

Spectrum™ Technology Platform

バージョン 2019.1.0

Enterprise Data Integration ガイド



目次

1 - はじめに

エンタープライズ データ管理アーキテクチャ	5
スター スキーマによるデータ ウェアハウス設計	8

2 - データ ソースおよびデータ ウェアハウスへの接続

データ ソース接続	14
接続の定義	15
クラウド ファイル サーバーの圧縮のサポート	75
接続の削除	75
操作方法ビデオ - 接続の設定	76

3 - データ ウェアハウスの設定

データの準備	78
時間ディメンション テーブルの設定	79
ディメンション テーブルの設定	80
ファクト テーブルの設定	82
データ ウェアハウス内のレコードへのタイム スタンプの追加	87

4 - データ ウェアハウスの更新

データ ウェアハウスの更新スケジュールの定義	91
ファクト テーブルの更新	92
クエリのためのグローバル キャッシュの使用	97
クエリのためのローカル キャッシュの使用	99

5 - ステージ リファレンス

Call Stored Procedure	103
DB Change Data Reader	107
DB Loader	110
Field Parser	121
Field Combiner	124
Field Selector	126
Generate Time Dimension	127
Query Cache	134
Query DB	148
Query NoSQL DB	152
Read From DB	155
Read From File	165
Read from Hadoop Sequence File	183
Read from Hive File	188
Read From HL7 File	191
Read from NoSQL DB	204
Read from SAP	209
Read from Spreadsheet	215
Read from Variable Format File	218
Read From XML	233
SQL Command	241
Transposer	252
Unique ID Generator	255
Write to Cache	265
Write to DB	267
Write to File	273
Write to Hadoop Sequence File	292
Write to Hive File	296
Write to NoSQL DB	303
Write to Spreadsheet	307
Write to Variable Format File	310
Write to XML	321
日付および数字パターン	331

6 - 設定

Oracle LogMiner の設定 339

7 - パフォーマンスの最適化

最適なフェッチ サイズの決定 342

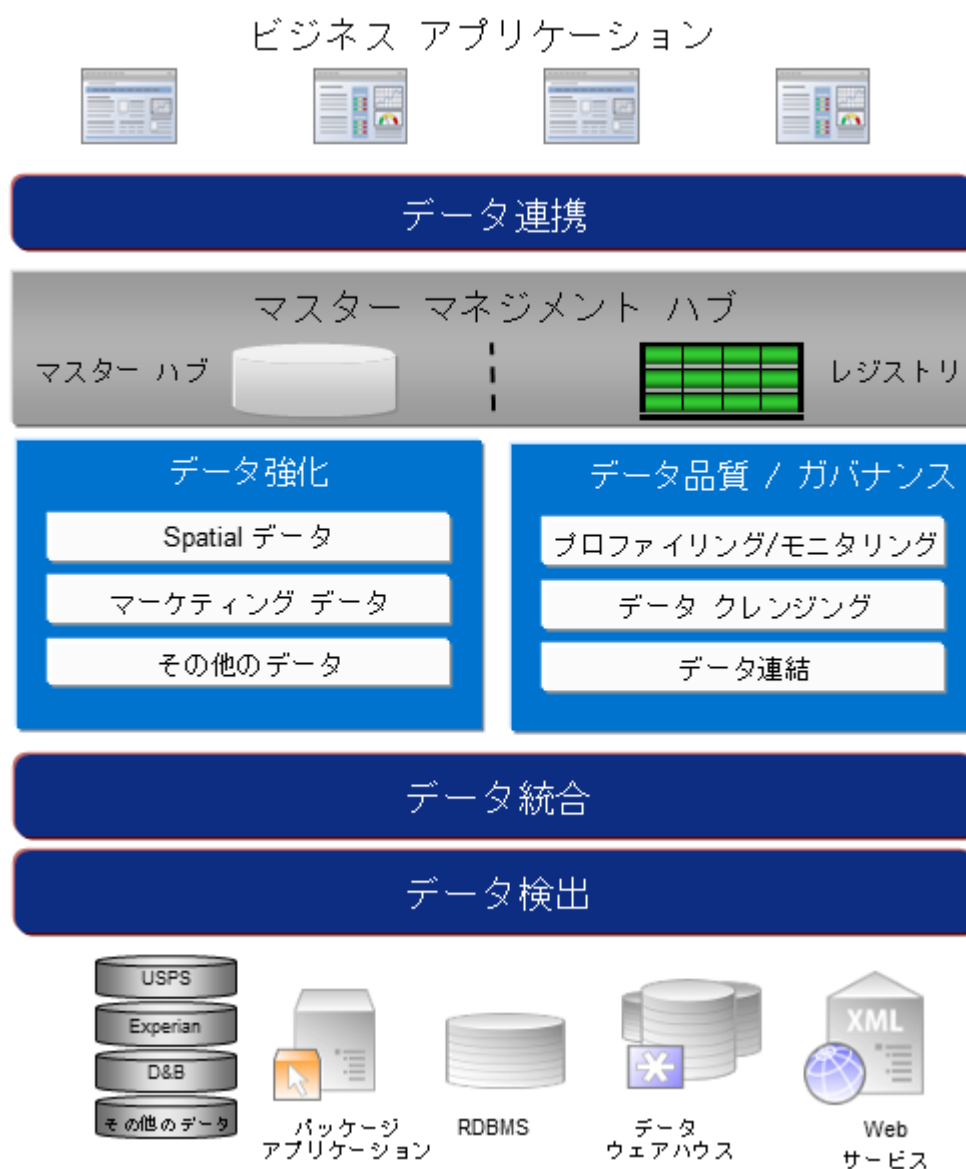
1 - はじめに

このセクションの構成

エンタープライズ データ管理アーキテクチャ	5
スター スキーマによるデータ ウェアハウス設計	8

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、不正確な情報がなければ、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せん。

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。

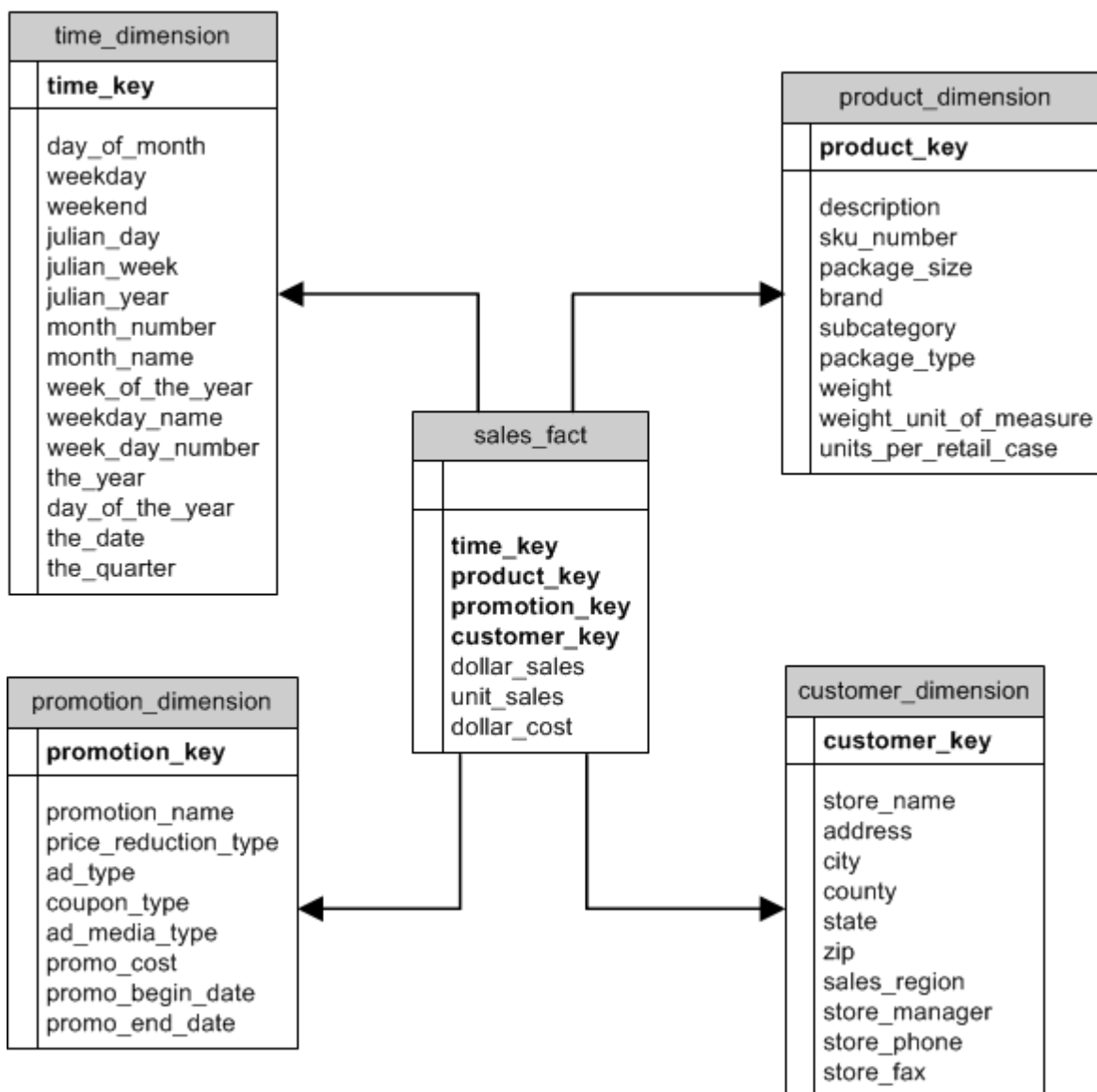
Spectrum™ Technology Platform は、MDM ハブに対する 2 つのアプローチをサポートします。マスター ハブのアプローチでは、データは単一の MDM データベースに維持され、アプリケーションは MDM データベースからデータにアクセスします。レジストリのアプローチでは、データは各ビジネス アプリケーションに維持され、MDM ハブ レジストリに、関連レコードの検索に用いられるキーが含まれます。例えば、顧客のレコードが、注文入力データベースと顧客サポートデータベースに存在する場合があります。この場合 MDM レジストリには、両方の場所の顧客データへのアクセスに使用できる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。

スター スキーマによるデータ ウェアハウス設計

Spectrum™ Technology Platform は、スター スキーマ設計を使用したデータ ウェアハウスの作成とメンテナンスをサポートしています。スター スキーマでは、イベントの具体的な記述であるファクト、またはファクト テーブルでのファクトの記述であるディメンション属性のどちらかとしてデータが保存されます。ファクトは定期的に変化しますが、ディメンションはゆっくりと変化するか、またはまったく変化しません。

次の図に、スター スキーマの設計を示します。



この図には、スタースキーマの主な特徴である、ファクトテーブル、ディメンションテーブル、および結合が示されています。

ファクト テーブル

ファクトテーブルは、データウェアハウスのスタースキーマで中心となるテーブルです。通常、ファクトテーブルには、特定のイベントを記述する数値的または定量的な情報 (尺度と呼びます) が含まれます。例えば、会社の収益に関するレポートを生成するために使用するデータウェアハウスの場合、ファクトテーブル内に **dollar_sales** および **dollar_cost** という列を持つことになります (上の図を参照)。一般的に、ファクトは継続的に評価され、追加的なものです。"継続的に評価される"とは、ファクトが、測定のために別の値を持つ数値的な測定データであることを意味します。"追加的"とは、追加によってファクトを集約できることを意味します。

ファクトテーブルには、連結されたキー、つまり複合キーを形成する一連の列も含まれます。連結されたキーの各列は、ディメンションテーブルの主キーから得られた外部キーです。例えば、上の図では、ファクトを `product_dimension` テーブル内の特定の製品と関連付ける `product_key` という列がファクトテーブルに含まれています。

ファクトテーブル内の詳細レベルを粒度と呼びます。ファクトテーブル内のすべての行は、同じ詳細レベルで記録されている必要があります。上の図では、ファクトテーブル内の測定データがそれぞれの販売製品の日単位の売上合計(ドル)、販売単位、費用(ドル)になっています。粒度は日単位です。ファクトテーブル内の各レコードは、ある小売店での1日の特定の製品の売上合計を表しています。製品、店舗、または日の新たな組み合わせごとに、異なるレコードがファクトテーブル内に生成されます。

ファクトテーブルには、データソースから抽出されたデータが設定されます。データソースは、OLTP システムにすることもデータウェアハウスにすることもできます。Spectrum™ Technology Platform は、ソースデータのスナップショットを定期的を取得し、通常は毎日、毎週、または毎月、同じ時間に、データをデータウェアハウスに移します。

スタースキーマは複数のファクトテーブルを持つことができます。ディメンションテーブルの共通のサブセットを共有する一連の測定データを分離したり、測定データを異なる粒度で追跡したりするには、複数のファクトテーブルを持つスキーマを使用します。

ディメンションテーブル

ディメンションテーブルは、ファクトテーブル内の情報を記述するデータを保存します。例えば、ある月と次の月で `sales_total` が異なっていた場合、その理由を知るにはディメンションを調べることになります。同じディメンションテーブルを別々のファクトテーブルで使用できます。

ディメンションテーブルは、属性と、そのディメンションテーブルをファクトテーブルに結合する単一パートの主キーを持ちます。属性は、ディメンションテーブル内の列です。単一パートの主キーにより、1つのディメンションテーブルをすばやく参照できます。ディメンションテーブルの参照は、ファクトテーブルに対してクエリを実行する最適な方法を決定するのに役立つことがあります。

必要な日付データをレコードから容易には抽出できないことがあるため、正確な時間ベースの計算では時間ディメンションテーブルが必要になります。例えば、以下のレコードが売上データベース内にあるとします。レコード間に時間的なギャップがあることに注意してください。例えば、2012年1月4日のレコードがありません。

Date	製品	量
2012年1月3日	赤いシャツ	10.00 ドル
2012年1月5日	赤いシャツ	5.00 ドル

Date	製品	量
2012年1月7日	赤いシャツ	15.00 ドル

これらのレコードに対するクエリを実行して1日あたりの平均売上を計算した場合、結果は10.00ドル(30ドル/3レコード)になります。しかし、これは正しくありません。この3つのレコードは実際には5日間にわたって得られたものだからです。日ごとのレコードを持つ時間ディメンションテーブルがあれば、そのテーブルを上記のテーブルと結合して次のテーブルを得ることができます。

Date	製品	量
2012年1月3日	赤いシャツ	10.00 ドル
2012年1月4日		
2012年1月5日	赤いシャツ	5.00 ドル
2012年1月6日		
2012年1月7日	赤いシャツ	15.00 ドル

これらのレコードを使用して1日あたりの平均売上を計算すると、6ドル(30ドル/5日)という正しい答えが得られます。

また、休日、週末、四半期など、任意の時間属性を計算で考慮することもできます。例えば、2012年1月6日が休日で、1営業日あたりの平均売上にのみ関心があるとした場合、答えは7.50ドルになります。

結合

結合は、スタースキーマ内のファクトテーブルとディメンションテーブルとの関連性を定義します。ディメンションテーブルの主キーは、ファクトテーブルの外部キーです。ファクトテーブルには、各ディメンションテーブルの主キーの値が含まれている必要があります。外部キーから主キーへの参照は、この2つのテーブル間で値を検証するためのメカニズムです。このタイプの結合関連性により、データウェアハウスの参照整合性が保証されます。有効なクエリ結果が確実に得られるようにするには、参照整合性を維持する必要があります。

ディメンションテーブル内の各レコードは、ファクトテーブル内の多数のレコードを記述でき、ディメンションテーブルからファクトテーブルへの結合のカーディナリティは1対多になります。

上の図では、`product_key`が `product_dimension` テーブルの主キーであり、`sales_fact` テーブルの外部キーです。この結合は、会社の製品とその売上との関連性を表しています。

スター スキーマの利点

適切に設計されたスキーマを使用すると、大規模な多次元データ セットの理解、ナビゲーション、および分析が迅速に行えます。意思決定支援環境におけるスター スキーマの主な利点は、次のとおりです。

クエリ パフォーマンス

スター スキーマに対するクエリの実行は、OLTP システムの場合よりも高速です。これは、スター スキーマのほうがテーブルやクリア結合パスが少ないからです。スター スキーマ設計では、各ディメンションが中央のファクト テーブルを介してリンクされます。ディメンションどうしはファクト テーブルと交差する 1 本の結合パスによって相互にリンクされています。こうした設計上の特徴により、正確で一貫性のあるクエリ結果が確実に得られます。

ロード パフォーマンスと管理

スター スキーマ構造では、データの大規模なバッチをデータベース内にロードするのに必要な時間が短縮されます。ファクトとディメンションを定義してそれらを別々のテーブルに分離することで、ロード操作の影響が軽減されます。ディメンション テーブルは一度設定した後、ときどき更新できます。新しいファクトは、ファクト テーブルにレコードを追加することで定期的かつ選択的に追加できます。

組み込まれている参照整合性

スター スキーマは、ロードされたデータの参照整合性を強制的に適用するように設計されています。参照整合性は、主キーおよび外部キーの使用によって強制されます。ディメンション テーブル内の主キーは、ファクト テーブル内の外部キーとなり、ディメンションおよびファクト テーブルの全体にわたって各レコードがリンクされます。

効率的なデータのナビゲーション

データに対するナビゲーションは、各ディメンションがファクト テーブルによって結合されているので、効率的に行われます。こうした結合が重要なのは、実際のビジネス プロセスどうしの基本的な関連性を表しているからです。ディメンション テーブルを 1 つ閲覧するだけで、効率的なクエリを構築するための属性値を選択できます。

2 - データ ソースおよびデータ ウェアハウスへの接続

このセクションの構成

データ ソース接続	14
接続の定義	15
クラウド ファイル サーバーの圧縮のサポート	75
接続の削除	75
操作方法ビデオ - 接続の設定	76

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](#) (155ページ)
- [Read From File](#) (165ページ)
- [Read from Hadoop Sequence File](#) (183ページ)
- [Read from Hive File](#) (188ページ)
- [Read From HL7 File](#) (191ページ)
- [Read from NoSQL DB](#) (204ページ)
- [Read from SAP](#) (209ページ)
- [Read from Spreadsheet](#) (215ページ)
- [Read from Variable Format File](#) (218ページ)
- [Read From XML](#) (233ページ)

• **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ー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、ユーザー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 \(18ページ\)](#)」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。

- [接続を追加] ボタン  をクリックします。
- [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。

- [タイプ] フィールドで、**[Apache Cassandra]** を選択します。
- [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。
- [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。
- [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。
- 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。
- [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。
- [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。
- 接続をテストするには、**[テスト]** をクリックします。
- [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](#) (183ページ)、[Write to Hadoop Sequence File](#) (292ページ)、[Read From File](#) (165ページ)、[Write to File](#) (273ページ)、[Read From XML](#) (233ページ)、[Write to XML](#) (321ページ)、[Read from Hive File](#) (188ページ)、[Write to Hive File](#) (296ページ)、[Read From HL7 File](#) (191ページ) 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

MapReduce 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 (38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
```

- 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。
- [保存]** をクリックします。
- 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** (40ページ)

- 目的のデータベース用のすべての JDBC ドライバ ファイルを適切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。

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

- 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が。


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だけに、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](#) (39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。

ホーム > リソース: データソース > データソースの追加

データソースの追加

*名前

接続


*タイプ

*Host

*Port

*Instance

*encryption

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。

[ホーム](#) > [システム:ドライバ](#) > [ドライバの編集](#)

展開済みドライバの編集

*名前

com.mysql.jdbc.Driver.5.1


*JDBC ドライバクラス名 

com.mysql.jdbc.Driver

*接続文字列テンプレート 

jdbc:mysql://\${host}/\${instance}

プロパティおよびドライバ

接続プロパティ 



ラベル	プロパティトークン
<input type="checkbox"/> username	user
<input type="checkbox"/> password	password
<input type="checkbox"/> Use SSL	useSSL

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。

ホーム > リソース:データソース > データソースの追加

データソースの追加

*名前

MyConnection

接続

*タイプ

com.mysql.jdbc.Driver.5.1


*Host

*Instance

User Name

Password

Use SSL

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切なフォルダに入れます。フォルダの位置は特に重要ではありません。
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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。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用いられる、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](#) (183ページ)、[Write to Hadoop Sequence File](#) (292ページ)、[Read From File](#) (165ページ)、[Write to File](#) (273ページ)、[Read From XML](#) (233ページ)、[Write to XML](#) (321ページ)、[Read from Hive File](#) (188ページ)、[Write to Hive File](#) (296ページ)、[Read From HL7 File](#) (191ページ) 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]、[ユーザー名]、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 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時、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。

3 - データ ウェアハウスの設定

このセクションの構成

データの準備	78
時間ディメンション テーブルの設定	79
ディメンション テーブルの設定	80
ファクト テーブルの設定	82
データ ウェアハウス内のレコードへのタイム スタンプの追加	87

データの準備

データ ウェアハウスを設定する前に、データが良質であること、また意味のある洞察をビジネス ユーザーに提供するために必要なすべての属性をデータが備えていることを確認する必要があります。一般的な方法の 1 つは、オペレーショナル データ ストア (ODS) をこの目的のために使用することです。ODS は、データ ウェアハウス内にロードするためにデータを準備するデータ品質操作を実行できるデータベースです。Spectrum™ Technology Platform には、データの品質を向上したり、空間データや人口統計データなどによってデータを強化したりするために ODS 内に実装できるさまざまな機能があります。次の機能が現在ライセンスされていない場合は、Pitney Bowes まで詳細をお問い合わせください。

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。

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。

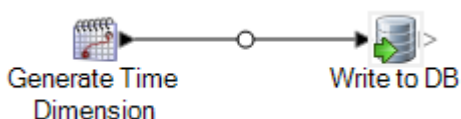
時間ディメンション テーブルの設定

時間ディメンション テーブルは、複雑な SQL 計算を使用することなく長期的なデータの分析を可能にするデータベース内のテーブルです。例えば、営業日と休日のデータ、または平日と週末のデータを対比して分析したり、会計期間ごとまたは特別なイベントごとのデータを分析したりできます。

以下の手順は、Spectrum™ Technology Platform を使用してデータ ウェアハウス内に時間ディメンション テーブルを設定する方法を記述したものです。

注：この手順を開始する前に、時間ディメンション テーブルの作成先となるデータ ウェアハウスへの接続を定義しておく必要があります。必要な接続を定義していない場合は、[データ ソース接続](#) (14ページ) を参照してください。

1. Enterprise Designer で、[ファイル] > [新規作成] > [データフロー] > [ジョブ] を選択します。
2. [Generate Time Dimension] ステージをキャンバス上にドラッグします。
3. [Write to DB] ステージをキャンバス上にドラッグし、[Generate Time Dimension] ステージをこのステージに接続します。
すると、データフローは次のようになるはずで



4. [Generate Time Dimension] ステージをダブルクリックし、必要な時間ディメンションを作成するようにこのステージを設定します。詳細については、「[Generate Time Dimension \(127ページ\)](#)」を参照してください。

注：粒度が日またはそれより大きい場合は、通常、ユリウス通日が時間ディメンションテーブルの主キーとして使用されます。粒度が日より小さい場合は、データフローに Unique ID Generator ステージを追加することで別のキーを生成できます。ユリウス通日をキーとして使用する場合は、ユリウス通日の値のために整数列と、日付の値のためにデータ タイプが日付または日時の列を作成するように Generate Time Dimension を設定します。

5. キャンバス上の [Write to DB] ステージをダブルクリックし、ディメンション テーブルを作成するデータベースおよびテーブルを指すようにこのステージを設定します。Write to DB の設定については、[Write to DB \(267ページ\)](#) を参照してください。
6. 時間ディメンションテーブルに書き込む前に、時間ディメンション値をプレビューするには:
 - a) [Generate Time Dimension] ステージおよび [Write to DB] ステージに接続しているチャンネルを右クリックし、[\[インスペクション ポイントの追加\]](#) を選択します。
 - b) [\[実行\] > \[現在のフローのインスペクション\]](#) を選択します。
Enterprise Designer ウィンドウの下部にインスペクション ペインが現れ、時間ディメンション テーブルに書き込まれるデータが表示されます。必要であれば、Generate Time Dimension ステージに対する調整を行ったうえで、インスペクション プロセスを再実行して変更の影響を確認できます。
7. データフローに問題がなければ、[\[実行\] > \[現在のフローの実行\]](#) を選択して、データフローの実行と時間ディメンション テーブルの設定を行います。

ディメンション テーブルの設定

ディメンション テーブルは、スター スキーマの一部であり、ファクト テーブル内の列に関する詳細な情報を格納します。ディメンション テーブルは、属性と、そのディメンション テーブルをファクト テーブルに結合する単一部分の主キーを持ちます。単一部分の主キーにより、1つのディメンション テーブルをすばやく参照できます。ディメンション テーブルの参照は、ファクト テーブルに対してクエリを実行する最適な方法を決定するのに役立つことがあります。

以下の手順は、Spectrum™ Technology Platform を使用して、データ ウェアハウス内にあるディメンション テーブルの初期設定を実行する方法を記述したものです。

注：データベース、ファイル サーバ、または Web サービスをデータのソースとして使用する場合は、この手順を開始する前に、ディメンション テーブルのソースとして使用する外部リソースへの接続を定義しておく必要があります。また、ディメンション テーブルを作成するデータ ウェアハウスへの接続を定義する必要もあります。必要な接続を定義していない場合は、[データ ソース接続](#) (14ページ) を参照してください。

1. データ ウェアハウスで、ディメンション テーブルとして使用するテーブルを作成します。
2. Management Console で、データ ソースおよびデータ ウェアハウスへの接続を作成します。
 - データベースへの接続については、[JDBC データベースへの接続](#) (38ページ) を参照してください。
 - ファイル サーバへの接続については、[FTP サーバへの接続](#) (27ページ) を参照してください。
3. Enterprise Designer で、**[ファイル]** > **[新規作成]** > **[データフロー]** > **[ジョブ]** を選択します。
4. ソース ステージをキャンバス上にドラッグします。
 - データベースのデータを使用してテーブルを設定するには、**[Read from DB]** ステージをキャンバス上にドラッグします。
 - フラット ファイルのデータを使用してテーブルを設定するには、**[Read from File]** ステージをキャンバス上にドラッグします。
 - 可変形式ファイルのデータを使用してテーブルを設定するには、**[Read from Variable Format File]** ステージをキャンバス上にドラッグします。
 - XML ファイルのデータを使用してテーブルを設定するには、**[Read from XML]** ステージをキャンバス上にドラッグします。
5. キャンバス上に配置したソース ステージをダブルクリックし、ディメンション テーブルに対して設定するデータのソースを指すようにそのステージを設定します。
 - Read from DB の設定については、[Read From DB](#) (155ページ) を参照してください。
 - Read from File の設定については、[Read From File](#) (165ページ) を参照してください。
 - Read from Variable Format File の設定については、[Read from Variable Format File](#) (218ページ) を参照してください。
 - Read from XML の設定については、[Read From XML](#) (233ページ) を参照してください。
6. **[Unique ID Generator]** ステージをキャンバス上にドラッグし、ソース ステージをこのステージに接続します。例えば、Read from DB をソース ステージとして使用する場合は、Read from DB を Unique ID Generator に接続します。
7. キャンバス上の **[Unique ID Generator]** ステージをダブルクリックし、サロゲート キーを作成するようにこのステージを設定します。

注：通常、運用システムのキーは、ウェアハウス内にあるディメンション テーブルの主キーとしては使用されません。このことは、時間が経過しても一貫性を維持するのに役立ちます。運用システムではキーの値が変化する可能性があるからです。

8. [Write to DB] ステージをキャンバス上にドラッグし、Unique ID Generator をこのステージに接続します。
9. キャンバス上の [Write to DB] ステージをダブルクリックし、設定するデータベースおよびディメンション テーブルを指すようにこのステージを設定します。Write to DB の設定については、[Write to DB](#) (267ページ) を参照してください。
10. [ファイル] > [保存] を選択し、データフローを保存します。
11. 今すぐデータフローを実行してディメンション テーブルを設定するには、[実行] > [現在のフローを実行] を選択します。

ファクト テーブルの設定

データ ウェアハウスでディメンション テーブルの設定が完了したら、いつでもファクト テーブルを設定できます。OLTP データベース内のテーブルにある数値的な測定データをファクト テーブルに設定します。

重要：ファクト テーブルの設定前に、ディメンション テーブルを設定する必要があります。

以下の手順は、Spectrum™ Technology Platform を使用してデータ ウェアハウス内にファクト テーブルを設定する方法を記述したものです。この手順では、ソース データベース内のテーブルからソース データを読み込むデータフローを作成し、ソース テーブルのナチュラル キーをディメンション テーブルのサロゲート キーで置き換え、サロゲート キーを含む更新されたレコードとソース テーブルのファクト データをファクト テーブル内にロードします。

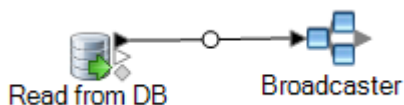
1. Management Console で、データ ソースおよびデータ ウェアハウスへの接続を作成します。
 - データベースへの接続については、[JDBC データベースへの接続](#) (38ページ) を参照してください。
 - ファイル サーバへの接続については、[FTP サーバーへの接続](#) (27ページ) を参照してください。
2. Enterprise Designer で、[ファイル] > [新規作成] > [データフロー] > [ジョブ] を選択します。
3. ファクト テーブルに書き込むデータのソースに基づいて、適切なステージをキャンバス上にドラッグします。
 - データベースのデータを使用してテーブルを設定するには、[Read from DB] ステージをキャンバス上にドラッグします。

- フラット ファイルのデータを使用してテーブルを設定するには、**[Read from File]** ステージをキャンバス上にドラッグします。
 - 可変形式ファイルのデータを使用してテーブルを設定するには、**[Read from Variable Format File]** ステージをキャンバス上にドラッグします。
 - XML ファイルのデータを使用してテーブルを設定するには、**[Read from XML]** ステージをキャンバス上にドラッグします。
4. キャンバス上に配置したソース ステージをダブルクリックし、ファクト テーブルに対して設定するデータのソースを指すようにそのステージを設定します。
- Read from DB の設定については、[Read From DB](#) (155ページ) を参照してください。
 - Read from File の設定については、[Read From File](#) (165ページ) を参照してください。
 - Read from Variable Format File の設定については、[Read from Variable Format File](#) (218ページ) を参照してください。
 - Read from XML の設定については、[Read From XML](#) (233ページ) を参照してください。

注：通常、ファクト テーブルを設定するデータフローでは、ファイルではなく、データベースからデータを読み込みます。これは非常に一般的なシナリオなので、この手順の残り部分の例では、[Read from DB](#) を使用します。

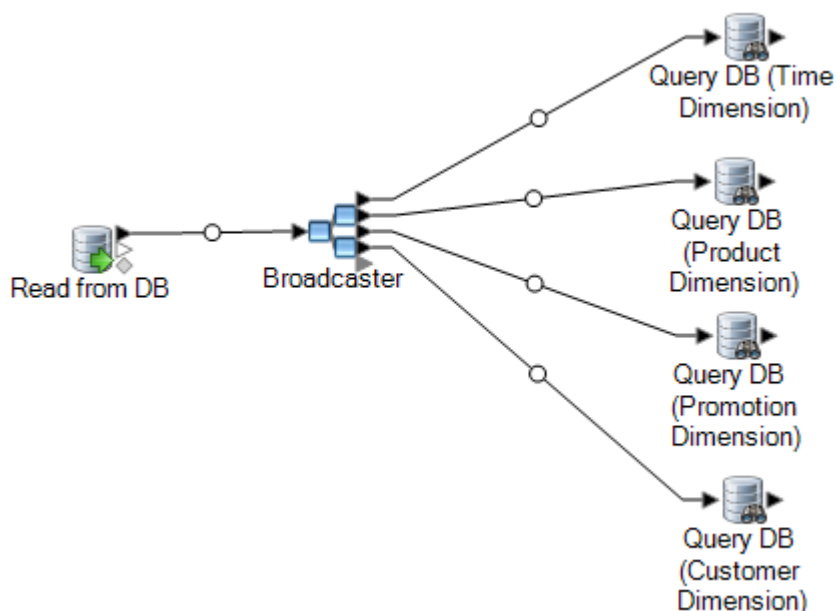
5. **[Broadcaster]** ステージをキャンバス上にドラッグし、ソース ステージをこのステージに接続します。

データフローは次のようになります。



6. データ ウェアハウス内のディメンション テーブルごとに **[Query DB]** ステージを 1 つキャンバス上にドラッグし、それらのステージを **[Broadcaster]** ステージに接続します。

例えば、データ ウェアハウス内にディメンション テーブルが 4 つある場合は、4 つの **Query DB** ステージをキャンバス上にドラッグします。データフローは次のようになります。



Query DB ステージは、データソースのナチュラルキーを使用して各ディメンションのサロゲートキーを検索するために使用されます。その後、ファクトテーブル内にロードされる各レコードのナチュラルキーがサロゲートキーで置き換えられます。

ヒント：ステージの名前は、各ステージによるクエリの実行対象テーブルがわかりやすくなるように変更できます。

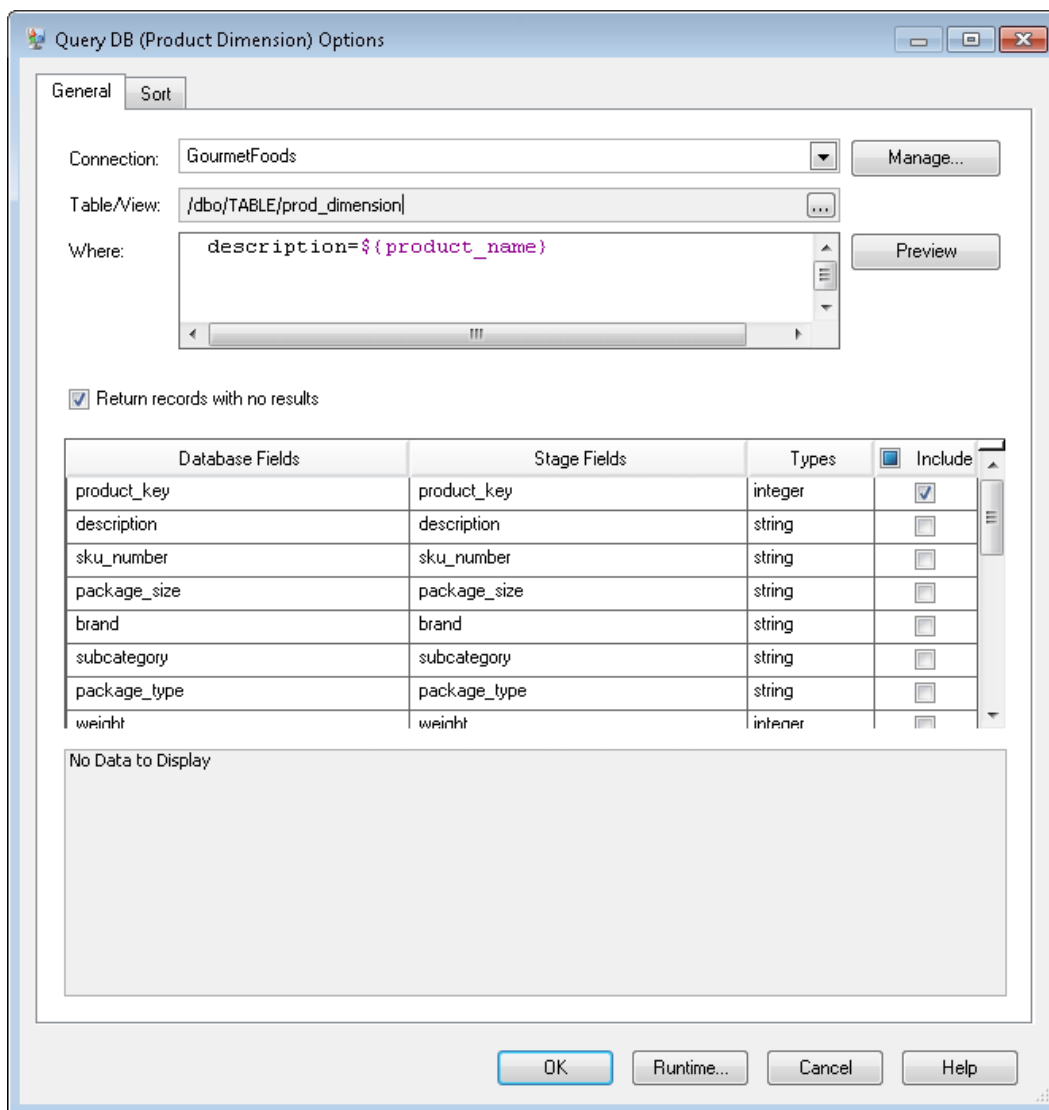
7. データソースのナチュラルキーのそれぞれについてサロゲートキーの検索を行うように各 Query DB ステージを設定します。これを行うには、次の手順を実行します。
 - a) **[接続]** フィールドで、データウェアハウスへの接続を指定します。
 - b) **[テーブル/ビュー]** フィールドで、このステージによってクエリを実行するディメンションテーブルを選択します。
 - c) **[WHERE]** フィールドに、適切なデータフローフィールドの値に基づいてサロゲートキーを検索する WHERE 文を記述します。

例えば、次の例は `description` 列の値がデータソースの `product_name` フィールドの値に一致するディメンションテーブル内のレコードを探すことにより、製品のサロゲートキーを検索します。

```
description=${product_name}
```

- d) **[含める]** 列で、サロゲートキーを含むデータベース列を選択します。

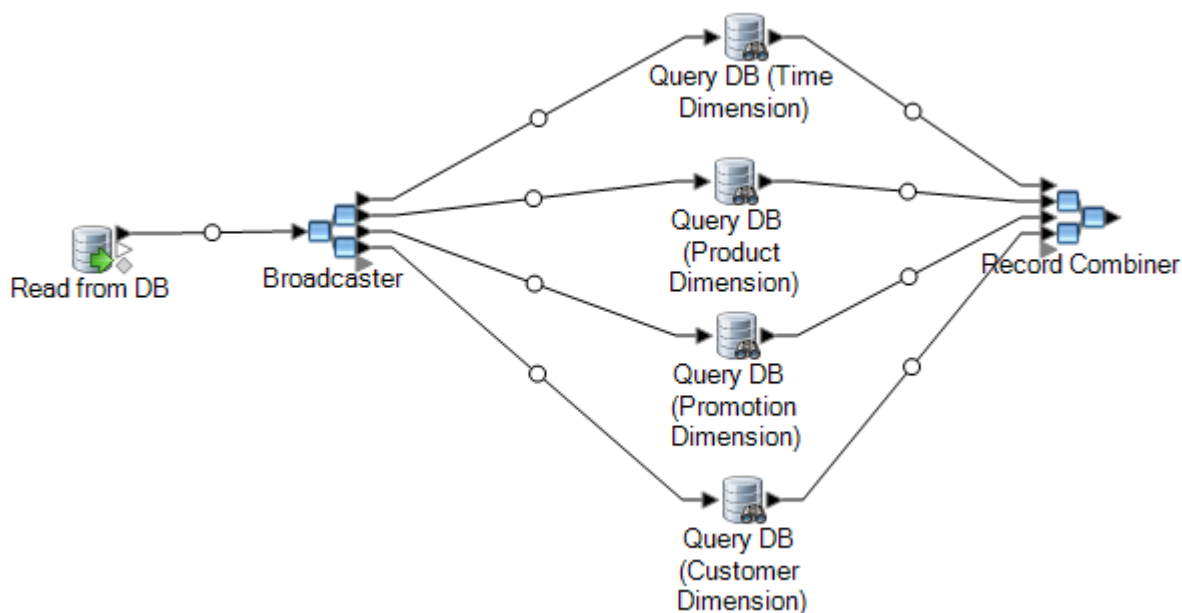
例えば、製品名のサロゲートキーを検索する Query DB ステージは次のようになります。



この例のクエリは、prod_dimension テーブル内に、description 列の値が product_name データフローフィールドの値に一致するレコードを探すことにより、製品キーを検索します。このステージは、product_key フィールドを返し、このフィールドをデータフローに追加します。これは **含む** 列のボックスにチェックが入っていることからわかります。

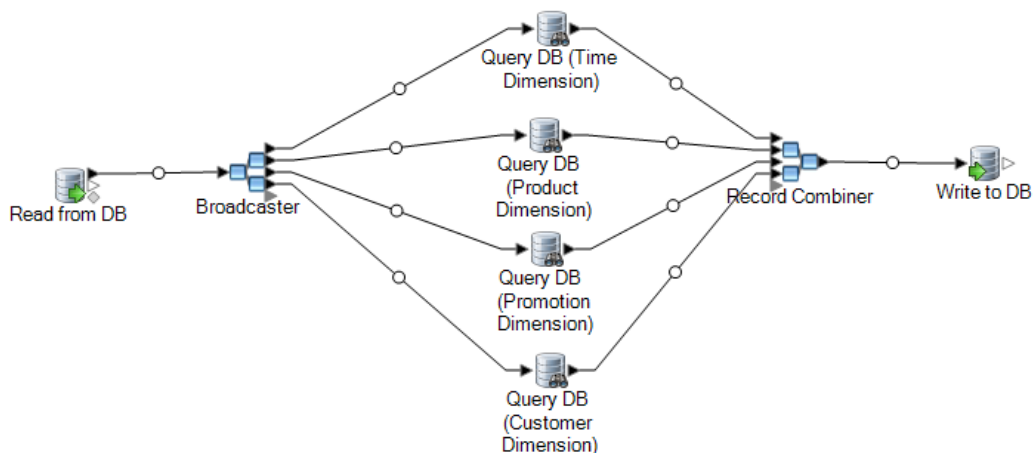
8. [Record Combiner] ステージをキャンバス上にドラッグし、すべての [Query DB] ステージをこのステージに接続します。

すると、データフローは次のようになるはずです。



9. [Write to DB] ステージをキャンバス上にドラッグし、このステージを [Record Combiner] ステージに接続します。

データフローは次のようになるはずです。



10. レコードをファクト テーブルに書き込むように Write to DB ステージを設定します。これを行うには、次の手順を実行します。
- [接続]** フィールドで、データ ウェアハウスへの接続を指定します。
 - [テーブル/ビュー]** フィールドで、このステージによってクエリを実行するファクト テーブルを選択します。ファクト テーブルがまだデータウェアハウス内にはない場合は、**[テーブルの作成]** をクリックしてデータ ウェアハウスにファクト テーブルを作成します。
 - ファクト テーブルに書き込むフィールドのそれぞれについて、**[含める]** 列のボックスにチェックを入れます。

- d) **[実行時]** タブでは、デフォルトで **[挿入]** オプションが書き込みモードに対して選択されています。通常、ファクトテーブルの設定は挿入モードで実行されるので、このオプションは選択されたままにしておいて構いません。

11. データフローを保存して実行します。

ソース データをディメンション テーブルのキーで置換する例

次のレコードを考えます。

