Platform サーバーのアクティビティが非常に少ないかまったくない時間帯に実行されます。バックアップの実行中は、サービスの呼び出しがタイムアウトになったり、ジョブが正常に実行できなかったりする場合があります。

システム バックアップのスケジュール

この手順では、Spectrum™ Technology Platform を構成してバックアップを定期的に作成する方法を説明します。

Spectrum のシステム バックアップの詳細については次を参照してください。**バックアップのプロパティ**（270ページ）

> **注：** Spectrum™ Technology Platform をクラスタ環境で実行している場合は、クラスタ全体を停止して、すべてのノードを等しく設定した後に、クラスタを再起動する必要があります。

1. Spectrum™ Technology Platform サーバーを停止します。
2. 次のファイルをテキスト エディタで開きます。

   SpectrumDirectory\server\conf\spectrum-container.properties

3. 次のパラメータを指定します。

```
spectrum.backup.enabled=true
spectrum.backup.cron=Interval
```

   説明:

注： cron 設定の詳細については、**https://freeformatter.com/cron-expression-generator-quartz.html** にアクセスしてください。

Interval

バックアップ データベースを作成する間隔を指定する cron 式。cron 式は、スペースで区切られた 6 つの値で構成されます。オプションとして、第 7 の値を指定できます。

フィールド	有効な値	有効な特殊文字
秒	0 ～ 59	, - * /
分	0 ～ 59	, - * /
時間	0 ～ 23	, - * /
月の日数	1 ～ 31	, - * ? / L W
月	1 ～ 12 または JAN ～ DEC	, - * /
曜日	1 ～ 7 または SUN ～ SAT	, - * ? / L #
年 (オプション)	1970 ～ 2099	, - * /

例えば、この式は構成データベースを毎日午前 10 時にバックアップします。

```
spectrum.backup.cron=0 0 10 * * ?
```

この式は構成データベースを毎月 1 日の午前 2 時にバックアップします。

```
spectrum.backup.cron=0 0 2 1 * ?
```

特殊文字の使い方は次のとおりです。

*****

すべての値を指定します。例えば、月の日数フィールドで * を使用した場合は、その月の毎日を意味します。

?

特定の値を指定しません。この文字は他のフィールドと組み合わせて使います。例えば、月の初日にバックアップを実行し、その日が何曜日でもかまわない場合は、曜日フィールドに `?` を指定し、月の日数フィールドに `1` を指定します。

-

値の範囲を指定します。例えば、`SAT-SUN` は土曜日から日曜日を意味します。

,

複数の値を区切ります。例えば、月の日数フィールドの `15,30` は、その月の 15 日と 30 日を意味します。

/

間隔を指定します。例えば、時間フィールドの `0/3` は、バックアップを午前 0 時に実行し、それ以降 3 時間おきに実行することを意味します。

L

"最後 (last)" を指定します。これは使うフィールドによって意味が異なります。月の日数フィールドで使用した場合は、その月の最終日を意味します。曜日フィールドで単体で使用すると、土曜日を意味します。ただし、曜日フィールドで曜日と組み合わせて使用した場合は、その月の最後の曜日を意味します。例えば、`6L` はその月の最後の金曜日を意味します。

W

この値は月の日数フィールドで使用して、特定の日付に最も近い平日を指定します。例えば、`15W` はその月の 15 日に最も近い平日を意味します。

デスティネーション

バックアップ データベースの保存先とするディレクトリ。例を次に示します。

```
spectrum.backup.directory=\\\\exampleserver1\\Shared\\Backup
```

パスでバックスラッシュを指定するときは、エスケープ文字 \ を使用する必要があります。

> **注：** Spectrum™ Technology Platform をクラスタ環境で使用している場合は、一元化された場所をバックアップの保存先として指定する必要があります。クラスタ環境では、スケジュールに基づくバックアップがクラスタ内の不特定のノードで実行されるからです。一元化された場所を指定すると、クラスタから最新のバックアップを取得するのが容易になります。

4. Neo4j リポジトリをバックアップするには、次のプロパティを指定します。

```
spectrum.backup.repository.enabled=true
spectrum.backup.repository.databaseURL=URLOrHostMachine
spectrum.backup.repository.directory=Destination
```

5. Elasticsearch インデックス リポジトリをバックアップするには、次のプロパティを指定します。

```
spectrum.backup.index.enabled=true
spectrum.backup.index.directory=Destination
```

6. プロパティ ファイルを保存して閉じます。
7. Spectrum™ Technology Platform サーバーを開始します。
8. オプション: Spectrum™ Technology Platform をクラスタ環境で使用している場合は、クラスタ内の各ノードに対してこの手順を繰り返します。

   注： いずれのプロパティの値も、すべてのノードで等しく指定する必要があります。

9. モジュールの中には、Spectrum™ Technology Platform のスケジュールされたバックアップ処理ではバックアップされない追加データを保存するものがあります。このデータを手動でバックアップするか、このデータをバックアップする別のプロセスを作成する必要があります。
10. 固有のデータがあるモジュールがインストールされている場合は、それらのデータをバックアップします。

モジュール	バックアップ項目
Advanced Matching モジュール、Data Normalization モジュールおよび Universal Name モジュール	*SpectrumLocation*/server/modules の下にある以下のサブフォルダの内容をバックアップします。 • cdqdb • lucene • matcher • parser • searchindex • tables 。

モジュール	バックアップ項目
Data Hub モジュール	Relationship Analysis Client を起動し、**[管理]** をクリックします。バックアップするモデルを選択し、**[バックアップ]** をクリックします。 モデルのほかに、以下の 2 つのプロパティ ファイルもバックアップします。 • `server\modules\hub\hub.properties` • `server\modules\db\neo4j.properties`
Location Intelligence モジュール	名前付きリソース、データおよび構成ファイルをバックアップします。

バックアップのプロパティ

以下は、バックアップのプロパティとその関数のリストです。

一般バックアップのプロパティ

プロパティ	説明
`spectrum.backup.enabled`	Enable or disable all system backups
`spectrum.backup.cron`	Quartz cron configuration for scheduled backups: For more information on the cron configuration, visit **https://freeformatter.com/cron-expression-generator-quartz.html**.

Neo4j リポジトリ バックアップのプロパティ

プロパティ	説明
`spectrum.backup.repository.enabled`	Enable or disable the backup of the Neo4j repository, specifically: Overrides the general enabled flag (`spectrum.backup.enabled`)
`spectrum.backup.repository.databaseURL`	URL/Host of the machine where Neo4j repository runs; Do not modify unless Neo4j is running on a different machine than the server
`spectrum.backup.repository.directory`	Directory where Neo4j backup files are stored. By default, this location is `../server/backup/repository`. This location has changed from the previous default location,

プロパティ	説明
	which was `../server/app/repository/store/backup` in previous releases. In clustered setups, we suggest that you point this directory to a network share location.

Elasticsearch Index バックアップのプロパティ

プロパティ	説明
`spectrum.backup.index.enabled`	Enable or disable the backup of the Elasticsearch index repository, specifically: Overrides the general enabled flag (`spectrum.backup.enabled`)
`spectrum.backup.index.directory`	Directory where backup files are stored: By default, this location is `../server/backup/index`. IMPORTANT: In a clustered environment, this property **must** point to a network share location.

バックアップを手動で作成する

Spectrum™ Technology Platform サーバーをバックアップするには、サーバーの構成データベースのバックアップ コピーを作成する必要があります。構成データベースには、セキュリティ設定、データフロー、サービス オプション、データ リソース定義、スナップショットのほか、さまざまな構成情報が含まれています。サーバーのシステム障害やその他の災害でサーバーが稼働できなくなった場合、構成データベースのバックアップを使用してサーバー構成を別の Spectrum™ Technology Platform サーバーに復元できます。

Spectrum™ Technology Platform 構成データベースを手動で作成するには、管理ユーティリティの `server backup` コマンドを使用します。詳細については、「**server backup**（526ページ）」を参照してください。

これに加えて、一部のモジュールには、管理ユーティリティのバックアップ処理に含まれないデータがあります。次のデータを個別にバックアップする必要があります。

モジュール	バックアップ項目
Advanced Matching モジュール、Data Normalization モジュールおよび Universal Name モジュール	*SpectrumLocation*/server/modules の下にある以下のサブフォルダの内容をバックアップします。 • cdqdb • lucene • matcher • parser • searchindex • tables 。
Data Hub モジュール	Relationship Analysis Client を起動し、**[管理]** をクリックします。バックアップするモデルを選択し、**[バックアップ]** をクリックします。 モデルのほかに、以下の2つのプロパティ ファイルもバックアップします。 • server\modules\hub\hub.properties • server\modules\db\neo4j.properties
Location Intelligence モジュール	名前付きリソース、データおよび構成ファイルをバックアップします。

サーバーの復元

深刻なシステム障害や災害などでサーバーが稼働しなくなった場合は、構成データベースのバックアップを使用してサーバーを復元できます。バックアップを用意するには、手動で作成しておくか、Spectrum™ Technology Platform を構成して定期的にバックアップが作成されるようにする必要があります。デフォルトで、Spectrum™ Technology Platform は構成データベースのバックアップを作成しません。

> **注：** この手順は、Spectrum™ Technology Platform サーバーが 1 台のケースを想定しています。Spectrum™ Technology Platform サーバーがクラスタ化されているシステムで 1 つのノードだけを復元する場合は、新しいサーバーをインストールし、それをノードに追加してください。クラスタの構成が新しいノードに自動的に適用されるため、ノードを復元したのと同じ結果が得られます。クラスタ環境でバックアップからの復元を行う必要がある唯一のシナリオは、クラスタに含まれるすべてのノードが完全に失われた場合です。

1. 新しい Spectrum™ Technology Platform サーバーをインストールします。詳細については、『*インストール ガイド*』を参照してください。
2. サーバーが実行している場合は、サーバーを停止します。
3. バックアップの zip ファイルを取得し、以下の場所に解凍して既存のファイルを上書きします。

   SpectrumDirectory\repository\data\databases

   これによって、既存の graph.db フォルダが置き換えられます。
4. インストールしたすべてのモジュールに対し、モジュール固有のデータを復元します。

モジュール	バックアップ項目
Advanced Matching モジュール、Data Normalization モジュールおよび Universal Name モジュール	*SpectrumDirectory*/server/modules の下にある以下のサブフォルダの内容を復元します。 • cdqdb • lucene • matcher • parser • searchindex • tables
Data Hub モジュール	モデルを復元します。 モデルのほかに、以下の 2 つのプロパティ ファイルも復元します。 • server\modules\hub\hub.properties • server\modules\db\neo4j.properties
Spatial モジュール	名前付きリソース、データ、および構成ファイルを復元します。

5. サーバーを開始します。
6. サーバーが完全に起動するまで待ちます。
7. サーバーを停止します。
8. プラットフォームとインストール済みのモジュールに対するすべてのアップデートを適用します。アップデートの一覧については、Pitney Bowes サポート Web サイトの「**アップデートの概要**」を参照してください。

11 - 設定

このセクションの構成

Business Steward の設定

はじめに

Business Steward の設定の概要

Business Steward の設定は、書き込み権限を持つユーザに次の機能を提供します。

- **検索** — 特定のデータセットを修正に使用する例外レコード修正機能を提供します。
- **ドメイン** — 評価するデータの種類を指定できます。
- **メトリクス** — データを測定する方法を指定できます。
- **通知** — 特定のドメインまたはメトリクスに関連付けられた例外が一定の件数に達した時点で 1 つ以上の電子メール アドレスにメッセージを送信できます。
- **データ品質レポート** — 合否条件と KPI の追跡に関する設定を指定します。
- **検索ツール サービス** — Business Steward Portal 例外エディタの検索ツールに関する設定を指定します。
- **オプション** — 監査ログと進行状況の追跡に関する設定を指定します。

Business Steward の設定へのアクセス

Business Steward の設定にアクセスするには

1. Web ブラウザで次の URL を表示します。

   http://*server*:*port*/managementconsole

   ここで *server* は、 Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* はSpectrum™ Technology Platformが使用する HTTP ポートです。デフォルトの HTTP ポートは 8080 です。

2. 有効なユーザ名とパスワードを入力します。
3. **[リソース]** ボタンをクリックします。
4. **[Business Steward の設定]** を選択します。

検索

例外エディタでのレコードの更新時に、特定のフィールドに対する値のリストを選択肢に指定できるようにする検索ツールです。この機能は、変更したいフィールドにデータを含むレコードが複数ある場合に、特に便利です。

例えば、銀行データを含む例外レコードがいくつかあるとします。このようなデータには、例えば、そのレコードに関連付けられた口座の種類を表すコード (1 = 当座預金、2 = 貯蓄、3 = 金融市場など) で構成されるフィールドを含めることができます。これらの例外レコードには住所の [国] フィールドに国名ではなく ISO コードが含まれており、住所の検証が不可能になっていると仮定します。これを修正するには、ISO コードと国名のペアを提供する検索を作成し、例外エディターで修正を実行します。例外エディターで ISO コードをリストから選択すると、その ISO コードに対応する国名がフィールドに設定されます。

このツールを使うもう 1 つの利点は、修正に使えるオプションを制限できることです。修正しようとしてかえってエラーを招く可能性を減らすことができます。先程と同様に、国名の誤りまたは欠落があると仮定します。例外レコードごとに国名を手動で入力するよう要求するのではなく、国名のリストを提供する検索を作成することで、国名のスペルの間違いを排除し、修正後に正しく検証できるレコードになる可能性を高めることができます。

検索プロセスの概要

検索プロセスは、3 つのステップから構成されます。例外を調査し、それらに共通して見られる問題点 ([国] フィールドに無効なデータがある、など) を特定した後で、検索を実行します。

- 誤ったデータを上書きする正確なデータの値ペアまたは値 / ラベル ペアを使って、検索を作成します。
- 例外レコードを生成しているデータフローで Write Exceptions ステージを使って、作成した検索に問題のフィールドを指定し、データフローを再実行します。
- 問題となっているフィールドの誤ったデータを検索から提供される正しいデータで上書きし、Business Steward Portal の例外エディターで例外レコードを修正します。

検索の作成

検索は値のペアまたは値とラベルのペアで構成され、データフロー内で例外レコードが生成される原因となっているデータを置き換える値を指定できます。値は問題のデータと置き換えるために使用し、ラベルは Exception Editor でレコードを修正するために検索テーブルを使うときにリストから選択する項目として使用します。

> **注：** 検索テーブルに値だけを含めた場合は、その値がラベルとしても使用されます。

検索の各項目を設定するには、情報を手動で入力するか、外部ソースからコピーして **[多数を追加]** ダイアログ ボックスに貼り付けます。情報がカンマ、タブ、またはセミコロンを区切り文字とする 1 つまたは 2 つの列で記述されている限り、スプレッドシートやテキスト ファイルに限らずほとんどあらゆる形式のファイルを外部ソースに使用できます。

> **注：** **[多数を追加]** 機能を使ってから **[保存]** をクリックすると、その検索に設定した既存の値ペアまたは値 / ラベル ペアは削除されます。ただし、[多数を追加] 機能を使った後で、値ペアや値 / ラベル ペアを手動で追加できます。

1. **[検索を追加]** ボタンをクリックします。
2. 新しい検索の名前をテキスト ボックスに入力します。
3. 値 / ラベル ペアを追加します。

   値 / ラベル ペアを手動で追加するには

   1. **[検索値を追加]** ボタンをクリックします。
   2. 検索に使う値またはラベル (またはその両方) を入力します。

   [多数を追加] 機能を使用して値 / ラベル ペアを追加するには

   1. **[多数を追加]** ボタンをクリックして、ダイアログ ボックスを開きます。
   2. 使用するデータに適切な列と区切り文字を選択します。データを Microsoft Excel から貼り付ける場合は、区切り文字としてタブを使用します。区切り文字の選択を誤ると、行全体が 1 つの値またはラベルとしてインポートされ、最初の列として設定されます。
   3. すべてのコンテンツの値、区切り文字、またはラベルを入力するか、別のアプリケーションから貼り付けます。

   すべての値または値 / ラベル ペアを追加した後、**[値]** 列または **[ラベル]** 列を昇順または降順に並べ替えることができます。

   > **注：** いったん並べ替えたリストは、並べ替える前の状態に戻せません (逆順に並べ替えたり、他の列で並べ替えることは可能です)。

4. 別の値 / ラベル ペアを追加するには、ステップ **3**（277ページ） を繰り返します。
5. **[保存]** をクリックします。

検索の割り当て

検索を作成した後で、データフローの Write Exceptions ステージで問題となっているデータがあるフィールドに割り当てる必要があります。

1. Enterprise Designer で、例外レコードを生成しているデータフローを開きます。
2. Write Exceptions ステージを開きます。
3. 問題となっているデータがあるフィールドの **[検索名]** 列で、ドロップダウン リストから正しいデータを含む検索を選択し、**[OK]** をクリックします。

4. データフローを保存して再実行します。

レコードの作成

検索を作成し、データフローのフィールドに割り当てた後で、Business Steward Portal で例外レコードを修正する必要があります。

1. 例外エディターで、例外レコードを生成しているデータフローを選択します。
2. 問題となっている最初のレコードで、検索を割り当てたフィールドをクリックします。
3. そのフィールドのドロップダウン ボタンをクリックし、目的のレコードのラベルを選択します。

   要確認: このラベルは必ずしも値と同じではありません。例えば、フィールドの値を"California"にしたい場合に "CA" と記されたラベルをクリックするようなケースがあります。

4. 問題となっているレコードのそれぞれについて、ステップ **3**（278ページ）を繰り返します。
5. 変更した例外を保存します。

検索の変更または削除

1. [検索] 画面を開きます。
2. 目的の検索の横にあるボックスをチェックします。
3. 検索を変更または削除します。

オプション	説明
検索を変更するには	**[検索を編集]** ボタンをクリックし、必要に応じて検索を変更し、**[保存]** をクリックします。
検索を削除するには	**[検索を削除]** ボタンをクリックします。

ドメイン

ドメインは、評価対象となるデータの種類を指定します。データに発生した例外の種類を表示するレポート目的で使用されます。例えば、条件によって住所検証の成功/失敗を評価する場合、データ ドメインは "住所" です。条件によってジオコーディング操作の成功/失敗を評価する場合、データ ドメインは "空間" になります。

> **注：** ここで確立するドメインは、Business Steward 構成と Exception Monitor ステージの両方でデフォルト オプションとして使用されます。

以下に示す定義済みのドメインから 1 つを選択するか、**[項目を追加]** ボタンをクリックして必要なフィールドに値を入力し、独自のドメインを指定できます。また、ドメインを選択し、**[項目を編集]** ボタンをクリックして必要な変更を加えることでドメインを編集できます。検索データを **[フィルタ]** フィールドに入力すれば、表示されるドメインの一覧をフィルタすることもできます。結果は動的に更新されます。

- Account— セールス アカウントに関連付けられた企業または組織の名前をチェックします。
- Address— 完全な郵送先住所や郵便番号などの住所データをチェックします。
- Asset—物理的資産、不動産、人材、その他のアセットなど、企業の資産に関するデータをチェックします。
- Date— 日付データをチェックします。
- Email— 電子メール データをチェックします。
- Financial— 通貨、証券などに関連するデータをチェックします。
- Name— 名や姓などの個人名データをチェックします。
- Phone— 電話番号データをチェックします。
- Product— 材料、部品、商品などに関するデータをチェックします。
- Spatial— 洪水発生地帯、海岸線、家屋、販売区域など定義済みの地勢を表す点、ポリゴン、または線データをチェックします。
- SSN— 米国社会保障番号データをチェックします。
- Uncategorized— この条件を分類しない場合は、このオプションを選択します。

メトリクス

メトリクスは、データを測定する方法を指定します。データに発生した例外の種類を表示するレポート目的で使用されます。例えば、レコードの完全性 (すべての住所に郵便番号が含まれているかなど) を評価するための条件を設計する場合、データ品質指標として "完全性" を指定することができます。

> **注：** ここで確立するメトリクスは、Business Steward 構成と Exception Monitor ステージの両方でデフォルト オプションとして使用されます。

以下に示す定義済みのメトリクスから 1 つを選択するか、**[項目を追加]** ボタンをクリックして必要なフィールドに値を入力し、独自のメトリクスを指定できます。また、メトリクスを選択し、**[項目を編集]** ボタンをクリックして必要な変更を加えることでメトリクスを編集できます。検索データを **[フィルタ]** フィールドに入力すれば、表示されるメトリクスの一覧をフィルタすることもできます。結果は動的に更新されます。

- Accuracy— データを信頼できるソースに対して検証できるかどうかを評価します。例えば、郵政当局のデータを使用して住所を検証することはできません。正確ではないので、例外と見なされます。

- Completeness— データに不可欠な属性が欠落しているかどうかを評価します。例えば、郵便番号が欠落している住所や、連絡先が欠落しているアカウントなどです。
- Consistency— データが複数のシステム間で一貫しているかどうかを評価します。例えば、顧客データ システムでは M と F という性別コードが使用されているが、処理しているデータの性別コードが 0 と 1 の場合、データには一貫性の問題があると見なされます。
- Interpretability— データが他のシステムによって解釈可能なデータ構造に正しくパースされているかどうかを評価します。例えば、社会保障番号に含めることができるのは数値データだけです。このデータに xxx-xx-xxxx などの文字が含まれる場合、データには解釈可能性の問題があると見なされます。
- Recency— データが最新かどうかを評価します。例えば、ある個人が引っ越したが、システム内の住所が古いままの場合、データには最新の問題があると見なされます。
- Uncategorized— この条件を分類しない場合は、このオプションを選択します。
- Uniqueness— 重複データがあるかどうかを評価します。データフローが重複データを統合できなかった場合、そのレコードは例外と見なされます。

通知

通知機能を使うと、特定のドメインまたはメトリクスに関連付けられた例外が一定の件数に達した時点で 1 つ以上の電子メール アドレスにメッセージを送信できます。送信される電子メールには Business Steward Portal の例外エディター内の失敗レコードへのリンクが含まれ、例外エディターでは正しいデータを手動で入力できます。特定の電子メール アドレスでの通知の受信を停止するには、そのアドレスを、[ドメインを編集] ページの [通知送信先] 行にある受取人一覧から削除します。

> **注：** Business Steward 構成内で通知を正しく使用できるようにするには、Management Console で通知を設定する必要があります。通知の設定については、『管理ガイド』を参照してください。

1. Business Steward 構成で、**[ドメイン]** ページまたは **[メトリック]** ページを開きます。
2. 通知対象に追加するドメインまたはメトリックを選択し、**[項目を編集]** ボタンをクリックします。
3. 通知の送信先として、(Management Console で設定された) ドロップダウン リストからユーザ名を選択するか、新しい電子メール アドレスを入力します。
4. 通知をトリガーする例外レコードの数を選択します。
5. 通知の件名に使うテキストを入力します。
6. 通知の本文に使うメッセージを入力します。

   例外に関する重要な情報を差し替えるため、メッセージ中で変数を使うことができます。使用できるのは次の変数です。

- ${jobID} — 例外レコードを生成したジョブの ID 番号。
- ${jobName} — 例外レコードを生成したジョブの名前。
- ${userName} — 例外レコードを生成したジョブのユーザ名。
- ${stageLabel} — 例外レコードを生成したデータフロー ステージの名前。
- ${link} — 特定のデータフローのレコードが表示される、Business Steward Portal のエディター ページへのリンク。

7. リマインダー メッセージを送信する場合は **[リマインダーを送信]** ボックスをチェックし、リマインダーを送信するまでの経過日数を選択します。
8. ステップ **3**（280ページ）で指定した電子メール アドレス以外のアドレスに通知を送信するには、ドロップダウン リストから (Management Console で設定された) ユーザ名を選択するか、**[リマインダー送信先]** フィールドに電子メール アドレスを入力します。
9. リマインダー通知の件名に使うテキストを入力します。
10. リマインダー通知の本文に使うメッセージを入力します。

    リマインダーには次の変数も使用できます。

    - ${Count} — 特定のデータフローまたはステージで生成された未解決の例外の数。

11. 例外が解決されるまでリマインダーを毎日送信する場合は、**[リマインダーを毎日送信]** ボックスをチェックします。

データ品質レポート

1. **[データ品質レポート]** をクリックすると、Exception Monitor ステージで合否条件を追跡するようになります。このオプションをオフにすると、Business Steward Portal の [データ品質] 画面にデータが表示されなくなります。同様に、全てのデータフローの Exception Monitor ステージのにおいて、[レポートのみ] フィールドが無効になります。
2. **[保持]** ドロップダウンで、このデータを保持する期間を月数で指定します。

主要パフォーマンス指標の設定

[データ品質レポート] タブの **[KPI の設定]** セクションでは、主要パフォーマンス指標 (KPI) をデータに指定し、それらの KPI が特定の条件を満たした場合の通知を割り当てることができます。

1. **[KPI の追加]** をクリックします。
2. 主要パフォーマンス指標の **[名前]** を入力します。この名前は Spectrum™ Technology Platform サーバー上で一意である必要があります。
3. この主要パフォーマンス指標で使用するデータ品質の **[指標]** を 1 つ選択します。ここで指標を選択しない場合、この主要パフォーマンス指標は、すべての**指標**に関連付けられます。

4. この主要パフォーマンス指標で使用する **[データフロー]** の名前を選択します。ここで名前を選択しない場合、この主要パフォーマンス指標は、すべての Business Steward Module データフローに関連付けられます。
5. この主要パフォーマンス指標で使用する **[ステージ ラベル]** を選択します。ここでステージ ラベルを選択しない場合、この主要パフォーマンス指標は、データフロー内のすべての Business Steward Module ステージに関連付けられます。
6. この主要パフォーマンス指標で使用するデータの **[ドメイン]** を選択します。ここでデータ ドメインを選択しない場合、この主要パフォーマンス指標は、すべてのドメインに関連付けられます。ここでドメインを選択すると、[条件] フィールドが無効になることに注意してください。
7. 主要パフォーマンス指標で使用する **[条件]** を選択します。ここで条件を選択しない場合、この主要パフォーマンス指標は、デフォルトで "すべて" に設定されます。条件を選択するには、最初に [ドメイン] フィールドで "すべて" を選択する必要があることに注意してください。条件が選択されていると、[ドメイン] フィールドは無効になります。
8. **[KPI 期間]** を選択して、Business Steward モジュールでデータを監視して通知を送信する期間を指定します。例えば、"1" と "毎月" を選択した場合、基準とするしきい値または分散から前月比で例外の割合が増加していると、KPI 通知が送信されます。
9. **[相違]** または **[しきい値]** の割合を指定します。分散値は、前の期間からの例外レコードの失敗の増加率を表します。しきい値は、通知を送信するときの失敗の割合を表します。この値は 1 以上である必要があります。
10. これらの条件を満たした場合に送信する必要がある通知の **[受取人]** の電子メール アドレスをリストから選択するか、入力します。可能な場合、電子メール アドレスを入力し始めると、このフィールドは自動的に設定されます。複数のアドレスを設定する場合に、各アドレスをカンマ、セミコロン、またはその他の句読文字で区切る必要はありません。
11. 通知電子メールで使用する **[件名]** を入力します。
12. これらの条件を満たしたときに通知で送信する **[メッセージ]** を入力します。
13. **[OK]** をクリックします。他の既存の KPI の中に新しい KPI が表示されます。KPI は、データを含む任意の列でソートできます。
14. **[保存]** をクリックします。

KPI の変更や削除を行うには、**[選択した KPI を編集]** または **[選択した KPI を削除]** をクリックします。

検索ツール サービス

1. Buisness Steward Portal 例外エディターで使用する検索ツール サービスを選択します。使用可能なサービスはユーザの権限によって異なり、Spectrum Technology Platform で使用する

ためのライセンスを得ているモジュールとサービスによって選び出されます。**[フィルタ]** を使ってフィルタ条件を指定すると、サービスのリストが絞り込まれます。

2. **[プレミアム]** をクリックすると、これらのサービス (Dun & Bradstreet サービスなど) の使用に別途料金が課されることをユーザに示すことができます。

オプション

1. **[監査例外イベント]** をクリックすると、例外レコードの作成、読み取り、更新、または削除がいつ行われたかのログが Business Steward モジュールによって保持されます。
2. **[進捗の追跡]** をクリックすると、Business Steward Portal で行った例外レコードの承認が追跡されます。このオプションをオフにすると、Business Steward Portal のダッシュボードに進捗グラフが表示されなくなります。

Data Hub の設定

Data Hub の設定では、表示および変更権限を持つ管理ユーザによる監査ログとモデル バックアップの優先設定が可能です。

Data Hub の設定へのアクセス

[Data Hub の設定] ページは、Management Console アプリケーションにあります。

1. アプリケーション メニュー で、**[Management Console]** をクリックします。
2. バナーで、**[リソース] > [Data Hub の設定]** の順にクリックします。

Data Hub の設定

Management Console アプリケーションに [Data Hub の設定] を表示するには、**[リソース] > [Data Hub の設定]** の順にクリックします。

監査モデルのイベント このチェックボックスをオンにすると、モデルの作成、変更、または削除の際のログが Data Hub モジュールで保持されます。

メタデータ イベントを含める このチェックボックスをオンにすると、メタデータ アクティビティが監査ログに記録されます。このアクティビティを含めるには、監査モデル イベントは有効化する必要があります。

読み込みイベントを含める このチェックボックスをオンにすると、モデルが表示されたときのデータが監査ログに記録されます。なお、このデータを含めると、監査ログのストレージの制限に大きく影響するためご注意ください。このアクティビティを含めるには、監査モデル イベントは有効化する必要があります。

履歴を追跡 このチェックボックスをオンにすると、Relationship Analysis Client で履歴機能が有効になります。有効にすると、エンティティおよび関連性に対する変更内容を表示できます。

デフォルトのバックアップ ディレクトリをオーバーライド このチェックボックスをオンにすると、Data Hub モデルのバックアップを保存する既存フォルダのパスを指定できます。各バックアップは、この場所にある `model.ModelName` という名前のフォルダに保存されます。この設定は、Data Hub モデルがデフォルトでバックアップされる場所 () より優先されます。Spectrum™ Technology Platform サーバーのログオン アカウントには、指定されたフォルダに対する書き込み権限が付与されている必要があります。

> **注：** デフォルトのバックアップ ディレクトリの場所 (`SpectrumFolder/server/modules/hub/db/backups`) は `SpectrumFolder/server/modules/hub/hub.properties` ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

ページ キャッシュをバックアップ (MB) モデルのバックアップ中に使用されるメモリ サイズをメガバイト (MB) 単位で増減できます。この値が大きくなると、パフォーマンスが向上しますが、より多くの RAM が使用されます。設定可能な値は、8 ～ 8192 です。

スケジュールバックアップ このチェックボックスをオンにすると、モデルのバックアップが有効になり、バックアップの頻度と時間を指定できます。**[増分]** をオンにすると、システムがトランザクション ログを使用して前回のバックアップ以降に変更された内容を判断し、その変更内容を既存のバックアップに追加します。

トランザクションをログに記録 このチェックボックスをオンにすると、トランザクションがモデルへのデータのコミットを開始したときと、トランザクションが完了したときのメッセージが wrapper.log に記録されます。

例外時に入力データをログに記録 このチェックボックスをオンにすると、例外が発生したときに処理される入力レコードからのデータが wrapper.log に記録されます。社会保障番号、ID、口座番号などの機密情報に対してセキュリティ上の問題が発生する可能性を排除するには、このボックスをオフのままにします。

仮想クエリをログに記録	このチェックボックスをオンにすると、仮想クエリをログに記録されます。
クエリ タイムアウト	サーバーがクエリが完了するまで待機する時間を秒単位で指定するには、このチェックボックスをオンにします。設定可能な値は、1 〜 100000 です。

12 - 管理ユーティリティ

このセクションの構成

管理ユーティリティを使用する前に

管理ユーティリティでは、管理機能をコマンドラインから実行できます。コマンドの実行は対話的に行うことも、スクリプトで行うこともできます。一部の管理機能は、管理ユーティリティでは利用できません。以下の機能は、Management Console を使用して実行できます。

> **注：** The Administration Utility requires Java 8 or later. Verify that Java 8 is in the system's path before running the Administration Utility.

1. Spectrum™ Technology Platform の **[ホーム]** ページで **[プラットフォーム クライアント ツール]** をクリックします。
2. **[コマンド ライン]** をクリックします。
3. **[管理ユーティリティ]** で **[ダウンロード]** をクリックして、管理ユーティリティを使用するコンピュータに zip ファイルをダウンロードします。
4. zip ファイルの内容を解凍します。
5. コマンドライン インターフェイスを起動するには、次のいずれかの操作を実行します。
   - サーバーが Unix または Linux システムで実行されている場合は、`cli.sh` を実行します。
   - サーバーが Windows システムで実行されている場合は、`cli.cmd` を実行します。

     > **注：** 必要に応じて、Java のインストール パスが正しく使用されるように `.sh` または `.cmd` ファイルを変更します。

6. 次のコマンドを入力して Spectrum™ Technology Platform サーバーに接続します。

   `connect` --h *servername*:*port* --u *username* --p *password* --s *SSLTrueOrFalse*

   例を次に示します。

   `connect` --h myserver:8080 --u admin --p myPassword1--s true

7. 接続が完了すると、コマンドを実行できます。コマンドのヒントを紹介します。
   - 使用できるコマンドの一覧を表示するには、`help` と入力するか、Tab キーを押します。
   - コマンドを自動補完するには、最初の数文字を入力してから Tab キーを押します。例えば、`us` と入力してから Tab キーを押すと、自動補完されて `user` コマンドになります。もう一度 Tab キーを押すと、`user` コマンドの全一覧が表示されます。
   - 空白を含むオプション値を指定する場合は、値を二重引用符で囲みます。
8. 作業が終わったら、`exit` コマンドを入力して管理ユーティリティを終了します。

HTTPS 対応サーバー環境でのコマンド ライン インターフェイス (CLI) プロパティの設定

自己署名証明書を使用している場合は、必ずその証明書をローカル マシンにインポートしてください。

1. 自己署名証明書をインポートします。例:

```
keytool -importkeystore -srckeystore "C:\Pitney
Bowes\Spectrum\server\conf\certs\node-keystore.p12"
-destkeystore "C:\Pitney
Bowes\Spectrum\server\conf\certs\truststore.p12" -deststoretype pkcs12
```

2. CLI 実行可能ファイルと同じディレクトリに、`cli.properties` というファイルを作成します。

   サンプル ファイルとその内容を以下に示します。

```
# sample properties
spectrum.encryption.keystoreType=pkcs12
spectrum.encryption.keystore=C:\\Users\\Spectrum\\mycerts\\node-keystore.p12
spectrum.encryption.keystorePassword=p1tn3yb0w3s
spectrum.encryption.keystoreAlias=spectrum
spectrum.encryption.truststoreType=pkcs12
spectrum.encryption.truststore=C:\\Users\\Spectrum\\mycerts\\truststore.p12
spectrum.encryption.truststorePassword=p1tn3yb0w3s
spectrum.encryption.truststoreAlias=spectrum
spectrum.encryption.trustAllHosts=true
spectrum.encryption.trustSelfSigned=false
```

管理ユーティリティをスクリプトから使用する

管理ユーティリティのいくつかのコマンドは、スクリプト ファイルから実行できます。これは、管理ユーティリティや Management Console でコマンドを手動で実行する代わりに、スクリプトを使用して管理上のアクションを自動化または規格化したい場合に便利です。

1. テキスト エディタでスクリプト ファイルを作成します。スクリプト ファイルには、実行したいコマンドを記述します。

コマンドをスクリプト ファイルに追加するには、コマンド プロンプトでコマンドを入力する場合と同様にコマンドと必要なパラメータを入力します。コマンドは、1 行に 1 つ入力します。

スクリプト ファイルにコメントを挿入する場合は、次の表記方法を使用します。

`/*`	コメント ブロックの開始を示します。
`*/`	コメント ブロックの終了を示します。
`//`	インライン コメントを示します。行の先頭のみに使用します。
`;`	インライン コメントを示します。行の先頭のみに使用します。

2. スクリプトは、管理ユーティリティを実行するコンピュータに保存するか、管理ユーティリティを実行するコンピュータからアクセスが可能な場所に保存します。ファイル名と拡張子は任意に選択できます。ファイル拡張子 `.cli` の使用を推奨します。
3. スクリプトを実行するには、次のいずれかの操作を行います。

オプション	説明
コマンド ラインでスクリプトを実行するには	コマンド ライン、またはバッチ スクリプトやシェル スクリプトの中で、以下のように指定します。 `cli.cmd --cmdfile ScriptFile`
管理ユーティリティでスクリプトを実行するには	管理ユーティリティを開き、Spectrum™ Technology Platform コマンドを使用して `connect` サーバーに接続します。続いて `script` コマンドを使用してスクリプトを実行します。このコマンドの詳細については、**system_script.dita**を参照してください。

例: データフローをステージングから実稼働に移動する

Deduplication、AddressValidation、および DrivingDirections の 3 つのデータフローがあります。これらのデータフローに修正を加え、テストするためのステージング サーバーと、データフローを実行可能にする実稼働環境があります。これらのデータフローをステージング サーバーから実稼働ササーバーに移動する作業に一貫性を与え、自動化する必要があるため、管理ユーティリティ スクリプトを利用することにしました。使用するスクリプトは、次のような内容です。

```
// Connect to the staging server
connect --h stagingserver:8080 --u allan12 --p something123
```

```
// Export from staging
dataflow export --d "Deduplication" --e true --o exported
dataflow export --d "AddressValidation" --e true --o exported
dataflow export --d "DrivingDirections" --e true --o exported

// Close connection to the staging server
close

// Connect to the production server
connect --h productionserver:8080 --u allan12 --p something123

// Import to production
dataflow import --f exported\Deduplication.df
dataflow import --f exported\AddressValidation.df
dataflow import --f exported\DrivingDirections.df

// Close the connection to the production server
close
```

監査ログ情報

auditlog export

`auditlog export` コマンドは、すべての監査ログ ファイルに JSON アクティビティ ログを追加します。時間の形式は `yyyyMMddHHmmss` です。特定の期間を指定しなかった場合、開始日時はデフォルトで現在の日付と `auditlog export` コマンドを実行した時刻になります。

使用方法

`auditlog export --n` *name* `--v` *value* `--s` *startTime* `--e` *endTime* `--f` *filterBy* `--fw` *filterByWild* `--fa` *filterByAdditional*

必須	引数	説明
いいえ	`--n` *name*	アクティビティ ログで使用するフィールドの名前を指定します (例: "username")。
いいえ	`--v` *value*	`name` 定義に従った値を指定します (例: "admin")。
いいえ	**--s** *startTime*	監査ロギングの開始時刻と開始日を指定します。日付の形式は `yyyyMMddHHmmss` です。

必須	引数	説明
いいえ	**--e** *endTime*	監査ロギングの終了時刻と終了日を指定します。日付の形式は `yyyyMMddHHmmss` です。
いいえ	**--f** *filterBy*	アクティビティ ログにある情報のフィルタリングで使用するエンティティを指定します。例: `username:system`。
いいえ	**--fw** *filterByWild*	アスタリスク (*) を使用して、返された情報をフィルタリングできます。例えば、"info" という文字列が含まれているオブジェクト ID を検索するには、`objectID:*info` と指定します。
いいえ	**--fa** *filterByAdditional*	返された情報のフィルタリング時に使用する追加の値を指定します。例えば、返される情報を暦日に制限するために特定の日付を使用できます。

例

この例は、admin レベルのユーザ向けに、24 時間分の結果を返します。

```
auditlog export --s 20191231000000 --e 20200101000000 --f
userlevel:admin
```

auditlog info (audit log information summary)

`auditlog info` コマンドは、JSON 形式のカウント情報ファイルを監査ログ ファイルに追加します。時間の形式は `yyyyMMddHHmmss` です。特定の期間を指定しなかった場合、開始日時はデフォルトで現在の日付と auditlog info コマンドを実行した時刻になります。このコマンドには、返されるデータに関する複数のフィルタリング オプションが用意されています。JSON カウント ファイルは、選択した出力ディレクトリに割り当てられます。

使用方法

`auditlog info --n` *fieldName* `--s` *startTime* `--e` *endTime* `--f` *filterBy* `--fw` *filterByWild* `--fa` *filterByAdditional* `--ob` *orderBy* `--a` *ascending* `--o`*directory*

必須	引数	説明
いいえ	`--n` *fieldName*	返された監査ログ情報を含めるフィールドの名前を指定します。複数のフィールド名を指定できます。例えば、"ユーザ名" と "値" は対になるフィールドなので、結果には両方を含めたい場合があります。

必須	引数	説明
いいえ	`--s` *startTime*	監査ロギングの開始時刻と開始日を指定します。日付の形式は `yyyyMMddHHmmss` です。
いいえ	`--e` *endTime*	監査ロギングの終了時刻と終了日を指定します。日付の形式は `yyyyMMddHHmmss` です。
いいえ	`--f` *filterBy*	結果フィルタとして使用する特定のエンティティを指定します (例えば、"username:system")。
いいえ	**--fw** *filterByWild*	アスタリスク (*) を使用して、返された情報をフィルタリングできます。例えば、"info" という文字列が含まれているオブジェクト ID を検索するには、`objectID:*info` と指定します。
いいえ	**--fa** *filterByAdditional*	返された情報のフィルタリング時に使用する追加の値を指定します。例えば、返される情報を暦日に制限するために特定の日付を使用できます。
いいえ	**--ob** *orderByType*	返された情報を "loglevel" (ログ レベル) または "timestamp" (タイム スタンプ) によって順序付けできます。デフォルトは、タイム スタンプによる順序付けです。
いいえ	**--a** *ascending*	Boolean の結果を昇順で表示します。デフォルトの順序付けは true であり、このフィルタが指定されない場合は false になります。
いいえ	**--o** *directory*	監査ログ情報の出力ディレクトリを指定します。

例

この例は、admin レベルのユーザ向けに、最も古いイベントから最新のイベントへの順序で、24 時間分の結果を返し、`c:\PitneyBowes\auditlog_info\results` というディレクトリにその結果を送るように要求します。

```
auditlog info --s 20191231000000 --e 20200101000000 --f
userlevel:admin --ob timestamp --o
c:\PitneyBowes\auditlog_info\results
```

Business Glossary モジュール

用語集エンティティのエクスポート

このコマンドを使用すると、用語集エンティティを、Spectrum サーバーから、指定したディレクトリに CSV 形式でエクスポートできます。

使用方法

`glossaryentity export --n` ***name*** `--o` ***outputPath*** `--d` ***delimiter***

必須	引数	説明
はい	`--n` ***name***	エクスポートする用語集エンティティの名前を指定します。 **ヒント：** 用語集エンティティ名が正確にわからない場合は、`glossaryentity list` コマンドを使用して、名前のリストを取得できます。
いいえ	`--o` ***outputPath***	用語集エンティティをエクスポートする出力ディレクトリを指定します。 **注：** このパスを指定しない場合、エンティティは、コマンドを実行しているディレクトリに保存されます。
いいえ	`--d` ***delimiter***	エクスポート ファイルで使用する区切り文字を指定します。サポートされている区切り文字は、カンマ（`,`）、セミコロン（`;`）、パイプ（`|`）、タブ（`\t`）です。デフォルト値は、パイプ (|) です。 **注：** エンティティの説明にカンマが含まれる場合は、カンマを区切り文字に使用しないでください。その状況でカンマを使用すると、インポートが失敗する場合があります。

例

この例では、用語集エンティティ "Customer" を、Spectrum サーバーから `MyGlossary` フォルダにエクスポートします。エクスポート ファイルで使用する区切り文字はセミコロンです。

```
glossaryentity export --n Customer --o D:/Export/MyGlossary
--d ;
```

用語集エンティティのインポート

このコマンドを使用すると、用語集エンティティを CSV 形式でインポートして、バージョン 1.0 のドラフトを作成できます。

使用方法

`glossaryentity import --i` ***inputPath*** `--d` ***delimiter***

必須	引数	説明
はい	`--i` *inputPath*	用語集エンティティのインポート元となる、CSV ファイルのパス、または CSV ファイルを含むフォルダを指定します。 **注：** その入力パスがフォルダの場合、フォルダ内のすべてのファイルで同じ区切り文字を使用してください。 **注：** 問題が発生しないように、ファイルとフォルダのパスにはスラッシュを使用してください。
いいえ	`--d` *delimiter*	インポート ファイルで使用する区切り文字を指定します。サポートされている区切り文字は、カンマ（`,`）、セミコロン（`;`）、パイプ（`|`）、タブ（`\t`）です。デフォルト値は、パイプ (|) です。 **注：** エンティティの説明にカンマが含まれる場合は、カンマを区切り文字に使用しないでください。その状況でカンマを使用すると、インポートが失敗する場合があります。

例

次の例では、用語集エンティティを、`MyGlossary` フォルダの CSV ファイルからインポートします。フォルダ内のすべてのファイルで、区切り文字にカンマを使用します。

```
glossaryentity import --i D:/Import/MyGlossary --d ,
```

インポート ファイルのサンプル

```
EntityName:CustomerEntity
```

```
Description:顧客情報
PropertyName|PropertyDescription|PropertyDataType
FirstName|First Name|string
LastName|Last Name|string
Phone|Phone Number|Phone
EmailID|Email Address|Email
```

注： インポート用の *CSV* ファイルで使用しているデータ タイプがいずれも、意味型であることを確認してください。そうでない場合、インポート前に、対象のエンティティをサーバーに作成して、問題が発生しないようにします。

Glossaryentity List

このコマンドを使用して、既存の用語集エンティティのリストを表示します。

使用方法

```
glossaryentity list
```

例

この例では、すべての既存の用語集エンティティのリストを表示します。

```
glossaryentity list
```

Business Steward モジュール

bsm delete exceptions

このコマンドを使用して、リポジトリから例外レコードを削除します。データフロー内の特定のジョブにより作成された例外レコードを削除するか、データフロー内のすべてのジョブにより作成された例外レコードを削除するかを選択できます。

使用方法

```
bsm delete exceptions --n name --i id --r reports
```

必須	引数	説明
はい	`--n` *name*	データフロー名を指定します。
いいえ	`--i` *id*	ジョブ ID を指定します。データフロー内の 1 つのジョブにより作成された例外レコードを削除するには、この引数を追加します。データフロー内のすべてのジョブにより作成された例外レコードを削除するには、この引数を省略します。
いいえ	`--r` *reports*	例外レコードに関連するデータ品質レポートを削除するかどうかを指定します。 **true** 例外レコードと一緒にパフォーマンス データも削除します。これがデフォルト設定です。 **false** 例外レコードはリポジトリから削除されますが、パフォーマンス データは保持され、[Exception Monitor] の [データ品質] ページにその後も表示されます。

例

この例では、My Dataflow という名前のデータフロー内の、ジョブ ID 24 の例外をすべて削除します。

```
bsm delete exceptions --n "My Dataflow" --i 24
```

Data Hub モジュール

ハブ アルゴリズム媒介性

モデルで媒介性アルゴリズムを実行し、モデル プロパティに各エンティティの結果を保存します。

中心性アルゴリズムは、個々のエンティティと関係性における重要性と重要度を測定します。中心性アルゴリズムを実行する場合、アルゴリズムが返す値は要素の重要性を示します。媒介性アルゴリズムは、あるエンティティとその他のエンティティを結ぶ最短パスの個数を反映します。通常、どのエンティティが、グラフのある部分とその他の部分を結ぶ役割を果たしているかを検出する際に使用されます。

使用方法

`hub algorithm betweenness --m` *model* `--d` *direction* `--wp` *weightProperty* `--lv`*significantLowValues* `--op` *outputProperty* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	モデルを指定します。
いいえ	`--d` *direction*	*direction* を次のいずれかに設定して、アルゴリズムを適用する方向を指定します。 **in** エンティティにおける内部への関連性に基づいて結果を返します。 **out** エンティティにおける外部への関連性に基づいて結果を返します。 **both** エンティティにおける内部と外部、両方向への関連性に基づいて結果を返します。この値がデフォルトです。
いいえ	`--wp` *weightProperty*	関連性プロパティを指定します。これにより、関連性がどのくらい好ましくないかを測定できます。デフォルトでは、値が大きいほど、否定的な関連性が高まります。デフォルト設定は NULL です。
いいえ	`--lv` *significantLowValues*	関連性プロパティを重みとして使用している場合は、低い値を高い値よりも好ましいとするかどうかをこのプロパティで指定します。 **true** 重みとして使用する関連性プロパティで、低い値が高い値よりも好ましいとすることを指定します。例えば、プロパティがある種のランク付けシステムの場合、1 または第 1 位が最良の値と見なされます。また、プロパティが距離を示し、最短経路を決定するものである場合、5 マイルは 10 マイルより好ましいと見なされます。 **false** 重みとして使用する関連性プロパティで、高い値が低い値よりも好ましいとす

必須	引数	説明
		ることを指定します。この値がデフォルトです。
いいえ	`--op` *outputProperty*	出力プロパティ名に、アルゴリズムの名前とは異なる名前を指定します。デフォルトは Betweeness です。
いいえ	`--w` *waitForComplete*	同期モードにおいてジョブの完了を待つかどうかを指定します。 **true** 同期モードにおいてジョブの完了を待つように指定します。 **false** 同期モードにおいてジョブの完了を待たないように指定します。この値がデフォルトです。

例

次のコマンドにより、911 モデルで媒介性アルゴリズムが実行されます。

```
hub algorithm betweenness --m 911
```

ハブアルゴリズム近接性

モデル上で近接性アルゴリズムを実行し、モデル プロパティの各エンティティの結果を保存します。

中心性アルゴリズムは、個々のエンティティと関係性における重要性と重要度を測定します。中心性アルゴリズムを実行する場合、アルゴリズムが返す値は要素の重要性を示します。エンティティの近接中心性は、他のすべてのエンティティに対する平均距離 (逆距離) を測定します。近接性スコアの高いエンティティは、他のすべてのノードに対して最短距離に位置しています。

使用方法

`hub algorithm closeness --m` *model* `--d` *direction* `--m` *method* `--wp` *weightProperty* `--lv`*significantLowValues* `--op` *outputProperty* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	モデルを指定します。

必須	引数	説明
いいえ	`--d` *direction*	*direction* を次のいずれかに設定して、アルゴリズムを適用する方向を指定します。 **in** エンティティにおける内部への関連性に基づいて結果を返します。 **out** エンティティにおける外部への関連性に基づいて結果を返します。 **both** エンティティにおける内部と外部、両方向への関連性に基づいて結果を返します。この値がデフォルトです。
いいえ	`--me` *method*	結果が返る方法を指定します。 • **s**—[標準]。エンティティの連結 (すなわち、関連性) の数と、各エンティティへの最短パスの合計の逆数に基づいて結果が生じます。この値がデフォルトです。 • **d**—[Dangalchev]。別のエンティティにリンクされたエンティティの数だけでなく、リンクされた各エンティティの関連性の数も結果に影響します。 • **o**—[Opsahl]。各エンティティへの最短パスの逆数の合計に基づいて結果が生じます。
いいえ	`--wp` *weightProperty*	関連性プロパティを指定します。これにより、関連性がどのくらい好ましくないかを測定できます。デフォルトでは、値が大きいほど、否定的な関連性が高まります。デフォルト設定は NULL です。
いいえ	`--lv` *significantLowValues*	関連性プロパティを重みとして使用している場合は、低い値を高い値よりも好ましいとするかどうかをこのプロパティで指定します。 **true** 重みとして使用する関連性プロパティで、低い値が高い値よりも好ましいとすることを指定します。例えば、プロパティがある種のランク付けシステムの場合、1 または第 1 位が最良の値と見なされます。また、プロパティが距離を示し、最短経路を決定するものである場合、5 マ