```
March 28 2013,Parsley Garlic Pasta,Mile High Gourmet Market,78.35
```

この例では、日付フィールドの後に、製品名 (Parsley Garlic Pasta)、顧客 (Mile High Gourmet Market)、数量 (78.25) が続いています。データ ウェアハウスには、日付、製品名、および顧客の各ディメンション テーブルがあります。そのため、レコード内のナチュラル キーは各ディメンション テーブルのサロゲート キーで置き換える必要があります。これを実現するために、データフローは 3 つの Query DB ステージを持つこととなります。それぞれ、日付、製品名、および顧客のサロゲート キーを検索するステージです。

各 Query DB には、サロゲート キーを検索する WHERE 文があります。

これらの検索の結果、ファクト テーブルへの書き込みが行われたときのレコードは次のようになります。

```
711,1,15,78.35
```

日付、製品名、および顧客のナチュラル キーがサロゲート キーで置き換えられていることに注意してください。

データ ウェアハウス内のレコードへのタイム スタンプの追加

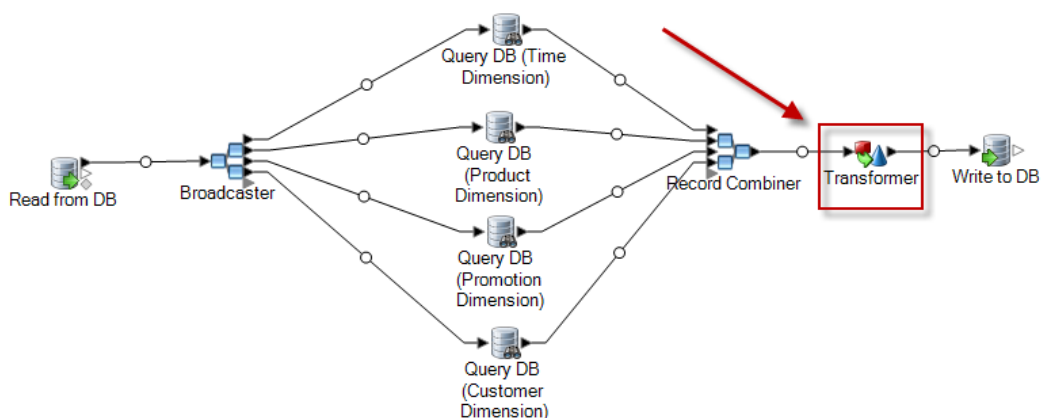
データの品質を保証する便利な方法が、データ ウェアハウス内のレコードがロードされた日付に基づいてそれらのレコードにフラグを設定することです。ロード処理が完了していない場合や、データのロード後に問題に気付いた場合は、タイム スタンプ列により、どのレコードが影響を受けているかを容易に識別できます。その後、特定のフェーズで処理されたすべてのレコードを削除し、ロード前の状態に戻して、問題に対処したうえでデータのロードを再び試みることができ

ます。SQL Command ステージを使用してファクト テーブルに `load_date` のような追加の列を設けることで、ロード操作にタイム スタンプを付与できます。

データ ウェアハウスの設定または更新時にデータフローによってタイム スタンプを追加するには:

1. Enterprise Designer で、データ ウェアハウスを設定または更新するデータフローを次のようにして開きます。
2. [Transformer] ステージをキャンバス上にドラッグし、このステージをデータフローの [Write to DB] ステージのすぐ前に接続します。

例:



3. [Transformer] ステージをダブルクリックします。
4. [追加] をクリックします。
5. [全般] の下で、[カスタム] を選択します。
6. [Custom Transform 名] フィールドに、このトランスフォームの名前を入力します。任意の名前が使用できます。例えば、"Add Time Stamp" (タイム スタンプの追加) とします。
7. [カスタム スクリプト] フィールドには、次のように入力します。

```
data['<timestamp field>']=currentDateTime()
```

ここで、<timestamp field> はタイム スタンプを格納するデータフロー フィールドの名前です。

例えば、Timestamp という名前のデータフロー フィールドにタイム スタンプを設定する場合、カスタム スクリプトは次のようになります。

```
data['Timestamp']=currentDateTime()
```

8. ウィンドウの下部にある [追加] ボタンをクリックします。
9. [閉じる] をクリックします。

10. **[OK]** をクリックして **[Transformer オプション]** ウィンドウを閉じます。

これで、データフローによって現在の時刻が各レコード内のフィールドに追加され、各レコードのロード時点を示すタイムスタンプがデータウェアハウス内で実現されます。

4 - データ ウェアハウスの更新

このセクションの構成

データ ウェアハウスの更新スケジュールの定義	91
ファクト テーブルの更新	92
クエリのためのグローバル キャッシュの使用	97
クエリのためのローカル キャッシュの使用	99

データ ウェアハウスの更新スケジュールの定義

Spectrum™ Technology Platform はデータフローのスケジュールを設定して、データソース内の正規化構造からデータを抽出したり、データウェアハウス内のスタースキーマ構造に変換したりできます。ほとんどのロード操作は営業日には使用できないシステムリソースを必要とするので、データフローのスケジューリングは有用です。

更新スケジュールを決定する際には、次の点を考慮します。

- 頻度
- シーケンス
- 依存関係

頻度

きわめて詳細なファクトテーブルの粒度に基づいて実行するデータフローをスケジュールする必要があります。例:

- ファクトテーブルの粒度が日単位の場合は、ファクトテーブルの設定データフローを毎日実行するようにスケジュールします。
- 粒度が月単位の場合、ユーザは過去何か月かのデータのみを使用して作業するので、設定データフローをすぐに実行するのではなく、毎月実行するようにスケジュールします。

ほとんどの設定データフローは大量のデータを処理するので、Spectrum™ Technology Platform サーバ、ソースデータベース、データウェアハウスデータベース、およびネットワークの使用量が最小のときに設定データフローを実行するようにスケジュールします。

初期ロード時にすべてのディメンションテーブルとファクトテーブルを設定します。初期ロードの後、追加または変更された内容に基づいてテーブルを更新します。通常、ファクトテーブルはディメンションテーブルよりも頻繁に更新されます。その理由は次のとおりです。

- 通常、ディメンションテーブルはソース内の属性が変更または追加されない限り、静的です。
- 一般的に、意思決定支援データベース内のファクトテーブルデータは長期的なものであり、最新の状態を維持するために定期的な追加や更新を必要とします。初期ロードやほとんどの増分ロードはファクトテーブルに影響を与えます。

シーケンス

データウェアハウスデータベース内のデータ間には依存関係があるので、実行スケジュールの設定前に設定データフローを実行する順序を決定します。

ファクト テーブルの前にディメンション テーブルを設定します。関連ファクト テーブルを設定する前に、すべてのディメンション レコードおよびキーが存在している必要があるからです。この制限は、スター スキーマにおけるディメンション テーブルとファクト テーブルとの間の主キーと外部キーの関連性によるものです。

意思決定支援データベース内の集計テーブルを設定する前に、ベースレベルのテーブルを更新します。この順序により、ベースレベル テーブルと集計テーブルの同期が確実に維持されます。

設定データフローを実行する適切な順序は、次のとおりです。

1. ベースレベル ディメンション テーブルの計画
2. ベースレベル ファクト テーブルの計画
3. 集計ディメンション テーブルの計画
4. 集計ファクト テーブルの計画

依存関係

複数の設定データフローを特定の順序で実行する必要がある場合や、データフローの実行にかかる合計時間が予測できない場合は、データフロー依存関係を作成できます。データフローは、前のデータフローの完了、前のデータフローでのエラー発生など、特定の要件が満たされた場合のみ実行されます。

データフロー依存関係を作成するには、Enterprise Designer でプロセス フローを作成します。プロセス フローの詳細については、『*Dataflow Designer ガイド*』を参照してください。

ファクト テーブルの更新

この手順は、ソース データベースまたはソース ファイルからデータを読み込んでそのデータを使用してデータ ウェアハウス内のファクト テーブルを更新するデータフローの作成方法を記述したものです。

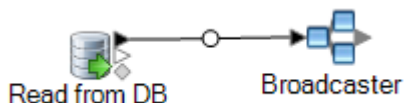
1. Enterprise Designer で、[ファイル] > [新規作成] > [データフロー] > [ジョブ] を選択します。
2. ファクト テーブルに書き込むデータのソースに基づいて、適切なステージをキャンバス上にドラッグします。
 - データベースのデータを使用してテーブルを設定するには、[Read from DB] ステージをキャンバス上にドラッグします。
 - フラット ファイルのデータを使用してテーブルを設定するには、[Read from File] ステージをキャンバス上にドラッグします。
 - 可変形式ファイルのデータを使用してテーブルを設定するには、[Read from Variable Format File] ステージをキャンバス上にドラッグします。

- XML ファイルのデータを使用してテーブルを設定するには、**[Read from XML]** ステージをキャンバス上にドラッグします。

注：データベースではなくファイルからデータを読み込む場合は、ファクト テーブルに追加する新しいレコードのみがそのファイルに含まれていて、ファクト テーブルに既に存在するレコードは含まれていないことを確認します。データベースからデータを読み込む場合は、レコードをフィルタリングするためのクエリをこの手順の後のステップで定義することになります。

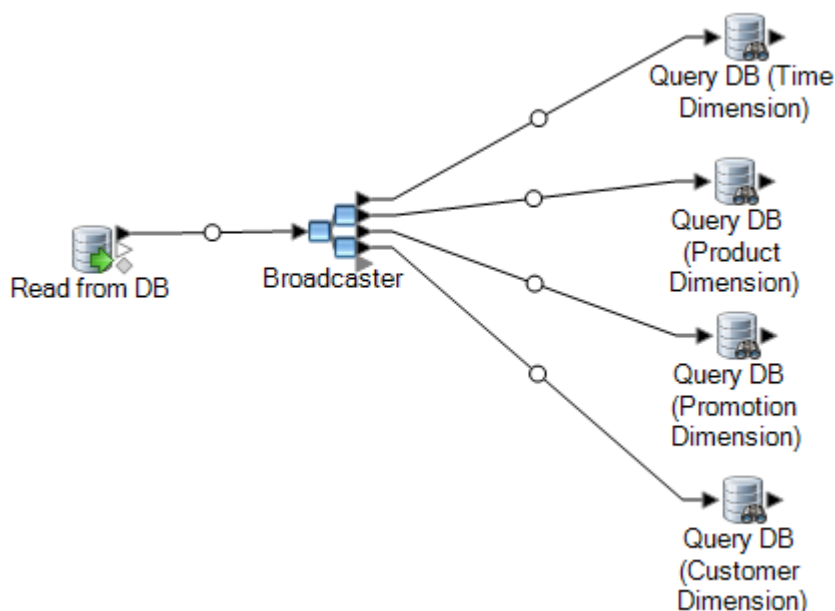
3. キャンバス上に配置したソース ステージをダブルクリックし、ファクト テーブルに対して設定するデータのソースを指すようにそのステージを設定します。
 - Read from DB の設定については、[Read From DB](#)（155ページ）を参照してください。
 - Read from File の設定については、[Read From File](#)（165ページ）を参照してください。
 - Read from Variable Format File の設定については、[Read from Variable Format File](#)（218ページ）を参照してください。
 - Read from XML の設定については、[Read From XML](#)（233ページ）を参照してください。
4. データベースからデータを読み込む場合は、新しいレコードのみがファクト テーブルに追加されるように、レコードをフィルタリングします。この操作は、ファクト テーブルの最後の更新以降に変更されたレコードのみを読み込むための **SQL SELECT** ステートメントを定義することで実行できます。
5. **[Broadcaster]** ステージをキャンバス上にドラッグし、ソース ステージをこのステージに接続します。

データフローは次のようになります。



6. データ ウェアハウス内のディメンション テーブルごとに **[Query DB]** ステージを 1 つキャンバス上にドラッグし、それらのステージを **[Broadcaster]** ステージに接続します。

例えば、データ ウェアハウス内にディメンション テーブルが 4 つある場合は、4 つの Query DB ステージをキャンバス上にドラッグします。データフローは次のようになります。



Query DB ステージは、データソースのナチュラルキーを使用して各ディメンションのサロゲートキーを検索するために使用されます。その後、ファクトテーブル内にロードされる各レコードのナチュラルキーがサロゲートキーで置き換えられます。

ヒント：ステージの名前は、各ステージによるクエリの実行対象テーブルがわかりやすくなるように変更できます。

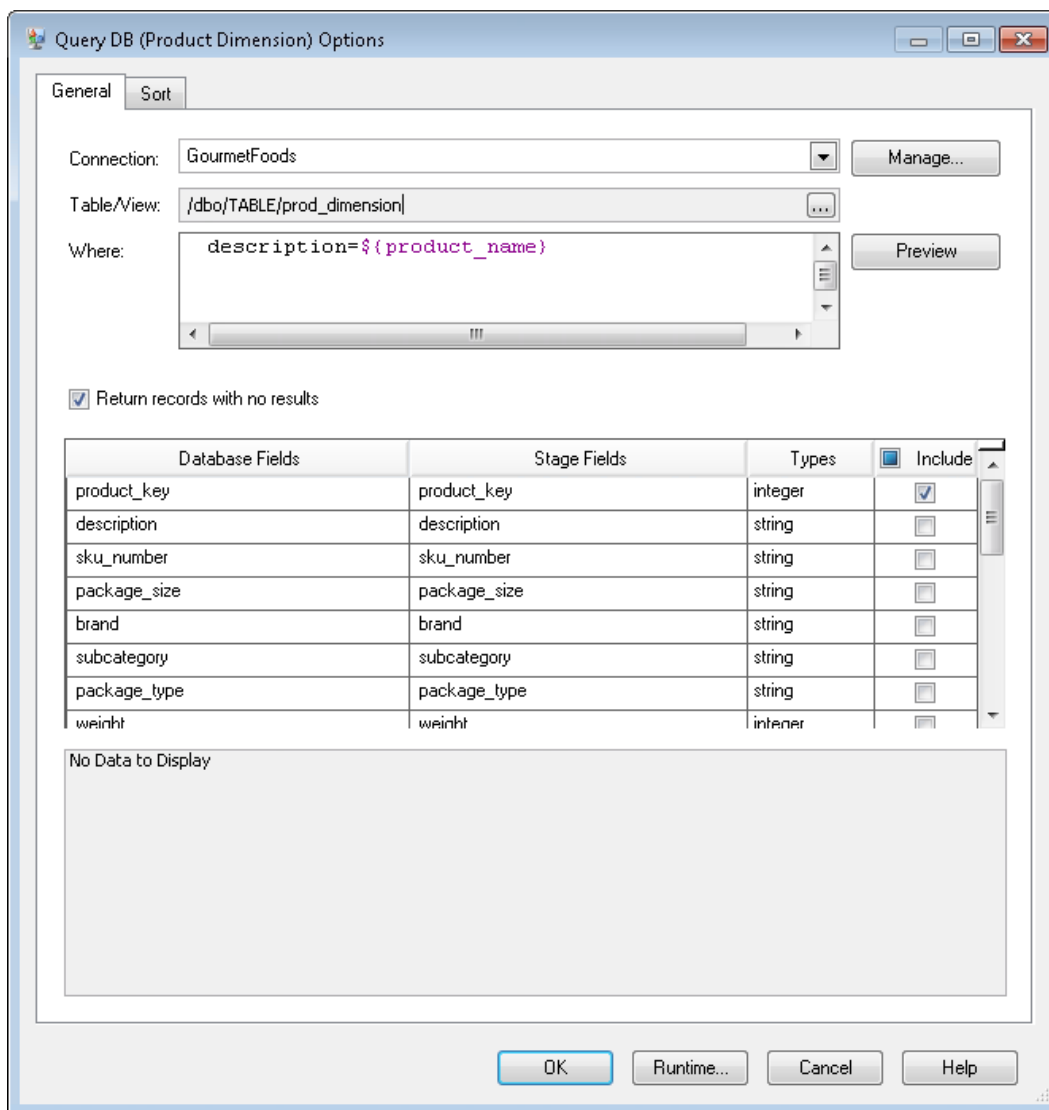
7. データソースのナチュラルキーのそれぞれについてサロゲートキーの検索を行うように各 Query DB ステージを設定します。これを行うには、次の手順を実行します。
 - a) **[接続]** フィールドで、データウェアハウスへの接続を指定します。
 - b) **[テーブル/ビュー]** フィールドで、このステージによってクエリを実行するディメンションテーブルを選択します。
 - c) **[WHERE]** フィールドに、適切なデータフローフィールドの値に基づいてサロゲートキーを検索する WHERE 文を記述します。

例えば、次の例は `description` 列の値がデータソースの `product_name` フィールドの値に一致するディメンションテーブル内のレコードを探すことにより、製品のサロゲートキーを検索します。

```
description=${product_name}
```

- d) **[含める]** 列で、サロゲートキーを含むデータベース列を選択します。

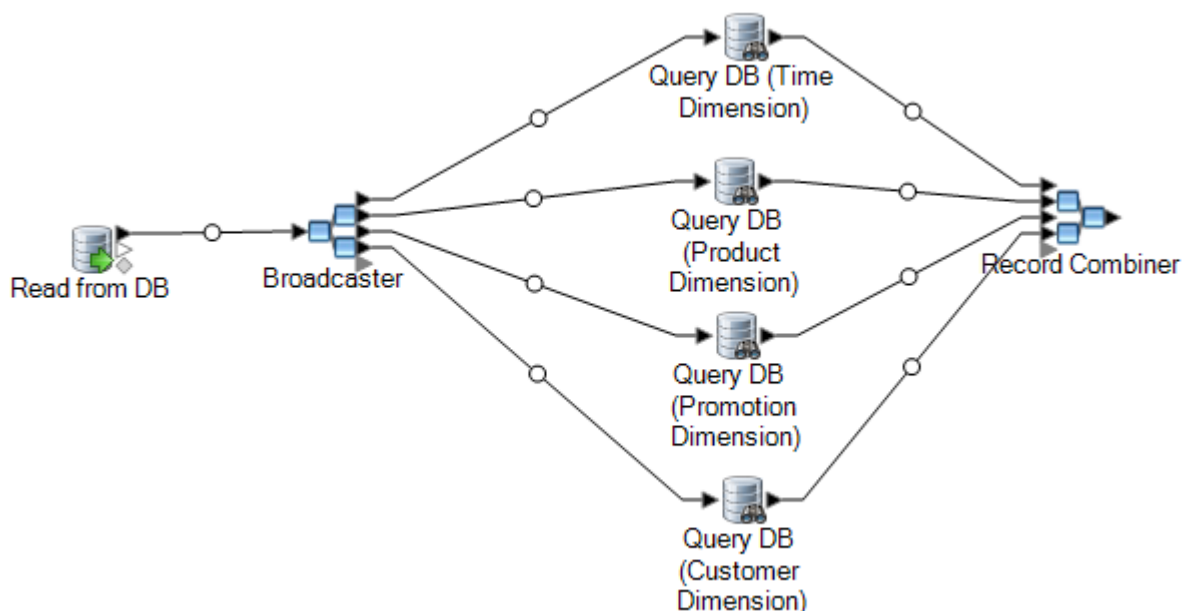
例えば、製品名のサロゲートキーを検索する Query DB ステージは次のようになります。



この例のクエリは、prod_dimension テーブル内に、description 列の値が product_name データフローフィールドの値に一致するレコードを探すことにより、製品キーを検索します。このステージは、product_key フィールドを返し、このフィールドをデータフローに追加します。これは **含む** 列のボックスにチェックが入っていることからわかります。

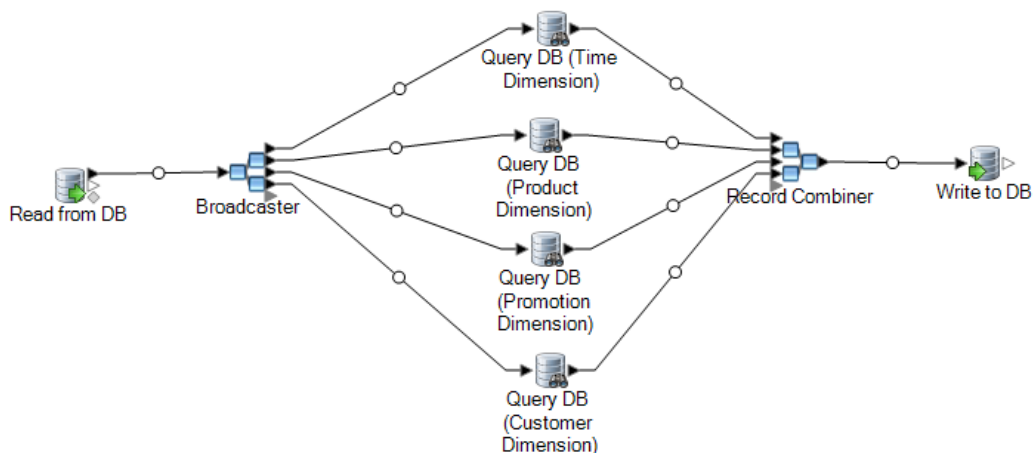
8. [Record Combiner] ステージをキャンバス上にドラッグし、すべての [Query DB] ステージをこのステージに接続します。

すると、データフローは次のようになるはずです。



9. [Write to DB] ステージをキャンバス上にドラッグし、このステージを [Record Combiner] ステージに接続します。

データフローは次のようになるはずです。



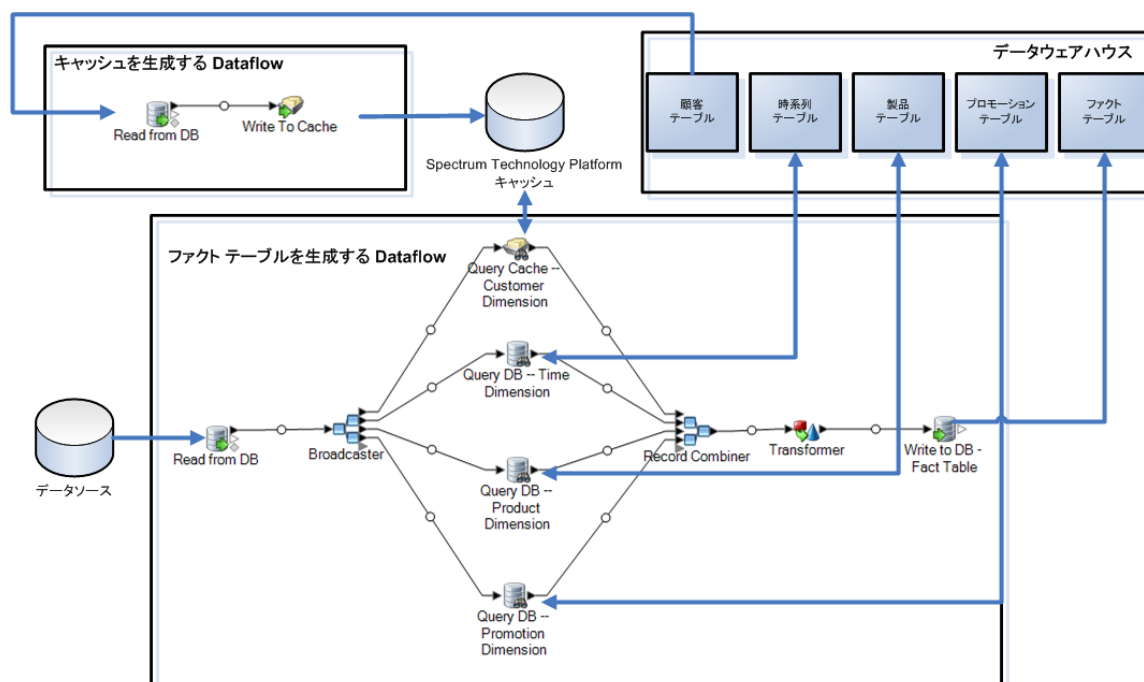
10. レコードをファクト テーブルに書き込むように Write to DB ステージを設定します。これを行うには、次の手順を実行します。
- [接続]** フィールドで、データ ウェアハウスへの接続を指定します。
 - [テーブル/ビュー]** フィールドで、このステージによってクエリを実行するファクト テーブルを選択します。ファクト テーブルがまだデータウェアハウス内にはない場合は、**[テーブルの作成]** をクリックしてデータ ウェアハウスにファクト テーブルを作成します。
 - ファクト テーブルに書き込むフィールドのそれぞれについて、**[含める]** 列のボックスにチェックを入れます。

- d) [実行時] タブでは、デフォルトで [挿入] オプションが書き込みモードに対して選択されています。通常、ファクトテーブルの設定は挿入モードで実行されるので、このオプションは選択されたままにしておいて構いません。

クエリのためのグローバル キャッシュの使用

大きなディメンションテーブルが存在する場合は、ディメンションテーブルのデータをキャッシュにロードし、そのキャッシュを使用してサロゲートキーを検索できます。キャッシュを使用すると、Query DB によってディメンションテーブルの検索を直接実行する場合よりもパフォーマンスが向上します。

キャッシュを使用するには、2つのデータフローを作成する必要があります。キャッシュにディメンションテーブルのデータを設定するデータフローと、ファクトテーブルの更新時にキャッシュを使用するデータフローです。次の図は、この2つのデータフローがどのように連携するかを示しています。



1. 大きなディメンションテーブルのディメンションテーブルデータをキャッシュに設定するデータフローを作成します。

このデータフローは、次の2つのステージで構成する必要があります。

- キャッシュにロードするディメンションテーブルのデータを読み込む Read from DB ステージ。
 - ディメンション テーブルのデータをキャッシュに設定する Write to Cache ステージ。
2. キャッシュを設定するには、このデータフローを実行します。
 3. ファクト テーブルを設定するデータフローに、Query Cache を追加します。
 4. この Query Cache ステージで、Write to Cache ステージによって作成されたキャッシュに対してクエリを実行するためのステージを設定します。
 5. ファクト テーブルを設定するには、このデータフローを実行します。


ファクトテーブルの更新のたびにキャッシュにディメンションテーブルの最新のデータが設定されているようにしたい場合は、ディメンション テーブルを設定するジョブをまず実行してからファクトテーブルを更新するジョブを実行する、プロセスフローを作成できます。そうすれば、双方のデータフローを続けて実行するために、このプロセスフローを実行できます。プロセスフローの詳細については、『*Dataflow Designer* ガイド』を参照してください。

キャッシュの削除

グローバル キャッシュは、データ検索のために Query Cache ステージによって使用されます。キャッシュがなくなっただけの場合には、以下の手順に従ってキャッシュを削除できます。キャッシュを削除するとメモリが解放されます。

注: **[キャッシュの削除]** は、グローバル キャッシュを Management Console から削除する場合にのみ使用されます。

警告: キャッシュを削除する前に、そのキャッシュが Query Cache ステージを含むどのデータフローでも使用されていないことを確認します。データフローで使用されているキャッシュを削除すると、そのデータフローでエラーが発生します。

1. Management Console を開きます。
2. **[リソース]** > **[キャッシュ管理]** を展開します。
グリッドに既存のグローバル キャッシュが表示されます。
3. 削除するキャッシュを選択します。
4.  をクリックします。

クエリのためのローカル キャッシュの使用

大きなディメンション テーブルが存在する場合は、ディメンション テーブルのデータをキャッシュにロードし、そのキャッシュを使用してサロゲート キーを検索できます。キャッシュを使用すると、Query DB によってディメンション テーブルの検索を直接実行する場合よりもパフォーマンスが向上します。

ローカル キャッシュとは、Query Cache ステージの実行時にのみ使用される一時的なキャッシュです。さらに、キー フィールドと検索条件に基づいてキャッシュ内のデータを検索し、キャッシュ内のマッチングレコードからデータを返して、そのキャッシュレコードのデータをデータフロー内のレコードに追加します。1つのデータフローでのみ使用される場合や、検索テーブルが頻繁に変化する場合は、グローバル キャッシュの代わりにローカル キャッシュを使用します。

ローカル キャッシュをクエリに使用するには、以下の手順に従います。

1. Enterprise Designer で、キャッシュを使用してクエリを実行するデータフローを開きます。
2. [Query Cache] ステージをキャンバス上にドラッグし、このステージをデータフローに接続します。
3. [Query Cache] ステージをダブルクリックします。
4. [ローカル キャッシュ] を選択します。
5. 使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。

データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。

接続名	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。
データベース ドライバ	適切なデータベース タイプを選択します。
接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。

6. クエリの実行対象とするデータベース内のテーブルまたはビューを指定します。
7. [キー フィールド] オプションの下でキーを選択します。
8. [入力フィールド] オプションの下で入力フィールドを選択します。[入力フィールド]には前のステージから得られたフィールドが格納されます。このフィールド内の値がデータベースの[キー フィールド]に一致している場合、クエリはデータベース内にあるそのレコードからデータを返します。

9. **[OK]** をクリックします。
10. データフローを実行します。

5 - ステージ リファレンス

このセクションの構成

Call Stored Procedure	103
DB Change Data Reader	107
DB Loader	110
Field Parser	121
Field Combiner	124
Field Selector	126
Generate Time Dimension	127
Query Cache	134
Query DB	148
Query NoSQL DB	152
Read From DB	155
Read From File	165
Read from Hadoop Sequence File	183
Read from Hive File	188
Read From HL7 File	191
Read from NoSQL DB	204
Read from SAP	209
Read from Spreadsheet	215
Read from Variable Format File	218
Read From XML	233
SQL Command	241
Transposer	252
Unique ID Generator	255
Write to Cache	265
Write to DB	267
Write to File	273
Write to Hadoop Sequence File	292
Write to Hive File	296
Write to NoSQL DB	303

Write to Spreadsheet	307
Write to Variable Format File	310
Write to XML	321
日付および数字パターン	331

Call Stored Procedure

ストアド プロシージャの呼び出しは、データベース内のストアド プロシージャを実行し、**Call Stored Procedure** の結果をデータフローの入力として返すソース ステージです。テーブルまたはビューに対するクエリではなく、データベースのストアド プロシージャを使用してデータベースからデータを取得する場合は、**Call Stored Procedure** を使用します。

注： テーブルまたはビューからデータフローに直接データを読み込む場合は、**Read from DB** ステージを使用します。

ストアド プロシージャにビジネス ロジックが埋め込まれていてそのロジックを **Spectrum™ Technology Platform** 環境で使用する場合は、ストアド プロシージャの呼び出しを使用してデータをデータフロー内に読み込むことができます。例えば、多くの運用システムでは、持続的に更新される大きなテーブルに対してデータベース内の参照整合性チェックを行うとパフォーマンスが低下するので、そうしたチェックを使用していません。そこで、参照整合性を維持するために、ストアド プロシージャを作成し、システムに対するすべての更新でそうしたプロシージャを使用することができます。

ストアド プロシージャは、**Spectrum™ Technology Platform** 環境の管理を簡素化するためにも使用できます。例えば、同じデータを読み込む ETL プロセスが何百もある場合は、1つの場所に収まるようにそのクエリをストアド プロシージャ内に配置できます。そうすると、何百もの異なるプロセスではなく、1つのストアド プロシージャを変更するだけで済むので、管理が容易になります。

オプション名

説明

接続

使用するデータベース接続を選択します。使用できるデータベース接続は、**Management Console** の **Connection Manager** に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、**【管理】** をクリックします。

注: このオプションは **Enterprise Designer** によってのみ使用できます。

データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。

接続名 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。

データベースドライバ 適切なデータベース タイプを選択します。

接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。

スキーマ

呼び出すストアード プロシージャを含むスキーマを指定します。

プロシージャ

呼び出すストアード プロシージャを指定します。

オプション名

説明

<p>ストアド プロシージャのパラメータ</p>	<p>このテーブルでは、ストアド プロシージャのパラメータの値を指定します。</p>
<p>パラメータ</p>	<p>この列には、ストアド プロシージャで定義されたパラメータが表示されます。</p>
<p>ステージ フィールド</p>	<p>OUT、INOUT、および RETURN パラメータの場合、この列には、パラメータによって返されるデータを格納するデータフロー フィールド名が表示されます。最初は、このフィールド名がパラメータ名と同じになっています。ステージ フィールド名をクリックしてパラメータの新しい名前を入力すると、そのフィールド名を変更できます。IN パラメータの場合、この列は使用されません。</p>
<p>方向</p>	<p>次のいずれかです。</p> <p>IN このパラメータは入力パラメータです。このパラメータに指定した値は、入力としてストアド プロシージャに渡されます。</p> <p>OUT このパラメータは出力パラメータです。ストアド プロシージャは、このパラメータ内のステージにデータを返します。</p> <p>INOUT このパラメータは、ストアド プロシージャに値を渡す入力パラメータとしても、ストアド プロシージャによって返されるデータを受け取る出力パラメータとしても使用できます。</p> <p>RETURN このパラメータには、ストアド プロシージャからのリターンコードが含まれます。</p>
<p>タイプ</p>	<p>この列には、パラメータ値のデータ タイプが表示されます。データタイプが Spectrum™ Technology Platform でサポートされていない場合、タイプは "Unsupported" (未サポート) となり、ストアド プロシージャは正しく実行されません。</p>
<p>値</p>	<p>この列には、パラメータに設定する値を入力します。OUT パラメータの場合、この列は無効になります。</p>

オプション名	説明
結果セット フィールド	<p>このテーブルでは、ストアド プロシージャによって返されるデータのためにどのデータフロー フィールドを使用するかを指定します。</p> <p>データベース テーブル この列には、ストアド プロシージャによって返されたデータの取得元のテーブルが表示されます。</p> <p>データベース フィールド この列には、ストアド プロシージャによって返されたデータの取得元のフィールドが表示されます。</p> <p>ステージ フィールド この列には、データベース フィールドから取得したデータを格納するデータフロー フィールド名が表示されます。</p> <p>タイプ この列には、フィールドのデータ タイプが表示されます。データ タイプが Spectrum™ Technology Platform でサポートされていない場合、タイプは "Unsupported" (未サポート) になります。</p> <p>含む フィールドをデータフローに含むには、この列にあるボックスにチェックを入れます。ボックスにチェックが入っていない場合、そのフィールドはデータフローで使用されません。</p>
フィールドを取得	<p>ストアド プロシージャによって返された結果セット スキーマを使用して結果セット フィールド テーブルを設定するには、このボタンをクリックします。そうすると、ストアド プロシージャが実行され、結果セット スキーマが取得されます。</p>
追加	<p>結果セット フィールドを手動で追加するには、このボタンをクリックします。</p>
削除	<p>使用可能なフィールドのリストから結果セット フィールドを削除するには、このボタンをクリックします。</p>

DB Change Data Reader

DB Change Data Reader ステージでは、チェンジ データ キャプチャ機能が有効化されている列から、現在のジョブフローに含める列を選択できます。

このステージの中で、必要なデータ ソース テーブルであるチェンジ データ キャプチャ (CDC) リソースを作成できます。チェンジ データ キャプチャ機能が有効化されている CDC リソースの列は、チェックボックスによってそれが示されます。

チェンジ データ キャプチャ

チェンジ データ キャプチャ機能では、列に対するすべての変更をキャプチャできます。選択された各列に対し、すべての挿入、更新、削除がキャプチャされます。

サポートされているデータベース

Spectrum™ Technology Platform は現在、MS SQL と Oracle のデータベースに対してのみチェンジ データ キャプチャ (CDC) 機能をサポートしています。

MS SQL MS SQL データ ソースの場合は、チェンジ データ キャプチャ機能をテーブル列に対してバックエンドから有効または無効にできます。そのために必要な手順については、[こちら](#)を参照してください。

注：Spectrum™ のチェンジ データ キャプチャ機能は、SQL Server の Express エディションではサポートされていません。

Oracle Oracle データ ソースの場合は、Spectrum™ は Oracle の LogMiner ユーティリティを使用してテーブル列のデータ変更を追跡します。Oracle データ ソースのテーブルの列に対し、Spectrum™ Technology Platform によって CDC を有効または無効にすることはできません。

insert、update、および delete の各クエリによるデータ変更が、入力された開始日時から現在の日時まで追跡およびキャプチャされます。この入力開始時間は、CDC をオンにした後の最初のクエリ実行に適用されます。

同じ Oracle 接続に対するその後の実行では、データ変更は最後の実行時間から現在の時間までキャプチャされます。

注：これは、漸進的な処理です。最初のキャプチャでは、入力された開始日時から現在の日時までの変更がキャプチャされます。その後のキャプチャでは、1つ前のキャプチャの終了日時から現在の日時までの変更がキャプチャされます。

CDC リソースの追加

注：Spectrum™ のチェンジ データ キャプチャ機能を使用するには、SQL Server Agent が MS SQL Server 上で実行していることを確認します。

1. 次の2つのいずれかの方法で、**[チェンジ データ キャプチャ管理]** ポップアップを開きます。
 - **[ツール]** > **[チェンジ データ キャプチャ管理]** に移動します。
 - **DB Change Data Reader** ステージをジョブに追加し、ステージ設定を開いて **[管理]** をクリックします。
2. **[追加]** をクリックします。
3. CDC リソースの **[名前]** を入力します。
4. **[接続]** フィールドで、使用する SQL データベース接続を選択します。新しいデータベース接続を作成するには、**[管理]** をクリックします。データベース接続の作成の詳細については、「**Database Connection Manager (271ページ)**」を参照してください。
5. **[テーブル/ビュー]** フィールドで、ジョブフローに列を含めるテーブルを指定します。参照ボタン (**[...]**) をクリックして、使用するテーブルまたはビューに移動します。下のグリッドに、選択されたテーブルのすべての列が、各列のデータタイプとともに表示されます。
6. **[接続]** フィールドで Oracle 接続を選択する場合は、**[開始日]** フィールドが使用可能になります。このフィールドには、デフォルト値として現在の日付と 12:00 AM の時間が設定されます。開始日時を自由に変更できます。終了日時は、現在の日時であるとみなされます。

重要： Oracle 接続に対して CDC 機能を使用するには、Oracle LogMiner ユーティリティの実行権限が必要です。詳細については、「**Oracle LogMiner の設定 (339ページ)**」を参照してください。

注：**[接続]** フィールドで MS SQL を選択した場合は、**[開始日]** フィールドは使用できません。

7. **[OK]** をクリックします。

以上で、作成された CDC リソース テーブルを **DB Change Data Reader** ステージで使用するための準備が整います。テーブルの列は、ジョブフローに含めるか、または除外することができます。

CDC 機能が有効になっているテーブル列が、ステージに表示されます。

CDC リソースの編集

1. 次の2つのいずれかの方法で、**[チェンジ データ キャプチャ管理]** ポップアップを開きます。
 - **[ツール]** > **[チェンジ データ キャプチャ管理]** に移動します。
 - **DB Change Data Reader** ステージをジョブに追加し、ステージ設定を開いて **[管理]** をクリックします。
2. 変更する CDC リソースを選択します。
3. **[編集]** をクリックします。
4. 追加された CDC リソースの詳細を必要に応じて変更します。
5. **[OK]** をクリックします。

CDC リソースの削除

1. 次の2つのいずれかの方法で、**[チェンジ データ キャプチャ管理]** ポップアップを開きます。
 - **[ツール]** > **[チェンジ データ キャプチャ管理]** に移動します。
 - **DB Change Data Reader** ステージをジョブに追加し、ステージ設定を開いて **[管理]** をクリックします。
2. 削除する CDC リソースを選択します。
3. **[削除]** をクリックします。

Change Data Reader オプションの選択

[DB Change Data Reader オプション] には、CDC 機能が有効になっている選択された CDC リソースのテーブル列が表示されます。

現在のジョブフローに含める列または除外する列を選択できます。

1. ジョブの中で、**DB Change Data Reader** ステージを追加します。
2. ステージアイコンをダブルクリックすることによって、**[DB Change Data Reader オプション]** を開きます。
3. **[リソースを選択]** ドロップダウンから対象の CDC リソースを選択します。
[管理] をクリックすることによって、CDC リソースを追加または変更できます。

下のグリッドに、すべてのテーブル列がそのデータタイプとともに表示されます。また、特定の列がジョブフローに含まれているかどうかや、その列がチェンジデータキャプチャ機能に対して選択されているかどうかも表示されます。

4. グリッドの **[含める]** 列の下のチェックボックスを使用して、ジョブ フローに含めるテーブル列を選択します。
5. グリッドの **[CDC 有効]** 列の下のチェックボックスは、CDC 機能が有効になっているテーブル列を示します。

[CDC 有効] のチェックボックスは読み取り専用で、CDC 機能が有効になっている列に対応するものがオンになっています。

注：MS SQL データソースの場合は、バックエンドから特定のテーブル列に対してチェンジ データ キャプチャ機能を有効または無効にする手順を[こちら](#)で参照してください。

6. **[OK]** をクリックします。

チェンジデータキャプチャ機能に対して選択されているテーブル列のデータのみが、キャプチャおよび保存されます。

DB Loader

DB Loader ステージでは、Spectrum™ Data Integration モジュールで設定されたデータベースにアクセスし、データを読み書きすることができます。このステージは、高速データロードユーティリティに対するインターフェイスとして機能します。現時点で、Spectrum™ Data Integration プラットフォームは **Oracle Loader**、**DB2 Loader**、**PostgreSQL Loader**、および **Teradata Loader** をサポートしています。

Oracle Loader

Oracle Loader により、Spectrum™ Data Integration プラットフォームで設定された任意の Oracle データベースにデータをロードすることができます。

注：Oracle Loader を使用するには、Oracle クライアントが管理者によってインストールされ、設定されている必要があります。

オプション名	説明
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名 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。</p> <p>データベース ドライバ 適切なデータベース タイプを選択します。</p> <p>接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
テーブル/ビュー	<p>接続を選択した後、出力先のテーブルまたはビューを指定します。参照ボタン ([...]) をクリックして使用するテーブルまたはビューに移動するか、[テーブルの作成] をクリックしてデータベースに新しいテーブルを作成します。</p>
リスナ	<p>Oracle データベースへの接続を確率するのに必要なアドレスと接続の詳細情報が含まれる変数を指定します。例えば、"XE" を指定します。この変数は、tnsnames.ora という名前のファイルの中にあります。このファイルには、クライアント側のネットワーク設定パラメータが格納されます。</p>
ステージ フィールド	<p>この列には、データフローで使用されるフィールド名のリストが表示されます。これらのフィールド名は変更できません。</p>
タイプ	<p>この列には、各フィールドのデータ タイプのリストが表示されます。</p>

[実行時] タブ

オプション名	説明
ロード方法	データを Oracle データベースに書き込む方法を指定します。 追加 データをターゲットのテーブルに追加します。テーブルにある既存のデータは上書きされません。 挿入 データをデータベースにロードします。ロード先のテーブルは空でなければなりません。実行時インスタンスが複数あると、この操作は実行できません。 切り捨てて挿入 既存の行を削除してから、入力データをテーブルにロードします。実行時インスタンスが複数あると、この操作は実行できません。
ダイレクト パス ローダを使用	通常のデータ処理を大幅に省いて、データを Oracle データベースに直接ロードします。
回復不能	[ダイレクト パス ローダを使用] を選択すると、このチェック ボックスが有効になります。REDOログをデータベースに書き込まない場合は、このチェック ボックスをオンにします。REDOログの詳細については、 http://docs.oracle.com/cd/B28359_01/server.111/b28310/onlineredo001.htm#ADMIN11302 を参照してください。
ログ ファイル フォルダ	フォルダへのパスを指定します。省略記号ボタン (...) をクリックし、必要なフォルダを参照して選択します。ログ ファイルには、ロード セッション中のこのステージのアクティビティが記録されます。
不正ファイル フォルダ	フォルダへのパスを指定します。省略記号ボタン (...) をクリックし、必要なフォルダを参照して選択します。不正ファイルには、ステージでデータベースへのロードに失敗したレコードのリストが記録されます。
許容最大エラー数	ロード操作を中止するまでに許容するエラーの最大数を指定します。最初のエラーでロード操作を中止するには、0 を設定します。最大で 32767 個のエラーを許容できます。

注： この操作の複数の実行時インスタンスを使用することによって、パフォーマンスを大幅に向上させることができます。これを行うには、**[実行時]** ボタンをクリックして、必要な数を **[実行時インスタンス]** フィールドに入力します。

DB2 Loader

DB2 Loaderにより、Spectrum™ Data Integration プラットフォームで設定された任意のDB2 データベースにデータをロードすることができます。Spectrum サーバーが動作しているのと同じマシン上に DB2 ユーティリティをセットアップする必要があります。

以下の手順を実行します。

1. 管理者セットアップにより、DB2 ランタイム クライアントをインストールします。
2. ローダー ユーティリティの設定を行います (以下の表を参照)。
3. Spectrum サーバーを起動します。

注: 設定を開始した時点で Spectrum サーバーが既に稼働中だった場合は、設定を有効にするためにこのサーバーを再起動する必要があります。

オプション名	説明
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注: このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名 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。</p> <p>データベース ドライバ 適切なデータベース タイプを選択します。</p> <p>接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
テーブル/ビュー	<p>接続を選択した後、出力先のテーブルまたはビューを指定します。参照ボタン ([...]) をクリックして使用するテーブルまたはビューに移動するか、[テーブルの作成] をクリックしてデータベースに新しいテーブルを作成します。</p>

オプション名	説明
データベース/エイリアス	<p>DB2 サーバーとデータベースをカタログ化する変数です。</p> <p>DB2サーバーをカタログ化するには Spectrum サーバー コンピュータ上で DB2 コマンドライン プロセッサを使用して、次のコマンドを入力します。</p> <pre data-bbox="730 472 1412 556">CATALOG TCPIP NODE <nodename> REMOTE <hostname> SERVER <port></pre> <p>説明:</p> <p>nodename: 接続名</p> <p>hostname: DB2 サーバー コンピュータの TCP/IP 名</p> <p>port: サーバー ポート</p> <p>データベースをカタログ化するには 次のコマンドを使用します。</p> <pre data-bbox="730 840 1412 934">CATALOG DATABASE <databasename> AS <local_database_alias> AT NODE <nodename></pre> <p>説明:</p> <p>databasename: DB2 サーバー上のデータベース名</p> <p>local_database_alias: サーバー コンピュータからの接続中にデータベースに与えられるローカル名</p> <p>nodename: 前回の CATALOG TCP/IP コマンドで使用された名前</p>
ステージ フィールド	<p>この列には、データフローで使用されるフィールド名のリストが表示されます。これらのフィールド名は変更できません。</p>
タイプ	<p>この列には、各フィールドのデータ タイプのリストが表示されます。</p>

[実行時] タブ

オプション名	説明
ロード方法	データを DB2 テーブルに書き込むモードを示します。
	挿入 既存のテーブル データはそのまま、ロードされたデータをテーブルに挿入します。
	置換 既存のテーブル データをすべて削除した後で、ロードされたデータをテーブルに挿入します。 テーブルのスキーマとインデックス定義は変更されません。
	再起動 前回のロード処理が中断された場合に、データのロードを再開します。
回復不能	<p>このロード トランザクションが回復不能であることを示します。</p> <p>このオプションを選択すると、ロード トランザクションは回復不能としてマーク付けされます。テーブル スペースはロード操作の後に Backup Pending 状態にならず、また、ロード操作中にロードされたデータのコピーを作成する必要はなくなります。したがって、rollforward が後で試みられた場合も、回復不能の トランザクションがデータロード失敗時に回復できません。</p> <p>このオプションを選択すると、DB2 rollforward ユーティリティを使用しても トランザクションを回復できません。このユーティリティは、このような回復不能な トランザクションをスキップし、テーブルに "無効" としてマーク付けするからです。また、このテーブルに対するそれ以降の トランザクションも rollforward では無視されます。</p> <p>回復不能な トランザクションが含まれるテーブルを回復するには、回復不能なロードの後のコミット ポイントで作成されたテーブル スペース レベルのバックアップまたはフルバックアップを使用する必要があります。</p> <p style="text-align: center;">注： File Link Control 属性が存在する Datalink 列を含むデータについては、このオプションは選択しないでください。</p>
CPU	ロード ユーティリティが、各データベース パーティションにテーブル オブジェクトを構築しながら、レコードのロード、パーシング、およびフォーマットのために生成して維持できる並列スレッドの数。
ディスク	ロード ユーティリティが、テーブル スペース コンテナニデータを書き込むために生成し、維持できる並列スレッドの数。

オプション名	説明
インデックス作成モード	ロードユーティリティのインデックス処理モードを示します。
自動選択	ロードユーティリティは、データの量とインデックスツリーの深さに基づいて、再構築モードまたはインクリメントモードのどちらを適用するかを決定します。
再構築	すべてのインデックスが再構築されます。
インクリメント	新しいデータが既存のインデックスに追加されます。
	インデックスオブジェクトが有効で、ロード処理を開始した時点でそれらにアクセスできる場合にのみ、このモードを適用できます。
	注：以下の条件がすべて満たされる場合、インクリメンタルモードは使用できません。
	1. ロードコピーのオプションが指定されている (logretain または userexit が有効)。
	2. テーブルが DMS テーブルスペースに存在する。
	3. インデックスオブジェクトが、ロードされるテーブルに属する他のテーブルオブジェクトと共有されるテーブルスペースに存在する。
	この制約を回避するには、インデックスを別のテーブルスペースに配置します。
据え置き	ロードユーティリティは、インデックスを作成しません。既存のインデックスは要更新としてマーク付けされます。
	注：インデックス構築に必要な時間は、据え置きモードの方が再構築モードより長くなります。そのため、複数のロード処理を実行する場合は、ロード処理以外での最初のアクセスでインデックスを再構築するのではなく、最後のロード処理でインデックスを再構築するようにしてください。
	注：このモードは、非ユニークインデックスを持つテーブルに対してのみサポートされます。
高速パース	パフォーマンスを向上するために、列値の構文検証を省くかどうかを示します。
	このオプションを選択すると、パフォーマンスの最適化を優先し、データの構文エラーが無視されます。
	例えば、12wxvg56 という文字列値が ASCII ファイル内の整数列にマッピングされたフィールド内に見つかった場合、ロードユーティリティは通常であれば構文エラーにします。しかし、[高速パース] を選択した場合は、この構文エラーは無視され、ランダムな数値が整数フィールドにロードされます。
	注：このオプションは、正確でエラーのないデータに対してのみ使用してください。

オプション名	説明
スキーマ名	例外テーブルを格納するスキーマ。
テーブル名	ロード中にエラーが発生した場合にその列をコピーして格納する例外テーブル。
ログ ファイル フォルダ	<p>ログ ファイルが格納されるディレクトリへのパス。</p> <p>ログ ファイルには、DB Loader ステージによって 1 回のロード セッションで実行されたデータベース ロード トランザクションのリストが格納されます。</p> <p>省略記号ボタン (...) をクリックして、ログ ファイルを保存するディレクトリを指定します。</p>
不正ファイル フォルダ	<p>不正ファイルが格納される DB2 サーバ上のディレクトリへのパス。</p> <p>不正ファイルには、DB Loader ステージでデータベースへのロードに失敗したレコードのリストが記録されます。</p> <p>省略記号ボタン (...) をクリックして、不正ファイルを保存するディレクトリを指定します。</p>
許容最大エラー数	<p>ロード処理を中止するまで許容するエラーの最大数。</p> <p>エラーが 1 つでも発生したら即座にロード処理を中止する場合は、このフィールドに 0 を設定します。</p> <p>注：最大で 32767 個のエラーを許容できます。</p>
並列処理	<p>環境を複数の物理ノードに複製することで、DB2 データベースは複数のパーティションに分割することができます。</p> <p>データベースからの個々のデータフェッチと更新のリクエストが自動的に複数のパーティションに分割され、並列で処理されることにより、パフォーマンスが最適化されます。</p>
例外処理	<p>DB2 データベースでは、クエリとプロシージャの実行中に発生したエラーと例外を記録し、適切に処理することができます。</p> <p>これを行うため、DB2 データベースは、例外のソースと各データベース例外のログトレースを保存する固有の例外テーブルとスキーマを備えています。</p> <p>例外テーブルを使用する場合は、以下の点を確認してください。</p> <ul style="list-style-type: none"> • DB2 実行時クライアントがバージョン 10.5 以降であること • サービス パックのバージョンが 7 以降であること

PostgreSQL Loader

PostgreSQL Loader により、Spectrum™ Data Integration プラットフォームで設定された任意の PostgreSQL データベースにデータをロードすることができます。

オプション名	説明
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名 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。</p> <p>データベース ドライバ 適切なデータベース タイプを選択します。</p> <p>接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
テーブル/ビュー	<p>参照ボタン ([...]) をクリックしてテーブルに移動するか、使用するテーブルを表示します。</p>
データベース フィールド	<p>この列には、データベース内のフィールド名のリストが表示されます。これらのフィールド名は変更できません。</p>
ステージ フィールド	<p>この列には、データフローで使用されるフィールド名のリストが表示されます。これらのフィールド名は変更できません。</p>
タイプ	<p>この列には、各フィールドのデータ タイプのリストが表示されます。</p>

[実行時] タブ

オプション名	説明
ロード方法	<p>PostgreSQL データベース テーブルにデータを書き込む方法を指定します。</p> <ul style="list-style-type: none"> 空のテーブルにデータを書き込むか、既存のデータ テーブルに追加する場合は、[挿入] を選択します。 データベース テーブルにロードする前にデータを切り捨てる場合は、[切り捨てて挿入] を選択します。

注：この操作の複数の実行時インスタンスを使用することによって、パフォーマンスを大幅に向上させることができます。これを行うには、**[実行時]** ボタンをクリックして、必要な数を **[実行時インスタンス]** フィールドに入力します。

Teradata Loader

Teradata Loader により、Spectrum™ Data Integration プラットフォームで設定された任意の Teradata データベースにデータをロードすることができます。

注：Teradata Loader は、Windows システムのみでサポートされています。

オプション名	説明						
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <table border="0"> <tr> <td>接続名</td> <td>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。</td> </tr> <tr> <td>データベース ドライバ</td> <td>適切なデータベース タイプを選択します。</td> </tr> <tr> <td>接続オプション</td> <td>データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</td> </tr> </table>	接続名	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。	データベース ドライバ	適切なデータベース タイプを選択します。	接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。
接続名	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。						
データベース ドライバ	適切なデータベース タイプを選択します。						
接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。						

オプション名	説明
テーブルビュー	参照ボタン ([...]) をクリックしてテーブルに移動するか、使用するテーブルを表示します。
データベース フィールド	この列には、データベース内のフィールド名のリストが表示されます。これらのフィールド名は変更できません。
ステージ フィールド	この列には、データフローで使用されるフィールド名のリストが表示されます。これらのフィールド名は変更できません。
タイプ	この列には、各フィールドのデータ タイプのリストが表示されます。

[実行時] タブ

オプション名	説明
ロード方法	<p>Teradata データベース テーブルにデータを書き込む方法を指定します。</p> <ul style="list-style-type: none"> 空のテーブルにデータを書き込む場合は、[挿入] を選択します。 既存のデータ テーブルにデータを書き込む場合は、[追加] を選択します。 データベース テーブルに書き込む前にデータを切り捨てる場合は、[切り捨てて挿入] を選択します。
ログ ファイル フォルダ	アップロード処理のログ ファイルを保存するフォルダを選択します。
不良ファイルの生成	このチェックボックスをオンにすると、アップロードに失敗したレコードのログが生成されます。
不正ファイル フォルダ	失敗したレコードのログを保存するフォルダの場所を選択します。ログは、失敗したレコードのエラー コードやエラー フィールドの名前などの詳細情報を表示します。
許容最大エラー数	アップロードセッションにおいて許容するエラー数の上限を指定します。エラー数がこの値を超えると、アップロード処理は一時停止します。

注： Teradata Loader は、実行時インスタンスが複数ある場合をサポートしません。

Field Parser

[Field Parser] ステージは、指定された入力列の XML データおよび区切り記号付きデータからフィールドを抽出します。**[Field Parser]** のオプションを設定するには、以下の操作を実行します。

1. **[ソース]** フィールドから、パースする XML データまたは区切り記号付きデータのある列を選択します。

注：ドロップダウンにすべての文字列入力列が表示されます。

2. パースするデータのタイプに応じて XML または区切り記号付き形式を選択し、以下のオプションを選択します。

XML データ用の Field Parser オプション

オプション名	説明
サーバ名	スキーマの推定用に選択されたファイルが Enterprise Designer を実行しているコンピュータ上にあるか、またはサーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。
スキーマ ファイル	<p>XSD スキーマファイルへのパスを指定します。省略希望ボタン ([...]) をクリックして、ファイルの場所を参照します。サーバーまたはローカル システムにあるスキーマファイルを使用できます。</p> <p>また、XSD ファイルの代わりに XML ファイルを指定することもできます。XML ファイルを指定した場合は、その XML ファイルの構造に基づいてスキーマが推定されます。XSD ファイルの代わりに XML ファイルを使用する場合は、次のような制限があります。</p> <ul style="list-style-type: none"> • XML ファイルは 1 MB 以下でなければなりません。XML ファイルのサイズが 1 MB を越える場合は、XML の構造を維持しつつ、データの一部の削除を試みてください。 • データ ファイルは推定されるスキーマに照らして検証されません。 <p>注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>

オプション名

説明

出力フィールド

このセクションに、選択したスキーマの詳細情報が表示されます。ルート要素と子要素が順番に表示され、さらにそれらの属性が表示されます。

デフォルトでは、スキーマのすべてのノードが選択されています。ただし、次のステージに渡さないノードのチェックボックスはオフにすることができます。

- **[Search node (検索ノード)]**: スキーマツリー内のナビゲート先のノードの名前を入力します。入力したフィールドは、このフィールドの下のプレビューウィンドウで強調表示されます。
- **XPath**: このフィールド内の任意の場所をクリックすると、スキーマツリーでハイライト表示されているノードの要素の XML パス (XPath) と属性が表示されます。以前に表示したすべての XPath を確認するには、右端のフィールドの下矢印をクリックします。

注: XPath は、XML ドキュメント内で情報を検索するための言語です。プロファイリングの詳細については、

「https://www.w3schools.com/xml/xml_xpath.asp」を参照してください。

区切り記号付きデータ用の Field Parser オプション

オプション名

説明

フィールド区切り文字

ドロップダウンリストから、パースする区切り記号付きの列で使われているフィールド区切り文字を選択します。

これ以外の文字がフィールド区切り文字として使われている区切り記号付きの列については、省略記号ボタン (...) をクリックし、別の文字をフィールド区切り文字として選択してください。

Text qualifier

ドロップダウンリストから、パースする区切り記号付きの列で使われているテキスト修飾子を選択します。

注: テキスト修飾子は、区切り記号付きデータ内のテキスト値を囲むために使われる文字です。

これ以外の文字がテキスト修飾子として使われている区切り記号付きの列については、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。

オプション名

説明

名前

パース済み出力を **[リスト]** (値が階層的に表示される) と **[フィールド]** のどちらの形式にするかを選択します。

注: 出力タイプとして **[リスト]** を選択した場合は、1つの出力フィールドのみを追加できます。一方、**[フィールド]** オプションを選択した場合は、複数のフィールドを追加し、パース時に分離された値をそこに取得できます。

出力フィールド

このセクションでは、区切り記号付きの列から詳細なデータを分離するための各種のフィールドを追加または変更することができます。追加した出力フィールドを削除することもできます。

パース済み出力を表示する新しいフィールドを追加するには、**[追加]** ボタンをクリックし、表示される **[フィールド設定]** ポップアップで以下の手順を実行します。

1. フィールドの **[名前]** を入力します。
2. **[タイプ]** ドロップダウンで、追加するフィールドのデータタイプを選択します。選択するタイプに基づいて、さらに追加のフィールドを定義できます。例えば **[日付]** の場合に設定できる形式は、M/d/yy、MMM d. yyyy、または MMMM d. yyyy です。データタイプとその詳細の定義方法については、[区切り記号付き入力ファイルのフィールドの定義 \(171ページ\)](#) を参照してください。

注: データタイプとして **[String]** を選択すると、あらゆるタイプの区切り記号付きデータがパースされます。ただし、フィールド内でパースしたいデータに基づいて、特定のタイプを使用することもできます。

3. **[位置]** フィールドに、このフィールドにパースされる (入力ファイル内の) データタイプの位置を入力します。例えば、次のファイルスニペットで、追加するフィールドに日付/時刻値をパースしたい場合は、**[位置]** に 3 を入力します。