必須	引数	説明
		イルは 10 マイルより好ましいと見なされます。 **false** 重みとして使用する関連性プロパティで、高い値が低い値よりも好ましいとすることを指定します。この値がデフォルトです。
いいえ	`--op` *outputProperty*	出力プロパティ名に、アルゴリズムの名前とは異なる名前を指定します。デフォルトは Closeness です。
いいえ	`--w` *waitForComplete*	同期モードにおいてジョブの完了を待つかどうかを指定します。 **true** 同期モードにおいてジョブの完了を待つように指定します。 **false** 同期モードにおいてジョブの完了を待たないように指定します。この値がデフォルトです。

例

以下により、911 モデルで近接性アルゴリズムが実行されます。

```
hub algorithm closeness --m 911
```

ハブ アルゴリズム次数

モデルで次数アルゴリズムを実行し、モデル プロパティに各エンティティの結果を保存します。

中心性アルゴリズムは、個々のエンティティと関係性における重要性と重要度を測定します。中心性アルゴリズムを実行する場合、アルゴリズムが返す値は要素の重要性を示します。次数アルゴリズムは、エンティティにおける関連性の個数を反映します次数中心性アルゴリズムは、グラフ内で、重要なエンティティや、多くの関連性を持つエンティティを確認するのに役立ちます。

使用方法

`hub algorithm degree --m` *model* `--d` *direction* `--wp` *weightProperty* `--lv`*significantLowValues* `--op` *outputProperty* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	モデルを指定します。
いいえ	`--d` *direction*	*direction* を次のいずれかに設定して、アルゴリズムを適用する方向を指定します。 **in** エンティティにおける内部への関連性に基づいて結果を返します。 **out** エンティティにおける外部への関連性に基づいて結果を返します。 **both** エンティティにおける内部と外部、両方向への関連性に基づいて結果を返します。この値がデフォルトです。
いいえ	`--wp` *weightProperty*	関連性プロパティを指定します。これにより、関連性がどのくらい好ましくないかを測定できます。デフォルトでは、値が大きいほど、否定的な関連性が高まります。デフォルト設定は NULL です。
いいえ	`--lv` *significantLowValues*	関連性プロパティを重みとして使用している場合は、低い値を高い値よりも好ましいとするかどうかをこのプロパティで指定します。 **true** 重みとして使用する関連性プロパティで、低い値が高い値よりも好ましいとすることを指定します。例えば、プロパティがある種のランク付けシステムの場合、1 または第 1 位が最良の値と見なされます。また、プロパティが距離を示し、最短経路を決定するものである場合、5 マイルは 10 マイルより好ましいと見なされます。 **false** 重みとして使用する関連性プロパティで、高い値が低い値よりも好ましいとすることを指定します。この値がデフォルトです。

必須	引数	説明
いいえ	`--op` *outputProperty*	出力プロパティ名に、アルゴリズムの名前とは異なる名前を指定します。デフォルトは Degree です。
いいえ	`--w` *waitForComplete*	同期モードにおいてジョブの完了を待つかどうかを指定します。 **true** 同期モードにおいてジョブの完了を待つように指定します。 **false** 同期モードにおいてジョブの完了を待たないように指定します。この値がデフォルトです。

例

以下により、911 モデルで次数アルゴリズムが実行されます。

```
hub algorithm degree --m 911
```

ハブ アルゴリズム影響性

モデルで影響性アルゴリズムを実行し、モデル プロパティに各エンティティの結果を保存します。影響性アルゴリズムは、高得点エンティティとの関連性に基づくエンティティの重要性を測定します。

中心性アルゴリズムは、個々のエンティティと関係性における重要性と重要度を測定します。中心性アルゴリズムを実行する場合、アルゴリズムが返す値は要素の重要性を示します。影響性アルゴリズムは、固有ベクトル中心性を実装して、エンティティの遷移的な影響性または中心性を測定します。高得点エンティティと関連している場合、低得点エンティティと関連している場合よりも、あるエンティティに高い得点を与えることができます。高得点は、あるエンティティが、得点の高い別のエンティティと関係していることを意味します。

使用方法

`hub algorithm influence --m` *model* `--d` *direction* `--p` *precision* `--wp` *weightProperty* `--lv`*significantLowValues* `--op` *outputProperty* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	モデルを指定します。

必須	引数	説明
いいえ	`--d` *direction*	*direction* を次のいずれかに設定して、アルゴリズムを適用する方向を指定します。 **in** エンティティにおける内部への関連性に基づいて結果を返します。 **out** エンティティにおける外部への関連性に基づいて結果を返します。 **both** エンティティにおける内部と外部、両方向への関連性に基づいて結果を返します。この値がデフォルトです。
いいえ	`--p` *precision*	結果の精度を指定します。精度が低ければより正確な結果が返されますが、アルゴリズムの実行速度は遅くなります。この引数に設定できる値の範囲は 0.00001 ～ 0.1 です。デフォルト値は 0.01 です。
いいえ	`--wp` *weightProperty*	関連性プロパティを指定します。これにより、関連性がどのくらい好ましくないかを測定できます。デフォルトでは、値が大きいほど、否定的な関連性が高まります。デフォルト設定は NULL です。
いいえ	`--lv` *significantLowValues*	関連性プロパティを重みとして使用している場合は、低い値を高い値よりも好ましいとするかどうかをこのプロパティで指定します。 **true** 重みとして使用する関連性プロパティで、低い値が高い値よりも好ましいとすることを指定します。例えば、プロパティがある種のランク付けシステムの場合、1 または第 1 位が最良の値と見なされます。また、プロパティが距離を示し、最短経路を決定するものである場合、5 マイルは 10 マイルより好ましいと見なされます。 **false** 重みとして使用する関連性プロパティで、高い値が低い値よりも好ましいとす

必須	引数	説明
		ることを指定します。この値がデフォルトです。
いいえ	`--op` *outputProperty*	出力プロパティ名に、アルゴリズムの名前とは異なる名前を指定します。デフォルトは Influence です。
いいえ	`--w` *waitForComplete*	同期モードにおいてジョブの完了を待つかどうかを指定します。 **true** 同期モードにおいてジョブの完了を待つように指定します。 **false** 同期モードにおいてジョブの完了を待たないように指定します。この値がデフォルトです。

例

次のコマンドにより、911 モデルでアルゴリズムが実行されます。

```
hub algorithm influence --m 911
```

hub backup all

すべての Data Hub モデルをバックアップします。

すべての Data Hub モデルのフル バックアップまたは増分バックアップを実行するには、`hub backup all` コマンドを使用します。増分バックアップでは、前回の更新以降に行われたモデルの変更が追加されます。別の場所を指定しない限り、モデルは Data Hub モデル用のデフォルト バックアップ ディレクトリに追加されます。

使用方法

`hub backup all --f` *fullBackup* `--p` *path*

必須	引数	説明
いいえ	`--f` *fullBackup*	すべてのモデルのフル バックアップまたは増分バックアップを実行します。*fullBackup* には、次のいずれかを指定します。 **true**

必須	引数	説明
		すべてのモデルのフル バックアップを実行します。フルバックアップにより、モデルの既存バックアップはすべて置き換えられます。これがデフォルト設定です。 **false** 既存のバックアップに対するすべてのモデルの増分バックアップを実行します。
いいえ	`--p` *path*	バックアップを保存するパスおよびフォルダを指定します。このオプションを省略すると、バックアップはデフォルトのバックアップ ディレクトリに配置されます。 **注：** デフォルトのバックアップ ディレクトリの場所 (*SpectrumFolder*`/server/modules/hub/db/backups`) は *SpectrumFolder*`/server/modules/hub/hub.properties` ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

例

この例では、既存のすべてのモデルをローカルの C ドライブ上の HubBackup というフォルダにバックアップします。

```
hub backup all --f true --p C:\HubBackup
```

hub backup delete

Data Hub モデルのバックアップを削除します。

Data Hub モデルのバックアップを削除するには、`hub backup delete` コマンドを使用します。別の場所を指定しない限り、モデルは Data Hub モデル用のデフォルト バックアップ ディレクトリから削除されます。

使用方法

`hub backup delete --m` *model* `--p` *path*

必須	引数	説明
はい	`--m` *model*	削除するモデル バックアップの名前を指定します。

必須	引数	説明
いいえ	`--p` *path*	バックアップが保存されているパスおよびフォルダを指定します。このオプションを省略すると、デフォルトのバックアップ ディレクトリから、モデルのバックアップが削除されます。 **注：** デフォルトのバックアップ ディレクトリの場所 (*SpectrumFolder*`/server/modules/hub/db/backups`) は *SpectrumFolder*`/server/modules/hub/hub.properties` ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

例

この例では、デフォルトのバックアップ フォルダから PersonalBanking というバックアップ モデルを削除します。

```
hub backup delete --m PersonalBanking
```

hub backup list

Data Hub モデルのバックアップのリストを取得します。

`hub backup list` コマンドは、特定のフォルダにバックアップされているすべての Data Hub モデルのリストを返します。このコマンドは、別の場所を指定しない限り、Data Hub モデルのデフォルト バックアップ ディレクトリにあるモデルのリストを取得します。

使用方法

`hub backup list --p` *path*

必須	引数	説明
いいえ	`--p` *path*	バックアップが保存されているパスおよびフォルダを指定します。このオプションを省略すると、コマンドはデフォルトのバックアップ ディレクトリにあるバックアップ済みモデルのリストを返します。 **注：** デフォルトのバックアップ ディレクトリの場所 (*SpectrumFolder*`/server/modules/hub/db/backups`) は *SpectrumFolder*`/server/modules/hub/hub.properties` ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

例

この例は、デフォルトのバックアップ フォルダからバックアップ済みモデルのリストを返します。

```
hub backup list
```

hub backup model

特定の Data Hub モデルをバックアップします。

指定の Data Hub モデルのフル バックアップまたは増分バックアップを実行するには、`hub backup model` コマンドを使用します。増分方式では、前回の更新以降に行われたモデルの変更が追加されます。バックアップは、`path` (`--p`) オプションによって別の場所を指定しない限り、Data Hub モデル用のデフォルトのバックアップ フォルダに配置されます。

使用方法

`hub backup model --m` ***model*** `--f` ***fullBackup*** `--p` ***path***

必須	引数	説明
はい	`--m` ***model***	バックアップするモデルの名前を指定します。
いいえ	`--f` ***fullBackup***	モデルのフル バックアップまたは増分バックアップを実行します。***fullBackup*** には、次のいずれかを指定します。 **true** モデルの完全な初期バックアップを実行します。これがデフォルト設定です。 **false** 既存のバックアップに対するモデルの増分バックアップを実行します。
いいえ	`--p` ***path***	バックアップを保存するパスおよびフォルダを指定します。このオプションを省略すると、バックアップはデフォルトのバックアップ ディレクトリに配置されます。 **注：** デフォルトのバックアップ ディレクトリの場所 (*SpectrumFolder*`/server/modules/hub/db/backups`) は *SpectrumFolder*`/server/modules/hub/hub.properties` ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

例

この例では、ConsumerFraud という 1 つのモデルを `C:\DataHub` ディレクトリの下の HubModelBackup フォルダにバックアップします。既に同じ名前のモデルが存在する場合は、復元されるモデルが更新されます。

```
hub backup model --m ConsumerFraud --f false --p
C:\DataHub\HubModelBackups\model.ConsumerFraud
```

hub backup restore

Data Hub モデルをバックアップからリストアします。

Data Hub モデルのバックアップをリストアするには、`hub backup restore` コマンドを使用します。オプションで、同じ名前の既存モデルが存在しない場合にのみモデルをリストアするかどうかを選択できます。別の場所を指定しない場合は、デフォルトのバックアップ場所からモデルがリストアされます。

使用方法

`hub backup restore --m` ***model*** `--d` ***deleteIfExists*** `--p` ***path***

必須	引数	説明
はい	`--m` ***model***	リストアするモデルの名前を指定します。
いいえ	`--d` ***deleteIfExists***	同じ名前の既存モデルを削除するかどうかを指定します。ここで、***deleteIfExists*** は次のいずれかです。 **true** 既存のモデルを削除し、バックアップされているモデルをリストアします。これがデフォルト設定です。 **false** 既存のモデルをそのまま残し、バックアップされているモデルのリストアを行いません。
いいえ	`--p` ***path***	バックアップが保存されているパスおよびフォルダを指定します。このオプションを省略すると、バックアップはデフォルトのバックアップ ディレクトリからリストアされます。 **注：** デフォルトのバックアップ ディレクトリの場所 (*SpectrumFolder*`/server/modules/hub/db/backups`) は *SpectrumFolder*`/server/modules/hub/hub.properties`

必須	引数	説明
		ファイル内の `hub.models.path.base` 設定のコメントアウトを外して編集することで変更できます。

例

この例は、バックアップされている ConsumerFraud というモデルを `C:\DataHub\HubModelBackup` フォルダからリストアします。同名のモデルが既に存在する場合は、リストアされるモデルによって上書きされます。

```
hub backup restore --m ConsumerFraud --d true --p
C:\DataHub\HubModelBackup
```

hub job list

すべての Data Hub ジョブのリストを返します。

`hub job list` コマンドは、日時を指定して、または指定せずに、すべての Data Hub ジョブのリストを返します。

使用方法

`hub job list --f` *from datetime* `--t` *to datetime*

必須	引数	説明
いいえ	`--f` *from datetime*	特定の日時の範囲のリストを表示する場合に、範囲の開始日時を 'MM-dd-yyyy HH:mm:ss' の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `'12-31-2014 13:00:00'` と指定します。 日時の範囲を指定すると、`--f` 引数で指定した日付以降で、`--t` 引数で指定した日付より前に実行が開始したジョブがリストに含まれます。 この引数を省略すると、現在の日付に実行が開始されたジョブがリストに含まれます。
いいえ	`--t` *to datetime*	特定の日時の範囲のリストを表示する場合に、範囲の終了日時を 'MM-dd-yyyy HH:mm:ss' の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `'12-31-2014 13:00:00'` と指定します。 日時の範囲を指定すると、`--f` 引数で指定した日付以降で、`--t` 引数で指定した日付より前に実行が開始したジョブがリストに含まれます。

必須	引数	説明
		この引数を省略すると、`--f` 引数で指定した日付以降に実行が開始されたジョブがリストに含まれます。

例

この例では、2010 年 1 月 1 日の 00:00:00 以降に実行されたすべての Data Hub ジョブを一覧表示します。

```
hub job list --f '01-01-2010 00:00:00'
```

hub job status

Data Hub ジョブのステータスを返します。

Data Hub ジョブのステータスを返すには、`hub job status` コマンドを使用します。

使用方法

`hub job status --id` ***jobID***

必須	引数	説明
はい	`--id` ***jobID***	Data Hub ジョブの ID を指定します。

例

次の例は、Data Hub ジョブ 24 のステータスを返します。

```
hub job status --id 24
```

hub model clear

Data Hub モデルのコンテンツを削除します。

Data Hub モデルのコンテンツを削除し、モデル自体とそのメタデータをそのまま残すには、`hub model clear` コマンドを使用します。

使用方法

`hub backup clear --m` ***model***

必須	引数	説明
はい	`--m` *model*	コンテンツをクリアするモデルの名前を指定します。

例

次の例は、CustomerDB_032018 というモデルをクリアします。

```
hub model clear --m CustomerDB_032018
```

hub model copy

Data Hub モデルのコンテンツをコピーします。

Data Hub モデルのコンテンツ、オプションでそのモニターとクエリをコピーするには、`hub model copy` コマンドを使用します。

使用方法

`hub model copy --m` *model* `--nm` *newmodel* `--cm` *copymonitors* `--cq` *copyqueries*

必須	引数	説明
はい	`--m` *model*	コピーするモデルの名前を指定します。
はい	`--nm` *newmodel*	新しいモデルの名前を指定します。
いいえ	`--cm` *copymonitors*	古いモデルから既存のモニターを新しいモデルにコピーするかどうかを指定します。ここで、*copymonitors* は次のいずれかです。 **true** モニターをコピーします。これがデフォルト設定です。 **false** モニターをコピーしません。
いいえ	`--cq` *copyqueries*	保存されたクエリを古いモデルから新しいモデルにコピーするかどうかを指定します。ここで、*copyqueries* は次のいずれかです。 **true** クエリをコピーします。これがデフォルト設定です。 **false** クエリをコピーしません。

例

この例は、デフォルトのバックアップ フォルダから CustomerBanking_DataType という名前のモデルをコピーして、コピーに CustomerBanking_DataType_New という名前を付けます。また、古いモデルに関連付けられているすべてのモニターまたはクエリを新しいモデルにコピーします。

```
hub model copy --m CustomerBanking_DataType --nm
CustomerBanking_DataType_New --cm true --cq true
```

hub model create security

Data Hub モデルのセキュア エンティティを作成します。

Data Hub モデルのセキュア エンティティを作成するには、`hub model create security` コマンドを使用します。Management Console `System > Security > Access Control` でのオーバーライド オプションが用意されています。

使用方法

`hub model create security --m` ***model***

必須	引数	説明
はい	`--m` ***model***	セキュア エンティティを作成するモデルの名前を指定します。

例

次の例は、Single_Account_Holders というモデルのセキュア エンティティを作成します。

```
hub model create security --m Single_Account_Holders
```

hub model delete

Data Hub モデルを削除します。

特定の Data Hub モデルを削除するには、`hub model delete` コマンドを使用します。

使用方法

`hub model delete --m` ***model***

必須	引数	説明
はい	`--m` *model*	削除するモデルの名前を指定します。

例

次の例は PersonalBanking というモデルを削除します。

```
hub model delete --m PersonalBanking
```

hub model export

Data Hub モデルをエクスポートします。

Data Hub モデルをフォルダ構造としてエクスポートするには、`hub model export` コマンドを使用します。

使用方法

`hub model export --m` *ModelName* `--p` *Path* `--xd` *TrueOrFalse*

必須	引数	説明
はい	`--m` *ModelName*	エクスポートするモデルの名前を指定します。
はい	`--p` *Path*	エクスポート フォルダを保存するパスを指定します。
いいえ	`--xd` *TrueOrFalse*	エクスポート対象からデータを除外するかどうかを指定します。ここで、*TrueOrFalse* は次のいずれかです。 **true** データを除外します。これがデフォルト設定です。 **false** データを除外しません。

例

次の例では、Fraud_2015 というモデルを C ドライブの `HubModels` ディレクトリにフォルダ形式でエクスポートします。エクスポート先ではデータも保持されます。

```
hub model export --m Fraud_2015 --p C:\HubModels --xd false
```

hub model import

Data Hub モデルをインポートします。

Data Hub モデルをインポートするには、`hub model import` コマンドを使用します。

使用方法

`hub model import --m` *ModelName* `--p` *Path*

必須	引数	説明
はい	`--m` *ModelName*	エクスポートするモデルの名前を指定します。
はい	`--p` *Path*	インポートするモデルの場所へのパスを指定します。

例

次の例は、C ドライブの `HubModels` フォルダ から Fraud_2015 というモデルをインポートします。

```
hub model import --m Fraud_2015 --p C:\HubModels
```

hub model list

Data Hub モデルをリスト化します。

`hub model list` コマンドは、すべての Data Hub モデルのリストを各モデルのエンティティおよび関連性の数と共に返します。

使用方法

`hub model list --c` *counts*

必須	引数	説明
いいえ	`--c` *counts*	エンティティおよび関連性の数を含めるかどうかを指定します。 *counts* には、次のいずれかを指定します。 **true** 数を含めます。これがデフォルト設定です。 **false** 数を含めません。

例

この例は、すべての Data Hub モデルのリスト、および各モデルのエンティティと関連性の数を返します。

```
hub model list --c true
```

hub model reindex

Data Hub モデルのインデックスを再作成します。

`hub model reindex` コマンドは、単一の Data Hub モデル、またはすべての Data Hub モデルのインデックスを再作成します。このユーティリティは、以下に示すようにモデルごとのステータス メッセージを個別の行で返します。

```
|   MODEL NAME   | STATUS | INDEX TYPE        |       FAILURE MESSAGE IF APPLICABLE
|
+----------------+--------+-------------------+-----------------------------------------+
| Fraud_Index    | PASSED |lucene+native-2.0 |
| Index_Insured | PASSED |lucene+native-2.0 |
```

また、このユーティリティは必要に応じてエラー メッセージも返します。

```
|   MODEL NAME   | STATUS | INDEX TYPE        |       FAILURE MESSAGE IF APPLICABLE
|
+----------------+--------+-------------------+-----------------------------------------+
| HUB_Index      | FAILED |null               | Model does not exist.
```

使用方法

`hub model reindex --m` ***model*** `--a` ***all***

必須	引数	説明
いいえ	`--m` ***model***	単一のモデルのインデックスを再作成する場合は、コンテンツのインデックスを再作成するモデルの名前を指定します。 **注：** --m または --a を指定する必要があります。両方は指定できません。
いいえ	`--a` ***all***	すべてのモデルのインデックスを再作成することを指定します。***all*** には、次のいずれかを指定します。 **true** すべてのモデルのインデックスを再作成します。 **false**

必須	引数	説明
		すべてのモデルのインデックスは再作成しません。これがデフォルト設定です。 **注：** --a または --m を指定する必要があります。両方は指定できません。

例

この例では、すべての Data Hub モデルのインデックスを再作成します。

```
hub model reindex --a
```

hub schema copy

モデル メタデータをコピーします。

Data Hub モデルのメタデータ、オプションでそのモニターおよびクエリをコピーするには、`hub schema copy` コマンドを使用します。

使用方法

`hub schema copy --m` ***model*** `--nm` ***newmodel*** `--cm` ***copymonitors*** `--cq` ***copyqueries***

必須	引数	説明
はい	`--m` ***model***	スキーマをコピーするモデルの名前を指定します。
はい	`--nm` ***newmodel***	新しいモデルの名前を指定します。
いいえ	`--cm` ***copymonitors***	古いモデルから既存のモニターを新しいモデルにコピーするかどうかを指定します。ここで、***copymonitors*** は次のいずれかです。 **true** モニターをコピーします。これがデフォルト設定です。 **false** モニターをコピーしません。
いいえ	`--cq` ***copyqueries***	保存されたクエリを古いモデルから新しいモデルにコピーするかどうかを指定します。ここで、***copyqueries*** は次のいずれかです。 **true**

必須	引数	説明
		クエリをコピーします。これがデフォルト設定です。 **false** クエリをコピーしません。

例

次の例は、デフォルトのバックアップ フォルダにある PersonalLending というモデルのスキーマをコピーし、コピーしたものに PersonalLending_New という名前を付けます。また、古いモデルに関連するモニターをすべてコピーしますが、古いモデルに関連するクエリはコピーしません。

```
hub schema copy --m PersonalLending --nm PersonalLending_New
--cm true --cq false
```

hub schema delete entityProperty

モデル プロパティを削除します。

Data Hub モデルのプロパティを削除するには、`hub schema delete entityProperty` コマンドを使用します。

使用方法

`hub schema delete entityProperty --m` ***model*** `--e` ***entityType*** `--p` ***property*** `--w` ***waitForComplete***

必須	引数	説明
はい	`--m` ***model***	エンティティ タイプのプロパティを削除するモデルの名前を指定します。
いいえ	`--e` ***entityType***	対象となるエンティティ タイプを指定します。指定しない場合は、すべてのエンティティ タイプが対象となります。
はい	`--p` ***property***	削除するプロパティを指定します。
いいえ	`--w` ***waitForComplete***	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、***waitForComplete*** は次のいずれかです。 **true** ジョブの完了を待機します。 **false**

必須	引数	説明
		ジョブの完了を待機しません。これがデフォルト設定です。

例

この例では、Staff というモデルのエンティティ タイプ EmployeeName からプロパティ HireDate を削除します。

```
hub schema delete entityProperty --m Staff --e EmployeeName
--p HireDate
```

hub schema delete entityType

モデルのエンティティ タイプを削除します。

Data Hub モデルのエンティティ タイプを削除するには、`hub schema delete entityType` コマンドを使用します。オプションで、ジョブを同期モードで実行するかどうかを指定できます。

使用方法

`hub schema delete entityType --m` *model* `--e` *entityType* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	エンティティ タイプを削除するモデルの名前を指定します。
はい	`--e` *entityType*	削除するエンティティのタイプを指定します。
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例では、PersonalLending というモデルからエンティティ タイプ Employee を削除します。

```
hub schema delete entityType --m PersonalLending --e Employee
```

hub schema delete relationshipLabel

関連性ラベルをモデルから削除します。

関連性ラベルをモデルから削除するには、`hub schema delete relationshipLabel` コマンドを使用します。オプションで、同期モードで他のジョブを待機することもできます。

使用方法

`hub schema delete relationshipLabel --m` ***model*** `--r` ***relationshipLabel*** `--s` ***sourceEntityType*** `--t` ***targetEntityType*** `--w` ***waitForComplete***

必須	引数	説明
はい	`--m` ***model***	関連性ラベルを削除するモデルの名前を指定します。
はい	`--r` ***relationshipLabel***	削除する関連性ラベルを指定します。
いいえ	`--s` ***sourceEntityType***	ソース エンティティのタイプを指定します。
いいえ	`--t` ***targetEntityType***	ターゲット エンティティのタイプを指定します。
いいえ	`--w` ***waitForComplete***	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、***waitForComplete*** は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

次の例は、関連性ラベル Hired を Staff というモデルから削除します。

```
hub schema delete relationshipLabel --m Staff --r Hired
```

hub schema delete relationshipProperty

関連性プロパティをモデルから削除します。

Data Hub モデルの関連性プロパティを削除するには、`hub schema delete relationshipProperty` コマンドを使用します。

使用方法

`hub schema delete relationshipProperty --m` *model* `--r` *relationshipLabel* `--p` *property* `--s` *sourceEntityType* `--t` *targetEntityType* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	エンティティ タイプまたは関連性ラベルのプロパティを削除するモデルの名前を指定します。
いいえ	`--r` *relationshipLabel*	対象となる関連性ラベルを指定します。指定しない場合は、すべての関連性ラベルが対象となります。
はい	`--p` *property*	削除するプロパティを指定します。
いいえ	`--s` *sourceEntityType*	ソース エンティティのタイプを指定します。
いいえ	`--t` *targetEntityType*	ターゲット エンティティのタイプを指定します。
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例では、Staff というモデルの関連性ラベル Hired からプロパティ HireDate を削除します。

```
hub schema delete relationshipProperty --m Staff --r Hired --p HireDate
```

hub schema export

モデルをエクスポートします。

`hub schema export` コマンドは、Data Hub モデル、そのメタデータ、モニター、およびクエリをエクスポートします。モデルのエクスポート先のパスを指定しない場合は、ユーザが指定した名前を使用して、現在の作業ディレクトリにエクスポートされます。

使用方法

`hub schema export --m` *model* `--f` *file* `--cm` *copymonitors* `--cq` *copyqueries*

必須	引数	説明
はい	`--m` *model*	エクスポートするモデルの名前を指定します。
はい	`--p` *path*	(非推奨) エクスポート フォルダを保存するパスを指定します。このパスは、Spectrum™ Technology Platform サーバーをインストールした場所への相対パスです。
いいえ	`--f` *file*	エクスポート フォルダを保存するパスを指定します。このパスは、Spectrum™ Technology Platform 管理ユーティリティをインストールした場所への相対パスです。
いいえ	`--cm` *copymonitors*	既存のモニターをエクスポートするかどうかを指定します。*copymonitors* には、次のいずれかを指定します。 **true** モニターをエクスポートします。これがデフォルト設定です。 **false** モニターをエクスポートしません。
いいえ	`--cq` *copyqueries*	保存済みクエリをエクスポートするかどうかを指定します。*copyqueries* には、次のいずれかを指定します。 **true** クエリをエクスポートします。これがデフォルト設定です。 **false** クエリをエクスポートしません。

例

この例では、Fraud_2015 というモデルのスキーマを C ドライブの HubModels ディレクトリにエクスポートします。モニターはエクスポートしませんが、モデルに関連付けられているクエリはエクスポートします。

```
hub schema export --m Fraud_2015 --f C:\HubModels --cm false
--cq true
```

hub schema import

モデルをインポートします。

`hub schema import` コマンドは、Data Hub モデル、そのメタデータ、モニター、およびクエリをインポートします。モデルをインポートするパスを指定しない場合は、ユーザが指定した名前のファイルが現在の作業ディレクトリで検索されます。

使用方法

`hub schema import --m` ***model*** `--f` ***file***

必須	引数	説明
はい	`--m` ***model***	スキーマをインポートするモデルの名前を指定します。
いいえ	`--p` ***path***	(非推奨) スキーマをインポートするモデルの場所のパスを指定します。このパスは、Spectrum™ Technology Platform サーバーをインストールした場所への相対パスです。
いいえ	`--f` ***file***	スキーマをインポートするモデルの場所のパスを指定します。このパスは、Spectrum™ Technology Platform 管理ユーティリティをインストールした場所への相対パスです。

例

この例では、Fraud_2015 というモデルのスキーマを C ドライブの HubModels ディレクトリからインポートします。

```
hub schema import --m Fraud_2015 --f C:\HubModels
```

hub schema importLogicalModel

Metadata Insights Logical Model を Data Hub にインポートします。

`hub schema importLogicalModel` コマンドは、Metadata Insights の論理モデルを Data Hub にインポートします。

使用方法

`hub schema importLogicalModel --m` ***model*** `--n` ***logicalModelName***

必須	引数	説明
はい	`--m` ***model***	Data Hub のモデルに付ける名前を指定します。

必須	引数	説明
いいえ	`--n` *logicalModelName*	スキーマをインポートする Metadata Insights モデルの名前を指定します。

例

この例は、Insured という名前の Metadata Insights モデルをインポートして、Insured2018 という名前を付けます。

```
hub schema importLogicalModel --m Insured2018 --n Insured
```

hub schema list all

モデルのエンティティ タイプ、関連性のラベルおよび総数をリスト化します。

あるモデルについて、エンティティ タイプと関連性のラベルおよび総数のリストを返すには、基本の `hub schema list all` コマンドを使用します。エンティティ プロパティ、関連性ラベルの接続、関連性プロパティ、インデックス作成済みプロパティを含めるには、*verbose* 引数を追加します。

使用方法

`hub schema list all --m` *model* `--v` *verbose*

必須	引数	説明
はい	`--m` *model*	スキーマ情報を一覧表示するモデルの名前を指定します。
いいえ	`--v` *verbose*	詳細な出力を含めるかどうかを指定します。ここで、*verbose* は次のいずれかです。 **true** 詳細な出力を含めます。 **false** 詳細な出力を含めません。これがデフォルト設定です。

例

次の例は、PersonalBanking というモデルのすべての関連性プロパティを一覧表示し、詳細な出力は表示しません。

```
hub schema list all --m PersonalBanking
```

hub schema list entityProperties

モデルのエンティティ プロパティをリスト化します。

モデルのすべてのエンティティ プロパティのリストを返すには、`hub schema list entityProperties` コマンドを使用します。

使用方法

`hub schema list entityProperties --m` *model* `--e` *entityType* `--i` *indexedOnly*

必須	引数	説明
はい	`--m` *model*	エンティティ プロパティを一覧表示するモデルの名前を指定します。
いいえ	`--e` *entityType*	結果を指定のエンティティ タイプに制限します。
いいえ	`--i` *indexedOnly*	結果をインデックス作成済みプロパティのみに制限するかどうかを指定します。ここで、*indexedOnly* は次のいずれかです。 **true** 結果を制限します。これがデフォルト設定です。 **false** 結果を制限しません。

例

次の例は、PersonalBanking というモデルのすべてのエンティティ プロパティを一覧表示し、Customer タイプとインデックス作成済みプロパティのみが含まれるように結果をフィルタリングします。

```
hub schema list entityProperties --m PersonalBanking --e
Customer
```

hub schema list entityTypes

モデルのエンティティ タイプをリスト化します。

`hub schema list entityTypes` コマンドは、モデルのすべてのエンティティ タイプのリストを返します。

使用方法

```
hub schema list entityTypes --m model
```

必須	引数	説明
はい	`--m` *model*	エンティティ タイプのリストを取得するモデルの名前を指定します。

例

この例では、Fraud という名前のモデルのすべてのエンティティ タイプのリストを取得します。

```
hub schema list entityTypes --m Fraud
```

hub schema list relationshipLabels

モデルの関連性ラベルをリスト化します。

`hub schema list relationshipLabels` コマンドは、モデルのすべての関連性ラベルの一覧を返します。

使用方法

```
hub schema list relationshipLabels --m model --s sourceEntityType --t targetEntityType
--c showConnections
```

必須	引数	説明
はい	`--m` *model*	関連性ラベルを返すモデルの名前を指定します。
いいえ	`--s` *sourceEntityType*	ソース エンティティのタイプを指定します。
いいえ	`--t` *targetEntityType*	ターゲット エンティティのタイプを指定します。
いいえ	`--c` *showConnections*	ソースおよびターゲット エンティティのタイプを表示するかどうかを指定します。*showConnections* には、次のいずれかを指定します。 **true** 接続を表示します。 **false** 接続を表示しません。これがデフォルト設定です。 .

例

この例は、June2017 という名前のモデルの、Customer タイプのソース エンティティと AccountType タイプのターゲット エンティティについて、関連性ラベルの一覧とソースおよびターゲット エンティティ タイプを返します。

```
hub schema list relationshipLabels --m June2017 --s Customer --t AccountType --c
```

hub schema list relationshipProperties

モデルのエンティティ プロパティをリスト化します。

モデルのすべてのエンティティ プロパティのリストを返すには、`hub schema list relationshipProperties` コマンドを使用します。

使用方法

`hub schema list relationshipProperties --m` ***model*** `--r` ***relationshipLabel***

必須	引数	説明
はい	`--m` ***model***	関連性プロパティの一覧を返すモデルの名前を指定します。
いいえ	`--r` ***relationshipLabel***	結果をフィルタリングして、指定した関連性ラベル タイプのみを返します。

例

この例は、PrivateBanking という名前のモデルのすべての関連性プロパティのリストを取得し、結果をフィルタリングして Current タイプのみを返します。

```
hub schema list relationshipProperties --m PrivateBanking --r
Current
```

hub schema modify indexType

モデルのインデックス タイプを変更します。

Data Hub モデル プロパティのインデックス タイプを変更するには、`hub schema modify indexType` コマンドを使用します。

使用方法

`hub schema modify indexType --m` ***model*** `--p` ***property*** `--i` ***index*** **`--w`** ***waitForComplete***

必須	引数	説明
はい	`--m` *model*	エンティティ タイプを変更するモデルの名前を指定します。
はい	`--p` *property*	インデックスを作成するプロパティを指定します。
はい	`--i` *index*	プロパティ インデックスを指定します。 **NONE** プロパティ インデックスを削除します。 **EXACT** プロパティ インデックスを完全一致に設定します。 **CASE_INSENSITIVE** プロパティ インデックスで大文字と小文字を区別しません。
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例は、Staff という名前のモデルの HireDate プロパティのインデックス タイプを変更し、インデックスを完全一致に設定します。

```
hub schema modify indexType --m Staff --p HireDate --i EXACT
```

hub schema rename entityProperty

モデル プロパティの名前を変更します。

Data Hub モデル プロパティを削除するには、`hub schema rename entityProperty` コマンドを使用します。

使用方法

`hub schema rename entityProperty --m` *model* `--e` *entityType* `--p` *property* `--np` *newProperty* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	エンティティ プロパティの名前を変更するモデルの名前を指定します。
いいえ	`--e` *entityType*	対象となるエンティティ タイプを指定します。指定しない場合は、すべてのエンティティ タイプが対象となります。
はい	`--p` *property*	名前を変更するプロパティを指定します。
はい	`--np` *newProperty*	新しいプロパティ名を指定します。
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例は、関連性ラベル Hired が設定された Staff という名前のモデルの HireDate プロパティの名前を Start Date に変更します。

```
hub schema rename entityProperty --m Staff --r Hired --p
HireDate --np StartDate
```

hub schema rename entityType

モデルのエンティティ タイプの名前を変更します。

`hub schema rename entityType` コマンドは、Data Hub モデルのエンティティ タイプの名前を変更します。

使用方法

`hub schema rename entityType --m` *model* `--e` *entityType* `--ne` *newEntityType* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	エンティティ タイプの名前を変更するモデルの名前を指定します。
はい	`--e` *entityType*	名前を変更するエンティティ タイプを指定します。

必須	引数	説明
はい	`--ne` *newEntityType*	新しいエンティティ タイプを指定します。
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例では、PersonalLending というモデルのエンティティ タイプの名前を Employee から Staff に変更します。

```
hub schema rename entityType --m PersonalLending --e Employee
--ne Staff
```

hub schema rename relationshipLabel

関連性ラベルの名前を変更します。

`hub schema rename relationshipLabel` コマンドは、Data Hub モデルの関連性ラベルの名前を変更します。

使用方法

`hub schema rename relationshipLabel --m` *model* `--r` *relationshipLabel* `--nr` *newRelationshipLabel* `--s` *sourceEntityType* `--t` *targetEntityType* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	関連性ラベルの名前を変更するモデルの名前を指定します。
はい	`--r` *relationshipLabel*	名前を変更する関連性ラベルを指定します。
はい	`--nr` *newRelationshipLabel*	関連性ラベルの新しい名前を指定します。
いいえ	`--s` *sourceEntityType*	ソース エンティティのタイプを指定します。
いいえ	`--t` *targetEntityType*	ターゲット エンティティのタイプを指定します。

必須	引数	説明
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例では、Staff というモデルの関連性ラベル Hired の名前を Employed に変更します。

```
hub schema rename relationshipLabel --m Staff --r Hired --nr
Employed
```

hub schema rename relationshipProperty

モデル プロパティの名前を変更します。

Data Hub モデルのプロパティの名前を変更するには、`hub schema rename relationshipProperty` コマンドを使用します。

使用方法

`hub schema rename relationshipProperty --m` *model* `--r` *relationshipLabel* `--p` *property* `--np` *newProperty* `--s` *sourceEntityType* `--t` *targetEntityType* `--w` *waitForComplete*

必須	引数	説明
はい	`--m` *model*	関連性プロパティの名前を変更するモデルの名前を指定します。
いいえ	`--r` *relationshipLabel*	ターゲットとなる関連性ラベルを指定します。
はい	`--p` *property*	名前を変更するプロパティを指定します。
はい	`--np` *newProperty*	新しいプロパティ名を指定します。
いいえ	`--s` *sourceEntityType*	ソース エンティティのタイプを指定します。
いいえ	`--t` *targetEntityType*	ターゲット エンティティのタイプを指定します。

必須	引数	説明
いいえ	`--w` *waitForComplete*	ジョブの完了まで同期モードで待機するかどうかを指定します。ここで、*waitForComplete* は次のいずれかです。 **true** ジョブの完了を待機します。 **false** ジョブの完了を待機しません。これがデフォルト設定です。

例

この例は、関連性ラベル Hired が設定された Staff という名前のモデルの HireDate 関連性プロパティの名前を Start Date に変更します。

```
hub schema rename relationshipProperty --m Staff --r Hired --p HireDate --np StartDate
```

データ ソース

FTP

data source ftp add

`data source ftp add` コマンドは、Spectrum™ Technology Platform と FTP サーバーの間の接続を作成します。

使用方法

`data source ftp add --n` *ConnectionName* `--h` *Host* `--o` *Port* `--u` *Username* `--p` *Password*

必須	引数	説明
はい	`--n` *ConnectionName*	接続の名前を指定します。任意の名前にすることができます。
はい	`--h` *Host*	FTP サーバーのホスト名または IP アドレスを指定します。

必須	引数	説明
いいえ	`--o` *Port*	FTP サーバーとの通信で使うネットワーク ポートを指定します。
いいえ	`--u` *Username*	FTP サーバーへの接続で使うユーザ名 (必要な場合)。
いいえ	`--p` *Password*	FTP サーバーへの接続で使うパスワード (必要な場合)。

例

この例では、FTP サーバー MyFTPServer に対する接続を作成します。

```
data source ftp add --n NorthernRegionCustomers --h MyFTPServer
--u ExampleUsername --p Example123
```

data source ftp delete

`data source ftp delete` コマンドは、Spectrum™ Technology Platform と FTP サーバーの間の接続を削除します。

使用方法

`data source ftp delete--n`*ConnectionName*

必須	引数	説明
はい	`--n` *ConnectionName*	削除する接続の名前を指定します。接続のリストを表示するには、`data source ftp list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続を削除します。

```
data source ftp delete --n NorthernRegionCustomers
```

data source ftp list

`data source ftp list` コマンドは、Spectrum™ Technology Platform サーバー上で定義されている FTP データベース接続のリストを返します。

使用方法

```
data source ftp list
```

data source ftp test

`data source ftp test` コマンドは、Spectrum™ Technology Platform と FTP サーバーとの間の接続をテストします。

使用方法

`data source ftp test--n`*ConnectionName*

必須	引数	説明
はい	`--n` *ConnectionName*	テストする接続の名前を指定します。接続のリストを表示するには、`data source ftp list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続をテストしています。

```
data source ftp test --n NorthernRegionCustomers
```

data source ftp update

`data source ftp update` コマンドは、Spectrum™ Technology Platform と FTP サーバーの間の接続に変更を加えます。

使用方法

`data source ftp update--n`*ConnectionName*`--h`*Host*`--o`*Port*`--u`*Username*`--p`*Password*

必須	引数	説明
はい	`--n` *ConnectionName*	変更する接続の名前を指定します。接続のリストを表示するには、`data source ftp list` コマンドを使用します。
はい	`--h` *Host*	FTP サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	FTP サーバーとの通信で使うネットワーク ポートを指定します。
いいえ	`--u` *Username*	FTP サーバーへの接続で使うユーザ名 (必要な場合)。
いいえ	`--p` *Password*	FTP サーバーへの接続で使うパスワード (必要な場合)。

例

この例では、FTP 接続 NorthernRegionCustomers に変更を加えます。ホストドを `MyFTPServer2` に変更します。

```
data source ftp update --n NorthernRegionCustomers --h
MyFTPServer2
```

SFTP

data source sftp add

`data source sftp add` コマンドは、Spectrum™ Technology Platform と SFTP サーバーの間の接続を作成します。

使用方法

`data source sftp add --n` *ConnectionName* `--h` *Host* `--o` *Port* `--s` *strictHostCheck* `--u` *Username* `--a` *key-based authentication* `--k` *privateKeyFile* `--e` *passphrase* `--f` *knownHostFile*

必須	引数	説明
はい	`--n` *ConnectionName*	接続の名前を指定します。任意の名前にすることができます。
はい	`--h` *Host*	SFTP サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	SFTP サーバーとの通信で使うネットワーク ポートを指定します。デフォルト値は 22 です。
いいえ	`--s` *strictHostCheck*	厳密なホスト キー チェックを有効にする場合に指定します。デフォルトは "false" です。
いいえ	`--u` *Username*	SFTP サーバーへの接続で使うユーザ名 (必要な場合)。
いいえ	`--a` *key-based authentication*	認証タイプとしてパスワードまたはキーベースを指定します。デフォルトはパスワードです。
いいえ	`--k` *privateKeyFile*	秘密鍵ファイルのパスを指定します。
いいえ	`--e` *passphrase*	秘密鍵の生成に使うパスフレーズのセットを指定します。
いいえ	`--f` *knownHostFile*	既知のホストの詳細を保管するファイルの場所を指定します。
いいえ	`--p` *Password*	SFTP サーバーへの接続で使うパスワード (必要な場合)。

例

この例では、`key-based` を指定し、MyFTPServer という SFTP サーバーへの接続を作成します。

```
data source sftp add --n NorthernRegionCustomers --h
MySFTPServer --o 22 --u ExampleUserName --a Key-Based --k
ExampleKeyFile --e ExamplePassphrase
```

例 2:

この例では、**[認証タイプ]** としてパスワードを使用し、MyFTPServer という SFTP サーバーへの接続を作成します。

```
data source sftp add --n NorthernRegionCustomers --h
MySFTPServer --o 22 --u ExampleUserName --a Password --p
Example123
```

data source sftp delete

`data source sftp delete` コマンドは、Spectrum™ Technology Platform と SFTP サーバーの間の接続を削除します。

使用方法

`data source sftp delete--n`*ConnectionName*

必須	引数	説明
はい	`--n` *ConnectionName*	削除する接続の名前を指定します。接続のリストを表示するには、`data source sftp list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続を削除します。

```
data source sftp delete --n NorthernRegionCustomers
```

data source sftp list

`data source sftp list` コマンドは、Spectrum™ Technology Platform サーバー上で定義されている SFTP 接続のリストを返します。

使用方法

```
data source sftp list
```

data source sftp test

`data source sftp test` コマンドは、Spectrum™ Technology Platform と SFTP サーバーの間の接続をテストします。

使用方法

`data source sftp test --n` *ConnectionName*

必須	引数	説明
はい	`--n` *ConnectionName*	テストする接続の名前を指定します。接続のリストを表示するには、`data source sftp list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続をテストしています。

```
data source sftp test --n NorthernRegionCustomers
```

data source sftp update

`data source sftp update` コマンドは、Spectrum™ Technology Platform と SFTP サーバーの間の接続を変更します。

使用方法

`data source sftp update --n` *ConnectionName* `--h` *Host* `--o` *Port* `--s` *strictHostCheck* `--u` *Username* `--a` *key-based authentication* `--k` *privateKeyFile* `--e` *passphrase* `--f` *knownHostFile*

必須	引数	説明
はい	`--n` *ConnectionName*	接続の名前を指定します。任意の名前にすることができます。
はい	`--h` *Host*	SFTP サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	SFTP サーバーとの通信で使うネットワークポートを指定します。デフォルト値は 22 です。
いいえ	`--s` *strictHostCheck*	厳密なホスト キー チェックを有効にする場合に指定します。デフォルトは "false" です。
いいえ	`--u` *Username*	SFTP サーバーへの接続で使うユーザ名 (必要な場合)。

必須	引数	説明
いいえ	`--a` *key-based authentication*	認証タイプとしてパスワードまたはキーベースを指定します。デフォルトはパスワードです。
いいえ	`--k` *privateKeyFile*	秘密鍵ファイルのパスを指定します。
いいえ	`--e` *passphrase*	秘密鍵の生成に使うパスフレーズのセットを指定します。
いいえ	`--f` *knownHostFile*	既知のホストの詳細を保管するファイルの場所を指定します。
いいえ	`--p` *Password*	SFTP サーバーへの接続で使うパスワード (必要な場合)。

例

この例では、SFTP 接続 `NorthernRegionCustomers` に変更を加えます。ホストを `HostServer2` に変更します。

```
data source sftp update --n ConnectionName --h HostServer2 --o
Port --s strictHostCheck --u Username --a key-based
authentication --k privateKeyFile --e passphrase --f
knownHostFile
```

JDBC データベース

接続

dbconnection add

`dbconnection add` コマンドは、Spectrum™ Technology Platform とデータベースとの間の接続を作成します。

使用方法

`dbconnection add --n` *ConnectionName* `--d` *Driver* `--h` *Host* `--o` *Port* `--i` *Instance* `--u` *Username* `--p` *Password* `--l` "*property*:*value*"

必須	引数	説明
はい	`--n` *ConnectionName*	接続の名前を指定します。任意の名前にすることができます。

必須	引数	説明
はい	`--d` *Driver*	接続するデータベースのタイプに対応したドライバを指定します。サーバーで使用可能なデータベース ドライバのリストを表示するには、`dbdriver list` コマンドを使用します。
はい	`--h` *Host*	データベース サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	データベース サーバーとの通信で使用するネットワーク ポートを指定します。
いいえ	`--i` *Instance*	接続するデータベース インスタンスを指定します。
いいえ	`--u` *Username*	データベースへの接続に使用するユーザ名 (必要な場合)。
いいえ	`--p` *Password*	データベースへの接続に使用するパスワード (必要な場合)。
いいえ	`--l` "*property*:*value*"	ドライバに対する接続プロパティと値のペアのカンマ区切りリストを指定します。ドライバに対する有効なプロパティの一覧を表示するには、Management Console を開いて **[リソース] > [データ ソース]** に移動し、**[ドライバ]** タブをクリックします。必要なドライバを選択してから編集ボタン をクリックすると、その接続プロパティが表示されます。

例

この例では、ホスト MyServer に配置されているデータベースへの接続を作成します。接続名は NorthernRegionCustomers です。ドライバとして ExampleSQLDriver を使用します。このドライバには 2 つの接続プロパティがあり、ExampleProp1 には値 123、ExampleProp2 には値 456 が指定されています。

```
dbconnection add --n NorthernRegionCustomers --d
ExampleSQLDriver --h MyServer --l
"ExampleProp1:123,ExampleProp2:456"
```

dbconnection delete

`dbconnection delete` コマンドは、Spectrum™ Technology Platform とデータベースとの間の接続を削除します。

使用方法

`dbconnection delete--n`*ConnectionName*

必須	引数	説明
はい	`--n` *ConnectionName*	削除する接続の名前を指定します。接続のリストを表示するには、`dbconnection list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続を削除します。

```
dbconnection delete --n NorthernRegionCustomers
```

dbconnection export

`dbconnection export` コマンドは、データベース接続定義を JSON ファイルにエクスポートします。

使用方法

`dbconnection export --n` *ConnectionName* `--o` *OutputDirectory*

必須	引数	説明
はい	`--n` *ConnectionName*	エクスポートするデータベース接続の名前を指定します。接続名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 接続名が正確にわからない場合は、`dbbconnection list` コマンドを使用して接続名のリストを取得できます。
いいえ	`--o` *OutputDirectory*	データベース接続のエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、データベース接続は管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例は、"My Connection" というデータベース接続を、管理ユーティリティのインストール場所にある `exported` というサブフォルダにエクスポートします。

```
dbconnection export --n "My Connection" --o exported
```

dbconnection import

`dbconnection import` コマンドは、データベース接続定義ファイルをサーバーにインポートします。`dbconnection export` コマンドを使用してサーバーからデータベース接続をエクスポートすると、データベース接続定義ファイルが作成されます。インポートできるデータベース接続は、同じバージョンの Spectrum™ Technology Platform からエクスポートされたもののみです。

使用方法

```
dbconnection import--fDatabaseConnectionFile--uTrueOrFalse
```

必須	引数	説明
はい	`--f` *DatabaseConnectionFile*	インポートするデータベース接続ファイルを指定します。接続ファイルには `.json` という拡張子が付けられています。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *TrueOrFalse*	同じ名前のデータベース接続が既にサーバーに存在する場合にそのデータベース接続を上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true** インポートするデータベース接続と同じ名前のデータベース接続がサーバー上に存在する場合、サーバー上のドライバは上書きされます。これがデフォルト設定です。 **false** インポートするデータベース接続と同じ名前のデータベース接続がサーバー上に存在する場合も、接続のインポートは行われません。

例

この例は、`MyDatabaseConnection.json` というデータベース接続定義をインポートします。このファイルは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。

```
dbconnection import --f exported\MyDatabaseConnection.json
```

dbconnection list

`dbconnection list` コマンドは、Spectrum™ Technology Platform サーバー上で定義されるデータベース接続のリストを返します。

使用方法

```
dbconnection list
```

dbconnection test

`dbconnection test` コマンドは、Spectrum™ Technology Platform とデータベースとの間の接続をテストします。

使用方法

```
dbconnection test--nConnectionName
```

必須	引数	説明
はい	`--n` *ConnectionName*	テストする接続の名前を指定します。接続のリストを表示するには、`dbconnection list` コマンドを使用します。

例

この例では、NorthernRegionCustomers という接続をテストしています。

```
dbconnection test --n NorthernRegionCustomers
```

dbconnection update

`dbconnection update` コマンドは、Spectrum™ Technology Platform とデータベースとの間の接続を変更します。

使用方法

```
dbconnection
update--nConnectionName--dDriver--hHost--oPort--iInstance--uUsername--pPassword--l
"property:value"
```

必須	引数	説明
はい	`--n` *ConnectionName*	変更する接続の名前を指定します。接続のリストを表示するには、`dbconnection list` コマンドを使用します。
はい	`--d` *Driver*	接続するデータベースのタイプに対応したドライバを指定します。サーバーで使用可能なデータベース ドライバのリストを表示するには、`dbdriver list` コマンドを使用します。
はい	`--h` *Host*	データベース サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	データベース サーバーとの通信で使用するネットワーク ポートを指定します。
いいえ	`--i` *Instance*	接続するデータベース インスタンスを指定します。
いいえ	`--u` *Username*	データベースへの接続に使用するユーザ名 (必要な場合)。
いいえ	`--p` *Password*	データベースへの接続に使用するパスワード (必要な場合)。
いいえ	`--l` "*property:value*"	ドライバに対する接続プロパティと値のペアのカンマ区切りリストを指定します。ドライバに対する有効なプロパティの一覧を表示するには、Management Console を開いて **[リ**

必須	引数	説明
		ソース] > [データ ソース] に移動し、[ドライバ] タブをクリックします。必要なドライバを選択してから編集ボタン をクリックすると、その接続プロパティが表示されます。

例

この例では、NorthernRegionCustomers というデータベース接続を変更します。ドライバを `MSSQLServer2` に変更し、ホストを `MyServer2` に変更し、インスタンスを `MyInstance2` に変更します。

```
dbconnection update --n NorthernRegionCustomers --d MSSQLServer2
--h MyServer2 --i MyInstance2
```

ドライバ

dbdriver delete

`dbdriver delete` コマンドは、JDBC ドライバ定義を削除します。ドライバ ファイルは削除せず、Management Console で作成された定義のみを削除します。

使用方法

`dbdriver delete --n` *DriverName*

必須	引数	説明
はい	`--n` *DriverName*	削除する JDBC ドライバの名前を指定します。ドライバのリストを表示するには、`dbdriver list` コマンドを使用します。

例

この例では、MyDriver という JDBC ドライバを削除します。

```
dbconnection delete --n MyDriver
```

dbdriver export

`dbdriver export` コマンドは、JDBC ドライバ定義を JSON ファイルにエクスポートします。ドライバ ファイルはエクスポートせず、Management Console で作成されたドライバ定義のみをエクスポートします。

使用方法

`dbdriver export--n`*DriverName*`--o`*OutputDirectory*

必須	引数	説明
はい	`--n` *DriverName*	エクスポートするデータベース ドライバの名前を指定します。ドライバ名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** ドライバ名が正確にわからない場合は、`dbdriver list` コマンドを使用してドライバ名のリストを取得できます。
いいえ	`--o` *OutputDirectory*	データベース ドライバのエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、データベース ドライバは管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例は、"My Driver" というデータベース ドライバを、管理ユーティリティのインストール場所にある `exported` というサブフォルダにエクスポートします。

```
dbdriver export --n "My Driver" --o exported
```

dbdriver import

`dbdriver import` コマンドは、JDBC データベース ドライバ定義ファイルをサーバーにインポートします。`dbdriver export` コマンドを使用してサーバーからデータベース ドライバ定義をエクスポートすると、データベース ドライバ定義ファイルが作成されます。インポートできるデータベース ドライバ定義は、同じバージョンの Spectrum™ Technology Platform からエクスポートされたもののみです。

使用方法

`dbdriver import --f` *DriverDefinitionFile* `--u` *TrueOrFalse*

必須	引数	説明
はい	`--f` *DriverDefinitionFile*	インポートするデータベース ドライバ JSON ファイルを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *TrueOrFalse*	同じ名前のデータベース ドライバが既にサーバー上に存在する場合、既存のデータベース ドライバ定義を上書きするかどうかを指定します。ここで、*TrueOrFalse* は次のいずれかです。 **true** インポートするデータベース ドライバと同じ名前のデータベース ドライバがサーバー上に存在する場合、サーバー上のドライバは

必須	引数	説明
		上書きされます。これがデフォルト設定です。
	false	インポートするデータベース ドライバと同じ名前のデータベース ドライバがサーバー上に存在する場合も、ドライバのインポートは行われません。

例

この例は、`MyDatabaseDriver.json` というデータベース ドライバ定義をインポートします。このファイルは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。

```
dbdriver import --f exported\MyDatabaseDriver.json
```

dbdriver list

`dbdriver list` コマンドは、Spectrum™ Technology Platform サーバー上で定義されるデータベース ドライバのリストを返します。

使用方法

```
dbdriver list
```

データフロー

dataflow delete

`dataflow delete` コマンドは、データフローをシステムから削除します。

使用方法

`dataflow delete --d` ***DataflowName***

必須	引数	説明
はい	`--d` *DataflowName*	削除するデータフローを指定します。データフローの名前にスペースが含まれる場合は、データフロー名を引用符で囲みます。

例

この例では、My Dataflow という名前のデータフローを削除します。

```
dataflow delete --d "My Dataflow"
```

dataflow export

`dataflow export` コマンドは、データフローをサーバーから `.df` ファイルにエクスポートします。このデータフローは、さらに別のサーバーにインポートできます。

注： データフローの交換は、同じバージョンの Spectrum™ Technology Platform の間でのみ行うことができます。

使用方法

`dataflow export --d` *DataflowName* `--e` *TrueOrFalse* `--o` *OutputDirectory*

必須	引数	説明
はい	`--d` *DataflowName*	エクスポートするデータフローの名前を指定します。データフローの名前にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。
はい	`--e` *TrueOrFalse*	データフローの公開済みバージョンをエクスポートするかどうかを指定します。ここで、*TrueOrFalse* は次のいずれかです。 **true** データフローの公開済みバージョンをエクスポートします。 **false** データフローの最新の保存済みバージョンをエクスポートします。

必須	引数	説明
いいえ	`--o` *OutputDirectory*	データフローのエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、データフローは管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例では、"My Dataflow" という名前のデータフローの公開済みバージョンを、管理ユーティリティがインストールされたディレクトリの下にある `exported` という名前のサブフォルダにエクスポートします。

```
dataflow export --d "My Dataflow" --e true --o exported
```

dataflow expose

`dataflow expose` コマンドは、データフローを実行できる状態にします。サービス データフローの場合、データフローを公開すると、そのサービスは Web サービス要求や API 呼び出しを受け付けるようになり、ログ レベルの設定が可能になります。サブフローの場合は、データフローを公開すると、データフローでの使用が可能になります。ジョブ データフローの場合は、データフローを公開すると、 Job Executor コマンドライン ツールによるジョブの実行が可能になります。プロセス フローを公開するには、 `processflow expose` コマンドを使用します。

> **注：** Enterprise Designer でデータフロー ビジョニングを使用する場合、`dataflow expose` コマンドはデータフローの最新の保存済みバージョンを公開します。

使用方法

`dataflow expose--d`*DataflowName*

必須	引数	説明
はい	`--d` *DataflowName*	公開するデータフローの名前を指定します。データフローの名前にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。

例

この例では、My Dataflow という名前のデータフローを公開します。

```
dataflow expose --d "My Dataflow"
```

dataflow import

`dataflow import` コマンドは、データフロー ファイル (`.df` ファイル) をサーバにインポートします。データフロー ファイルは、`dataflow export` コマンドを使用してデータフローをサーバーからエクスポートすると作成されます。

使用方法

`dataflow import --f` *DataflowFile* `--u` *TrueOrFalse* `--p` *Path* `--c` *TrueOrFalse*

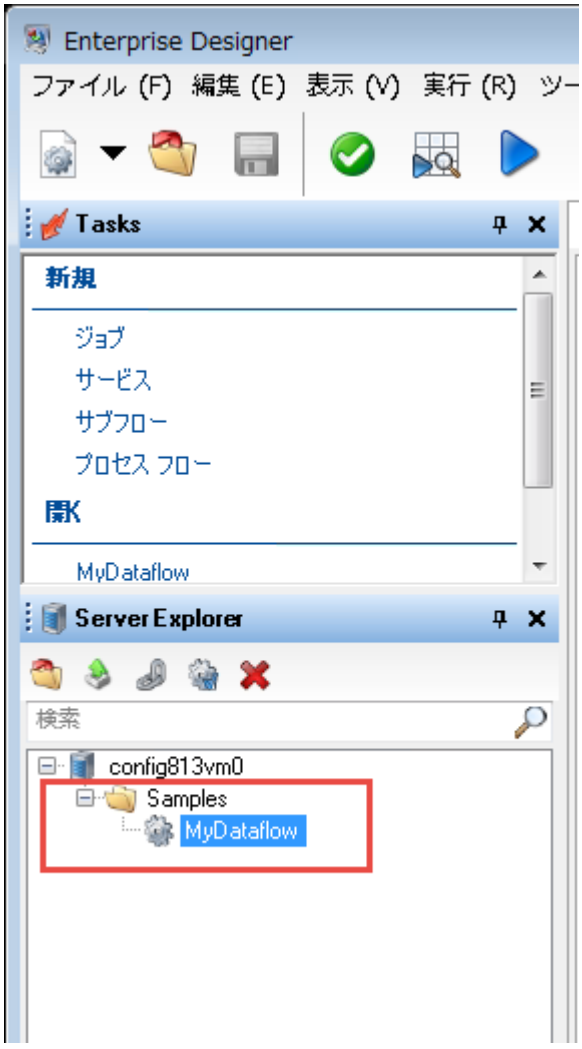
必須	引数	説明
はい	`--f` *DataflowFile*	インポートするデータフロー ファイル (`.df` ファイル) を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。絶体パスを使用することもできます。
いいえ	`--u` *TrueOrFalse*	同じ名前のデータフローがすでにサーバーに存在する場合にそのデータフローを上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true** インポートするデータフローと同じ名前のデータフローがサーバーに存在する場合は、サーバーにあるデータフローを上書きします。これがデフォルト設定です。 **false** インポートするデータフローと同じ名前のデータフローがサーバーに存在する場合は、データフローをインポートしません。
いいえ	`--p` *Path*	このフローのインポート先である Enterprise Designer サーバーのエクスプローラ フォルダを指定します。
いいえ	`--c` *TrueOrFalse*	存在しない場合に、`--p` で指定されたフォルダを作成するかどうかを指定します。 **true** 存在しない場合に、`--p` で指定されたフォルダを作成します。デフォルト

必須	引数	説明
	false	存在しない場合、--p で指定されたフォルダを作成しません。--p で指定されたフォルダが存在しなければ、フローはインポートされません。

例

この例では、MyDataflow.df という名前のデータフローを、管理ユーティリティを実行しているディレクトリの下にある exported という名前のサブフォルダにインポートします。データフローは Enterprise Designer の Samples フォルダにインポートされます。

```
dataflow import --f exported\MyDataflow.df --p Samples
```



dataflow list

`dataflow list` コマンドは、サーバーにあるすべてのデータフローを一覧に表示します。各データフローに関連する情報として、データフロー名、データフローの種類(ジョブ、サービス、またはサブフロー)、データフローがエクスポーズされているかどうか、などが表示されます。

使用方法

```
dataflow list
```

dataflow lock list

`dataflow lock list` コマンドは、ユーザが編集しているときにロックされるデータフローを一覧表示します。ユーザが Enterprise Designer でデータフローを開くとそのデータフローはロックされ、データフローを閉じるとロックは解除されます。

使用方法

```
dataflow lock list
```

dataflow sourcesink list

`dataflow sourcesink list` コマンドは、データフローの入力を指定するステージとデータフローからの出力を指定するステージをデータフローから特定して一覧表示します。

使用方法

`dataflow sourcesink list --d` *DataflowName* `--e` *TrueOrFalse* `--o` *TrueOrFalse*

必須	引数	説明
はい	`--d` *DataflowName*	ソースとシンクを一覧表示するデータフローの名前を指定します。データフローの名前にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。

必須	引数	説明
はい	`--e` *TrueOrFalse*	ソースとシンクを、データフローのエクスポーズバージョンと最新保存バージョンのどちらから一覧表示するかを指定します。 **true** ソースとシンクをデータフローのエクスポーズバージョンから一覧表示します。 **false** ソースとシンクをデータフローの最新保存バージョンから一覧表示します。
いいえ	`--o` *TrueOrFalse*	実行時にファイルをオーバーライドできるソースとシンクのみを一覧表示するかどうかを指定します。ファイルのオーバーライドとは、ステージで指定された読み取りファイルまたは書き込みファイルを実行時に別のファイルで差し替えることを意味します。Read from File や Write to File などのステージが、ファイルのオーバーライドに対応しています。Write to Null や Terminator などのステージは、ファイルのオーバーライドに対応していません。 **true** ファイルのオーバーライドに対応しているステージのみを一覧表示します。 **false** すべてのソースとシンクを一覧表示します。これがデフォルト設定です。

例

この例は、"My Dataflow" というデータフローのエクスポーズ バージョンからソースとシンクを一覧表示します。ファイルのオーバーライドに対応していないものを含め、すべてのソースとシンクを一覧表示します。

```
dataflow sourcesink list --d "My Dataflow" --e true
```

dataflow unexpose

`dataflow unexpose` コマンドは、データフローをサービスまたはジョブとして実行できないように設定します。

使用方法

`dataflow unexpose--d`*DataflowName*

必須	引数	説明
はい	`--d` *DataflowName*	公開を解除するデータフローの名前を指定します。データフローの名前にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。

例

この例では、My Dataflow という名前のデータフローの公開を解除します。

```
dataflow unexpose --d "My Dataflow"
```

dataflow unlock

`dataflow unlock` コマンドは、データフローのロックを解除し、他のユーザが Enterprise Designer でそのデータフローを編集できるようにします。通常は、ユーザが Enterprise Designer でデータフローを閉じたときに、データフローのロックは自動的に解除されます。一部の状況において、管理者が `dataflow unlock` コマンドを使用してデータフローのロックを解除しなければならない場合があります。例えば、あるユーザが Enterprise Designer でデータフローを開き、1 日中開いたままにした場合、そのデータフローはロックされたままとなり、他のユーザはそれを編集できません。そのような場合に `dataflow unlock` コマンドを使用して、データフローのロックを解除することができます。データフローのロックが解除されたら、Enterprise Designer のユーザはそのデータフローをいったん閉じて開き直さないと保存できません。

`dataflow unlock` コマンドを使用するには、**[データフロー - ロック解除]** 権限が必要です。

警告： データフローのロックを解除すると、そのデータフローをロックしていたユーザは、未保存の変更内容を保存できなくなります。

使用方法

`dataflow unlock--d`*DataflowName*

必須	引数	説明
はい	`--d` *DataflowName*	ロックを解除するデータフローを指定します。データフローの名前にスペースが含まれる場合は、データフロー名を引用符で囲みます。

dataflow version list

`dataflow version list` コマンドは、特定のデータフローの利用可能なバージョンをすべて表示します。`--n` コマンド パラメータを使用して、データフロー名を指定します。作成されたデータフローは、削除するまで Spectrum によって保持され、それぞれに保存バージョン (1.0.0、1.0.1 など) が適用されます。

使用方法

`dataflow version list--n`*DataflowName*

必須	引数	説明
はい	`--n` *DataflowName*	バージョンを一覧表示するデータフローの名前を指定します。データフロー名にスペースが含まれる場合は、名前を二重引用符で囲みます。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。

エンティティ

エンティティ セキュリティ オーバーライドのエクスポート

役割またはユーザのセキュリティ オーバーライドを JSON 形式でエクスポートするには、管理ユーティリティのコマンド ライン インターフェイス (CLI) を使用します。

使用方法

`entity override export --e` *role_or_user_value* `--p` *role_or_user_literal*

必須	引数	説明
はい	`--e` *role_or_user_value*	オーバーライドをエクスポートする役割またはユーザの値 (userName/RoleName) を指定します。
はい	`--p` *role_user_literal*	"役割" または "ユーザ" のリテラル名を指定します。

例

"Sally" というユーザのオーバーライドをエクスポートするには、次の構文を使用します。

```
entity override export --e user --p Sally
```

役割 "designer" を持つ "Sally" のオーバーライドをエクスポートするには、次の構文を使用します。

```
entity override export --e Sally --p designer
```

フォルダ

folder browse

`folder browse` コマンドは、Server Explorer フォルダの内容を一覧表示します。

使用方法

`folder browse--p`*Path*

必須	引数	説明
いいえ	`--p` *Path*	内容を一覧で表示したいフォルダのパスを指定します。このパラメータを省略すると、ルート フォルダの内容が一覧表示されます。

folder create

`folder create` コマンドは、フォルダを Server Explorer 内に作成します。

使用方法

`folder create--p`*Path*

必須	引数	説明
はい	`--p` *Path*	作成するフォルダのパスを指定します。

例

この例では、フォルダ `Example123` をフォルダ `ExampleABC` の内部に作成します。

```
folder create --p ExampleABC/Example123
```

folder delete

`folder delete` コマンドは、フォルダを Server Explorer から削除します。

使用方法

`folder delete--p`*Path*`--r`*TrueFalse*

必須	引数	説明
はい	`--p` *Path*	削除するフォルダのパスを指定します。
いいえ	`--r` *TrueFalse*	そこにフローまたはサブフォルダがある場合にフォルダを削除するかどうかを指定します。 **true** フローまたはサブフォルダがあってもフォルダを削除します。 **false** フローまたはサブフォルダがある場合、フォルダを削除しません。デフォルト

例

この例では、フォルダ `Example123` を削除します。そこにフローまたはサブフォルダが含まれていてもフォルダは削除されます。

```
folder delete --p ExampleABC/Example123 --r true
```

folder move

`folder move` コマンドは、Server Explorer 内のフォルダを別の場所に移動します。

使用方法

```
folder move--pPath--tTarget
```

必須	引数	説明
はい	`--p` *Path*	移動するフォルダのパスを指定します。
はい	`--t` *Target*	フォルダの移動先のパスを指定します。

例

この例では、フォルダ `ExampleABC` をフォルダ `Example123` 内に移動します。

```
folder move --p ExampleABC --t Example123
```

folder rename

`folder rename` コマンドは、Server Explorer 内のフォルダの名前を変更します。

使用方法

```
folder rename--pPath--nNewName
```

必須	引数	説明
はい	`--p` *Path*	名前を変更するフォルダのパスを指定します。
はい	`--n` *NewName*	フォルダの新しい名前を指定します。

Information Extraction モジュール

iemodel delete

`iemodel delete` コマンドは、Information Extraction モジュールのモデルの全一覧を返します。

使用方法

```
iemodel delete --n modelName
```

必須	引数	説明
はい	`--n` *modelName*	削除するモデルの名前を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。

例

この例では、"MyModel" というモデルを削除します。

```
iemodel delete --n MyModel
```

iemodel evaluate model

`iemodel evaluate` コマンドは、Information Extraction モジュールのトレーニング済みモデルを評価します。

使用方法

`iemodel evaluate model--n`*modelName*`--t`*testFileName*`--o`*outputFileName*`--c`*categoryCount*`--d`*trueOrfalse*

必須	引数	説明
はい	`--n` *modelName*	評価するモデルの名前と場所を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
はい	`--t` *testFileName*	モデルの評価に使用するテスト ファイルの名前と場所を指定します。
いいえ	`--o` *outputFileName*	評価結果を格納する出力ファイルの名前と場所を指定します。
いいえ	`--c` *categoryCount*	モデル内のカテゴリ数を指定します。数値を指定する必要があります。 **注 :** テキスト分類モデルのみに適用されます。
いいえ	`--d` *trueOrfalse*	テーブルをエンティティに関する詳細分析とともに表示するかどうか指定します。値は次の `true` または `false` にする必要があります。 **true** 詳細な評価結果が必要。 **false**

必須	引数	説明
		詳細な評価結果は必要ない。 デフォルトは `false` です。 *モデル評価の結果*テーブルと、その列を示す*混同行列*には、以下に示すように、各エンティティの数が表示されます。 **注：** この引数を指定しないか、この引数の値を `false` に指定してコマンドを実行すると、*モデル評価の結果*テーブルと*混同行列*は表示されません。*モデル評価の統計値*のみが表示されます。

出力

モデル評価の統計値

このコマンドを実行すると、テーブル形式で以下の評価の統計情報が表示されます。

- **適合率**: 厳密性を示す指標です。適合率は、正しく識別された組の割合を示します。
- **再現率**: 結果の完全性を示す指標です。再現率は、関連するインスタンスのうち、検出されたインスタンスの比率として定義できます。
- **F1 値**: テストの正確性を示す指標です。F1 スコアの計算では、テストの適合率と再現率の両方が考慮されます。これは適合率と再現率の加重平均として解釈でき、F1 スコアの最高値は 1、最低値は 0 になります。
- **正確度**: 結果の正確性を示す指標です。これは測定値と既知値の近さを示します。

モデル評価の結果

コマンドを引数 `--d true` を指定して実行すると、すべてのエンティティのマッチ数がテーブル形式で表示されます。そのテーブルには次の列があります。

Input Count (入力数)	入力データ内のエンティティの発生数。
Mismatch Count (不一致数)	エンティティのマッチが失敗した回数。
Match Count (マッチ数)	エンティティのマッチに成功した回数。

混同行列

混同行列(以下を参照) では、アルゴリズムの性能を視覚化できます。分類モデルの性能を示します。

```
Confusion Matrix:
+------------------+-------+-----------+-----------+
| CONFUSION MATRIX |       | PREDICTED |           |
+------------------+-------+-----------+-----------+
|                  |       |  POSITIVE | NEGATIVE  |
|           ACTUAL |  TRUE |         3 |         0 |
|                  | FALSE |         0 |         0 |
+------------------+-------+-----------+-----------+
```

列は予測クラスのインスタンスを表し、行は実際クラスのインスタンスを表します。混同行列に関連する語として、次のようなものがあります。

実際	実際クラス内のエンティティの発生数。
予測	予測クラス内のエンティティの発生数。
TP	**真陽性 (True Positive):** 陽性 (positive) と予測され、実際にも真 (true) だったエンティティ発生数。
TN	**真陰性 (True Negative):** 陰性 (negative) と予測されたが、実際には真 (true) だったエンティティ発生数。
FP	**偽陽性 (False Positive):** 陽性 (positive) と予測されたが、実際には偽 (false) だったエンティティ発生数。
FN	**偽陰性 (False Negative):** 陰性 (negative) と予測され、実際にも偽 (false) だったエンティティ発生数。

例

この例の内容は次のとおりです。

- "MyModel" というモデルを評価
- 同じ場所にある "ModelTestFile" というテスト ファイルを使用
- "MyModelTestOutput" というファイルに評価の出力を格納
- カテゴリ数は 4
- 評価の詳細分析は必須

```
iemodel evaluate model --n MyModel --t
C:\Spectrum\IEModels\ModelTestFile --o
C:\Spectrum\IEModels\MyModelTestOutput --c 4 --d true
```

iemodel evaluate train_model

`iemodel evaluate train_model` コマンド は、**Information Extraction** モジュールの既存のモデルを評価およびトレーニングします。この機能は、新しいモデルには実行できません。

注： 既存の **Information Extraction** モジュールの評価とトレーニングでより良い結果を得るには、コマンド `iemodel trainAndevaluate model` を使用します。詳細については、**iemodel trainAndevaluate model**（364ページ）を参照してください。

使用方法

`iemodel evaluate train_model--f`*trainingOptionsFile*`--u`*trueOrFalse*`--o`*outputFileName*`--c`*categoryCount*`--d`*trueOrfalse*

必須	引数	説明
はい	`--f` *trainingOptionsFile*	モデルのトレーニングに使用するトレーニング オプション ファイルの名前と場所を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *overWriteIfExists*	既存のトレーニング済みモデルが存在する場合に、それを上書きするかどうかを指定します。*TrueOrFalse* は次のいずれかです。 **true** 既存のモデルを上書きします。 **false** 既存のモデルを上書きしません。
いいえ	`--o` *outputFileName*	評価結果を格納する出力ファイルの名前と場所を指定します。
いいえ	`--c` *categoryCount*	モデル内のカテゴリ数を指定します。数値を指定する必要があります。 **注：** テキスト分類モデルのみに適用されます。
いいえ	`--d` *trueOrfalse*	テーブルをエンティティに関する詳細分析とともに表示するかどうか指定します。値は次の `true` または `false` にする必要があります。 **true** 詳細な評価結果が必要。 **false** 詳細な評価結果は必要ない。 デフォルトは `false` です。 *モデル評価の結果*テーブル、および下記で説明されているその列は、エンティティごとのカウントを表示します。 **注：** このコマンドを引数なしに実行するか、引数値を `false` で実行すると、モデル評価の結果テーブルは表示されません。モデル評価の統計値のみが表示されます。

出力

モデル評価の統計値

このコマンドを実行すると、テーブル形式で以下の評価の統計情報が表示されます。

- 適合率
- 再現率
- F1 値

モデル評価の結果

コマンドを引数 `--d true` を指定して実行すると、すべてのエンティティのマッチ数がテーブル形式で表示されます。そのテーブルには次の列があります。

Input Count (入力数)	入力データ内のエンティティの発生数。
Mismatch Count (不一致数)	エンティティのマッチが失敗した回数。
Match Count (マッチ数)	エンティティのマッチに成功した回数。

例

この例の内容は次のとおりです。

- "C:\Spectrum\IEModels" にある "ModelTrainingFile" というトレーニング オプション ファイルを使用
- 同じ名前の既存の出力ファイルがあれば上書き
- "MyModelTestOutput" というファイルに評価の出力を格納
- カテゴリ数は 4
- 評価の詳細分析は必須

```
iemodel evaluate train_model --f
C:\Spectrum\IEModels\ModelTrainingFile --u true --o
C:\Spectrum\IEModels\MyModelTestOutput --c 4 --d true
```

iemodel export

`iemodel export` コマンドは、Information Extraction モジュールのモデルとそのメタデータをエクスポートします。

使用方法

`iemodel export--n`*modelName*`--o`*outputDirectory*

必須	引数	説明
はい	`--n` *modelName*	エクスポートするモデルの名前を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
はい	`--o` *outputDirectory*	エクスポートしたモデルとそのメタデータを格納するフォルダの場所を指定します。

例

この例では、`MyModel` という名前のモデルをエクスポートします。出力は、"C:\Spectrum\IEModels\MyModelExport" にある "MyModelExport" というフォルダに配置します。

```
iemodel export --n MyModel --o
C:\Spectrum\IEModels\MyModelExport
```

iemodel import

`iemodel import` コマンドは、Information Extraction モジュールのモデルとそのメタデータをインポートします。

使用方法

`iemodel import--n`*modelName*`--o`*inputDirectory*`--u`*trueOrFalse*

必須	引数	説明
はい	`--n` *modelName*	インポートするモデルの名前を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
はい	`--o` *inputDirectory*	インポートしたモデルとそのメタデータを格納するフォルダの場所を指定します。
いいえ	`--u` *overWriteIfExists*	既存のモデルが存在する場合に、それを上書きするかどうかを指定します。*TrueOrFalse* は次のいずれかです。 **true** 既存のモデルを上書きします。 **false** 既存のモデルを上書きしません。

例

この例では、`MyModel` という名前のモデルをインポートします。モデルは、"C:\Spectrum\IEModels\MyModelExport" にある "MyModelExport" というフォルダに格納します。また、同じ名前の既存のモデルがあれば、それを上書きします。

```
iemodel import --n MyModel --o
C:\Spectrum\IEModels\MyModelExport --u true
```

iemodel list

`iemodel list` コマンド は、Information Extraction モジュールのモデルの全一覧を返します。

使用方法

```
iemodel list
```

例

この例は、モデルの全一覧を列挙します。

```
iemodel list
```

iemodel train

`iemodel train` コマンドは、Information Extraction モジュールのモデルをトレーニングします。入力ファイルの場所を示すトレーニング オプション ファイルを使用し、指定されたオプションを適用します。

使用方法

`iemodel train--f`*`trainingOptionsFile`*`--u`*`trueOrFalse`*

必須	引数	説明
はい	`--f` *trainingOptionsFile*	モデルのトレーニングに使用するトレーニング オプション ファイルの名前と場所を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *trueOrFalse*	同じ名前の既存のモデルが存在する場合に、それを上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true**

必須	引数	説明
		既存のモデルを上書きします。
		false 既存のモデルを上書きしません。

例

この例では、c: ドライブに格納されている *TrainingOptions.xml* ファイルに記されたモデルをトレーニングし、同じ名前の既存のモデルがあればそれを上書きします。

```
iemodel train --f c:/TrainingOptions.xml --u true
```

iemodel trainAndevaluate model

`iemodel trainAndevaluate model` コマンドは、新しいモデルと既存モデルを評価およびトレーニングします。既存モデルの場合は、コマンドの引数 --u で "true" を指定することにより、新しくトレーニングされたモデルで既存モデルを上書きする必要があります。

このコマンドは、トレーニング オプション ファイルを使用し、評価結果ファイルを出力するように指定されている場合は、そのファイルを出力します。

使用方法

`iemodel trainAndevaluate model --f`*trainingOptionsFile* `--u`*trueOrFalse* `--o`*outputFileName* `--c`*categoryCount* `--d`*trueOrfalse*

必須	引数	説明
はい	`--f` *trainingOptionsFile*	モデルのトレーニングに使用するトレーニング オプション ファイルの名前と場所を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *overWriteIfExists*	既存のトレーニング済みモデルが存在する場合に、それを上書きするかどうかを指定します。 **true** 既存のモデルを上書きします。 **false** 既存のモデルを上書きしません。
いいえ	`--o` *outputFileName*	評価結果を格納する出力ファイルの名前と場所を指定します。

必須	引数	説明
いいえ	`--c` *categoryCount*	モデル内のカテゴリ数を指定します。数値を指定する必要があります。 **注：** テキスト分類モデルのみに適用されます。
いいえ	`--d` *trueOrfalse*	テーブルをエンティティに関する詳細分析とともに表示するかどうか指定します。値は次の `true` または `false` にする必要があります。 **true** 詳細な評価結果が必要。 **false** 詳細な評価結果は必要ない。 デフォルトは `false` です。 *モデル評価の結果*テーブルと、その列を示す*混同行列*には、以下に示すように、各エンティティの数が表示されます。 **注：** この引数を指定しないか、この引数の値を `false` に指定してコマンドを実行すると、*モデル評価の結果*テーブルと*混同行列*は表示されません。*モデル評価の統計値*のみが表示されます。

出力

モデル評価の統計値

このコマンドを実行すると、テーブル形式で以下の評価の統計情報が表示されます。

- **適合率**: 厳密性を示す指標です。適合率は、正しく識別された組の割合を示します。
- **再現率**: 結果の完全性を示す指標です。再現率は、関連するインスタンスのうち、検出されたインスタンスの比率として定義できます。
- **F1 値**: テストの正確性を示す指標です。F1 スコアの計算では、テストの適合率と再現率の両方が考慮されます。これは適合率と再現率の加重平均として解釈でき、F1 スコアの最高値は 1、最低値は 0 になります。
- **正確度**: 結果の正確性を示す指標です。これは測定値と既知値の近さを示します。

モデル評価の結果

コマンドを引数 `--d true` を指定して実行すると、すべてのエンティティのマッチ数がテーブル形式で表示されます。そのテーブルには次の列があります。

Input Count (入力数) 入力データ内のエンティティの発生数。

Mismatch Count (不一致数) エンティティのマッチが失敗した回数。

Match Count (マッチ数) エンティティのマッチに成功した回数。

混同行列

混同行列(以下を参照) では、アルゴリズムの性能を視覚化できます。分類モデルの性能を示します。

```
Confusion Matrix:
+------------------+-------+-----------+-----------+
| CONFUSION MATRIX |       | PREDICTED |           |
+------------------+-------+-----------+-----------+
|                  |       |  POSITIVE | NEGATIVE  |
|           ACTUAL |  TRUE |         3 |         0 |
|                  | FALSE |         0 |         0 |
+------------------+-------+-----------+-----------+
```

列は予測クラスのインスタンスを表し、行は実際クラスのインスタンスを表します。混同行列に関連する語として、次のようなものがあります。

実際 実際クラス内のエンティティの発生数。

予測 予測クラス内のエンティティの発生数。

TP **真陽性 (True Positive):** 陽性 (positive) と予測され、実際にも真 (true) だったエンティティ発生数。

TN **真陰性 (True Negative):** 陰性 (negative) と予測されたが、実際には真 (true) だったエンティティ発生数。

FP **偽陽性 (False Positive):** 陽性 (positive) と予測されたが、実際には偽 (false) だったエンティティ発生数。

FN **偽陰性 (False Negative):** 陰性 (negative) と予測され、実際にも偽 (false) だったエンティティ発生数。

例

この例の内容は次のとおりです。

- "C:\Spectrum\IEModels" にある "ModelTrainingFile" というトレーニング オプション ファイルを使用
- 同じ名前の既存の出力ファイルがあれば上書き
- "MyModelTestOutput" というファイルに評価の出力を格納
- カテゴリ数は 4
- 評価の詳細分析は必須

```
iemodel trainAndevaluate model --f
C:\Spectrum\IEModels\ModelTrainingFile --u true --o
C:\Spectrum\IEModels\MyModelTestOutput --c 4 --d true
```

ジョブ

job history list

コマンド `job history list` は、ジョブの実行履歴を表示します。

使用方法

`job status list--j`*JobName*`--f`*FromDateTime*`--t`*ToDateTime*

必須	引数	説明
はい	`--j` *JobName*	履歴を取得するジョブの名前を指定します。ジョブ名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント:** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。
いいえ	`--f` *FromDateTime*	特定の日時の範囲で履歴を表示する場合に、範囲の開始日時を MM-dd-yyyy HH:mm:ss の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `12-31-2014 13:00:00` と指定します。 日時の範囲を指定すると、`--f` 引数で指定した日付以降で、`--t` 引数で指定した日付以前に実行が開始したジョブが履歴リストに含まれます。 この引数を省略すると、現在の日付に実行が開始されたジョブが履歴に含まれます。
いいえ	`--t` *ToDateTime*	特定の日時の範囲で履歴を表示する場合に、範囲の終了日時を MM-dd-yyyy HH:mm:ss の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `12-31-2014 13:00:00` と指定します。 日時の範囲を指定すると、`--f` 引数で指定した日付以降で、`--t` 引数で指定した日付以前に実行が開始したジョブが履歴リストに含まれます。

必須	引数	説明
		この引数を省略すると、`--f`引数で指定した日付以降に実行が開始されたすべてのジョブが履歴に含まれます。

例

この例は、"My Job" というジョブのステータスを取得します。

```
job history list --j "My Job"
```

job execute

`job execute`コマンドは、1つ以上のジョブを実行します。ジョブの実行後、ジョブ名とジョブ ID が次の形式で返されます。

<JobName=JobID>

使用方法

`job execute --j` *JobNames* `--f` *JobPropertyFile* `--i` *PollInterval* `--d` *ReportDelimiter* `--n` *NotificationEmails* `--o` *OptionPropertyFile* `--r` *ReportTrueOrFalse* `--t` *Timeout* `--w` *WaitTrueOrFalse* `--l` *FileOverrides* `--v` *VerboseTrueOrFalse*

必須	引数	説明
はい	`--j` *JobNames*	実行する1つ以上のジョブの名前を指定します。複数のジョブを指定する場合は、それぞれのジョブをカンマで区切ってください。ジョブがリストの順番に実行されます。ジョブ名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント:** 使用するデータフローの名前が正確にわからない場合は、`dataflow list`コマンドを使用してデータフロー名のリストを取得できます。
いいえ	`--f` *JobPropertyFile*	ジョブ プロパティ ファイルへのパスを指定します。ジョブ プロパティ ファイルには、ジョブの実行を制御する引数が含まれています。詳細については、「ジョブ プロパティ ファイルの使用（214ページ）」を参照してください。
いいえ	`--i` *PollInterval*	`--w`を`true`に設定した場合、完了したジョブを確認する間隔をこのオプションで秒単位で指定します。デフォルトは5です。

必須	引数	説明
いいえ	`--d` *ReportDelimiter*	`--w true` または `--r true` も指定した場合、レポート出力で使用する区切り文字を設定します。デフォルトは、パイプ記号 (|) です。
いいえ	`--n` *NotificationEmails*	電子メール アドレスを 1 つ以上指定します。このメール アドレスによって、Management Console で設定されている、ジョブのステータスに関する通知を受信します。各電子メールアドレスの区切りにはカンマを使用します。

いいえ	`--o` *OptionPropertyFile*	フロー オプション プロパティ ファイルのパスを指定します。フロー オプション プロパティ ファイルを使用すると、フローのステージ オプションを設定できます。プロパティ ファイルを使用してフロー オプションを設定するには、実行時にステージオプションをエクスポーズするようにフローを構成する必要があります。詳細については、「**フロー実行時オプションの追加**（225ページ）」を参照してください。
		
		以下に、Assign GeoTAX Info ステージを含むフローのフロー オプション プロパティ ファイルの例を示します。
		
		`OutputCasing=U`
		`UseStreetLevelMatching=N`
		`TaxKey=T`
		`Database.GTX=gsl`

いいえ	`--r` *ReportTrueOrFalse*	ジョブについての詳細レポートを返すには、`true` を指定します。このオプションは、`--w true` も指定している場合にのみ機能します。レポートには、次の情報が含まれます。 • **位置 1** - ジョブの名前 • **位置 2** - ジョブ プロセス ID • **位置 3** - ステータス • **位置 4** - 開始日時 (MM/DD/YYYY HH:MM:SS) • **位置 5** - 終了日時 (MM/DD/YYYY HH:MM:SS) • **位置 6** - 成功したレコードの数 • **位置 7** - 失敗したレコードの数 • **位置 8** - 形式に誤りのあるレコードの数 • **位置 9** - 現在使用されていない 例を次に示します。 `MySimpleJob|4|succeeded|04/09/2010 14:50:47|04/09/2010 14:50:47|100|0|0|`

必須	引数	説明
		情報は、`--d`引数で指定された区切り文字で区切られて出力されます。
いいえ	`--t` *Timeout*	同期モードのタイムアウト値を秒単位で設定します。デフォルト値は 3600 です。
いいえ	`--w` *WaitTrueOrFalse*	同期モードでジョブを 1 つずつ実行するには、`true` を指定します。すべてのジョブを同時に実行するには `false` を指定します。デフォルトは false です。
いいえ	`--l` *FileOverrides*	入力/出力ファイルおよびファイル形式をオーバーライドします。詳細については、ジョブ ファイルのオーバーライド（372ページ）とファイル形式のオーバーライド（375ページ）を参照してください。
いいえ	`--v` *VerboseTrueOrFalse*	ジョブを実行するために使用する引数に関する情報と、ジョブ実行に関するその他の詳細情報を返すには、`true` を指定します。

例

この例では、Example1 というジョブを実行します。カンマ区切りレポートが返されます。`--w true` を指定しているのは、実行中のジョブが 1 つだけでもレポートを返すように要求するためです。Read From File ステージで指定されている入力ファイルは、このステージで指定されているファイルから、`CandidateHomes2.csv` という名前の別のファイルに変更されます。詳細な出力も返されます。

```
job execute --j Example1 --w true --d "," --r true --l "Read
from
File=file://e:/SampleDataflows/DataFiles/DataFiles/CandidateHomes2.csv"
--v true
```

ジョブ プロパティ ファイルの使用

ジョブ プロパティ ファイルには、ジョブの実行を制御する引数が含まれており、Job Executor または管理ユーティリティを使用したジョブの実行を制御できます。引数を再使用する場合に、各引数をコマンド ラインで個別に指定するのではなく、単一の引数 (`-f`) を指定するには、ジョブ プロパティ ファイルを使用します。

プロパティ ファイルを作成するには、各行に 1 つの引数を含むテキスト ファイルを作成します。

```
d %
h spectrum.mydomain.com
i 30
```

```
j validateAddressJob1
u user
p password
s 8888
t 9999
w true
```

ジョブ プロパティ ファイルには次の引数を含めることができます。

必須	引数	説明
いいえ	`?`	使用方法を出力します。
いいえ	`d` *delimiter*	インスタンス/ステータス区切り文字を設定します。これは、同期出力にのみ表示されます。
いいえ	`e`	Spectrum™ Technology Platform サーバーとの通信にセキュアな HTTPS 接続を使用します。
いいえ	`h` *hostname*	Spectrum™ Technology Platform サーバーの名前または IP アドレスを指定します。
いいえ	`i` *pollinterval*	完了したジョブを確認する間隔を秒数で指定します。これは、同期モードにのみ適用されます。
はい	`j` *jobname*	実行するジョブをカンマで区切って指定します。ジョブ名は大文字と小文字が区別されます。ジョブは、列挙された順序で開始されます。
いいえ	`n` *emaillist*	ジョブ通知設定に対する追加の電子メール アドレスをカンマで区切って指定します。
はい	`p` *password*	ユーザのパスワードです。
いいえ	`r`	標準出力に出力されるジョブに関する以下の情報を、区切り文字で区切って返します。 • **位置 1** - ジョブの名前 • **位置 2** - ジョブ プロセス ID • **位置 3** - ステータス • **位置 4** - 開始日時 (MM/DD/YYYY HH:MM:SS) • **位置 5** - 終了日時 (MM/DD/YYYY HH:MM:SS) • **位置 6** - 成功したレコードの数 • **位置 7** - 失敗したレコードの数 • **位置 8** - 形式に誤りのあるレコードの数 • **位置 9** - 現在使用されていない

必須	引数	説明
		情報は、-d 引数で指定された区切り文字で区切られて出力されます。例: `MySimpleJob|4|succeeded|04/09/2019 14:50:47|04/09/2019 14:50:47|100|0|0|`
いいえ	`s` *port*	Spectrum™ Technology Platform サーバーが実行されているソケット (ポート)。デフォルト値は 8080 です。
いいえ	`t` *timeout*	同期モードの場合のタイムアウト (秒単位) を設定します。デフォルト値は 3600 です。最大数は 2147483 です。これは、すべてを合わせたグローバルなタイムアウト値で、spawn されたすべてのジョブが完了するまでの最大待機時間を表します。
はい	`u` *username*	ユーザのログイン名を指定します。
いいえ	`v`	詳細な出力を返します。
いいえ	`w`	同期モードにおいてジョブの完了を待つように指定します。

コマンドライン引数とプロパティ ファイルの両方の使用

コマンド ライン入力とプロパティ ファイル入力を組み合わせて使用することもできます。例:

```
java -jar jobexecutor.jar -f /dcg/job.properties -j job1
```

この場合、コマンドライン引数の方が、プロパティ ファイルで指定された引数よりも優先されます。上の例では、ジョブ job1 の方が、プロパティ ファイルで指定されたジョブよりも優先されることになります。

ジョブ ファイルのオーバーライド

Job Executor または管理ユーティリティを使用してコマンド ラインでジョブを実行する際に、フローのソース ステージ (Read from File など) で指定された入力ファイル、およびフローのシンク ステージ (Write to File など) で指定された出力ファイルをオーバーライドできます。

Job Executor でこれを行うには、Job Executor コマンドの終わりに次のコマンドを指定します。

```
StageName=Protocol:FileName
```

管理ユーティリティでは、`job execute` コマンドで次のように `--l` 引数を使用します。

```
--l StageName=Protocol:FileName
```

説明:

StageName

Enterprise Designer で、フローのステージのアイコンの下に表示されるステージラベル。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read from File` と指定します。

埋め込まれたフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたフローまたはサブフローの名前とピリオドを付けて、次のようにします。

`EmbeddedOrSubflowName.StageName`

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write to File
```

別の埋め込まれたフローの中にある埋め込まれたフロー内のステージを指定するには、親フローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded Dataflow 1.Embedded Dataflow 2.Write to File
```

Protocol

通信プロトコルには、次のいずれかのタイプを指定できます。

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```

と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注：** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生し

ました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー*`/`*ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

FileName

入力または出力として使用するファイルへのフル パス。

注: ファイルのパスには、スラッシュを使用する必要があります。バックスラッシュは使用できません。

複数のオーバーライドを指定する場合は、カンマで区切ってください。

ファイルのオーバーライドの例

この例では、TestJob というジョブを実行します。Write to File ステージで指定されたファイルに出力を書き込むのではなく、出力を `outputoverride.txt` に書き込みます。

```
job execute --j TestJob --l "Write to
File=file:/Users/me/outputoverride.txt"
```

ファイル形式のオーバーライド

Job Executor または管理ユーティリティを使用してジョブを実行する際に、フローの Read from File ステージおよび Write to File ステージで指定されたファイルのファイル レイアウト (またはスキーマ) をオーバーライドできます。

Job Executor でこれを行うには、Job Executor コマンド ライン コマンドの終わりに以下を指定します。

```
StageName:schema=Protocol:SchemaFile
```

管理ユーティリティでは、`job execute` コマンドで次のように `--l` 引数を使用します。

```
--l StageName:schema=Protocol:SchemaFile
```

説明:

StageName

Enterprise Designer で、フローのステージのアイコンの下に表示されるステージ ラベル。例えば、ステージのラベルが "Read from File" の場合は、ステージ名として `Read from File` と指定します。

埋め込まれたフローまたはサブフロー内のステージを指定するには、ステージ名の前に埋め込まれたフローまたはサブフローの名前とピリオドを付けて、次のようにします。

```
EmbeddedOrSubflowName.StageName
```

例えば、"Subflow1" という名前のサブフローの中の "Write to File" という名前のステージを指定するには、次のようにします。

```
Subflow1.Write to File
```

別の埋め込まれたフローの中にある埋め込まれたフロー内のステージを指定するには、親フローをそれぞれピリオドで区切って追加します。例えば、Embedded Dataflow 2 が Embedded Dataflow 1 の中にあり、Embedded Dataflow 2 内の Write to File ステージを指定する場合は、次のようにします。

```
Embedded Dataflow 1.Embedded Dataflow 2.Write to File
```

Protocol

通信プロトコル:

file ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータ上にある場合は、file プロトコルを使用します。例えば、Windows では、

```
"file:C:/myfile.txt"
```

と指定し、Unix または Linux では、

```
"file:/testfiles/myfile.txt"
```

と指定します

esclient ファイルがジョブを実行するコンピュータ上にあり、そのコンピュータが Spectrum™ Technology Platform サーバーを実行するコンピュータと異なる場合は、esclient プロトコルを使用します。次の形式を使用します。

`esclient:`*コンピュータ名/ファイルへのパス*

例を次に示します。

```
esclient:mycomputer/testfiles/myfile.txt
```

> **注：** ジョブをサーバー自体で実行している場合は、file プロトコルを使用しても esclient プロトコルを使用しても構いませんが、file プロトコルを使用した方がパフォーマンスが高くなる可能性があります。

Job Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*`/server/conf/spectrum-container.properties`。プロパティ `spectrum.runtime.hostname` にサーバーの IP アドレスを設定します。

esfile ファイルがファイル サーバー上にある場合は、esfile プロトコルを使用します。このファイル サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`esfile://`*ファイル サーバー/ファイルへのパス*

例を次に示します。

```
esfile://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された FTP ファイル サーバー リソースです。

webhdfs ファイルが Hadoop Distributed File Server 上にある場合は、webhdfs プロトコルを使用します。HDFS サーバーは Management Console でリソースとして定義する必要があります。次の形式を使用します。

`webhdfs://`*ファイル サーバー/ファイルへのパス*

例を次に示します。

```
webhdfs://myserver/testfiles/myfile.txt
```

ここで、myserver は、Management Console で定義された HDFS ファイル サーバー リソースです。

SchemaFile

使用するレイアウトを定義したファイルへのフル パス。

注: ファイルのパスには、スラッシュを使用する必要があります。バックスラッシュは使用できません。

スキーマ ファイルを作成するには、目的のレイアウトを Read from File または Write to File で定義し、**[エクスポート]** ボタンをクリックして、レイアウトを定義する XML ファイルを作成します。

注: Job Executor を使用する場合は、スキーマ ファイル内のフィールドのデータ タイプをオーバーライドできません。FieldSchema 要素の子である <Type> 要素の値は、フローの Read from File ステージまたは Write to File ステージで指定されたフィールドのタイプと一致している必要があります。

ファイル フォーマットのオーバーライドの例

この例では、TestJob というジョブを実行します。Write to File ステージで指定されたファイルに出力を書き込むのではなく、出力を `outputoverride.txt` に書き込みます。ジョブは、フロー内の Write to File ステージで指定されたファイル スキーマを使用するのではなく、`output-data.xml` に指定されたスキーマを使用します。

```
job execute --j TestJob --l "Write to
File=file:/Users/me/outputoverride.txt,Write to
File:schema=file:/Users/me/output-data.xml"
```

系統および影響分析

notes export

`notes export` コマンドは、系統および影響分析エンティティ メモを JSON ファイルにエクスポートします。エンティティ メモは、ユーザによって作成されたメモであり、系統および影響分析キャンバス上にあるエンティティの [プロパティ] ウィンドウに表示されます。

使用方法

`notes export--o`*`OutputDirectory`*

必須	引数	説明
いいえ	`--o` *OutputDirectory*	系統および影響分析メモのエクスポート先とするディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、データフローは管理ユーティリティと同じディレクトリにエクスポートされます。

notes import

`notes import` コマンドは、`notes export` コマンドによって作成されたファイルから系統および影響分析エンティティ メモをインポートします。既存のメモは上書きされます。

使用方法

`notes import--f`*NotesFile*

必須	引数	説明
はい	`--f` *NotesFile*	インポートする系統および影響分析エンティティ メモが含まれている JSON ファイルを指定します。

Machine Learning モジュール

binning delete

このコマンドは、ビニングをサーバーから削除します。

エクスポーズされているビニングは、削除できません。

使用方法

`binning delete` --binningname "*ビニング名*"

必須	引数	説明
はい	`--n` *BinningName*	サーバーから削除するビニングの名前を指定します。

例

この例では、Home Sequence という名前のビニングを削除します。

```
binning delete --n "Home Sequence"
```

binning expose

このコマンドは、Binning Lookup ステージで使用できるようにビニングを公開 (エクスポーズ) します。

ビニングをエクスポーズすると、検索できるようになり、削除することができなくなります。

使用方法

`binning expose --n` ***binningname***

必須	引数	説明
はい	`--n` ***binningname***	サーバーでエクスポーズするビニングの名前を指定します。

例

この例では、Home Sequence という名前のビニングをエクスポーズします。

```
binning expose --n "Home Sequence"
```

binning list

`binning list` コマンドは、サーバー上のすべてのビニングの一覧をアルファベット順で表示します。

使用方法

```
binning list
```

binning unexpose

このコマンドは、Binning Lookup ステージで使用できないようにビニングを公開解除 (アンエクスポーズ) します。

ビニングをアンエクスポーズすると、検索できなくなり、削除できるようになります。

使用方法

```
binning unexpose --n binningname
```

必須	引数	説明
はい	`--n` *binningname*	サーバーでアンエクスポーズするビニングの名前を指定します。

例

この例では、Home Sequence という名前のビニングをアンエクスポーズします。

```
binning unexpose --n "Home Sequence"
```

machinelearning model expose

このコマンドは、機械学習モデルを Java Model Scoring ステージで使用できるようにします。機械学習モデルをスコアリングで使用するためには、エクスポーズされていることが必要です。

使用方法

```
machinelearning model expose --m model
```

必須	引数	説明
はい	`--m` *model*	機械学習モデルを指定します。

例

この例では、ConsumerFraud という名前のモデルをエクスポーズします。