```
true;"02/02/2022";"10/2/92 5:05
AM";598985994665542.25634;1;
"Arjun";74785.155;5:05PM,1,Deepak,65152
false;"15/03/1923";"3/23/90 11:55
AM";3425699466554.2563;2;
"sharma";5.1;5:45AM,2,Arjun,365273
```

4. **[フィールドの追加]** をクリックし、**[閉じる]** をクリックします。

追加したフィールドとその詳細がボックスに表示されます。

注: フィールドの値文字列の先頭と末尾から余分なスペース文字を削除するには、**[トリム]** チェックボックスをオンにします。

[変更]: 追加した出力フィールドの詳細を変更するには、このボタンをクリックします。

[削除]: 追加した出力フィールドを削除するには、このボタンをクリックします。

[Runtime (ランタイム)]: パーサーのランタイム インスタンスを複数指定する場合はこのボタンを使用します。これにより、パフォーマンスが大幅に向上します。

[OK]: このステージで入力した詳細情報をすべて保存するには、このボタンをクリックします。

[Cancel (キャンセル)]: 実施した更新をすべてキャンセルする場合は、このボタンをクリックします。

[Help (ヘルプ)]: このステージのヘルプ ファイルを参照する場合は、このボタンをクリックします。

Field Combiner

Field Combiner ステージは、データフローの前のステージから渡されたフィールドを結合して、XML 文字列を作成します。

Field Combiner のオプション

オプション名	説明
出力列名	結合されたフィールドを XML 文字列として取得する列の名前を指定します。

オプション名	説明
出力フィールド	<p>このセクションでは、結合するフィールドを選択し、それらのフィールドにさまざまなアクションを実行できます。</p> <p>注：スキーマ ウィンドウで 1 つ以上の要素/属性を選択すると、[Apply Namespace (名前空間の適用)] ドロップダウン リストと、[Element (要素)]、[Attribute (属性)]、および [Remove (削除)] ボタンが有効になります。</p> <ul style="list-style-type: none"> • [Quick Add (すばやく追加)]: このボタンをクリックすると、[Add/Remove Fields (フィールドの追加/削除)] ポップアップ ウィンドウが開きます。このウィンドウには前のステージから取得したすべてのフィールドのリストが表示されます。結合するフィールドを選択します。 <p>注：選択したフィールドは、[Search node (検索ノード)] フィールドの下のスキーマ ウィンドウに表示されます。</p> <ul style="list-style-type: none"> • [Search node (検索ノード)]: スキーマ ウィンドウでのナビゲート先のノードの名前を入力します。入力したノードがハイライト表示され、その XML パスがウィンドウの下の [XPath] フィールドに表示されます。 • [Element (要素)]: XML 文字列の属性を要素に変換する場合は、このボタンをクリックします。 • [Attribute (属性)]: XML 文字列の要素を属性に変換する場合は、このボタンをクリックします。 • [Apply Namespace (名前空間の適用)]: 要素または属性の XML ネームスペースを指定する場合は、ここで選択します。 <p>注：名前空間は、上の [Namespace (名前空間)] ボタンをクリックして作成できます。この詳細については、「名前空間の定義」を参照してください。</p> <ul style="list-style-type: none"> • [Remove (削除)]: XML 文字列に不要な要素/属性を削除する場合は、このボタンをクリックします。
XPath	<p>このフィールド内の任意の場所をクリックすると、スキーマ ウィンドウ内でハイライト表示されているノードの XML パス (XPath) が表示されます。以前に表示したすべての XPath を確認するには、右端のフィールドの下矢印をクリックします。</p> <p>注：XPath は、XML ドキュメント内で情報を検索するための言語です。プロファイリングの詳細については、「https://www.w3schools.com/xml/xml_xpath.asp」を参照してください。</p>
	<ul style="list-style-type: none"> • [Runtime (ランタイム)]: Field Combiner のランタイム インスタンスを複数指定する場合はこのボタンを使用します。これにより、パフォーマンスが大幅に向上します。 • [OK]: このステージで入力した詳細情報をすべて保存するには、このボタンをクリックします。 • [Cancel (キャンセル)]: 実施した更新をすべてキャンセルする場合は、このボタンをクリックします。

- **[Help (ヘルプ)]**: このステージのヘルプ ファイルを参照する場合は、このボタンをクリックします。

名前空間の定義

名前空間を使用すると、各要素または属性を XML 名前空間に割り当てることによって、出力内に重複する要素名および属性名を含めることができます。

1 つ以上の名前空間を定義するには

1. **[Field Combiner Options (Field Combiner のオプション)]** 画面で、**[Namespace (名前空間)]** ボタンをクリックします。**[Namespace Details (名前空間の詳細情報)]** ポップアップ ウィンドウが表示されます。
2. **[Prefix (接頭辞)]** 列で、要素または属性に関連付ける接頭辞を入力します。
3. **[名前空間]** 列で、ネームスペースの URL を指定します。
4. これを繰り返して、使用するすべてのネームスペースを定義します。

Field Selector

Field Selector を使用すると、データフローの次のステージに渡すフィールドを選択できます。Field Selector を使用して、データフローから不要なフィールドを削除できます。例えば、2 つのフィールドのデータを結合して新しいフィールドを作成し、2 つのソースフィールドが不要になった場合、Field Selector を使用すると、新しいフィールドのみを保持して 2 つのソース フィールドをデータフローから削除できます。

オプション

オプション	説明
次のステージに送信するフィールドの選択	データフローの次のステージに送信する各フィールドの横にあるボックスにチェックを入れます。フィールドが次のステージに送信されないようにするには、チェックボックスをクリアします。
すべて選択	データフローのすべてのフィールドを選択するには、このボックスにチェックを入れます。このボックスをクリアすると、すべてのフィールドが非選択の状態になります。

Generate Time Dimension

Generate Time Dimension では、指定するデータ範囲のそれぞれの日について日付レコードが 1 つ作成されます。作成されたレコードは、Write to DB ステージを使用してデータベースの時間ディメンションテーブルに書き込むことができます。その後、時間ディメンションテーブルを使用して、期間に基づく正確な計算を実行できます。例えば、四半期ごとの売上、四半期ごとの予算消費、毎日の収益などはすべて、時間ディメンションを必要とする分析です。時間ディメンションテーブルを使用すると、会計年度や標準外の四半期を分析で考慮できます。

時間ディメンション テーブルの使用例

必要な日付データをレコードから容易には抽出できないことがあるため、正確な時間ベースの計算では時間ディメンション テーブルが必要になります。例えば、以下のレコードが売上データベース内にあるとします。レコード間に時間的なギャップがあることに注意してください。例えば、2012 年 1 月 4 日のレコードがありません。

Date	製品	量
2012 年 1 月 3 日	赤いシャツ	10.00 ドル
2012 年 1 月 5 日	赤いシャツ	5.00 ドル
2012 年 1 月 7 日	赤いシャツ	15.00 ドル

これらのレコードに対するクエリを実行して 1 日あたりの平均売上を計算した場合、結果は 10.00 ドル (30 ドル / 3 レコード) になります。しかし、これは正しくありません。この 3 つのレコードは実際には 5 日間にわたって得られたものだからです。日ごとのレコードを持つ時間ディメンションテーブルがあれば、そのテーブルを上記のテーブルと結合して次のテーブルを得ることができます。

Date	製品	量
2012 年 1 月 3 日	赤いシャツ	10.00 ドル
2012 年 1 月 4 日		

Date	製品	量
2012年1月5日	赤いシャツ	5.00 ドル
2012年1月6日		
2012年1月7日	赤いシャツ	15.00 ドル

これらのレコードを使用して1日あたりの平均売上を計算すると、6ドル (30ドル / 5日) という正しい答えが得られます。

また、休日、週末、四半期など、任意の時間属性を計算で考慮することもできます。例えば、2012年1月6日が休日で、1営業日あたりの平均売上にのみ関心があるとした場合、答えは7.50ドルになります。

オプション

Generate Time Dimension には、以下のオプションがあります。

オプション	説明
開始日	時間ディメンションレコードを生成する日付範囲の最初の日です。
終了日	時間ディメンションに対して特定の終了日を指定する場合は、このオプションを選択します。開始日からここで指定する日付までの間の各日についてレコードが1つ生成されます。
期間	特定の日数、月数、または年数にわたる時間ディメンションが必要な場合は、このオプションを選択します。この期間の各日についてレコードが1つ生成されます。例えば、1週間で指定した場合、【開始日】フィールドに指定した日から始まる7つのレコードが生成されます。

オプション

説明

オプション	説明
時間属性	

オプション

説明

	<p>時間ディメンションに含める時間情報のタイプを指定します。各属性は、それぞれの日のレコード内のフィールドになります。次のいずれかです。</p>
Date	<p><日付> <月> <年> (月の名前はサーバーのロケール設定の言語) という形式で表されたその日の日付です。例えば、サーバーが英語ロケールで実行中の場合、日付の表記は "30 October 2012" のようになります。</p>
月初からの日数	<p>その月の何日目であるかを表す数値です。例えば、10 はその月の 10 番目の日であることを意味します。</p>
年初からの日数	<p>その年の何日目であるかを表す数値です。例えば、304 はその年の 304 番目の日であることを意味します。</p>
うるう年	<p>うるう年の日であるかどうかを示す boolean 値です。うるう年の場合は値が true になり、そうでない場合は値が false。</p>
平日	<p>その日が平日 (月曜から金曜まで) であるかどうかを示す boolean 値です。平日の場合は値が true になり、そうでない場合は値が false。</p>
週末	<p>その日が週末 (土曜または日曜) であるかどうかを示す boolean 値です。週末の場合は値が true になり、そうでない場合は値が false。</p>
ユリウス通日	<p>その日に対するユニークな数値です。ユリウス通日の時刻系では、紀元前 4713 年 1 月 1 日を 1 日目としてそれ以降のすべての日に通し番号が与えられます。例えば、2456231 は紀元前 4713 年の 1 月 1 日から数えて 2,456,231 番目の日であることを意味します。</p>
ユリウス通週	<p>その日の週に対するユニークな数値です。ユリウス時刻系では、紀元前 4713 年の最初の週を 1 週目としてそれ以降のすべての週に通し番号が与えられます。例えば、350890 はその日が紀元前 4713 年の最初の週から数えて 350,890 番目の週にあたることを意味します。</p>
ユリウス通年	<p>その年に対するユニークな数値です。ユリウス時刻系では、紀元前 4713 年を 1 年目としてそれ以降のすべての年に通し番号が与えられます。例えば、6725 はその日が西暦 2012 年にあたることを意味します。</p>
月名	<p>その日の月の英語名です。</p>
月番号	<p>その年の何番目の月であるかを示します。例えば、3 はその日がその年の 3 番目の月であることを意味します。</p>
四半期	<p>その年の四半期を表す数値です。例えば、1 はその日がその年の第 1 四半期にあたることを意味します。</p>
年初からの週	<p>その年の何番目の週であるかを表す数値です。例えば、43 はそ</p>

オプション	説明
	<p>数 その日がその年の 43 番目の週にあたることを意味します。</p> <p>曜日名 その曜日の英語名です。例: Monday。</p> <p>曜日番号 月曜日を 1 という日としてその日の曜日を表す数値です。つまり、火曜日は 2、水曜日は 3 というようになります。</p> <p>年 グレゴリオ暦によるその日の年です。例えば、2012 はその日が 2012 年にあたることを意味します。</p>
フィールド	その時間属性を格納するためにデータフロー内に作成されるフィールドの名前です。デフォルトのフィールド名が与えられますが、このフィールド名は変更できます。
タイプ	このフィールドのデータ タイプです。Generate Time Dimension では、各時間属性のデータ タイプが自動的に選択されます。
カレンダー	その時間属性またはデフォルトのグレゴリオ暦の計算時にカスタム カレンダーを使用するかどうかを指定します。カスタム カレンダーを定義するには、 [カレンダー] をクリックします。

カレンダーの作成

カレンダーでは、実行する分析のタイプにとって適切な方法でその年の重要な特性を定義します。デフォルトでは、標準のグレゴリオ暦のカレンダーになっています。ただし、標準のグレゴリオ暦のカレンダーが常に適切であるとは限りません。例えば、会社の事業年度が 1 月 1 日ではなく 6 月 1 日から始まる場合は、年度の開始月を 6 月として時間属性 "month of year" (年度の月) で (1 月を月 1、2 月を月 2 のようにはせずに) 6 月を月 1、7 月を月 2 といった具合に定義できます。

- 以下のいずれかの方法を実行します。
 - Enterprise Designer で Generate Time Dimension ステージを設定する場合は、**[カレンダー]** をクリックします。
 - Management Console で、**[リソース] > [カレンダー]** に移動します。
- [追加]** をクリックします。
- [カレンダー名]** フィールドで、意味のある名前をカレンダーに与えます。
- [開始月]** フィールドで、年度の開始月を選択します。
例えば、7 月を選択すると、7 月が月 1、8 月が月 2、というようになります。

注: 開始月の変更は、以下のフィールドがどのように計算されるかに影響を与えます。
DayOfYear、MonthNumber、Quarter、および WeekOfYear。

5. **[保存]** をクリックします。

出力

Generate Time Dimension では、以下の出力フィールドが作成されます。

表 1 : Generate Time Dimension の出力

フィールド名	説明
Date	<日付> <月> <年> (月の名前はサーバーのロケール設定の言語) という形式で表されたその日の日付です。例えば、サーバーが英語ロケールで実行中の場合、日付の表記は "30 October 2012" のようになります。
DayOfMonth	その月の何日目であるかを表す数値です。例えば、10 はその月の 10 番目の日であることを意味します。
DayOfYear	その年の何日目であるかを表す数値です。例えば、304 はその年の 304 番目の日であることを意味します。
IsLeapYear	うるう年の日であるかどうかを示す boolean 値です。うるう年の場合は値が true になり、そうでない場合は値が false。
IsWeekday	その日が平日 (月曜から金曜まで) であるかどうかを示す boolean 値です。平日の場合は値が true になり、そうでない場合は値が false。
IsWeekend	その日が週末 (土曜または日曜) であるかどうかを示す boolean 値です。週末の場合は値が true になり、そうでない場合は値が false。

フィールド名

説明

JulianDay	その日に対するユニークな数値です。ユリウス通日の時刻系では、紀元前 4713 年 1 月 1 日を 1 日目としてそれ以降のすべての日に通し番号が与えられます。例えば、2456231 は紀元前 4713 年の 1 月 1 日から数えて 2,456,231 番目の日であることを意味します。
JulianWeek	その日の週に対するユニークな数値です。ユリウス時刻系では、紀元前 4713 年の最初の週を 1 週目としてそれ以降のすべての週に通し番号が与えられます。例えば、350890 はその日が紀元前 4713 年の最初の週から数えて 350,890 番目の週にあたることを意味します。
JulianYear	その年に対するユニークな数値です。ユリウス時刻系では、紀元前 4713 年を 1 年目としてそれ以降のすべての年に通し番号が与えられます。例えば、6725 はその日が西暦 2012 年にあたることを意味します。
MonthName	その日の月の英語名です。
MonthNumber	その年の何番目の月であるかを示します。例えば、3 はその日がその年の 3 番目の月であることを意味します。
Quarter	その年の四半期を表す数値です。例えば、1 はその日がその年の第 1 四半期にあたることを意味します。
WeekOfYear	その年の何番目の週であるかを表す数値です。例えば、43 はその日がその年の 43 番目の週にあたることを意味します。
WeekdayName	その曜日の英語名です。例: Monday。
WeekdayNumber	月曜日を 1 という日としてその日の曜日を表す数値です。つまり、火曜日は 2、水曜日は 3 というようになります。
Year	グレゴリオ暦によるその日の年です。例えば、2012 はその日が 2012 年にあたることを意味します。

Query Cache

Query Cache は、1つ以上のデータフロー フィールドに基づいてキャッシュ内のデータを検索し、キャッシュ内のマッチング レコードからデータを返して、キャッシュ レコードのデータをデータフロー内のレコードに追加します。キャッシュ内のデータを検索することにより、データベース内のデータを検索する場合よりパフォーマンスが向上します。

キャッシュには、グローバル キャッシュとローカル キャッシュの 2 種類があります。

グローバル キャッシュのオプション

グローバル キャッシュは、メモリ内に存在するシステム全体の共有キャッシュです。キャッシュを複数のデータフローで使用できるようにする場合や、データが頻繁には変化せず比較的静的な状態の、およびストレージが制限されていない場合は、グローバル キャッシュを選択します。グローバル キャッシュは、書き込みが一度しかできないので静的です。このキャッシュは、いったん作成されると更新ができません。

グローバル キャッシュは、**Write to Cache** ステージによって作成されます。グローバル キャッシュを使用する前に、検索するデータをキャッシュに設定する必要があります。そのためには、**Write to Cache** ステージを含むデータフローを作成します。

オプション名	説明
キャッシュ タイプ	グローバル キャッシュ オプションを選択します。
キャッシュ名	クエリの実行対象とするキャッシュを指定します。 キャッシュを作成するには、 Write to Cache ステージを使用します。
キャッシュ フィールド	この列には、キャッシュ内のフィールドのリストが表示されます。これらのフィールド名は変更できません。
ステージ フィールド	この列には、データフローで使用されるフィールド名のリストが表示されます。フィールド名を変更する場合は、そのフィールド名をクリックして新しい名前を入力します。

オプション名	説明
タイプ	この列には、各データフロー フィールドのデータ タイプのリストが表示されます。
含める	クエリによってキャッシュ フィールドの値を返すには、この列内のボックスにチェックを入れます。クエリによってキャッシュ フィールドを返さない場合は、ボックスをクリアします。

オプション名	説明
--------	----

デフォルト エラー値	
------------	--

オプション名

説明

クエリでエラーが発生した場合にデータフロー フィールドに表示する値を指定します。ドロップダウンリストに、クエリ対象のフィールドのデータタイプに応じた有効な値が表示されます。例えば、**integer** の場合、オプションとして **-1** が表示されます。

このフィールドに値を入力することもできます。さまざまなデータタイプに対する有効なデフォルト エラー値の一覧については、以下の表を参照してください。

データ 有効なデフォルト エラー値とデータ タイプ (カッコ内) タイプ

	Null	-1 (整数)	1899- 12-30 12:00:00 (日付/ 時刻)	1899- 12-30 12:00:00 (日付/ 時刻)	12:00:00 (時刻)	False	空
Date							
Integer		✓					
長		✓					
Float		✓					
Big 小数度		✓					
Double		✓					
文字列	✓	✓	✓	✓	✓	✓	✓
時間					✓		
Date Time			✓				

オプション名

説明

データ 有効なデフォルト エラー値とデータ タイプ (カッコ内)
タイプ

Null	-1	1899-	1899-	12:00:00	False	空
	(整数)	12-30	12-30	(時刻)		
		12:00:00	(日付)			
		(日付/	時刻)			

Boolean



キー フィールド

検索キーとして使用される、キャッシュ内のフィールドを指定します。**【入力フィールド】**列にあるフィールドの値がキャッシュ内のキー フィールドの値と一致している場合、クエリはキャッシュ内のそのレコードからデータを返します。

入力フィールド

値がキーとして使用されるデータフロー フィールドを指定します。このフィールドの値がキャッシュ内のキーフィールドの値と一致している場合、クエリはキャッシュ内のそのレコードからデータを返します。

ローカル キャッシュのオプション

ローカル キャッシュは、**Query Cache** ステージの実行時にのみ使用される一時的なキャッシュです。**Query Cache** は、選択されたデータベース テーブルからキャッシュを構築します。そして、キー フィールドと検索条件に基づいてキャッシュ内のデータを検索し、キャッシュ内で一致したレコードからデータを返して、そのキャッシュ レコードのデータをデータフロー内のレコードに追加します。

ローカル キャッシュは、**Query Cache** のジョブ実行時に作成されるので、動的です。**Query Cache** がデータの読み込みを完了すると、このキャッシュは自動的にメモリから削除されます。ローカル キャッシュは、**Query Cache** ステージが実行されるたびに作り直されます。1 つのデータフローでのみ使用される場合や、検索テーブルが頻繁に変化する場合は、ローカル キャッシュを選択します。

オプション名	説明
キャッシュ タイプ	ローカル キャッシュ オプションを指定します。
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名 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。</p> <p>データベース ドライバ 適切なデータベース タイプを選択します。</p> <p>接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
テーブル/ビュー	クエリの実行対象とするデータベース内のテーブルまたはビューを指定します。
データベース フィールド	この列には、データベース内のフィールドのリストが表示されます。これらのフィールド名は変更できません。
ステージ フィールド	この列には、データフローで使用されるフィールド名のリストが表示されます。フィールド名を変更する場合は、そのフィールド名をクリックして新しい名前を入力します。
タイプ	この列には、各データフロー フィールドのデータ タイプのリストが表示されます。
含める	クエリによってキャッシュ フィールドの値を返すには、この列内のボックスにチェックを入れます。クエリによってキャッシュ フィールドを返さない場合は、ボックスをクリアします。

オプション名	説明
--------	----

デフォルト エラー値	
------------	--

オプション名 説明

クエリでエラーが発生した場合にデータフロー フィールドに表示する値を指定します。ドロップダウン リストに、クエリ対象のフィールドのデータ タイプに応じた有効な値が表示されます。例えば、**integer** の場合、オプションとして **-1** が表示されます。

このフィールドに値を入力することもできます。さまざまなデータ タイプに対する有効なデフォルト エラー値の一覧については、以下の表を参照してください。

データ タイプ 有効なデフォルト エラー値とデータ タイプ (カッコ内)

データ タイプ	Null	-1 (整数)	1899- 12-30 12:00:00 (日付)	1899- 12-30 12:00:00 (日付/時 刻)	12:00:00 (時刻)	False	空
Date							
Integer		✓					
長		✓					
Float		✓					
Big 小数度		✓					
Double		✓					
文字列	✓	✓	✓	✓	✓	✓	✓
時間					✓		
Date Time			✓				

オプション名	説明							
データタイプ	有効なデフォルト エラー値とデータ タイプ (カッコ内)							
	<table border="1"> <tr> <td>Null</td> <td>-1 (整数)</td> <td>1899-12-30 12:00:00 (日付/時刻)</td> <td>1899-12-30 12:00:00 (日付/時刻)</td> <td>12:00:00 (時刻)</td> <td>False</td> <td>空</td> </tr> </table>	Null	-1 (整数)	1899-12-30 12:00:00 (日付/時刻)	1899-12-30 12:00:00 (日付/時刻)	12:00:00 (時刻)	False	空
Null	-1 (整数)	1899-12-30 12:00:00 (日付/時刻)	1899-12-30 12:00:00 (日付/時刻)	12:00:00 (時刻)	False	空		
	Boolean ✓							
キー フィールド	検索キーとして使用される、データベース内のフィールドを指定します。【入力フィールド】列にあるフィールドの値がデータベース内の【キー フィールド】の値と一致している場合、クエリはデータベース内のそのレコードからデータを返します。							
タイプ	キー フィールドの値のデータ タイプ							
演算子	<p>必要な演算子を選択します。サポートされる演算子は以下のとおりです。</p> <ul style="list-style-type: none"> • = • != • > • >= • < • <= 							
定数	クエリを実行し、入力フィールドではなく、入力する定数に基づいて値を返す場合は、このチェック ボックスを選択します。							
入力フィールド	値がキーとして使用されるデータフロー フィールドを指定します。このフィールドの値がデータベース内の【キー フィールド】の値に一致している場合、クエリはデータベース内のそのレコードからデータを返します。							

高度なキャッシュ オプション

高度なキャッシュとは、ローカルキャッシュに似た一時的なキャッシュのことです。Query Cache ステージの実行時に使用されます。クエリで指定されたテーブルからデータを読み込む、SQL ク

エリに基づくキャッシュを構築します。そして、**where** 句で指定された検索キーに基づいてキャッシュ内のデータを検索し、キャッシュ内で一致したレコードからデータを返して、そのキャッシュレコードのデータをデータフロー内のレコードに追加します。

高度なキャッシュは、**Query Cache** のジョブ実行時に作成されるので、動的です。**Query Cache** がデータの読み込みを完了すると、このキャッシュは自動的にメモリから削除されます。高度なキャッシュは、**Query Cache** ステージが実行されるたびに作り直されます。複数のテーブルからデータを読み込む場合や、キャッシュの作成に複雑なクエリを実行する必要がある場合は、高度なキャッシュ オプションを選択します。

オプション名	説明
キャッシュ タイプ	高度なキャッシュ オプションを指定します。
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名</p> <p>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。</p> <p>データベース ドライバ</p> <p>適切なデータベース タイプを選択します。</p> <p>接続オプション</p> <p>データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
クエリ	<p>データベースからデータを読み込む SQL クエリ。クエリは、複数のテーブルからデータを読み込むことができます。</p> <p>注：クエリではエイリアスの入力が必要で。</p>
説明：	<p>クエリに基づいて作成されたキャッシュを検索する where 句として使用するテキスト。\$ 演算子を前に付けることによって入力フィールドをクエリ内で指定できます。例えば、_id = \${_inputId} と指定します。ここで _inputId は入力フィールドで、_id はキャッシュの検索列です。</p>

オプション名	説明
フィールドを取得	SQL クエリを用いてキャッシュするように選択されたフィールドを、グリッドに設定します。
データベース フィールド	この列には、データベース内で取得されたフィールドのリストが表示されます。これらのフィールド名は変更できません。
ステージ フィールド	この列には、データフローで使用されるフィールド名のリストが表示されます。フィールド名を変更する場合は、そのフィールド名をクリックして新しい名前を入力します。
タイプ	この列には、各データフロー フィールドのデータ タイプのリストが表示されます。

オプション名	説明
--------	----

デフォルト エラー値	
------------	--

オプション名 説明

クエリでエラーが発生した場合にデータフロー フィールドに表示する値を指定します。ドロップダウン リストに、クエリ対象のフィールドのデータ タイプに応じた有効な値が表示されます。例えば、**integer** の場合、オプションとして **-1** が表示されます。

このフィールドに値を入力することもできます。さまざまなデータ タイプに対する有効なデフォルト エラー値の一覧については、以下の表を参照してください。

データ タイプ 有効なデフォルト エラー値とデータ タイプ (カッコ内)

データ タイプ	Null	-1 (整数)	1899- 12-30 12:00:00 (日付/時刻)	1899- 12-30 12:00:00 (日付)	12:00:00 (時刻)	False	空
Date							
Integer		✓					
長		✓					
Float		✓					
Big 小数度		✓					
Double		✓					
文字列	✓	✓	✓	✓	✓	✓	✓
時間					✓		
Date Time			✓				

オプション名	説明
データ タイプ	有効なデフォルト エラー値とデータ タイプ (カッコ内)
Null	-1 (整数)
	1899-12-30 12:00:00 (日付/時刻)
	1899-12-30 12:00:00 (日付)
	12:00:00 (時刻)
Boolean	False 空

[実行時] タブ

[実行時] タブにあるオプションは、グローバル キャッシュ、ローカル キャッシュ、高度なキャッシュで共通です。

オプション名	説明
マッチ オプション	クエリにマッチするキャッシュ内のレコードが複数ある場合にするかを指定します。
すべてのマッチを返す	マッチする値をキー フィールドに持つ、キャッシュ内のすべてのレコードからデータを返します。
最初にマッチしたレコードを返す	マッチする値をキー フィールドに持つ、キャッシュ内の最初のレコードのみからデータを返します。
最後にマッチしたレコードを返す	マッチする値をキー フィールドに持つ、キャッシュ内の最後のレコードのみからデータを返します。

オプション名

説明

ステージ オプション

このセクションには、このステージの SQL クエリで使用されるデータフロー オプションの一覧が表示され、そのすべてのオプションに対してデフォルト値を設定できます。**【名前】**列にはオプションが表示され、そのデフォルト値を対応する**【値】**列に入力できます。SQL クエリに変数を挿入すると、ステージオプションが有効になります。例えば、SQL フィールドのこのクエリにより、ステージオプションの CustomerID と InvoiceID が表示されるため、それぞれに個別のデフォルト値を設定できます。

```
Select * From [Sales].[CustomerTransactions]
where#{CustomerID}#{InvoiceID}
```

注：ここで設定したデフォルト値は、**【データフロー オプション】**ダイアログボックスの**【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】**セクションにも表示されます。このダイアログボックスでも、デフォルト値を変更できます。**【ステージオプション】**、**【データフロー オプション】**、および **Job Executor** で設定されたオプションのデフォルト値が競合する場合の優先順位は、**Job Executor** で指定された値 > **【データフロー オプション】**ダイアログボックスで定義された値 > **【ステージオプション】**で入力された値の順になります。

Query DB

Query DB ステージでは、フィールドをデータベース クエリのパラメータとして使用し、クエリの結果をデータフローの新しいフィールドとして返すことができます。

注：空間データベースに対してクエリを実行する場合は、Query DB の代わりに Query Spatial Data を使用してください。

[全般] タブ

オプション	説明						
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p>注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <table><tr><td data-bbox="565 741 808 772">接続名</td><td data-bbox="816 741 1421 804">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。</td></tr><tr><td data-bbox="565 825 808 856">データベースドライバ</td><td data-bbox="816 825 1421 856">適切なデータベース タイプを選択します。</td></tr><tr><td data-bbox="565 877 808 909">接続オプション</td><td data-bbox="816 877 1421 940">データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</td></tr></table>	接続名	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。	データベースドライバ	適切なデータベース タイプを選択します。	接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。
接続名	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。						
データベースドライバ	適切なデータベース タイプを選択します。						
接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。						
テーブル/ビュー	クエリの実行対象とするデータベース内のテーブルまたはビューを指定します。						

オプション

説明

説明：

WHERE 文を使用する場合は、ここに入力します。文の中に単語 WHERE を実際に含める必要はありません。WHERE 文の目的は、指定された条件と一致するレコードのデータのみを返すことです。

データフロー フィールドの値を指定するには、次の構文を使用します。

```
${<field name>}
```

<field name> は、データフローのフィールドの名前です。

例：

```
account_number=${customer_key}
```

この例では、クエリは、テーブル列 account_number の値がデータフロー フィールド customer_key の値と一致するレコードのデータを返します。

注：大文字と小文字が区別されるデータベースに対してクエリを実行している場合は必ず、データベース テーブル内で使用されているのと同じ形式でフィールド名を入力してください。つまり、テーブル作成時にフィールド名を引用符 (") で囲んだ場合は、フィールド名を引用符で囲みます。

[プレビュー] をクリックし、定義した条件に基づいてデータをプレビューします (最初の 50 件のレコードがプレビューされます)。

注：WHERE 文でデータフロー フィールドを使用している場合は、Query DB のプレビュー機能は動作しません。その代わりに、Enterprise Designer のデータフロー インспекション ツールを使用して結果をプレビューできます。

結果のないレコードを返す

クエリが結果を返さないレコードを QueryDB によって返させる場合は、このボックスをオンにします。このチェック ボックスをオフにすると、レコードは返されません。このオプションをオンのままにしておくことを推奨します。

含める

フィールド テーブルで、フィールドの横にある **[含める]** ボックスをクリックして、条件フィールドを選択します。

[ソート] タブ

フィールドの値に基づいて、レコードをソートする場合は、ソートするフィールドを指定します。

Query DB を実行時にパラメータ化

Query DB ステージを設定して、実行時に WHERE 句に値を指定することができます。これは、WHERE 句に使う列名をデータフロー オプションで設定できるようにしたい場合に便利です。

1. Enterprise Designer でデータフローを開きます。
2. **[接続]** フィールドと **[テーブル/ビュー]** フィールドを設定して、クエリの実行対象とするデータベースを指定します。
3. **[WHERE]** フィールドには、パラメータ化する値を $\${パラメータ}$ 形式で表した WHERE 文を入力します。

例:

```
 $\${COL}=\${EmployeeID}$ 
```

ここで COL は、テーブルの列名を実行時に埋め込む Dataflow オプションを意味します。

4. Query DB オプションのウィンドウを閉じます。
5. 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]** ウィンドウが表示されます。
6. **[追加]** をクリックします。**[データフロー オプションの定義]** ウィンドウが表示されます。
7. Query DB ステージを選択します。
8. **[オプション名]** と **[オプション ラベル]** を指定します。

[オプション名] フィールドは、WHERE 句に $\${パラメータ}$ 形式で入力した値と同じにする必要があります。**[オプション ラベル]** フィールドで、オプションのラベル名を指定できます。オプション名とラベル名を同じにしてもかまいません。

例: COL

9. **[デフォルト値]** を指定します。例: "EmpID".
10. **[OK]** をクリックします。

この手順で、実際のデータベースの列名、つまり EmpID が実行時にオプション名 "COL" に対応付けられます。データベースの列名は、使用するデータベースで有効な引用符識別子を使って適切に囲む必要があります。

Query NoSQL DB

Query NoSQL DB ステージでは、必要なデータを NoSQL データベースから検索できます。サポートされているのは MongoDB データベースです。

[全般] タブ

フィールド名	説明
接続	<p>ドロップダウンリストから必要なデータベース接続を選択します。表示されるオプションは、Management Console に定義されている接続によって異なります。</p> <p>新しい接続を追加するには、NoSQL への接続 (62ページ) を参照してください。</p> <p>既存の接続を変更するには、Management Console の [データ ソース] ページの接続リストから接続を選択して開き、必要な更新を行い、[保存] ボタンをクリックします。</p>
テーブル/ビュー	<p>クエリの実行対象とするデータベース内のコレクションまたはビューを指定します。</p> <p>注： MongoDB データベースでは、テーブル/ビューは「コレクション」と呼ばれます。</p>
スキーマ ファイル	<p>参照ボタン (...) をクリックして JSON スキーマ ファイル を選択します。このファイルはオプションです。[フィールド] タブのフィールドは、スキーマ ファイルまたはデータベース テーブル/ビューのいずれかを使用して再生成できます。</p> <p>選択されたファイル パスを非選択にするには、[クリア] をクリックします。</p> <p>注：スキーマファイルが選択されている場合は、フィールドは必ずスキーマファイルを用いて生成されます。</p>
説明：	<p>where 句を入力して検索の条件を定義します。</p> <p>WHERE 句でサポートされる演算子については、「http://docs.mongodb.org/manual/reference/operator/query/」を参照してください。</p>

フィールド名	説明
プレビュー	<p>選択したテーブルまたはビューのレコードを表示します。</p> <p>注：[プレビュー] をクリックすると、[Where] フィールドに指定されたフィルタ条件を適用しないで、選択されたデータベースから最初の 50 件のレコードが取得されます。</p>
すべて展開	プレビュー ツリーの項目を展開します。
すべて折りたたむ	プレビュー ツリーの項目を折りたたみます。

[フィールド] タブ

[フィールド] タブでは、次のステージに渡すデータを選択できます。詳細については、「[フィールドの定義 - Query NoSQL DB \(153ページ\)](#)」を参照してください。

フィールドの定義 - Query NoSQL DB

[フィールド] タブに、NoSQL DB/スキーマ ファイルに定義されたフィールドとタイプが表示されます。

- [フィールド] タブで、[フィールドの再生成] をクリックします。

これによって、先頭 50 件のレコードに基づく集計データが生成されます。データは、Fieldname (datatype) という形式で表示されます。

注：スキーマ ファイルを参照する場合は、スキーマ ファイルを使用してフィールドが生成されます。テーブル/ビューは無視されます。スキーマ ファイルをリセットするには、[クリア] をクリックします。
- フィールドの名前とタイプを変更するには、フィールドをハイライト表示し、[変更] をクリックします。
- [名前] フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。
- [タイプ] フィールドで、データに対して数学的な操作を行う予定がない場合は、データ タイプを文字列のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータ タイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータ タイプに変換されます。

このステージでは、以下のデータ タイプがサポートされています。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
long	正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
string	文字シーケンス。

5. テーブルまたはスキーマファイルに存在しないフィールドを追加することもできます。**[追加]** をクリックして新しいフィールドを追加します。フィールドを削除する場合は、**[削除]** をクリックします。

注：リスト タイプの下にのみ、新しいフィールドを追加できます。

6. **[OK]** をクリックします。

データフロー オプションの設定 - Query NoSQL DB

以下では、**Query NoSQL DB** の実行時オプションをサポートするようにデータフローを設定する手順について説明します。

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. **[Query NoSQL DB]** ステージを展開します。
6. MongoDB データベースにクエリを実行するため、次のデータフロー オプションがエクスポートされます。
 - 接続

- テーブル

選択した **[Query NoSQL DB]** オプション名が、**[オプション名]** フィールドと **[オプション ラベル]** フィールドに表示されます。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。

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、**[選択ステージ]** オプションを選択します。
9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **[OK]** をクリックします。
12. 必要に応じて、オプションの追加を続けます。
13. 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。
14. データフローを保存してエクスポートします。

Read From DB

Read From DB ステージでは、データフローへの入力として、データベーステーブルまたはビューからデータを読み込みます。このステージは、ジョブ、サービス、サブフローに対して使用できますが、プロセスフローには使用できません。このステージでは、入力フィールドのタイプの変更が可能です。

注：**Read from DB** ステージは、date データタイプのすべての値を String 値として読み取ります。これは、Spectrum で使用されるデフォルト ドライバである *jTDS* ドライバの動作です。すべての date データタイプの値をそのまま処理するには、Microsoft の JDBC ドライバを使用します。

関連するタスク:

Read from DB ステージを使用できるようにするには、**Management Console** の **Connection Manager** を使用してそれぞれのデータベースへの接続を作成する必要があります。詳細については、「**Write to DB (267ページ)**」の **Database Connection Manager** を参照してください。

[全般] タブ

フィールド名	説明
接続	<p>使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、[管理] をクリックします。</p> <p style="text-align: center;">注：このオプションは Enterprise Designer によってのみ使用できます。</p> <p>データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。</p> <p>接続名 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。</p> <p>データベースドライバ 適切なデータベース タイプを選択します。</p> <p>接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。</p>
SQL	<p>データフローの実行時にデータ ソースから読み込む必要のあるレコードを指定する SQL クエリを入力します。このフィールドに SQL クエリを手動で入力できます。または、[SQL の作成] をクリックして、Visual Query Builder によってクエリを作成することもできます。</p> <p>SQL クエリには、実際の列名の代わりに変数を含めることができます。変数を使用すると、クエリを実行時にカスタマイズすることができます。詳細については、「クエリ変数 (158ページ)」を参照してください。</p> <p>データベースの <i>public.Branch</i> テーブルから <i>BranchID</i> と <i>BranchType</i> を公開するクエリのサンプルは以下のようになります。</p> <pre style="background-color: #f0f0f0; padding: 10px;">select * from "public"."Branch" where "BranchID" = # {ID} and "BranchType" = # {Type}</pre> <p style="text-align: center;">注： <i>Integer</i> タイプフィールドの値は引用符なしで入力できますが、<i>String</i> タイプは単一引用符で囲む必要があります。</p>

フィールド名 説明

SQL の作成

注：このオプションは **Enterprise Designer** によってのみ使用できます。

[SQL の作成] をクリックすることにより、複数の列を選択し、結合や入れ子になったクエリを作成して、複雑なクエリを作成します。**[Visual Query Builder]** が開きます。詳細については、「**Visual Query Builder (161ページ)**」を参照してください。

注：Visual Query Builder を使用して作成したクエリは、列やテーブルの完全修飾名とともに **[SQL]** フィールドに表示されます。

フィールドの再生成/スキーマの生成

クエリによって取得されるデータのスキーマを表示するには、**[フィールドの再生成]** (Enterprise Designer でアクセスしている場合)、および **[スキーマの生成]** (Flow Designer を使用している場合) をクリックします。

既存のクエリを編集した場合は、このボタンをクリックすると変更されたスキーマが取得されます。

注：このボタンをクリックするとき、SQL クエリのエンティティ名はそのまま維持され、完全修飾名に置き換えられません。

プレビュー

SQL クエリによって取得されるレコードのサンプルを表示するには、**[プレビュー]** をクリックします。

[実行時] タブ

フィールド名 説明

フェッチ サイズ

データベース テーブルから一度に読み取るレコードの数を指定するには、このオプションを選択します。例えば、フェッチ サイズの値が 100 で、読み取るレコードの総数が 1000 である場合、すべてのレコードを読むためにデータベースに 10 回アクセスすることになります。

最適なフェッチ サイズを設定すれば、パフォーマンスを大幅に向上できます。

注：使用している環境に最適なフェッチ サイズは、Read from DB ステージと Write to Null ステージ間の実行時間をテストすることで計算できます。詳細については、「**最適なフェッチ サイズの決定 (342ページ)**」を参照してください。

フィールド名

説明

ステージ オプション

このセクションには、このステージの SQL クエリで使用されるデータフロー オプションの一覧が表示され、そのすべてのオプションに対してデフォルト値を設定できます。**[名前]**列にはオプションが表示され、そのデフォルト値に対応する**[値]**列に入力できます。SQL クエリに変数を挿入すると、ステージ オプションが有効になります。例えば、SQL フィールドのこのクエリにより、ステージ オプションの CustomerID と InvoiceID が表示されるため、それぞれに個別のデフォルト値を設定できます。

```
Select * From [Sales].[CustomerTransactions]
where#{CustomerID}#{InvoiceID}
```

注：ここで設定したデフォルト値は、**[データフロー オプション]** ダイアログボックスの**[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]**セクションにも表示されます。このダイアログボックスでも、デフォルト値を変更できます。**[ステージ オプション]**、**[データフロー オプション]**、および **Job Executor** で設定されたオプションのデフォルト値が競合する場合の優先順位は、**Job Executor** で指定された値 > **[データフロー オプション]** ダイアログボックスで定義された値 > **[ステージ オプション]** で入力された値の順になります。

クエリ変数

Read From DB ステージにおいて、クエリを定義する際に、実際の列名の代わりに変数を含めることができます。クエリの中で変数を使用すると、クエリ条件を (データフロー オプションを使用して) 実行時にカスタマイズしたり、**Job Executor** によってカスタマイズしたりできます。

一方、**[実行時]** タブでステージ オプションの値を指定して、クエリによって取得されるレコードのスキーマとサンプルを表示することもできます。それぞれ、**[フィールドの再生成]** ボタンと **[プレビュー]** ボタンを使用します。

変数は **#{variable}** という形式で定義し、SQL クエリの select 句または where 句に挿入します。例えば、以下のクエリでは customerID が変数として挿入されていますが、これは **[実行時]** タブで定義できます。

```
Select * From [Sales].[CustomerTransactions] where#{CustomerID}
```

注：**Visual Query Builder** を使用して生成したクエリを編集し、変数を含めることができます。ただし、編集したクエリはその後、**Visual Query Builder** に読み込むことができなくなります。手動で記述または生成した SQL クエリに変数を含めると、**[SQL の作成]** ボタンは無効になります。

クエリ変数の挿入

1. **Read From DB** ステージを含む、必要なジョブを開きます。あるいは、**Read From DB** ステージをジョブに追加します。
2. **Read from DB** ステージの **[Read from DB オプション]** を開きます。
3. **[SQL]** フィールドで SQL クエリを作成します。手動で作成するか、**Visual Query Builder** を使用します。詳細については、「**Visual Query Builder (161ページ)**」を参照してください。
4. クエリの where 句に、`#{variable}` という構文で変数を使用して必要な条件を追加します。
例えば、テーブル CUSTOMERS があり、列 AGE には 28、32、30 などの値が入っており、列 SALARY には 1000、1500、2200 などの値が入っている場合、以下のように SQL クエリを構成します。

```
select * from CUSTOMERS where #{condition1} > 28 and #{condition2} > 1200
```

注：この SQL クエリの where 句に変数を挿入すると、**[SQL の作成...]** ボタンが無効になります。

5. クエリによって取得されるスキーマとサンプル レコードを表示するには、**[実行時]** タブでステージ オプションの値を指定して、**[フィールドの再生成]** ボタンと **[プレビュー]** ボタンをそれぞれクリックします。
6. **[OK]** をクリックします。

これでこの SQL クエリの where 句は、**[データフロー オプション]** を使用して実行時に、または Job Executor によってジョブを実行する際に、カスタマイズできます。

注：変数は、SQL クエリの select 句にも配置できます。ただし、その場合の変数名は、クエリ対象のテーブルのいずれかの列の名前と一致する必要があります。

データフロー オプションとしてのクエリ変数の設定

1. **Read From DB** ステージで変数を含むクエリが定義済みの、必要なジョブを開きます。
2. **[編集]** > **[データフロー オプション]** を開きます。
3. **[追加]** をクリックします。
4. **[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]** セクションで、**[Read From DB]** エントリを展開します。
Read From DB ステージで SQL クエリに対して定義した変数が、ステージの他の属性とともに一覧表示されます。
5. 対応するチェックボックスを使用して、カスタマイズする変数を選択します。
6. **[オプション ラベル]** フィールドに、変数に関連する名前を入力します。

7. **[デフォルト値]** フィールドに、SQL クエリの `where` 句で、選択された変数の代わりに使用する列名を入力します。または、`where` 句の変数の代わりに使用する定数値を入力します。例えば、**Read From DB** ステージで定義された次の SQL クエリがあるとします。

```
select * from CUSTOMERS where #{condition1} > 28 and #{condition2} > 1200
```

テーブル `CUSTOMERS` の列 `AGE` を変数 `condition1` のデフォルト値として選択し、列 `SALARY` を変数 `condition2` のデフォルト値として選択できます。

実行時に、クエリは次のように解釈されます。

```
select * from CUSTOMERS where AGE > 28 and SALARY > 1200
```

8. **Read From DB** ステージで SQL クエリに挿入されたすべての変数に対して、ステップ 5～7 を繰り返します。
9. **[OK]** をクリックします。
- データフローを実行すると、カスタマイズされたクエリを用いて必要なデータが取得されます。

Job Executor のクエリ変数の設定

注：Spectrum™ Job Executor がお使いのサーバー上にダウンロード済みであることを確認します。

1. ジョブの **Read From DB** ステージの SQL クエリに含まれている変数のデフォルト値を定義するテキストファイルを作成します。
- 列名 `AGE` を変数 `condition1` のデフォルト値として割り当てるには、例えば `variables.txt` といった名前のテキストファイルを作成し、次の行をこのファイルに入れます。

```
condition1=AGE
```

20 などの定数値を変数 `condition1` のデフォルト値として割り当てるには、次の行をこのファイルに入れます。

```
condition1=20
```

2. コマンドプロンプトで **Job Executor** を使用してジョブを実行する際に、引数 `-o` の後に作成したテキストファイルのパスを指定します。
- 例えば、テキストファイル `ReadCustomerDataJob` に変数のデフォルト値が定義されたジョブ `variables.txt` を実行するには、コマンドプロンプトで次のコマンドを実行します。

```
java -jar jobexecutor.jar -h "localhost" -u "admin" -p "admin" -s "8080" -j "ReadCustomerDataJob" -o "variables.txt"
```


Job Executor を使用してジョブを実行すると、カスタマイズされたクエリを用いて必要なデータが取得されます。

注: 手順とコマンドライン構文については、『データフロー デザイナーズ ガイド』の「ジョブの実行」を参照してください。

Visual Query Builder

注: このオプションは **Enterprise Designer** によってのみ使用できます。

Visual Query Builder は、Read from DB ステージで複雑な SQL クエリを作成するための視覚的なインターフェイスを備えています。Visual Query Builder を使用するには、SQL の概念に関する基礎知識が必要です。

Visual Query Builder にアクセスするには、Read from DB で **[SQL の作成]** ボタンをクリックします。

クエリ ビルダのメイン ウィンドウは次の 2 つの部分に分かれています。

- **[Query Building]** 領域は、クエリの視覚的表現が表示されるメイン領域です。この領域を使用して、ソース データベース オブジェクトやそれらのオブジェクト間のリンクを定義したり、テーブルとリンクのプロパティを設定したりできます。
- **[Columns]** ウィンドウは、クエリ作成領域の下にあります。この領域を使用して、クエリ出力列および式に関して必要なすべての操作を実行します。この領域では、フィールドのエイリアスの定義、ソートおよびグループ化、条件の定義を行うことができます。
- クエリ作成領域の上のページ コントロールを使用して、メイン クエリとサブクエリを切り替えることができます。

クエリへのオブジェクトの追加

オブジェクトをクエリに追加するには、**[データベース オブジェクト]** ツリーを使用します。ツリー内のオブジェクトは、データベース、スキーマ、およびタイプごとにグループ化されます。クエリに追加するオブジェクトをドラッグして、クエリ作成領域にドロップします。または、オブジェクトをダブルクリックすることによって、クエリに追加することもできます。

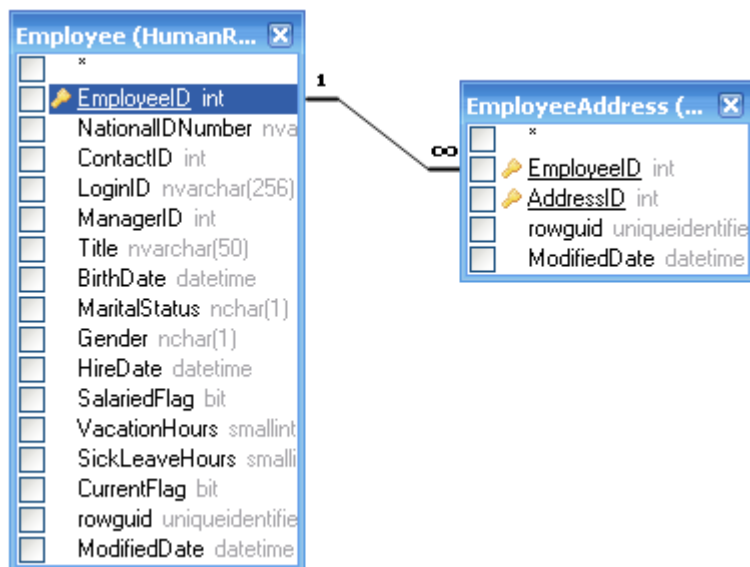
オブジェクトのエイリアスの設定

クエリ内のオブジェクトまたは派生テーブルのエイリアスを設定するには、オブジェクトをダブルクリックして、**[プロパティ]** を選択します。**[Datasource Properties]** ダイアログが表示されます。このダイアログには、他のサーバー固有のデータソース オプションが表示されることがありますが、**[エイリアス]** プロパティはすべてのデータベース サーバーに対して同一です。

テーブルの結合

外部キー関係で参照される 2 つのオブジェクトをクエリに追加すると、これらのオブジェクトは INNER JOIN で自動的に結合されます。JOIN 句をサポートしていないサーバーの場合、クエリビルダはクエリの WHERE 部に条件を追加します。

2 つのオブジェクトを手動で結合するには、オブジェクトをもう一方のオブジェクトにリンクするフィールドを選択して、もう一方のオブジェクトの対応するフィールドにドラッグします。ドラッグが完了すると、リンクされたフィールドを接続する線が表示されます。対応する関係がデータベース内に存在する場合は、キー カーディナリティ記号がリンクの両端に配置されます。



オブジェクト間のリンクを削除するには、リンクの線をダブルクリックして、**[削除]** を選択します。

結合タイプを変更するには、リンクの線をダブルクリックします。

出力フィールドの選択

クエリ出力フィールドの一覧にフィールドを追加するには、**[Query Building]** 領域のデータソースフィールド一覧でフィールド名の左側にあるボックスをオンにします。オブジェクトのすべてのフィールドを含めるには、データソースフィールド一覧でアスタリスク項目の左側にあるボックスをオンにします。**[Query Building]** 領域から **[Columns]** ウィンドウにフィールドをドラッグすることによっても、同じ結果が得られます。

クエリデータソースからフィールドを選択しない場合、作成されるクエリの SELECT リストにアスタリスク項目が追加されます ("Select * From ..."). これは、大部分のデータベースサーバーでは、列を指定しない SELECT クエリがエラーになるためです。フィールドを選択するか、クエリに出力式を追加した場合、アスタリスク項目は削除されます。

ヒント：フィールドを追加するもう 1 つの方法は、**[Columns]** ウィンドウの **[Expression]** 列のドロップダウンリストからフィールド名を選択することです。**[Columns]** ウィンドウの **[Expression]** 列に有効な式を入力することもできます。**[Columns]** ウィンドウに空の行を挿入するには、Alt キーと Insert キーを同時に押します。

[Columns] ウィンドウからフィールドを削除するには、**[Query Building]** 領域のフィールド名の左にあるチェックボックスをオフにするか、**[Columns]** ウィンドウで Alt キーと Delete キーを同時に押します。

行を上移動するには、Alt キーと Up キーを同時に押します。行を下移動するには、Alt キーと Down キーを同時に押します。

クエリの SELECT リストから式を削除するには、**[Output]** 列のチェックボックスをオフにします。

式のエイリアスを設定するには、**[Alias]** 列にエイリアスを入力します。エイリアスは、生成されるデータセットの列の見出しになります。

データセットのソート

生成されるデータセットをソートするには、**[Columns]** ウィンドウの **[Sort Type]** 列と **[Sort Order]** 列を使用します。**[Sort Type]** 列を使用すると、昇順または降順でソートできます。**[Sort Order]** 列を使用すると、複数のフィールドをソートする場合に、フィールドをソートする順序を設定できます。

フィールドによるソートを無効にするには、そのフィールドの **[Sort Type]** 列をオフにします。

条件の定義

条件を定義するには、**[Columns]** ウィンドウの **[Criteria]** 列と **[Or]** 列を使用します。これらの列に条件を入力する場合、式そのものは省略します。例えば、クエリに次の条件を指定するには、

```
WHERE (Field1 >= 10) AND (Field1 <= 20)
```

Field1 の式の **[Criteria]** セルに以下を入力します。

```
>= 10 AND <= 20
```

[Or] 列に配置された条件は、AND 演算子を使用して列ごとにグループ化されてから、OR 演算子を使用して WHERE (または HAVING) 句内に連結されます。例えば、以下に示す条件によって、次の SQL 文が生成されます。Field1 に対する条件が **[Criteria]** 列と **[Or]** 列の両方に配置されることに注意してください。

Output	Expression	Aggregate	Alias	Sort Type	Sort Order	Grouping	Criteria	Or...	Or...
<input checked="" type="checkbox"/>	Field1					<input type="checkbox"/>	= 10	= 10	
<input checked="" type="checkbox"/>	Field2					<input type="checkbox"/>	< 0	> 10	
<input type="checkbox"/>						<input type="checkbox"/>			