```
machinelearning model expose --m ConsumerFraud
```

machinelearning model unexpose

このコマンドは、機械学習モデルを Java Model Scoring ステージで使用できないようにします。アンエクスポーズされた機械学習モデルをスコアリングに使用することはできません。

使用方法

```
machinelearning model unexpose --m model
```

必須	引数	説明
はい	`--m` *model*	機械学習モデルを指定します。

例

この例では、ConsumerFraud という名前のモデルをアンエクスポーズします。

```
machinelearning model unexpose --m ConsumerFraud
```

マッチ ルール

matchrule delete

`matchrule delete` コマンドは、システムからマッチ ルールを削除します。詳細については、『*データ品質ガイド*』の「マッチング」セクションを参照してください。

使用方法

`matchrule delete` *matchRuleName*

必須	引数	説明
はい	`--d` *matchRuleName*	削除するマッチ ルールを指定します。

例

この例は、My Match Rule というマッチ ルールを削除します。

```
matchrule delete My Match Rule
```

matchrule export

`matchrule export` コマンドは、Enterprise Designer でマッチング ステージ (Interflow Match、Intraflow Match、Transactional Match) の 1 つを使用して作成されたマッチ ルールをエクスポー

トします。その後、このマッチ ルールは別のサーバーにインポートできます。マッチ ルールは *.mr* ファイルまたは *.json* ファイルとしてエクスポートできます。

詳細については、『*データ品質ガイド*』の「マッチング」セクションを参照してください。

使用方法

`matchrule export` ***matchRuleName*** `--o` ***OutputDirectory***

必須	引数	説明
はい	`--m` ***matchRuleName***	エクスポートするマッチ ルールの名前を指定します。 **ヒント：** マッチ ルールの名前が正確にわからない場合は、`matchrule list` コマンドを使用してマッチ ルール名のリストを取得できます。
いいえ	`--j` ***matchRuleName***	JSON ファイル形式でエクスポートするマッチ ルールの名前を指定します。 **ヒント：** マッチ ルールの名前が正確にわからない場合は、`matchrule list` コマンドを使用してマッチ ルール名のリストを取得できます。
いいえ	`--o` ***OutputDirectory***	マッチルールのエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、マッチ ルールは管理ユーティリティと同じディレクトリにエクスポートされます。

例: マッチ ルールのエクスポート

この例は、*mr* 形式の "My Match Rule" というマッチ ルールを、管理ユーティリティを実行している場所にある `export` というサブフォルダにエクスポートします。

```
matchrule export My Match Rule --o export
```

例: JSON 形式でのマッチ ルール エクスポート

この例は、*JSON* 形式の "My Match Rule" というマッチ ルールを、管理ユーティリティが含まれているディレクトリにエクスポートします。

```
matchrule export My Match Rule --j
```

matchrule import

`matchrule import` コマンドは、マッチ ルール ファイルをサーバーにインポートします。マッチ ルールは、Enterprise Designer でマッチング ステージ (Interflow Match、Intraflow Match、Transactional Match) の 1 つを使用して作成されます。詳細については、『*データ品質ガイド*』の「マッチング」セクションを参照してください。

使用方法

`matchrule import` *MatchRule*`--u` *TrueOrFalse*

必須	引数	説明
はい	`--f` *MatchRuleFile*	インポートするマッチ ルール ファイルを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *TrueOrFalse*	同じ名前のマッチ ルール ファイルが既にサーバー上に存在する場合にそのマッチ ルール ファイルを上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true** インポートするマッチ ルール ファイルと同じ名前のマッチ ルール ファイルがサーバー上に存在する場合、サーバー上のマッチ ルール ファイルは上書きされます。これがデフォルト設定です。 **false** インポートするマッチ ルール ファイルと同じ名前のマッチ ルール ファイルがサーバー上に存在する場合、マッチ ルール ファイルはインポートされません。

例

この例は、`MyMatchRule.mr` というマッチ ルールをインポートします。このマッチ ルールは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。--u コマンドが指定されていないため、サーバー上に同じ名前のマッチ ルール ファイルがあれば上書きされます。

```
matchrule import exported\MyMatchRule.mr
```

matchrule list

`matchrule list` コマンドは、サーバーにあるすべてのマッチ ルールを一覧表示します。マッチ ルールごとに、マッチ ルール名と、そのマッチ ルールがエクスポーズされているかどうかの情報が表示されます。詳細については、『*データ品質ガイド*』の「マッチング」セクションを参照してください。

使用方法

```
matchrule list
```

メタデータ接続

resourceconnection export

`resourceconnection export` コマンドは、Spectrum サーバーにあるリソース接続を JSON ファイルにエクスポートします。

使用方法

`resourceconnection export --n` *ConnectionName* `--o` *OutputDirectory*

必須	引数	説明
はい	`--n` *ConnectionName*	エクスポートするリソース接続の名前を指定します。接続名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** 接続名が正確にわからない場合は、`resourceconnection list` コマンドを使用して接続名のリストを取得できます。
はい	`--t` *ConnectionType*	接続タイプを指定します。MS SQL Server、Oracle、Salesforce、Apache Cassandra などの値を指定できます。
いいえ	`--o` *OutputDirectory*	接続のエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、接

必須	引数	説明
		続は管理ユーティリティと同じディレクトリにエクスポートされます。
いいえ	`--encrypt` *fileEncryption*	ファイル暗号化を有効にするか無効にするかを指定します。 **true** ファイル暗号化を有効にします。これがデフォルト設定です。 **false** ファイル暗号化を無効にします。

例

この例は、接続タイプが "SalesForce" の "FrameallSalesforce_bulkon" というリソース接続を `exported` というフォルダにエクスポートします。このフォルダは、管理ユーティリティをインストールした場所にあるサブフォルダです。

```
resourceconnection export --n "FrameallSalesforce_bulkon" --t
SalesForce --o exported
```

resourceconnection import

`resourceconnection import` コマンドは、データベース接続を Spectrum サーバーにインポートします。

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に」を参照してください。

使用方法

`resourceconnection import --f` *connectionJsonFileName*

注： パラメータのリストを表示するには、"`help resourceconnection import`" と入力します。

必須	引数	説明
はい	`--f` *file*	インポートする接続の JSON ファイル名を指定します。
いいえ	`--u` *update*	同じ名前の接続が既にサーバーに存在する場合にその接続を上書きするかどうかを指定します。 **true**

必須	引数	説明
		インポートするリソースと同じ名前のリソースがサーバーに存在する場合、サーバーにあるリソースは上書きされます。
	false	インポートするリソースと同じ名前のリソースがサーバーに存在する場合、そのリソースはインポートされません。これがデフォルト設定です。

例

この例では "DynamicsTrial.json" という接続をインポートし、既存の接続が既にサーバー上に存在する場合は、その接続をアップデートします。

```
resourceconnection import --f D:\trunk_19\DynamicsTrial.json
--u true
```

resourceconnection list

`resourceconnection list` コマンドは、Spectrum™ Technology Platform サーバー上で定義されているすべてのリソース接続のリストを返します。

使用方法

```
resourceconnection list
```

必須	引数	説明
いいえ	`--t`*connectionType*	接続のタイプを指定します。

例

この例は、Spectrum™ Technology Platform で設定されているすべての接続のリストを表示します。

```
resourceconnection list
```

通知

notification daystoexpire set

`notification daystoexpire set` コマンドは、ライセンスまたはデータの有効期限をその何日前に通知するかを、有効期限までの日数で指定します。

使用方法

`notification daystoexpire set--d` ***Days***

必須	引数	説明
はい	`--d` ***Days***	ライセンスまたはデータの有効期限をその何日前に通知するかを、有効期限までの日数で指定します。

例

この例は、ライセンスまたはデータの有効期限が切れる 30 日前に通知を送信するように設定します。

```
notification daystoexpire set --d 30
```

notification enabled set

`notification enabled set` コマンドは、電子メールによるライセンスまたはデータの有効期限の事前通知を有効または無効にします。

使用方法

`notification enabled set--e`***TrueOrFalse***

必須	引数	説明
はい	`--e` ***TrueOrFalse***	ライセンスの有効期限の通知を有効または無効にします。ここで ***TrueOrFalse*** は次のいずれかです。 **true** ライセンスの有効期限の通知を有効にします。 **false** ライセンスの有効期限の通知を無効にします。

例

この例は、通知を有効にします。

```
notification enabled set --e true
```

notification expirationsettings list

`notification expirationsettings list` コマンドは、システムに適用されている有効期限通知設定を表示します。このコマンドは、通知が送信される有効期限までの日数、通知を受け取るユーザ、通知が有効か無効化を表示します。

使用方法

```
notification expirationsettings list
```

notification smtp get

`notification smtp get` コマンドは、ライセンス有効期限の通知の送信に使用される電子メールの設定を表示します。

使用方法

```
notification smtp get
```

notification smtp set

`notification smtp set` コマンドは、ライセンス有効期限通知の電子メール送信に使用する電子メールの設定を指定します。

使用方法

`notification smtp set --h` *Host* `--o` *Port* `--u` *UserName* `--p` *Password* `--e` *FromEmail*

必須	引数	説明
いいえ	`--h` *Host*	通知の電子メール送信に使用する SMTP サーバーのホスト名または IP アドレスを指定します。
いいえ	`--o` *Port*	SMTP サーバーによって使用されるポート番号または範囲を指定します。デフォルト値は 25 です。

必須	引数	説明
いいえ	`--u` *UserName*	電子メールの送信に使用するユーザ名を指定します。SMTP サーバー上の有効なユーザ アカウントでなければなりません。
いいえ	`--p` *Password*	`--u` パラメータで指定したユーザ アカウントのパスワードを指定します。
いいえ	`--e` *FromEmail*	通知電子メールの送信元電子メール アドレスを指定します。

例

この例は、電子メールによる通知の SMTP 設定を指定します。

```
notification smtp set --h mail.example.com --o 25 --u me123
--p MyPassword --e spectrum@example.com
```

notification smtp test

`notification smtp test` コマンドは、指定の電子メール アドレスにテスト電子メールを送信します。ライセンス有効期限の通知に使用する SMTP 設定を検証するには、このコマンドを使用します。

使用方法

`notification smtp test --e` *DestinationEmail*

必須	引数	説明
はい	`--e` *DestinationEmail*	テスト電子メール メッセージの送信先電子メール アドレスを指定します。

例

この例は、テスト用の電子メール アドレスを `me@example.com` に送信します。

```
notification smtp test --e me@example.com
```

notification subscriber add

`notification subscriber add` コマンドは、ライセンス有効期限の通知を受け取る電子メール アドレスを追加します。

使用方法

```
notification subscriber add --e Email
```

必須	引数	説明
はい	`--e` *Email*	ライセンス有効期限の通知を受け取る電子メール アドレスを指定します。

例

この例は、ライセンス有効期限の通知を受け取る電子メール アドレスとして `me@example.com` を追加します。

```
notification subscriber add --e me@example.com
```

notification subscriber delete

`notification subscriber delete` コマンドは、ライセンス有効期限の通知を受け取る電子メール アドレスのリストから電子メール アドレスを削除します。

使用方法

```
notification subscriber delete --e Email
```

必須	引数	説明
はい	`--e` *Email*	ライセンス有効期限の通知を受け取る電子メール アドレスのリストから削除する電子メール アドレスを指定します。

例

この例は、ライセンス有効期限の通知を受け取る電子メール アドレスのリストから電子メール アドレス `me@example.com` を削除します。

```
notification subscriber delete --e me@example.com
```

Open Parser カルチャー

openparser culture export

`openparser culture export` コマンドは、すべての Open Parser カルチャーをエクスポートします。その後、このカルチャーは別のサーバーにインポートできます。詳細については、『*データ品質ガイド*』の「パーシング」セクションを参照してください。

使用方法

`openparser culture export` ***OutputFile***

必須	引数	説明
いいえ	`--f` ***OutputFile***	エクスポートするカルチャーが含まれているファイルを指定します。この引数を省略すると、管理ユーティリティによって cultures.xml ファイルが自動的にエクスポートされます。

例

この例は、`culturesFR.xml` ファイルをエクスポートします。

```
openparser culture export culturesFR.xml
```

openparser culture import

`openparser culture import` コマンドは、Open Parser カルチャー ファイルをサーバーにインポートします。詳細については、『*データ品質ガイド*』の「パーシング」セクションを参照してください。

使用方法

`openparser culture import`***CultureFileName***

必須	引数	説明
はい	`--f` ***CultureFileName***	インポートするカルチャーが含まれているファイルを指定します。

例

この例は、`MyCulture.xml` というファイルをインポートします。

```
openparser culture import MyCulture.xml
```

Open Parser ドメイン

openparser domain export

`openparser domain export` コマンドは、Open Parser ドメインをエクスポートします。その後、このドメインは別のサーバーにインポートできます。詳細については、『*データ品質ガイド*』の「パーシング」セクションを参照してください。

使用方法

`openparser domain export` *DomainName* `--o` *OutputDirectory*

必須	引数	説明
はい	`--d` *DomainName*	エクスポートするドメインの名前を指定します。 **ヒント：** ドメイン名が正確にわからない場合は、`openparser domain list` コマンドを使用してドメイン名のリストを取得できます。
いいえ	`--o` *OutputDirectory*	ドメインのエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、ドメインは管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例は、"MyDomain" というドメインを、管理ユーティリティのインストール場所にある `exported` というサブフォルダにエクスポートします。

```
openparser domain export MyDomain --o exported
```

openparser domain import

`openparser domain import` コマンドは、Open Parser ドメインをサーバーにインポートします。詳細については、『*データ品質ガイド*』の「パーシング」セクションを参照してください。

使用方法

`openparser domain import`*DomainFileName*

必須	引数	説明
はい	`--f` *DomainFileName*	インポートするドメインが含まれているファイルを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。

例

この例は、`MyDomain.xml` というファイルをインポートします。このファイルは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。

```
openparser domain import exported\MyDomain.xml
```

openparser domain list

`openparser domain list` コマンドは、サーバー上のすべての Open Parser ドメインを一覧表示します。ドメインごとに、ドメイン名と、ドメインがエクスポーズされているかどうかの情報が表示されます。

使用方法

```
openparser domain list
```

パフォーマンス モニタ

performancemonitor enabled get

コマンド `performancemonitor enabled get` は、パフォーマンス モニタリングを有効にしたジョブおよびサービスを表示します。

使用方法

```
performancemonitor enabled get
```

performancemonitor enabled set

コマンド `performancemonitor enabled set` は、ジョブまたはサービスのパフォーマンス モニタリングを有効または無効にします。パフォーマンス モニタリングを有効にすると、ジョブまたはサービスのパフォーマンス情報がパフォーマンス ログに書き出されます。パフォーマンス ログには、ジョブまたはサービスの全般的なパフォーマンス情報に加えて、ジョブまたはサービス データフロー内の各ステージのパフォーマンス情報が含まれます。

使用方法

`performancemonitor enabled set--e`*TrueOrFalse*`--d`*DataflowName*

必須	引数	説明
はい	`--e` *TrueOrFalse*	パフォーマンス モニタリングを有効または無効にします。ここで *TrueOrFalse* は次のいずれかです。 **true** パフォーマンス モニタリングを有効にします。 **false** パフォーマンス モニタリングを無効にします。
はい	`--d` *DataflowName*	パフォーマンス モニタリングを有効または無効にするデータフローの名前を指定します。 **ヒント：** 使用するデータフローの名前が正確にわからない場合は、`dataflow list` コマンドを使用してデータフロー名のリストを取得できます。

例

この例では、データフロー MyNameParsingDataflow のパフォーマンス モニタリングを有効にします。

```
performancemonitor enabled set --e true --d
"MyNameParsingDataflow"
```

パフォーマンス ログ

パフォーマンス ログには、ジョブまたはサービスの実行にかかった時間の詳細が記録されます。ジョブまたはサービスの全般的なパフォーマンス情報に加えて、ジョブまたはサービス データフロー内の各ステージのパフォーマンス情報が含まれます。この情報を使用して、各ステージの実行時間と処理時間を確認し、ボトルネックを特定できます。実行時間と処理時間が大きく異なる場合、そのステージでは上流ステージからのデータを待つ時間が長いことを意味します。この場合は、上流ステージがデータフローにおけるボトルネックである可能性があります。シンクについては、実行時間と処理時間が大きく異なることが、必ずしもパフォーマンスの問題を意味するとは限らないことに注意してください。シンクは一般的に、データフローのその他の部分からの最初のレコードを待たなければならないためです。

ジョブまたはサービスのパフォーマンス モニタリングを有効にするには、管理ユーティリティでコマンド `performancemonitor enabled set` を使用します。

パフォーマンス ログは、お使いの Spectrum™ Technology Platform サーバーの次の場所にあります。

SpectrumDirectory\server\logs\performance.log

パフォーマンス ログは、モニタリング対象のジョブまたはサービスの各実行に対して 1 行で書き出されます。循環式のログで、最大 5 ファイルで構成されます。各ファイル サイズの上限は 10 MB です。この上限に達すると、最も古いパフォーマンス データが削除され、新しいパフォーマンス データが記録されます。

パフォーマンス ログの各エントリには、次の情報が含まれます。

> **注：** 以下では読みやすいように、改行とインデントを挿入しています。実際のログでは、エントリは 1 行で書き出されます。

```
Date Time [performance]
{
     "username":"UserName",
     "dataflowId":"DataflowName",
     "runMode":"BatchOrRealTime",
     "elapsedTime":Nanoseconds,
     "stageInfo":[
          {
               "stageName":"Name",
```

```
                "stageLabel":"Label",
                "options":{
                     OptionsList
                },
                "recordsRead":Count,
                "recordsWritten":Count,
                "executionTime":Nanoseconds,
                "processingTime":Nanoseconds
                "readBlockingTime":Nanoseconds
                "writeBlockingTime":Nanoseconds
                "readBlockingPercent":Percentage
                "writeBlockingPercent":Percentage
           }
      ]
}
```

説明:

username

ジョブまたはサービスを実行したユーザ。

dataflowID

Enterprise Designer で定義されているサービスまたはジョブの名前。

runMode

ジョブとサービスのどちらのログ エントリであるかを示します。次のいずれかです。

Batch	ジョブに対するログ エントリです。
RealTime	サービスに対するログ エントリです。

elapsedTime

ジョブまたはサービスのリクエストを実行するのにかかった時間 (単位: ナノ秒)。

stageInfo

データフローの各ステージにおける実行時間の情報を一覧表示します。各ステージに対して、次の情報が表示されます。

stageName

ステージの永続的な名前。

stageLabel

ステージのユーザ定義の名前。ステージ ラベルは、Enterprise Designer のキャンバス上に表示されます。

options

実行時に指定されたオプションがあれば、その設定内容がここに表示されます。

recordsRead

ステージの全入力ポートを通してこのステージに引き渡されたレコードの総数。

recordsWritten

ステージの全出力ポートに書き出されたレコードの総数。

executiontime

ステージで最初のレコードが処理されてから最後のレコードが処理されるまでの時間。これには、データフローの他のステージからのデータを待つ間、ステージがアイドル状態だった時間が含まれます。

processingtime

ステージがレコードを実際に処理していた時間。データフローの他のステージからのデータ待ちでアイドル状態だった時間を除きます。

readBlockingTime

次のレコードの読み取り中にブロックする時間の長さ。読み取りのブロック時間を長くすると、前のプロセスにかかる時間がこのステージよりも長くなり、追加のチューニングが必要になる場合があります。

writeBlockingTime

次のレコードの書き込み中にブロックする時間の長さ。書き込みのブロック時間を長くすると、前のプロセスにかかる時間がこのステージよりも長くなり、追加のチューニングが必要になる場合があります。

readBlockingPercent

レコードの読み取り時にステージがブロックしていた合計実行時間の割合。

writeBlockingPercent

レコードの書き込み時にステージがブロックしていた合計実行時間の割合。

権限

permission list

`permission list` コマンドは、権限を割り当てることができるエンティティの名前を一覧表示します。

使用方法

```
permission list
```

物理モデルおよび論理モデル

logicalmodel bulkExport

コマンドは、`logicalmodel bulkExport` すべての論理モデルおよびそのメタデータを、Metadata Insights から、指定したディレクトリにエクスポートします。出力ディレクトリを指定しない場合、モデルは、コマンドを実行しているディレクトリにエクスポートされます。論理モデルを、依存関係のある物理モデルとともにエクスポートするには、*exportDependency* 引数を使用します。

使用方法

`logicalmodel bulkExport--o`*outputDirectory*`--d`*exportDependencytrueOrFalse*

必須	引数	説明
いいえ	`--o` *outputDirectory*	論理モデルのエクスポート先となるディレクトリを指定します。
いいえ	`--d` *exportDependency*	モデルを、すべての依存関係とともにエクスポートするかどうかを指定します。 **true** モデルを依存関係とともにエクスポートします。 **false** モデルを依存関係のあるモデルなしでエクスポートします。

例

この例では、すべての論理モデルを、依存関係とともに、`C:\Spectrum\LogicalModels` にある "MyModelExport" フォルダにエクスポートします。

```
logicalmodel bulkExport --o
C:\Spectrum\LogicalModels\MyModelExport --d true
```

logicalmodel bulkImport

`logicalmodel bulkImport` コマンドは、すべての論理モデルおよびそのメタデータを、指定したディレクトリから Metadata Insights にインポートします。論理モデルを、依存関係のある物理モデルとともにインポートするには、*importDependency* 引数を使用します。

使用方法

`logicalmodel bulkImport --i` *inputDirectory* `--u` *trueOrFalse* `--d` *trueOrFalse*

必須	引数	説明
いいえ	`--i` *inputDirectory*	論理モデルのインポート元となるディレクトリを指定します。
いいえ	`--u` *updateIfExists*	既存モデルを更新するかどうかを指定します。 **true** 既存の論理モデルを更新します。 **false** 既存の論理モデルを更新しません。
いいえ	`--d` *importDependency*	論理モデルをその依存関係とともにインポートするかどうかを指定します。 **true** 論理モデルを依存関係とともにインポートします。 **false** 論理モデルを、依存関係のあるモデルなしでインポートします。

例

この例では、すべての論理モデルを、依存関係とともに、`C:\Spectrum\LogicalModels` にある "MyModel" フォルダにインポートします。また、同じ名前の既存モデルがあれば、それを更新します。

```
logicalmodel bulkImport --i C:\Spectrum\LogicalModels\MyModel
--u true --d true
```

logicalmodel export

`logicalmodel export` コマンドは、指定した論理モデルおよびそのメタデータを、Metadata Insights から、指定したディレクトリにエクスポートします。出力ディレクトリを指定しない場合、モデルは、コマンドを実行しているディレクトリにエクスポートされます。論理モデルを、

依存関係のある物理モデルとともにエクスポートするには、*exportDependency* 引数を使用します。

使用方法

```
logicalmodel export--nlogicalModelName--ooutputDirectory--dtrueOrFalse
```

必須	引数	説明
はい	`--n` *logicalModelName*	エクスポートする論理モデルの名前を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--o` *outputDirectory*	エクスポートした論理モデルを保存するフォルダの場所を指定します。
いいえ	`--d` *exportDependency*	論理モデルをその依存関係とともにエクスポートするかどうかを指定します。 **true** モデルを依存関係とともにエクスポートします。 **false** モデルを依存関係のあるモデルなしでエクスポートします。

例

この例では、論理モデル "MyModel" を、すべての依存関係とともに、`C:\Spectrum\LogicalModels` にある "MyModelExport" フォルダにエクスポートします。

```
logicalmodel export --n MyModel --o
C:\Spectrum\logicalModels\MyModelExport --d true
```

logicalmodel import

`logicalmodel import` コマンドは、指定した論理モデルおよびそのメタデータを、Metadata Insights にインポートします。論理モデルを、依存関係のある物理モデルとともにインポートするには、*importDependency* 引数を使用します。

使用方法

```
logicalmodel import--ilogicalModelInputFile--utrueOrFalse--dtrueOrFalse
```

必須	引数	説明
はい	`--i` *logicalModelInputFile*	インポートする論理モデルのファイルを指定します。

必須	引数	説明
いいえ	`--u` *updateIfExists*	Metadata Insights 内の同じ名前の既存モデルを、インポートしたモデルで更新するかどうかを指定します。 **true** 既存の論理モデルを更新します。 **false** 既存の論理モデルを更新しません。
いいえ	`--d` *importDependency*	論理モデルをその依存関係とともにインポートするかどうかを指定します。 **true** 論理モデルを依存関係とともにインポートします。 **false** 論理モデルを依存関係のあるモデルなしでインポートします。

例

この例では、論理モデルのファイル "MyModel" を、依存関係のあるモデルとともに Metadata Insights にインポートし、そのファイルで既存のファイルを更新します。

```
logicalmodel import --i MyModel --u true --d true
```

logicalmodel list

`logicalmodel list` コマンドは、論理モデルの全一覧を返します。

使用方法

```
logicalmodel list
```

例

この例は、論理モデルの全一覧を列挙します。

```
logicalmodel list
```

modelstore bulkExport

`modelstore bulkExport` コマンドは、すべてのモデル ストアを Metadata Insights からエクスポートします。

使用方法

`modelstore bulkExport --o` *outputDirectory* `--d` *trueOrFalse*

必須	引数	説明
いいえ	`--o` *outputDirectory*	モデル ストアのエクスポート先となるディレクトリを指定します。
いいえ	`--d` *exportDependency*	モデル ストアを、その依存関係とともにエクスポートするかどうかを指定します。 **true** モデル ストアを依存関係とともにエクスポートします。 **false** モデル ストアを依存関係のあるモデルなしでエクスポートします。

例

この例では、すべてのモデル ストアを依存関係とともに、`C:\Spectrum\ModelStore` にある "MyModelStoreExport" フォルダにエクスポートします。

```
modelstore bulkExport --o
C:\Spectrum\ModelStore\MyModelStoreExport --d true
```

modelstore deploy

`modelstore deploy` コマンドは、指定したモデル ストアを Spectrum サーバーに展開します。

使用方法

`modelstore deploy --n` *modelstoreName*

必須	引数	説明
はい	`--n` *modelStoreName*	展開するモデル ストアの名前を指定します。

例

この例では、"MyModelStore" という名前のモデル ストアを Spectrum サーバーに展開します。

```
modelstore deploy --n MyModelStore
```

modelstore export

`modelstore export` コマンドは、指定したモデル ストアを、Metadata Insights から、指定したフォルダにエクスポートします。

使用方法

`modelstore export--n`*modelStoreName*`--o`*outputDirectory*`--d`*trueOrFalse*

必須	引数	説明
はい	`--n` *modelStoreName*	エクスポートする論理モデルの名前を指定します。
いいえ	`--o` *outputDirectory*	エクスポートされたモデル ストアが保存されるディレクトリを指定します。このパスを指定しない場合、モデル ストアは、コマンドを実行しているディレクトリに保存されます。
いいえ	`--d` *exportDependency*	モデル ストアを、その依存関係とともにエクスポートするかどうかを指定します。 **true** モデル ストアを依存関係とともにエクスポートします。 **false** モデル ストアを依存関係のあるモデルなしでエクスポートします。

例

この例では、"MyModelStore" を、すべての依存関係とともに、`C:\Spectrum\ModelStores` にある "MyModelStore" フォルダにエクスポートします。

```
modelstore export --n MyModelStore --o C:\Spectrum\ModelStores
--d true
```

modelstore import

`modelstore import` コマンドは、指定したモデル ストア ファイルを Metadata Insights にインポートします。

使用方法

`modelstore import--i`*modelstoreInputFile*`--u`*trueOrFalse*`--d`*trueOrFalse*

必須	引数	説明
はい	`--i` *modelstoreInputFile*	インポートするモデル ストア ファイルを指定します。
いいえ	`--u` *updateIfExists*	既存の (同じ名前の) モデル ストアを、インポートするモデル ストアで更新するかどうかを指定します。 **true** 既存のモデル ストアを更新します。 **false** 既存のモデル ストアを更新しません。
いいえ	`--d` *importDependency*	モデル ストアを、その依存関係とともにインポートするかどうかを指定します。 **true** モデル ストアを依存関係とともにインポートします。 **false** モデル ストアを依存関係のあるモデルなしでインポートします。

例

この例では、モデル ストア ファイル "MyModelStore" を、その依存関係とともにインポートして、このファイル名ですでに存在するモデル ストアを更新します。

```
modelstore import --i MyModelStore --u --d
```

modelstore bulkImport

`modelstore bulkImport` コマンドは、すべてのモデル ストアを Metadata Insights にインポートします。

使用方法

`modelstore bulkImport --i` *inputDirectory* `--u` *trueOrFalse* `--d` *trueOrFalse*

必須	引数	説明
いいえ	`--i` *inputDirectory*	モデル ストアのインポート元となるディレクトリを指定します。
いいえ	`--u` *updateIfExists*	同じ名前の既存のモデル ストアを、インポートするモデル ストアで更新するかどうかを指定します。 **true** 既存のモデル ストアを更新します。 **false** 既存のモデル ストアを更新しません。

必須	引数	説明	
いいえ	`--d` *importDependency*	モデル ストアを、すべての依存関係とともにインポートするかどうかを指定します。	
		true	モデル ストアを依存関係とともにインポートします。
		false	モデル ストアを依存関係のあるモデルなしでインポートします。

例

この例では、すべてのモデル ストアを、その依存関係とともに、`C:\Spectrum\modelstore` にある "MyModel" フォルダから Metadata Insights にインポートします。また、すべての既存モデルを同じ名前で上書きします。

```
modelstore bulkImport --i C:\Spectrum\modelstore\MyModel --u
true --d true
```

modelstore list

`modelstore list` コマンドは、モデル ストアの全一覧を返します。

使用方法

```
modelstore list
```

例

この例は、モデル ストアの全一覧を列挙します。

```
modelstore list
```

modelstore undeploy

`modelstore undeploy` コマンドは、指定したモデル ストアの展開を、Spectrum サーバーから解除します。

使用方法

`modelstore undeploy --n` ***modelstoreName***

必須	引数	説明
はい	`--n` *modelstoreName*	展開を解除するモデル ストアの名前を指定します。

例

この例では、"MyModelStore" という名前のモデル ストアの展開を解除します。

```
modelstore undeploy --n MyModelStore
```

physicalmodel bulkExport

コマンドは、`physicalmodel bulkExport` すべての物理モデルおよびそのメタデータを、Metadata Insights から、指定したディレクトリにエクスポートします。

使用方法

`physicalmodel bulkExport--o`*OutputDirectory*

必須	引数	説明
いいえ	`--o` *outputDirectory*	物理モデルのエクスポート先となるディレクトリを指定します。指定しない場合、モデルは、コマンドを実行しているディレクトリにエクスポートされます。

例

この例では、すべての物理モデルを、`C:\Spectrum\PhysicalModels` にある "MyModelExport" フォルダにエクスポートします。

```
physicalmodel bulkExport --o
C:\Spectrum\PhysicalModels\MyModelExport
```

physicalmodel bulkImport

`physicalmodel bulkImport` コマンドは、すべての物理モデルを、指定した入力ディレクトリから Metadata Insights にインポートします。

使用方法

`physicalmodel bulkImport--i`*inputDirectory*

必須	引数	説明
いいえ	`--i` *inputDirectory*	物理モデルのインポート元となるディレクトリを指定します。
いいえ	`--u` *updateIfExists*	同じ名前の既存モデルを、インポートするモデルで更新するかどうかを指定します。

例

この例では、すべての物理モデルを `C:\Spectrum\PhysicalModels` にある "MyModel" フォルダからインポートします。また、同じ名前の既存のモデルがあれば、それを上書きします。

```
physicalmodel bulkImport --i C:\Spectrum\PhysicalModels\MyModel
--u
```

physicalmodel export

`physicalmodel export` コマンドは、指定した物理モデルおよびそのメタデータを、Metadata Insights から、指定したディレクトリにエクスポートします。

使用方法

`physicalmodel export --n` *physicalModelName* `--o` *outputDirectory*

必須	引数	説明
はい	`--n` *physicalModelName*	エクスポートする物理モデルの名前を指定します。
いいえ	`--o` *outputDirectory*	モデルのエクスポート先となるディレクトリを指定します。指定しない場合、モデルは、コマンドを実行しているディレクトリにエクスポートされます。

例

この例では、物理モデル "MyModel" を、`C:\Spectrum\PhysicalModels` にある "MyModelExport" フォルダにエクスポートします。

```
physicalmodel export --n MyModel --o
C:\Spectrum\PhysicalModels\MyModelExport
```

physicalmodel import

`physicalmodel import` コマンドは、指定した物理モデルのファイル、およびそのメタデータを、Metadata Insights にインポートします。

使用方法

`physicalmodel import --i` *physicalModelInputFile* `--u` *trueOrFalse*

必須	引数	説明
はい	`--i` *physicalModelInputFile*	インポートする物理モデル ファイルを指定します。
いいえ	`--u` *updateIfExists*	同じ名前の既存のモデルを、インポートするモデル ストアで更新するかどうかを指定します。 **true** 既存の物理モデルを更新します。 **false** 既存の物理モデルを更新しません。

例

この例では、物理モデルのファイル "MyModel" をインポートし、そのファイルで既存のファイルを更新します。

```
physicalmodel import --i MyModel --u true
```

physicalmodel list

`physicalmodel list` コマンドは、物理モデルの全一覧を返します。

使用方法

`physicalmodel list --t` *dataSourceType*

必須	引数	説明
いいえ	`--t` *dataSourceType*	一覧表示する物理モデルのデータソース タイプを指定します。 **注 :** 指定しない場合は、Metadata Insights にあるすべてのタイプの物理モデルが表示されます。

例

この例は、Salesforce の物理モデルの全一覧を列挙します。

```
physicalmodel list --t Salesforce
```

この例は、Metadata Insights 内のすべての物理モデルを表示します。

```
physicalmodel list
```

プロセスフロー

processflow delete

`processflow delete` コマンドは、システムからプロセスを削除します。

使用方法

`processflow delete--n`*`ProcessFlowName`*

必須	引数	説明
はい	`--n` *ProcessFlowName*	削除するプロセス フローを指定します。プロセス フローの名前にスペースが含まれている場合は、プロセス フロー名を引用符で囲みます。

例

この例は、My Process Flow というプロセス フローを削除します。

```
processflow delete --n "My Process Flow"
```

processflow execute

`processflow execute` コマンドは、1 つ以上のプロセス フローを実行します。このコマンドは、プロセス フローをコマンド ラインから実行できる 2 つの方法のうちの 1 つです。もう 1 つの方法は、サーバーの Spectrum™ Technology Platform ウェルカム ページから実行できるコマンド ライン ユーティリティである Process Flow Executor を使用することです。管理ユーティリティの `processflow execute` コマンドを使用する利点は、1 つのスクリプトまたはバッチ ファ

イルに他のコマンドも含められることです。例えば、`processflow expose` コマンドを使用してプロセス フローを公開した後で `processflow execute` コマンドを使用してそのフローを実行できます。`processflow execute` コマンドは、Process Flow Executor と同じ機能を備えています。

使用方法

`processflow execute --r` *ProcessFlowNames* `--f` *propertyFile* `--i` *PollInterval* `--d` *DelimiterCharacter* `--t` *Timeout* `--w` *WaitToComplete* `--o` *StageName=File*

必須	引数	説明
いいえ	`--?`	使用方法を出力します。
いいえ	`--d` *DelimiterCharacter*	コマンド実行時にコマンド ラインに表示されるステータス情報を区切るのに使用する区切り文字を設定します。デフォルトは "|" です。例えば、デフォルト文字を使用して、"MyProcessflow" というプロセス フローを実行した場合、コマンド ラインには次のメッセージが表示されます。 `MyProcessflow|1|Succeeded`
いいえ	`--f` *PropertyFile*	プロパティ ファイルへのパスを指定します。プロパティ ファイルの詳細については、プロセス フローのプロパティ ファイルの使用（223ページ） を参照してください。
いいえ	`--i` *PollInterval*	完了したジョブを確認する間隔を秒数で指定します。デフォルト値は "5" です。
はい	`--r` *ProcessFlowNames*	実行するプロセス フローのリストをカンマで区切って指定します。必須 **注：** ファイル オーバーライドを指定する場合は、この引数を最後の引数として指定しないでください。
いいえ	`--t` *Timeout*	このオプションは非推奨で、将来は無視されます。
いいえ	`--v` *Verbose*	詳細な出力を返します。ここで、*Verbose* は次のいずれかです。 **true** 詳細な出力を返します。 **false** 詳細な出力を返しません。 **注：** ファイル オーバーライドを指定する場合は、この引数を最後の引数として指定しないでください。

必須	引数	説明
いいえ	`--w` *WaitToComplete*	このオプションは非推奨で、将来は無視されます。
いいえ	`--o` *StageName=File*	ジョブで指定されている入力または出力ファイルをオーバーライドします。詳細については、「プロセス フロー ファイルの場所のオーバーライド（411ページ）」を参照してください。

例

この例は、"My Process Flow" というプロセス フローを実行します。

```
processflow execute --r "My Process Flow"
```

プロセス フロー ファイルの場所のオーバーライド

管理ユーティリティの `process flow execute` コマンドを使ってプロセス フローを実行する場合、ジョブで指定されたものとは別の入力ファイルや出力ファイルの使用を指定できます。これを行うには、`--o` 引数を使います。

```
--o "JobName|StageName=File"
```

説明:

JobName

は、プロセス フロー内で参照されるジョブの名前です。

StageName

データフロー内のステージのアイコンの下にあるステージ ラベルに表示される、ジョブの Read from File ステージまたは Write to File ステージの名前。例えば、入力ステージのラベルが "Read From File" である場合は、次のように指定します。

```
"Job1|Read From File=file:C:/inputfile.txt"
```

入力ステージのラベルが "Illinois Customers" である場合は、次のように指定します。

```
"Job1|Illinois Customers=file:C:/inputfile.txt"
```

File

ファイルのプロトコルとフル パス。ファイルのパスには、バックスラッシュではなくスラッシュを使用する必要があります。プロトコルは次のいずれかを指定します。

file:

ファイルが Spectrum™ Technology Platform サーバーと同じコンピュータに存在する場合は、パスの先頭に "file:" プロトコルを付けます。例えば、Windows で

は、"file:C:/myfile.txt" と指定し、Unix または Linux では、"file:/testfiles/myfile.txt" と指定します。

> **注:** クライアントとサーバーが同じコンピュータ上で稼働している場合は、"file:" プロトコルを使用しても "esclient:" プロトコルを使用しても構いませんが、"file:" プロトコルを使用した方がパフォーマンスが高くなる可能性があります

esclient:

ファイルが Process Flow Executor と同じコンピュータに存在する場合は、パスの先頭に "esclient:" プロトコルを付けます。例えば、Windows では、"esclient:C:/myfile.txt" と指定し、Unix または Linux では、"esclient:/testfiles/myfile.txt" と指定します。

> **注:** Process Flow Executor を実行しているコンピュータが Spectrum™ Technology Platform サーバーのホスト名を解決できない場合、"ファイルのアクセスでエラーが発生しました" というエラーが表示される可能性があります。この問題を解決するには、サーバー上で次のファイルを開きます: *SpectrumDirectory*/server/conf/spectrum-container.properties。プロパティ spectrum.runtime.hostname にサーバーの IP アドレスを設定します。

ftp:

Management Console で定義されたファイル サーバーを使用するには、ftp:*NameOfFileServer*/*PathToFile* という形式を使用します。例えば、ftp://FS/testfiles/myfile.txt と指定します。ここで FS は、Management Console で定義されたファイル サーバー リソースです。

次の例は、--o 引数を使ってファイルの場所をオーバーライドする方法を示しています。

```
--o "Job1|Read from File=file:C:/myfile_input.txt,Job1|Write to
File=file:C:/myfile_output.txt"
```

processflow export

processflow export コマンドは、サーバーから .pf ファイルにプロセス フローをエクスポートします。その後、このプロセス フローは別のサーバーにインポートできます。

> **注:** プロセス フローの交換は、同じバージョンの Spectrum™ Technology Platform の間でのみ行うことができます。

使用方法

`processflow export --n` ***ProcessFlowName*** `--o` ***OutputFile***

必須	引数	説明
はい	`--n` ***ProcessFlowName***	エクスポートするプロセス フローの名前を指定します。プロセス フローの名前にスペースが含まれている場合は、その名前を引用符で囲みます。 **ヒント：** プロセス フローの名前が正確にわからない場合は、`processflow list` コマンドを使用してプロセス フロー名のリストを取得できます。
いいえ	`--o` ***OutputFile***	プロセスフローのエクスポート先ディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、プロセスフローは管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例は、"My Process Flow" というプロセス フローを、管理ユーティリティのインストール場所にある `exported` というサブフォルダにエクスポートします。

```
processflow export --n "My Process Flow" --o exported
```

processflow expose

`processflow expose` コマンドは、プロセス フローを実行できる状態にします。

> **注：** Enterprise Designer におけるデータフローのバージョン管理を使用する場合は、`processflow expose` コマンドはデータフローの最新バージョンを公開します。

使用方法

`processflow expose--n`***ProcessFlowName***

必須	引数	説明
はい	`--n` *ProcessFlowName*	エクスポーズするプロセス フローの名前を指定します。プロセス フローの名前にスペースが含まれている場合は、その名前を引用符で囲みます。 **ヒント：** プロセス フローの名前が正確にわからない場合は、`processflow list` コマンドを使用してプロセス フロー名のリストを取得できます。

例

この例は、"My Process Flow" というプロセス フローをエクスポーズします。

```
processflow expose --n "My Process Flow"
```

processflow import

`processflow import` コマンドは、プロセス フロー ファイル (`.pf` ファイル) をサーバーにインポートします。`processflow export` コマンドを使用してサーバーからプロセス フローをエクスポートすると、プロセス フロー ファイルが作成されます。

使用方法

`processflow import--f`*ProcessFlowFile*`--u`*TrueOrFalse*`--p` *Path*`--c` *TrueOrFalse*

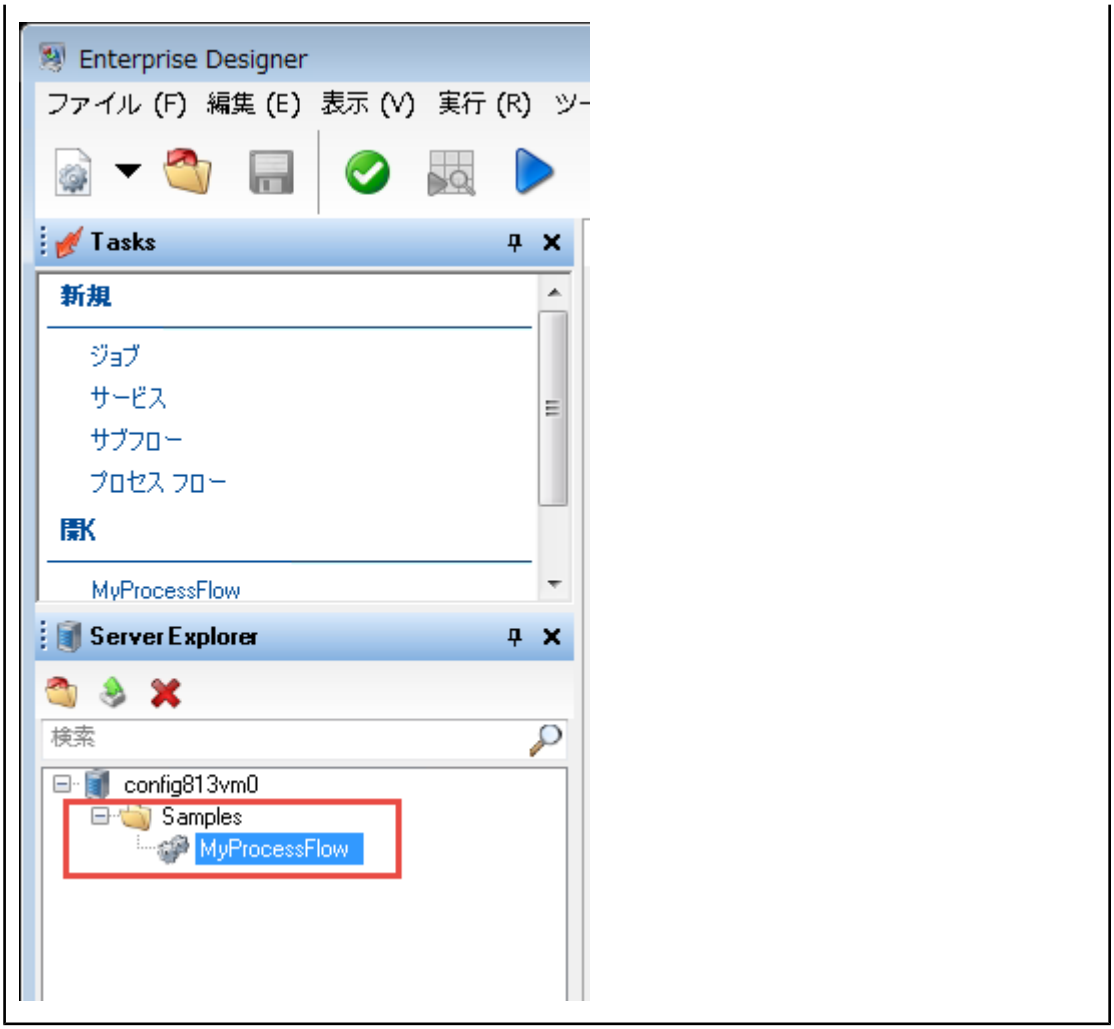
必須	引数	説明
はい	`--f` *ProcessFlowFile*	インポートするプロセス フロー ファイル (`.pf` ファイル) を指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。絶体パスを使用することもできます。
いいえ	`--u` *TrueOrFalse*	同じ名前のプロセス フローが既にサーバー上に存在する場合、既存のプロセス フローを上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true** インポートするプロセス フローと同じ名前のプロセスフローがサーバー上に存在する場合、サーバー上のプロセス フローは上書きされます。これがデフォルト設定です。 **false**

必須	引数	説明
		インポートするプロセス フローと同じ名前のプロセスフローがサーバー上に存在する場合、プロセスフローのインポートは行われません。
いいえ	`--p` *Path*	このフローのインポート先である Enterprise Designer サーバーのエクスプローラ フォルダを指定します。
いいえ	`--c` *TrueOrFalse*	存在しない場合に、`--p` で指定されたフォルダを作成するかどうかを指定します。 **true** 存在しない場合に、`--p` で指定されたフォルダを作成します。デフォルト **false** 存在しない場合、`--p` で指定されたフォルダを作成しません。`--p` で指定されたフォルダが存在しなければ、フローはインポートされません。

例

この例は、`MyProcessFlow.pf` というプロセス フローをインポートします。このファイルは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。プロセス フローは Enterprise Designer の `Samples` フォルダにインポートされます。

```
processflow import --f exported\MyProcessFlow.pf --p Samples
```



processflow list

`processflow list` コマンドは、サーバー上のすべてのプロセス フローを一覧表示します。プロセス フローごとに、プロセス フロー名と、そのプロセス フローがエクスポーズされているかどうかが表示されます。

使用方法

```
processflow list
```

processflow history list

コマンド `processflow history list` は、プロセス フローの実行履歴を表示します。

使用方法

`processflow history list--n`*ProcessFlowName*`--f`*FromDateTime*`--t`*ToDateTime*

必須	引数	説明
はい	`--n` *ProcessFlowName*	ステータスを取得するプロセス フローの名前を指定します。プロセス フローの名前にスペースが含まれている場合は、その名前を引用符で囲みます。 **ヒント：** プロセス フローの名前が正確にわからない場合は、`processflow list` コマンドを使用してプロセス フロー名のリストを取得できます。
いいえ	`--f` *FromDateTime*	特定の日時の範囲で履歴を表示する場合に、範囲の開始日時を MM-dd-yyyy HH:mm:ss の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `12-31-2014 13:00:00` と指定します。 日時の範囲を指定すると、`--f`引数で指定した日付以降で、`--t`引数で指定した日付以前に実行が開始したプロセス フローが履歴リストに含まれます。 この引数を省略すると、現在の日付に実行が開始されたプロセス フローが履歴に含まれます。
いいえ	`--t` *ToDateTime*	特定の日時の範囲で履歴を表示する場合に、範囲の終了日時を MM-dd-yyyy HH:mm:ss の形式で指定します。例えば、2014 年 12 月 31 日の午後 1:00 の場合は `12-31-2014 13:00:00` と指定します。 日時の範囲を指定すると、`--f`引数で指定した日付以降で、`--t`引数で指定した日付以前に実行が開始したプロセス フローが履歴リストに含まれます。 この引数を省略すると、`--f` 引数で指定した日付以降に実行が開始されたすべてのプロセス フローが履歴に含まれます。

例

この例では、"My Process Flow" というプロセス フローの履歴を取得します。

```
processflow history list --n "My Process Flow"
```

processflow unexpose

`processflow unexpose` コマンドは、プロセス フローを実行できない状態にします。

使用方法

`processflow unexpose--n`*ProcessFlowName*

必須	引数	説明
はい	`--n` *ProcessFlowName*	アンエクスポーズするプロセス フローの名前を指定します。プロセス フローの名前にスペースが含まれている場合は、その名前を引用符で囲みます。 **ヒント：** プロセス フローの名前が正確にわからない場合は、`processflow list` コマンドを使用してプロセス フロー名のリストを取得できます。

例

この例は、"My Process Flow" というプロセス フローをアンエクスポーズします。

```
processflow unexpose --n "My Process Flow"
```

processflow version list

コマンド `processflow version list` は、プロセス フローのすべての保存済みバージョンのバージョン情報を表示します。これには、バージョン番号、作成者、作成日、コメント、公開バージョンが含まれます。

使用方法

`processflow version list--n`*ProcessFlowName*

必須	引数	説明
はい	`--n` *ProcessFlowName*	バージョン情報を表示するプロセス フローの名前を指定します。プロセス フローの名前にスペースが含まれている場合は、その名前を引用符で囲みます。 **ヒント：** プロセス フローの名前が正確にわからない場合は、`processflow list` コマンドを使用してプロセス フロー名のリストを取得できます。

例

この例では、"My Process Flow" という名前のプロセス フローのバージョン情報を表示します。

```
processflow version list --n "My Process Flow"
```

製品データ

リスト

productdata list

`productdata list` コマンドは、現在インストールされている Spectrum 製品データ (SPD) の詳細を表示します。ファイル システムにアクセスしなくても、この結果から現在の情報がわかります。十分な情報に基づいて削除するデータを決定できるよう、このコマンドの実行後に `productdata delete` コマンドを実行することをお勧めします。

インストールされている製品ごとに、以下の詳細情報によって製品データの詳細な説明が作成されます。

- Product
- Component
- Qualifier
- Vintage
- Expiration [date]
- Identifier

使用方法

```
productdata list
```

抽出

productdata extract list

`productdata extract list` コマンドは、製品データ ファイルが製品名に基づいて抽出される場所を表示します。コマンド出力には、ファイルが製品ごとに保存されるディレクトリが表示されます。抽出場所 "platform" は、デフォルトでデータが抽出される場所を示しています。

使用方法