```
WHERE (Field1= 10) AND ((Field2 < 0) OR (Field2 > 10))
```

EXISTS 句など、一部の式は Boolean タイプであることがあります。その場合は、そのような式の [Criteria] 列に "= True" と入力するか、式の前に NOT 演算子を配置する場合は "= False" と入力する必要があります。

出力フィールドのグループ化

グループ化を使用するクエリを作成するには、[Grouping] チェック ボックスを使用してグループ化する式をマークします。

グループ化を使用するクエリでは、グループ化または集計する式のみを SELECT リストに表示できます。したがって、クエリビルダを使用すると、式をグループ化および集計する [Output] チェック ボックスをオンにできます。グループ化または集計関数が設定されていない列に対してこのチェック ボックスをオンにしようとした場合、[Grouping] チェック ボックスが自動的にオンになり、生成される SQL クエリの妥当性が維持されます。

[Columns] ウィンドウに [Grouping] チェック ボックスがオンの列が含まれる場合、[Criteria for] という新しい列がグリッドに表示されます。この列は、条件を式グループまたはその値に適用します。

例えば、Aggregate 関数 "Avg" が設定された列 "Quantity" がクエリにあり、[Criteria] 列に > 10 と入力します。[Criteria for] 列に "for groups" 値を設定すると、生成されるクエリには、平均数量が 10 より大きいグループのみが含まれ、クエリの HAVING 句に "Avg(Quantity) > 10" という条件が含まれるようになります。[Criteria for] 列に "for values" 値を設定すると、生成されるクエリは Quantity 値が 10 より大きいレコードに対してのみ Average 集計関数を計算し、クエリの WHERE 句に "Quantity > 10" という条件が含まれるようになります。

SQL クエリ プロパティの定義

[Query Building] 領域のコンテキスト ポップアップ メニューを使用して、データベース サーバーに固有のオプションを定義できます。

Read From File

Read from File ステージでは、ジョブまたはサブフローの入力ファイルを指定します。これをサービスで使うことはできません。

注：データフローの入力として XML ファイルを使いたい場合は、**Read from File** ではなく **Read from XML** ステージを使用してください。入力として可変フォーマット ファイルを使いたい場合は、**Read from Variable Format File** を使用してください。

前提条件: ファイルシステム接続タイプ (FTP、クラウド、Amazon AWS S3、HDFS など) からファイルを読み込むには、以下の手順を実行します。

1. **Management Console** または **Metadata Insights** を用いてこれらのファイルサーバーへの接続を作成します。詳細については、「[接続の定義 \(15ページ\)](#)」セクションを参照してください。
2. **[ファイル プロパティ]** タブの **[ファイル名]** フィールドを使用してファイルを選択します (以下を参照)。

[ファイル プロパティ] タブ

フィールド名	説明
サーバ名	入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。

フィールド名	説明
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを選択します。</p> <p>ワイルドカード文字を使用して、ディレクトリ内の複数のファイルからデータを読み込むことができます。サポートされているワイルドカード文字は、* と ? です。例えば、*.csv と指定して、ディレクトリ内にある、拡張子が .csv のファイルをすべて読み込むことができます。複数のファイルを正常に読み込むには、各ファイルが同じレイアウト (同じ位置に同じフィールド) を持つ必要があります。[フィールド] タブで指定したレイアウトに一致しないレコードは、形式に誤りのあるレコードとして扱われます。</p> <p>HDFS ファイル サーバーからのファイルの読み込みでサポートされる圧縮形式を次に示します。</p> <ol style="list-style-type: none"> 1. GZIP (.gz) 2. BZIP2 (.bz2) <p>注：ファイルの拡張子は、そのファイルの解凍に使用される圧縮形式を示します。</p> <p>重要： なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
レコードタイプ	<p>ファイル内のレコードのフォーマット。次のいずれかを選択します。</p> <p>行順次 ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキストファイル。</p> <p>固定長 ファイル内の各レコードの長さ (文字数) が一定で、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキストファイル。</p> <p>区切り記号付き ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドがカンマ (,) などの特定の文字で区切られているテキストファイル。</p>

フィールド名	説明
文字エンコーディング	テキスト ファイルのエンコーディング。次のいずれかを選択します。 CP1252 このエンコーディングは Windows-1252 文字セット、または単純に Windows 文字セットとも呼ばれています。これは ISO-8859-1 の上位クラスであり、128 ~ 159 のコード範囲を使用して、ISO-8859-1 文字セットに含まれていない追加の文字を表示します。 UTF-8 すべての Unicode 文字をサポートし、かつ ASCII との下位互換性があります。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 UTF-16 すべての Unicode 文字をサポートします。しかし、ASCII との下位互換性はありません。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 US-ASCII 英語のアルファベット順に従う文字エンコーディング。 UTF-16BE ビッグエンディアン UTF-16 エンコーディング (下位アドレスが上位バイトとなるようにシリアル化)。 UTF-16LE リトルエンディアン UTF-16 エンコーディング (下位アドレスが下位バイトとなるようにシリアル化)。 ISO-8859-1 主として西ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-1 と呼ばれます。 ISO-8859-3 主として南ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-3 と呼ばれます。 ISO-8859-9 主としてトルコ語で使われる ASCII 文字エンコーディング。Latin-5 と呼ばれます。 CP850 西ヨーロッパの言語を書くための ASCII コード ページ。 CP500 西ヨーロッパの言語を書くための EBCDIC コード ページ。 Shift_JIS 日本語のための文字エンコーディング。 MS932 NEC 特殊文字、NEC 選定 IBM 拡張文字、IBM 拡張文字を含めた Microsoft の拡張版 Shift_JIS 文字コード。 CP1047 Latin-1 文字セット全体を含む EBCDIC コード ページ。

フィールド名	説明
フィールド区切り文字	<p>区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは () 記号がフィールド区切り文字として使われています。</p> <pre>7200 13TH ST MIAMI FL 33144</pre> <p>フィールド区切り文字として定義できるのは次の文字です。</p> <ul style="list-style-type: none">• スペース• タブ• カンマ• ピリオド (.)• セミコロン• パイプ () <p>これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。</p>

Text qualifier	<p>区切り記号付きファイル内のテキスト値を囲むのに使用する文字。例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。</p> <pre>"7200 13TH ST" "MIAMI" "FL" "33144"</pre> <p>テキスト修飾子として定義できるのは次の文字です。</p> <ul style="list-style-type: none">• 一重引用符 (')• 二重引用符 (") <p>これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。</p>
----------------	--

フィールド名	説明
レコード区切り文字	<p>順次ファイルまたは区切り記号付きファイル内のレコードを区切るのに使用する文字を指定します。[デフォルトの EOL を使用] チェック ボックスをオンにすると、このフィールドは使用できません。</p> <p>使用できるレコード区切り文字の設定は次のとおりです。</p> <p>Unix (U+000A) 改行 (LF) 文字でレコードを区切ります。これは Unix システムの標準のレコード区切り文字です。</p> <p>Macintosh (U+000D) 復帰 (CR) 文字でレコードを区切ります。これは Macintosh システムの標準のレコード区切り文字です。</p> <p>Windows (U+000D U+000A) 復帰改行 (CR+LF) でレコードを区切ります。これは Windows システムの標準のレコード区切り文字です。</p> <p>これ以外の文字がレコード区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をレコード区切り文字として選択してください。</p>
デフォルトの EOL を使用	<p>Spectrum™ Technology Platform サーバーが実行されているオペレーティングシステムのデフォルトの行末 (EOL) 文字をファイルのレコード区切り文字として使用します。</p> <p>ファイルの EOL 文字がサーバーのオペレーティング システムで使われているデフォルトの EOL 文字と異なる場合は、このオプションをオンにしないでください。例えば、ファイルで Windows の EOL が使われていて、サーバーの動作プラットフォームが Linux の場合は、このオプションをオンにしないでください。代わりに、[レコード区切り文字] フィールドで [Windows] オプションを選択します。</p>
レコード長	<p>固定長ファイルでは、個々のレコードの文字数を指定します。</p> <p>行順次ファイルでは、ファイル内の最も長いレコードの長さ (文字数) を指定します。</p>
最初の行はヘッダ レコード	<p>区切り記号付きファイルの先頭レコードの内容がデータではなくヘッダ情報であるかどうかを指定します。</p> <p>次のファイル スニペットは、先頭レコードのヘッダ行の例です。</p> <pre>"AddressLine1" "City" "StateProvince" "PostalCode" "7200 13TH ST" "MIAMI" "FL" "33144" "One Global View" "Troy" "NY" 12180</pre>

フィールド名	説明
フィールド数が定義よりも少ないレコードを、形式誤りとみなす	区切り記号付きファイルのレコードのうち、 [フィールド] タブに定義されている数よりもフィールド数が少ないレコードを形式に誤りのあるレコードとみなします。
インポート	ファイルレイアウト定義、エンコーディング設定、およびソート オプションを設定ファイルからインポートします。設定ファイルは、同じ入力ファイル、あるいは操作しているファイルと同じレイアウトを持つファイルを使用した、別の Read from File ステージまたは Write to File ステージから設定をエクスポートすることによって作成されます。
エクスポート	<p>ファイルレイアウト定義、エンコーディング設定、およびソート オプションを設定ファイルに保存します。その後、同じ入力ファイル、または現在操作しているファイルと同じ特性を持つファイルを使用する他の Read from File ステージまたは Write to File ステージにこれらの設定をインポートできます。Job Executor で設定ファイルを使用して、実行時にファイル設定を指定することもできます。</p> <p>設定ファイルの詳細については、ファイル定義設定ファイル (178ページ) を参照してください。</p>

[フィールド] タブ

[フィールド] タブでは、ファイル内の各フィールドの名前と位置を定義します。また、固定長ファイルと行順次ファイルについては、さらにフィールドの長さを定義します。詳細については、以下を参照してください。

[区切り記号付き入力ファイルのフィールドの定義](#) (171ページ)

[行順次ファイルまたは固定長ファイルのフィールドの定義](#) (174ページ)

[ソート フィールド] タブ

[ソート フィールド] タブでは、データフローに送出される前の入力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。詳細については、「[入力レコードのソート](#) (176ページ)」を参照してください。

[実行時] タブ

フィールド名	説明
ファイル名	最初のタブで選択したファイル名が表示されます。

フィールド名	説明
開始レコード	レコードをデータフローに読み込むときファイルの先頭部分にあるレコードをスキップしたければ、読み込みたい最初のレコードを指定します。例えば、最初の 50 個のレコードをスキップする場合は 51 と指定します。これで 51 番目のレコードがデータフローに読み込まれる最初のレコードとなります。
すべてのレコード	[開始レコード] フィールドで指定したレコードからファイルの最後までレコードをすべて読み込む場合は、このオプションをオンにします。
最大レコード数	[開始レコード] フィールドで指定したレコードを起点にそこから一定の数のレコードを読み込む場合は、このオプションをオンにします。例えば、最初の 100 個のレコードを読み込みたい場合は、このオプションをオンにして 100 と入力します。

区切り記号付き入力ファイルのフィールドの定義

[フィールド] タブでは、ファイルの各フィールドの名前と位置、またファイルの種類によっては、さらにフィールドの長さも定義します。**[ファイル プロパティ]** タブで入力ファイルを定義したら、フィールドを定義できます。

入力ファイルにヘッダレコードが含まれていない場合、またはフィールドを手動で定義する場合は、**[フィールド]** タブで以下の手順に従います。

1. 入力ファイル内に既に存在するフィールドを定義するには、**[再生成]** をクリックします。その後、**[検出タイプ]** をクリックします。これにより、ファイルの最初の 50 個のレコードに基づいて、各フィールドのデータタイプが自動的に設定されます。
2. 出力にフィールドを追加するには、**[追加]** をクリックします。
3. **[名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。
4. **[タイプ]** フィールドで、データに対して数学的または日時に関する操作を行う予定がない場合は、データタイプを `string` のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータタイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータタイプに変換されます。

Spectrum™ Technology Platform では、以下のデータタイプがサポートされています。

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。
- bytearray** バイトの配列 (リスト)。
注: bytearray は REST サービスの入力としてはサポートされていません。
- 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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- list** 厳密に言えば、リストはデータタイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。
- ```
<Names>
 <Name>John Smith</Name>
 <Name>Ann Fowler</Name>
</Names>
```
- XML のリスト データ タイプが複数の値で構成される単純データタイプであるのに対し、Spectrum™ Technology Platform のリスト データタイプは XML の複合データタイプに似ているという点で、Spectrum™ Technology Platform のリスト データタイプは XML スキーマのリスト データタイプと異なることに注意してください。
- long** 正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
- 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。

- 日付、時間、または数値データタイプを選択した場合は、デフォルトの日付と時間の形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォル

トの形式は、**Management Console** のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、**Enterprise Designer** のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]** を選択したままにします。別の形式を指定するには、**[カスタム]** を選択し、以下の手順に従います。

注：日付/時間形式は、ファイルから読み込むデータを正確に反映する形式を選択することが重要です。例えば、ファイルに含まれている日付データの形式は月/日/年であるが、選択した日付形式が日/月/年であると、データフローで実行する日付計算（日付によるソートなど）に正確な日付が反映されません。また、レコードのタイプ変換が失敗することもあります。その場合、**Management Console** または **Enterprise Designer** のタイプ変換オプションで指定された失敗操作が有効になります。

- a) **[ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータタイプによって異なります。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。

選択した形式の例が **[形式]** フィールドの右側に表示されます。

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**（331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**（334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。

6. **[位置]** フィールドで、レコード内のこのフィールドの位置を入力します。

例えば、この入力ファイルで、**AddressLine1** は位置 1、**City** は位置 2、**StateProvince** は位置 3、**PostalCode** は位置 4 です。

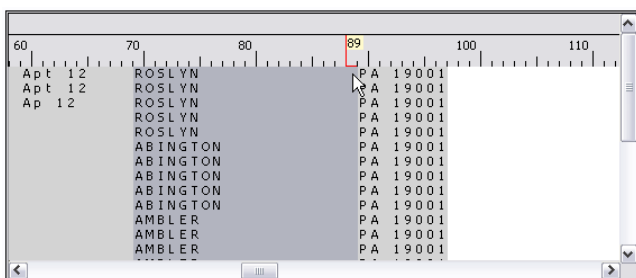
```
"AddressLine1"|"City"|"StateProvince"|"PostalCode"
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
"One Global View"|"Troy"|"NY"|"12180"
```

7. フィールドの値文字列の先頭と末尾から余分なスペース文字を削除するには、**[トリム]** チェックボックスをオンにします。

## 行順次ファイルまたは固定長ファイルのフィールドの定義

Read from File ステージで、**【フィールド】** タブを使い、ファイル内の各フィールドの名前と位置、またファイルの種類によっては、さらにフィールドの長さを定義します。**【ファイル プロパティ】** タブで入力ファイルを定義したら、フィールドを定義できます。

1. **【フィールド】** タブの **【プレビュー】** で、フィールドの先頭をクリックして左へドラッグすると、次のようにフィールドがハイライト表示されます。



2. **【名前】** フィールドで、追加したいフィールドを入力します。
3. **【タイプ】** フィールドでは、データに、数学的、または日付と時間の操作を行わない場合、データ タイプを文字列のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータ タイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータ タイプに変換されます。

Spectrum™ Technology Platform では、以下のデータ タイプがサポートされています。

**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。

**bytearray** バイトの配列 (リスト)。

注: bytearray は REST サービスの入力としてはサポートされていません。

**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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2-2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$  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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- list** 厳密に言えば、リストはデータタイプではありません。しかし、フィールドが階層データを含む場合、"リスト"フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。
- ```
<Names>
  <Name>John Smith</Name>
  <Name>Ann Fowler</Name>
</Names>
```
- XML のリスト データ タイプが複数の値で構成される単純データ タイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データ タイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。
- long** 正と負の整数を含む数値データタイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
- 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。

4. フィールドに packed decimal ストレージ形式を適用する場合は、**[パック 10 進数]** チェックボックスをオンにします。packed decimal タイプは 4 ビットを使用します。integer タイプの場合は 8 ビットです。

注: このストレージ形式は、行順ファイルと固定長ファイルの読み込み時に double、integer、および long のデータタイプを選択する場合のみ使用可能です。

5. 日付、時間、または数値データタイプを選択した場合は、デフォルトの日付と時間の形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォルトの形式は、Management Console のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、Enterprise Designer のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]** を選択したままにします。別の形式を指定するには、**[カスタム]** を選択し、以下の手順に従います。

注：日付/時間形式は、ファイルから読み込むデータを正確に反映する形式を選択することが重要です。例えば、ファイルに含まれている日付データの形式は月/日/年であるが、選択した日付形式が日/月/年であると、データフローで実行する日付計算 (日付によるソートなど) に正確な日付が反映されません。また、レコードのタイプ変換が失敗することもあります。その場合、Management Console または Enterprise Designer のタイプ変換オプションで指定された失敗操作が有効になります。

- a) **[ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータタイプによって異なります。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。

選択した形式の例が **[形式]** フィールドの右側に表示されます。

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](#) (331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](#) (334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。

6. **[開始位置]** フィールドと **[長さ]** フィールドは、ファイル プレビューでの選択操作に応じて自動的に設定されます。
7. フィールドの文字列の先頭と末尾から余分なスペース文字を削除するには、**[空白をトリム]** チェック ボックスを選択します。
8. **[OK]** をクリックします。

入力レコードのソート

Read from File ステージで、**[ソート フィールド]** タブを使い、データフローに送出される前の入力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。

1. **[ソート フィールド]** タブで、**[追加]** をクリックします。
2. **[フィールド名]** 列のドロップダウン矢印をクリックし、ソートに使うフィールドを選択します。選択できるフィールドは、この入力ファイルに定義されているフィールドによって異なります。
3. **[順序]** 列で、Ascending または Descending を選択します。

- ソートに使用するすべての入力フィールドを追加するまでこれを繰り返します。ソート順序を変更するには、移動するフィールドの行をハイライト表示して、**[上へ]**または**[下へ]**をクリックします。
- システムのデフォルトのソート パフォーマンス オプションの設定は、**Management Console**で行います。システムの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場合は、**[詳細設定]**をクリックします。**[詳細オプション]** ダイアログ ボックスには、次のソート パフォーマンス オプションがあります。

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。

$$\frac{(\text{NumberOfRecords} \times 2)}{\text{InMemoryRecordLimit}} = \text{NumberOfTempFiles}$$

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。 $(\text{InMemoryRecordLimit} \times \text{MaxNumberOfTempFiles} \div 2) \geq \text{TotalNumberOfRecords}$

ファイル定義設定ファイル

ファイル定義設定ファイルには、Read from File ステージまたは Write to File ステージからエクスポートされたファイルレイアウト、エンコーディング、およびソートオプションが含まれます。ファイル定義設定ファイルを Read from File または Write to File にインポートして、そのステージのオプションを手動で指定する代わりに、すばやく設定することができます。

ファイル定義設定ファイルを作成する最も簡単な方法は、Read from File または Write to File を使用してファイル設定を指定し、**[エクスポート]** ボタンをクリックして、ファイル定義設定ファイルを生成することです。

参考として、ファイル定義設定ファイルのスキーマを以下に示します。XML ファイル内の各要素はタイプを持ちます。そのタイプが文字列または整数以外の場合、許容できる値が表示されています。これらの値は、そのステージのダイアログ ボックスのオプションに直接対応しています。例えば、FileTypeEnum 要素は [ファイル プロパティ] タブの [レコード タイプ] フィールドに対応しており、スキーマに、linesequential、fixedwidth、delimited の 3 つの値が表示されます。

注： LineSeparator フィールド、FieldSeparator フィールド、または TextQualifier フィールドに対して "custom" と入力した場合、対応するカスタム要素 (例えば "CustomLineSeparator"、"CustomFieldSeparator"、"CustomTextQualifier" など) も使用する文字または文字列を表す 16 進数とともに含める必要があります。

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="FileSchema" nillable="true" type="FileSchema"/>
  <xs:complexType name="FileSchema">
    <xs:sequence>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        default="linesequential"
        name="Type"
        type="FileTypeEnum"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        default="UTF-8" name="Encoding" type="xs:string"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        name="RecordLength"
        type="xs:int"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
```

```
        default="default"
        name="LineSeparator"
        type="LineSeparatorEnum"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="CustomLineSeparator"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="comma"
  name="FieldSeparator"
  type="FieldSeparatorEnum"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="CustomFieldSeparator"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="none"
  name="TextQualifier"
  type="TextQualifierEnum"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="CustomTextQualifier"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="false"
  name="HasHeader"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="true"
  name="EnforceColumnCount"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Fields"
  type="ArrayOfFieldSchema"/>
</xs:sequence>
</xs:complexType>
<xs:simpleType name="FileTypeEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="linesequential"/>
    <xs:enumeration value="fixedwidth"/>
  </xs:restriction>
</xs:simpleType>
```

```
    <xs:enumeration value="delimited"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="LineSeparatorEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="default"/>
    <xs:enumeration value="windows"/>
    <xs:enumeration value="unix"/>
    <xs:enumeration value="mac"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="FieldSeparatorEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="comma"/>
    <xs:enumeration value="tab"/>
    <xs:enumeration value="space"/>
    <xs:enumeration value="semicolon"/>
    <xs:enumeration value="period"/>
    <xs:enumeration value="pipe"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="TextQualifierEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="none"/>
    <xs:enumeration value="single"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="ArrayOfFieldSchema">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="unbounded"
      name="Field"
      nillable="true"
      type="FieldSchema"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="FieldSchema">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Name"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      default="string"
      name="Type"
```

```
        type="xs:string"/>
<xs:element
  minOccurs="1"
  maxOccurs="1"
  name="Position"
  type="xs:int"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Length"
  type="xs:int"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="false"
  name="Trim"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Locale"
  type="Locale"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Pattern"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="none"
  name="Order"
  type="SortOrderEnum"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Locale">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Country"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Language"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Variant"
      type="xs:string"/>
  </xs:sequence>
```

```
</xs:complexType>
<xs:simpleType name="SortOrderEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="none"/>
    <xs:enumeration value="ascending"/>
    <xs:enumeration value="descending"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

データフロー オプションの設定

以下では、**Read from File** ステージの実行時オプションをサポートするようにデータフローを設定する手順について説明します。

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. **[Read from File]** ステージを展開します。

以下のデータフロー オプションがエクスポートされています。

1. 文字エンコード
 2. Field Separator
 3. Text Qualifier
 4. レコード長
 5. 最初の行はヘッダ レコード
 6. 開始レコード
 7. 最大レコード数
6. 選択した **Read from File** オプション名が、**[オプション名]** フィールドと **[オプション ラベル]** フィールドに表示されます。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。
 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、**[選択ステージ]** オプションを選択します。

9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **[OK]** をクリックします。
12. 必要に応じて、オプションの追加を続けます。
13. 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。
14. データフローを保存してエクスポートします。

データフロー オプションのルール

1. 文字エンコーディング: 基盤の JVM で有効なすべての種類のエンコーディングが使用できます。このオプションを空にすることはできません。
2. フィールド区切り文字: 任意の 1 文字が区切り文字として使用できます。現時点では、16 進数や空白はサポートされていません。
3. *Text Qualifier*: 任意の 1 文字が修飾子として使用できます。16 進数はサポートされていません。
4. レコード長: 整数値のみが指定できます。空白や数値以外にすることはできません。
5. 開始レコード: 整数値のみが指定できます。数値以外にすることはできません。
6. 最大レコード数: 整数値のみが指定できます。数値以外にすることはできません。
7. 最初の行はヘッダレコード: true または false の boolean 値のみが指定できます。空白にすることはできません。

Read from Hadoop Sequence File

Read from Hadoop Sequence File ステージでは、シーケンシャル ファイルからデータが入力としてデータフローに読み込まれます。シーケンシャル ファイルは、キー/値ペアで構成されるフラット ファイルです。詳細については、

「<https://netjs.blogspot.com/2018/06/how-read-and-write-sequencefile-hadoop.html>」を参照してください。

注： Read from Hadoop Sequence File ステージは、**Hadoop 分散ファイル システム (HDFS)** 上にある区切り記号付きの未圧縮シーケンシャルファイルのみをサポートします。

関連するタスク:

Hadoop への接続 (31ページ) : Hadoop システム上にあるファイルを読み取ったり、Hadoop システム上のファイルに書き込むためには、Hadoop ファイル サーバーへの接続を作成する必要があります。それが済むと、その接続を保存するための名前がサーバー名として表示されます。

[ファイルプロパティ] タブ

フィールド	説明
サーバー	<p>[ファイル名] フィールドで指定したファイルが Hadoop システム上にあることを示します。</p> <p>注： Hadoop ファイル サーバーにあるファイルを使用する前に、Hadoop ファイルサーバーへの接続を作成する必要があります。接続の作成の詳細については、Hadoop への接続 (31ページ) を参照してください。</p> <p>Hadoop システム上のファイルを選択すると、そのサーバー名が、ファイルサーバーの作成時に指定する名前になります。</p>
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを選択します。</p>
フィールド区切り文字	<p>区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは () 記号がフィールド区切り文字として使われています。</p> <pre>7200 13TH ST MIAMI FL 33144</pre> <p>フィールド区切り文字として定義できるのは次の文字です。</p> <ul style="list-style-type: none"> • スペース • タブ • カンマ • ピリオド (.) • セミコロン • パイプ () <p>これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。</p>

フィールド	説明
Text qualifier	<p>区切り記号付きファイル内のテキスト値を囲むのに使用する文字。</p> <p>例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。</p> <pre>"7200 13TH ST" "MIAMI" "FL" "33144"</pre> <p>テキスト修飾子として定義できるのは次の文字です。</p> <ul style="list-style-type: none"> 一重引用符 (') 二重引用符 (") <p>これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。</p>

[フィールド] タブ

[フィールド] タブでは、ファイルの各フィールドの名前、位置、およびタイプを定義します。詳細については、「[入力シーケンシャルファイルのフィールドの定義 \(185ページ\)](#)」を参照してください。

[ソート フィールド] タブ

[ソート フィールド] タブでは、データフローに送出される前の入力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。詳細については、「[入力レコードのソート \(186ページ\)](#)」を参照してください。

[フィルタ] タブ

[フィールド] タブでは、データフローに送出される前の入力レコードのソートに使うフィールドを定義します。詳細については、「[入力レコードをフィルタリング \(187ページ\)](#)」を参照してください。

入力シーケンシャル ファイルのフィールドの定義

Read from Hadoop Sequence File ステージで、**[フィールド]** タブに、ファイル内のフィールドの名前、位置、タイプを定義します。**[ファイル プロパティ]** タブで入力ファイルを定義したら、フィールドを定義できます。

[フィールド] タブでは、ファイルの各フィールドの名前と位置、またファイルの種類によっては、さらにフィールドの長さも定義します。**[ファイル プロパティ]** タブで入力ファイルを定義したら、フィールドを定義できます。

- 出力にフィールドを追加するには、**[追加]** をクリックします。
- [タイプ]** フィールドで、データに対して数学的な操作を行う予定がない場合は、データ タイプを文字列のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータタイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータタイプに変換されます。

このステージでは、以下のデータタイプがサポートされています。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
long	正と負の整数を含む数値データタイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
string	文字シーケンス。

- [名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。

入力レコードのソート

Read from Hadoop Sequence File ステージで、**[ソート フィールド]** タブを使い、データフローに送出される前の入力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。

- Read from Hadoop Sequence File で、**[ソート フィールド]** タブをクリックします。
- [ソート フィールド]** タブで、**[追加]** をクリックします。
- [フィールド名]** 列のドロップダウン矢印をクリックし、ソートに使うフィールドを選択します。選択できるフィールドは、この入力ファイルに定義されているフィールドによって異なります。
- [順序]** 列で、**Ascending** または **Descending** を選択します。
- ソートに使用するすべての入力フィールドを追加するまでこれを繰り返します。ソート順序を変更するには、移動するフィールドの行をハイライト表示して、**[上へ]** または **[下へ]** をクリックします。

入力レコードをフィルタリング

Read from Hadoop Sequence File ステージで、**【フィルタ】** タブを使い、データフローに送出される前の入力レコードのフィルタに使うフィールドを定義します。フィルタを行うかどうかはオプションです。

1. Read from Hadoop Sequence File で、**【フィルタ】** タブをクリックします。
2. **【式の結合方法】** フィールドで、すべての式が真として評価される場合にのみ、このポートにレコードを送信するときは **【すべて】** を選択し、1つ以上の式が真として評価される場合に、このポートにレコードを送信するときは **【いずれか】** を選択します。
3. **【追加】** をクリックし、テストするフィールド、演算子、および値を指定します。演算子を次の表に示します。

演算子	説明
等しい	フィールドの値が指定された値とマッチするかどうかを確認します。
が次の値と異なる	フィールドの値が指定された値とマッチしないかどうかを確認します。
が次の値より大きい	フィールドの数値が指定された値よりも大きいかどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
が次の値以上	フィールドの数値が指定された値以上かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
が次の値より小さい	フィールドの数値が指定された値未満かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
が次の値以下	フィールドの数値が指定された値以下かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
が NULL である	フィールドが NULL 値かどうかを確認します。
が NULL でない	フィールドが NULL 値でないかどうかを確認します。

4. 必要に応じて、[トリム]オプションを選択します。このオプションは、フィールドのデータのフィルタリングを行う前に、フィールドの値の前後の空白文字をトリムします。
5. フィルタリングに使用するすべての入力フィールドを追加するまでこれを繰り返します。

Read from Hive File

Read from Hive File ステージは、ORC、RC、Parquet、Avro のいずれかのフォーマットのファイルからデータを読み取ります。

関連するタスク:

Hadoop への接続 (31ページ) : **Read from Hive File** ステージを使用できるようにするには、Hadoop ファイルサーバーへの接続を作成する必要があります。それが済むと、その接続を保存するための名前がサーバー名として表示されます。

[ファイルプロパティ] タブ

フィールド	説明
サーバー	<p>[ファイル名] フィールドで指定したファイルが Hadoop システム上にあることを示します。</p> <p>注: Hadoop ファイルサーバーにあるファイルを使用する前に、Hadoop ファイルサーバーへの接続を作成する必要があります。接続の作成の詳細については、Hadoop への接続 (31ページ) を参照してください。</p> <p>Hadoop システム上のファイルを選択すると、そのサーバー名が、ファイルサーバーの作成時に指定する名前になります。</p>
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、必要なファイルを参照して選択します。ただし、必要であればスキーマの列の名前は変更できます。ファイルを選択すると、最初の 50 件のレコードが [プレビュー] グリッドに取得されます。</p> <p>注: 入力ファイルのスキーマは、正しい場所を参照し、ファイルを選択した時点ですぐにインポートされます。このインポートされるスキーマは編集できません。</p>

フィールド	説明
ファイル タイプ:	読み込まれるファイルのタイプを選択します。 <ul style="list-style-type: none"> • ORC • RC • Parquet • Avro

注：RC ファイルの場合、プレビューを生成するには、**[フィールド]** タブでスキーマを定義した後に、**[ファイル プロパティ]** タブの**[プレビュー]** をクリックします。

[フィールド] タブ

[フィールド] タブでは、フィールドのユーザ定義名のほか、フィールドの名前、データタイプ、および位置を入力ファイルに保存されるとおりに定義します。詳細については、「[Hive ファイル読み取りのためのフィールドの定義 \(189ページ\)](#)」を参照してください。

Hive ファイル読み取りのためのフィールドの定義

Read from Hive File ステージの **[フィールド]** タブには、スキーマ名、データタイプ、位置、およびファイル内で指定されているフィールドの名前が表示されます。

1. **[再生成]** をクリックします。

ORC ファイル、Avro ファイル、Parquet ファイルの場合は、これによって、既存ファイルのメタデータに基づくスキーマが生成されます。RC ファイルの場合は、**[プレビュー]** をクリックする前に追加したすべてのフィールドがクリアされます。

グリッドには、**[名前]**、**[タイプ]**、**[ステージフィールド]**、**[含める]** の各列が表示されます。

[名前] 列には、ファイルのヘッダ レコードから取得されたフィールド名が表示されます。

[タイプ] 列には、ファイルの各フィールドのデータタイプが表示されます。

このステージでは、以下のデータ タイプがサポートされています。

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。

注：Spectrum の `datetime` データタイプは、Hive ファイルの `timestamp` データタイプにマッピングされます。

- 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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ となります。
- 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。
- 注：RC ファイル、Avro ファイル、Parquet Hive ファイルの場合、入力ファイル内の decimal データタイプのフィールドは、bigdecimal データタイプに変換されます。
- long** 正と負の整数を含む数値データタイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
- 注：Spectrum の long データタイプは、Hive ファイルの `bigint` データタイプにマッピングされます。
- integer** 正と負の整数を含む数値データタイプ。値の範囲は、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- 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、 $2^{-149} \sim (2 \cdot 2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ となります。
- string** 文字シーケンス。

注：RC ファイルの場合、`smallint` データタイプと複合データタイプはサポートされません。

[位置] 列には、レコード内における各フィールドの開始位置が表示されます。

- [ステージフィールド]** 列では、各フィールドの既存のフィールド名を任意の名前に編集できます。デフォルトで、この列にはファイルから読み込まれたフィールド名が表示されます。
- [含める]** 列で、ステージの出力に含めるフィールドのチェックボックスを選択します。デフォルトで、この列はすべてのフィールドで選択されています。
- RC ファイルの場合は、以下のボタンを使用して、フィールドを追加または削除したり、出力における選択列の順序を変更したりできます。

オプション名	説明
追加	フィールドを出力に追加します。
変更	選択されているフィールドの名前とデータタイプを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されているフィールドの出力時の並び順を変更します。

注：この機能は、RC ファイルに対してのみ使用可能です。

5. **[OK]** をクリックします。

Read From HL7 File

Read From HL7 File ステージは、データフローへの入力としてテキスト ファイルから Health Level Seven (HL7) データを読み取ります。HL7 は、保健医療業界向けのメッセージ標準で、システム間でのデータ交換に使用されます。HL7 の詳細については、「<http://www.hl7.org>」を参照してください。

HL7 メッセージング形式

HL7 メッセージ内のデータは、以下のように階層的に構成されています。

```

message
  segment
    field
      component
        subcomponent

```

HL7 メッセージの各行がセグメントです。セグメントとは、フィールドを論理的にグループ化したものです。セグメント内の先頭 3 文字はセグメント タイプを識別します。上記のメッセージには、MSH (メッセージ ヘッダ)、PID (患者 ID)、2 つの NK1 (近親者)、および IN1 (保険) の 5 つのセグメントがあります。

各セグメントはフィールドで構成されます。フィールドには、セグメントの目的に関する情報が含まれ、例えば、IN1 (保険) セグメントには保険会社の名前が含まれます。フィールドは、一般的には (常にではない) | 文字で区切られます。

フィールドは、コンポーネントに分割できます。コンポーネントは、一般的には ^ 文字によって示されます。上記の例では、PID (患者 ID) セグメントには、姓 (LEVERKUHN)、名 (ADRIAN)、およびミドルネームのイニシャル (C) の 3 つから成る LEVERKUHN^ADRIAN^C を含む患者名フィールドがあります。コンポーネントは、サブコンポーネントに分割できます。サブコンポーネントは、一般的には & 文字によって示されます。

HL7 メッセージの例を以下に示します。

```
MSH|^~\&||.||||199908180016||ADT^A04|ADT.1.1698593|P|2.7
PID|1||000395122||LEVERKUHN^ADRIAN^C||19880517180606|M|||6 66TH AVE
NE^^WEIMAR^DL^98052|| (157) 983-3296|||S||12354768|87654321
NK1|1|TALLIS^THOMAS^C|GRANDFATHER|12914 SPEM
ST^^ALIUM^IN^98052| (157) 883-6176
NK1|2|WEBERN^ANTON|SON|12 STRASSE MUSIK^^VIENNA^AUS^11212| (123) 456-7890
IN1|1|PRE2||LIFE PRUDENT BUYER|PO BOX
23523^WELLINGTON^ON^98111|||19601|||THOMAS^JAMES^M|F|ZKA535529776
```

注：与えられたサンプル テキストを使用して HL7 ファイルを作成するには、以下の手順を実行します。

1. 任意のテキスト編集ソフトウェアを使用して、サンプル テキストを新規ドキュメントにコピー アンド ペーストします。
2. 内容を必要に応じて変更します。
3. テキストの EOL (行末) を表示するように設定します。テキスト エディターで、**[表示] > [記号を表示] > [行末を表示]** の順に選択します。
4. EOL (行末) 変換フォーマットを CR (復帰) に変更します。テキスト エディターで、**[編集] > [改行コードを変更] > [Mac フォーマット (CR) に変換]** の順に選択します。
5. フォーマットをこのように変更したら、HL7 ファイルを保存します。

[ファイル プロパティ] タブ

フィールド名	説明
サーバ名	入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。

フィールド名	説明
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン(...)をクリックして、必要なファイルにアクセスします。</p> <p>注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
HL7 バージョン	<p>指定したファイルで使用される HL7 標準のバージョン。例えば、"2.7" は、ファイルで HL7 バージョン 2.7 が使用されることを意味します。HL7 バージョンは、MSH セグメントの最後のフィールドに示されます。</p>

フィールド名	説明
文字エンコーディング	テキスト ファイルのエンコーディング。次のいずれかを選択します。 CP1252 このエンコーディングは Windows-1252 文字セット、または単純に Windows 文字セットとも呼ばれています。これは ISO-8859-1 の上位クラスであり、128 ~ 159 のコード範囲を使用して、ISO-8859-1 文字セットに含まれていない追加の文字を表示します。 UTF-8 すべての Unicode 文字をサポートし、かつ ASCII との下位互換性があります。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 UTF-16 すべての Unicode 文字をサポートします。しかし、ASCII との下位互換性はありません。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 US-ASCII 英語のアルファベット順に従う文字エンコーディング。 UTF-16BE ビッグエンディアン UTF-16 エンコーディング (下位アドレスが上位バイトとなるようにシリアル化)。 UTF-16LE リトルエンディアン UTF-16 エンコーディング (下位アドレスが下位バイトとなるようにシリアル化)。 ISO-8859-1 主として西ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-1 とも呼ばれます。 ISO-8859-3 主として南ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-3 とも呼ばれます。 ISO-8859-9 主としてトルコ語で使われる ASCII 文字エンコーディング。Latin-5 とも呼ばれます。 CP850 西ヨーロッパの言語を書くための ASCII コード ページ。 CP500 西ヨーロッパの言語を書くための EBCDIC コード ページ。 Shift_JIS 日本語のための文字エンコーディング。 MS932 NEC 特殊文字、NEC 選定 IBM 拡張文字、IBM 拡張文字を含めた Microsoft の拡張版 Shift_JIS 文字コード。 CP1047 Latin-1 文字セット全体を含む EBCDIC コード ページ。

フィールド名

説明

Validate

これらのオプションは、ファイルが HL7 2.7 標準に準拠していることを確認するかどうかを指定します。ファイル内の任意のメッセージが検証に失敗すると、そのメッセージは形式に誤りのあるレコードとして処理され、ジョブに対して (Enterprise Designer の **【編集】** > **【ジョブ オプション】** で) またはシステムに対して (Management Console で) 指定した形式に誤りのあるレコードのオプションが有効になります。

必須フィールド 各セグメント、フィールド、コンポーネント、およびサブコンポーネントに HL7 2.7 標準の必須要素が含まれているかどうかを確認する場合は、このボックスをオンにします。

長さ 各要素が、HL7 2.7 標準に定義されている要素の最小および最大長要件を満たしているかどうかを確認する場合は、このボックスをオンにします。

フィールド名	説明
--------	----

例外を無視	
-------	--

フィールド名

説明

想定される位置にないセグメント、フィールド、コンポーネント、およびサブコンポーネントをメッセージに含めることができるようにする場合は、これらのオプションを選択します。想定される位置は、HL7 標準に定義されています。カスタム メッセージ タイプの場合は、Enterprise Designer の HL7 スキーマ管理ツールで定義します。

例えば、次のカスタム メッセージ スキーマがあるとします。

```
MSH
[PID]
{ZSS}
PV1
NK1
{[DG1]}
```

データは次のようになっています。

```
MSH|^~\&|Pharm|GenHosp|CIS|GenHosp|198807050000||RAS^O17|RAS1234|P|2.7
ZSS|100|abc
PID|1234||PATID1234^5^M11^ADT1^MR^GOOD HEALTH
HOSPITAL~123456789^^^USSSA^SS|
PV1||O|O/R|||0148^ADDISON, JAMES|0148^ADDISON, JAMES
NK1|Son|John
```

この場合、セグメント PID は ZSS セグメントの前にあるため、例外となります。

予期せぬ位置にある要素を含むメッセージは形式に誤りのあるレコードとして処理され、ジョブに対して (Enterprise Designer の **【編集】** > **【ジョブ オプション】**) またはシステムに対して (Management Console で) 指定した形式に誤りのあるレコードのオプションが有効になります。

デフォルトでは、できるだけ多くのレコードが正しく処理されるように、すべての **【例外を無視】** オプションが有効になります。

セグメント HL7 2.7 標準に定義されていないセグメントがメッセージに含まれることを許可する場合は、このボックスをオンにします。例外セグメントは無視され、メッセージ内のその他のセグメントが処理されます。

フィールド HL7 2.7 標準に定義されていないフィールドがセグメントに含まれることを許可する場合は、このボックスをオンにします。例外フィールドは無視され、セグメント内のその他のフィールドが処理されます。

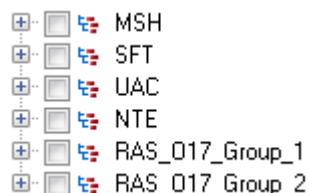
コンポーネン HL7 2.7 標準に定義されていないコンポーネントがフィールドに含まれることを許可する場合は、このボックスをオンにします。例外コンポーネントは無視され、フィールド内のその他のコンポーネントが処理されます。

フィールド名	説明
サブコンポーネント	HL7 2.7 標準に定義されていないサブコンポーネントがコンポーネントに含まれることを許可する場合は、このボックスをオンにします。コンポーネント内の例外サブコンポーネントは無視され、その他のサブコンポーネントが処理されます。

[フィールド] タブ

[フィールド] タブには、セグメント、フィールド、コンポーネント、およびサブコンポーネントが表示されます。[フィールド] タブを使用して、データフローに読み込むデータを選択します。

セグメント グループは、あるカテゴリのデータを含めるために一緒に使用されるセグメントのコレクションで、メッセージスキーマにおけるそのグループの位置を示す番号付与体系を使用して表示されます。各セグメントには、"Group_n" というラベルが付与されます。"n" は、メッセージスキーマにおけるグループの位置に対応する数字です。番号付与体系の仕組みを説明するため、次の例を考えてみましょう。

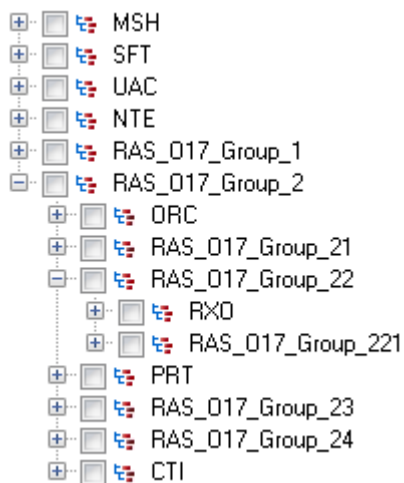


この例は、メッセージ RAS^017 のフィールド リストを示しています。このメッセージには、RAS_017_Group_1 と RAS_017_Group_2 という 2 つのセグメントグループがあります。"Group_1" セグメントグループは RAS^017 スキーマ内の最初のセグメントグループを指し、2 番目のグループ "Group_2" は RAS^017 スキーマ内に示される 2 番目のグループを指します。

"Group_1" および "Group_2" がどのセグメント グループを表しているかを判断するには、ドキュメント『HL7 バージョン 2.7 標準』のメッセージ RAS^017 についての説明を探してください。このドキュメントのコピーは、<http://www.hl7.org> からダウンロードできます。

メッセージの説明で、最初のグループ (RAS^017 の場合は PATIENT グループ) を探します。スキーマ内の 2 番目のグループは ORDER グループです。

1 つのセグメントグループの下にネストされているセグメントグループの場合は、グループ番号にさらに番号が付加されます。例えば、Group_21 は、2 番目のグループの下にネストされている最初のグループを表します。その下のサブグループには、Group_221 のように、さらに番号が付加されます。メッセージ RAS^017 の場合、これはセグメントグループ ORDER_DETAIL_SUPPLEMENT を表します。ネストされているグループの例を以下に示します。



次の表に、【フィールド】タブのコントロールの詳細を示します。

表 2 : 【フィールド】タブ

オプション名	説明
再生成	<p>このボタンをクリックすると、入力ファイルに含まれるメッセージ タイプのセグメント、フィールド、コンポーネント、およびサブコンポーネントが【フィールド】タブにすべて入力されます。入力ファイルにすべての要素が含まれているかどうかにかかわらず、HL7 スキーマに基づいてメッセージ タイプの要素がすべて表示されます。例えば、ファイルにRAS メッセージが含まれている場合、入力ファイルにセグメント、フィールド、コンポーネント、およびサブコンポーネントのすべてのデータが実際に含まれているかどうかに関係なく、RAS メッセージ全体のスキーマが表示されます。</p> <p>Enterprise Designer の HL7 スキーマ管理ツールを使用してカスタム要素を定義した場合は、その要素も一覧表示されます。</p>
すべて展開	<p>フィールド タブのすべての要素を展開すると、ファイルに含まれるメッセージ タイプのセグメント、フィールド、コンポーネント、およびサブコンポーネントをすべて表示できます。</p>
すべて折りたたむ	<p>ビュー内のすべてのノードを閉じて、セグメントのみを表示します。これを使用すると、ファイル内のメッセージ タイプのセグメントを容易に表示できます。この後、個々のセグメントを展開すれば、セグメント内のフィールド、コンポーネント、およびサブコンポーネントを表示できます。</p>

オプション名

説明

すべて選択

ファイルに含まれるすべてのメッセージタイプについて、セグメント、フィールド、コンポーネント、およびサブコンポーネントのデータフロー フィールドをすべて作成する場合は、このボックスをオンにします。

HL7 データのフラット化

HL7 データは階層的に構成されています。多くのステージではフラット形式のデータが必要とされるので、下流のステージでの住所検証やジオコーディングなどを使用できるようにするためにデータをフラット化することが必要となる場合があります。

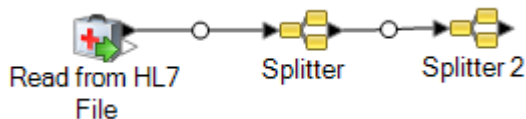
次の手順では、**Splitter** ステージを使用して HL7 データをフラット化する方法について説明します。

1. **Read from HL7 File** ステージをデータ フローに追加して設定します。
2. **Splitter** ステージを追加して **Read from HL7 File** に接続します。
3. フラット化するセグメント、フィールド、またはコンポーネントごとに 1つの **Splitter** ステージとなるように、必要に応じて **Splitter** ステージを追加します。

注：下流のステージで処理するデータのみをフラット化します。それ以外のデータは階層形式に残しておくことができます。例えば、住所データのみを処理する場合は、住所データのみをフラット化します。

4. すべての **Splitter** ステージを接続します。

データ フローは次のようになっているはずです。



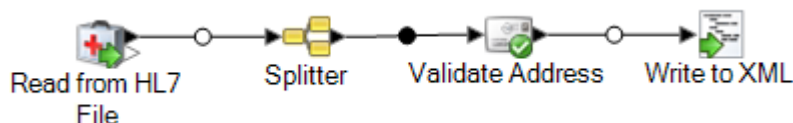
5. 最初の **Splitter** ステージをダブルクリックして、ステージのオプション を開きます。
6. **[分割位置]** フィールドで、フラット化するセグメント、フィールド、またはコンポーネントを選択します。
7. **[OK]** をクリックします。
8. 追加の各 **Splitter** ステージを設定し、各 **Splitter** の**[分割位置]** フィールドで異なるセグメント、フィールド、またはコンポーネントを選択します。
9. 必要に応じて、最後の **Splitter** の後にステージを追加してデータフローを完成させます。

例

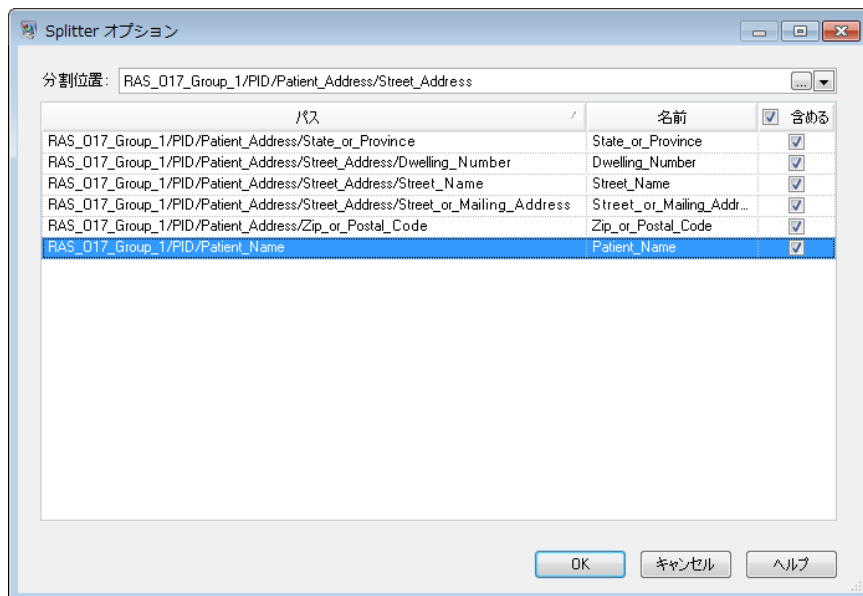
次のような HL7 データがあり、PID セグメントに含まれる住所を検証します。

```
MSH|^~\&||.||||199908180016||RAS^O17|ADT.1.1698594|P|2.7
PID|1||000395122||SMITH^JOHN^D||19880517180606|M|||One Global
View^^Troy^NY^12180|||(630)123-4567|||S||12354768|87654321
```

このためには、Validate Address ステージで処理できるように、住所データをフラットデータに変換する必要があります。したがって、次のように Splitter ステージに続けて Validate Address ステージを含むデータフローを作成します。



Splitter ステージは、PID/Patient_Address/Street_Address コンポーネントで分割するように設定され、それによって、このデータをフラットデータに変換します。



Splitter ステージを Validate Address ステージに接続するチャンネルは、フィールド名を Validate Address で必要となるフィールド名に変更します。つまり、Street_or_Mailing_Addres は AddressLine1 に、State_or_Province は StateProvince に、Zip_or_Postal_Code は PostalCode に変更されます。

この例では、出力は、次のデータを含む XML ファイルに書き込まれます。

```
<?xml version='1.0' encoding='UTF-8'?>
<XmlRoot xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<PatientInformation>
  <Confidence>95</Confidence>
  <RecordType>Normal</RecordType>
  <CountryLevel>A</CountryLevel>
  <ProcessedBy>USA</ProcessedBy>
  <MatchScore>0</MatchScore>
  <AddressLine1>1 Global Vw</AddressLine1>
  <City>Troy</City>
  <StateProvince>NY</StateProvince>
  <PostalCode>12180-8371</PostalCode>
  <PostalCode.Base>12180</PostalCode.Base>
  <PostalCode.AddOn>8371</PostalCode.AddOn>
  <Country>United States Of America</Country>
  <Patient_Name>
    <Family_Name>
      <Surname>SMITH</Surname>
    </Family_Name>
    <Given_Name>JOHN</Given_Name>

  <Second_and_Further_Given_Names_or_Initials_Thereof>
    D

  </Second_and_Further_Given_Names_or_Initials_Thereof>
</Patient_Name>
</PatientInformation>
</XmlRoot>
```

カスタム HL7 メッセージの追加

Read from HL7 File ステージでは、HL7 2.7 スキーマを使用してメッセージを検証します。ただし、HL7 データには、HL7 標準の一部ではないメッセージが含まれることがあります。Read from HL7 File ステージで、カスタマイズされた HL7 データを検証する場合は、カスタム HL7 スキーマを作成する必要があります。このトピックでは、HL7 スキーマ管理ツールを使用して、カスタム HL7 スキーマを作成する方法について説明します。HL7 の詳細については、www.hl7.org を参照してください。

1. Enterprise Designer で、**[ツール] > [HL7 スキーマ管理]** の順に選択します。
サポートされているメッセージの一覧を含む HL7 スキーマ管理ウィンドウが開きます。これらのメッセージは、HL7 によってあらかじめ定義されています。
2. **[HL7 スキーマ管理]** ウィンドウで、**[追加]** をクリックします。
3. **[メッセージ タイプ]** フィールドに、カスタム HL7 メッセージのタイプを指定します。

メッセージタイプは、健康に関連するどのような情報をこのメッセージで提供するかを示します。例えば、ADT (Admin Discharge Transfer) メッセージタイプは医療機関内で患者の状態を交換するために使用し、ORU (Observation Result) メッセージタイプは LIS (Lab Information System) から HIS (Hospital Information System) に所見と結果を転送するために使用します。

4. **【トリガー イベント】** フィールドに、イベント コードを指定します。

トリガー イベントは、通信とメッセージの送信を開始する実世界のイベントです。メッセージタイプとトリガー イベントは、メッセージの MSH-9 フィールドにあります。例えば、MSH-9 フィールドに値 ADT^A01 が含まれているとします。ADT が HL7 のメッセージタイプで、A01 がトリガー イベントです。

5. **【説明】** フィールドに、カスタム HL7 メッセージの説明を入力します。

このフィールドは、メッセージタイプの詳細を理解するのに役立ちます。例えば、XYZ メッセージタイプを追加する場合、このメッセージタイプは医療機関内で患者の状態を交換するために使用するという説明を入力できます。

新規作成されたメッセージが **【定義】** の下に表示されます。[+] 記号をクリックして、メッセージを展開します。MSH セグメントが自動的に追加されていることがわかります

6. 既存のセグメントをメッセージに追加するには

- a) **【セグメントの選択】** をクリックします。
- b) メッセージに追加するセグメントを選択し、**【OK】** をクリックします。

選択したセグメントのスキーマが **【セグメント スキーマ】** グリッドに表示され、チェックしたメッセージがメッセージスキーマに追加されます。

7. カスタム セグメントをメッセージに追加するには

- a) **【セグメントの選択】** をクリックします。
- b) **【セグメントの追加】** をクリックします。
- c) **【名前】** フィールドに、セグメントの名前を指定し、**【OK】** をクリックします。

新しく追加されたセグメントが **【サポートされているセグメント】** 一覧の下部に表示されます。

- d) 追加されたカスタムセグメントを選択し、**【フィールドの追加】** ボタンをクリックします。
- e) **【名前】** フィールドに、選択したセグメントのフィールド名を指定します。

例えば、PID (患者情報) セグメントには、患者ID、患者名、患者住所、国コードなどのフィールドが含まれます。

- f) **【タイプ】** フィールドで、適切なデータタイプを選択します。

HL7 データ タイプは、フィールドに含めることができるデータの種類を定義するもので、HL7 メッセージ構造全体で使用されます。例えば、文字列は ST、テキストデータは TX、書式を整えたデータは FT となります。

- g) **[規範長]** フィールドに、m..n という形式で、フィールドの最小長と最大長を指定します。x,y,z という形式で、フィールドの長さとして有効な値のリストを指定することもできます。

例えば、1..3 は項目の長さが 1、2、または 3 のいずれかであることを意味し、1, 3, 4 は項目の長さが 1、3、または 4 であるが 2 ではないことを意味します。1、3、4 以外の値は無効として扱われます。

- h) **[オプション]** フィールドに、フィールドがオプションか必須かを指定します。

O

フィールドはオプションです。

R

フィールドは必須です。

- i) セグメント内にフィールドが複数回出現することを許可する場合は、**[反復]** ボックスをオンにして、**[反復]** フィールドに、そのフィールドを使用できる回数を指定します。

例えば、値 3 は、そのフィールドが 3 回出現可能であることを意味します。指定しない場合、フィールドの出現は 1 回だけで、繰り返されないことを意味します。

8. **[OK]** をクリックします。

セグメント プロパティの **[オプション]** および **[繰り返し]** オプションを選択することもできます。

9. 選択したセグメントをオプションにするには **[オプション]** を、選択したセグメントのメッセージ内での繰り返しを許可するには **[繰り返し]** を選択します。

10. **[OK]** をクリックします。

新しく追加したメッセージが一覧の下部に表示されます。

Read from NoSQL DB

Read from NoSQL DB ステージでは、データフローへの入力として、データベース テーブルからデータを読み込みます。このステージは、MongoDB と Couchbase のデータベース タイプをサポートします。

[全般] タブ

フィールド名	説明
接続	<p>ドロップダウンリストから必要なデータベース接続を選択します。表示されるオプションは、Management Console に定義されている接続によって異なります。</p> <p>新しい接続を追加するには、NoSQL への接続 (62ページ) を参照してください。</p> <p>既存の接続を変更するには、Management Console の [データ ソース] ページの接続リストから接続を選択して開き、必要な更新を行い、[保存] ボタンをクリックします。</p>
テーブル/ビュー	<p>クエリの実行対象とするデータベース内のコレクションまたはビューを指定します。</p> <p>注：ユーザ インターフェイスでは "テーブル/ビュー" という語が使用されていますが、MongoDB ではこれを "コレクション" と呼び、Couchbase ではこれを "ビュー" と呼びます。</p>
スキーマ ファイル	<p>参照ボタン (...) をクリックして JSON スキーマ ファイルを選択します。このファイルはオプションです。[フィールド] タブのフィールドは、スキーマ ファイルまたはデータベース テーブル/ビューのいずれかを使用して再生成できます。</p> <p>選択されたファイル パスを非選択にするには、[クリア] をクリックします。</p> <p>注：スキーマファイルが選択されている場合は、フィールドは必ずスキーマ ファイルを用いて生成されます。</p>

フィールド名	説明
説明：	<p>必要なフィルタ条件があれば入力します。特定のレコードを取得するために MongoDB 構文を使用できます。フィルタ条件が不要な場合はフィールドを空白のままにします。</p> <p><i>equal to</i> 演算子を使用する句の構文を以下に示します。</p> <pre data-bbox="552 493 1412 556">{ "<column name>" : "<filter value>" }</pre> <p>複数の句を必要な演算子を使って結合できます。where 句でサポートされる演算子については、「http://docs.mongodb.org/manual/reference/operator/query/」を参照してください。</p> <p>例えば、customer_name 列の値が John で、customer_age 列の値が 45 以上であるレコードを取得するには、次のように入力します。</p> <pre data-bbox="552 777 1412 861">{ \$and: [{"customer_name": "John"}, { \$gte: ["customer_age", "45"] }] }</pre> <p>重要： このフィールドにキーワードの where を含めないように注意してください。</p> <p>注： 現在、このフィールドは MongoDB 接続を選択した場合にだけ表示されます。</p>
未指定のフィールドを無視	<p>このオプションが選択されている場合、スキーマで定義されていても、実際のレコードに存在しないフィールドは、次のステージに引き渡されません。</p> <p>注： このオプションを有効にしない場合は、データベース テーブルまたはビューに存在しないフィールドが追加され、値が NULL であるとして処理されます。</p>
プレビュー	<p>選択したテーブルのレコードを表示します。</p> <p>注： MongoDB データ ソースでは、[Where] フィールドに 1 つ以上の where 句が入力されている場合に [プレビュー] をクリックすると、フィルタリングされたレコードが表示されます。where 句が入力されていない場合は、すべてのレコードがプレビューされます。</p> <p>注： Couchbase データ ソースでは、[プレビュー] をクリックすると、キーを格納する追加の _id フィールドも表示されます。レコードに既に _id フィールドが存在する場合は、フィールドのプレビュー時に追加の _id フィールドによって上書きされます。</p>
すべて展開	<p>プレビュー ツリーの項目を展開します。</p>

フィールド名	説明
すべて折りたたむ	プレビュー ツリーの項目を折りたたみます。

[フィールド] タブ

[フィールド] タブでは、次のステージに渡すデータを選択できます。詳細については、[NoSQL データベースのフィールドの定義 \(207ページ\)](#)

NoSQL データベースのフィールドの定義

[フィールド] タブに、NoSQL DB/スキーマファイルに定義されたフィールドとタイプが表示されます。

1. [フィールド] タブで、[フィールドの再生成] をクリックします。

これによって、先頭 50 件のレコードに基づく集計データが生成されます。データは、`Fieldname (datatype)` という形式で表示されます。

注：スキーマファイルを参照する場合は、スキーマファイルを使用してフィールドが生成され、テーブルやビューは無視されます。スキーマファイルのリセットするには、**[クリア]** をクリックします。

注：Couchbase DB からデータを読み込む際には、各レコードのキーも読み込まれます。このキーはフィールドの再生成時に、追加された `_id` フィールドを用いてレコードの一部として保存され、次のステージに送信されるデータにも含められます。レコードに既に `_id` フィールドが存在する場合は、フィールドの再生成時に、追加された `_id` フィールドによってそれが上書きされます。

2. フィールドの名前とタイプを変更するには、フィールドをハイライト表示し、**[変更]** をクリックします。
3. **[名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。
4. **[タイプ]** フィールドで、データに対して数学的な操作を行う予定がない場合は、データ タイプを文字列のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータタイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータタイプに変換されます。
このステージでは、以下のデータタイプがサポートされています。

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、 $2^{-1074} \sim (2-2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- long** 正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
- string** 文字シーケンス。

5. テーブルまたはスキーマファイルに存在しないフィールドを追加することもできます。**[追加]** をクリックして新しいフィールドを追加します。フィールドを削除する場合は、**[削除]** をクリックします。

注：リスト タイプの下にのみ、新しいフィールドを追加できます。

6. **[OK]** をクリックします。

NoSQL DB データフロー オプション

以下では、NoSQL DB の実行時オプションをサポートするようにデータフローを設定する手順について説明します。

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. NoSQLDB ステージを展開します。
6. データフロー オプションは、次の表に示すようにエクスポートされます。

データベース	読み込み	書き込み
Mongo DB	接続	接続
	テーブル	テーブル

データベース	読み込み	書き込み
Couchbase DB	接続	接続
	ビュー	
	設計ドキュメント名	

選択した NoSQL DB オプション名が、**【オプション名】** フィールドと **【オプション ラベル】** フィールドに表示されます。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。

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、**【選択ステージ】** オプションを選択します。
9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **【OK】** をクリックします。
12. 必要に応じて、オプションの追加を続けます。
13. 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。
14. データフローを保存してエクスポートします。

Read from SAP

Read from SAP ステージでは、データフローへの入力として、SAP データベースからデータを読み取ります。単一または複数のテーブルからデータを読み取ることができます。複数のテーブルからデータを読み取る場合は、結合操作を実行して、データフローに読み込むレコードを決定します。

表 3 : SAP アプリケーションの最小要件

コンポーネント	リリース	SP レベル	サポート パッケージ
SAP_BASIS	740	0005	SAPKB74005
SAP_ABA	740	0005	SAPKA74005

注：SAP Business Suite 7 アプリケーションのサポートされている最小バージョンは ECC 6.0 EHP 7 および CRM 7.0 EHP 3 です。次の表に、コンポーネント、リリース、およびサポート パック レベルを示します。

SAP への接続

Read from SAP を使用して SAP からのデータをデータフローに読み込むには、Spectrum™ Technology Platform と SAP システムの間の接続を作成する必要があります。

注：サポート用データベースおよびデータフローを Spectrum™ Technology Platform サーバーにインストールし、SAP システムを Spectrum™ Technology Platform と通信するように設定する方法の詳細については、『インストール ガイド』の「Read from SAP 用のサポート ファイルのインストール」セクションを参照してください。

1. SAP 接続マネージャを開きます。Enterprise Designer の [ツール] > [SAP 接続管理] の下で、または Read from SAP ステージで [接続] フィールドの横にある [管理] ボタンをクリックすることによってこれを行うことができます。
2. [追加] をクリックします。
3. [接続名] フィールドで、この接続に名前を付けます。
4. 他のフィールドに、接続する SAP サーバーに関する情報を入力します。必要な情報については、SAP Basis アドミニストレータにお問い合わせください。

重要： ユーザ ID とパスワードは、管理者権限を持つ SAP アカウントのものである必要があります。

5. [テスト] をクリックして、接続を確認します。
6. [OK] をクリックします。

これで、Read from SAP ステージにおいて SAP からのデータをデータフローに読み込むために使用可能な接続が作成されました。

単一の SAP テーブルからのデータの読み取り

Read from SAP ステージは、SAP データベース内の単一テーブル、または複数のテーブルからデータを読み取るように設定できます。この手順は、単一のテーブルからデータを読み取るように Read from SAP を設定する方法を示しています。

1. Enterprise Designer で、[Read from SAP] をキャンバス上にドラッグします。
2. キャンバス上の [Read from SAP] ステージをダブルクリックします。
3. **[接続]** フィールドで、データフロー内に読み込むデータが含まれている SAP サーバーを選択します。SAP サーバー用に定義されている接続がない場合は、**[管理]** をクリックして接続を作成する必要があります。
4. **[ソース タイプ]** フィールドで、**[単一]** を選択します。
5. **[選択]** をクリックします。
6. データフロー内に読み込むテーブルを選択し、**[OK]** をクリックします。

注：最初の 200 テーブルのみがリストに表示されます。最初の 200 件として表示されていないテーブルを探すには、検索機能を使用します。検索フィールドでは、**[名前]** および **[ラベル]** 列の値のみが検索されます。

7. データフローで使用されるフィールド名を表示するには、**[テクニカル名の表示]** チェックボックスをオンにします。

SAP のフィールドには、表示のために使用されるわかりやすい名前と、可読性が低い場合がある一意の名前があります。例えば、1 つのフィールドが "Distribution Channel" というわかりやすい名前と "DIS_CHANNEL" というテクニカル名を持つ場合があります。フィールド名を確実にデータフロー内で有効なものにするために、テクニカル名がフィールド名として使用されます。

8. データフロー内に読み込むそれぞれのフィールドについて、**[含める]** 列にあるチェックボックスをオンにします。
9. **[OK]** をクリックします。
10. 特定のレコードのみを読み込む場合は、**[フィルタ]** タブでフィルタ条件を指定できます。定義したすべての条件を満たすレコードのみが、データフローに読み込まれます。
11. **[実行時]** タブで適切なフェッチ サイズを指定することにより、パフォーマンスを向上させることができます。

データベース テーブルから一度に読み取るレコードの数を指定するには、このオプションを選択します。例えば、フェッチ サイズの値が 100 で、読み取るレコードの総数が 1000 である場合、すべてのレコードを読むためにデータベースに 10 回アクセスすることになります。

最適なフェッチ サイズを設定すれば、パフォーマンスを大幅に向上できます。

注：使用している環境に最適なフェッチ サイズは、Read from DB ステージと Write to Null ステージ間の実行時間をテストすることで計算できます。詳細については、「[最適なフェッチ サイズの決定 \(342ページ\)](#)」を参照してください。

Read from SAP のデフォルトのフェッチ サイズは 10,000 です。

12. **[OK]** をクリックします。

これで Read from SAP ステージは、SAP データベースの単一のテーブルからのデータをデータフローに読み込むように設定されました。

複数の SAP テーブルからのデータの読み取り

Read from SAP ステージは、SAP データベース内の単一テーブル、または複数のテーブルからデータを読み取るように設定できます。この手順は、複数のテーブルからデータを読み取るように Read from SAP を設定する方法を示しています。複数のテーブルからデータを読み取るには、データを結合して 1 つのストリームにまとめる JOIN ステートメントを定義します。

1. Enterprise Designer で、**[Read from SAP]** をキャンバス上にドラッグします。
2. キャンバス上の **[Read from SAP]** ステージをダブルクリックします。
3. **[接続]** フィールドで、データフロー内に読み込むデータが含まれている SAP サーバーを選択します。SAP サーバー用に定義されている接続がない場合は、**[管理]** をクリックして接続を作成する必要があります。
4. **[ソース タイプ]** フィールドで、**[複数]** を選択します。
5. **[追加]** をクリックします。
6. データフロー内に読み込むテーブルを選択し、**[OK]** をクリックします。

注：最初の 200 テーブルのみがリストに表示されます。最初の 200 件として表示されていないテーブルを探すには、検索機能を使用します。検索フィールドでは、**[名前]** および **[ラベル]** 列の値のみが検索されます。

7. リストの最初のテーブルを選択して、**[関係の作成]** をクリックします。これがソース テーブルです。
8. **[ソース キー]** フィールドで、ソース テーブルの列を選択します。その列の値が、他のテーブルからのレコードとのマッチングに使用されます。
9. **[結合タイプ]** フィールドで、次のいずれかを選択します。

内部結合 ソーステーブルとターゲット テーブルの間で一致するレコードのみを返します。

左結合 ソース テーブルとターゲット テーブルの間に一致がない場合でも、ソース テーブルからのすべてのレコードを返します。このオプションは、ソース テーブルからのすべてのレコードに加えて、ターゲット テーブルで一致するレコードがあればそれを返します。

10. **[テーブル]** フィールドで、ターゲット テーブルを選択します。
11. **[テーブル キー]** フィールドで、ターゲット テーブルの列を選択します。そのデータを **[ソース キー]** フィールドからのデータと比較することで、レコードが結合条件を満たすかどうか判定されます。
12. **[OK]** をクリックします。
13. **[スキーマの選択]** をクリックします。
14. データフローに読み込むフィールドを選択します。データフローで使用されるフィールド名を表示するには、**[テクニカル名の表示]** チェックボックスをオンにします。

SAP のフィールドには、表示のために使用されるわかりやすい名前と、可読性が低い場合がある一意の名前があります。例えば、1つのフィールドが "Distribution Channel" というわかりやすい名前と "DIS_CHANNEL" というテクニカル名を持つ場合があります。フィールド名を確実にデータフロー内で有効なものにするために、テクニカル名がフィールド名として使用されます。

15. **[OK]** をクリックします。
16. 特定のレコードのみを読み込む場合は、**[フィルタ]** タブでフィルタ条件を指定できます。定義したすべての条件を満たすレコードのみが、データフローに読み込まれます。
17. **[実行時]** タブで適切なフェッチ サイズを指定することにより、パフォーマンスを向上させることができます。

データベース テーブルから一度に読み取るレコードの数を指定するには、このオプションを選択します。例えば、**フェッチ サイズ**の値が 100 で、読み取るレコードの総数が 1000 である場合、すべてのレコードを読むためにデータベースに 10回アクセスすることになります。

最適なフェッチ サイズを設定すれば、パフォーマンスを大幅に向上できます。

注：使用している環境に最適なフェッチ サイズは、**Read from DB** ステージと **Write to Null** ステージ間の実行時間をテストすることで計算できます。詳細については、「[最適なフェッチ サイズの決定 \(342ページ\)](#)」を参照してください。

Read from SAP のデフォルトのフェッチ サイズは 10,000 です。

これで Read from SAP ステージは、SAP データベースの複数のテーブルからのデータをデータフローに読み込むように設定されました。

Read from SAP におけるレコードのフィルタリング

Read from SAP のフィルタ設定によって、SAP テーブルからすべてのレコードではなくレコードのサブセットを読み込むことができます。レコードをフィルタするには、データフローに読み込むレコードに含まれている必要がある値を指定します。フィルタ条件を指定しない場合、テーブルのすべてのレコードがデータフローに読み込まれます。フィルタ条件の使用はオプションです。

注：Read from SAP ステージが複数の SAP テーブルからデータを読み込むように設定されている場合、フィルタは JOIN 操作の実行後に適用されます。

1. Read from SAP ステージで、**[フィルタ]** タブをクリックします。
2. **[追加]** をクリックします。
3. **[テーブル名]** フィールドで、フィルタを適用するレコードを含むテーブルを選択します。
4. **[フィルタ]** フィールドで、フィルタリングのベースとして使用するデータを含むフィールドを選択します。
5. 次のいずれかの演算子を選択します。

注：使用可能な演算子は、フィルタを適用するフィールドのデータ タイプによって異なります。

演算子	説明
が次の値を含む	文字列が指定された値を含むかどうかを確認します。
が次に等しい	フィールドの値が指定された値とマッチするかどうかを確認します。
が等しくない	フィールドの値が指定された値とマッチしないかどうかを確認します。
次の値より大きい	フィールドの数値が指定された値よりも大きいかどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
次の値以上	フィールドの数値が指定された値以上かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
次の値より小さい	フィールドの数値が指定された値未満かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。

演算子	説明
次の値以下	フィールドの数値が指定された値以下かどうかを確認します。この演算子は、数値データ タイプおよび数字を含む文字列フィールドに対してのみ適用できます。
が NULL である	フィールドが NULL 値かどうかを確認します。
が NULL でない	フィールドが NULL 値でないかどうかを確認します。
が次の値で始まる	フィールドが指定された値で始まっているかどうかを確認します。
次で終わる	フィールドが指定された値で終了しているかどうかを確認します。

6. 選択フィールドの値と比較する値を入力します。
7. **[OK]** をクリックします。
8. 必要に応じて、フィルタ条件をさらに追加します。

注：複数のフィルタ条件を指定した場合は、すべてのフィルタ条件が真でなければレコードはデータフローに読み込まれません。いずれかの条件が真でない場合、レコードはデータフローに読み込まれません。

Read from Spreadsheet

Read from Spreadsheet は、データフローへの入力として、Excel スプレッドシートからデータを読み取ります。サポートされている形式は *.xls および *.xlsx です。

[ファイル プロパティ] タブ

[ファイル プロパティ] タブには、スプレッドシートとデータフローに読み込むデータを指定するためのオプションがあります。

フィールド名	説明
サーバ名	<p>入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。</p>
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。</p> <p>注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
データ選択	<p>データフローに読み込むデータをスプレッドシートから選択する方法を次のように指定します。</p> <p>シート データ スプレッドシート内のシートからすべてのデータを読み込むには、このオプションを選択します。</p> <p>範囲データ 読み込むセルの範囲を指定して、シートからデータのサブセットを読み込むには、このオプションを選択します。</p> <p>名前付き範囲 スプレッドシートの名前付き範囲を指定して、シートからデータのサブセットを読み込むには、このオプションを選択します。</p>
シート選択	<p>[データ選択] フィールドで [シート データ] または [範囲データ] を選択した場合、このオプションを使用して、データフローにデータを読み込むシートを選択します。</p>
範囲	<p>[データ選択] フィールドで [範囲データ] を選択した場合、このオプションを使用して、範囲を開始するセルと範囲を終了するセルを指定します。</p>
名前付き範囲	<p>[データ選択] フィールドで [名前付き範囲] を選択した場合、このオプションを使用して、データフローに読み込む範囲の名前を指定します。範囲はスプレッドシートに定義されています。一覧表示される範囲がない場合は、スプレッドシートに範囲が定義されていないことを意味します。</p>

フィールド名	説明
ヘッダ行	<p>列の見出しを含む行を指定する場合は、このボックスをオンにします。列の見出しはデータフロー フィールド名になりますが、フィールド名は【フィールド】タブで変更できます。このチェックボックスをオンにしない場合、データフローフィールドには汎用のデフォルト名 (Column1、Column2 など) が付与されます。</p> <p>指定するヘッダ行は、データ選択における相対位置になります。例えば、【データ選択】フィールドで【範囲データ】を選択し、その範囲が 5 行目から始まる場合、ヘッダ行として 1 を指定すると、スプレッドシートの 5 行目がヘッダ行として使用されます。その範囲の先頭の行はスプレッドシートの 5 行目だからです。</p>
ヘッダからのデータ オフセット	<p>ヘッダ行を指定した場合、このフィールドは、データが含まれる先頭行をヘッダからの相対位置で指定します。例えば、1 を指定すると、ヘッダの下の 1 行目が、データフローに読み込むデータの先頭行になります。2 を指定すると、ヘッダの下の 2 行目が、データフローに読み込むデータの先頭行になります。</p>
データの先頭行	<p>ヘッダ行を指定しない場合、このフィールドは、データフローに読み込むデータの先頭行がデータ選択のどの行に含まれるかを指定します。指定する行は、データ選択における相対位置になります。例えば、【データ選択】フィールドで【範囲データ】を選択し、その範囲が 5 行目から始まる場合、データの先頭行として 1 を指定すると、データフローに読み込むデータの先頭行は 5 行目になります。</p>
空の行を無視	<p>空の行をデータフローから除外したい場合は、このオプションを選択します。このオプションを選択しない場合は、スプレッドシートの空の行は、データフロー内で空のレコードになります。</p> <p>注：このオプションは、プレビューに表示されるデータには反映されません。このオプションを選択した場合でも、プレビューには空の行が常に表示されます。</p>

【フィールド】タブ

【フィールド】タブには、スプレッドシートのデータをデータフロー内のフィールドにマッピングするためのオプションがあります。

オプション	説明
再生成	このボタンをクリックすると、[ファイル プロパティ] タブで定義した入力ファイルのフィールドが [フィールド] タブに設定されます。
タイプの検出	このボタンをクリックすると、すべてのフィールドのデータ タイプが自動的に判断されます。フィールドを選択して [変更] をクリックすることにより、フィールドのデータ タイプを手動で変更することができます。
変更	フィールド名またはデータ タイプを変更するには、フィールドを選択してからこのボタンをクリックします。

Read from Variable Format File

Read from Variable Format File は、さまざまなレイアウトのレコードを含むファイルからデータを読み込みます。各レコードは、リスト フィールドとして読み込まれます。親レコードタイプを示すタグを指定でき、それ以外のすべてのレコードタイプが親の下のリストフィールドになります。

可変フォーマット ファイルには、次の特性があります。

- ファイル内のレコードは、異なるフィールド、および異なる数のフィールドを持つことができる。
- 各レコードには、レコードのタイプを識別するタグ (通常は数字) を含める必要がある。
- 階層的な関連性をサポートしている。

可変フォーマット ファイルの例

次の例は、2 人の顧客 Joe Smith と Anne Johnson の当座預金口座の取引に関する情報を含む、可変フォーマット ファイルを示しています。この例のファイルは、フィールド区切り文字としてカンマを使用する区切り記号付きファイルです。

```
001 Joe,Smith,M,100 Main St,555-234-1290
100 CHK12904567,12/2/2007,6/1/2012,CHK
200 1000567,1/5/2012,Fashion Shoes,323.12
001 Anne,Johnson,F,1202 Lake St,555-222-4932
100 CHK238193875,1/21/2001,4/12/2012,CHK
```

```

200 1000232,3/5/2012,Blue Goose Grocery,132.11
200 1000232,3/8/2012,Trailway Bikes,540.00
    
```

各レコードの先頭フィールドには、レコードのタイプ、およびレコードのフォーマットを識別するタグが含まれています。

- 001: Customer レコード
- 100: Account レコード
- 200: Account transaction レコード

区切り記号付きファイルにおいて、上の例に示すように、タグ値 (001、100、200) をレコード先頭の固定長バイトに含めることは一般的です。

各レコードには独自のフォーマットがあります。

- 001: FirstName,LastName,Gender,Address,PhoneNumber
- 100: AccountID,DateOpened,ExpirationDate,TypeOfAccount
- 200: TransactionID,DateOfTransaction,Vendor,Amount

レコードフォーマット 100 (アカウント レコード) は、前のレコード 001 の子で、レコードフォーマット 200 (アカウント トランザクション レコード) は、前のレコード 100 (アカウント レコード) の子です。したがって、例のファイルでは、Joe Smith のアカウント CHK12904567 に Fashion Shoes で 2012/1/5 に数量 323.12 のトランザクションが発生していました。同様に、Anne Johnson のアカウント CHK238193875 には、Blue Goose Grocery で 2012/3/5 に 1 つ、Trailway Bikes で 2012/3/8 に 1 つのトランザクションが発生していました。

[ファイル プロパティ] タブ

オプション名	説明
サーバ名	入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。

オプション名

説明

ファイル名

ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを選択します。

ワイルドカード文字を使用して、ディレクトリ内の複数のファイルからデータを読み込むことができます。サポートされているワイルドカード文字は、* と ? です。例えば、*.csv と指定して、ディレクトリ内にある、拡張子が .csv のファイルをすべて読み込むことができます。複数のファイルを正常に読み込むには、各ファイルが同じレイアウト (同じ位置に同じフィールド) を持つ必要があります。**[フィールド]** タブで指定したレイアウトに一致しないレコードは、形式に誤りのあるレコードとして扱われます。

HDFS ファイル サーバーからのファイルの読み込みでサポートされる圧縮形式を次に示します。

1. GZIP (.gz)
2. BZIP2 (.bz2)

注：ファイルの拡張子は、そのファイルの解凍に使用される圧縮形式を示します。

重要： なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。

レコードタイプ

ファイル内のレコードのフォーマット。次のいずれかを選択します。

- | | |
|---------|---|
| 行順次 | ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキスト ファイル。 |
| 固定長 | ファイル内の各レコードの長さ (文字数) が一定で、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキスト ファイル。 |
| 区切り記号付き | ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドがカンマ (,) などの特定の文字で区切られているテキスト ファイル。 |

オプション名	説明
文字エンコーディング	テキスト ファイルのエンコーディング。次のいずれかを選択します。
CP1252	このエンコーディングは Windows-1252 文字セット、または単純に Windows 文字セットとも呼ばれています。これは ISO-8859-1 の上位クラスであり、128 ~ 159 のコード範囲を使用して、ISO-8859-1 文字セットに含まれていない追加の文字を表示します。
UTF-8	すべての Unicode 文字をサポートし、かつ ASCII との下位互換性があります。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。
UTF-16	すべての Unicode 文字をサポートします。しかし、ASCII との下位互換性はありません。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。
US-ASCII	英語のアルファベット順に従う文字エンコーディング。
UTF-16BE	ビッグエンディアン UTF-16 エンコーディング (下位アドレスが上位バイトとなるようにシリアル化)。
UTF-16LE	リトルエンディアン UTF-16 エンコーディング (下位アドレスが下位バイトとなるようにシリアル化)。
ISO-8859-1	主として西ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-1 と呼ばれます。
ISO-8859-3	主として南ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-3 と呼ばれます。
ISO-8859-9	主としてトルコ語で使われる ASCII 文字エンコーディング。Latin-5 と呼ばれます。
CP850	西ヨーロッパの言語を書くための ASCII コード ページ。
CP500	西ヨーロッパの言語を書くための EBCDIC コード ページ。
Shift_JIS	日本語のための文字エンコーディング。
MS932	NEC 特殊文字、NEC 選定 IBM 拡張文字、IBM 拡張文字を含めた Microsoft の拡張版 Shift_JIS 文字コード。
CP1047	Latin-1 文字セット全体を含む EBCDIC コード ページ。
レコード長	固定長ファイルでは、個々のレコードの文字数を指定します。

オプション名	説明
フィールド区切り文字	<p>区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは () 記号がフィールド区切り文字として使われています。</p> <pre data-bbox="560 451 1421 514">7200 13TH ST MIAMI FL 33144</pre> <p>フィールド区切り文字として定義できるのは次の文字です。</p> <ul data-bbox="560 577 706 808" style="list-style-type: none">• スペース• タブ• カンマ• ピリオド (.)• セミコロン• パイプ () <p>これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。</p>
タグ区切り文字	<p>区切り記号付きファイル内の各レコードの識別フィールドを区切るために、タグフィールドの後ろに配置する文字を指定します。タグ区切り文字は 1 文字でなければなりません。</p> <p>デフォルトでは、以下の文字がタグ区切り文字として選択できます。</p> <ul data-bbox="560 1155 706 1386" style="list-style-type: none">• スペース• タブ• カンマ• ピリオド (.)• セミコロン• パイプ () <p>これ以外の文字がタグ区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックして、カスタムなタグ区切り文字を追加して選択します。</p> <p>注：デフォルトで、[レコード区切り文字] は [フィールド区切り文字] として選択された文字と同じです。このフィールドを有効にして別の文字を選択するには、[フィールド区切り文字と同じ] チェックボックスをオフにします。</p>
[フィールド区切り文字と同じ]	<p>タグ区切り文字がフィールド区切り文字と同じかどうかを示します。別の文字をタグ区切り文字として選択するには、このチェックボックスをオフにします。</p> <p>注：デフォルトで、このチェックボックスはオフで、[タグ区切り文字] フィールドは無効になっています。</p>

オプション名

説明

Text qualifier

区切り記号付きファイル内のテキスト値を囲むのに使用する文字。

例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。

```
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
```

テキスト修飾子として定義できるのは次の文字です。

- 一重引用符 (')
- 二重引用符 (")

これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。

レコード区切り文字

順次ファイルまたは区切り記号付きファイル内のレコードを区切るのに使用する文字を指定します。[デフォルトの EOL を使用] チェック ボックスをオンにすると、このフィールドは使用できません。

使用できるレコード区切り文字の設定は次のとおりです。

Unix (U+000A) 改行 (LF) 文字でレコードを区切ります。これは Unix システムの標準のレコード区切り文字です。

Macintosh (U+000D) 復帰 (CR) 文字でレコードを区切ります。これは Macintosh システムの標準のレコード区切り文字です。

Windows (U+000D U+000A) 復帰改行 (CR+LF) でレコードを区切ります。これは Windows システムの標準のレコード区切り文字です。

これ以外の文字がレコード区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をレコード区切り文字として選択してください。

ルート タグ名

他のレコード タイプの親であるレコードに対して使用するタグ。例えば、3 つのレコード タイプ 001、100、200 があり、レコード タイプ 100 および 200 がレコード タイプ 001 の子である場合、001 はルート タグです。

オプション名

説明

固定長タグを使用

レコード タグを配置する各レコードの先頭に固定長スペースを割り当てるかどうかを指定します。この例では、固定長フィールドに、タグ 001、100、200 があるファイルを示しています。

```
001   Joe,Smith,M,100 Main St,555-234-1290
100   CHK12904567,12/2/2007,6/1/2012,CHK
200   1000567,1/5/2012,Mike's Shoes,323.12
```

タグ開始位置

[固定長タグを使用] ボックスをオンにした場合、各レコード内のタグの開始位置を指定します。例えば、タグがレコードの 4 文字目で始まる場合は 4 と指定します。

タグ長

[固定長タグを使用] ボックスをオンにした場合、**[タグの開始位置]** フィールドで指定した位置から始まるタグに割り当てるスペースの数を指定します。例えば、**[タグの開始位置]** フィールドで 3 と指定し、**[タグ長]** フィールドで 7 と指定した場合、4～10 の位置がレコードタグと見なされます。指定する値は、最も長いタグ名のすべての文字を含められるだけの大きさである必要があります。

[ルート タグ名] フィールドのタグ名を長くした場合、**[タグ長]** フィールドの値は自動的に増えます。

最大タグ長は 1024 です。

デフォルトの EOL を使用

Spectrum™ Technology Platform サーバーが実行されているオペレーティングシステムのデフォルトの行末 (EOL) 文字をファイルのレコード区切り文字として使用します。

ファイルの EOL 文字がサーバーのオペレーティング システムで使われているデフォルトの EOL 文字と異なる場合は、このオプションをオンにしないでください。例えば、ファイルで Windows の EOL が使われていて、サーバーの動作プラットフォームが Linux の場合は、このオプションをオンにしないでください。代わりに、**[レコード区切り文字]** フィールドで [Windows] オプションを選択します。

オプション名	説明
フィールド数が定義よりも少ないレコードを、形式誤りとみなす	<p>このオプションを有効にすると、完全なレコードより少ない数のフィールドを含む子レコードは形式誤りと見なされます。形式誤りレコードが発生すると、処理は次のルート タグまで進み、その間にあるすべての子タグが無視されます。例外は、行番号とともに、形式誤りの子レコードについての情報を格納するログに書き出されます。</p> <p>このオプションが有効かどうかにかかわらず、次の場合にはレコードは常に形式誤りと見なされます。</p> <ul style="list-style-type: none"> • タグが不明である • 行が空である • データを持たないタグがある • 別のタグの子であるタグを持つレコードがルート タグを持つレコードの直後に表示される

[フィールド] タブ

[フィールド] タブでは、ファイルから読み込む各フィールドの特性を指定します。

[実行時] タブ

フィールド名	説明
ファイル名	最初のタブで選択したファイル名が表示されます。
開始レコード	レコードをデータフローに読み込むときファイルの先頭部分にあるレコードをスキップしたければ、読み込みたい最初のレコードを指定します。例えば、最初の50個のレコードをスキップする場合は51と指定します。これで51番目のレコードがデータフローに読み込まれる最初のレコードとなります。
すべてのレコード	[開始レコード] フィールドで指定したレコードからファイルの最後までレコードをすべて読み込む場合は、このオプションをオンにします。
最大レコード数	[開始レコード] フィールドで指定したレコードを起点にそこから一定の数のレコードを読み込む場合は、このオプションをオンにします。例えば、最初の100個のレコードを読み込みたい場合は、このオプションをオンにして100と入力します。

区切り記号付き可変フォーマット ファイルのフィールドの定義

この手順では、区切り記号付きファイルに対して **Read from Variable Format File** ステージでフィールドを定義する方法について説明します。

1. **Read from Variable Format File** ステージで、**[フィールド]** タブをクリックします。
2. **[再生成]** をクリックします。

レコードタイプごとに、すべてのフィールドの一覧が表示されます。フィールドごとに、以下の情報が表示されます。

Parent フィールドを表示するレコードタイプを示す、入力ファイルのタグ。タグが数字で始まる場合、タグには "NumericTag_" という接頭辞が付きます。例えば、100 というタグは **NumericTag_100** になります。データフロー フィールド名の先頭を数字にすることはできないため、接頭辞が必要です。

フィールド データフローでフィールドのために使われる名前。デフォルトでは、フィールド名の形式は <タグ名>_<列番号> です。例えば、レコードタイプが **Owner** である最初のフィールドは **Owner_Column1**、2 番目のフィールドは **Owner_Column2** となります。

タイプ フィールドのデータタイプ。

注：最初の 50 個のレコードがフィールド一覧の生成に使用されます。フィールド一覧を生成するため、入力ファイルには少なくとも 2 つのルート タグを含める必要があります。

3. タグ間の親/子関係を変更する場合は
 - a) **[タグ階層の変更]** をクリックします。
 - b) タグをクリックしてドラッグし、変更するタグ階層を定義します。
 - c) **[OK]** をクリックします。
4. フィールドの名前またはデータタイプを変更する場合は、フィールドを選択し、**[変更]** をクリックします。
5. **[名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。

通常は、デフォルト名をフィールド内のデータを表す意味のある名前置き換えます。例えば、次の入力データを考えます。