```
productdata extract list
```

productdata extract register

一連の製品データを抽出する代替 (デフォルト以外) の場所をサーバー上に設定するには、`productdata extract register` コマンドを使用します。

使用方法

`productdata extract register --p` ***product*** **--d** ***directory***

必須	引数	説明
はい	`--p` ***product***	この抽出によって参照される製品を指定します。最も一般的な値は "Platform" です。製品データをその他の場所に配置している場合は、代わりに "Global Addressing Module" などの値を使用します。
いいえ	`--d` ***directory***	指定した製品の抽出データを格納するディレクトリを定義します。デフォルトの場所は `/server/ref-data` です。

productdata extract unregister

`productdata extract unregister` を使用して、サーバーから抽出した製品データを削除します。

使用方法

`productdata extract unregister --p` ***product***

必須	引数	説明
はい	`--p` ***product***	削除される製品抽出データを指定します。例: "Global Addressing モジュール"。

アーカイブ

productdata archive list

`productdata archive list` コマンドは、展開済み製品データ ファイルが製品名に基づいてアーカイブされる場所を表示します。コマンド出力には、各ファイルが製品ごとにアーカイブされるディレクトリが表示されます。アーカイブ場所 "platform" はデフォルトのアーカイブ場所を示しています。

使用方法

```
productdata archive list
```

productdata archive register

一連の製品データの代替 (デフォルト以外の) アーカイブ場所をサーバー上に設定するには、`productdata archive register` コマンドを使用します。

使用方法

`productdata archive register --p` ***product*** **--d***directory*

必須	引数	説明
はい	`--p` ***product***	このアーカイブによって参照される製品を指定します。例: "Platform"。
いいえ	`--d` ***directory***	指定された製品のアーカイブを格納するディレクトリを定義します。デフォルトの場所は `/server/archive/ref-data` です。

productdata archive unregister

`productdata archive unregister` を使用して、アーカイブした製品データをサーバーから削除します。

使用方法

`productdata archive unregister --p` ***product***

必須	引数	説明
はい	`--p` ***product***	削除する製品アーカイブを指定します。例えば、"Platform" と指定します。

インストール

productdata install

`productdata install` コマンドを使用すると、コマンド ライン インターフェイス (CLI) から Spectrum 製品データ (SPD) をインストールできます。

使用方法

`productdata install --f`*fileOrDirectory* `--w`*waitOrReturn*

必須	引数	説明
はい	`--f` *fileOrDirectory*	インストール用の製品ファイルまたはディレクトリの場所を指定します。 • インストールする SPD ファイルを指定します。 • すべての SPD ファイルのインストール先となるディレクトリを指定します。
いいえ	`--w` *waitOrReturn*	インストール中に実行するアクションとして、完全にインストールされるまで待機してから復帰するか、直ちに復帰するかを指定します。

削除

productdata delete

`productdata delete` コマンドは、指定された Spectrum 製品データ (SPD) を Spectrum™ Technology Platform から削除します。削除する SPD データを含むすべての Spectrum のデータベースでこのコマンドを実行します。

このコマンドを実行する前に、`productdata list` コマンドを実行して、データ削除前の製品、コンポーネント、修飾子、ヴィンテージの詳細を確認しておきます。`productdata delete` コマンドのパラメータは `productdata list` コマンドを実行することで得られます。

使用方法

`productdata delete --p`*productName* `--c` *productComponent* `--q` *qualifier* `--v`*dataVintage*

必須	引数	説明
はい	`--p` *productName*	削除されるデータを使用している製品を指定します。例: "Platform"。
いいえ	`--c` *productComponent*	削除するデータコンポーネントの名前を指定します。例: "CanadaAddresses" のような製品コンポーネント。
いいえ	`--q` *qualifier*	データの一意識別子を指定します。
いいえ	`--v` *dataVintage*	データがコンパイルまたはリリースされた日付を指定します。例: "Q42018" のようなヴィンテージ。

プロファイル

profile run

`profile run` コマンドは指定されたプロファイルを実行します。

使用方法

`profile run --r` *profileName* `--w` *waitForComplete*

必須	引数	説明
はい	`--n` *profileName*	実行するプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント:** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。
いいえ	`--w` *waitForComplete*	`True` の場合、コマンドは、プロファイルが同期モードで完了するまで待ち、その後ステータスを表示します。 **注:** 指定しない場合のデフォルト値は `False` です

例

この例では、プロファイル "Profile_Demo" を実行して、その完了を待たずに、ステータスを表示します。

```
profile run --profileName Profile_Demo
```

profile cancel

`profile cancel` コマンドは指定されたプロファイルの実行をキャンセルします。

使用方法

`profile cancel --n` ***profileName***

必須	引数	説明
はい	`--n` ***profileName***	実行をキャンセルするプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。

例

この例では、"Profile_CLI" プロファイルの実行をキャンセルします。

```
profile cancel --profileName Profile_CLI
```

profile delete

`profile delete` コマンドは指定されたプロファイルを削除します。

使用方法

`profile delete --n` ***profileName***

必須	引数	説明
はい	`--n` *profileName*	削除するプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。

例

この例では、"Profile_CLI" プロファイルを削除します。

```
profile delete --profileName Profile_CLI
```

profile export

`profile export` コマンドにより、指定したプロファイルのレポートを対象のディレクトリにエクスポートできます。

使用方法

`profile export --n` *profileName* `--r` *profileRunId* `--t` *type* `--o` *OutputDirectory*

必須	引数	説明
はい	`--n` *profileName*	エクスポートするプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。
いいえ	`--r` *profileRunId*	エクスポートするプロファイルの ID を指定します。ID を指定しない場合、対象のプロファイルが最近実行された際の ID がシステムによって取得されます。
いいえ	`--t` *type*	プロファイルのエクスポート時に使用するレポート形式を指定します。レポートは PDF または Excel ファイルとしてエクスポート可能です。 **注：** 指定しない場合は Excel のレポートが生成されます。

必須	引数	説明
いいえ	`--o` *OutputDirectory*	プロファイルのエクスポート先となるディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。 **注：** この引数を省略すると、接続は管理ユーティリティと同じディレクトリにエクスポートされます。

例

この例は、"profile1" というプロファイルを `exported` というフォルダにエクスポートします。このフォルダは、管理ユーティリティをインストールした場所にあるサブフォルダです。レポートの形式は PDF ドキュメントです。

```
profile export --profileName profile1 --type pdf --o exported
```

profile update

`profile update` コマンドは、既存のプロファイルを更新します。このコマンドによって更新できるのは、ローカル マシン、またはサーバーにあるファイルを基に作成されたプロファイルのみです。

使用方法

`profile update --n` *profileName* `--d` *description* `--t` *profileOn* `--f` *fileName*

注： パラメータのリストを表示するには、"`help resourceconnection import`" と入力します。

必須	引数	説明
はい	`--n`*profileName*	更新するプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。
いいえ	`--d`*description*	更新するプロファイルの説明を指定します。
True	`--t` *profileOn*	プロファイルが作成されたソースを指定します。 **注：** 更新でサポートされているのは、ファイルのプロファイリングのみです。

必須	引数	説明
True	`--f` *fileName*	プロファイルを更新するファイルの名前を指定します。 **注：** そのファイルは、その他のプロファイル ファイルと同じディレクトリに存在する必要があります。

例

次の例では、"Scorecard" プロファイルの "profile_CLI.txt" というファイルを更新します。

```
profile update --n Scorecard --d "Running from CLI" --profileOn
File --f "profile_CLI.txt"
```

profile status

`profile status` コマンドにより、プロファイルのステータスを表示できます。

使用方法

`profile status --n` *profileName* `--r` *profileRunId*

注： パラメータのリストを表示するには、"`help resourceconnection import`" と入力します。

必須	引数	説明
はい	`--n`*profileName*	ステータスを表示するプロファイルの名前を指定します。プロファイル名にスペースが含まれる場合は、名前を引用符で囲みます。 **ヒント：** プロファイル名が正確にわからない場合は、`profile list` コマンドを使用して、プロファイル名のリストを取得できます。
いいえ	`--r`*profileRunId*	ステータスを表示するプロファイル実行の ID を指定します。ID を指定しない場合、対象のプロファイルが最近実行された際の ID がシステムによって取得されます。

例

次の例では、プロファイル名 "Profile_CLI" の実行 ID "92" のステータスを表示できます。

```
profile status --profileName Profile_CLI --profileRunId 92
```

profile list

`profile list` コマンドは、作成済みプロファイルの全一覧を返します。

使用方法

```
profile list
```

> **注：** パラメータのリストを表示するには、"`help resourceconnection import`" と入力します。

必須	引数	説明
いいえ	`--n`*jsonFormat*	プロファイルのリストが JSON 形式で返るように指定します。

例

この例では、Spectrum™ Technology Platform で定義したすべてのプロファイルが返ります。

```
profile list
```

Roles

role create

`role create` コマンドは、JSON ファイルに定義された権限で新しい役割を作成します。

使用方法

`role create --r`*RoleName* `--f` *JSONFile*

必須	引数	説明
はい	`--r` *RoleName*	新しい役割の名前です。
はい	`--f` *JSONFile*	新しい役割の定義が含まれている JSON ファイルへのパス。

役割を定義するファイルの形式

役割定義ファイルを作成する場合、`role permission export` コマンドを使って既存の役割のファイルを生成し、それを変更するのが最も簡単な方法です。このファイルでは、Management Console に表示される権限リストと同じように権限がグループ化されます。セキュア エンティティごとに、EXECUTE、DELETE、CREATE、MODIFY、または VIEW の権限を指定できます。有効な値は次のとおりです。

true

権限を与えます。

false

権限を与えません。

NULL

NULL 値は、セキュア エンティティに適用されない権限を意味します。

次の例は、新しい役割を `MyNewRole` という名前で作成します。この役割は、権限グループ `Matching` の権限を持ちます。権限は、`Open Parser Cultures`、`Open Parser Domains`、および`OpenParser Tables` です。

```
{
  "name" : "MyNewRole",
  "userNames" : [ ],
  "groups" : [ {
    "name" : "Matching",
    "permissions" : [ {
      "name" : "Open Parser Cultures",
      "permissions" : {
        "EXECUTE" : "",
        "DELETE" : "",
        "CREATE" : "",
        "MODIFY" : "true",
        "VIEW" : "true"
      }
    }, {
      "name" : "Open Parser Domains",
      "permissions" : {
        "EXECUTE" : "",
        "DELETE" : "",
        "CREATE" : "",
        "MODIFY" : "false",
        "VIEW" : "false"
      }
    }, {
      "name" : "OpenParser Tables",
      "permissions" : {
        "EXECUTE" : "",
        "DELETE" : "false",
        "MODIFY" : "false",
        "CREATE" : "false",
```

```
          "VIEW" : "false"
        }
      } ]
    } ],
}
```

例

この例は、SalesAnalyst という新しい役割を作成し、ファイル `c:\roles\SalesAnalyst.json` にある役割定義を使用します。

```
role create --r SalesAnalyst --f C:\roles\SalesAnalyst.json
```

role delete

`role delete` コマンドは、役割を削除します。

使用方法

`role delete --r`*RoleName*

必須	引数	説明
はい	`--r` *RoleName*	削除する役割の名前を指定します。

例

この例は、SalesAnalyst という名前の役割を削除します。

```
role delete --r SalesAnalyst
```

role export

`role export` コマンドは、すべての役割定義を `roles.json` という名前の JSON ファイルにエクスポートします。このファイルは、`role import` コマンドへの入力として使用されます。

使用方法

`role export --o` *Folder*

必須	引数	説明
いいえ	`--o` *Folder*	役割をエクスポートするフォルダの名前を指定します。相対パスを指定する場合、管理ユーティリティがある場所からの相対パスを使

必須	引数	説明
		用します。パスを指定しない場合、役割は管理ユーティリティがあるフォルダにエクスポートされます。

例

この例は、役割を RoleExports フォルダにエクスポートします。

```
role export --o RoleExports
```

注： "\n" または "\t" で始まる名前が許可されていないディレクトリに、役割権限をエクスポートすることはできません。これらの文字列は、それぞれ次の行、およびタブ文字として認識されます。回避策として、フォワード スラッシュを使用できます。

role import

`role import` コマンドは、`role export` コマンドを使用して定義された JSON ファイル `roles.json` から役割定義とその関連権限をインポートします。

使用方法

`role import --f`*File*

必須	引数	説明
はい	`--f` *File*	役割と権限を含むファイルの名前を指定します。役割が存在しない場合は、役割がファイル内の権限で作成されます。役割が存在する場合は、役割が更新されます。 パスを指定しない場合、役割は管理ユーティリティのインストール場所からインポートされます。

例

この例では、役割を RoleImports ファイルにインポートします。

```
role import --f RoleImports
```

role list

`role list` コマンドは、システムにあるすべての役割の名前を一覧表示します。

使用方法
```
role list
```

role permission export

`role permission export` コマンドは、役割定義を JSON ファイルにエクスポートします。

使用方法
`role permission export --r`*RoleName* `--o` *OutputFolder*

必須	引数	説明
はい	`--r` *RoleName*	エクスポートする役割の名前を指定します。
いいえ	`--o` *OutputFolder*	役割をエクスポートするフォルダの名前を指定します。相対パスを指定する場合、管理ユーティリティがある場所からの相対パスを使用します。パスを指定しない場合、役割は管理ユーティリティがあるフォルダにエクスポートされます。

例

この例は、SalesAnayst という役割をフォルダ c:\roles にエクスポートします。

```
role permission export --r SalesAnalyst --o C:\roles
```

role permission import

`role permission import` コマンドは、権限設定を JSON ファイルからインポートして既存の役割を変更します。

使用方法
`role permission import --r`*RoleName* `--f`*PermissionsFile*

必須	引数	説明
はい	`--r` *RoleName*	変更する役割の名前を指定します。

必須	引数	説明
はい	`--f` *PermissionsFile*	役割に追加する権限が含まれている JSON ファイルの名前を指定します。相対パスを指定する場合、管理ユーティリティがある場所からの相対パスを使用します。

例

この例は、SalesAnayst という役割を、`c:\roles\permissions.json` で定義されている権限を持つように変更します。

```
role permission import --r SalesAnalyst --f
C:\roles\permissions.json
```

検索インデックス

index delete

`index delete` コマンドは、Advanced Matching モジュールの検索インデックスを削除します。

使用方法

`index delete --d` *Name*

必須	引数	説明
はい	`--d` *Name*	削除する検索インデックスの名前を指定します。

例

この例では、"CustomerIndex" という名前の検索インデックスを削除します。

```
index delete --d CustomerIndex
```

index list

`index list` コマンドは、すべての Advanced Matching モジュール検索インデックスの一覧をテーブル形式で返します。詳細情報としてインデックス名、インデックス タイプ、レコード数が

含まれます。インデックス タイプは *legacy* (従来のインデックス) と *clustered* (クラスタ化インデックス) のどちらかです。クラスタ化インデックスは、バックアップおよび復元が可能です。[Legacy] および [Clustered] インデックスはどちらも、エクスポート ユーティリティを使って `*.txt` ファイルにエクスポートできます。

インデックスのリストを任意の場所の *.csv* ファイルに書き込むこともできます。

使用方法

```
index list --f filePath
```

必須	引数	説明
いいえ	`--f` *filePath*	インデックスのリストを書き込むファイル パス。 **注 :** 出力はカンマ区切り形式のテキスト ファイルです。

例 1

この例は、検索インデックスのリストを *c:/exportLocation* のファイル *listOutput.csv* に書き込みます。

```
index list --f c:/exportLocation/listOutput.csv
```

例 2

この例は、すべての検索インデックスを一覧表示します。

```
index list
```

index export cancel

`index export cancel` コマンドにより、検索インデックスのエクスポートがキャンセルされます。キャンセル コマンドの実行前にエクスポートされたデータは、指定した出力場所に保存されています。

使用方法

```
index export cancel --i Export_Id
```

必須	引数	説明
はい	`--i` *Export ID*	指定したエクスポート ID の操作がキャンセルされます。 **注 :** エクスポート ID を確認するには、index export progress コマンドを使用します。

例

Export_ID という ID のエクスポート操作をキャンセルする場合、以下のように入力します。

```
index export cancel --i Export_ID
```

index export progress

`index export progress` コマンドにより、エクスポート中の検索インデックスのステータスを表示できます。詳細には、エクスポート ID、インデックス名、全レコード数、エクスポートされたレコード数、およびエクスポートされる場所が含まれます。

エクスポート中のすべてのインデックスのステータスをファイルに書き込むこともできます。

使用方法

`index export progress --f` ***filePath***

必須	引数	説明
いいえ	`--f` ***filePath***	エクスポート中のすべてのインデックスのステータスを書き込むファイル パス。 **注 :** 出力はカンマ区切り形式のテキスト ファイルです。

例 1

エクスポート中のすべての検索インデックスを一覧表示するには、以下のように入力します。

```
index export progress
```

例 2

この例は、エクスポート中のすべてのインデックスのステータスを *c:/exportLocation* のファイル *exportProgress.csv* に書き込みます。

```
index export progress --f c:/exportLocation/exportProgress.csv
```

index export start

`index export start` コマンドにより、検索インデックスを、指定した出力場所に `*.zip` ファイル形式でエクスポートできます。この `*.zip` ファイルには、テキスト修飾子としてパイプ区切り文字と二重引用符を使用した `*.txt` ファイルが含まれています。出力ファイル名は、対応する検索インデックスの名前と、その後に続くタイム スタンプで構成されています。

- **[Legacy]** インデックスの場合は、インデックス作成時に **[保存]** とマークされたフィールドのみがこのコマンドによってエクスポートされます。
- **[Clustered]** インデックスの場合は、すべてのインデックス フィールドがこのコマンドによってエクスポートされます。
- 改行タイプは、Windows の場合、`CRLF`、また、Windows でない場合は、`LF` になります。

使用方法

`index export start --i` *indexName* `--o` *outputLocation*

必須	引数	説明
はい	`--i` *indexName*	エクスポートする検索インデックスの名前を指定します。
はい	`--o` *outputLocation*	エクスポート対象インデックスの出力場所を指定します。出力場所を指定しなかった場合は、その指定を促すメッセージが表示されます。

例

この例では、"CustomerIndex" という名前の検索インデックスを出力場所 "pbIndexExports" にエクスポートします。

```
index export start --i CustomerIndex --o c:/pbIndexExports
```

index snapshot create

`index snapshot create` コマンドは、検索インデックスのスナップショットを作成します。スナップショットを正常に作成するには、インデックス内に設定されているデータ セット全体が有効である必要があります。いずれかのプライマリ シャードが欠けていると、スナップショットの作成に失敗します。スナップショットを一旦作成しておくと、その後のバックアップ時間が大幅に短縮されます。データ追加および削除の増分のみが必要になるためです。

スナップショットの作成は、検索インデックスに対して並行して実行されているどの操作にも影響されません。ただし、その特定の時間にインデックスに存在するレコードのみがスナップショットに記録されます。

作成したスナップショットのステータスを表示するには、コマンド `index snapshot list` を使用します。詳細については、**index snapshot list**（437ページ）を参照してください。

> **注：** 検索インデックスのスナップショットを作成する前に、インデックス スナップショット リポジトリを作成しておく必要があります。詳細については、「**index snapshot repository**（439ページ）」を参照してください。

使用方法

```
index snapshot create
```

必須	引数	説明
はい	`--i` *Index name*	作成するスナップショットの中の検索インデックスの名前。
はい	`--s` *Snapshot name*	作成するスナップショットの名前。 **注：** 名前は一意である必要があります。 **注：** コマンドで使用する大文字小文字の区別に関係なく、この名前は小文字になります。

例

この例では、検索インデックス "customer_index" の新しいスナップショット "my_snapshot" を作成します。

```
index snapshot create --i customer_index --s my_snapshot
```

index snapshot list

`index snapshot list` コマンドは、検索インデックス スナップショットの全一覧を返します。次の詳細情報が表示されます。

- スナップショットの名前
- 検索インデックスの名前
- スナップショットが正常に作成されたかどうか
- 失敗の理由 (スナップショットが正常に作成されなかった場合)
- スナップショット作成の開始時間、終了時間、総所要時間
- スナップショット内の合計シャード数と、成功シャードおよび失敗シャード (存在する場合)

検索インデックス スナップショットのリストをファイルに書き込むこともできます。

使用方法

```
index snapshot list --f filePath
```

必須	引数	説明
いいえ	`--f` *filePath*	検索インデックス スナップショットのリストを書き込むファイル パス。 **注：** 出力はカンマ区切り形式のテキスト ファイルです。

例 1

この例は、すべての検索インデックス スナップショットを一覧表示します。

```
index snapshot list
```

例 2

この例は、検索インデックス スナップショットのリストを *c:/exportLocation* のファイル *snapshotList.csv* に書き込みます。

```
index snapshot list --f c:/exportLocation/snapshotList.csv
```

index snapshot delete

`index snapshot delete` コマンドは、検索インデックス スナップショットを削除します。

使用方法

```
index snapshot delete
```

必須	引数	説明
はい	`--i` *Index name*	スナップショットを削除する検索インデックスの名前。
はい	`--s` *Snapshot name*	削除するスナップショットの名前。

例

この例では、検索インデックス "customer_index" のスナップショット "my_snapshot" を削除します。

```
index snapshot delete --i customer_index --s my_snapshot
```

index snapshot repository

`index snapshot repository` コマンドは、検索インデックス スナップショット リポジトリの共有ファイル システム パスを設定します。任意の数のリポジトリを設定できます。ただし、検索インデックス エンジンは必ず、現在設定されているファイル パスをデータのバックアップと復元に使用します。

検索インデックスのインデックス スナップショット リポジトリを (CLI で) 使用するにはまず、以下の場所でリポジトリ パス `path.repo` を指定して、サーバーを再起動する必要があります。クラスタ設定の場合は、この変更をすべてのノードで行う必要があります。このパスは、リポジトリ *`SpectrumDirectory`*`\index\elasticsearch.template` の作成時に使用されます。

例えば、次のようにパスを指定できます。

- `path.repo: ["/mount/backups"]`
- `path.repo: ["C:/SIbackups"]`

上記のパスを使用して作成されたリポジトリの例を以下に示します。

- `index snapshot repository --p /mount/backups/index_customer`
- `index snapshot repository --p C:/SIbackups/index_customer`

使用方法

```
index snapshot repository
```

必須	引数	説明
はい	`--p` *Name*	インデックス スナップショット リポジトリを設定するパスを指定します。 **注：** 共有ファイル システム パスを使用することが必須です。
いいえ	`--o` *Overwrite*	既に設定されているパスを新しいもので上書きします。

例

この例では、新しいインデックス スナップショット リポジトリをパス "c:\\MyIndexRepository" に設定します。

```
index snapshot repository --p c:\\MyIndexRepository
```

index snapshot restore

`index snapshot restore` コマンドは、検索インデックス スナップショットを復元します。

注： スナップショットを復元中の検索インデックスに、操作を行うことはできません。

スナップショットの復元のステータスを表示するには、コマンド `index restore list` を使用します。詳細については、**index snapshot list**（437ページ）

使用方法

```
index snapshot restore
```

必須	引数	説明
はい	`--i` *Index name*	スナップショットを復元する検索インデックスの名前。
はい	`--s` *Snapshot name*	復元するスナップショットの名前。

例

この例では、検索インデックス "customer_index" のスナップショット "my_snapshot" を復元します。

```
index snapshot restore --i customer_index --s my_snapshot
```

index restore list

`index restore list` コマンドは、Advanced Matching モジュールの復元されたインデックス スナップショットの全一覧を返します。テーブル形式の詳細情報には、インデックスの名前とその復元ステータスが含まれます。また、各シャードの復元にかかった総所要時間、ステータス、復元の説明など、シャード全体の復元詳細情報も示されます。

復元されたすべてのインデックス スナップショットのリストをファイルに書き込むこともできます。

使用方法

```
index restore list --f filePath
```

必須	引数	説明
いいえ	`--f` *filePath*	復元されたすべてのインデックス スナップショットのリストを書き込むファイル パス。 **注：** 出力はカンマ区切り形式のテキスト ファイルです。

例 1

この例は、復元されたすべての検索インデックス スナップショットを一覧表示します。

```
index restore list
```

例 2

この例は、復元されたスナップショットのステータスを *c:/exportLocation* のファイル *restoreList.csv* に書き込みます。

```
index restore list --f c:/exportLocation/restoreList.csv
```

サービス

service list

`service list` コマンドは、サーバーに公開されたサービスを一覧に表示します。公開されていないサービスは、一覧に表示されません。

注： サービスを公開するには、`dataflow expose` コマンドを使用します。

使用方法

```
service list
```

service loglevel list

`service loglevel list` コマンドは、各サービスのログに含まれる詳細レベルを一覧に示します。次のログ レベルがあります。

Default サービスでは、システム デフォルトのログ レベルが使用されます。システム ログ レベルは、`systemloglevel set` コマンドを使用して設定できます。

Disabled すべてのイベント ログを無効にします。

Fatal 最小限のログです。致命的なエラーのみがログに記録されます。致命的なエラーとは、システムを使用不可能にするエラーのことです。

Error エラーと致命的なエラーがログに記録されます。エラーとは、システムの一部が使用不可能になる単発的な問題を指します。例えば、1 つのサービスが機能しなくなる問題によってエラーが生成されます。

Warn	イベント警告、エラー、致命的なエラーがログに記録されます。警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。
Info	抽象度の高いシステム情報がログに記録されます。これは、実稼働に適した最も詳細なログ レベルです。通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。
Debug	システムの問題のデバッグ時に適した、非常に詳細なログ レベルです。
Trace	プログラムの実行 (メソッドの開始と終了) をトレースする、最も詳細なログ レベルです。デバッグに使用する詳細なプログラム フロー情報を提供します。

使用方法

```
service loglevel list
```

service loglevel set

`service loglevel set` コマンドは、サービス ログに含まれる詳細レベルを指定します。

デフォルトのログ レベルや、システム上の各サービスのログ レベルを指定することができます。ログ レベルを変更しても、変更前に作成されたログ エントリにはその変更は反映されません。

使用方法

```
service loglevel set--sServiceName--l LogLevel
```

必須	引数	説明
はい	`--s` *ServiceName*	ログ レベルを設定するサービスの名前を指定します。
はい	`--l` *LogLevel*	サービスのログ レベルを指定します。ここで、*LogLevel* は次のいずれかです。 **Default** サービスでは、システム デフォルトのログ レベルが使用されます。システム ログ レベルは、`systemloglevel set` コマンドを使用して設定できます。 **Disabled** すべてのイベント ログを無効にします。 **Fatal** 最小限のログです。致命的なエラーのみがログに記録されます。致命的なエラーとは、システムを使用不可能にするエラーのことです。 **Error** エラーと致命的なエラーがログに記録されます。エラーとは、システムの一部が使用不可能になる単発的な問題を指します。

必須	引数	説明

例えば、1つのサービスが機能しなくなる問題によってエラーが生成されます。

Warn イベント警告、エラー、致命的なエラーがログに記録されます。警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。

Info 抽象度の高いシステム情報がログに記録されます。これは、実稼働に適した最も詳細なログ レベルです。通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。

Debug システムの問題のデバッグ時に適した、非常に詳細なログ レベルです。

Trace プログラムの実行 (メソッドの開始と終了) をトレースする、最も詳細なログ レベルです。デバッグに使用する詳細なプログラム フロー情報を提供します。

各ログ レベルは、1つ上のログ レベルの内容を含みます。つまり、警告ログ レベルが選択されている場合、エラーと致命的なエラーも記録されます。情報ログ レベルが選択されている場合は、情報メッセージ、警告、エラー、および致命的なエラーが記録されます。

注： 最も詳細なログ レベルを選択すると、システムのパフォーマンスに影響が生じる恐れがあります。したがって、必要なログ要件を満たす最小レベルの設定を選択する必要があります。

例

この例では、ValidateAddress のログ レベルを Warn に設定します。

```
service loglevel set --s ValidateAddress --l Warn
```

service option list

`service option list` コマンドは、サービスに適用できるオプションを一覧に表示します。各サービスのオプションと値については、『*API ガイド*』、『*REST Web サービス ガイド*』、または『*SOAP Web サービス ガイド*』を参照してください。

使用方法

`service option list--s`*ServiceName*

必須	引数	説明
はい	`--s` *ServiceName*	オプションを表示するサービスの名前を指定します。サービス名は大文字と小文字を区別します。

例

この例では、ValidateAddress サービスに適用できるオプションのリストを表示します。

```
service option list --s ValidateAddress
```

service option set

`service option set` コマンドは、サービス オプションのデフォルト設定を指定します。

デフォルト サービス オプションは、システム上の各サービスのデフォルトの動作を制御します。サービス内の各オプションのデフォルト値を指定できます。デフォルト オプション設定は、API 呼び出しまたは Web サービス リクエストで、オプションの値が明示的に定義されていない場合に有効です。また、デフォルト サービス オプションは、Enterprise Designer でこのサービスを使用してフローを作成する際にデフォルトとして使用される設定です。

使用方法

`service option set--s`*ServiceName*`--o` *OptionName*`--v`*Value*

必須	引数	説明
はい	`--s` *ServiceName*	オプションを設定するサービスの名前を指定します。サービス名は大文字と小文字を区別します。

必須	引数	説明
はい	`--o` *OptionName*	設定するオプションの名前。各サービスのオプションと値については、『*API ガイド*』、『*REST Web サービス ガイド*』、または『*SOAP Web サービス ガイド*』を参照してください。
はい	`--v` *Value*	オプションに設定する値。各サービスのオプションと値については、『*API ガイド*』、『*REST Web サービス ガイド*』、または『*SOAP Web サービス ガイド*』を参照してください。

例

この例では、ValidateAddress サービスの MaximumResults オプションを 15 に設定します。

```
service option set --s ValidateAddress --o MaximumResults --v
15
```

Spectrum のデータベース

グローバル データベース用の Enterprise Geocoding モジュール

egmglobaldb create sample file

コマンド `egmglobaldb create_sample_file` は、1 つまたは 2 つのデータベース リソースのサンプル JSON ファイルを作成します。生成されたこれらのファイルは、データベース リソースを作成するための設定の参考として使用できます。このコマンドにより、現在のディレクトリまたは指定したフォルダ場所に、JSON ファイル *egmGlobalSingleDictDbResource.txt* と *egmGlobalDoubleDictDbResource.txt* が作成されます。

使用方法

`egmglobaldb create_sample_file`***outputpath***

必須	引数	説明
いいえ	*outputpath*	すべてのデータベース リソースのサンプル JSON ファイルは、指定した出力パスに作成されます。指定しない場合は、現在のフォルダにエクスポートされます。

例

この例では、グローバル データベース リソースのサンプル JSON ファイルを現在のフォルダに作成します。2 つめの例では、すべてのデータベース リソースが C:\OutputFolder にエクスポートされます。

```
egmglobaldb create_sample_file
```

```
egmglobaldb create_sample_file C:\OutputFolder
```

データベース リソースのサンプル JSON ファイル

```
[{"product":"InternationalGeocoder GLOBAL",
"module":"igeocode-global", "name":"$$DATABASE_NAME$$",
"maxActive":4,
"properties":{"COUNTRY_CODE1":"$$COUNTRY_CODE1$$",
"$$COUNTRY_CODE1$$_DICTIONARY_PATH1":"$$DICTIONARY_PATH1$$",
"COUNTRY_COUNT":"1",
"$$COUNTRY_CODE1$$_DICTIONARY_PATH_NAME1":"$$DICTIONARY_PATH_NAME1$$"}}]
```

egmglobaldb delete

コマンド `egmglobaldb delete` は、設定されている Enterprise Geocoding モジュールのグローバル データベースを削除します。

使用方法

`egmglobaldb delete--n`*Name*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。

例

この例では、グローバル モジュールからグローバル データベースを削除します。

```
egmgbrdb delete --n Global
```

egmglobaldb import

コマンド `egmglobaldb import` は、Enterprise Geocoding モジュールのグローバル データベース プロパティ ファイルをインポートします。これによって、現在のシステム上でグローバル データベース リソースが設定されます。

使用方法

`egmglobaldb import--f`*File*

必須	引数	説明
はい	`--f` *File*	JSON 形式のデータベース プロパティ ファイルを指定します。このファイルは必須です。

例

この例では、JSON形式ファイルである `egmGlobalSingleDictDbResource.txt` 内の設定の定義に従ってグローバル データベース リソースを作成します。

```
egmglobaldb import --f egmGlobalSingleDictDbResource.txt
```

egmglobaldb export

`egmglobaldb export` コマンドは、すべてのグローバル データベース リソース情報を、指定された場所にあるデータベース プロパティ ファイル `EgmGlobalDbResource.txt` にエクスポートします。出力ファイルの場所が指定されていない場合、`EgmGlobalDbResource.txt` ファイルは現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `egmglobaldb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`egmglobaldb export --o` *outputpath*

必須	引数	説明
いいえ	`--o` *outputpath*	すべてのデータベース リソースは、指定された出力パスにエクスポートされます。パスが指定されていない場合は、すべてのリソースが現在のフォルダにエクスポートされます。エクスポートされた JSON 形式の出力ファイル EgmGlobalDbResource.txt には、データベース プロパティ情報が含まれます。

例

この例は、データベース リソース情報を指定された場所にエクスポートします。

```
egmglobaldb export --o C:\DBs\
```

egmglobaldb get

コマンド `egmglobaldb get` は、Global Enterprise Geocoding モジュール データベースに関する情報を返します。

使用方法

`egmglobaldb get --n` *名前*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。このファイルは必須です。

例

この例では、設定されているグローバル データベース リソースのすべての情報を表示します。

```
egmglobaldb get --n Global
```

egmglobaldb list

コマンド `egmglobaldb list` は、設定されているすべての Global Enterprise Geocoding モジュール データベースと、そのプール サイズを表示します。

使用方法

`egmglobaldb list` このコマンドにはプロパティはありません。

例

この例は、Enterprise Geocoding モジュールのグローバル データベースとプール サイズを一覧表示します。

```
egmglobaldb list
```

egmglobaldb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`egmglobaldb memory set` コマンドは、グローバル データベース用の Enterprise Geocoding モジュールで使用されるメモリ サイズを定義します。このコマンドを使うには、Global Geocoding モジュール (グローバル) がインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`egmglobaldb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

> **注：** パラメータのリストを表示するには、“`help egmglobaldb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`egmglobaldb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、第 2 四半期用データベースのデータベース メモリ サイズを設定します。

```
egmglobaldb memory set --name egmglobalq2 --mn 1200 --mx 65536
```

egmglobaldb poolsize set

コマンド `egmglobaldb poolsize set` は、設定されているグローバル データベース リソースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

`egmglobaldb poolsize set--n`***Name***`--s` ***Poolsize***

必須	引数	説明
はい	`--n` ***Name***	データベースの名前を指定します。
いいえ	`--s` ***Poolsize***	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。

例

この例では、既に設定済みのグローバル データベース リソースのプール サイズを 10 に設定します。

```
egmglobaldb poolsize set --n global --s 10
```

米国データベース用の Enterprise Geocoding モジュール

egmusadb add

`egmusadb add` コマンドは、新しい US Enterprise Geocoding モジュール データベース リソースをサーバー上に作成します。このコマンドを使うには、米国用 Enterprise Geocoding モジュールがインストールされている必要があります。

使用方法

`egmusadb add --f` *file* `--mn` *minimum_memory_size* `--mx` *maximum_memory_size*

必須	引数	説明
はい	`--f` *file*	追加するデータベース リソースのディレクトリと名前を指定します。
いいえ	`--mn` or `--minMem` *minimum_memory_size*	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

egmusadb delete

コマンド `egmusadb delete` は、設定されている Enterprise Geocoding モジュールの米国データベースを削除します。

使用方法

`egmusadb delete--n` *name*

必須	引数	説明
はい	`--n` *name*	データベースの名前を指定します。

例

この例は、"EGM_CENTRUS_POINTS" という名前のデータベースを削除します。

```
egmusadb delete --n EGM_CENTRUS_POINTS
```

egmusadb import

`egmusadb import` コマンドは、`egmusadb export` コマンドにより作成された Enterprise Geocoding モジュールの US データベース プロパティ ファイルをインポートします。`egmusadb import` コマンドにより、現在のシステム上に US データベース リソースが設定されます。

使用方法

`egmusadb import --f` ***file***

必須	引数	説明
はい	`--f` *file*	JSON 形式のデータベース プロパティ ファイルを指定します。

例

この例では、JSON 形式ファイルである `EgmDbResource.txt` 内の設定の定義に従って米国データベース リソースを作成します。

```
egmusadb import --f EgmDbResource.txt
```

egmusadb export

コマンド `egmusadb export` は、すべての米国データベース リソース情報を、指定された場所にあるデータベース プロパティ ファイル `EgmDbResource.txt` にエクスポートします。出力ファイルの場所が指定されていない場合は、EgmDbResource.txt ファイルは現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `egmusadb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`egmusadb export --o` ***directory***

必須	引数	説明
いいえ	`--o` *directory*	データベース プロパティ情報を含む JSON 形式の出力ファイルである EgmDbResource.txt を保存する、出力ディレクトリを指定します。

例

この例は、データベース情報を指定された場所にエクスポートします。

```
egmusadb export --o C:\DBs\
```

出力ファイル `EgmDbResource.txt` には、次のようなデータベース プロパティ情報が含まれます。

```
[{"product":"GeoStan",
"module":"geostan",
"name":"TomTomStreets",
"maxActive":4,
"properties":{"BASE_DB_PATHS":"C:/Dataset/DVDGDT",
"DataSetName":"TomTomStreets"}},
{"product":"GeoStan",
"module":"geostan",
"name":"CentrusPoints",
"maxActive":4,
"properties":{"BASE_DB_PATHS":"C:/Dataset/DVDCPoints;C:/Dataset/DVDGDT",
"DataSetName":"CentrusPoints"}}]
```

egmusadb get

コマンド `egmusadb get` は、US Enterprise Geocoding モジュール データベースに関する情報を返します。

使用方法

`egmusadb get--n` ***name***

必須	引数	説明
はい	`--n` *name*	データベースの名前を指定します。

例

この例は、"EGM_CENTRUS_PTS" という名前のデータベースの情報を取得します。

```
egmusadb get --n EGM_CENTRUS_PTS
```

次のような情報が返されます。

```
DATABASE NAME = EGM_CENTRUS_PTS
POOL SIZE = 4
BASE_DB_PATH = C:\DBs\EGM\CENTRUS_JUL14
DataSetName = EGM_CENTRUS_PTS
```

egmusadb list

コマンド `egmusadb list` は、設定されているすべての US Enterprise Geocoding モジュール データベースと、そのプール サイズを表示します。

使用方法

`egmusadb list` このコマンドにはプロパティはありません。

例

この例は、Enterprise Geocoding モジュールの米国データベースをすべて一覧表示します。

```
egmusadb list
```

次のような情報が返されます。

```
+-----------------+----------+
|  DATABASE NAME  |POOL SIZE|
+-----------------+----------+
| TomTomStreets   |        4 |
| TomTomPoints    |        4 |
| NAVTEQStreets   |        4 |
| CentrusPoints   |        4 |
+-----------------+----------+
```

egmusadb memory set

> **注：** 管理ユーティリティをインストールして実行する手順については、「**管理ユーティリティを使用する前に**（288ページ）」を参照してください。

`egmusadb memory set` コマンドは、米国データベース用の Enterprise Geocoding モジュールで使用されるメモリ サイズを定義します。このコマンドを使うには、Enterprise Geocoding for USA モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`egmusadb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、“`help egmusadb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` *database_name*	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`egmusadb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` *minimum_memory_size*	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、Enterprise Geocoding for USA モジュール データベースのデータベース メモリ サイズを設定します。

```
egmusadb memory set --name EGMMUS --mn 1200 --mx 65536
```

egmusadb poolsize set

コマンド `egmusadb poolsize set` は、設定されている AUS Enterprise Geocoding モジュール データベースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

`egmusadb poolsize set --n` *name* `--s` *poolsize*

必須	引数	説明
はい	`--n` *name*	データベースの名前を指定します。
いいえ	`--s` *poolsize*	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。

例

この例では、"EGM_CENTRUS_PTS" データベースのプール サイズを '3' に設定します。

```
egmusadb poolsize set --n EGM_CENTRUS_PTS --s 3
```

世界データベース用の Enterprise Geocoding モジュール

egmworlddb create_sample_file

コマンド `egmworlddb create_sample_file` は、1 つまたは 2 つのデータベース リソースのサンプル JSON ファイルを作成します。生成されたこれらのファイルは、データベース リソースを作成するための設定の参考として使用できます。このコマンドにより、現在のディレクトリまたは指定したディレクトリに、JSON ファイル *egmWorldSingleDictDbResource.txt* および *egmWorldDoubleDictDbResource.txt* が作成されます。

使用方法

`egmworlddb create_sample_file`***outputpath***

必須	引数	説明
いいえ	*outputpath*	すべてのデータベース リソースのサンプル JSON ファイルは、指定した出力パスに作成されます。指定しない場合は、現在のフォルダにエクスポートされます。

例

この例では、World データベース リソースのサンプル JSON ファイルを現在のフォルダに作成します。2 つめの例では、すべてのデータベース リソースが C:\OutputFolder にエクスポートされます。

```
egmworlddb create_sample_file
```

```
egmworlddb create_sample_file C:\OutputFolder
```

データベース リソースのサンプル JSON ファイル

```
[{"product":"InternationalGeocoder WORLD",
"module":"igeocode-world", "name":"$$DATABASE_NAME$$",
"maxActive":4,
"properties":{"COUNTRY_CODE1":"$$COUNTRY_CODE1$$",
```

```
"$$COUNTRY_CODE1$$_DICTIONARY_PATH1":"$$DICTIONARY_PATH1$$",
"COUNTRY_COUNT":"1",
"$$COUNTRY_CODE1$$_DICTIONARY_PATH_NAME1":"$$DICTIONARY_PATH_NAME1$$"}}]
```

egmworlddb delete

コマンド `egmworlddb delete` は、設定されている Enterprise Geocoding モジュールの World データベースを削除します。

使用方法

`egmworlddb delete--n`*Name*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。

例

この例では、World モジュールから World データベースを削除します。

```
egmgbrdb delete --n world
```

egmworlddb import

コマンド `egmworlddb import` は、Enterprise Geocoding モジュールの World データベース プロパティ ファイルをインポートします。これによって、現在のシステム上で World データベース リソースが設定されます。

使用方法

`egmworlddb import--f`*File*

必須	引数	説明
はい	`--f` *File*	JSON 形式のデータベース プロパティ ファイルを指定します。このファイルは必須です。

例

この例では、JSON 形式ファイルである `egmGlobalSingleDictDbResource.txt` 内の設定の定義に従って World データベース リソースを作成します。

```
egmglobaldb import --f egmWorldSingleDictDbResource.txt
```

egmworlddb export

`egmworlddb export` コマンドは、すべての World データベース リソース情報を、指定された場所にあるデータベース プロパティ ファイル `EgmWorldDbResource.txt` にエクスポートします。出力ファイルの場所が指定されていない場合、`EgmWorldDbResource.txt` ファイルは現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `egmworlddb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`egmworlddb export--o`*出力パス*

必須	引数	説明
いいえ	`--o` *outputpath*	すべてのデータベース リソースは、指定された出力パスにエクスポートされます。パスが指定されていない場合は、すべてのリソースが現在のフォルダにエクスポートされます。エクスポートされた JSON 形式の出力ファイル EgmWorldDbResource.txt には、データベース プロパティ情報が含まれます。

例

この例は、データベース リソース情報を指定された場所にエクスポートします。

```
egmworlddb export --o C:\DBs\
```

egmworlddb get

コマンド `egmworlddb get` は、Global Enterprise Geocoding モジュール データベースに関する情報を返します。

使用方法

`egmworlddb get --n` *名前*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。このファイルは必須です。

例

この例では、設定されている World データベース リソースのすべての情報を表示します。

```
egmworlddb get --n World
```

egmworldb list

コマンド `egmworlddb list` は、設定されているすべての World Enterprise Geocoding モジュール データベースと、そのプール サイズを表示します。

使用方法

`egmworlddb list` このコマンドにはプロパティはありません。

例

この例は、Enterprise Geocoding モジュールの World データベースとプール サイズを一覧表示します。

```
egmworlddb list
```

egmworlddb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`egmworlddb memory set` コマンドは、Enterprise Geocoding (世界) モジュールで使用されるメモリ サイズを定義します。このコマンドを使うには、Enterprise Geocoding (世界) モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`egmworlddb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、“`help egmworlddb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソー

必須	引数	説明
		スの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` *minimum_memory_size*	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、Enterprise Geocoding (世界) モジュールの米国データベースのデータベース メモリ サイズを設定します。

```
egmworlddb memory set --name EGMUS --mn 1200 --mx 65536
```

egmworlddb poolsize set

コマンド `egmworlddb poolsize set` は、設定されている World データベース リソースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

`egmworlddb poolsize set--n`*Name*`--s` *Poolsize*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。
いいえ	`--s` *Poolsize*	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。

例

この例では、既に設定済みのグローバル データベース リソースのプール サイズを 10 に設定します。

```
egmworlddb poolsize set --n world --s 10
```

Enterprise Tax モジュール データベース

geotaxdb delete

`geotaxdb delete` コマンドは、設定されている Enterprise Tax モジュール データベースを削除します。

使用方法

`geotaxdb delete--n` ***name***

必須	引数	説明
はい	`--n` ***name***	データベースの名前を指定します。

例

この例は、"ETM_CENTRUS_POINTS" という名前のデータベースを削除します。

```
geotaxdb delete --n ETM_CENTRUS_POINTS
```

geotaxdb import

`geotaxdb import` コマンドは、`geotaxdb export` コマンドで作成された Enterprise Tax モジュールのデータベース プロパティ ファイルをインポートします。`geotaxdb import` コマンドにより、現在のシステム上でデータベース リソースが設定されます。

使用方法

`geotaxdb import --f` ***file***

必須	引数	説明
はい	`--f` ***file***	JSON 形式のデータベース プロパティ ファイルを指定します。

例

この例は、`GeotaxDbResource.txt JSON-formatted` ファイル内の設定の定義に従って Enterprise Tax モジュールのデータベース リソースを作成します。

```
geotaxdb import --f GeotaxDbResource.txt
```

geotaxdb export

`geotaxdb export` コマンドは、すべての Enterprise Tax モジュール データベース リソース情報を、指定した場所にあるデータベース プロパティ ファイル `GeotaxDbResource.txt` にエクスポートします。出力ファイルの場所が指定されていない場合、`GeotaxDbResource.txt` ファイルは現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `geotaxdb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`geotaxdb export --o` ***directory***

必須	引数	説明
いいえ	`--o` ***directory***	データベース プロパティ情報を含む JSON 形式の出力ファイルである GeotaxDbResource.txt を保存する、出力ディレクトリを指定します。

例

この例は、データベース情報を指定された場所にエクスポートします。

```
geotaxdb export --o C:\Data\
```

出力ファイル `GeotaxDbResource.txt` には、次のようなデータベース プロパティ情報が含まれます。

```
[ {
  "product" : "Enterprise Tax Module",
  "module" : "gtx",
  "name" : "ETM_DB",
  "maxActive" : 4,
  "properties" : {
    "BASE_DB_PATH" : "C:/Datasets/DVDGTX",
    "POINTS_DB_PATH" : "C:/Datasets/DVDMLD"
  }
} ]
```

geotaxdb get

`geotaxdb get` コマンドは、Enterprise Tax モジュール データベースに関する情報を返します。

使用方法

`geotaxdb get--n` ***name***

必須	引数	説明
はい	`--n` *name*	データベースの名前を指定します。

例

この例は、"ETM_6" という名前のデータベースの情報を取得します。

```
geotaxdb get --n ETM_6
```

次のような情報が返されます。

```
DATABASE NAME = ETM_6
POOL SIZE = 4
BASE_DB_PATH = C:/Datasets/DVDGTX
POINTS_DB_PATH = C:/Datasets/DVDMLD
```

geotaxdb list

`geotaxdb list` コマンドは、設定されているすべての Enterprise Tax モジュール データベースと、そのプール サイズを表示します。

使用方法

`geotaxdb list` このコマンドにはプロパティはありません。

例

この例では、すべての Enterprise Tax モジュール データベースを一覧表示します。

```
geotaxdb list
```

次のような情報が返されます。

```
+----------------+---------+
|  DATABASE NAME |POOL SIZE|
+----------------+---------+
| TomTomStreets  |        4 |
| TomTomPoints   |        4 |
| NAVTEQStreets  |        4 |
| CentrusPoints  |        4 |
+----------------+---------+
```

geotaxdb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`geotaxdb memory set` コマンドは、Enterprise Tax モジュール データベースのメモリ サイズを定義します。このコマンドを使うには、Enterprise Tax モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が0の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`geotaxdb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

> **注：** パラメータのリストを表示するには、"`help geotaxdb memory set`" と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は0より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、ETM Centrus Points というデータベースのデータベース メモリ サイズを設定します。

```
geotaxdb memory set --name ETM_CENTRUS_POINTS --mn 1200 --mx
65536
```

geotaxdb poolsize set

`geotaxdb poolsize set` コマンドは、設定されている Enterprise Tax モジュール データベースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

```
geotaxdb poolsize set --n  name --s  poolsize
```

必須	引数	説明
はい	`--n` *name*	データベースの名前を指定します。
いいえ	`--s` *poolsize*	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。

例

この例では、"ETM_CENTRUS_PTS" データベースのプール サイズを '3' に設定します。

```
geotaxdb poolsize set --n ETM_CENTRUS_PTS --s 3
```

Global Addressing モジュールのデータベース

gamdb create

`gamdb create` コマンドは、Global Addressing モジュール データベースを作成および設定します。

使用方法

`gamdb create --n` *Name* `--d` *Dataset Name* `--v` *Dataset Vintage* `--c` *Country* `--t` *Type* `--g` *Group* `--p` *Poolsize*`--mn` *minimum_memory_size* `--mx` *maximum_memory_size*

必須	引数	説明
はい	`--n` *Name*	作成するデータベース リソースの名前を指定します。
はい	`--d` *Dataset Name*	SPD データセットの名前を指定します。
はい	`--v` *Dataset Vintage*	データセットのヴィンテージを指定します。
いいえ	`--c` *Country*	"t" オプション (SPD のタイプ) で指定されたデータベースに含める各国の 3 桁の ISO コードを指定します。Countries は、3 桁の ISO コードがセミコロンで区切られたリストです。ISO コードの詳細については、『Spectrum™ Technology Platform Addressing ガイド』を参照してください。
はい	`--t` *Type*	データセットのタイプを指定します。 **GAV** Global Address Validation データセット。

必須	引数	説明	
		GTA	Global Type Ahead データセット。
はい	`--g` *Group*	Global Address Validation のコーダーを指定します。	
		Global	Global Address Validation 国際コーダー。
		US	Global Address Validation 米国コーダー。
いいえ	`--p` *Poolsize*	このデータベースで処理する同時要求の最大数を指定します。デフォルトは 4 です。	
いいえ	`--mn` or `--minMem` *minimum_memory_size*	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。	
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。	

例

この例では、2018 年 12 月ヴィンテージおよび国際コーダーのデータベース リソース "GAV_EMEA" を使用して、"GAV_DEU" という名前のドイツ用の Global Addressing Validation データベースを作成します。ここでは GAV_DEU データベースのプール サイズを 5 に設定し、メモリの割り当てを 12200 ～ 65536 MB にします。

```
gamdb create --n GAV_DEU --d GAV_EMEA --v DEC2018 --c DEU --t
GAV --g Global --p 5 --mn 12200 --mx 65536
```

例

この例では、2018 年 12 月ヴィンテージのデータベース リソース "GTA_EMEA" を使用して、"GTA_AUT" という名前のオーストリア用の Global Type Ahead データベースを作成します。ここでは GTA_AUT データベースのプール サイズを 6 に設定し、メモリの割り当てを 12200 ～ 65536 MB にします。