```
001 Joe,Smith,M,100 Main St,555-234-1290
```

このレコードには親タグ 001 があり、デフォルトでは次のフィールドが作成されます。

```
NumericTag_001_Column1: Joe
```

NumericTag_001_Column2: Smith
 NumericTag_001_Column3: M
 NumericTag_001_Column4: 100 Main St
 NumericTag_001_Column5: 555-234-1290

名前がデータを説明するように、フィールド名を変更します。例:

FirstName: Joe
 LastName: Smith
 Gender: M
 AddressLine1: 100 Main St
 PhoneNumber: 555-234-1290

注: リスト フィールド名は変更できません。リスト フィールドには指定されたレコード タイプのすべてのフィールドが含まれるため、フィールド名として常に入力ファイルのタグ名が使用されます。

6. フィールドのデータ タイプを変更するには、使用するデータ タイプを **[タイプ]** フィールドで選択します。

次のデータ タイプを使用できます。

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。

bytearray バイトの配列 (リスト)。

注: bytearray は REST サービスの入力としてはサポートされていません。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。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、 $-2^{31} (-2,147,483,648) \sim 2^{31} - 1 (2,147,483,647)$ 。

list 厳密に言えば、リストはデータタイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。

```
<Names>
  <Name>John Smith</Name>
  <Name>Ann Fowler</Name>
</Names>
```

XML のリスト データ タイプが複数の値で構成される単純データ タイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データ タイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。

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。

- 日付、時間、または数値データ タイプを選択した場合は、デフォルトの日付と時間の形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォルトの形式は、Management Console のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、Enterprise Designer のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]** を選択したままにします。別の形式を指定するには、**[カスタム]** を選択し、以下の手順に従います。

注：日付/時間形式は、ファイルから読み込むデータを正確に反映する形式を選択することが重要です。例えば、ファイルに含まれている日付データの形式は月/日/年であるが、選択した日付形式が日/月/年であると、データフローで実行する日付計算 (日付によるソートなど) に正確な日付が反映されません。また、レコードのタイプ変換が失敗することもあります。その場合、Management Console または Enterprise Designer のタイプ変換オプションで指定された失敗操作が有効になります。

- [ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータタイプによって異なります。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。

選択した形式の例が **[形式]** フィールドの右側に表示されます。

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](#)（331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](#)（334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。

8. **[OK]** をクリックします。

行順次ファイルまたは固定長可変フォーマット ファイルのフィールドの定義

この手順では、行順次ファイルまたは固定長ファイルに対して **Read from Variable Format File** ステージでフィールドを定義する方法について説明します。

1. **Read from Variable Format File** ステージで、**[フィールド]** タブをクリックします。
2. **[タグの取得]** をクリックします。

レコードタイプごとに、すべてのフィールドの一覧が表示されます。フィールドごとに、以下の情報が表示されます。

Parent フィールドを表示するレコードタイプを示す、入力ファイルのタグ。タグが数字で始まる場合、タグには "NumericTag_" という接頭辞が付きます。例えば、100 というタグは NumericTag_100 になります。データフロー フィールド名の先頭を数字にすることはできないため、接頭辞が必要です。

フィールド データフローでフィールドのために使われる名前。デフォルトでは、フィールド名の形式は <タグ名>_<列番号> です。例えば、レコードタイプが Owner である最初のフィールドは Owner_Column1、2 番目のフィールドは Owner_Column2 となります。

タイプ フィールドのデータタイプ。

注：最初の 50 個のレコードがフィールド一覧の生成に使用されます。フィールド一覧を生成するため、入力ファイルには少なくとも 2 つのルート タグを含める必要があります。

3. **[フィルタ]** フィールドで、フィールドを定義するレコード タイプのタグを選択し、**[追加]** をクリックします。

注：フィルタを設定しても、どのフィールドがデータフローに読み込まれるかには影響しません。参照のしやすさを考慮してフィールドの一覧をフィルタリングするだけです。

4. **[名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。
5. **[タイプ]** フィールドで、データに対して数学的または日時に関する操作を行う予定がない場合は、データ タイプを `string` のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータ タイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータ タイプに変換されます。

Spectrum™ Technology Platform では、以下のデータ タイプがサポートされています。

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。

bytearray バイトの配列 (リスト)。

注：bytearray は REST サービスの入力としてはサポートされていません。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。

list 厳密に言えば、リストはデータ タイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクション

ンです。例えば、**Names** フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。

```
<Names>
  <Name>John Smith</Name>
  <Name>Ann Fowler</Name>
</Names>
```

XML のリスト データ タイプが複数の値で構成される単純データ タイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データ タイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。

- 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。

6. 日付、時間、または数値データ タイプを選択した場合は、デフォルトの日付と時間の形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォルトの形式は、Management Console のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、Enterprise Designer のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]**を選択したままにします。別の形式を指定するには、**[カスタム]**を選択し、以下の手順に従います。

注：日付/時間形式は、ファイルから読み込むデータを正確に反映する形式を選択することが重要です。例えば、ファイルに含まれている日付データの形式は月/日/年であるが、選択した日付形式が日/月/年であると、データフローで実行する日付計算 (日付によるソートなど) に正確な日付が反映されません。また、レコードのタイプ変換が失敗することもあります。その場合、Management Console または Enterprise Designer のタイプ変換オプションで指定された失敗操作が有効になります。

- a) **[ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータ タイプによって異なります。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。

選択した形式の例が **[形式]** フィールドの右側に表示されます。

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](#)（331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](#)（334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。

7. **[開始位置]** フィールドで、フィールドの最初の文字の位置を入力し、**[長さ]** フィールドで、フィールド内の文字数を入力します。

例えば、フィールドがレコードの 10 文字目から始まり、長さが 5 文字の場合は、開始位置 10 と長さ 5 を指定します。

8. **[追加]** をクリックします。
9. このプロセスを繰り返して、その他のフィールドをレコードタイプに追加します。あるいは、フィールドの追加が終了したら、**[閉じる]** をクリックします。

可変フォーマット データのフラット化

多くの場合、可変フォーマット ファイルデータには、1つのレコードタイプがその他のレコードタイプの親となっている、階層的関係を持つレコードが含まれます。多くのステージではフラット形式のデータが必要とされるので、下流のステージで使えるようにするためにデータをフラット化しなければならないこともあります。例えば、次の入力データを考えます。

```
001   Joe,Smith,M,100 Main St,555-234-1290
100   CHK12904567,12/2/2007,6/1/2012,CHK
200   1000567,1/5/2012,Fashion Shoes,323.12
001   Anne,Johnson,F,1202 Lake St,555-222-4932
100   CHK238193875,1/21/2001,4/12/2012,CHK
200   1000232,3/5/2012,Blue Goose Grocery,132.11
200   1000232,3/8/2012,Trailway Bikes,540.00
```

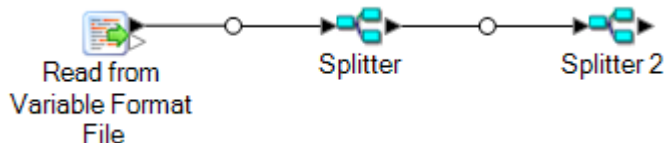
トランザクションごとに 1レコードとなるように、レコードをフラット化できます。上記の例では、トランザクションレコード (タグ 200 のレコード) を受け入れ、アカウント所有者情報 (タグ 001 のレコード) とアカウント詳細 (タグ 100 のレコード) を含めるようにフラット化することを意味します。

次の手順は、**Splitter** ステージを使用してレコードをフラット化する方法を説明したものです。

1. **Read from Variable Format File** ステージをデータ フローに追加して、設定します。詳細については、「[Read from Variable Format File](#) (218ページ)」を参照してください。
2. **Splitter** ステージを追加して、**Read from Variable Format File** に接続します。
3. 入力データ内の子レコードタイプごとに 1つの **Splitter** ステージとなるように、必要に応じて **Splitter** ステージを追加します。

- すべての **Splitter** ステージを接続します。

データ フローは次のようになっているはずです。



- 最初の **Splitter** ステージをダブルクリックして、ステージのオプション を開きます。
- [分割位置]** フィールドで、いずれかの子レコード タイプを選択します。
- [OK]** をクリックします。
- 追加した **Splitter** ステージをそれぞれ設定し、各 **Splitter** の **[分割位置]** フィールドで異なる子レコード タイプを選択します。

Read From XML

Read from XML ステージでは、XML ファイルがジョブまたはサブフローに読み込まれます。ここでは、ファイルのパスとデータ フォーマット (XML スキーマとデータ要素の詳細など) が定義されます。

単純 XML 要素はフラットフィールドに変換され、次のステージに渡されます。単純 XML データは、データのみを含み、子要素を含まない XML 要素からなるレコードで構成されます。例えば、以下は単純 XML データ ファイルです。

```
<customers>
  <customer>
    <name>Sam</name>
    <gender>M</gender>
    <age>43</age>
    <country>United States</country>
  </customer>
  <customer>
    <name>Jeff</name>
    <gender>M</gender>
    <age>32</age>
    <country>Canada</country>
  </customer>
  <customer>
    <name>Mary</name>
    <gender>F</gender>
    <age>61</age>
    <country>Australia</country>
  </customer>
</customers>
```

```
</customer>
</customers>
```

この例では、各レコードに <name>、<gender>、<age>、<country> などの単純な XML 要素が含まれています。子要素を含んでいる要素はありません。

大部分のステージではフラット形式のデータが必要とされるので、**Read from XML** ステージはこのような単純データを自動的にフラット化します。階層構造を保持する場合は、**Read from XML** ステージの後で **Aggregator** ステージを使用して、データを階層データに変換します。

複合 XML 要素は階層形式のまま、リストフィールドとして渡されます多くのステージはフラット形式のデータを必要とするので、下流のステージでデータを使えるようにするために、複合 XML 要素のフラット化が必要になることがあります。詳細については、[複合 XML 要素のフラット化 \(238ページ\)](#) を参照してください。

注：Read From XML は、XML タイプ `xs:anyType` および `xs:anySimpleType` をサポートしていません。

[ファイル プロパティ] タブ

表 4: [ファイル プロパティ] タブ

オプション名	説明
スキーマ ファイル	<p>XSD スキーマ ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。データ ファイルがスキーマに照らして検証されるようにするためには、スキーマ ファイルをサーバー上に置いておかなければなりません。スキーマ ファイルがサーバー上に存在しないと検証が無効になります。</p> <p>別のやり方として、XSD ファイルの代わりに XML ファイルを指定してもかまいません。XML ファイルを指定した場合は、その XML ファイルの構造に基づいてスキーマが推定されます。XSD ファイルの代わりに XML ファイルを使用する場合は、次の制限があります。</p> <ul style="list-style-type: none"> XML ファイルは 1 MB 以下でなければなりません。XML ファイルのサイズが 1 MB を越える場合は、XML の構造を維持しつつ、データの一部の削除を試みてください。 データ ファイルは推定されるスキーマに照らして検証されません。 <p>注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>

オプション名	説明
データ ファイル	<p>XML データ ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。</p> <p>注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
プレビュー	<p>スキーマまたは XML ファイルのプレビューを表示します。XSD ファイルを指定すると、選択した XSD がツリー構造に反映されます。スキーマファイルとデータ ファイルの両方を指定したら、太字のスキーマ要素をクリックして、その要素に含まれるデータのプレビューを見ることができます。</p>

[フィールド] タブ

表 5: [フィールド] タブ

オプション名	説明
フィルタ	<p>参照のしやすさを考慮して要素や属性の一覧をフィルタリングします。フィルタを設定しても、どのフィールドが出力されるかには影響しません。あくまでも参照のために要素や属性の一覧をフィルタリングするだけです。</p>
XPath	<p>XPath 列には要素または属性の XPath 式が表示されます。これは専ら情報表示のためのものです。XPath の詳細については、こちらのページを参照してください。</p>
フィールド	<p>データフローで要素または属性のために使われる名前。フィールド名を変更するには、ダブルクリックして新しいフィールド名を入力します。</p>

オプション名	説明
--------	----

タイプ	
-----	--

オプション名

説明

フィールドで使用するデータ タイプ。

- 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。date のフォーマットは yyyy-MM-dd でなければなりません。例えば、2012-01-30 のようになります。
- datetime** 月、日、年、時、分、秒を含むデータ タイプ。datetime のフォーマットは yyyy-MM-dd'THH:mm:ss でなければなりません。例えば、2012-01-30T06:15:30 のようになります。
- 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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- list** 厳密に言えば、リストはデータ タイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。


```

                <Names>
                  <Name>John Smith</Name>
                  <Name>Ann Fowler</Name>
                </Names>
            
```

XML のリスト データ タイプが複数の値で構成される単純データ タイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データ タイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。
- long** 正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9223372036854775808) \sim 2^{63}-1 (9223372036854775807)$ 。

オプション名	説明
string	文字シーケンス。
time	時刻を含むデータ タイプ。time のフォーマットは <i>HH:mm:ss</i> でなければなりません。例えば、21:15:59 のようになります。
含める	このフィールドをデータフローで使えるようにするか除外するかを指定します。

例: 簡単な XML ファイル

この例では、次のファイルをデータフローに読み込みます。

```
<addresses>
  <address>
    <addressline1>One Global View</addressline1>
    <city>Troy</city>
    <state>NY</state>
    <postalcode>12128</postalcode>
  </address>
  <address>
    <addressline1>1825B Kramer Lane</addressline1>
    <city>Austin</city>
    <state>TX</state>
    <postalcode>78758</postalcode>
  </address>
</addresses>
```

この例で、<addressline1>、<city>、<state>、および <postalcode> を含めたとします。その場合は、<address> 要素ごとに1つのレコードが生成されます。<address> が <addressline1>、<city>、<state>、および <postalcode> の共通の親要素だからです。

複合 XML 要素のフラット化

データフローのほとんどのステージで、フラット形式のデータが必要とされます。つまり、XML ファイルの階層データをデータフローに読み込む場合に、データに複合 XML 要素が含まれているときは、データをフラット化する必要があります。複合 XML 要素は、他の要素または属性を含む

要素です。例えば、次のデータ ファイルでは、<address> 要素と<account> 要素が複合 XML 要素を示しています。

```
<customers>
  <customer>
    <name>Sam</name>
    <gender>M</gender>
    <age>43</age>
    <country>United States</country>
    <address>
      <addressline1>1253 Summer St.</addressline1>
      <city>Boston</city>
      <stateprovince>MA</stateprovince>
      <postalcode>02110</postalcode>
    </address>
    <account>
      <type>Savings</type>
      <number>019922</number>
    </account>
  </customer>
  <customer>
    <name>Jeff</name>
    <gender>M</gender>
    <age>32</age>
    <country>Canada</country>
    <address>
      <addressline1>26 Wellington St.</addressline1>
      <city>Toronto</city>
      <stateprovince>ON</stateprovince>
      <postalcode>M5E 1S2</postalcode>
    </address>
    <account>
      <type>Checking</type>
      <number>238832</number>
    </account>
  </customer>
  <customer>
    <name>Mary</name>
    <gender>F</gender>
    <age>61</age>
    <country>Australia</country>
    <address>
      <addressline1>Level 7, 1 Elizabeth Plaza</addressline1>
      <city>North Sydney</city>
      <stateprovince>NSW</stateprovince>
      <postalcode>2060</postalcode>
    </address>
    <account>
      <type>Savings</type>
      <number>839938</number>
    </account>
  </customer>
</customers>
```

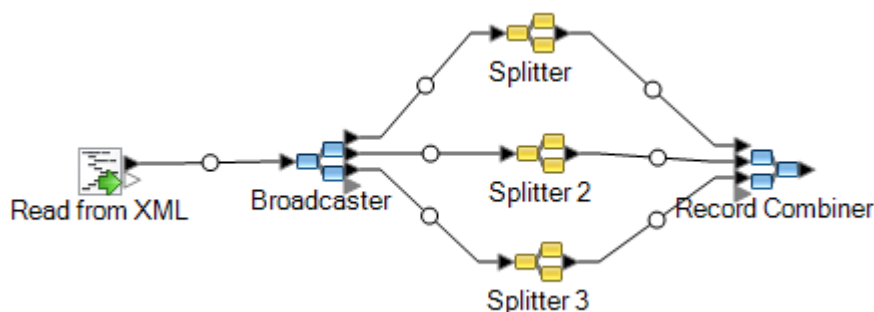
```
</customer>
</customers>
```

次の手順では、**Splitter** ステージを使用して、複数の複合 XML 要素を含む XML データをフラット化する方法を説明します。

注：データに 1 つの複合 XML 要素が含まれる場合は、1 つの **Splitter** ステージを使用して、**Read from XML** ステージを **Splitter** ステージに接続するだけで、データをフラット化できます。1 つの複合 XML 要素を含むデータファイルに対しては、この手順で説明するように **Broadcaster** ステージと **Record Combiner** ステージを使用する必要はありません。

1. **Read from XML** ステージをデータフローに追加して、設定します。詳細については、「[Read From XML \(233ページ\)](#)」を参照してください。
2. **Broadcaster** ステージを追加して、**Read from XML** ステージを接続します。
3. データ内の複合 XML 要素ごとに **Splitter** ステージを追加します。
4. **Broadcaster** ステージを各 **Splitter** に接続します。
5. **Record Combiner** ステージを追加して、各 **Splitter** を接続します。

データフローは次のようになっているはずです。



6. 最初の **Splitter** ステージをダブルクリックして、ステージのオプションを開きます。
7. **[分割位置]** フィールドで、いずれかの複合フィールドを選択します。上記の例のデータファイルでは、**address** フィールドを選択できます。
8. **[OK]** をクリックします。
9. 追加した **Splitter** ステージをそれぞれ設定し、各 **Splitter** の **[分割位置]** フィールドで異なる複合 XML 要素を選択します。

データフローは、複合 XML 要素を持つレコードを含む XML 入力を受け入れ、データをフラット化するように設定されました。**Record Combiner** から生成されるレコードは、フラットデータを必要とするすべてのステージに送ることができます。例えば、**Record Combiner** ステージを **Validate Address** ステージに接続して、住所を検証できます。

SQL Command

SQL Command では、データ フローの各レコードに対して 1 つ以上の SQL Command を実行します。SQL Command を使用すると、次の操作を実行できます。

- サブクエリ/他のテーブルとの結合を含む文など、複雑な INSERT/UPDATE 文を実行する。
- データを挿入/更新して参照の整合性を保持した後にテーブルを更新する。
- データベースのレコードを更新または削除してから置換レコードをロードする。
- 単一のトランザクションで複数のテーブルを更新する。

主要な SQL Command を実行する前および後に追加の SQL Command を実行し、ストアド プロシージャを呼び出すことができます。

注：ストアド プロシージャを実行するには、次の構文を使用します。

```
Call <Procedure Name>
```

SQL Command から呼び出されるストアド プロシージャでは OUT パラメータを使用できません。

注：SQL Command の複数の実行時インスタンスを使用することで、パフォーマンスを大きく改善できます。複数の実行時インスタンスを指定するには、**[実行時]** ボタンをクリックします。

全般

[全般] タブは、各レコードに対して 1 回実行する動的な SQL 文を指定する場所です。次の表に、**[全般]** タブで使用可能なオプションを示します。

オプション

説明

接続

使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、**【管理】** をクリックします。

注：このオプションは **Enterprise Designer** によってのみ使用できます。

データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。

接続名 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。

データベースドライバ 適切なデータベース タイプを選択します。

接続オプション データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。

SQL 文

データ フローで各レコードに対して実行する SQL 文を入力します。入力を開始すると、自動入力ポップアップウィンドウに有効な SQL Command が表示されます。複数の SQL 文を入力する場合は、各 SQL 文をセミコロン (;) で区切ります。

データフロー フィールドの値を指定するには、次の構文を使用します。

`${<field name>}`

<field name> は、データ フローのフィールドの名前です。

例を次に示します。

```
UPDATE MyDatabase.dbo.customer
SET name=${Name}
WHERE id=${ID};
```

この例では、`${Name}` はデータフローの **Name** フィールドの値で置き換えられ、`${ID}` はデータフローの **ID** フィールドの値で置き換えられます。

注：クエリでは完全修飾名前を使用する必要があります。例えば、`MyDatabase.dbo.customer` です。

トランザクション処理中

レコードをバッチ処理するか、すべてのレコードを同時に処理するかを指定します。以下のいずれか:

バッチ サイズ レコードを指定されたサイズのバッチにレコードをグループ化し、一度に1つずつバッチを処理します。

実行全体 すべてのレコードから1つの大きなバッチを作成し、すべてのトランザクションを同時に処理します。

オプション

説明

エラー処理

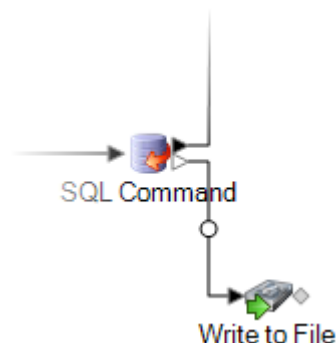
SQL Command の実行中にエラーが検出された場合の処理を指定します。次のいずれかです。

エラー時にデータフローを終了しない SQL Command の実行中にデータベースがエラーを返しても、データフローは実行を継続します。

データフローを終了するエラー数 データベースが返すエラー数が指定された数に達すると、データフローは実行を停止します。

注：SQL の構文エラーがある場合は、ここで選択する設定にかかわらず、データフローは常に終了します。

また、オプションでエラーレコードをシンクに書き込むこともできます。その場合は、SQL Command エラーポートを目的のシンクのタイプに接続します。エラーポートは、データフロー内のステージアイコンの右側に白の三角形で表されます。例えば、エラーレコードをフラットファイルに書き込むには、次のように、SQL Command エラーポートを Write to File ステージに接続します。



プリ/ポスト SQL

[プリ/ポスト SQL] タブは、データフローの実行あたり 1 回実行する SQL 文を指定する場所です。一方、レコードあたり 1 回実行する SQL 文は [全般] タブで指定します。次の表に、[プリ/ポスト SQL] タブで使用可能なオプションを示します。

オプション

説明

プリ SQL

ステージに入力されるレコードが処理される前に実行する 1 つ以上の SQL Command を入力します。ここで入力する SQL 文は、データフローの実行が開始されてから SQL Command ステージで最初のレコードが処理されるまでの間に、実行あたり 1 回実行されます。

プリ SQL の使用例としては、処理されるレコード用のテーブルを作成することがあります。

オプション

説明

プリ SQL の自動コミット

[全般] タブの SQL 文を実行する前にプリ SQL 文をコミットする場合は、このボックスをチェックします。

このボックスをチェックしない場合、プリ SQL 文は **[全般]** タブの SQL 文と同じトランザクションにコミットされます。

注: **[プリ SQL の自動コミット]** ボックスも、**[ポスト SQL の自動コミット]** ボックスもチェックしない場合、ステージの SQL 文はすべて 1 つのトランザクションにコミットされます。

ポスト SQL

すべてのレコードが処理された後に実行する 1 つ以上の SQL 文を入力します。ここで入力する SQL 文は、SQL Command ステージが終了してからデータフローが終了するまでの間に、実行あたり 1 回実行されます。

プリ SQL の使用例としては、レコードを処理した後にインデックスを構築することがあります。

ポスト SQL の自動コミット

[全般] タブの SQL Command がコミットされた後に、ポスト SQL のトランザクションにそのポスト SQL をコミットする場合は、このボックスをチェックします。

このボックスをチェックしない場合、ポスト SQL 文は **[全般]** タブの SQL 文と同じトランザクションにコミットされます。

注: **[プリ SQL の自動コミット]** ボックスも、**[ポスト SQL の自動コミット]** ボックスもチェックしない場合、ステージの SQL 文はすべて 1 つのトランザクションにコミットされます。

[実行時] タブ

[実行時] タブには、**[ステージオプション]** が表示され、ステージオプションのデフォルト値を自由に定義できます。

フィールド名

説明

ステージ オプション

このセクションには、このステージの SQL クエリで使用されるデータフロー オプションの一覧が表示され、そのすべてのオプションに対してデフォルト値を設定できます。**【名前】**列にはオプションが表示され、そのデフォルト値に対応する**【値】**列に入力できます。SQL クエリに変数を挿入すると、ステージオプションが有効になります。例えば、SQL フィールドのこのクエリにより、ステージオプションの CustomerID と InvoiceID が表示されるため、それぞれに個別のデフォルト値を設定できます。

```
Select * From [Sales].[CustomerTransactions]
where#{CustomerID}#{InvoiceID}
```

注：ここで設定したデフォルト値は、**【データフロー オプション】**ダイアログ ボックスの**【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】**セクションにも表示されます。このダイアログ ボックスでも、デフォルト値を変更できます。**【ステージオプション】**、**【データフロー オプション】**、および **Job Executor** で設定されたオプションのデフォルト値が競合する場合の優先順位は、**Job Executor** で指定された値 > **【データフロー オプション】**ダイアログ ボックスで定義された値 > **【ステージオプション】**で入力された値の順になります。

実行時の SQL Command の指定

この手順では、SQL Command の実行時オプションをサポートするようにデータフローを設定する方法と、そのための Job Executor の引数の指定方法について説明します。

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. SQL Command ステージを展開します。
6. SQL Command オプションを選択します。**【PreSqlCommand】**、**【SqlCommand】**、または **【PostSqlCommand】** を選択できます。

PreSqlCommand ステージに入力されるレコードが処理される前に実行する SQL 文を入力します。これらの SQL 文は、データフローの実行が開始されてから SQL Command ステージで最初のレコードが処理されるまでの間に、実行あたり 1 回実行されます。

プリ SQL の使用例としては、処理されるレコード用のテーブルを作成することがあります。

SqlCommand データフローで各レコードに対して実行する SQL 文です。

PostSqlCommand すべてのレコードが処理された後に実行する SQL 文です。これらの SQL 文は、SQL Command ステージが終了してからデータフローが終了するまでの間に、実行あたり 1 回実行されます。

例えば、ポスト SQL を使うと、レコードを処理した後にインデックスを構築できます。

選択した SQL Command オプション名が、**[オプション名]** フィールドと **[オプション ラベル]** フィールドに表示されます。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。

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、**[選択ステージ]** オプションを選択します。
9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **[OK]** をクリックします。
12. 必要に応じて、オプションの追加を続けます。
13. 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。
14. データフローを保存してエクスポートします。
15. 実行時に使用したい SQL 文を含むテキスト ファイルを作成します。
テキスト ファイルは次のようになります。

```
SqlCommand = UPDATE CustomersSET
ContactName='Alfred Schmidt'
```

```
City='Hamburg'
WHERE CustomerName='Alfreds Futterkiste';
```

この例では、SqlCommand は SQL Command ステージのオプション名の 1 つです。

16. Job Executor をコマンド ラインから実行するときは、-o 引数を使用します。