```
gamdb create --n GTA_AUT --d GTA_EMEA --v DEC2018 --c AUT --t
GTA --p 6 --mn 12200 --mx 65536
```

gamdb delete

`gamdb delete` コマンドは、Global Addressing モジュール データベースを削除します。

使用方法

`gamdb delete --n` ***Name*** `--g` ***Group***

必須	引数	説明
はい	`--n` ***Name***	データベースの名前を指定します。
はい	`--g` ***Group***	Global Address Validation のコーダーを指定します。 **Global** Global Address Validation 国際コーダー。 **US** Global Address Validation 米国コーダー。

例

この例は、"GAV_DEU" という名前のドイツ用の Global Address Validation データベースを削除します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb delete --n GAV_DEU --g Global
```

例

この例は、"GTA_AUT" という名前のオーストリア用の Global Type Ahead データベースを削除します。

```
gamdb delete --n GTA_AUT
```

gamdb export

`gamdb export` コマンドは、すべての Global Addressing データベース リソース情報を、データベース プロパティ ファイル `GlobalAddressingDbResource.txt` にエクスポートします。`GlobalAddressingDbResource.txt` ファイルは、指定された場所、または出力ファイルの場所が指定されていない場合は現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `gamdb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`gamdb export --o` ***outputpath*** `--g` ***Group***

必須	引数	説明
いいえ	`--o` ***outputpath***	Global Addressing データベース リソースに関する情報は、指定された出力ディレクトリの `GlobalAddressingDbResource.txt` にエクスポートされます。パスが指定されない場合は、現在のフォルダに `GlobalAddressingDbResource.txt` が書き出されます。

必須	引数	説明	
はい	`--g` *Group*	Global Address Validation コーダーを指定します。	
		Global	Global Address Validation 国際コーダー。
		US	Global Address Validation 米国コーダー。

例

この例では、Global Addressing データベース リソース情報を指定されたディレクトリにエクスポートします。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb export --o C:\DBs\ --g Global
```

gamdb get info

`gamdb get info` コマンドは、Global Addressing データベースに関する詳細情報を返します。

使用方法

`gamdb get info --n` *Name* `--g` *Group*

必須	引数	説明	
はい	`--n` *Name*	データベースの名前を指定します。	
はい	`--g` *Group*	Global Address Validation のコーダーを指定します。	
		Global	Global Address Validation 国際コーダー。
		US	Global Address Validation 米国コーダー。

例

この例は、ドイツに対して設定されている Global Addressing Validation データベースのすべての情報を表示します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb get info --n GAV_DEU --g Global
```

次のような情報が返されます。

```
DATABASE NAME = GAV_DEU
POOL SIZE = 5
BASE_DB_PATH = C:\DBs\DEU\
```

例

この例は、Global Addressing Validation 用のテーブル内の情報を返します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb get info --n GAV --g Global
```

次のような情報が返されます。

```
+--------------+---------+---------+
|    SPDNAME   | SPDTYPE | COUNTRY |
+--------------+---------+---------+
|     GAV_APAC |     GAV |     ALL |
|     GAV_EMEA |     GAV |     ALL |
| GAV_AMERICAS |     GAV |     ALL |
+--------------+---------+---------+
```

例

この例は、オーストリアに対して設定されている Global Type Ahead データベースのすべての情報を表示します。

```
gamdb get info --n GTA_AUT
```

次のような情報が返されます。

```
DATABASE NAME = GAV_AUT
POOL SIZE = 6
BASE_DB_PATH = C:\DBs\AUT\
```

例

この例は、Global Type Ahead 用のテーブル内の情報を返します。

```
gamdb get info --n GTA
```

次のような情報が返されます。

```
+--------------+---------+---------+
|    SPDNAME   | SPDTYPE | COUNTRY |
+--------------+---------+---------+
|     GTA_APAC |     GTA |     ALL |
|     GTA_EMEA |     GTA |     ALL |
| GTA_AMERICAS |     GTA |     ALL |
+--------------+---------+---------+
```

gamdb import

`gamdb import` コマンドは、Global Addressing データベース プロパティ ファイルをインポートします。このファイルは、現在のシステム上にあるデータベース リソースを設定します。

使用方法

`gamdb import --f` ***File***

必須	引数	説明
はい	`--f` ***File***	JSON 形式のデータベース プロパティ ファイルを指定します。このファイルは必須です。

例

この例では、JSON 形式ファイルである `GlobalAddressingDbResource.txt` 内の設定の定義に従って Global Addressing データベース リソースを作成します。

```
gamdb import --f GlobalAddressingDbResource.txt
```

gamdb listdatasets

`gamdb listdatasets` コマンドは、プラットフォーム上に登録されている Global Addressing モジュール データベースを表示します。

使用方法

`gamdb listdatasets` このコマンドにはプロパティはありません。

例

この例は、プラットフォーム上に登録されている Global Addressing モジュール データベースを一覧表示します。

```
gamdb listdatasets
```

gamdb listdbresources

`gamdb listdbresources` コマンドは、設定されているすべての Global Addressing モジュール データベースと、各データベースのプール サイズを表示します。

使用方法

`gamdb listdbresources --g` ***Group***

必須	引数	説明
はい	`--g` *Group*	Global Address Validation のコーダーを指定します。 **Global** Global Address Validation 国際コーダー。 **US** Global Address Validation 米国コーダー。

例

この例は、Global Addressing モジュール データベースと、各データベースのプール サイズを一覧表示します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb listdbresources --g Global
```

gamdb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`gamdb memory set` コマンドは、 Global Addressing モジュール データベースで使用されるメモリ サイズを定義します。このコマンドを使うには、Global Addressing モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`gamdb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、“`help gamdb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。

必須	引数	説明
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、EMEA 用の Global Addressing モジュール データベースのデータベース メモリ サイズを設定します。

```
gamdb memory set --n GAV_EMEA --mn 1200 --mx 65536
```

gamdb modify

`gamdb modify` コマンドは、Global Addressing モジュール データベースを変更および更新します。

使用方法

`gamdb modify --n` *Name* `--d` *Dataset Name* `--v` *Dataset Vintage* `--c` *Country* `--t` *Type* `--g` *Group* `--p` *Poolsize*

必須	引数	説明
はい	`--n` *Name*	変更するデータベース リソースの名前を指定します。
はい	`--d` *Dataset Name*	SPD データセットの名前を指定します。
はい	`--v` *Dataset Vintage*	データセットのヴィンテージを指定します。
いいえ	`--c` *Country*	"t" オプション (SPD のタイプ) で指定されたデータベースに含める各国の 3 桁の ISO コードを指定します。Countries は、3 桁の ISO コードがセミコロンで区切られたリストです。ISO コードの詳細については、『Spectrum™ Technology Platform Addressing ガイド』を参照してください。
はい	`--t` *Type*	データセットのタイプを指定します。 **GAV** Global Address Validation データベース。 **GTA** Global Type Ahead データベース。
はい	`--g` *Group*	Global Address Validation のコーダーを指定します。 **Global** Global Address Validation 国際コーダー。 **US** Global Address Validation 米国コーダー。

必須	引数	説明
いいえ	`--p` *Poolsize*	このデータベースで処理する同時要求の最大数を指定します。デフォルトは 4 です。

例

この例は、"GAV_DEU" という名前のドイツ用の Global Addressing Validation データベースのプール サイズを変更します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb modify --n GAV_DEU --d GAV_EMEA --v DEC2018 --c DEU --t
GAV --g Global --p 6
```

例

この例は、"GTA_AUT" という名前のオーストリア用の Global Type Ahead データベースのプール サイズを変更します。

```
gamdb modify --n GTA_AUT --d GTA_EMEA --v DEC2018 --c AUT --t
GTA --p 3
```

gamdb poolsize set

`gamdb poolsize set` コマンドは、設定されている Global Addressing モジュール データベース リソースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

`gamdb poolsize set --n` *Name* `--s` *Poolsize* `--g` *Group*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。
いいえ	`--s` *Poolsize*	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。
はい	`--g` *Group*	Global Address Validation のコーダーを指定します。 **Global** Global Address Validation 国際コーダー。 **US** Global Address Validation 米国コーダー。

例

この例は、ドイツに対して既に設定されている Global Addressing Validation データベースのプール サイズを 5 に設定します。この例では、Global Address Validation 国際コーダーを指定します。

```
gamdb poolsize set --n GAV_DEU --s 5 --g Global
```

例

この例は、オーストリアに対して既に設定されている Global Type Ahead データベースのプール サイズを 7 に設定します。

```
gamdb poolsize set --n GTA_AUT --s 7
```

Global Geocoding データベース

globalgeocodedb create sample file

`globalgeocodedb create_sample_file` コマンドは、Global Geocoding データベース リソースのサンプル JSON ファイルを作成します。生成されたこれらのファイルは、データベース リソースを作成するための設定の参考として使用できます。*GeocodeGlobalSingleDictDbResource.txt* と *GlobalGeocodeDoubleDictDbResource.txt* という JSON ファイルが、現在のディレクトリまたは指定されたフォルダに作成されます。

使用方法

`globalgeocodedb create_sample_file --o` ***outputpath***

必須	引数	説明
いいえ	`--o` ***outputpath***	データベース リソースのサンプル JSON ファイルは、指定された出力ディレクトリに作成されます。出力パスが指定されない場合は、現在のフォルダにサンプル JSON ファイルが書き出されます。

例

この例では、データベース リソースのサンプル JSON ファイルを現在のフォルダに作成します。

```
globalgeocodedb create_sample_file
```

次の例では、C:\OutputFolder\にデータベース リソースの JSON ファイルを作成します。

```
globalgeocodedb create_sample_file --o C:\OutputFolder\
```

データベース リソースのサンプル JSON ファイル

```
[{"product":"GlobalGeocode",
"module":"GlobalGeocode",
"name":"$$DATABASE_NAME$$",
"maxActive":4,
"properties":
{"COUNTRY_CODE1":"$$COUNTRY_CODE1$$",
 "$$COUNTRY_CODE1$$_DICTIONARY_PATH1":"$$DICTIONARY_PATH1$$",

 "COUNTRY_COUNT":"1",

"$$COUNTRY_CODE1$$_DICTIONARY_PATH_NAME1":"$$DICTIONARY_PATH_NAME1$$"}
}]
```

globalgeocodedb delete

コマンド `globalgeocodedb delete` は、設定されている Global Geocoding データベースを削除します。

使用方法

`globalgeocodedb delete--n`*Name*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。

例

この例では、TomTomUSA データベースを削除します。

```
globalgeocodedb delete --n TomTomUSA
```

globalgeocodedb import

コマンド `globalgeocodedb import` は、Global Geocoding データベースのプロパティ ファイルをインポートします。これによって、現在のシステム上でデータベース リソースが設定されます。

使用方法

`globalgeocodedb import --f` *File*

必須	引数	説明
はい	`--f` *File*	JSON 形式のデータベース プロパティ ファイルを指定します。このファイルは必須です。

例

この例では、JSON 形式ファイルである `GlobalGeocodeDbResource.txt` 内の設定の定義に従って Global Geocoding データベース リソースを作成します。

```
globalgeocodedb import --f GlobalGeocodeDbResource.txt
```

コマンド `globalgeocodedb import` による結果としては、いくつかのケースがあり得ます。

- **ケース 1:** 指定されたルート フォルダの中のディレクトリがすべて無効。この場合、データベースは追加されません。

  ```
  spectrum> globalgeocodedb import --f ./GlobalGeocodeDbResource.txt
  /managers/GlobalGeocode/verify?rootFolder=D:/SGI_Data/
  ```

  結果は次のようになります。

  ```
  Invalid Folder locations found.
  ["D:\\SGI_Data\\IGEO-AT1"
  "D:\\SGI_Data\\IGEO-CZ1"]
  unable to add the database resource due to invalid paths
  ```

- **ケース 2:** 指定されたルート フォルダのうち、少なくとも 1 つのディレクトリが有効。この場合は、データベースが追加されます。

  ```
  spectrum> globalgeocodedb import --f ./GlobalGeocodeDbResource.txt
  /managers/GlobalGeocode/verify?rootFolder=D:/SGI_Data/GEO-DB
  ```

  結果は次のようになります。

  ```
  Invalid Folder locations found.
  ["D:\\SGI_Data\\IGEO-CZ1"]
  Database resource imported [./GlobalGeocodeDbResource.txt]
  ```

- **ケース 3:** 指定されたルート フォルダが無効であるか、存在しない。この場合は、データベースが追加されます。

  ```
  spectrum> globalgeocodedb import --f ./GlobalGeocodeDbResource.txt
  ```

  結果は次のようになります。

  ```
  unable to add the database resource due to invalid paths
  ```

globalgeocodedb export

コマンド `globalgeocodedb export` は、すべての Global Geocoding データベース リソース情報を、データベース プロパティ ファイル `GlobalGeocodeDbResource.txt` にエクスポートします。`GlobalGeocodeDbResource.txt` ファイルは、指定された場所、または出力ファイルの場所が指定されていない場合は現在のフォルダに書き出されます。その後、データベース プロパティ ファイルにコマンド `globalgeocodedb import` を使用することにより、別のシステム上のデータベースを設定できます。

使用方法

`globalgeocodedb export --o` ***outputpath***

必須	引数	説明
いいえ	`--o` ***outputpath***	Global Geocoding データベース リソースに関する情報は、指定された出力ディレクトリの `GlobalGeocodeDbResource.txt` にエクスポートされます。パスが指定されない場合は、現在のフォルダに `GlobalGeocodeDbResource.txt` が書き出されます。

例

この例では、Global Geocoding データベース リソース情報を指定されたディレクトリにエクスポートします。

```
globalgeocodedb export --o C:\DBs\
```

出力ファイル `GlobalGeocodeDbResource.txt` には、次のようなデータベース リソース情報が含まれます。

```
[{"product":"GlobalGeocode",
"module":"GlobalGeocode",
"name":"TomTomStreets",
"maxActive":4,
"properties":
{"BASE_DB_PATHS":"C:/Dataset/DVDGDT",
"DataSetName":"TomTomStreets"}},
{"product":"GlobalGeocode",
"module":"GlobalGeocode",
"name":"CentrusPoints",
"maxActive":4,
"properties":
{"BASE_DB_PATHS":"C:/Dataset/DVDCPoints;C:/Dataset/DVDGDT",
"DataSetName":"CentrusPoints"}}]
```

globalgeocodedb get

コマンド `globalgeocodedb get` は、Global Geocoding データベースに関する情報を返します。

使用方法

`globalgeocodedb get --n` *名前*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。このファイルは必須です。

例

この例では、設定されている Global Geocoding データベース リソースのすべての情報を表示します。

```
globalgeocodedb get --n CENTRUS_PTS
```

次のような情報が返されます。

```
DATABASE NAME = CENTRUS_PTS
POOL SIZE = 4
BASE_DB_PATH = C:\DBs\USA\
DataSetName = USA_POINTS
```

globalgeocodedb list

コマンド `globalgeocodedb list` は、設定されているすべての Global Geocoding データベースと、そのプール サイズを表示します。

使用方法

`globalgeocodedb list` このコマンドにはプロパティはありません。

例

この例は、Global Geocoding データベースとプール サイズを一覧表示します。

```
globalgeocodedb list
```

globalgeocodedb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`globalgeocodedb memory set` コマンドは、Global Geocoding モジュール データベースのメモリ サイズを定義します。このコマンドを使うには、Global Geocoding モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`globalgeocodedb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

> **注：** パラメータのリストを表示するには、“`help globalgeocodedb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、TomTomUSA データベースの最小と最大のデータベース メモリ サイズを設定します。

```
globalgeocodedb memory set --n TomTomUSA --mn 1200 --mx 65536
```

globalgeocodedb poolsize set

コマンド `globalgeocodedb poolsize set` は、設定されている Global Geocoding データベース リソースのプール サイズを設定します。プール サイズとは、データベースに対して許容される同時要求の最大数です。

使用方法

`globalgeocodedb poolsize set--n`*Name*`--s` *Poolsize*

必須	引数	説明
はい	`--n` *Name*	データベースの名前を指定します。
いいえ	`--s` *Poolsize*	データベースのプール サイズを設定します (整数値で指定)。デフォルトは 4 です。

例

この例では、既に設定済みの Global Geocoding データベース リソースのプール サイズを 10 に設定します。

```
globalgeocodedb poolsize set --n DEU_DB -s 10
```

Spatial モジュールとルーティング データベース

limrepo export

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`limrepo export` コマンドは、名前付きリソース (名前付きテーブルなど) を Spectrum Spatial リポジトリからローカル ファイル システムへエクスポートします。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

リソースはリポジトリのフル パスでターゲットのフォルダにエクスポートされます。例えば、`limrepo export` --s /Samples/NamedTables --o C:\export を実行すると、NamedTables フォルダ (ディレクトリ) の下の各名前付きテーブルに対して、`C:\export\Samples\NamedTables\WorldTable` などが作成されます。

注： `limrepo export` コマンドは必ず、空のフォルダを含むすべてのフォルダを再帰的にエクスポートします。

使用方法

`limrepo export--s`*SourceRepositoryPath*`--o`*OutputFilePath*

注： パラメータのリストを表示するには、“`help limrepo export`” と入力します。

必須	引数	説明
はい	`--s` or `source`	エクスポートするリソースまたはフォルダへのパスを指定します。
はい	`--o` or `output`	エクスポート先のローカル ファイル システム上のフォルダへのパスを指定します。新規フォルダでも既存フォルダでもかまいませんが、既存フォルダの場合は中身を空にしておく必要があります。空でない場合、エクスポートは失敗します。
いいえ	`--q` or `--quiet`	エクスポート中にコピー済みリソースの表示を無効にします (すなわち、クワイエット モードで動作します)。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--f` or `--fullpaths`	エクスポート元と出力先のフル パスをプリントします。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--r` or `--recursive`	サブフォルダ (指定されたエクスポート元フォルダの下にあるフォルダ) を再帰的にエクスポートします。 **true** 指定されたフォルダにあるすべてのファイルと、すべてのサブフォルダにあるファイルをエクスポートします。このフラグを指定しない場合、または値を設定しないでこのフラグを指定した場合、この動作がデフォルト設定です。 **false** 指定されたフォルダにあるファイルのみをエクスポートします (サブフォルダをエクスポートしません)。 false を指定すると、エクスポートされていないリソースを参照する名前付きリソースがエクスポート対象に含まれる可能性が高くなります。このフラグを使用するには、極めて注意する必要があります。リポ

必須	引数	説明
		ジトリのすべての関連性を把握したユーザが使用してください。
いいえ	`--c` or `--continueonerror`	エラーが発生した場合もエクスポートを続行します。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--a` or `--acl`	エクスポートするリソースの既存の権限をローカル ファイル システム上のエクスポート先ファイルでも維持します。アクセス制御リスト (ACL) は、ユーザーまたは役割が名前付きリソースに実行できる操作 (作成、表示、編集、削除など) を定義します。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。

例

この例は、リポジトリの \Samples フォルダにある名前付きリソースをローカル ファイル システムの C:\myrepository\samples にエクスポートします。

```
limrepo export --s /Samples --o C:\myrepository\samples
```

limrepo import

> **注：** 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`limrepo import` コマンドは、名前付きリソース (名前付きテーブルなど) をローカル ファイル システムから Spectrum Spatial リポジトリへインポートします。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

リソースをインポートするには、以前にリソースをエクスポートしたフォルダまたはディレクトリを指定する必要があります。例えば、`limrepo export --s /Samples/NamedTables --o C:\export` を実行すると、`C:\export\Samples\NamedTables\WorldTable` が、NamedTables フォルダ (ディレクトリ) にある名前付きテーブルごとに作成されます。リソースはリポジトリのフル パ

スでターゲットのフォルダにエクスポートされます。`limrepo import` --s C:\export を実行すると、WorldTable が`/Samples/NamedTables/WorldTable` へインポートされます。

> **注：** `limrepo import` コマンドは必ず、空のフォルダを含むすべてのフォルダを再帰的にインポートします。

インポートを行った後は多くの場合、Spectrum Spatial™ Manager を使用してその新しいパスを指すように名前付き接続を調整する必要があります。例えば、テスト インスタンスの`C:\myfiles` にインストールされていたのと同じネイティブ TAB ファイルが `E:\ApplicationData\Spectrum\Spatial\Q3` にインストールされた場合、インポート後に Spectrum Spatial™ Manager でその接続を修正する必要があります。Spectrum Spatial™ Manager で名前付き接続を編集する手順については、『*Spectrum Spatial ガイド*』の「ユーティリティ」セクションを参照してください。

> **注：** Spectrum™ Technology Platform の 12.0 以前のバージョンからエクスポーズされたサービス設定ファイルを復元するために `limrepo import` を使用すると、そのファイルが自動的に修正され、バージョン 12.0 以降と互換性が成立します (例えばリポジトリ URL が削除されます)。

使用方法

`limrepo import --s` *SourceFilePath*

> **注：** パラメータのリストを表示するには、“`help limrepo import`” と入力します。

必須	引数	説明
はい	`--s` or `source`	ローカル ファイル システムからインポートするリソースまたはフォルダへのパスを指定します。これは、ローカル ファイル システム上の以前のエクスポートに使用したルート フォルダである必要があります。
いいえ	`--q` or `--quiet`	インポート中にコピー済みリソースの表示を無効にします (すなわち、クワイエット モードで動作します)。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--u` or `--update`	同じ名前のリソースがサーバーに既に存在する場合、この既存のリソースを上書きするかどうかを指定します。

必須	引数	説明
		true インポートするリソースと同じ名前のリソースがサーバーに存在する場合、サーバーにあるリソースが上書きされます。このフラグを指定しない場合、または値を設定しないでこのフラグを指定した場合、この動作がデフォルト設定です。 **false** インポートするリソースと同じ名前のリソースがサーバーに存在する場合、リソースはインポートされません。
いいえ	`--f` or `--fullpaths`	エクスポート元と出力先のフルパスをプリントします。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--c` or `--continueonerror`	エラーが発生した場合もインポートを続行します。 このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。
いいえ	`--a` or `--acl`	以前にエクスポートされた権限を維持し、リソースのインポート中にそれらを既存の権限と統合します。アクセス制御リスト (ACL) は、各ユーザーまたは役割が名前付きリソースに実行できる操作 (作成、表示、変更、削除など) を定義します。 例えば、エクスポートするリソースに対して、あるユーザーが読み取りと書き込みの権限を持っています。このリソースのインポート時に、このユーザーがそのリソースの読み取り権限しか持っていない場合、インポートが正常に完了した後で書き込み権限が与えられます。 競合する権限は統合できず、無視されます。ターゲットのリポジトリに存在しないユーザーや役割の ACL 項目は無視されます。

必須	引数	説明
		このフラグを指定した場合のデフォルト値は true です。このフラグを指定しない場合のデフォルト値は false です。 **ヒント:** このフラグを使用する場合、エクスポート元のサーバー上のユーザが、インポート先のサーバー上にも存在する必要があります。例えば、アクセス制御が設定された "testuser" が存在し、ACL 付きのリソースをあるサーバーからエクスポートして、"testuser" が存在しない別のサーバーに、それらの名前付きリソースをインポートするとします。この場合、名前付きリソースは ACL なしでアップロードされます。

例

この例は、名前付きリソースをローカル ファイル システムの C:\myrepository\samples からインポートします。

```
limrepo import --s C:\myrepository\samples
```

limrepo mwsimport

注: 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`limrepo mwsimport` 管理ユーティリティの Spectrum™ Technology Platform により、MapInfo Pro または MapXtreme Workspace Manager で作成された MapInfo Workspace (MWS) ファイル内のマップを Spectrum Spatial リポジトリにプロビジョニングすることができます。インポートにより、名前付きマップとそのすべての従属リソース (レイヤ、テーブル、接続) が作成されます。接続には、マップ名に 'Connection' を付加した名前が付けられます。名前付きテーブルと名前付きレイヤは、サブフォルダに作成されます (それぞれ、NamedTables と NamedLayers)。

このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`limrepo mwsimport --s` *MWSFilePath* `--o` *Output* `--p` *ServerPath*

注: パラメータのリストを表示するには、“`help limrepo mwsimport`” と入力します。

必須	引数	説明
はい	`--s` or `source`	ローカル ファイル システムからインポートする MWS ファイルへのパスを指定します。
はい	`--o` or `output`	リポジトリ上の名前付きマップへのパスを指定します。すべてのリソースが、名前付きマップとして同じフォルダ内に作成されます。
はい	`--p` or `path`	データが配置されているサーバー上の場所へのファイル パスを指定します。このパス を使用して名前付き接続が作成され、この名前付き接続は、作成されるすべての名前付きテーブルによって参照されます。これらのテーブルは、その名前付き接続への相対ファイル パスを使用します 。
いいえ	`--l` or `local`	MWS にサーバー ファイル システム上に存在しないファイル システムが含まれる場合に、データが配置されているローカル ファイル システム上の場所へのファイル パスを指定します 。指定した値が MWS ファイル内に出現すると、それが指定したサーバー パスで置き換えられます。 MWS ファイルの中に部分パスがある場合は、これは不要です。MapXtreme で作成されたものならば通常、MWS ファイルの中に部分パスがあります。

例

この例では、D: ドライブ上の MWS ファイルをインポートし (サーバー上のデータは C:\mydata に存在します)、名前付きリソースをリポジトリ内の /Europe/Countries に配置します。

```
limrepo mwsimport --s D:\europe.mws --o /Europe/Countries --p
C:\mydata
```

結果

以下の名前付きリソースが作成されます。

/Europe/Countries/Europe *(名前付きマップ)*
/Europe/Countries/EuropeConnection *(名前付き接続)*
/Europe/Countries/NamedTables/austria *(名前付きテーブル*
/Europe/Countries/NamedTables/belgium *(名前付きテーブル)*
.
/Europe/Countries/NamedLayers/austria *(名前付きレイヤ)*

/Europe/Countries/NamedLayers/belgium *(名前付きレイヤ)*
..

ermdb list

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb list` コマンドは、サーバー上の既存のルーティング データベース リソースの全一覧を取得します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

```
ermdb list
```

例

この例では、サーバー上のすべてのデータベース リソースの全一覧を取得します。

```
ermdb list
```

ermdb get

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb get` コマンドでは、サーバー上に設定されているルーティング データベースに関する情報を取得することができます。返される情報は、データベースの名前、ファイル システム (パス) 上のデータベースの場所、データベース用に設定されているプール サイズです。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`ermdb get--name`***database_name***

注： パラメータのリストを表示するには、"`help ermdb get`" と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	情報を返すデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。

例

この例は、データベース リソース US の情報をサーバーから取得します。

```
ermdb get --name US
```

ermdb add

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb add` コマンドは、新しいルーティング データベース リソースをサーバー上に作成します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： `ermdb add` コマンドでは、追加する各データベースに対して一意の名前を使用する必要があります。

使用方法

`ermdb add --name` ***database_name*** `--poolsize` ***pool_size*** `--path` ***database_path*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、“`help ermdb add`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--poolsize` or `--s` ***pool_size***	データベースで処理する同時要求の最大数を指定します。指定しない場合のデフォルト値は 4 です。同時要求数としては、1 ～ 128 の任意の整数が指定可能です。
はい	`--path` ***database_path***	ファイル サーバー上のルーティング データベースの場所を指定します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、データベース リソース US を `E:/ERM-US/2019.09/driving/south` からサーバーに追加します。

```
ermdb add --name US --poolsize 10 --path
E:/ERM-US/2019.09/driving/south --mn 1200 --mx 65536
```

ermdb delete

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb delete` コマンドは、サーバー上の既存のルーティング データベース リソースを削除します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`ermdb delete --name` ***database_name***

注： パラメータのリストを表示するには、“`help ermdb delete`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	削除するデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。

例

この例では、データベース リソース US をサーバーから削除します。

```
ermdb delete --name US
```

ermdb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb memory set` コマンドは、ルーティング データベースのメモリ サイズを定義します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定され

ません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`ermdb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、“`help ermdb memory set`” と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、Spatial モジュール US データベースのメモリ サイズを設定します。

```
ermdb memory set --name ERM-US --mn 1200 --mx 65536
```

ermdb modify

注： 管理ユーティリティをインストールして実行する手順については、「**管理ユーティリティを使用する前に**（288ページ）」を参照してください。

`ermdb modify` コマンドは、サーバー上の既存のルーティング データベース リソースを変更します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`ermdb modify --name` ***database_name*** `--poolsize` ***pool_size*** `--path` ***database_path***

注： パラメータのリストを表示するには、“`help ermdb modify`” と入力します。

必須	引数	説明
はい	`--name` or `--n` *database_name*	変更するデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--poolsize` or `--s` *pool_size*	データベースで処理する同時要求の最大数を指定します。同時要求数としては、1 ～ 128 の任意の整数が指定可能です。新しいプール サイズと新しいデータベース パスのいずれかを指定する必要があります。
いいえ	`--path` *database_path*	ファイル サーバー上のルーティング データベースの新しい場所を指定します。新しいプール サイズと新しいデータベース パスのいずれかを指定する必要があります。

例

この例では、新しいヴィンテージのプール サイズとデータベース パスの両方を変更します。

```
ermdb modify --name US --poolsize 20 --path
E:/ERM-US/2015.03/driving/south
```

ermdb template

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb template` コマンドにより JSON ファイル (.json) が作成され、このファイルが Spectrum 製品データ (SPD) のインポート ファイルのテンプレートになります。`ermdb import` コマンドを使用する前にこのコマンドを使用します。これらコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`ermdb template` *path_to_JSON*

必須	引数	説明
いいえ	*path_to_JSON*	JSON ファイルへのパスを指定し、そこからテンプレートの (.json) ファイルを生成します。デフォ

必須	引数	説明
		ルトでは、`ermdb template` コマンドで現在の作業フォルダに JSON ファイルが生成されます。

例

この例では、Spectrum 製品データ (SPD) ファイルを Spectrum のデータベースにインポートするための、情報のテンプレートとして機能する .json ファイルが作成されます。この例では、テンプレートを現在の作業フォルダに作成します。

```
ermdb template
```

テンプレートを特定の場所にエクスポートするには、コマンドにファイル名を含めます。

```
ermdb templateC:/Downloads/File/A1T_Pedestrian_Mar_2019.json
```

作成されたテンプレート ファイル (.json ファイル) には、以下の内容が含まれます。

```
[{"product":"Spatial",
"module":"routing",
"name":"enter database name",
"properties":{"isSPD":"true",
"DatasetPaths":"${spectrum.spd.Spatial/routing/add IDENTIFIER
 from productdata list};"},
"maxActive":4}
```

SPD テンプレートに入力する `IDENTIFIER` を見つけるには、`productdata list` コマンドを実行し、インポートする SPD ファイルの情報を一覧表示します。インポートする特定の SPD ファイルの `IDENTIFIER` を見つけて、データベース名および `IDENTIFIER` 情報を追加して SPD テンプレートに入力します。

例えば、次のデータベースの名前が "Austria" (データにはどのような表示名でも入力可能です) で、`IDENTIFIER` が "A1T_Pedestrian_Mar_2019" とします。

```
[{"product":"Spatial",
  "module":"routing",
  "name":"Austria",
  "properties":{
     "isSPD":"true",

"DatasetPaths":"${spectrum.spd.Spatial/routing/A1T_Pedestrian_Mar_2019};"},

  "maxActive":4}]
```

以上で、ermdb import コマンドを実行してこれらのルーティング データの設定をインポートし、Spectrum サーバー上に "Austria" というデータベース リソースを作成する準備が完了しました。

ermdb import

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

ermdb import コマンドでは、ルーティング データベース設定を含むファイルをインポートして、データベース リソースをサーバー上に作成できます。インポート ファイルを作成するには、ermdb template コマンド、または ermdb export コマンドで作成したファイルを使用します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

インポート ファイルの形式は以下のとおりです。

```
[
{
    "product": "Spatial",
    "module": "routing",
    "name": "US",
    "maxActive": 4,
    "properties":
        {
            "DatasetPaths": "E:/ERM-US/2014.09/driving/northeast"
        }
    }
]
```

ここで、product と module はそれぞれ Spatial と routing でなければなりません。name はデータベース名、maxActive はこのデータベースで処理する同時リクエストの最大数 (またはプール サイズ)、DatasetPaths はデータベース リソースのデータ セットへのパスです。

1 つのインポート ファイルに複数のデータベースを追加できます (上の例をコピーします)。また、各データベース リソースに対して複数のデータセットをセミコロンで区切って追加できます。

注： インポート ファイルで UTF-8 文字を指定するには、CLI コマンド プロンプトの起動時に、値 UTF-8 を設定した JVM パラメータ file.encoding を追加する必要があります。例: `-Dfile.encoding=UTF-8`

使用方法

`ermdb import --file` ***file_name***

注： パラメータのリストを表示するには、“`help ermdb import`” と入力します。

必須	引数	説明
はい	`--file` or `--f` *file_name*	インポートファイルのディレクトリと名前を指定します。

例

この例では、それぞれ複数のデータセットで構成される、US1 と US2 という 2 つのデータベースをインポートします。

```
ermdb import --file E:/ERM-US/export/ermDbResource.txt
```

入力ファイルは次のように定義されています。

```
[
{
        "product": "Spatial",
        "module": "routing",
        "name": "US1",
        "maxActive": 4,
        "properties":
        {
            "DatasetPaths":
"E:/ERM-US/2014.09/driving/northeast;E:/ERM-US/2014.09/driving/south"

        }
    },
{
        "product": "Spatial",
        "module": "routing",
        "name": "US2",
        "maxActive": 4,
        "properties":
        {
            "DatasetPaths":
"E:/ERM-US/2014.09/driving/northeast;E:/ERM-US/2014.09/driving/central"


        }
    }
]
```

ermdb export

注：管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`ermdb export` コマンドでは、サーバー上に設定されているルーティング データベースをファイルにエクスポートできます。このファイルはその後、バックアップや、あるインスタンスから別のインスタンスへの移行を目的とした、`ermdb import` コマンドによる別のインスタンスへのインポートに使用できます。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： `ermdb export` コマンドは必ず、`ermDbResource.txt` という名前のエクスポート ファイルを作成します。

使用方法

`ermdb export--directory`*directory_name*

注： パラメータのリストを表示するには、“`help ermdb export`” と入力します。

必須	引数	説明
いいえ	`--directory` or `--o` *directory_name*	データベース ファイルのエクスポート先となるファイル システム上のディレクトリ名を指定します。このエクスポート コマンドは必ず、`ermDbResource.txt` という名前のエクスポート ファイルを作成します。このパラメータを指定しない場合は、このエクスポート コマンドを実行したディレクトリにエクスポート ファイルが作成されます。

例

この例では、`E:/ERM-US/export` ディレクトリにエクスポート データベース ファイルを作成します。

```
ermdb export --directory E:/ERM-US/export
```

erm getpointdata

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm getpointdata` コマンドは、ポイントに対するセグメント情報を返します。指定されたポイントに最も近い 1 つ以上のセグメントが返されます。返される情報のタイプとしては、セグメント ID、道路タイプ、長さ、速度、方向、時間、道路名があります。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

```
erm getpointdata --datasource db_resource --point "x,y,coordsys"
```

注： パラメータのリストを表示するには、“`help erm getpointdata`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	データを返すデータベースリソースの名前を指定します。既存のルーティングデータベースリソースの一覧を表示するには、`ermdb list` コマンドを使用します。
はい	`--point` *"x,y,coordsys"*	最も近いセグメントの情報を返すポイントを指定します。ポイントは、*"x,y,coordsys"* という形式で指定します。ここで、*coordsys* はポイントの座標系です。

例

この例では、サーバー上に設定されている US_NE データベースリソースから、指定されたポイントに最も近いセグメントのデータを返します。

```
erm getpointdata --datasource US_NE --point "-72,40,epsg:4326"
```

erm getsegmentdata

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm getsegmentdata` コマンドは、指定されたセグメント ID に対するセグメント情報を返します。返される情報のタイプとしては、セグメント ID、道路タイプ、長さ、速度、方向、時間、道路名があります。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

```
erm getsegmentdata --datasource db_resource --segmentid "segment_id"
```

注： パラメータのリストを表示するには、“`help erm getsegmentdata`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	データを返すデータベースリソースの名前を指定します。既存のルーティングデータベースリソー

必須	引数	説明
		スの一覧を表示するには、`ermdb list` コマンドを使用します。
はい	`--segmentid` *"segment_id"*	情報を返すセグメントを指定します。セグメントは、データ内で指定されている形式で指定します。例: *"7e3396fc:6e5251"*。

例

この例では、サーバー上に設定されている US_NE データベース リソースから、指定されたセグメントのデータを返します。

```
erm getsegmentdata --datasource US_NE --segmentid
"7e3396fc:6e5251"
```

erm createpointupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm createpointupdate` コマンドは、指定されたポイントに最も近いセグメントのルーティング データをオーバーライドします。このコマンドにより、速度を設定または変更したり、ルートのセクションを除外したりできます。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： 永続更新タイプは特定のデータ リソースに対してのみ有効で、データ更新後には有効でなくなる場合があります。

使用方法

`erm createpointupdate --datasource` *db_resource* `--point` *"x,y,coordsys"* `--exclude --velocity` *velocity_value* `--velocityunit` *velocity_unit* `--velocityadjustment` *velocity_adjustment_value* `--velocitypercentage` *velocity_percentage_value*

注： パラメータのリストを表示するには、“`help erm createpointupdate`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	データをオーバーライドするデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。

必須	引数	説明
はい	`--point` *"x,y,coordsys"*	最も近いセグメントの情報をオーバーライドするポイントを指定します。ポイントは、*"x,y,coordsys"* という形式で指定します。ここで、*coordsys* はポイントの座標系です。
いいえ	`--exclude`	`true` に設定された場合、指定されたポイントをすべてのルート計算から除外します。コマンドにこのパラメータを含めることにより、そのポイントを除外するかどうかを指定します。除外しない場合は、--exclude の後に `false` を追加します。
いいえ	`--velocity` *velocity_value*	速度更新を定義します。新しい速度を指定することにより、ポイントの新しい速度を指定します。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocityunit` *velocity_unit*	`velocity` または `velocityadjustment` オーバーライドの速度単位を定義します。デフォルトは mph (マイル/時) です。速度更新の場合、速度単位には kph (キロメートル/時)、mps (メートル/秒)、mph (マイル/時) のいずれかの値を指定できます。
いいえ	`--velocityadjustment` *velocity_adjustment_value*	速度更新を定義します。速度 (単位と値) の変更を指定することにより、ポイントの速度の変化を定義します。速度値は増加 (正の値) または減少 (負の値) させることができます。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocitypercentage` *velocity_percentage_value*	速度更新を定義します。速度を増加 (正の値) または減少 (負の値) させる割合を指定することにより、ポイントの速度の増加を定義します。

例

この例では、サーバー上に設定されている US_NE データベース リソースから、ポイントの速度を 15 mph にオーバーライドします。

```
erm createpointupdate --datasource US_NE --point
"-72,40,epsg:4326" --velocity 15 --velocityunit mph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、指定されたポイントを除外します。

```
erm createpointupdate --datasource US_NE --point
"-72,40,epsg:4326" --exclude true
```

この例では、サーバー上に設定されている US_NE データベースリソースから、ポイントの速度を 45 kph だけ増加させてオーバーライドします。

```
erm createpointupdate --datasource US_NE --point
"-72,40,epsg:4326" --velocityadjustment 45 --velocityunit kph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、ポイントの速度を 60% だけ減少させてオーバーライドします。

```
erm createpointupdate --datasource US_NE --point
"-72,40,epsg:4326" --velocitypercentage -60
```

erm resetpointupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm resetpointupdate` コマンドは、データの元の状態に対するオーバーライドがあれば、それを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`erm resetpointupdate--datasource`***db_resource***`--point`***"x,y,coordsys"***`--resettype`***reset_type***

注： パラメータのリストを表示するには、“`help erm resetpointupdate`” と入力します。

必須	引数	説明
はい	`--datasource` ***db_resource***	オーバーライドを持つデータベースリソースの名前を指定します。既存のルーティングデータベースリソースの一覧を表示するには、`ermdb list` コマンドを使用します。
はい	`--point` ***"x,y,coordsys"***	既存のオーバーライドが位置するポイントを指定します。ポイントは、***"x,y,coordsys"*** という形式で指定します。ここで、***coordsys*** はポイントの座標系です。

必須	引数	説明
はい	`--resettype` *reset_type*	削除 (取り消し) するオーバーライドのタイプです。 **速度** 速度更新を削除します。 **除外** 除外更新を削除します。

例

この例では、指定されたポイントの既存の除外オーバーライドを、サーバー上に設定されている US_NE データベース リソースからリセットします。

```
erm resetpointupdate --datasource US_NE --point
"-72,40,epsg:4326" --resettype exclude
```

erm createsegmentupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm createsegmentupdate` コマンドは、指定されたセグメントのルーティング データをオーバーライドします。このコマンドにより、速度を設定または変更したり、ルートのセクションを除外したり、道路タイプを変更したりできます。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： 永続更新タイプは特定のデータ リソースに対してのみ有効で、データ更新後には有効でなくなる場合があります。

使用方法

`erm createsegmentupdate --datasource` *db_resource* `--segmentid` *"segment_id"* `--exclude --velocity` *velocity_value* `--velocityunit` *velocity_unit* `--velocityadjustment` *velocity_adjustment_value* `--velocitypercentage` *velocity_percentage_value* `--roadtype` *road_type*

注： パラメータのリストを表示するには、“`help erm createsegmentupdate`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	データをオーバーライドするデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。

必須	引数	説明
はい	`--segmentid` *"segment_id"*	オーバーライドするセグメントを指定します。セグメントは、データ内で指定されている形式で指定します。例: *"7e3396fc:6e5251"*。
いいえ	`--exclude`	`true` に設定されている場合、指定されたセグメントをすべてのルート計算から除外します。コマンドにこのパラメータを含めることにより、そのセグメントを除外するかどうかを指定します。除外しない場合は、--exclude の後に `false` を追加します。
いいえ	`--velocity` *velocity_value*	速度更新を定義します。新しい速度を指定することにより、セグメントの新しい速度を指定します。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocityunit` *velocity_unit*	`velocity` または `velocityadjustment` オーバーライドの速度単位を定義します。値は mph (マイル/時) です。速度更新の場合、速度単位には kph (キロメートル/時)、mps (メートル/秒)、mph (マイル/時) のいずれかの値を指定できます。
いいえ	`--velocityadjustment` *velocity_adjustment_value*	速度更新を定義します。速度 (単位と値) の変更を指定することにより、セグメントの速度の変化を定義します。速度値は増加 (正の値) または減少 (負の値) させることができます。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocitypercentage` *velocity_percentage_value*	速度更新を定義します。速度を増加 (正の値) または減少 (負の値) させる割合を指定することにより、セグメントの速度の増加を定義します。
いいえ	`--roadtype` *road_type*	セグメントの新しい道路タイプを定義します。

例

この例では、サーバー上に設定されている US_NE データベース リソースから、セグメントの速度を 15 mph にオーバーライドします。

```
erm createsegmentupdate --datasource US_NE --segmentid
"7e3396fc:6e5251" --velocity 15 --velocityunit mph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、指定されたセグメントを除外します。

```
erm createsegmentupdate --datasource US_NE --segmentid
"7e3396fc:6e5251" --exclude true
```

この例では、サーバー上に設定されている US_NE データベースリソースから、セグメントの速度を 45 kph だけ増加させてオーバーライドします。

```
erm createsegmentupdate --datasource US_NE --segmentid
"7e3396fc:6e5251" --velocityadjustment 45 --velocityunit kph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、セグメントの速度を 60% だけ減少させてオーバーライドします。

```
erm createsegmentupdate --datasource US_NE --segmentid
"7e3396fc:6e5251" --velocitypercentage -60
```

この例では、サーバー上に設定されている US_NE データベースリソースから、セグメントの道路タイプをフェリーにオーバーライドします。

```
erm createsegmentupdate --datasource US_NE --segmentid
"7e3396fc:6e5251" --roadtype ferry
```

erm resetsegmentupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm resetsegmentupdate` コマンドは、データの元の状態に対するオーバーライドがあれば、それを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`erm resetsegmentupdate --datasource` ***db_resource*** `--segmentid` ***"segment_id"*** `--resettype` ***reset_type***

注： パラメータのリストを表示するには、“`help erm resetsegmentupdate`” と入力します。

必須	引数	説明
はい	`--datasource` ***db_resource***	オーバーライドを持つデータベースリソースの名前を指定します。既存のルーティングデータベースリソースの一覧を表示するには、`ermdb list` コマンドを使用します。

必須	引数	説明
はい	`--segment` *"segment_id"*	既存のオーバーライドが位置するセグメントを指定します。セグメントは、データ内で指定されている形式で指定します。例: *"7e3396fc:6e5251"*。
はい	`--resettype` *reset_type*	削除 (取り消し) するオーバーライドのタイプです。 **速度** 速度更新を削除します。 **除外** 除外更新を削除します。 **roadType** 道路タイプ更新を削除します。

例

この例では、指定されたセグメントの既存の道路タイプ オーバーライドを、サーバー上に設定されている US_NE データベース リソースからリセットします。

```
erm resetsegmentupdate --datasource US --segmentid
"7e3396fc:6e5251" --resettype roadtype
```

erm getsegmentupdates

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm getsegmentupdates` コマンドは、指定されたセグメントのルーティング データにおけるオーバーライドのリストを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： `segmentids` はオプションのパラメータです。セグメント ID を指定しない場合は、使用可能なすべてのセグメントに対するオーバーライドが返されます。

使用方法

`erm getsegmentupdates--datasource`*db_resource*`--segmentids`*"segment_ids"*`--velocityunit`*velocityunit*

注： パラメータのリストを表示するには、“`help erm getsegmentupdates`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	オーバーライドを持つデータベース リソースの名前を指定します。既存のルーティング データベー

必須	引数	説明
		スリソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--segmentids` *"segment_ids"*	オーバーライド情報を返すセグメント ID のカンマ区切りリスト。セグメントは、データ内で指定されている形式で指定します。例: *"7e3396fc:6e5251"*。
いいえ	`--velocityunit` *velocityunit*	レスポンスに表示される速度単位を指定します (mph - マイル/時、kph - キロメートル/時、mtps - メートル/秒、mtpm - メートル/分)。デフォルトは mph です。

例

この例では、サーバー上に設定されている US_NE データベース リソースから、セグメントのオーバーライドを返します。

```
erm getsegmentupdates --datasource US_NE --segmentids
"7e3396fc:6e5251" --velocityunit kph
```

erm createroadtypeupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm createroadtypeupdate` コマンドは、指定された道路タイプのルーティング データをオーバーライドします。このコマンドにより、特定の道路タイプのルートの速度を設定または変更できます。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： 永続更新タイプは特定のデータ リソースに対してのみ有効で、データ更新後には有効でなくなる場合があります。

使用方法

```
erm
createroadtypeupdate --datasource database --roadtype "roadtype" --velocity velocity --velocityunit velocityunit --velocityadjustment velocityadjustment --velocitypercentage velocitypercentage --roadtype roadtype
```

注： パラメータのリストを表示するには、“`help erm createroadtypeupdate`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	データをオーバーライドするデータベースリソースの名前を指定します。既存のルーティングデータベースリソースの一覧を表示するには、`ermdb list` コマンドを使用します。
はい	`--roadtype` *"road_type"*	オーバーロードする道路タイプ • 進入路 • 裏道路 • 接続道路 • フェリー • 小道 • 侵入制限道路 (密集都市部) • 侵入制限道路 (農村部) • 侵入制限道路 (郊外) • 侵入制限道路 (都市部) • 地方道路 (密集都市部) • 地方道路 (農村部) • 地方道路 (郊外) • 地方道路 (都市部) • 主要地方道路 (密集都市部) • 主要地方道路 (農村部) • 主要地方道路 (郊外) • 主要地方道路 (都市部) • 幹線道路 (密集都市部) • 幹線道路 (農村部) • 幹線道路 (郊外) • 幹線道路 (都市部) • 補助地方道路 (密集都市部) • 補助地方道路 (農村部) • 補助地方道路 (郊外) • 補助地方道路 (都市部) • 一般道路 (密集都市部) • 一般道路 (農村部) • 一般道路 (農村部) • 一般道路 (都市部) • 主要高速道路 (密集都市部)

必須	引数	説明
		• 主要高速道路 (農村部) • 主要高速道路 (郊外) • 主要高速道路 (都市部) • 出入路 (密集都市部) • 出入路 (侵入制限道路) • 出入路 (幹線道路) • 出入路 (主要高速道路) • 出入路 (農村部) • 出入路 (一般高速道路) • 出入路 (都市部) • 出入路 (郊外) • 一般高速道路 (密集都市部) • 一般高速道路 (農村部) • 一般高速道路 (郊外) • 一般高速道路 (都市部)
いいえ	`--velocity` *velocity_value*	速度更新を定義します。新しい速度を指定することにより、道路タイプの新しい速度を指定します。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocityunit` *velocity_unit*	`velocity` または `velocityadjustment` オーバーライドの速度単位を定義します。デフォルトは mph (マイル/時) です。速度更新の場合、速度単位には kph (キロメートル/時)、mps (メートル/秒)、mph (マイル/時) のいずれかの値を指定できます。
いいえ	`--velocityadjustment` *velocity_adjustment_value*	速度更新を定義します。速度 (単位と値) の変更を指定することにより、道路タイプの速度の変化を定義します。速度値は増加 (正の値) または減少 (負の値) させることができます。`velocityunit` パラメータを指定しない限り、デフォルトの単位は mph (マイル/時) です。
いいえ	`--velocitypercentage` *velocity_percentage_value*	速度更新を定義します。速度を増加 (正の値) または減少 (負の値) させる割合を指定することにより、道路タイプの速度の増加を定義します。

例

この例では、サーバー上に設定されている US_NE データベースリソースから、指定された道路タイプの速度を 25 mph にオーバーライドします。

```
erm createroadtypeupdate --datasource US_NE --roadtype "normal
road suburban" --velocity 25 --velocityunit kph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、指定された道路タイプの速度を 50 kph だけ増加します。

```
erm createroadtypeupdate --datasource US_NE --roadtype "normal
road suburban" --velocityadjustment 50 --velocityunit mph
```

この例では、サーバー上に設定されている US_NE データベースリソースから、指定された道路タイプの速度を 65% だけ減少させることによってオーバーライドします。

```
erm createroadtypeupdate --datasource US_NE --roadtype "normal
road suburban" --velocitypercentage -65
```

erm resetroadtypeupdate

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm resetroadtypeupdate` コマンドは、データの元の状態に対するオーバーライドがあれば、それを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`erm resetroadtypeupdate--datasource`***db_resource*** `--roadtype`***"road_type"***

注： パラメータのリストを表示するには、“`help erm resetroadtypeupdate`” と入力します。

必須	引数	説明
はい	`--datasource` ***db_resource***	オーバーライドを持つデータベースリソースの名前を指定します。既存のルーティングデータベースリソースの一覧を表示するには、`ermdb list` コマンドを使用します。
はい	`--roadtype` ***"road_type"***	既存のオーバーライドを持つ道路タイプを指定します。道路タイプの一覧については、「**erm**

必須	引数	説明
		createroadtypeupdate（503ページ）」を参照してください。

例

この例では、"normal road suburban" (一般道路 (郊外)) の道路タイプ オーバーライドを、サーバー上に設定されている US_NE データベース リソースからリセットします。

```
erm resetroadtypeupdate --datasource US_NE --roadtype "normal road suburban"
```

erm getroadtypeupdates

注： 管理ユーティリティをインストールして実行する手順については、「**管理ユーティリティを使用する前に**（288ページ）」を参照してください。

`erm getroadtypeupdates` コマンドは、指定された道路タイプのルーティング データにおけるオーバーライドのリストを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

注： `roadtypes` はオプションのパラメータです。道路タイプを指定しない場合は、使用可能なすべての道路タイプに対するオーバーライドが返されます。

使用方法

`erm getroadtypeupdates --datasource` ***db_resource*** `--roadtypes` ***"road_types"*** `--velocityunit` ***velocityunit***

注： パラメータのリストを表示するには、“`help erm getroadtypeupdates`” と入力します。

必須	引数	説明
はい	`--datasource` ***db_resource***	オーバーライドを持つデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--roadtypes` ***"road_types"***	オーバーライド情報を返す道路タイプのカンマ区切りリスト。道路タイプの一覧については、「**erm createroadtypeupdate**（503ページ）」を参照してください。

必須	引数	説明
いいえ	`--velocityunit` *velocityunit*	レスポンスに表示される速度単位を指定します (mph - マイル/時、kph - キロメートル/時、mtps - メートル/秒、mtpm - メートル/分)。デフォルトは mph です。

例

この例では、"normal road urban" (一般道路 (都市部)) の道路タイプのオーバーライドを、サーバー上に設定されている US_NE データベース リソースから返します。

```
erm getroadtypeupdates --datasource US_NE --roadtypes "normal road urban" --velocityunit kph
```

erm getallupdates

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm getallupdates` コマンドは、指定されたルーティング データベース リソースに対するすべてのオーバーライドのリストを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`erm getallupdates--datasource`*db_resource*`"segment_ids"--velocityunit`*velocityunit*

注： パラメータのリストを表示するには、“`help erm getallupdates`” と入力します。

必須	引数	説明
はい	`--datasource` *db_resource*	オーバーライドを持つデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--velocityunit` *velocityunit*	レスポンスに表示される速度単位を指定します (mph - マイル/時、kph - キロメートル/時、mtps - メートル/秒、mtpm - メートル/分)。デフォルトは mph です。

例

この例では、サーバー上に設定されている US_NE データベース リソースからのすべてのオーバーライドを返します。

```
erm getallupdates --datasource US_NE --velocityunit kph
```

erm resetallupdates

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`erm resetallupdates` コマンドは、データの元の状態に対するすべてのオーバーライドを返します。このコマンドを使うには、Spatial モジュールがインストールされている必要があります。

使用方法

`erm resetallupdates--datasource`***db_resource***

注： パラメータのリストを表示するには、“`help erm resetallupdates`” と入力します。

必須	引数	説明
はい	`--datasource` ***db_resource***	オーバーライドを持つデータベース リソースの名前を指定します。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。

例

この例では、サーバー上に設定されている US_NE データベース リソースからのすべてのオーバーライドをリセットします。

```
erm resetallupdates --datasource US_NE
```

Universal Addressing モジュールのデータベース

uamdb create

`uamdb create` コマンドは、新しい Universal Addressing モジュール データベースを作成します。

使用方法

`uamdb create --t` *Type* `--n` *Name* `--c` *CacheSize* `--i` *Country* `--pl` *PreloadingType* `--dt` *DatabaseType* `--b` *BasePath* `--d` *DPVPath* `--l` *LACSPath* `--s` *SuiteLinkPath* `--r` *RDIPath* `--e` *EWSPath* `--p` *Poolsize* `--mm` *minimumMemorySize* `--mx` *maximumMemorySize*

注： パラメータのリストを表示するには、"`help uamdb create`" と入力します。

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース **Loqate** Loqate データベース **Global** Validate Address Global データベース **Amas** オーストラリアのデータベース
はい	`--n` *Name*	データベースの名前を指定します。
いいえ	`--c` *CacheSize*	Validate Address Global データベースのキャッシュ サイズを指定します。 *CacheSize* は次のいずれかです。 **None** キャッシュなし **Small** 小さいキャッシュ **Large** 大きいキャッシュ (デフォルト)
いいえ	`--i` *Country*	使用する Validate Address Global データベース内の各国の 3 桁の ISO コードを指定します。 *Country* は、"All" (デフォルト)、またはカンマで区切られたコードのリストです。
いいえ	`--pl` *PreloadingType*	プリロードする Validate Address Global データベースの量を指定します。 *PreloadingType* は次のいずれかです。 **None** データをプリロードしません (デフォルト)。 **Partial** メタデータおよびインデックス構造をメモリにロードします。参照データ自体はハード ドライブに残ります。パフォーマンスがやや向上します。また、目的のデータベースをすべてロードするだけのメモリが使用できない場合の代替策となります。

必須	引数	説明
		Full 参照データベース全体をメモリに移します。米国や英国など、大規模なデータベースを持つ国では、大量のメモリが必要になることがありますが、処理速度は大幅に向上します。
いいえ	`--dt` *DatabaseType*	Validate Address Global データベースの処理モードを指定します。*DatabaseType* は次のいずれかです。 **Batch_Interactive** バッチ処理またはインタラクティブな環境で使用します。処理速度を重視して最適化されているため、自動で修正できないあいまいなデータが見つかった場合は、住所修正の試行を停止します (デフォルト)。 **認定** オーストラリアの郵便物を対象に、Postal Address File に対する郵便物の正規化と検証を行うためのバッチ処理環境で使用されます。 **高速実行** 住所フィールドに不完全なデータを入力し、Validate Address Global に候補を生成してもらう場合に使用します。
はい	`--b` *BasePath*	ベース サブスクリプション データベース パスを指定します。 **注 : USA**、**CAN**、**INTL**、**Loqate**、および **Validate Address Global** の場合は、データ パスの代わりにデータベース ヴィンテージを指定します。**例:**`NOV2017`
いいえ	`--d` *DPVPath*	DPV データベース ヴィンテージを指定します。
いいえ	`--l` *LACSPath*	LACS データベース ヴィンテージを指定します。
いいえ	`--s` *SuiteLinkPath*	SuiteLink データベース ヴィンテージを指定します。
いいえ	`--r` *RDIPath*	RDI データベース ヴィンテージを指定します。
いいえ	`--e` *EWSPath*	EWS データベース ヴィンテージを指定します。
いいえ	`--p` *Poolsize*	このデータベースで処理する同時要求の最大数を指定します。デフォルトは 4 です。
いいえ	`--mn` or `--minMem` *minimum_memory_size*	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。

必須	引数	説明
いいえ	`--mx` or `--maxMem` *maximum_memory_size*	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

注： データベース ヴィンテージは、*uamdb listdatasets* コマンドを使用して取得できます。詳細については、**uamdb listdatasets**（517ページ）を参照してください。

例

UAM US、*CAN*、*INTL*、*Loqate*、または *Validate Address Global* のデータベースを作成するには、入力を次の形式で指定します。

```
uamdb create --t USA --n UAM_US --b FEB2018 --d AUG2018 --r
SEP2018 --mn 1200 --mx 65536
```

uamdb modify

`uamdb modify` コマンドは、既存の Universal Addressing モジュール データベースを更新します。

使用方法

`uamdb modify--t`*Type*`--n` *Name*`--b`*BasePath*`--d`*DPVPath*`--l` *LACSPath*`--s` *SuiteLinkPath*`--r` *RDIPath*`--e` *EWSPath*`--p` *Poolsize*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース **Global** Validate Address Global データベース
はい	`--n` *Name*	データベースの名前を指定します。
はい	`--b` *BasePath*	ベース サブスクリプション データベース パスを指定します。 **注：USA**、**CAN**、**INTL**、**Loqate**、および **Validate Address Global** の場合は、データ パスの代わりにデータベース ヴィンテージを指定します。**例:**`NOV2017`
いいえ	`--d` *DPVPath*	DPV データベース ヴィンテージを指定します。

必須	引数	説明
いいえ	`--l` *LACSPath*	LACS データベース ヴィンテージを指定します。
いいえ	`--s` *SuiteLinkPath*	SuiteLink データベース ヴィンテージを指定します。
いいえ	`--r` *RDIPath*	RDI データベース ヴィンテージを指定します。
いいえ	`--e` *EWSPath*	EWS データベース ヴィンテージを指定します。
いいえ	`--p` *Poolsize*	このデータベースで処理する同時要求の最大数を指定します。デフォルトは 4 です。

注: データベース ヴィンテージは、*uamdb listdatasets* コマンドを使用して取得できます。詳細については、**uamdb listdatasets**（517ページ）を参照してください。

例

UAM US、*CAN*、*INTL*、*Loqate*、または *Validate Address Global* のデータベースを作成するには、入力を次の形式で指定します。

```
uamdb modify --n UAM_US --t USA --b SEP2018 --d AUG2018 --r
OCT2018
```

uamdb delete

`uamdb delete` コマンドは、Universal Addressing モジュール データベースを削除します。

使用方法

`uamdb delete --t` *Type* `--n` *Name*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース **Loqate** Loqate データベース **Global** Validate Address Global データベース **Amas** オーストラリアのデータベース
はい	`--n` *Name*	データベースの名前を指定します。

例

この例は、"UAM_CAN" という名前のカナダのデータベースを削除します。

```
uamdb delete --t CAN --n UAM_CAN
```

uamdb memory set

注： 管理ユーティリティをインストールして実行する手順については、「管理ユーティリティを使用する前に（288ページ）」を参照してください。

`uamdb memory set` コマンドは、Universal Addressing モジュール データベースのメモリ サイズを定義します。このコマンドを使うには、Universal Addressing モジュールがインストールされている必要があります。最小メモリと最大メモリを定義するフィールドは、空にすることができます。値が空の場合、値がないことが明示的に定義されたかのように、コンポーネントの起動時にコマンドラインに値が指定されません。値を指定しない場合、または値が 0 の場合は、そのプロパティはコマンド ライン インターフェイスに渡されません。

使用方法

`uamdb memory set --name` ***database_name*** `--mn` ***minimum_memory_size*** `--mx` ***maximum_memory_size***

注： パラメータのリストを表示するには、"`help uamdb memory set`" と入力します。

必須	引数	説明
はい	`--name` or `--n` ***database_name***	追加するデータベース リソースの名前を指定します。名前は、サーバー上で一意でなければなりません。既存のルーティング データベース リソースの一覧を表示するには、`ermdb list` コマンドを使用します。
いいえ	`--mn` or `--minMem` ***minimum_memory_size***	このデータベースに割りてられるメモリの最小サイズを定義します。この値は、 `--mx` で設定された値と同じか、それより小さい必要があります。
いいえ	`--mx` or `--maxMem` ***maximum_memory_size***	このデータベースに割りてられるメモリの最大サイズを定義します。この値は 0 より大きく、なおかつ 65536 MB より小さい必要があります。

例

この例では、Universal Addressing モジュール U.S. データベースのメモリ サイズを設定します。

```
uamdb memory set --name UAM_US --mn 1200 --mx 65536
```

uamdb import

`uamdb import` コマンドは、Universal Addressing モジュール データベースをエクスポートします。

使用方法

`uamdb import--t`*Type*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース **Loqate** Loqate データベース **Global** Validate Address Global データベース **Amas** オーストラリアのデータベース

例

この例は、米国のデータベースをインポートします。

```
uamdb import --t USA
```

uamdb export

`uamdb export` コマンドは、Universal Addressing モジュール データベースをエクスポートします。

使用方法

`uamdb export--t`*Type*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース

必須	引数	説明	
		CAN	カナダのデータベース
		INTL	国際データベース
		Loqate	Loqate データベース
		Global	Validate Address Global データベース
		Amas	オーストラリアのデータベース

例

この例は、国際データベースをエクスポートします。

```
uamdb export --t INTL
```

uamdb get resource info

`uamdb get resource info` コマンドは、データベースに関する情報を返します。

使用方法

`uamdb get resource info --t` *Type* `--n` *Name*

必須	引数	説明	
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。	
		USA	米国のデータベース
		CAN	カナダのデータベース
		INTL	国際データベース
		Loqate	Loqate データベース
		Global	Validate Address Global データベース
		Amas	オーストラリアのデータベース
はい	`--n` *Name*	データベースの名前を指定します。	

例

この例は、"UAM_US" という米国データベースの情報を取得します。

```
uamdb get resource info --t USA --n UAM_US
```

次のような情報が返されます。

```
DATABASE NAME = UAM_US
POOL SIZE = 4
LACS_DB_PATH = Z:\UAM\US_AUG12
SUITELINK_DB_PATH = Z:\UAM\US_AUG12
BASE_DB_PATH = Z:\UAM\US_AUG12
DPV_DB_PATH = E:\UAM_US_MAY_14_DB
RDI_DB_PATH = E:\UAM_US_MAY_14_DB
EWS_DB_PATH = Z:\UAM\US_AUG12
```

uamdb list

`uamdb list` コマンドは、該当するタイプのすべての Universal Addressing モジュール データベースをテーブル形式で返します。

使用方法

`uamdb list--t`*Type*

必須	引数	説明	
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。	
		USA	米国のデータベース
		CAN	カナダのデータベース
		INTL	国際データベース
		Loqate	Loqate データベース
		Global	Validate Address Global データベース
		Amas	オーストラリアのデータベース

例

この例は、すべてのカナダ データベースを一覧表示します。

```
uamdb list --t CAN
```

uamdb listdatasets

コマンドは、`uamdb listdatasets` プラットフォーム上に登録されている Universal Addressing モジュール データベースを表示します。**コンポーネント**、**名前**、**ヴィンテージ**などの詳細がここに表示されます。

使用方法

`uamdb listdatasets--t`*Type*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース

例

この例では、プラットフォーム上に登録されている米国データベースを一覧表示します。

```
uamdb listdatasets --t USA
```

uamdbglobalmultipath create_sample_file

`uamdbglobalmultipath create_sample_file` コマンドは、複数のパス要素を持つデータベース リソースを設定し、プレースホルダーやデータ パスが変更可能なサンプル JSON ファイル (`UamDbGlobalMultiPath.txt`) を作成します。このコマンドの後には、その他のデータベース設定を行うコマンド `uamdb_import` が続く必要があります。

注: テキスト ファイル内のトークン値を、絶対値とデータ パスで置き換える必要があります。

使用方法

`uamdbglobalmultipath create_sample_file --o` *OutputDirectory*`--n` *NumberOfPathElements*

必須	引数	説明
いいえ	`--o` *OutputDirectory*	ファイルを出力するディレクトリを指定します。現在のディレクトリがデフォルトの場所になります。
はい	`--n` *NumberOfPathElements*	パス内の要素数を指定します。

例

この例では、プロパティが *JSON* のキーと値の形式で記述された、*"UamDbGlobalMultiPath.txt"* という名前のサンプル *JSON* ファイルを作成します。このデータベース リソースには、3 つのパス要素があります。

```
uamdbglobalmultipath create_sample_file --n 3
```

uamdb poolsize set

`uamdb poolsize set` コマンドは、データベースのデフォルト プールサイズを設定します。

使用方法

`uamdb poolsize set --t` *Type* `--n` *Name* `--s` *Size*

必須	引数	説明
はい	`--t` *Type*	データベースのタイプを指定します。ここで *Type* は次のいずれかです。 **USA** 米国のデータベース **CAN** カナダのデータベース **INTL** 国際データベース
はい	`--n` *Name*	データベースの名前を指定します。
はい	`--s` *Size*	デフォルト プールサイズを指定します。

例

この例は、"UAM_CAN" というカナダ データベースのプールサイズを 4 に設定します。

```
uamdb poolsize set --t CAN --n UAM_CAN --s 4
```

サービスのプール サイズ

service poolsize default get

`service poolsize default get` コマンドは、すべてのサービスのプール サイズに関する、詳細なデフォルト情報を返します。プール サイズとは、リソース プールに対して許容される同時要求の最大数を表す整数値です。

使用方法

```
service poolsize default get
```

service poolsize default set

`service poolsize default set` コマンドは、すべての設定済みデータフロー サービスのデフォルト プール サイズを定義します。プール サイズとは、リソース プールに対して許容される同時要求の最大数を表す整数値です。

使用方法

```
service poolsize default set --p poolSize
```

必須	引数	説明
はい	`--p` *poolSize*	サービスのデフォルト プール サイズ (整数値で指定) を設定します。デフォルト値はありません。この値の設定についてご不明な点があれば、システム管理者または Pitney Bowes サポート担当者までお問い合わせください。

例

この例は、設定済みのドイツ向け Global Addressing Validation サービス (GAV_DEU) のプール サイズを 5 に設定します。

```
service poolsize default set --p 5
```

service poolsize get

`service poolsize get` コマンドは、データフロー サービス リソースに関する詳細なプール サイズ情報を返します。プール サイズとは、リソース プールに対して許容される同時要求の最大数を表す整数値です。

使用方法

`service poolsize get --s` ***serviceName***

必須	引数	説明
はい	`--s` ***serviceName***	リスト化するサービス詳細情報を指定します。

例

この例は、ユーザによって定義された住所解析サービス "ParsingAddresses" に関する情報を返します。

```
service poolsize get --s ParsingAddresses
```

service poolsize set

`service poolsize set` コマンドは、データフロー サービス リソースに対するプール サイズの上限を定めます。プール サイズとは、リソース プールに対して許容される同時要求の最大数を表す整数値です。

使用方法

`service poolsize set --s` ***serviceName*** **--p** ***poolSize***

必須	引数	説明
はい	`--s` ***serviceName***	この定義が適用されるサービス名を指定します。
はい	**--p** ***poolSize***	サービスのプール サイズを設定します (整数値で指定)。デフォルト値はありません。この値の設定についてご不明な点があれば、システム管理者または Pitney Bowes サポート担当者までお問い合わせください。

例

この例は、ユーザによって定義された住所解析サービス "ParsingAddresses" のデフォルト プール サイズを定義します。

```
service poolsize set --s ParsingAddresses --p 4
```

システム

close

`close` コマンドは、Spectrum™ Technology Platform サーバーとのセッションを閉じます。管理ユーティリティを終了しないでサーバーとのセッションを閉じる場合は、このコマンドを使用します。`exit` コマンドを使用すると、セッションを閉じて管理ユーティリティを終了できます。

使用方法

```
close
```

connect

`connect` コマンドは、指定した Spectrum™ Technology Platform サーバーとのセッションを開きます。他のコマンドを実行する前に、`connect` コマンドを実行する必要があります。

使用方法

`connect--h`*HostName*`--u` *UserName*`--p`*Password*`--s`*TrueOrFalse*

必須	引数	説明
はい	`--h` *HostName*	接続先のホスト名とポート。コロンで区切って指定します。例えば、MyServer のポート 8080 に接続するには、`--h MyServer:8080` と指定します。
はい	`--u` *UserName*	サーバーとの認証に使用するユーザ名。
はい	`--p` *Password*	そのユーザのパスワード。
いいえ	`--s` *TrueOrFalse*	HTTPS を使用してセキュアな接続を作成するかどうかを指定します。サーバーが HTTPS 通信をサポートするように構成されて

必須	引数	説明
		いる場合は、セキュアな通信を介して接続しなければなりません。*TrueOrFalse* は、次のいずれかです。 **true** HTTPS を使用します。 **false** HTTPS を使用しません。これがデフォルト設定です。

例

この例では、ユーザ名 admin とパスワード myPassword1 を使用して、サーバー MyServer のポート 8080 への接続を開きます。

```
connect --h MyServer:8080 --u admin --p myPassword1
```

date

`date` コマンドは、管理ユーティリティを実行しているコンピュータの現在の日付と時刻を表示します。

使用方法

```
date
```

exit

`exit` コマンドは、セッションを閉じ、管理ユーティリティを終了します。セッションを閉じるだけで、管理ユーティリティは終了しない場合は、`close` コマンドを使用します。

使用方法

```
exit
```

help

`help` コマンドは、管理ユーティリティで使用できるコマンドのリストを表示します。各コマンドで使用するパラメータについても、`help` コマンドで情報を表示できます。

使用方法

`help`*Command*

必須	引数	説明
いいえ	*Command*	コマンド名をパラメータとして `help` コマンドに指定すると、そのコマンドに関する詳しい情報が表示されます。コマンド名を指定しない場合は、すべてのコマンドのリストが表示されます。

例

次のコマンドを実行すると、管理ユーティリティで使用可能なコマンドのリストが表示されます。

```
help
```

次のコマンドは、`set serviceoption` コマンドの詳しい情報を表示します。

```
help set serviceoption
```

license expirationinfo export

`license expirationinfo export` コマンドは、有効期限が近づいているライセンスのリストをエクスポートします。これらのライセンスは、Management Console の **[通知]** タブで指定した期間内に期限切れとなるものです。

使用方法

`license expirationinfo export--o`*Directory*

必須	引数	説明
いいえ	`--o` *Directory*	ライセンス有効期限情報のエクスポート先とするディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。このオプションを省略すると、有効期限情報は管理ツールを実行しているフォルダ内のファイルに格納されます。

例

この例では、ライセンス有効期限情報を、管理ツールがあるフォルダの下の LicenseExpiration サブフォルダにエクスポートします。

```
license expirationinfo export --o LicenseExpiration
```

license expirationinfo list

`license expirationinfo list` コマンドは、有効期限が近づいているライセンスのリストを返します。表示されるライセンスは、Management Console の **[通知]** タブで指定した期間内に期限切れとなるものです。

使用方法

```
license expirationinfo list
```

licenseinfo export

`licenseinfo export` コマンドは、ライセンス情報をファイルにエクスポートします。ライセンス ファイルは、ライセンスの問題をテクニカル サポートが解決する際に必要となります。

使用方法

`licenseinfo export--o`*Directory*

必須	引数	説明
いいえ	`--o` *Directory*	ライセンス ファイルのエクスポート先とするディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。このオプションを省略すると、ライセンス ファイルは管理ツールを実行しているフォルダ内のファイルに格納されます。

例

この例では、ライセンス情報を、管理ツールがあるフォルダの下の License サブフォルダにエクスポートします。

```
licenseinfo export --o License
```

licenseinfo list

`licenseinfo list` コマンドは、インストールされているライセンス、ライセンスの有効期限までの日数、残りのトランザクション数などのライセンス情報を表示します。

使用方法

```
licenseinfo list
```

server backup

`server backup` サーバーのバックアップを行うには、Spectrum™ Technology Platform コマンドを使用します。

Spectrum™ Technology Platform サーバーをバックアップするには、サーバーの構成データベースのバックアップ コピーを作成する必要があります。構成データベースには、セキュリティ設定、データフロー、サービス オプション、データ リソース定義、スナップショットのほか、さまざまな構成情報が含まれています。サーバーのシステム障害やその他の災害でサーバーが稼働できなくなった場合、構成データベースのバックアップを使用してサーバー構成を別の Spectrum™ Technology Platform サーバーに復元できます。

重要： Spectrum™ Technology Platform サーバーにアクティビティがある時間帯には、バックアップを実行しないでください。バックアップの実行中は、サービスの呼び出しがタイムアウトになったり、ジョブが正常に実行できなかったりする場合があります。

使用方法

`server backup--o`*`Directory`*

必須	引数	説明
いいえ	`--o` *Directory*	サーバーのデータベースのバックアップ コピーの保存先ディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。このオプションを省略すると、管理ツールを実行しているフォルダがバックアップの置き場所になります。

例

この例は、管理ツールが存在するフォルダの下にある LatestServerBackup サブフォルダにバックアップを保存します。

```
server backup --o LatestServerBackup
```

script

`script` コマンドは、一連のコマンドが含まれるスクリプトの実行を管理ツールに指示します。スクリプトを使用して、管理タスクを自動化できます。

使用方法

`script--file`*ScriptFile*`--linenumbers` *TrueOrFalse*

必須	引数	説明
はい	`--file` *ScriptFile*	スクリプト ファイルへのパスを指定します。
いいえ	`--linenumbers` *TrueOrFalse*	スクリプトの実行中に行番号を表示するかどうかを指定します。ここで、*TrueOrFalse* は次のいずれかです。 **true** スクリプトの実行中に行番号を表示します。 **false** スクリプトの実行中に行番号を表示しません。この値がデフォルトです。

例

この例では、`myscript.cli` という名前でフォルダ `scripts` の下に格納されているスクリプトを実行します。このフォルダは、管理ユーティリティがあるフォルダのサブフォルダです。

```
script --file scripts/myscript.cli
```

system loglevel get

`system loglevel get` コマンドはサービスのデフォルトのログ レベルを返します。ログ レベルは次のとおりです。

Off すべてのイベント ログを無効にします。

Fatal 最小限のログです。致命的なエラーのみがログに記録されます。致命的なエラーとは、システムを使用不可能にするエラーのことです。

Error エラーと致命的なエラーがログに記録されます。エラーとは、システムの一部が使用不可能になる単発的な問題を指します。例えば、1つのサービスが機能しなくなる問題によってエラーが生成されます。

Warn イベント警告、エラー、致命的なエラーがログに記録されます。警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。

Info	抽象度の高いシステム情報がログに記録されます。これは、実稼働に適した最も詳細なログ レベルです。通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。
Debug	システムの問題のデバッグ時に適した、非常に詳細なログ レベルです。
Trace	プログラムの実行 (メソッドの開始と終了) をトレースする、最も詳細なログ レベルです。デバッグに使用する詳細なプログラム フロー情報を提供します。

使用方法

```
system loglevel get
```

system loglevel set

`system loglevel set` コマンドはシステム上のサービスのデフォルトのログ レベルを設定します。

使用方法

```
system loglevel set--l Level
```

必須	引数	説明
はい	`--l` *Level*	システム上のサービスのデフォルトのログ レベルを指定します。ここで、*レベル*は次のいずれかです。 **Off** すべてのイベント ログを無効にします。 **Fatal** 最小限のログです。致命的なエラーのみがログに記録されます。致命的なエラーとは、システムを使用不可能にするエラーのことです。 **Error** エラーと致命的なエラーがログに記録されます。エラーとは、システムの一部が使用不可能になる単発的な問題を指します。例えば、1 つのサービスが機能しなくなる問題によってエラーが生成されます。 **Warn** イベント警告、エラー、致命的なエラーがログに記録されます。警告とは、システムの動作を停止させることのない問題を指します。例えば、パラメータに無効な値が設定されているサービスをロードすると、警告が発行され、デフォルトのパラメータが使用されます。サービスの使用時に、結果は返ってきたが問題が存在するという場合に、警告がログに記録されます。 **Info** 抽象度の高いシステム情報がログに記録されます。これは、実稼働に適した最も詳細なログ レベルです。通常、情報イベントは、起動や初期化の際に発生し、バージョン情報やロードされたサービスなどの情報を提供します。

必須	引数	説明
	Debug	システムの問題のデバッグ時に適した、非常に詳細なログ レベルです。
	Trace	プログラムの実行(メソッドの開始と終了)をトレースする、最も詳細なログ レベルです。デバッグに使用する詳細なプログラム フロー情報を提供します。

注： 最も詳細なログ レベルを選択すると、システムのパフォーマンスに影響が生じる恐れがあります。したがって、必要なログ要件を満たす最小レベルの設定を選択する必要があります。

例

この例では、デフォルトのログ レベルを Warn に設定します。

```
systemloglevel set --l warn
```

system properties

`system properties` コマンドは、管理ユーティリティを実行するシェルに関する情報 (Java プロパティ、OS のバージョンなど) を表示します。Spectrum™ Technology Platform サーバーに関する情報は表示しません。

使用方法

```
system properties
```

versioninfo export

`versioninfo export` コマンドは、システム、コンポーネント、およびサービスのバージョン情報をファイルにエクスポートします。

使用方法

```
versioninfo export--o Directory
```

必須	引数	説明
いいえ	`--o` *Directory*	バージョン情報テキスト ファイルのエクスポート先ディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレク

必須	引数	説明
		トリへの相対パスを使用します。このオプションを省略すると、管理ツールを実行しているフォルダがバージョン情報ファイルの置き場所になります。

例

この例は、管理ツールが存在するフォルダの下にある VersionInformation サブフォルダにバージョン情報をエクスポートします。

```
versioninfo export --o VersionInformation
```

versioninfo list

`versioninfo list` コマンドは、システムにインストールされている Spectrum™ Technology Platform のバージョンや基盤となるコンポーネントに関する情報、およびその他のシステム情報を表示します。

使用方法

```
versioninfo list
```

テーブル

table delete

`table delete` コマンドは、テーブルをシステムから削除します。詳細については、『*データ品質ガイド*』の「検索テーブル」セクションを参照してください。

使用方法

`table delete` *TableName* `--t` *TableType*

必須	引数	説明
はい	`--n` *TableName*	削除するテーブルを指定します。

必須	引数	説明
はい	`--t` *TableType*	削除するテーブルのタイプ (AdvancedTransformer、OpenParser、または TableLookup) を指定します。

例

この例は、My Table という名前の Table Lookup テーブルを削除します。

```
table delete My Table --t TableLookup
```

table export

`table export` コマンドは、Enterprise Designer のテーブル管理機能を使用して作成されたカスタム テーブルをエクスポートします。その後、このテーブルは別のサーバーにインポートできます。詳細については、『*データ品質ガイド*』の「検索テーブル」セクションを参照してください。

使用方法

`table export` *TableName* `--t` *TableType* `--o` *OutputDirectory* `--f` *ExportedFileName*

必須	引数	説明
はい	`--n` *TableName*	エクスポートするテーブルの名前を指定します。 **ヒント:** テーブルの名前が正確にわからない場合は、`table list` コマンドを使用してテーブル名のリストを取得できます。
はい	`--t` *TableType*	エクスポートするテーブルのタイプ (AdvancedTransformer、OpenParser、または TableLookup) を指定します。
いいえ	`--o` *OutputDirectory*	テーブルのエクスポート先ディレクトリを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。この引数を省略すると、テーブルは管理ユーティリティと同じディレクトリにエクスポートされます。
いいえ	`--f` *ExportedFileName*	エクスポートするファイルの名前を指定します。エクスポートするテーブル名に '/' 文字が含まれている場合は、このオプションを使用して名前を変更することでエクスポートできます。

例

この例は、"My Table" という名前の Open Parser テーブルを、管理ユーティリティがインストールされている場所にエクスポートします。

```
table export --n My Table --t OpenParser
```

例

この例は、*"AC/E"* という名前の Open Parser テーブルを、*--f* オプションを使用して *"AC_E"* という名前に変更し、管理ユーティリティがインストールされている場所にエクスポートします。

```
table export --n AC/E --t OpenParser --f AC_E
```

table import

`table import` コマンドは、カスタム テーブルをサーバーにインポートします。カスタム テーブルは、Enterprise Designer のテーブル管理機能を使用して作成されます。詳細については、『*データ品質ガイド*』の「検索テーブル」セクションを参照してください。

使用方法

`table import` *CustomTable* `--u` *TrueOrFalse*

必須	引数	説明
はい	`--f` *CustomTable*	インポートするカスタム テーブルを指定します。ここでは、管理ユーティリティを実行しているディレクトリへの相対パスを使用します。
いいえ	`--u` *TrueOrFalse*	同じ名前のテーブルが既にサーバー上に存在する場合、既存のテーブルを上書きするかどうかを指定します。ここで、*TrueOfFalse* は次のいずれかです。 **true** インポートするテーブルと同じ名前のテーブルがサーバー上に存在する場合、サーバー上のテーブルは上書きされます。これがデフォルト設定です。 **false** インポートするテーブルと同じ名前のテーブルがサーバー上に存在する場合、テーブルのインポートは行われません。

例

この例では、`MyTable.db` というテーブルをインポートします。このテーブルは、管理ユーティリティを実行している場所の `exported` というサブフォルダにあります。--u コマンドが指定されていないため、サーバー上に同じ名前のテーブルがあれば上書きされます。

```
table import exported\MyTable.db
```

table list

`table list` コマンドは、サーバー上のすべてのテーブルを一覧表示します。テーブルごとに、テーブル名と、データフローがエクスポーズされているかどうかの情報が表示されます。

使用方法

`table list --t` *TableType*

必須	引数	説明
はい	`--t` *TableType*	一覧表示するテーブルのタイプ (AdvancedTransformer、OpenParser、または TableLookup) を指定します。

例

この例は、すべての Advanced Transformer テーブルを一覧表示します。

```
table list --t AdvancedTransformer
```

トークン

token list

`token list` コマンドは、Spectrum™ Technology Platform サーバーにあるアクティブなトークンのリストを返します。各トークンに関連して以下の情報が提供されます。

- ユーザ名
- トークンが作成された日付と時刻

- トークンが最後に使用された日付と時刻
- Spectrum™ Technology Platform サーバーへのアクセスにしようされたコンピュータの IP アドレス
- セッション ID
- トークン

これらのフィールドは、パイプ文字 (|) で区切られます。

使用方法

```
token list--u UserName
```

必須	引数	説明
いいえ	`--u` *UserName*	トークンを表示するユーザを指定します。この引数を指定しない場合、すべてのユーザのトークンが表示されます。

例

この例では、amy123 というユーザのトークンを表示します。

```
token list --u amy123
```

この例は、すべてのトークンを表示します。

```
token list
```

token refreshsecret

`token refreshsecret` コマンドは、秘密鍵を更新します。その結果、すべてのアクティブなトークンのレンダリングが無効になります。アクティブなトークンを使用するユーザは、再ログインして新しいトークンを取得する必要があります。

使用方法

```
token refreshsecret
```

token revoke

`token revoke` コマンドは、トークンを無効化します。ユーザは、再ログインして新しいトークンを取得する必要があります。

使用方法

```
token revoke--t Token
```

必須	引数	説明
はい	`--t` *Token*	取り消すトークンを指定します。アクティブなトークンのリストを表示するには、`token list` コマンドを使用します。

例

この例では、`--t` 引数に指定されたトークンが取り消されます。

```
token revoke --t
```

token userrevoke

`token userrevoke` コマンドは、すべてのユーザのトークンを無効化します。ユーザは、再ログインして新しいトークンを取得する必要があります。

使用方法

```
token userrevoke--uUserName
```

必須	引数	説明
はい	`--u` *UserName*	トークンを取り消すユーザを指定します。アクティブなユーザのリストを表示するには、`token list` コマンドを使用します。

例

この例では、amy123 というユーザのトークンを取り消します。

```
token userrevoke --u amy123
```

ユーザ アカウント

user create

`user create` コマンドは、新しいユーザを作成し、役割をそのユーザに割り当てます。

使用方法

`user create --u` *UserName* `--p` *Password* `--d` *Description* `--e` *EmailAddress* `--r` *Roles*

必須	引数	説明
はい	`--u` *UserName*	新しいユーザの名前を指定します。
はい	`--p` *Password*	新しいユーザのパスワードを変更します。
いいえ	`--d` *Description*	ユーザの説明を指定します。説明にスペースが含まれる場合は、説明を引用符で囲みます。
いいえ	`--e` *EmailAddress*	新しいユーザの電子メール アドレスを指定します。
いいえ	`--r` *Roles*	ユーザの役割を指定します。複数の役割を指定するには、カンマで区切ります。スペースは使用できません。役割の名前にスペースが含まれる場合は、名前を引用符で囲みます。

例

この例では、allan12 という名前で新しいユーザを作成し、myPassword1 というパスワード、 Allan P.Smith という説明、`allan@example.com` という電子メール アドレス、および USBanking と California Users という 2 つの役割をこのユーザに割り当てます。

```
user create --u allan12 --p myPassword1 --d "Allan P.Smith"
--e allan@example.com --r USBanking,"California Users"
```

user delete

`user delete` コマンドは、ユーザ アカウントをシステムから削除します。

ヒント: また、ユーザ アカウントは無効にすることもできます。ユーザ アカウントを無効にすると、アカウントを削除しなくても、システムにアクセスできなくなります。

使用方法

`user delete --u` *UserName*

必須	引数	説明
はい	`--u` *UserName*	削除するユーザの名前を指定します。

例

この例では、allan12 というユーザ アカウントを削除します。

```
user delete --u allan12
```

user description set

`user description set` コマンドは、アカウントの説明を変更します。

使用方法

`user description set --u` *UserName* `--d` *Description*

必須	引数	説明
はい	`--u` *UserName*	説明を変更するユーザ アカウントを指定します。
はい	`--d` *Description*	ユーザの説明を指定します。説明にスペースが含まれる場合は、説明を引用符で囲みます。

例

この例では、ユーザ allan12 の説明を "Allan P.。

```
user description set --u allan12 --d "Allan P.Smith" に変更します。
```

user email set

`user email set` コマンドは、ユーザに関連付けられた電子メール アドレスを変更します。

使用方法

```
user email set --u UserName --e  EmailAddress
```

必須	引数	説明
はい	`--u` *UserName*	電子メールを変更するユーザ アカウントを指定します。
はい	`--e` *EmailAddress*	ユーザに関連付けられた電子メール アドレスを指定します。

例

この例は、account allan12 というユーザ アカウントの電子メール アドレスを `allan@example.com` に設定します。

```
user email set --u allan12 --e allan@example.com
```

user enabled set

`user enabled set` コマンドは、ユーザ アカウントを有効または無効にします。

注： ユーザ アカウント "admin" は無効にできません。

使用方法

```
user enabled set--uUserName--e  TrueOrFalse
```

必須	引数	説明
はい	`--u` *UserName*	無効にするユーザの名前を指定します。
はい	`--e` *TrueOrFalse*	ユーザアカウントを有効にするか無効にするかを指定します。ここで *TrueOfFalse* は次のいずれかです。 **true** ユーザ アカウントを有効にします。 **false** ユーザ アカウントを無効にします。

例

この例では、allan12 というユーザ アカウントを無効にします。

```
user enabled set --u allan12 --e false
```

user list

`user list` コマンドは、ユーザのリストを返します。各ユーザーの役割、電子メールアドレス、説明、ユーザが有効か無効かを一覧にして返します。

使用方法

```
user list
```

user password set

`user password set` コマンドは、ユーザ アカウントのパスワードを変更します。

使用方法

`user password set--u`*UserName*`--p` *Password*

必須	引数	説明
はい	`--u` *UserName*	パスワードを変更するユーザアカウントを指定します。
はい	`--p` *Password*	ユーザ アカウントに割り当てるパスワードを指定します。

例

この例では、allan12 というユーザ アカウントのパスワードを mypassword1 に設定します。

```
user password set --u allan12 --p mypassword1
```

user role grant

`user role grant` コマンドは、1 つ以上の役割をユーザに割り当てます。

使用方法

`user role grant--u`*UserName*`--r` *Roles*

必須	引数	説明
はい	`--u` *UserName*	役割を割り当てるユーザの名前を指定します。

必須	引数	説明
はい	`--r` *Roles*	ユーザの役割を指定します。複数の役割を指定するには、カンマで区切ります。スペースは使用できません。役割の名前にスペースが含まれる場合は、名前を引用符で囲みます。

例

この例では、2 つの役割 USBanking と CaliforniaUsers をユーザ allan12 に割り当てます。

```
user role grant --u allan12 --r USBanking,CaliforniaUsers
```

user role list

`user role list` コマンドは、システムにある役割のリストを返します。

使用方法

```
user role list
```

user role revoke

`user role revoke` コマンドは、ユーザから役割を削除して、役割によって付与された権限を取り消します。

使用方法

`user role revoke--u`*UserName*`--r` *Roles*

必須	引数	説明
はい	`--u` *UserName*	役割を取り消すユーザの名前を指定します。
はい	`--r` *Roles*	取り消す役割を指定します。複数の役割を指定するには、カンマで区切ります。スペースは使用できません。役割の名前にスペースが含まれる場合は、名前を引用符で囲みます。

例

この例では、ユーザ allan12 の役割 USBanking を取り消します。

```
user role revoke --u allan12 --r USBanking
```

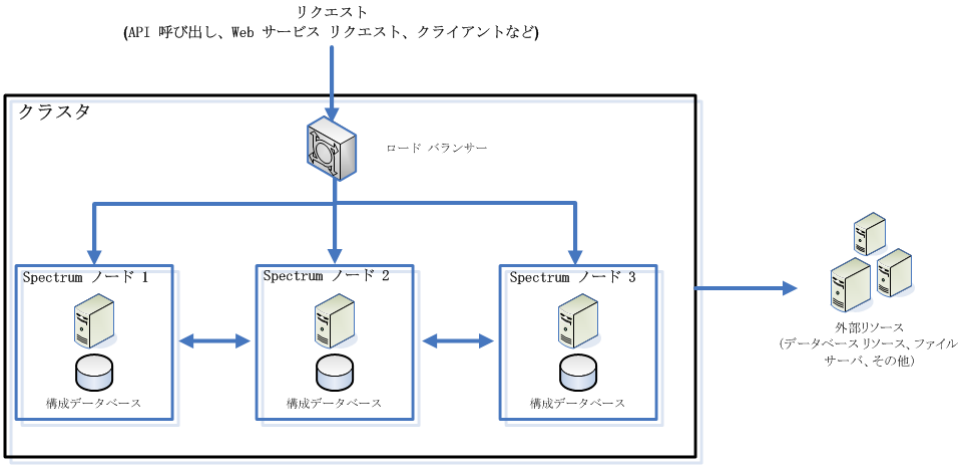
13 - クラスタ化

このセクションの構成

クラスタ アーキテクチャ

クラスタ環境では、サーバーの複数のインスタンスが処理を共有します。Spectrum™ Technology Platformとのすべての通信は、ロード バランサーを通して行われます。Spectrum™ Technology Platformサーバーの URL とポートを使用する代わりに、ロード バランサーの URL とポートを使用します。フェイルオーバー冗長性と大量の高性能処理が必要な場合は、この方法を検討してください。

次の図に、クラスタ アーキテクチャの構成を示します。



ロード バランサー

クラスタに要求が届くと、ロード バランサーは要求の処理に使用できる最善のSpectrum™ Technology Platformノードを特定します。要求はその Spectrum™ Technology Platformノードに渡されます。

分散アーキテクチャは、ユーザからは見えないところで自動的に処理されます。ユーザは、単独の Spectrum™ Technology Platformサーバーの場合と同じように、ロード バランサーの Spectrum™ Technology Platform の URL とポート (通常、分散環境の場合はポート 80) に要求を送信します。

ノード

ノードとは、インストールされたSpectrum™ Technology Platformサーバーのことです。各ノードは、構成データベースのコピーを保持します。各コピーは常に同期されています。これによって各ノードは、ライセンス情報、データフロー、データベース リソースなど、同じ設定を共有できます。

クラスタを構成するには、Management Console または Enterprise Designer で Spectrum™ Technology Platformに対するロード バランサーの URL とポート (通常、分散環境の場合はポート 80) を指定するだけです。

外部リソース

データベース リソース (郵便データベース、ジオコーディング データベースなど)、JDBC 接続、ファイル サーバーなどの外部リソースの定義は、構成データベースに存在します。リソース自体 (データベース、ファイル、Web サービス) は、ユーザが自由に選択できます。データベース リソースは、クラスタ内の各ノードにインストールするか、または、共有ネットワーク上の場所にインストールすることができます。

データベース リソース自体はクラスタの外部にあるので、複数のクラスタが同じデータベース リソースを共有できます。Management Console を使用してクラスタごとにリソース定義を作成する必要があります。例えば、複数のクラスタで同じジオコーディング データベースを共有する場合は、各クラスタからアクセスできるサーバーにジオコーディング データベースをインストールし、各クラスタがそのジオコーディング データベースを参照するように Management Console で設定します。

クラスタのインストール

クラスタのインストール手順については、『*Spectrum™ Technology Platformインストール ガイド*』を参照してください。

クラスタでの Enterprise Designer の使用

1. Enterprise Designer を起動します。
2. **[サーバー名]** フィールドに、ロード バランサーのサーバー名を入力します。
3. **[ポート]** フィールドに、ロード バランサーがリッスンするように設定したポートを入力します。

> **注：** 入力ファイル、出力ファイル、およびデータベース リソースは、共有ドライブ、ファイル サーバー、またはその他の共通にアクセスできる場所に置く必要があります。そうしない場合は、Spectrum™ Technology Platform サーバーをホストする各サーバーにすべてのファイルをロードする必要があり、すべてのファイルが同じパスに存在する必要があります。

ログインした後は、普通に Enterprise Designer を使用できます。実行するアクションは、ログインしているクラスタのすべての Spectrum™ Technology Platform インスタンスに対して適用されます。

クラスタの起動

次の手順は、サーバーが停止されていることを前提にしています。

クラスタ内のすべてのノードが停止した場合は、次の手順に従ってクラスタを安全に起動し、データの損失を回避する必要があります。

最後に停止したノードで、サーバーを起動します。これを、クラスタ内の各ノードで行います。

警告： 最新のデータを維持するために、最初に起動するノードは、最後に停止したノードでなければなりません。別のノードを最初に起動すると、ジョブ履歴や構成設定などのデータが失われる恐れがあります。最後に停止したノードが不明な場合は、各ノードのログでシャットダウン メッセージのタイム スタンプを確認します。ログは、次の場所にあります。*SpectrumDirectory*\server\logs\spectrum-server.log

a) サーバーを開始します。
b) アップグレードの後に、すべてのノードを続けて起動します。ノード 1 の起動後数秒以内にノード 2 を起動します。残りのノードもこのタイミングで起動してください。

Spectrum™ Technology Platform サーバーが完全に起動したかどうかは、次のログ ファイルで確認できます。*SpectrumDirectory*\server\logs\spectrum-server.logサーバーが完全に起動している場合は、次のメッセージが表示されます。

```
Pitney Bowes Spectrum(TM) Technology Platform (Version Version
Number) Started.
```

クラスタの停止

クラスタ全体を停止するには

1. シード ノードとして使用されているノードを識別します。これを行うには、*SpectrumDirectory*/server/conf/spectrum-container.properties ファイルを開き、spectrum.cluster.seeds プロパティに示される一連のノードに注目します。

2. クラスタに含まれる Spectrum™ Technology Platform サーバーを 1 つずつ停止します。最後に停止するノードが必ずシード ノードになるようにします。

クラスタのアップグレード

必要条件

- アップグレードを実行する前に、新しいバージョンのリリース ノートに目を通してください。リリース ノートには、互換性に関する重要な情報に加えて、サポートされているアップグレード パスや、データ バックアップに関するモジュール固有の推奨事項が記載されています。
- お使いのオペレーティング システムに対して提供されているすべての最新アップデートを適用してください。特に Java 関連の問題を修正するものは必須です。

以下の手順は、Spectrum™ Technology Platform サーバーと構成データベースがクラスタの各ノードにインストールされているクラスタをアップグレードするためのものです。クラスタをアップグレードするには、ノードを 1 つずつ順にアップグレードします。最初にアップグレードするノードに対する処理は、その他のノードと少し異なります。そのノードは、起動時にクラスタ内に他のノードが 1 つも実行していないことになるので、そのノード自体をシード ノードとして指定する必要があるためです。

以下の場合は、クラスタをアップグレードするための特別な手順があることに注意してください。

この場合は...	この情報を使用します...
... サーバー ノード用と構成データベース ノード用に別々のクラスタがある	**データベースが分離されたクラスタのアップグレード**.
... Spatial モジュールのクラスタのみアップグレードする	**Location Intelligence モジュールがあるクラスタのアップグレード**
... Spectrum と Spatial モジュール両方のクラスタをアップグレードする	**Location Intelligence モジュールがあるクラスタのアップグレード**
... Data Hub モジュールを実行しているクラスタをアップグレードする	すべてのノードをシャットダウンする前に、『Spectrum インストール ガイド』の「Data Hub モジュール用のクラスタのアップグレード」を参照してください。

上記のいずれにも該当しない場合は、以下の手順を実行してクラスタをアップグレードします。

1. サーバーをバックアップします。バックアップを作成する手順については、『*管理ガイド*』を参照してください。

2. クラスタ内のすべてのノードを停止します。詳細については、クラスタの停止（544ページ）を参照してください。
3. 最後に停止したノードで、次の処理をします。
   a. ノードをアップグレードします。詳細については、「サーバーのアップグレード」を参照してください。
   b. ファイル `spectrum-container.properties` をテキスト エディターで開き、クラスタ プロパティを設定します。詳細については、「クラスタ プロパティ」を参照してください。
4. 他の各ノードを1つずつ順にアップグレードします。**以下の手順は、最初のノード以外のノードをアップグレードする場合にのみ実行してください。**

   注： 必ずサーバーをバックアップしてから、次の操作に進んでください。この手順は、Spectrum™ Technology Platform バージョン 11.1 またはそれ以前のバージョンからアップグレードする場合にのみ適用されます。

   a. `spectrum.cluster.nodeID` を追加する、または設定済みである場合、最初のノードでこれを “1” に設定すると、この値は後続のノードで増加します。
   b. 次のフォルダがある場合は削除します。
   SpectrumDirectory`\server\repository\store\databases`
   c. ノードをアップグレードします。詳細については、「サーバーのアップグレード」を参照してください。
   d. ファイル `spectrum-container.properties` をテキスト エディターで開き、クラスタ プロパティを設定します。詳細については、「クラスタ プロパティ」を参照してください。設定を終えたら、ファイルを保存して閉じます。
5. アップグレードの後に、すべてのノードを続けて起動します。ノード 1 の起動後数秒以内にノード 2 を起動します。残りのノードもこのタイミングで起動してください。

ソフトウェア更新を適用する場合など、ノードを手動で停止することが必要な場合があります。クラスタのすべてのノードを手動で停止する場合、または、すべてのノードがダウンしている場合は、新しいクラスタ/セッションとして起動する必要があります。

クラスタからのノードの削除

クラスタからノードを削除するには、Spectrum™ Technology Platformサーバーを停止します。

1. 削除するノードを、次の手順で停止します。

- Unix または Linux では、作業ディレクトリを Spectrum™ Technology Platformサーバーの `bin` ディレクトリに変更し、セットアップ ファイルのソースを指定して、コマンド `./server.stop` を入力します。
- Windows の場合は、システム トレイの Spectrum™ Technology Platformアイコンを右クリックして、**[Spectrum™ を停止する]** を選択します。

2. ファイル `server/app/conf/spectrum-container.properties`をテキスト エディターで開き、`spectrum.cluster.enabled` を `false` に設定します。
3. クラスタ内の他の各ノード上で、ファイル `spectrum-container.properties`を開き、`spectrum.cluster.seeds` プロパティからノードを削除します。

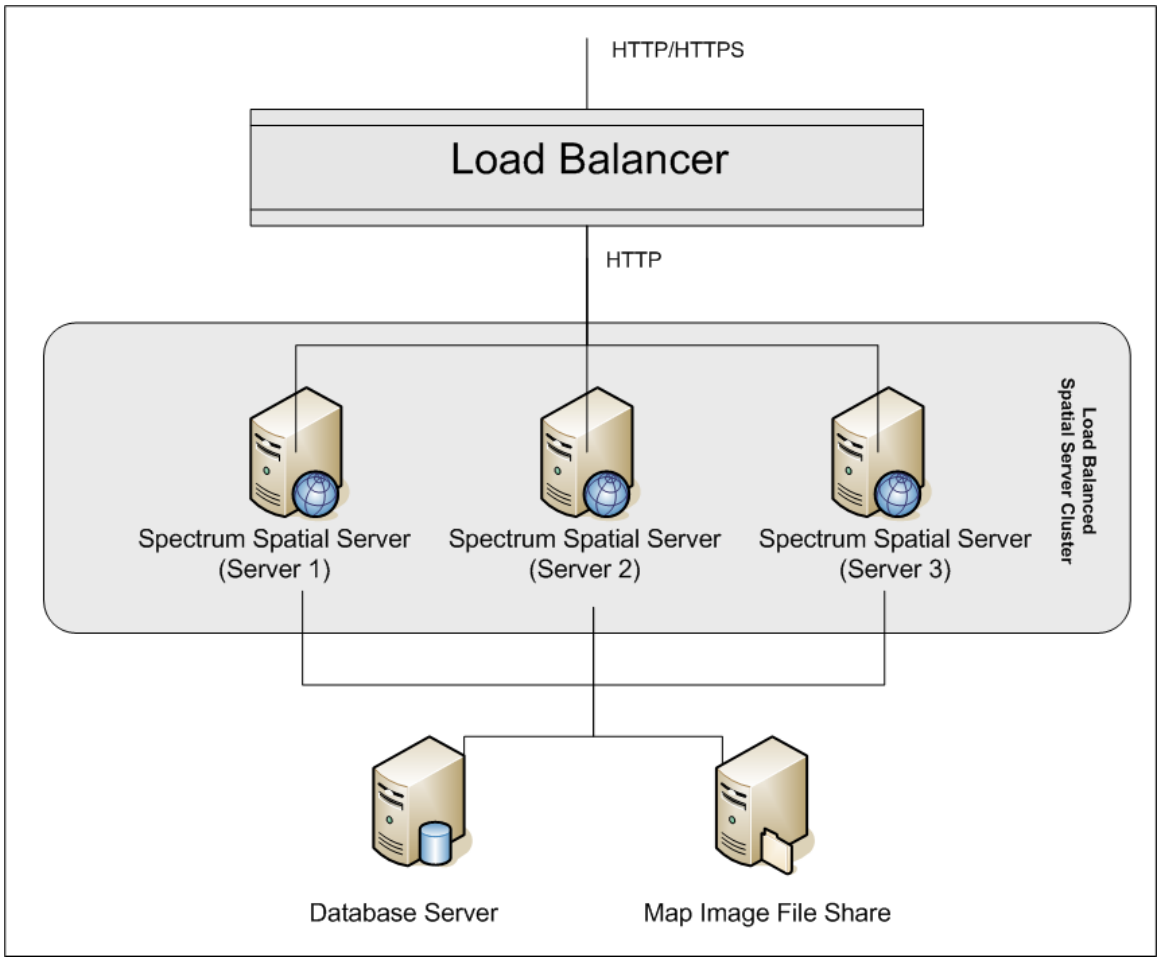
Location Intelligence モジュールを使用する場合: ノードをスタンドアロンのままにして、クラスタの外部で実行できるようにするには、元の `repository.xml` ファイルをコピーして戻し、Spectrum™ Technology Platform の各インスタンスの`/server/modules/spatial/jackrabbit` ディレクトリから、repository、version、workspaces というフォルダを削除します。サーバーを再起動し、リポジトリのコンテンツをインポートします。

Location Intelligence モジュール用のクラスタの管理

Spatial モジュールのクラスタ アーキテクチャ

クラスタ環境では、サーバーの複数のインスタンスが処理を共有します。以下の図は、このような構成の展開アーキテクチャを示したものです。負荷分散を利用することによって、高い可用性とスケーリングをサポートすることができます。展開アーキテクチャは、ロード バランサー (負荷分散サーバー)、Spectrum Spatial クラスタ、データベース、ファイル共有で構成されます。このアプローチによって、水平および垂直の両方のスケーリングが可能になります。Spatial モジュールは、プラットフォームをクラスタリングしてもしなくてもクラスタ化できます。

> **注：** Spectrum™ Technology Platform クラスタと Location Intelligence モジュール クラスタの両方を設定することを推奨します。これには、以下に示すいくつかの利点があります。
>
> - 名前付きリソースのセキュリティ (ACL) 同期が自動的に行われます。
> - 1つのノード上で作成されたデータフロー、ユーザ、役割が、すべてのノードに自動的に同期されます。
> - Location Intelligence モジュールのすべてのデモ ページとユーティリティ (Spectrum Spatial™ Manager など) がロード バランサーを参照できます。



ロード バランサー

ロード バランサーは、Spectrum Spatial インスタンス間にリクエストを分散します。HTTP/HTTPS リクエストの負荷分散をサポートする任意のロード バランサーを使用できます。

Spectrum Spatial クラスタ

このクラスタは、Spatial モジュール共有管理、名前付きリソース、地理メタデータ コンテンツ、および構成設定を伴う Spectrum インスタンスのコレクションです。クラスタにさらにノードを追加することによって、耐障害性を高めたり、さらに高い負荷に対応したりすることができます。各ノードは、ハードウェア リソースを追加するか、または、大規模リソースを伴うハードウェアに対して必要である場合はインスタンスを追加することによって、垂直にスケーリング可能です。Spectrum は、限られた数の CPU を使用するように構成できます。

データベース

Spectrum は名前付きリソース (マップ、レイヤ、テーブル、スタイル)、地理メタデータ、構成をリポジトリに保存します。デフォルトのシングル サーバー構成では、組み込みデータベースを使

用して、ローカル サーバー上にこれらのリソースを保存します。耐障害性に優れたスケーラブルなソリューションを構築するには、この組み込みデータベースを耐障害性を備えた独立したデータベースに置き換える必要があります。Oracle、PostGreSQL/PostGIS、Microsoft SQL Server が、リポジトリ データベースとしてサポートされています。

負荷分散構成において、Spectrum ノードはローカル キャッシュにこれらのリソースをキャッシュし、クラスタの各ノード内のインデックスを検索します。Spectrum ノードはリクエストを受信すると、ローカル キャッシュとインデックスを使用してリソースを検索します。名前付きリソースは、クラスタ内の任意のノードを介して追加できます。各ノードは、ローカル キャッシュと中央データベースとの違いをチェックして、キャッシュを最新の状態に保ちます。このチェックは、デフォルトで2秒ごとに実行されます。チェック時間の間隔は設定可能です。このアーキテクチャによって、サーバーは高いパフォーマンスでトランザクションを処理することができ、リポジトリ データベースの負荷は最小限に抑えられることになります。新しい Spectrum ノードがクラスタに追加されると、キャッシュとインデックスが自動的に作成されます。新しい Spectrum の追加は、障害が発生したノードを復旧したい場合や展開のキャパシティを拡大したい場合に、行われる可能性があります。

ファイル共有

ファイル共有は、Spectrum によって生成されたマップ イメージを格納するためのフォルダを提供します。Web サービスを使用してマップをレンダリングする際、サーバーは、URL を介して返されるマップ イメージまたは base 64 エンコード イメージとして返されるマップ イメージをサポートします。URL が返される場合、マップ イメージはファイルとして保存され、その URL がリクエストされた場合に提供されます。任意の Spectrum ノードがマップ イメージを返せるようにするために、イメージの保存にファイル シェアが使用されます。

共通リポジトリ データベースのセットアップ

クラスタに対して共通リポジトリ データベースを使用するように、Location Intelligence モジュールを設定する必要があります。これによって、名前付きリソース、地理メタデータ、構成設定が、クラスタ全体で管理されるようになります。

リポジトリは、一連の名前付きリソース、地理メタデータ、構成ファイルとともにインストールされています。これらのリソースを共通データベース リポジトリに移行するには、リソースをデフォルトの内部リポジトリ データベースから新しい共有リポジトリ データベースへとエクスポートする必要があります。

リポジトリ コンテンツを一括でエクスポートまたはインポートする場合は`limrepo import`、管理ユーティリティの `limrepo export` コマンドと コマンドを使用します。これらのコマンドには、権限を維持するオプションがあります (手順については『*Spectrum Spatial ガイド*』の「管理」セクションを参照してください)。

PostgreSQL、Oracle、または Microsoft SQL Server のいずれかの共通データベースにリポジトリをセットアップするには、次の手順を実行します。

1. 管理ユーティリティの `limrepo export` コマンドを使用して、すべてのリポジトリ リソースをローカル フォルダにエクスポートします (手順については『*Spectrum Spatial ガイド*』の「管理」セクションを参照してください)。

   インストールしたリポジトリのコンテンツをエクスポートする必要があります。このステップは 1 回だけ実行します。 Spectrum™ Technology Platformのすべてのインスタンスに対し、この時点でのリポジトリのコンテンツを同一にするためです。

2. すべてのノード上で Spectrum™ Technology Platform サーバーを停止します (手順についてはクラスタの停止（544ページ）を参照してください)。
3. Spectrum™ Technology Platform のすべてのノード上で、共通データベースを指定するように設定を変更します。
   a) `repository.<databaseType>.xml` の内容を、`repository.xml` にコピーします。このファイルは `server/modules/spatial/jackrabbit` フォルダにあります。ここで `<databaseType>` は、お使いのデータベースに対する適切なタイプです (postgres、oracle、または mssql)。
   b) `repository.xml` で、次の処理を行います。
      - DataSource セクションを、サーバーのホスト名、ポート、データベース、ユーザ、パスワードで変更します。
      - Cluster セクションを変更して、Node1 のような個別のクラスタ ID を割り当てます。クラスタ内の後続のすべてのノードに一意の ID を割り当てます (Node2、Node3 など)。
      - 変更を `repository.xml` に保存します。
   c) `/server/modules/spatial/jackrabbit` フォルダから、repository、version、workspaces というフォルダを削除します。
4. データベースにこれまでにリポジトリ コンテンツが含まれていたことがある場合は、クリーンなリポジトリが作成できるようにデータベースからテーブルを削除しておく必要があります。以下のテーブルを削除する必要があります。

default_binval	security_binval
default_bundle	security_bundle
default_names	security_names
default_refs	security_refs

rep_fsentry	version_binval
rep_global_revision	version_bundle
rep_journal	version_names
rep_local_revisions	version_refs

Oracle を使用している場合は、`version_seq_names_id`, `security_seq_names_id`、`default_seq_names_id` も削除します。

5. シード ノード上にのみ、バックアップしたリポジトリ コンテンツをインポートします。
   a) Spectrum™ Technology Platform サーバーを起動します (手順については「クラスタの起動（544ページ）」を参照してください)。
   b) `limrepo import` コマンドでシード ノードを指定して、コンテンツをインポートします。
6. クラスタ内の残りのノードを起動します (手順については「クラスタの起動（544ページ）」を参照してください)。

システムの設定

Spectrum™ Technology Platform をインストールして共通リポジトリを設定した後、別の仮想マシンに複製する前に、インスタンスを構成設定する必要があります。 仮想マシン環境を使用していない場合は、インストールされている各 Spectrum™ Technology Platform に対して以下の手順を実行する必要があります。

マップ ファイル共有の設定

Spectrum™ Technology Platform にマップ ファイル共有 (共有イメージ フォルダ) を設定するにはまず、共有マップ イメージ ディレクトリが必要です。マップ ファイル共有の作成については、「**Unix/Linux 上でのマップ イメージ ファイル共有の作成**（552ページ）」または「**Windows 上でのマップ イメージ ファイル共有の作成**」を参照してください。

マップ イメージ ディレクトリを作成したら、以下の手順でマップ ファイル共有を設定します。

1. マッピング サービス構成を、共有イメージ フォルダと負荷分散サーバーを使用するように変更します。ImageCache において、Directory パラメータを共通イメージ ディレクトリに変更し、AccessBaseURL パラメータをロード バランサー マシンのイメージ URL に変更します。

仮想マシン環境を使用している場合は、この IP アドレスを覚えておいてください。ロード バランサー VM をこの IP に設定する必要があるためです。

Unix/Linux インストールの場合:

```
<ImageCache>
<Directory>/<spatial server
root>/server/modules/spatial/images</Directory>
<AccessBaseURL>http://<loadbalance_IP_address>/rest/Spatial/MappingService/internal/imageCache</AccessBaseURL>

 <FileExpire>30</FileExpire>
 <ScanInterval>30</ScanInterval>
</ImageCache>
```

Windows インストールの場合:

```
<ImageCache>
<Directory>\\server\Share\images</Directory>
<AccessBaseURL>http://<loadbalance_IP_address>/rest/Spatial/MappingService/internal/imageCache</AccessBaseURL>

 <FileExpire>30</FileExpire>
 <ScanInterval>30</ScanInterval>
</ImageCache>
```

2. Unix/Linux インストールの場合は、マップ イメージを共有ファイル システムに配置できるようにシンボリック リンクを設定する必要がします。

   マウントされた共有フォルダに `images` サブフォルダを作成します。例: `/mnt/<linux mount>/images`。

```
cd /<spatial server root>/server/modules/spatial
rm -Rf images
ln -s /mnt/<linux mount>/images ./images
```

Unix/Linux 上でのマップ イメージ ファイル共有の作成

ファイル共有は、Spectrum Spatial によって生成されたマップ イメージを格納するためのフォルダを提供します。すべての Spectrum ノードからアクセス可能な共有フォルダを作成します。マップが Base64 エンコード イメージとして Web サービスから返される場合は、ファイル共有は必要ありません。

Unix/Linux 上でのマップ イメージ ファイル共有を作成するには

1. Spectrumをホスティングする各オペレーティング システム上で、共有フォルダをマウントします。以下のコマンドによって、Microsoft Windows Server または CIFS をサポートするネットワーク ドライブ上にドライブをマウントします。

```
mkdir /mnt/<linux mount>
mount -t cifs //<windows host>/<windows share> /mnt/<linux mount>-o
username=shareuser,password=sharepassword,domain=pbi
```

2. `/etc/fstab` において、起動時にイメージ シェアをロードするように設定します。

```
//<windows ip address for share>/share /path_to/mount cifs
username=server_user,password=secret,_netdev 0 0
```

クラスタ用の OGC サービス設定の変更

Spectrum™ Technology Platformクラスタと Location Intelligence モジュール クラスタの両方がある場合のクラスタ環境が正しく動作するように、Spectrum Spatial™ Manager によって OGC サービス設定ファイルを変更する必要があります。WFS、WMS、WMTS の設定ページで、オンライン リソース (サービス) URL をロード バランサーの IP アドレスおよびポートに変更します。詳細については、『*Spectrum Spatial ガイド*』の「ユーティリティ」セクションにある「*Spectrum Spatial™ Manager ガイド*」を参照してください。

複数の Spectrum インスタンスのポートの構成

単一のマシンに複数の Spectrum™ Technology Platform インスタンスがある場合は、各インスタンスのポート番号を変更する必要があります。`<Spectrum root>/server/conf/spectrum-container.properties` 内のすべてのポートを、使用されていない新しいポートに変更します。この HTTP ポートは、インストーラで入力されたポート番号を示しています。

共有される Spectrum ローカル データ

ファイル システム上の TAB ファイル データを使用する場合は、このデータを負荷分散環境に含まれるすべての Spectrum インスタンスがアクセス可能な共有場所に配置する必要があります。また、ファイル システム上のデータにアクセスするリポジトリ内のすべての名前付きリソースが、この共有場所を指す必要があるということも、重要な点です。

Spectrumをホスティングする各VMまたはマシンは、マウントされた共有ドライブにアクセスできる必要があります。

> **注：** データベース テーブルを指す名前付きリソースを使用するのに、共有ドライブは必要ではありません。リポジトリ内の名前付きリソースは、ファイル パスを使用してデータにアクセスするのではなく、データベース内のデータへの名前付き接続を使用するためです。

14 - Spectrum™ Technology Platform について

このセクションの構成

Spectrum™ Technology Platform とは

Spectrum™ Technology Platform は、データの正規化、検証、拡張 (価値向上) の 3 つの側面からデータの完全性、妥当性、一貫性、適時性、および正確性を高めるシステムです。データを正確かつ包括的に、最新の状態に維持することで、顧客への理解を深め、顧客とより良好な関連性を構築できます。

Spectrum™ Technology Platform は、以下の機能を実行して、データの品質を高めるビジネス ルールの設計と実装を支援します。

パーシング、名前の正規化、名前のバリデーション

正規化をきわめて正確に実行するには、一連のデータ列を複数のフィールドに分割する必要がある場合があります。Spectrum™ Technology Platform は、個人名、企業名、およびその他多くの語や略語をパースする高度なパーシング機能を備えています。また、スキャン/抽出操作のベースとして使用するカスタム用語のリストを独自に作成することもできます。Universal Name モジュールは、この機能を備えています。

重複除外統合

一意のエンティティを識別することで、レコードを統合する、重複レコードを排除する、および "最良の組み合わせ" レコードを作成できます。"最良の組み合わせ" レコードとは、別のレコードのデータを使用して作成する複合的なレコードです。Advanced Matching モジュールと Data Normalization モジュールは、この機能を備えています。

住所検証

住所検証では、管轄の郵便当局のルールを適用して、住所を標準形式に変換し、その住所が配達可能な住所であるかどうかを確認します。住所検証により、郵便料金の割引を受けやすくなり、郵便物の配達品質を高めることができます。Universal Addressing モジュールと Address Now モジュールは、この機能を備えています。

ジオコーディングのマッチ コードとロケーション コード

ジオコーディングとは、住所を地図上のポイント（緯度と経度）に変換する処理です。ジオコーディングは、地図製作に使用されますが、それは 1 つの使用例にすぎません。基盤を成すロケーション データがあると、ビジネス上の意思決定を行いやすくなります。処理を逆にすることで、ジオコード (緯度と経度で表現される地図上のポイント) を入力し、そのジオコードに関する住所情報を取得できます。Enterprise Geocoding モジュールは、この機能を備えています。

Location Intelligence

ロケーション インテリジェンスは、地理関係を調査、評価、分析、およびモデル化して、データに関する新しい情報を作成します。ロケーション インテリジェンス処理を使用すると、ロケーションを検証し、情報を有益なビジネス インテリジェンスに変換できます。Location Intelligence モジュールは、この機能を備えています。

マスターデータ管理

マスター データ管理では、重要なデータ アセットの関連性を中心に捉えたマスター データ ビューを作成できます。Data Hub モジュールは、インフルエンサーと明白でない関連性の特定、詐欺行為の検出、情報の品質、統合、およびアクセシビリティを高めるのに役立ちます。

税務管轄区域の割り当て

税務管轄区域の割り当てでは、住所の地域に適用される税務管轄区域を判断します。税務管轄区域を最も正確に割り当てると、経済上のリスクや、法的義務を軽減できます。

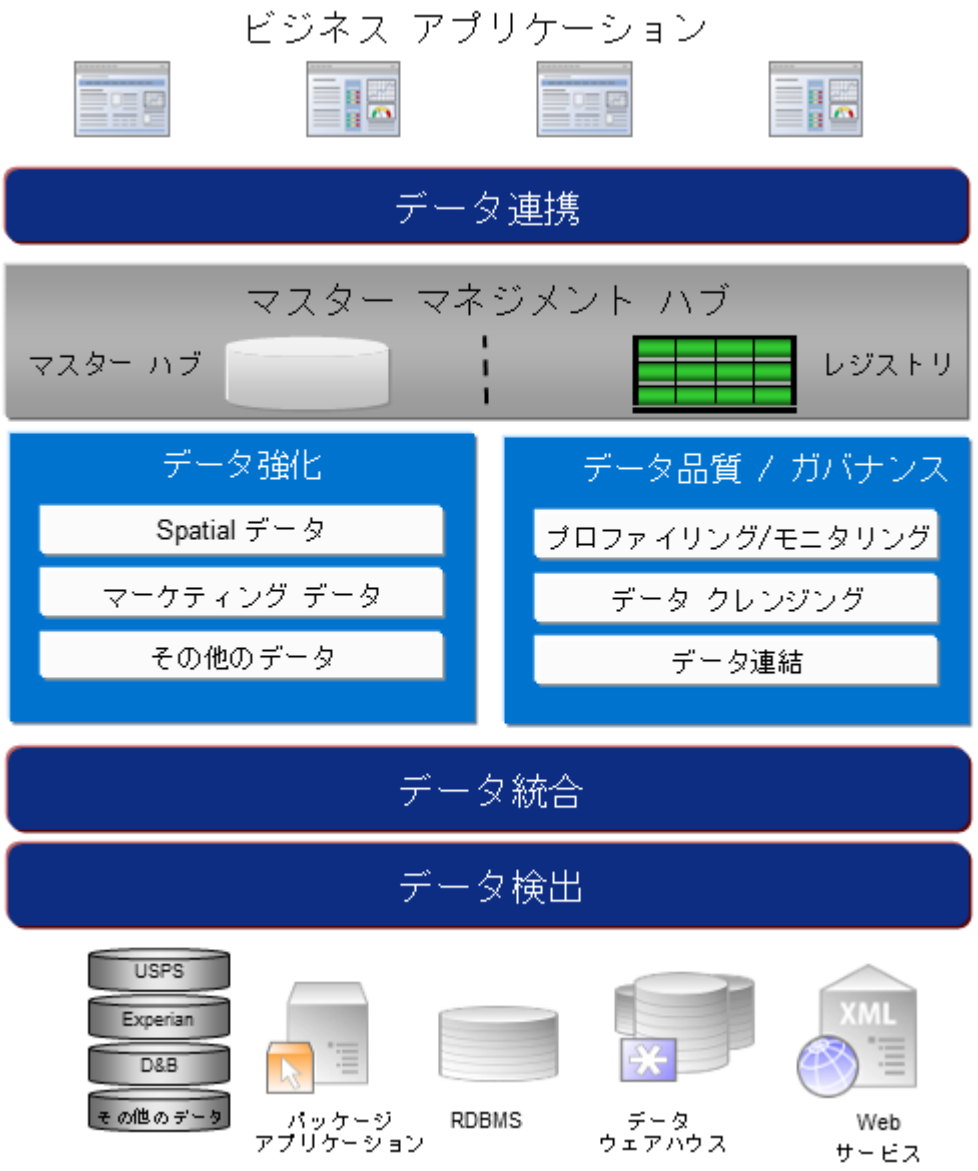
Spectrum™ Technology Platform が提供する Pitney Bowes ソフトウェアでは、最新の税務管轄区域と顧客レコードの正確な住所を統合して正確な州、郡、郡区、市、および特殊な税務管轄区域の情報をレコードに追加できます。税務管轄区域の割り当ての使用例を次に示します。

- 売上税と使用税
- 動産税
- 保険料税

Enterprise Tax モジュールは、この機能を備えています。

エンタープライズ データ管理アーキテクチャ

Spectrum™ Technology Platform では、包括的なエンタープライズ データ管理処理を構築できます。あるいは、対象をさらに絞り込んだソリューションとしてこれを活用することも可能です。次の図は、ソースからデータを取得し、データ強化およびデータ品質処理を経て、マスター データ管理ハブに引き渡す、包括的なソリューションを示したものです。MDM ハブは、データの単一のビューを作成して複数のビジネス アプリケーションに提供します。



データ検出

データ検出は、データ リソースをスキャンしてデータの状況を詳細に把握するプロセスです。Spectrum™ Technology Platform は、さまざまなデータプロファイリング手法を使用して、構造化されたデータ、構造化されていないデータ、および一部分のみ構造化されたデータをスキャンできます。スキャン結果は、会社のデータ アセットを記述するドキュメントのライブラリの生成とメタデータ リポジトリの作成に自動的に使用されます。このドキュメントと付属のメタデータ リポジトリから提供される情報を十分に吟味したうえで、データ統合、データ品質、データ制御、またはマスター データ管理プロジェクトを始めてください。

Spectrum™ Technology Platform のデータ検出モジュールの詳細については、営業担当者にお問い合わせください。

データ統合

データの状況を把握したら、次は、管理する必要があるデータへのアクセス方法を検討する必要があります。Spectrum™ Technology Platform は、複数のソースのデータに直接接続できます。また、既存のデータ アクセス手法を統合した方法で接続することもできます。データ ウェアハウス、データ品質、システム統合、移行といった多様なビジネス ニーズに対応するバッチおよびリアル タイム データ統合機能をサポートします。Spectrum™ Technology Platform は RDBMS データベース、データ ウェアハウス、XML ファイル、フラット ファイルなどのデータにアクセスできます。Spectrum™ Technology Platform は、複雑な結合や集計を含むSQL クエリをサポートし、視覚的なクエリ開発ツールを提供します。また、Spectrum™ Technology Platform は REST および SOAP Web サービスを介してデータにアクセスできます。

Spectrum™ Technology Platform は、指定されたフォルダ内の1つ以上のソースファイルの存在チェック結果に基づいてバッチ処理をトリガできます。この "ホット フォルダ" トリガは、FTP アップロードの監視と、アップロード直後の処理に役立ちます。

これらのデータ統合機能の一部には、Enterprise Data Integration モジュールのライセンスが必要です。詳細については、営業担当者に問い合わせてください。

最後に、Spectrum™ Technology Platform は SAP や Siebel などのパッケージ アプリケーションと統合可能です。

データ品質/ガバナンス

データ品質およびデータ ガバナンス処理では、重複レコード、矛盾した情報、不正確な情報がないか、データを確認します。

重複マッチングは、重複レコードの可能性や、レコード間の関連性を特定します。データが実際の名前や住所であるか、または他の種類の顧客情報であるかは関係ありません。Spectrum™ Technology Platform では、boolean 型マッチング方式、スコアリング方式、しきい値、アルゴリズム、および重みを使用する一貫したビジネスマッチルールを指定して、レコードのグループに重複が含まれているかどうかを調べることができます。Spectrum™ Technology Platform は、多種多様なカスタマイズをサポートしているため、ビジネス固有のニーズに適合するようにルールを調整できます。

重複レコードを特定したら、それらのレコードを統合することもできます。Spectrum™ Technology Platform は、重複レコードをリンクまたは結合して、収集した顧客情報から最も正確かつ包括的なレコードを作成する方法を指定できます。例えば、ある世帯のすべてのレコードに基づいて、1 つの Best-of-Breed (最良の組み合わせ) レコードを作成できます。重複の特定とその排除には、Advanced Matching モジュールを使用します。

データ品質処理では、データの正規化も行われます。正規化は、きわめて重要な処理です。レコードの照合とレコード間の関連性の識別において、最も可能性の高い結果を得るために、正規化データ要素が必要であるためです。モジュールによっては、複数のタイプの正規化を実行するものもありますが、Spectrum™ Technology Platform の Data Normalization モジュールは最も包括的な正規化機能セットを備えています。また、Universal Name モジュールは、個人名や企業名データを処理するための特定のデータ品質機能を提供します。

正規化データは、必ずしも正確なデータではありません。Spectrum™ Technology Platform は、データを既知の最新の参照データと比較して、その妥当性を確認できます。この処理に用いられるソースとしては、米国郵政公社などの規制機関、Experian やD&B などのサードパーティのデータプロバイダ、会計データなどの企業内の参照ソースがあります。Spectrum™ Technology Platform は、住所データの検証に特に優れています。世界中の 250 の国および地域の住所の検証または正規化が可能です。住所の検証を実行するモジュールには、Address Now モジュールと Universal Addressing モジュールの 2 つがあります。

どちらのモジュールがニーズに適しているかは、営業担当者と相談して判断してください。

Spectrum™ Technology Platform は、幅広いデータ品質問題を自動的に処理できますが、データスチュワードによる手動確認が適切である場合が存在します。これをサポートするために、Business Steward モジュールでは、手動確認をトリガするルールを指定するための方法と、例外レコードを確認するための Web ベースのツールが提供されています。確認および解決処理においてデータスチュワードを支援するための、Bing マップや Experian データといったサードパーティ ツールへの統合アクセスも含まれています。

データ強化(データ・エンリッチメント)

データ強化処理は、追加情報によってデータを増補します。強化は、データに詳細情報を追加するためにユーザが使用したいと考える、空間データ、マーケティング データ、または他のソースからのデータに基づいて行うことができます。例えば、顧客住所のデータベースが存在する場合、住所のジオコーディングを行って、住所の緯度/経度座標を特定し、その座標をレコードの一部として保存することができます。これによって顧客データは、顧客に最も近い銀行支店の検索など、多様な空間分析に使用できるようになります。Spectrum™ Technology Platform では、データをさまざまな情報で強化できます。例えば、ジオコーディング (Enterprise Geocoding モジュールを使用)、税務管轄区域の割り当て (Enterprise Tax モジュール)、地理空間分析 (Location Intelligence モジュール)、2 点間の車移動または徒歩経路 (Enterprise Routing モジュール) の情報を利用できます。

マスター データ管理ハブ

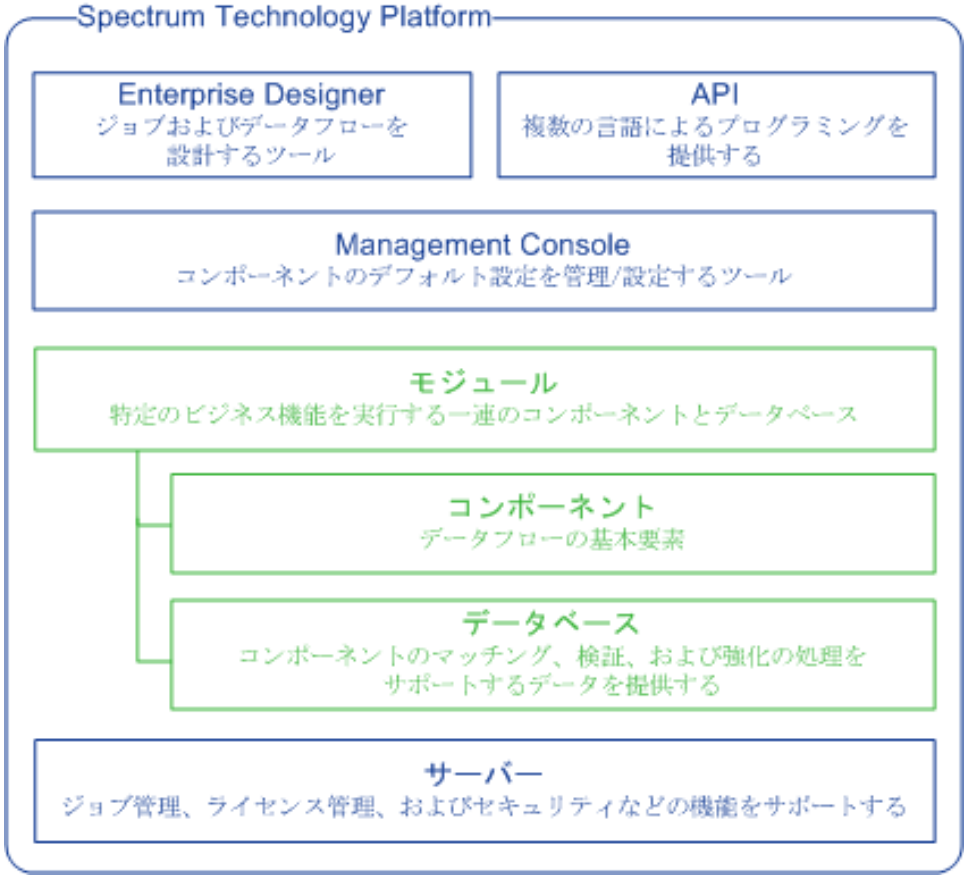
マスター データ管理 (MDM) ハブは、エンティティと、その役割、処理、やり取りの間の複雑な関連性の迅速なモデリングを可能にします。ソーシャル ネットワーク分析機能が組み込まれており、インフルエンサー (influencer) の理解、チャーンの予測、明白でない関係や不正パターンの検出、レコメンデーションを支援します。

Spectrum™ Technology Platform は、MDM ハブに対する 2 つのアプローチをサポートします。マスター ハブのアプローチでは、データは単一の MDM データベースに維持され、アプリケーションは MDM データベースからデータにアクセスします。レジストリのアプローチでは、データは各ビジネス アプリケーションに維持され、MDM ハブ レジストリに、関連レコードの検索に用いられるキーが含まれます。例えば、顧客のレコードが、注文入力データベースと顧客サポートデータベースに存在する場合があります。この場合MDM レジストリには、両方の場所の顧客データへのアクセスに使用できる単一のキーが含まれます。

Data Hub モジュールが、MDM 機能を提供します。

Spectrum™ Technology Platform のアーキテクチャ

Spectrum™ Technology Platform が提供する Pitney Bowes は、いくつかのモジュールを実行するサーバーで構成されます。これらのモジュールは、住所のバリデーション、ジオコーディング、高度なパーシング等、さまざまな機能を備えています。次の図に、Spectrum™ Technology Platform のアーキテクチャを示します。



サーバー

このサーバーが Spectrum™ Technology Platform の基盤となります。サーバーは、データを処理し、リポジトリ データを同期し、通信を管理します。また、ジョブ管理とセキュリティ機能も提供します。

モジュール

モジュールは、特定の機能を実行する機能群です。例えば、Universal Addressing モジュールは、郵便規格に準拠するように住所を正規化します。Enterprise Tax モジュールは、その住所に該当する税務管轄区域を判定します。共通のビジネス問題を解決する各種モジュールがまとめられて、バンドルとしてライセンス供与されています。

コンポーネント

モジュールは、特定の機能をフロー内で実行するか、サービスとして実行するコンポーネントで構成されます。例えば、Enterprise Geocoding モジュールの Geocode US Address コンポーネントは、住所を地図上のポイント (緯度と経度) に変換して返します。Universal Addressing モジュールの Get City State Province は、郵便番号が示す都市および州/省を返します。

システムで利用できるコンポーネントは、Spectrum™ Technology Platform のどのバンドルのライセンスを取得したかによって異なります。

データベース

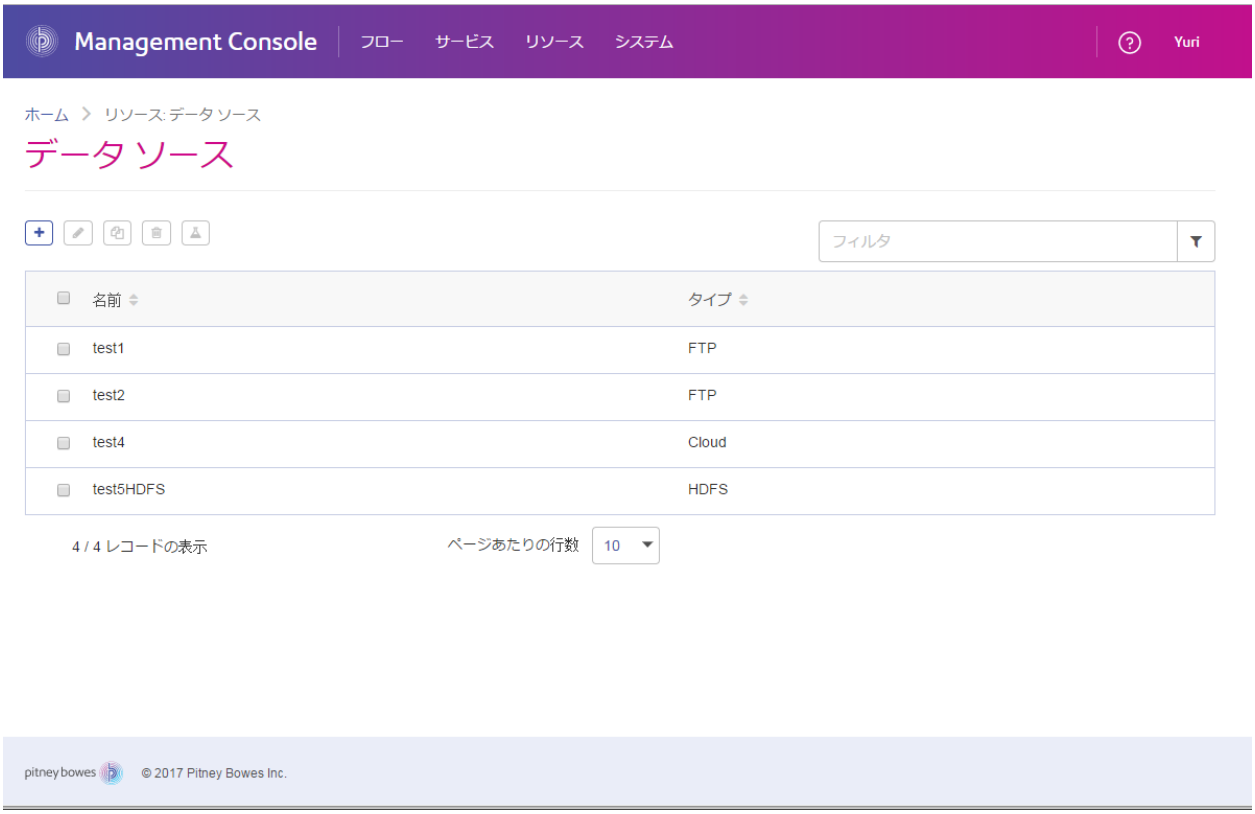
一部のモジュールは、参照データを含むデータベースに依存します。例えば、Universal Addressing モジュールは、米国の住所を検証して正規化するために米国郵政公社 (USPS) のデータにアクセスする必要があります。データベースは個別にインストールされ、一部のデータベースは最新データを提供するために定期的に更新されます。

モジュールには、必須データベースとオプションのデータベースがあります。オプションのデータベースは、Spectrum™ Technology Platform の処理を強化することのできる特定の機能に必要なデータを提供します。

Management Console

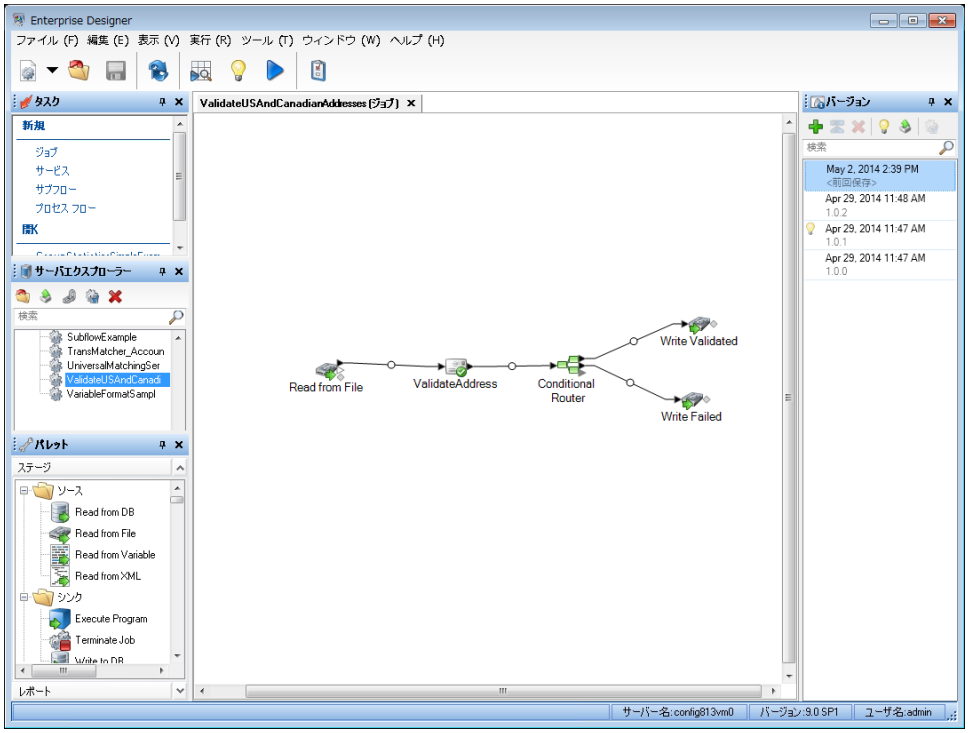
Management Console は、Spectrum™ Technology Platform を管理するためのツールです。Management Console で実行できる操作は、次のとおりです。

- Spectrum™ Technology Platform とデータの間の接続を定義する。
- サービスやフローのデフォルト設定を指定します。
- 権限やパスワード等、ユーザ アカウントを管理する。
- ログを表示する。
- ライセンス有効期限情報を含めて、ライセンスを表示する。



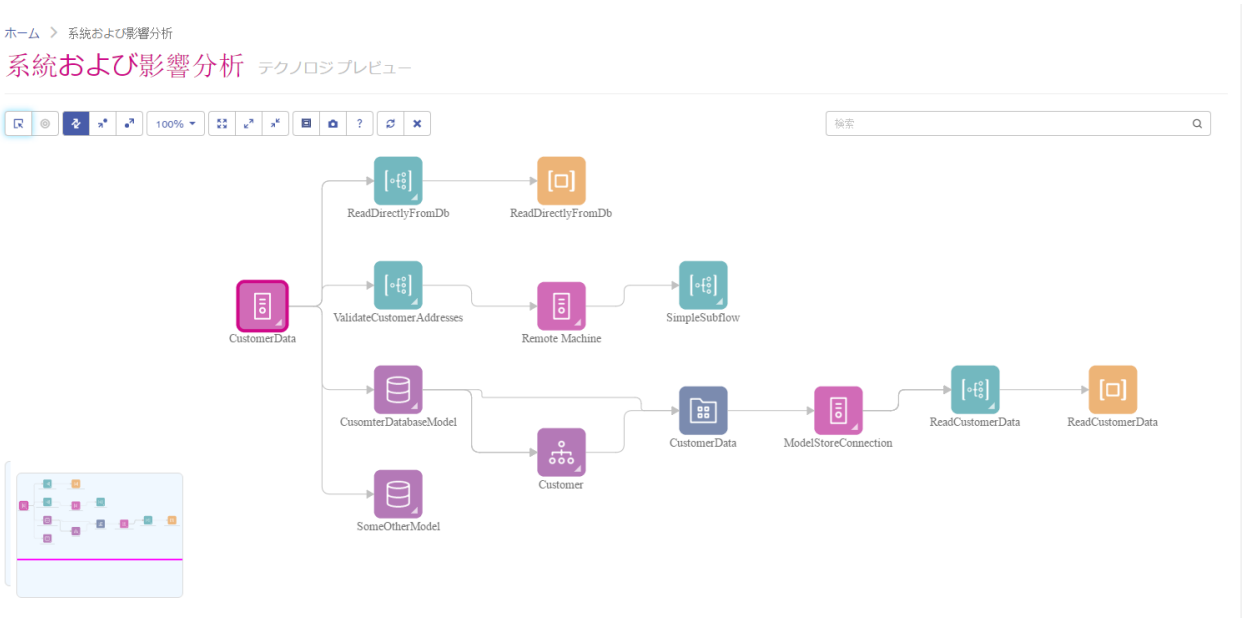
Enterprise Designer

Enterprise Designer は、Spectrum™ Technology Platform のジョブ、サービス、サブフロー、およびプロセス フローを作成するためのツールです。ドラッグアンドドロップ インターフェイスを利用して、複雑なデータフローをグラフィカルに、容易に作成できます。



Metadata Insights

Metadata Insights を使用すると、適切な時間に収集された正確なデータに基づくビジネス分析を得るために必要な制御が可能になります。Metadata Insights を使用して、データ モデルを開発し、ソースからビジネス アプリケーションまでのデータの流れを表示し、プロファイリングによってデータの品質を評価します。この分析を活用すれば、特定のビジネスの課題を解決できるデータ リソースの特定、ビジネス全体でデータの有益性と一貫性を向上するプロセスの適合と最適化、およびデータの問題のトラブルシューティングを行うことができます。



Web サービスと API

Web サービスとプログラミング API を使用して、Spectrum™ Technology Platform の機能を独自のアプリケーションに統合することができます。これらのインターフェイスは、シンプルな統合とレコード処理の効率化を可能とし、将来のバージョンの下位互換性をサポートします。

Spectrum™ Technology Platform API は、以下の言語で使用可能です。

- C
- C++
- COM
- Java
- .NET

Web サービスは、SOAP および REST を介して提供されています。

モジュールとコンポーネント

表 3 : モジュールとコンポーネント

モジュール	説明	コンポーネント
Address Now モジュール	米国以外の住所についての高度なバリデーションと正規化を提供します。また、別の住所処理も提供します。	Build Global Address Get Global Candidate Addresses Validate Global Address
Advanced Matching モジュール	入力ファイル内や入力ファイル間でレコードを照合します。	Best Of Breed Candidate Finder Duplicate Synchronization Filter Interflow Match Intraflow Match Match Key Generator Transactional Match
Analytics Scoring モジュール	入力ファイル内や入力ファイル間でレコードを照合します。	PMML Model Scoring Read from Miner Dataset Write to Miner Dataset
Business Steward モジュール	例外レコードを特定し、例外レコードを手動で確認するためのブラウザベースのツールを提供します。	Exception Monitor Read Exceptions Write Exceptions
Country Identifier	国名を、または郵便番号と州/省の組み合わせを受け取って、2 文字の ISO 国コード、3 文字の Universal Postal Union (UPU) コード、および国名を英語で返します。	Country Identifier

モジュール	説明	コンポーネント
Data Hub モジュール	データをリンクおよび分析して、関係と傾向を識別します。	Write to Hub Read From Hub Query Hub Graph Visualization
Data Integration モジュール	データ ウェアハウジング、データ品質、システム統合、および移行に便利な機能を備えています。	Field Selector Generate Time Dimension Query Cache Write to Cache
Data Normalization モジュール	データの不整合を取り除きます。	Advanced Transformer Open Parser Table Lookup Transliterator
Enterprise Data Integration	データ ウェアハウジングや、データ品質、システム統合、移行などのさまざまなビジネス ニーズに対応するために複数のソースのデータに接続します。	Call Stored Procedure Field Selector Generate Time Dimension Query Cache Write to Cache
Enterprise Geocoding モジュール	住所を表す地図上のポイント（緯度と経度）に変換します。また、指定された緯度と経度を住所に変換します。	Geocode Address AUS Geocode Address GBR Geocode Address Global Geocode Address World Geocode US Address GNAF PID Location Search Reverse APN Lookup Reverse Geocode Address Global Reverse Geocode US Location

モジュール	説明	コンポーネント
Enterprise Routing モジュール	自動車移動または徒歩の経路を取得し、移動時間および移動距離を計算し、始点から一定の時間または距離内に含まれる領域を特定します。	Get Route Data Get Travel Boundary Get Travel Cost Matrix Get Travel Directions Persistent Update
Enterprise Tax モジュール	特定の地域に適用する税務管轄区域を決定します。	Assign GeoTAX Info Calculate Distance
GeoConfidence モジュール	指定された領域に住所または交差点が含まれる可能性を判定します。	Geo Confidence Surface CreatePointsConvexHull
Global Addressing モジュール	米国以外の住所に対する高度な住所の正規化および検証機能を提供します。また、入力の途中で住所を自動的に予測し、入力に基づく候補を直ちに返します。機械学習の手法を使用して、郵便住所文字列を個々の住所要素に分割します。	Global Address Validation Global Type Ahead グローバル住所パーサー
Global Geocoding モジュール	住所を表す地図上のポイント（緯度と経度）に変換します。また、指定された緯度と経度を住所に変換します。	Global Geocode Global Reverse Geocode
Global Sentry	政府から提供される、各国のデータを含む警戒リストとトランザクションとの照合を試みます。	Global Sentry Global Sentry Address Check Global Sentry ID Number Check Global Sentry Name Check Global Sentry Other Data Check

モジュール	説明	コンポーネント
Location Intelligence モジュール	さまざまな地理空間データベースと照合して Point In Polygon と半径分析を実行します。	Closest Site Find Nearest Point In Polygon Query Spatial Data Read Spatial Data Spatial Calculator Spatial Union
Machine Learning モジュール	数値データをビニングし、教師ありと教師なしの機械学習モデルを適合し、これらのモデルでデータをスコアリングするための機能を提供します。	Binning Binning Lookup Java Model Scoring K-Means Clustering Linear Regression Logistic Regression Principal Component Analysis Random Forest Classification Random Forest Regression
Metadata Insights	適切な時間に収集された正確なデータに基づくビジネス分析を得るために必要な制御を可能にします。データ モデルを開発し、ソースからビジネス アプリケーションまでのデータの流れを表示し、プロファイリングによってデータの品質を評価できます。この分析を活用すれば、特定のビジネスの課題の解決に使用すべきデータ リソースの特定や、ビジネス全体でデータの有益性と一貫性を向上するプロセスの最適化を行うことができます。	モデル (論理および物理) Model Store プロファイル 系統および影響分析

モジュール	説明	コンポーネント
SAP モジュール	Spectrum™ Technology Platform と SAP Customer Relationship Management モジュール アプリケーションとの連携を有効にします。	SAP Generate Match Key SAP Generate Match Score SAP Generate Search Key SAP Generate Search Key Constant SAP Generate Search Key Metaphone SAP Generate Search Key Substring SAP Validate Address With Candidates
Siebel モジュール	Spectrum™ Technology Platform と Siebel アプリケーションとの連携を有効にします。	Siebel Generate Match Key Siebel Generate Match Score Siebel Generate Search Key Siebel Business Name Standardization Siebel Standardize Name. Siebel Geocode US Address With Candidates Siebel Geocode US Address With No Candidates Siebel Get Global Candidate Addresses Siebel Validate Address With Candidates Siebel Validate Address With No Candidates
Universal Addressing モジュール	郵便当局の規格に従って、住所を正規化してバリデーションを実行します。	Get Candidate Addresses Get City State Province Get Postal Codes Validate Address Validate Address AUS Validate Address Global

モジュール	説明	コンポーネント
Universal Name モジュール	個人名、企業名、住所、およびその他多くの語や略語をパースします。	Name Parser (非推奨) Name Variant Finder Open Name Parser

著作権に関する通知

USPS® 情報

Pitney Bowes Inc. は、ZIP + 4® データベースを光学および磁気媒体に発行および販売する非独占的ライセンスを所有しています。CASS、CASS 認定、DPV、eLOT、FASTforward、First-Class Mail、Intelligent Mail、LACS^{Link}、NCOA^{Link}、PAVE、PLANET Code、Postal Service、POSTNET、Post Office、RDI、Suite^{Link}、United States Postal Service、Standard Mail、United States Post Office、USPS、ZIP Code、および ZIP + 4 の各商標は United States Postal Service が所有します。United States Postal Service に帰属する商標はこれに限りません。

Pitney Bowes Inc. は、NCOA^{Link}® 処理に対する USPS® の非独占的ライセンスを所有しています。

Pitney Bowes Software の製品、オプション、およびサービスの価格は、USPS® または米国政府によって規定、制御、または承認されるものではありません。RDI™ データを利用して郵便送料を判定する場合に、使用する郵便配送業者の選定に関するビジネス上の意思決定が USPS® または米国政府によって行われることはありません。

データ プロバイダおよび関連情報

このメディアに含まれて、Pitney Bowes Software アプリケーション内で使用されるデータ製品は、各種商標によって、および次の 1 つ以上の著作権によって保護されています。

© Copyright United States Postal Service.All rights reserved.

© 2014 TomTom.All rights reserved.TomTom および TomTom ロゴは TomTom N.V の登録商標です。

© 2016 HERE

Fuente: INEGI (Instituto Nacional de Estadística y Geografía)

電子データに基づいています。© National Land Survey Sweden.

© Copyright United States Census Bureau

© Copyright Nova Marketing Group, Inc.

このプログラムの一部は著作権で保護されています。© Copyright 1993-2007 by Nova Marketing Group Inc. All Rights Reserved

© Copyright Second Decimal, LLC

© Copyright Canada Post Corporation

この CD-ROM には、Canada Post Corporation が著作権を所有している編集物からのデータが収録されています。

© 2007 Claritas, Inc.

Geocode Address World データ セットには、**http://creativecommons.org/licenses/by/3.0/legalcode** に存在するクリエイティブ コモンズ アトリビューション ライセンス (「アトリビューション ライセンス」) の下に提供されている GeoNames Project (**www.geonames.org**) からライセンス供与されたデータが含まれています。お客様による GeoNames データ (Spectrum™ Technology Platform ユーザ マニュアルに記載) の使用は、アトリビューション ライセンスの条件に従う必要があり、お客様と Pitney Bowes Software, Inc. との契約と、アトリビューション ライセンスの間に矛盾が生じる場合は、アトリビューション ライセンスのみに基づいてそれを解決する必要があります。お客様による GeoNames データの使用に関しては、アトリビューション ライセンスが適用されるためです。

3001 Summer Street
Stamford CT 06926-0700
USA

www.pitneybowes.com