```
java -jar jobexecutor.jar -h "noipa019sh-11" -u "admin" -p "admin" -s
"8080" -o "options.txt" -j "FetchOracleData" -w
```

ファイル名 (options.txt) では、ステップ 14 で作成したテキスト アイルの名前を指定します。詳細については、[コマンド ラインからのジョブの実行](#) (247ページ)

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トする必要があります。ジョブをエクスポートするには、Enterprise Designer でジョブを開き、**[ファイル]>[エクスポート/アンエクスポートして保存]** 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 -uUserID -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 <i>delimiter</i>	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 <i>property file</i>	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

必須	引数	説明
		ブ プロパティ ファイルの使用 を参照してください。
いいえ	<code>-h host name</code>	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	<code>-i poll interval</code>	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	<code>-j job name</code>	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	<code>-n email list</code>	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	<code>-o property file</code>	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ト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 」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。
		<pre>OutputCasing=U UseStreetLevelMatching=N TaxKey=T Database.GTX=gsl</pre>
はい	<code>-p password</code>	ユーザのパスワードです。
いいえ	<code>-r</code>	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、 <code>-w</code> 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

必須 引数	説明
	<ul style="list-style-type: none"> • 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 <p>例:</p> <pre>MySimpleJob 4 succeeded 04/09/2019 14:50:47 04/09/2019 14:50:47 100 0 0 </pre> <p>情報は、<code>-d</code> 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。</p>
いいえ <code>-s port</code>	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 <code>-t timeout</code>	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 <code>-u user name</code>	ユーザのログイン名を指定します。
いいえ <code>-v</code>	詳細な出力を返します。
いいえ <code>-w</code>	<p>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。</p> <p><code>-w</code> 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</p>

必須	引数	説明
		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	<code>StageName=Protocol:FileName</code>	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 」を参照してください。
いいえ	<code>StageName:schema=Protocol:SchemaFile</code>	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 」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。

データフローの前後で SQL コマンドを実行

Execute SQL アクティビティは、プロセスフローの途中でデータベースを操作します。このアクティビティを使うと、Spectrum™ Technology Platform データフローまたは外部プログラムの実行前後で SQL 文を実行できます。例えば、**Execute SQL** アクティビティは、Spectrum™ Technology Platform データフローの実行前にインデックスを削除し、データフローの実行後にインデックスを再作成するのに使用できます。**Execute SQL** アクティビティを使って SQL 文を実行するには、プロセスフローを作成する必要があります。

注：プロセス フローの作成とスケジュール設定の手順については、『*Dataflow Designer* ガイド』を参照してください。

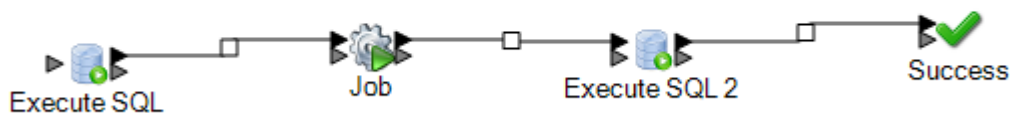
1. **Execute SQL** アクティビティをキャンバスにドラッグします。
2. **Execute SQL** アクティビティをダブルクリックします。
3. 使用するデータベース接続を選択します。

新しいデータベース接続を作成、あるいは既存のデータベース接続を変更または削除する必要がある場合は、**[管理]** をクリックします。

データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。

接続名	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。
データベース ドライバ	適切なデータベース タイプを選択します。
接続オプション	データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。

4. **[SQL 文]** ボックスに SQL 文を書き入れます。
デフォルトで、**[エラー時にプロセス フローを停止]** オプションがオンになっています。これは、プロセスフローが例外発生時に終了するということです。**[エラー時にプロセスフローを停止]** オプションをオフにした場合、例外が発生すると、プロセスフローは停止せず、例外がサーバー ログに記録されます。
5. プロセス フローで実行するアクションを追加します。
ジョブのアイコンをキャンバスにドラッグしてジョブを追加したり、プログラムの実行アイコンをキャンバスにドラッグして外部プログラムを追加したりできます。
6. 2つのアクティビティを接続します。
7. 必要であれば、さらに **Execute SQL** アクティビティを追加します。
手順 2 から 5 までのアクションを **Execute SQL** で行います。
8. プロセスフローで実行するすべてのジョブとプログラムの実行アクティビティおよび **Execute SQL** アクティビティを追加したら、成功アクティビティをキャンバスにドラッグし、プロセスフローの最後のアクティビティに接続します。



9. プロセス フローを実行します。

Transposer

Transposerは列を行に変換します。データの転置は、行のデータを列に変換する、Group Statistics ステージを使用したデータのピボット処理の逆の操作です。

Transposerを理解するために、次の例を考えます。テーブルには4四半期分の売上データが含まれており、生成されたすべての収益を追加して、最初の3四半期で達成された成長を分析することにします。これを実現するために、Transposerを使用して、転置された3四半期分の収益をすべて含む列を作成します。Transposerを使用して、異なる列で生成されたすべての収益を加えて1つの列にすると、別々の列でそれらを追加するよりも、パフォーマンスが向上する可能性があります。

次の表に、[Transposer] ダイアログ ボックスのオプションを示します。

オプション	説明
トランスポートされたフィールドのヘッダー	転置対象の列を格納する列のヘッダー名を入力します。この新しい列は、自動的にデータフローに追加されます。
トランスポートされた値のヘッダー	転置された列の値を格納する列のヘッダー名を入力します。この新しい列は、自動的にデータフローに追加されます。
トランスポートされたフィールドを保持	転置されたすべてのフィールドを出力で列として保持するには、このオプションにチェックを入れます。
フィールド名	入力ファイルのすべての列ヘッダーを表示します。

オプション 説明

タイプ

それぞれのフィールド (列ヘッダー) のデータ タイプを表示します。

入力ソース ファイルにおいて、転置される列は互換性のあるデータ タイプである必要があります。以下に、互換表を示します。マスにチェックが入っている場合は、互換性のあるデータ タイプであることを示します。

	整数	長整数	文字列	日付/時刻	倍精度浮動 小数点	大十進数	時刻	日付
Integer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>		
Long	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>	<input type="checkbox"/>		
文字列	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
日付/時刻			<input type="checkbox"/>	<input type="checkbox"/>				
Double	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>			
Bigdecimal	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>		
時間			<input type="checkbox"/>				<input type="checkbox"/>	
日付			<input type="checkbox"/>					<input type="checkbox"/>

転置

列に変換する各フィールドの横にあるこのボックスにチェックを入れます。列を転置せず
に出力で保持するためには、このチェック ボックスをクリアします。

Transposer の使用例

以下の入力データには、4 四半期分の店舗別売上が含まれています。Q1、Q2、Q3、
および Q4 は 4 つの四半期の売上を表しています (単位: 百万ドル)。

店舗 (米国)	Q1	Q2	Q3	Q4
ニューヨーク	100.00	200.10	300.00	400.00

店舗 (米国)	Q1	Q2	Q3	Q4
カリフォルニア	250.10	450.00	550.00	650.00
イリノイ	150.00	250.10	350.00	450.00

以下に示す事例は、ステージで提供されたオプションを使用した Transposer の動作を説明したものです。Quarter(四半期)はトランスポートされたフィールドのヘッダーの列名であり、Revenue (収益) は転置されたフィールド値の列名です。

事例 1

出力では、列 Q1、Q2、および Q3 を転置して Q4 はそのまま残したいとします。そのためには、転置される各列の横にある **[転置]** ヘッダーの下のボックスにチェックを入れます。すると、出力では Q1、Q2、および Q3 が行になるのに対し、Q4 は列のままになります。

店舗 (米国)	四半期	収益	Q4
ニューヨーク	Q1	100.00	400.00
ニューヨーク	Q2	200.10	400.00
ニューヨーク	Q3	300.00	400.00
カリフォルニア	Q1	250.10	650.00
カリフォルニア	Q2	450.00	650.00
カリフォルニア	Q3	550.00	650.00
イリノイ	Q1	150.00	450.00
イリノイ	Q2	250.10	450.00
イリノイ	Q3	350.00	450.00

事例 2

出力では、列 Q1 および Q2 を転置して Q3 および Q4 をそのまま残したいとします。また、転置されたフィールド (Q1 および Q2) のすべてを出力で列として保持したいとします。そのためには、**【トランスポーズされたフィールドを保持】** オプションと、転置する各列の横にある **【転置】** ヘッダーの下のボックスにチェックを入れます。これで、出力では Q1 および Q2 が行として表示されるのに対し、Q3 および Q4 は元の Q1 および Q2 と共に列として残されます。

店舗 (米 国)	四半期	収益	Q1	Q2	Q3	Q4
ニューヨーク	Q1	100.00	100.00	200.10	300.00	400.00
ニューヨーク	Q2	200.10	100.00	200.10	300.00	400.00
カリフォルニア	Q1	250.10	250.10	450.00	550.00	650.00
カリフォルニア	Q2	450.00	250.10	450.00	550.00	650.00
イリノイ	Q1	150.00	150.00	250.10	350.00	450.00
イリノイ	Q2	250.10	150.00	250.10	350.00	450.00

Unique ID Generator

この Unique ID Generator ステージでは、特定のレコードを識別するユニークキーを作成します。ユニーク ID は、トランザクションで名前および住所データをすべて送信するとは限らないもののトランザクションを同じレコードまたは連絡先に帰する必要があるデータウェアハウスインシアチブではきわめて重要です。ユニーク ID は、個人、世帯、企業、敷地レベルで実装されることがあります。Unique ID Generator は、ユニーク ID を作成する各種アルゴリズムを備えています。

ユニーク ID は、順番数字または日付/時間スタンプのいずれかに基づきます。また、オプションで、各種アルゴリズムを使用して、ID に付加するデータを生成することもできるので、より一意

である可能性の高い ID を作成できます。順番数字または日付/時間スタンプ ID は必須で、生成された ID から削除できません。

Unique ID Generator は、いずれかのキー生成アルゴリズムを使用して非ユニーク キーを生成するのに使用できます。非ユニーク モードでは、マッチングに使用するキーを作成できます。これは、データ ウェアハウスで、ディメンションにキーを追加済みで、新しいレコードが既存のレコードと一致するかどうかを確認するために新しいレコード用のキーを生成する場合に便利です。

この例では、入力の各レコードに出力で順番レコード ID を割り当てます。

レコード	レコード ID
John Smith	0
Mary Smith	1
Jane Doe	2
John Doe	3

Unique ID ステージによって、ユニーク ID を含む RecordID という名前のフィールドが作成されます。RecordID フィールドの名前は、必要に応じて変更できます。

ユニーク ID の定義

Unique ID Generator ステージでは、デフォルトの場合、連続した ID が作成されます。例えば、1 番目のレコードの ID は 0、2 番目のレコードの ID は 1、3 番目のレコードの ID は 2 になります。ユニーク ID の生成方法を変更する場合は、次の手順に従います。

1. Unique ID Generator ステージで、**[ルール]** タブの **[変更]** をクリックします。
2. ユニーク ID の生成方法を選択します。

オプション

説明

順番数字タグ開始位置

指定した数字を先頭の数字として、各レコードに増分値を割り当てます。例えば、先頭の数字として 0 を指定すると、1 番目のレコードの ID は 0、2 番目のレコードの ID は 1 になります。

注：この【ユニーク キー】では、【実行時インスタンス】(【実行時パフォーマンス】タブにある)を 1 より大きくしないようにしてください。重複する ID が作成されることがあります。

オプション	説明
-------	----

<p>順番数字タグをデータベース フィールドの値で開始する</p>	
-----------------------------------	--

オプション

説明

データベースフィールドから読み込んだ最大値を先頭の数字として、各レコードに増分値を割り当てます。この数字に 1 を加算した数字が 1 番目のレコードに割り当てられます。例えば、データベースフィールドから読み込まれる数字が 30 の場合、1 番目のレコードの ID は 31、2 番目のレコードの ID は 32 になります。

接続 使用するデータベース接続を選択します。使用できるデータベース接続は、Management Console の Connection Manager に定義されている接続によって異なります。新しいデータベースを作成、あるいは既存の接続を変更または削除する必要がある場合は、**[管理]** をクリックします。

データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。

接続名

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。

データベース ドライバ

適切なデータベース タイプを選択します。

接続オプション

データベースへの接続に使用するホスト、ポート、インスタンス、ユーザ名、およびパスワードを指定します。

テーブル表示 クエリの実行対象とするデータベース内のテーブルまたはビューを指定します。

[データベース] フィールド リストから列を選択して、ユニーク キーを生成します。

以下のデータタイプが、ユニーク ID の生成でサポートされています。

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)。

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)。

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。

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} ~ $(2 \cdot 2^{-52}) \times 2^{1023}$ 。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

オプション	説明
	範囲は、 $2^{-149} \sim (2-2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ となります。
日付/時間スタンプ	順番数字を割り当ててのではなく、日付/時間スタンプに基づいてユニークキーを作成します。
UUID	各レコードに対して、世界で一意 (universally unique) の 32 桁の識別キーを作成します。キーの桁は、ハイフンによって 5 つのグループに区切られて、8-4-4-4-12 という形式の合計 36 文字 (32 個の英数字と 4 つのハイフン) で表示されます。例: 123e4567-e89b-12d3-a456-432255330000
無効	このオプションは、アルゴリズムを使用して非ユニークキーを生成する場合にのみ選択します。

3. **[OK]** をクリックします。

アルゴリズムを使用したユニーク ID の拡張

Unique ID Generator は、各レコードに順番に数字を割り当てるか、各レコードに日時スタンプを生成して、各レコードにユニーク ID を生成します。オプションで、順番または日時ユニーク ID に追加情報を付加するアルゴリズムも使用できます。したがって、より複雑なユニーク ID や、真に一意である可能性の高いユニーク ID を作成できます。

1. Unique ID Generator ステージで、**[追加]** をクリックします。
2. **[アルゴリズム]** フィールドで、ID の追加情報の生成に使用するアルゴリズムを選択します。

Consonant 指定したフィールドから子音を削除して返します。
(子音)

Double Metaphone 文字の発音表記に基づくコードを返します。Double Metaphone は Metaphone アルゴリズムの改良版で、さまざまな言語に多数存在する不規則性を考慮しています。

Koeln ドイツ語で発音される名前に、音声によってインデックスを付けます。同じ発音を持つ名前を同じ表現にエンコードできるので、綴りに小さな相違があっても、マッチさせることができます。結果は常に一連の数字です。特殊文字およ

び空白は無視されます。このオプションは、**Soundex**の制限に対応するために作成されました。

- MD5** 128 ビットのハッシュ値を生成するメッセージ ダイジェスト アルゴリズム。このアルゴリズムは、データの一貫性の確認によく使用されます。
- Metaphone** 選択したフィールドを **Metaphone** コード化したキーを返します。**Metaphone** は、英語の発音を使用して単語をコード化するアルゴリズムです。
- Metaphone (スペイン語)** 選択したフィールドをスペイン語用に **Metaphone** コード化したキーを返します。この **Metaphone** アルゴリズムは、スペイン語の発音を使用して単語をコード化します。
- Metaphone 3** **Metaphone** アルゴリズムおよび **Double Metaphone** アルゴリズムを、より正確な子音および内部母音の設定で改良したもので、単語または名前の一致性を高く、または低くして、音声ベースで語を検索できるようにします。**Metaphone 3** では、音声エンコーディングの精度が **98%** に向上しています。このオプションは、**Soundex** の制限に対応するために作成されました。
- Nysiis** 近似の発音と正確な綴りをマッチさせ、同じように発音される単語にインデックスを付ける、音声コード アルゴリズム。**New York State Identification and Intelligence System** の一部です。例えば、住民のデータベースで誰かの情報を探しているとします。その人物の名前は **"John Smith"** のように聞こえますが、実際の綴りは **"Jon Smyth"** です。**"John Smith"** の完全一致を探す検索を実行した場合、返される結果はありません。しかし、**NYSIIS** アルゴリズムを使用してデータベースにインデックスを作成し、再度 **NYSIIS** アルゴリズムを使用して検索した場合は、正しいマッチが返されます。なぜなら、**"John Smith"** と **"Jon Smyth"** は、このアルゴリズムによってどちらも **"JAN SNATH"** というインデックスが付けられているからです。
- Phonix** 100 を越える変換ルールを適用することによって、名前文字列を単一の文字またはいくつかの文字のシーケンスに前処理します。これらのルールのうち **19** 個は文字がその文字列の先頭にある場合にのみ適用され、**12** 個はその文字列の中間にある場合にのみ適用され、**28** 個は文字列の終わりにある場合にのみ適用されます。変換された名前文字列は、開始文字とそれに続く **3** 桁 (ゼロおよび重複する数字を削除) で構成されるコードにエンコードされます。このオプションは、**Soundex** の制限に対応するために作成されました。このオプションは複雑なため、**Soundex** より遅くなります。
- Sonnex** このアルゴリズムは、文字の発音表記に基づいて、**2** つのフランス語の文字列間の類似性を判断します。
選択したフィールドを **Sonnex** コード化したキーを返します。
- Soundex** 選択したフィールドの **Soundex** コードを返します。**Soundex** は、単語の英語の発音に基づいて、固定長のコードを生成します。

部分文字列 選択したフィールドの指定した部分を返します。

3. **[フィールド名]** フィールドで、アルゴリズムを適用するフィールドを選択します。例えば、Soundex アルゴリズムを選択し、City という名前のフィールドを選択すると、City フィールドのデータに Soundex アルゴリズムを適用して ID が生成されます。
4. 部分文字列アルゴリズムを選択した場合、部分文字列で使用するフィールドの部分を指定します。
 - a) **[開始位置]** フィールドで、部分文字列を開始するフィールド内の位置を指定します。
 - b) **[長さ]** フィールドで、部分文字列に含める開始位置からの文字数を選択します。

例えば、LastName という名前のフィールドに次のデータが含まれているとします。

Augustine

開始位置を 3、終了位置を 6 に指定すると、次の部分文字列が作成されます。

gustin

5. **[ノイズ文字の削除]** ボックスをチェックすると、アルゴリズムを適用する前に、英数字以外の文字 (ハイフン、空白スペース、その他の特殊文字など) がフィールドからすべて削除されます。
6. Consonant (子音) および部分文字列アルゴリズムの場合、**[ソート入力]** ボックスをチェックすると、アルゴリズムを適用する前に、フィールドのデータをソートできます。その後、フィールド内の文字または語をアルファベット順にソートできます。
7. **[OK]** をクリックして、設定を保存します。
8. 他のアルゴリズムを追加して、より複雑な ID を作成する場合は、これらの操作を必要に応じて繰り返します。

注：ユニーク キーの定義は常に異なる色で表示され、削除できません。

非ユニーク ID の定義

Unique ID Generator は、いずれかのキー生成アルゴリズムを使用して非ユニーク キーを生成するのに使用できます。非ユニーク モードでは、マッチングに使用するキーを作成できます。これは、データ ウェアハウスで、ディメンションにキーを追加済みで、新しいレコードが既存のレコードと一致するかどうかを確認するために新しいレコード用のキーを生成する場合に便利です。

1. Unique ID Generator ステージで、**[ルール]** タブの **[変更]** をクリックします。
2. **[無効]** を選択します。

これで、ID 生成ルールのユニーク ID 部分が無効になります。このオプションを無効にすると、次の手順で選択するアルゴリズムのみが ID の作成に使用されます。つまり、ID の生成に使用するフィールドのデータが同じレコードには、すべて同じ ID が使用されます。ID は後でマッチングに使用できます。

3. **[OK]** をクリックします。
4. 警告が表示されたら、**[はい]** をクリックします。
5. Unique ID Generator ステージで、**[追加]** をクリックします。
6. **[アルゴリズム]** フィールドで、ID の追加情報の生成に使用するアルゴリズムを選択します。

Consonant 指定したフィールドから子音を削除して返します。
(子音)

Double Metaphone 文字の発音表記に基づくコードを返します。Double Metaphone は Metaphone アルゴリズムの改良版で、さまざまな言語に多数存在する不規則性を考慮しています。

Koeln ドイツ語で発音される名前に、音声によってインデックスを付けます。同じ発音を持つ名前を同じ表現にエンコードできるので、綴りに小さな相違があっても、マッチさせることができます。結果は常に一連の数字です。特殊文字および空白は無視されます。このオプションは、Soundex の制限に対応するために作成されました。

MD5 128 ビットのハッシュ値を生成するメッセージ ダイジェスト アルゴリズム。このアルゴリズムは、データの一貫性の確認によく使用されます。

Metaphone 選択したフィールドを Metaphone コード化したキーを返します。Metaphone は、英語の発音を使用して単語をコード化するアルゴリズムです。

Metaphone (スペイン語) 選択したフィールドをスペイン語用に Metaphone コード化したキーを返します。この Metaphone アルゴリズムは、スペイン語の発音を使用して単語をコード化します。

Metaphone 3 Metaphone アルゴリズムおよび Double Metaphone アルゴリズムを、より正確な子音および内部母音の設定で改良したもので、単語または名前の一致性を高く、または低くして、音声ベースで語を検索できるようにします。Metaphone 3 では、音声エンコーディングの精度が 98% に向上しています。このオプションは、Soundex の制限に対応するために作成されました。

Nysiis 近似の発音と正確な綴りをマッチさせ、同じように発音される単語にインデックスを付ける、音声コード アルゴリズム。New York State Identification and Intelligence System の一部です。例えば、住民のデータベースで誰かの情報を探しているとします。その人物の名前は "John Smith" のように聞こえますが、実際の綴りは "Jon Smyth" です。"John Smith" の完全一致を探す検索を実行した場合、返される結果はありません。しかし、NYSIIS アルゴリズムを使用し

てデータベースにインデックスを作成し、再度 NYSIIS アルゴリズムを使用し検索した場合は、正しいマッチが返されます。なぜなら、"John Smith" と "Jon Smyth" は、このアルゴリズムによってどちらも "JAN SNATH" というインデックスが付けられているからです。

- Phonix** 100 を越える変換ルールを適用することによって、名前文字列を単一の文字またはいくつかの文字のシーケンスに前処理します。これらのルールのうち 19 個は文字がその文字列の先頭にある場合にのみ適用され、12 個はその文字列の中間にある場合にのみ適用され、28 個は文字列の終わりにある場合にのみ適用されます。変換された名前文字列は、開始文字とそれに続く 3 桁 (ゼロおよび重複する数字を削除) で構成されるコードにエンコードされます。このオプションは、Soundex の制限に対応するために作成されました。このオプションは複雑なため、Soundex より遅くなります。
- Sonnex** このアルゴリズムは、文字の発音表記に基づいて、2 つのフランス語の文字列間の類似性を判断します。
選択したフィールドを Sonnex コード化したキーを返します。
- Soundex** 選択したフィールドの Soundex コードを返します。Soundex は、単語の英語の発音に基づいて、固定長のコードを生成します。

部分文字列 選択したフィールドの指定した部分を返します。

7. **[フィールド名]** フィールドで、アルゴリズムを適用するフィールドを選択します。例えば、Soundex アルゴリズムを選択し、City という名前のフィールドを選択すると、City フィールドのデータに Soundex アルゴリズムを適用して ID が生成されます。
8. 部分文字列アルゴリズムを選択した場合、部分文字列で使用するフィールドの部分を指定します。
 - a) **[開始位置]** フィールドで、部分文字列を開始するフィールド内の位置を指定します。
 - b) **[長さ]** フィールドで、部分文字列に含める開始位置からの文字数を選択します。

例えば、LastName という名前のフィールドに次のデータが含まれているとします。

Augustine

開始位置を 3、終了位置を 6 に指定すると、次の部分文字列が作成されます。

gustin

9. **[ノイズ文字の削除]** ボックスをチェックすると、アルゴリズムを適用する前に、英数字以外の文字 (ハイフン、空白スペース、その他の特殊文字など) がフィールドからすべて削除されます。
10. **Consonant (子音)** および部分文字列アルゴリズムの場合、**[ソート入力]** ボックスをチェックすると、アルゴリズムを適用する前に、フィールドのデータをソートできます。その後、フィールド内の文字または語をアルファベット順にソートできます。

11. **[OK]** をクリックして、設定を保存します。
12. 他のアルゴリズムを追加して、より複雑な ID を作成する場合は、これらの操作を必要に応じて繰り返します。

注：ユニーク キーの定義は常に異なる色で表示され、削除できません。

Write to Cache

Write to Cache では、データフローからの出力をグローバル キャッシュ内にロードし、データを Query Cache ステージからの検索で使えるようにします。データ検索にグローバル キャッシュを使用すると、データベースに対する検索を行うよりもパフォーマンスが向上します。

グローバル キャッシュは、メモリ内に存在するシステム全体の共有キャッシュです。キャッシュを複数のデータフローで使用できるようにする場合や、データが頻繁には変化せず比較的静的な状態の、およびストレージが制限されていない場合は、グローバル キャッシュを選択します。グローバル キャッシュは、書き込みが一度しかできないので静的です。このキャッシュは、いったん作成されると更新ができません。

注：Write to Cache では、データフローを実行するたびにキャッシュを上書きします。

全般

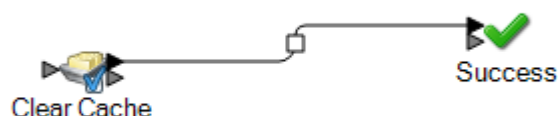
オプション名	説明
キャッシュ名	キャッシュに与える名前を指定します。システムにキャッシュが既に存在する場合は、そのリストが表示され、既存のキャッシュに新しいデータを設定する際にその 1 つを選択できます。新しいキャッシュを作成するには、そのキャッシュの名前を入力します。名前は文字で始まっている必要があります。アンダースコアを使用できますが、その他の特殊文字はどれも使用できません。名前の一部に数字を使用することはできません。
キャッシュ フィールド	この列には、キャッシュで使用されるフィールド名のリストが表示されます。フィールド名を変更する場合は、そのフィールド名をクリックして新しい名前を入力します。

オプション名	説明
ステージ フィールド	この列には、データフローで使用されるフィールド名のリストが表示されます。これらのフィールド名は変更できません。
タイプ	この列には、各フィールドのデータ タイプのリストが表示されます。
含める	フィールドをキャッシュに書き込むには、この列にあるボックスにチェックを入れます。フィールドをキャッシュに書き込まない場合は、ボックスをクリアします。
キー フィールド	<p>フィールドを Query Cache ステージでキーとして使用する場合は、この列にあるボックスにチェックを入れます。例えば、AccountNumber という名前のデータフロー フィールドがあって、Query Cache ステージでこの AccountNumber フィールド内のマッチング値を求めるクエリによってデータを検索する場合は、AccountNumber フィールドの [キー フィールド] 列にあるボックスにチェックを入れます。</p> <p>キー フィールドとして指定したフィールドは、Query Cache ステージでの選択でキー フィールドとして使用できます。</p>

グローバル キャッシュのクリア

グローバルキャッシュをクリアするには、プロセスフローを作成し、実行する必要があります。プロセスフローにはキャッシュのクリアアクティビティを含める必要があります。キャッシュのクリアアクティビティは、グローバルキャッシュをクリアしますが、削除はしません。また、プロセスフローのスケジュールを設定してキャッシュを自動的にクリアすることもできます。

注：プロセスフローの作成とスケジュール設定の手順については、『*Dataflow Designer ガイド*』を参照してください。



グローバル キャッシュのデータを手動でクリアするには、以下の手順に従います。

1. **[キャッシュのクリア]** アクティビティをキャンバスにドラッグします。
2. **[成功]** アクティビティをキャンバスにドラッグします。
3. 2つのアクティビティを接続します。
4. **[キャッシュのクリア]** アクティビティをダブルクリックします。
5. キャッシュを選択します。複数のキャッシュを選択してデータをクリアすることもできます。
Write to Cache ステージで作成したキャッシュが、キャッシュのクリア アクティビティに一覧で表示されます。
6. プロセス フローを実行します。

Write to DB


Write to DB ステージでは、データフローの出力をデータベースに書き込みます。ステージは、date データタイプのすべての値を String 値として書き出します。これは、Spectrum で使用されるデフォルト ドライバである *jTDS* ドライバの動作です。すべての date データタイプの値をそのまま処理するには、Microsoft の JDBC ドライバを使用します。

注：Write to DB の複数の実行時インスタンスを使用することで、パフォーマンスを大きく改善できます。複数の実行時インスタンスを指定するには、**[実行時]** ボタンをクリックします。

[全般] タブ を設定する


1. **[接続]** ドロップダウン リストから、使用するデータベースへの接続を選択します。
2. 新しいデータベース接続を作成するには、**[管理]** をクリックします。データベース接続の作成の詳細については、「[Database Connection Manager \(271ページ\)](#)」を参照してください。

注：このオプションは **Enterprise Designer** によってのみ使用できます。

3. データベースからテーブルまたはビューを選択するには、参照ボタン  をクリックして、使用するテーブルまたはビューに移動します。

テーブルを参照して選択すると、**[データベース フィールド]**、**[ステージ フィールド]**、**[データ タイプ]** などの **[テーブル スキーマ]** が表示されます。テーブルの **[プレビュー]** も使用可能です。

注：SQL データベースに出力する場合は、複数のテーブルを参照するビューには出力できません。これは SQL Server の制限によります。

4. データベースに新しいテーブルを作成するには、**[テーブルの作成]**  をクリックし、表示されるポップアップで **[テーブルの所有者]** を選択し、**[テーブル名]** を指定します。

注：テーブル名は大文字と小文字を区別します。

注：**Write to DB** ステージに関連付けられた入力ステージ (**Read from File** や **Read from DB** など) がない場合、次のようなエラー メッセージが表示されます: テーブルスキーマが定義されていないと、テーブルを作成できません。このステージに上流フィールドが定義されていることを確認してください。

5. テーブルスキーマで、以下の詳細を指定します。
 - a. 対応する **[プライマリ キー]** のチェックボックスをオンにして、プライマリ キーを指定します。
 - b. **[含める]** のチェックボックスをオンにして、新しいテーブルを出力先とするフィールドを指定します。
 - c. 文字列のデータ タイプの場合は、**[太さ]** 列でフィールドの長さを指定します。

注：デフォルトは 512 です。

- d. **[Null を許可]** 列のチェックボックスがオンで **[入力フィールド]** に NULL 値が含まれている場合、データフローはデータベースに NULL 値を書き込みます。
- e. 対応する **[出力フィールド]** の値を変更することによって、列名を編集できます。

[テーブルの作成] ボタンは、次のデータベースでのテーブル作成をサポートしています。

- Axion
- DB2
- Derby または Cloudscape
- Firebird
- HSQLDB
- Interbase
- MaxDB または SapDB
- McKoi
- MySQL
- Oracle
- PostgreSQL
- SQL Server
- Sybase

注：DB2 データベースでは、テーブルを作成しようとして、ページサイズがすべての文字列の列の長さの合計より小さい場合、"コンテンツから本体を作成できません。シリアル化可能クラスはブローカに使用できません" というエラーが表示されます。

6. **[OK]** ボタンをクリックして**[テーブルの作成]** ポップアップを閉じ、**[Write to DB オプション]** に戻ります。
7. **[テーブル スキーマ]** の **[ステージ フィールド]** 列で、データベースに書き込むフィールド名を、**[データベース フィールド]** 列に対応させる形で指定できます。
8. **[含める]** のチェックボックスをオンにして、出力先のフィールドを選択します。

注：パフォーマンスが低下するのを防ぐため、データベース テーブルにはソートされたインデックスまたはキーが必要です。

[実行時] タブの設定

オプション名	説明
書き込みモード	データベース書き込みのアクションを指定します。
挿入	新しいレコードをデータベースに挿入します。既存のレコードは更新しません。これがデフォルト設定です。
更新	データベース内の既存のレコードを更新します。新しいレコードは挿入しません。 注： [更新] を選択する場合は、入力テーブルで使用されているプライマリ キー列の名前が、出力テーブルでのプライマリ キー列の名前と一致している必要があります。プライマリ キー列の名前が入力と一致していないテーブル、またはプライマリ キー列が定義されていないテーブルを更新しようとした場合、更新は行われません。
更新できない場合は挿入	該当レコードが存在しない場合は、新しいレコードをデータベースに挿入し、存在する場合は既存のレコードを更新します。
一括確定	このオプションをオンにすると、所定の数のレコードが処理されたとき、データベースへの変更が確定されます。デフォルトでは、このオプションはオフになっていて、1レコード処理されるごとに変更が確定されます。このオプションをオンにすると、Write to DB ステージのパフォーマンスが大きく向上します。

オプション名	説明
バッチ サイズ	<p>[一括確定] オプションをオンにする場合、データベースに対して一括で確定するときのレコード数を指定します。デフォルト値は 1,000 です。7.0 およびそれ以前の Spectrum™ Technology Platform で作成されたデータフローの場合、デフォルト値は 100 です。</p> <p>バッチ サイズを大きくしてもロード パフォーマンスは必ずしも改善しません。バッチ サイズの設定に当たっては、次の要因を検討してください。</p> <ul style="list-style-type: none">• 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。
最終確定	<p>このオプションを選択すると、すべてのレコードがデータベースに転送された後、データベース操作が確定されます。</p>
一括確定数	<p>一定のレコード数ごとに確定を行うための値を指定します。データベースに対するレコードの確定は、(一括確定数*バッチサイズ)に相当する数のレコードがデータベースに転送されるたびに行われます。例えば、[バッチサイズ]が 1000、[一括確定数]が 3 の場合、3000 レコードがデータベースに転送されるたびに確定が行われます。</p>
データを挿入する前にテーブルを切り捨てる	<p>このオプションをオンにすると、データベースへの書き込みの前にテーブル内のデータがすべてクリアされます。</p>

オプション名

説明

テーブルが既に存在する場合は破棄して作成し直す

このオプションをオンにすると、データフローの出力をテーブルに書き込む前にテーブルを削除して作成し直します。このオプションは、テーブルのスキーマをデータフローからのフィールドに一致させ、余分なスキーマ情報を含まないようにしたい場合に便利です。

削除して作成し直すテーブルは、**[全般]** タブの **[テーブル/ビュー]** フィールドで指定されたものです。例えば、**[テーブル/ビュー]** フィールドで "Customers" テーブルを指定し、**[テーブルが既に存在する場合は破棄して作成し直す]** を選択した場合、"Customers" テーブルはデータベースから削除され、テーブルに書き込まれる実際のフィールドに一致するスキーマを持つ "Customers" という名前の新しいテーブルが作成されます。

Database Connection Manager

注：このオプションは **Enterprise Designer** によってのみ使用できます。

Database Connection Manager を使用すると、登録されているデータベース接続を管理できます。接続を追加、変更、削除、およびテストするには:

1. **[Write To DB オプション]** ダイアログ ボックスで、**[管理]** をクリックします。
2. **[追加]**、**[変更]**、または **[削除]** をクリックします。
3. データベース接続を追加または変更する場合は、次のフィールドに必要な値を指定します。
 - 接続名 — 新しい接続の名前を入力します。
 - データベース ドライバ — 適切なデータベースの種類を選択します。
 - 接続オプション — すべてのオプション (通常はホスト、ポート、インスタンス、ユーザ名、およびパスワード) を指定します。

注：接続をテストするには、**[テスト]** をクリックします。

4. データベース接続を削除する場合は、削除する接続を選択し、**[削除]** をクリックします。
5. 設定が終わったら、**[適用]** ボタンをクリックします。

Write to DB におけるエラー処理の設定

Write to DB ステージにはエラー ポートがあります。エラー ポートを使用すると、データベースにレコードを書き込む際にプライマリ キー制約違反や一意性制約違反などのデータベースエラーとなるレコードをフィルタして除外できます。このようなレコードはデータフローの別のパスにルーティングされ、それ以外のレコードは正常に確定されます。例えば、100 件のレコードを処理する際に、レコード 4、23、および 56 がデータベースエラーとなる場合、これら 3 件のレコードはエラー ポートを介してルーティングされ、その他 97 件のレコードはデータベースにコミットされます。

注：エラー ポートを使うかどうかは任意です。エラー ポートを使わない場合は、レコードが 1 つでもエラーになるとジョブは失敗に終わります。

- パレットから、エラーレコードを処理するタイプステージ (Write to File など) を選択してキャンバスにドラッグします。ステージを選択するオプションがいくつかあります。
 - 失敗したレコードをファイルに書き込むには、Write to File、Write to XML、または Write to Variable Format File のいずれかをキャンバスにドラッグします。
 - 失敗したレコードを単に破棄するには、Write to Null をキャンバスにドラッグします。
- Write to DB のエラー ポートを、失敗したレコードを処理するステージに接続します。

次の例では、Write to DB のエラー ポートが Write to File ステージに接続されています。この例では、データベースに書き込む際にエラーとなるレコードは、データベースではなく、Write to File ステージで指定されたファイルに書き込まれます。



データフローを実行すると、エラーとなるレコードは、エラー ポートを介してルーティングされます。エラーポートからのレコードには、Write to DB で指定されたフィールドと、次のフィールドが含まれます。

Error.code このフィールドには、データベースから返される数値エラーコードが含まれます。例えば、ORA-00001: unique constraint ANKUSH.SYS_C0010018) violated というエラーがある場合、Error.code フィールドの値は 1 になります。エラーコードの一覧については、データベースソフトウェアのマニュアルを参照してください。

Error.Message	このフィールドには、データベースから返されるエラーメッセージが含まれます。例:ORA-01034 ORACLE not available。この場合、ORACLE not available は、 Error.Message フィールドの値です。エラーメッセージの一覧については、データベースソフトウェアのマニュアルを参照してください。
Error.SQLState	このフィールドには、エラーの原因に関する詳細情報を示す SQLSTATE コードが含まれます。SQLSTATE コードの一覧については、データベースソフトウェアのマニュアルを参照してください。
Timestamp	Spectrum™ Technology Platform サーバーでエラーが発生したときの日時。
ユーザ名	データフローを実行した Spectrum™ Technology Platform ユーザの名前。

Write to File

Write to File は、データフロー出力をフラット ファイルに書き出します。

- 異なるフォーマットのレコードを書き出す場合は、「[Write to Variable Format File \(310ページ\)](#)」を参照してください。
- レコードを XML ファイルに書き出す場合は、「[Write to XML \(321ページ\)](#)」を参照してください。

ヒント: ソースをコピーしてデータフローにシンクとして貼り付けると、ファイルをすばやく設定し、ソースに定義したフィールドと同じフィールドを使用できます。

前提条件: ファイルシステム接続タイプ (FTP、クラウド、Amazon AWS S3、HDFS など) にファイルを書き出すには、以下の手順を実行します。

- Management Console** または **Metadata Insights** を用いてこれらのファイルサーバーへの接続を作成します。詳細については、「[接続の定義 \(15ページ\)](#)」セクションを参照してください。
- [ファイル プロパティ]** タブの **[ファイル名]** フィールドを使用して必須のファイルパスを選択します (以下を参照)。

[ファイル プロパティ] タブ

フィールド名	説明
サーバ名	<p>入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。</p>
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。</p> <p>HDFS ファイル サーバーへのファイルの書き出しでは、以下の圧縮形式がサポートされています。</p> <ol style="list-style-type: none"> 1. GZIP (.gz) 2. BZIP2 (.bz2) <p>注: ファイルの書き出し時に使用する望ましい圧縮形式を示すため、ファイル名に適切な拡張子を含めるようにしてください。</p> <p>重要: なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
レコード タイプ	<p>ファイル内のレコードのフォーマット。次のいずれかを選択します。</p> <p>行順次 ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキスト ファイル。</p> <p>固定長 ファイル内の各レコードの長さ (文字数) が一定で、レコード内の各フィールドの開始文字位置と終了文字位置が固定しているテキスト ファイル。</p> <p>区切り記号付き ファイル内の各レコードが復帰または改行 (CR または LF) などの行末 (EOL) 文字で区切られ、レコード内の各フィールドがカンマ (,) などの特定の文字で区切られているテキスト ファイル。</p>

フィールド名	説明
文字エンコーディング	テキスト ファイルのエンコーディング。次のいずれかを選択します。 CP1252 このエンコーディングは Windows-1252 文字セット、または単純に Windows 文字セットとも呼ばれています。これは ISO-8859-1 の上位クラスであり、128 ~ 159 のコード範囲を使用して、ISO-8859-1 文字セットに含まれていない追加の文字を表示します。 UTF-8 すべての Unicode 文字をサポートし、かつ ASCII との下位互換性があります。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 UTF-16 すべての Unicode 文字をサポートします。しかし、ASCII との下位互換性はありません。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。 US-ASCII 英語のアルファベット順に従う文字エンコーディング。 UTF-16BE ビッグエンディアン UTF-16 エンコーディング (下位アドレスが上位バイトとなるようにシリアル化)。 UTF-16LE リトルエンディアン UTF-16 エンコーディング (下位アドレスが下位バイトとなるようにシリアル化)。 ISO-8859-1 主として西ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-1 とも呼ばれます。 ISO-8859-3 主として南ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-3 とも呼ばれます。 ISO-8859-9 主としてトルコ語で使われる ASCII 文字エンコーディング。Latin-5 とも呼ばれます。 CP850 西ヨーロッパの言語を書くための ASCII コード ページ。 CP500 西ヨーロッパの言語を書くための EBCDIC コード ページ。 Shift_JIS 日本語のための文字エンコーディング。 MS932 NEC 特殊文字、NEC 選定 IBM 拡張文字、IBM 拡張文字を含めた Microsoft の拡張版 Shift_JIS 文字コード。 CP1047 Latin-1 文字セット全体を含む EBCDIC コード ページ。

フィールド名

説明

フィールド区切り文字

区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは (|) 記号がフィールド区切り文字として使われています。

```
7200 13TH ST|MIAMI|FL|33144
```

フィールド区切り文字として定義できるのは次の文字です。

- スペース
- タブ
- カンマ
- ピリオド (.)
- セミコロン
- パイプ (|)

これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。

Text qualifier

区切り記号付きファイル内のテキスト値を囲むのに使用する文字。

例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。

```
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
```

テキスト修飾子として定義できるのは次の文字です。

- 一重引用符 (')
- 二重引用符 (")

これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。

フィールド名

説明

レコード区切り文字

順次ファイルまたは区切り記号付きファイル内のレコードを区切るのに使用する文字を指定します。**[デフォルトの EOL を使用]** チェック ボックスをオンにすると、このフィールドは使用できません。

使用できるレコード区切り文字の設定は次のとおりです。

Unix (U+000A) 改行 (LF) 文字でレコードを区切ります。これは Unix システムの標準のレコード区切り文字です。

Macintosh (U+000D) 復帰 (CR) 文字でレコードを区切ります。これは Macintosh システムの標準のレコード区切り文字です。

Windows (U+000D U+000A) 復帰改行 (CR+LF) でレコードを区切ります。これは Windows システムの標準のレコード区切り文字です。

これ以外の文字がレコード区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をレコード区切り文字として選択してください。

デフォルトの EOL を使用

Spectrum™ Technology Platform サーバーが実行されているオペレーティングシステムのデフォルトの行末 (EOL) 文字をファイルのレコード区切り文字として使用します。

ファイルの EOL 文字がサーバーのオペレーティング システムで使われているデフォルトの EOL 文字と異なる場合は、このオプションをオンにしないでください。例えば、ファイルで Windows の EOL が使われていて、サーバーの動作プラットフォームが Linux の場合は、このオプションをオンにしないでください。代わりに、**[レコード区切り文字]** フィールドで **[Windows]** オプションを選択します。

レコード長

固定長ファイルでは、個々のレコードの文字数を指定します。

行順次ファイルでは、ファイル内の最も長いレコードの長さ (文字数) を指定します。

最初の行はヘッダ レコード

区切り記号付きファイルの先頭レコードの内容がデータではなくヘッダ情報であるかどうかを指定します。

次のファイル スニペットは、先頭レコードのヘッダ行の例です。

```
"AddressLine1"|"City"|"StateProvince"|"PostalCode"
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
"One Global View"|"Troy"|"NY"|12180
```

フィールド名	説明
フィールド数が定義よりも少ないレコードを、形式誤りとみなす	区切り記号付きファイルのレコードのうち、 【フィールド】 タブに定義されている数よりもフィールド数が少ないレコードを形式に誤りのあるレコードとみなします。
インポート	ファイルレイアウト定義、エンコーディング設定、およびソートオプションを設定ファイルからインポートします。設定ファイルは、同じ入力ファイル、あるいは操作しているファイルと同じレイアウトを持つファイルを使用した、別の Read from File ステージまたは Write to File ステージから設定をエクスポートすることによって作成されます。
エクスポート	ファイルレイアウト定義、エンコーディング設定、およびソートオプションを設定ファイルに保存します。その後、同じ入力ファイル、または現在操作しているファイルと同じ特性を持つファイルを使用する他の Read from File ステージまたは Write to File ステージにこれらの設定をインポートできます。 Job Executor で設定ファイルを使用して、実行時にファイル設定を指定することもできます。 設定ファイルの詳細については、 ファイル定義設定ファイル (178ページ) を参照してください。

[【フィールド】 タブ](#)

[フィールド] タブでは、ファイル内の各フィールドの名前と位置を定義します。また、固定長ファイルと行順次ファイルについては、さらにフィールドの長さを定義します。詳細については、[以下を参照してください](#)。

[区切り記号付き出力ファイルのフィールドの定義](#) (280ページ)

[行順次ファイルまたは固定長ファイルのフィールドの定義](#) (282ページ)

[【ソート フィールド】 タブ](#)

[ソート フィールド] タブでは、出力ファイルに書き出される前の出力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。詳細については、「[出力レコードのソート](#) (285ページ)」を参照してください。

[実行時] タブ

オプション名	説明
ファイル名	ここでは、 [ファイル プロパティ] タブで定義したファイルが表示されます。
複数ファイルの生成	<p>1つのファイルにすべてのレコードを書き込むのではなく、レコードをいくつかのファイルに書き込む場合は、このオプションを選択します。各レコードを書き込むファイルは、レコード自体に指定されます。各レコードには、レコードを書き込むファイルのファイル名または完全ファイルパスのいずれかを指定するフィールドが含まれている必要があります。例えば、(各種グループの) 各社の株価をすべてのクライアントに個別に送信する場合、この機能を使用すると、各社の株価を個別のファイルに書き込んで、それらのファイルを各クライアントに送信できます。[複数ファイルの生成] オプションを有効にする場合は、Spectrum™ Technology Platform サーバー上または FTP サーバー上のいずれかにある出力ファイルを指定する必要があります。FTP サーバー上のファイルにデータを書き込む場合は、Management Console を使用してファイル サーバーへの接続を定義する必要があります。</p> <p>注：[ファイル パス フィールド] で選択する列のレコードは、ソート順になっている必要があります。レコードにファイル名または完全ファイルパスのいずれかが含まれている場合は、この機能を使用します。</p>
ファイル パス フィールド	レコードを書き込むファイルのパス (ファイル名または完全ファイルパスのいずれか) を含むフィールドを選択します。このフィールドは、 [複数ファイルの生成] を選択している場合にのみ有効になります。
書き込みモード	<p>データフローの出力をファイルの最後に追加するか、ファイル内の既存のデータを削除した後に出力を書き込むかを指定します。</p> <p>Overwrite データフローが実行されるごとに、出力ファイル内の既存のデータを置き換えます。</p> <p>追加 データフローの出力をファイルの最後に追加します。ファイルの既存のデータは消去しません。</p>

区切り記号付き出力ファイルのフィールドの定義

Write to File ステージの【フィールド】タブでは、ファイルの各フィールドの名前と位置、またファイルの種類によってはフィールドの長さも定義します。【ファイル プロパティ】タブで出力ファイルを定義したら、フィールドを定義できます。

出力ファイルにヘッダ レコードが含まれている場合は、【再生成】をクリックするだけでフィールドを簡単に定義できます。

フィールドの位置、長さ、およびデータ タイプにデフォルト値を設定してフィールドを定義するには、【クイック追加】をクリックし、追加するフィールドを選択します。

入力ファイルにヘッダ レコードが含まれていない場合、またはフィールドを手動で定義したい場合は、以下のステップを続けます。

1. 【追加】をクリックします。
2. 【名前】フィールドで、追加したいフィールドを選択します。
3. 【タイプ】フィールドで、データフローから入力されるフィールドのデータ タイプを選択します。

Spectrum™ Technology Platform では、以下のデータ タイプがサポートされています。

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。

bytearray バイトの配列 (リスト)。

注: bytearray は REST サービスの入力としてはサポートされていません。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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)。

list 厳密に言えば、リストはデータタイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。

```
<Names>
  <Name>John Smith</Name>
  <Name>Ann Fowler</Name>
</Names>
```

XML のリスト データ タイプが複数の値で構成される単純データタイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データタイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。

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。

4. 日付、時間、または数値データタイプを選択した場合は、デフォルトの日付/時間形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォルトの形式は、Management Console のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、Enterprise Designer のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]** を選択したままにします。別の形式を指定するには、**[カスタム]** を選択し、以下の手順に従います。

- a) **[ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータタイプによって異なります。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。

選択した形式の例が **[形式]** フィールドの右側に表示されます。

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](#)（331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](#)（334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. **[追加]** をクリックします。

出力ファイルのフィールドを定義したら、フィールドのコンテンツとレイアウトを編集できます。

オプション名	説明
追加	フィールドを出力に追加します。現在のレイアウトの末尾にフィールドを追加できます。また、既存の位置にフィールドを挿入することもできます。その場合、残りのフィールドの位置は適宜調整されます。
変更	フィールドの名前とタイプを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されたフィールドの順序を変更します。

行順次ファイルまたは固定長ファイルのフィールドの定義

Write to File ステージの**[フィールド]** タブでは、ファイルの各フィールドの名前と位置、またファイルの種類によっては、さらにフィールドの長さも定義します。**[ファイル プロパティ]** タブで出力ファイルを定義したら、フィールドを定義できます。

フィールドの位置、長さ、およびデータ タイプにデフォルト値を設定してフィールドを定義するには、**[クイック追加]** をクリックし、追加するフィールドを選択します。

データフローで使われているフィールドのリストからフィールドを手動で追加するには、以下の手順に従います。

1. **[追加]** をクリックします。
2. **[名前]** フィールドで、追加したいフィールドを選択します。

3. **[タイプ]** フィールドで、データフローから入力されるフィールドのデータ タイプを選択します。

Spectrum™ Technology Platform では、以下のデータ タイプがサポートされています。

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。

bytearray バイトの配列 (リスト)。

注: bytearray は REST サービスの入力としてはサポートされていません。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。

list 厳密に言えば、リストはデータ タイプではありません。しかし、フィールドが階層データを含む場合、"リスト" フィールドとして扱われます。Spectrum™ Technology Platform では、リストは複数の値で構成されるデータのコレクションです。例えば、Names フィールドには名前の値のリストを含めることができます。これは、XML 構造では次のように表すことができます。

```
<Names>
  <Name>John Smith</Name>
  <Name>Ann Fowler</Name>
</Names>
```

XML のリスト データ タイプが複数の値で構成される単純データ タイプであるのに対し、Spectrum™ Technology Platform のリスト データ タイプは XML の複合データ タイプに似ているという点で、Spectrum™ Technology Platform のリスト データ タイプは XML スキーマのリスト データ タイプと異なることに注意してください。

- 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。

4. 日付、時間、または数値データ タイプを選択した場合は、デフォルトの日付/時間形式を使用することも、この特定のフィールド用に別の形式を指定することもできます。デフォルトの形式は、Management Console のタイプ変換オプションに設定されているシステムのデフォルトの形式であるか、Enterprise Designer のタイプ変換オプションに指定されているデータフローのデフォルトの形式です。有効な形式が表示されます。デフォルトの形式を使用するには、**[デフォルト]** を選択したままにします。別の形式を指定するには、**[カスタム]** を選択し、以下の手順に従います。

- a) **[ロケール]** フィールドで、使用する形式規則のある国を選択します。**[形式]** 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、日付データの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) **[形式]** フィールドで、データの形式を選択します。形式は、フィールドのデータ タイプによって異なります。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。

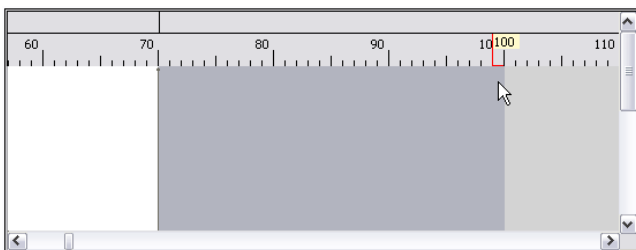
選択した形式の例が **[形式]** フィールドの右側に表示されます。

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** (331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** (334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. **[開始位置]** フィールドと **[長さ]** フィールドは、データフロー内のデータと、追加してあるフィールドの数に応じて自動的に設定されます。

6. **[追加]** をクリックします。

また、最初にフィールドの開始位置と長さを定義してフィールドを追加することもできます。これを行うには、**サンプル ファイル** で、フィールドを開始したい位置をクリックして左ヘドラッグします。これで次のようにフィールドがハイライト表示されます。



出力ファイルのフィールドを定義したら、フィールドのコンテンツとレイアウトを編集できます。**【開始位置の再計算】** オプションを使用すると、出力ファイル内のフィールドを変更、移動、または削除したときにフィールドの位置を **Write to File** ステージで再計算できます。位置を再計算せず、出力ファイルの編集後もフィールドを既存の位置のままにする場合は、このチェックボックスをオフにします。

オプション名	説明
追加	フィールドを出力に追加します。
変更	フィールドの名前、タイプ、開始位置、長さを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されたフィールドの順序を変更します。

出力レコードのソート

Write to File ステージで、**【ソート フィールド】** タブを使い、出力ファイルに書き出される前の出力レコードのソートに使うフィールドを定義します。ソートを行うかどうかはオプションです。

1. **Write to File** で、**【ソート フィールド】** タブをクリックします。
2. **【追加】** をクリックします。
3. **【フィールド名】** 列のドロップダウン矢印をクリックし、ソートに使うフィールドを選択します。選択できるフィールドは、データフロー内のフィールドによって異なります。
4. **【順序】** 列で、**Ascending** または **Descending** を選択します。
5. ソートに使用するすべての出力フィールドを追加するまでこれを繰り返します。ソート順序を変更するには、移動するフィールドの行をハイライト表示して、**【上へ】** または **【下へ】** をクリックします。
6. システムのデフォルトのソート パフォーマンス オプションの設定は、**Management Console**で行います。システムの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場合は、**【詳細設定】** をクリックします。**【詳細オプション】** ダイアログ ボックスには、次のソート パフォーマンス オプションがあります。

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。

$$\text{NumberOfRecords} \times 2 \div \text{InMemoryRecordLimit} = \text{NumberOfTempFiles}$$

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。 $(\text{InMemoryRecordLimit} \times \text{MaxNumberOfTempFiles} \div 2) \geq \text{TotalNumberOfRecords}$

ファイル定義設定ファイル

ファイル定義設定ファイルには、Read from File ステージまたは Write to File ステージからエクスポートされたファイルレイアウト、エンコーディング、およびソート オプションが含まれます。ファイル定義設定ファイルを Read from File または Write to File にインポートして、そのステージのオプションを手動で指定する代わりに、すばやく設定することができます。

ファイル定義設定ファイルを作成する最も簡単な方法は、**Read from File** または **Write to File** を使用してファイル設定を指定し、**[エクスポート]** ボタンをクリックして、ファイル定義設定ファイルを生成することです。

参考として、ファイル定義設定ファイルのスキーマを以下に示します。XML ファイル内の各要素はタイプを持ちます。そのタイプが文字列または整数以外の場合、許容できる値が表示されています。これらの値は、そのステージのダイアログ ボックスのオプションに直接対応しています。例えば、**FileTypeEnum** 要素は **[ファイル プロパティ]** タブの **[レコード タイプ]** フィールドに対応しており、スキーマに、**linesequential**、**fixedwidth**、**delimited** の 3 つの値が表示されます。

注： **LineSeparator** フィールド、**FieldSeparator** フィールド、または **TextQualifier** フィールドに対して "custom" と入力した場合、対応するカスタム要素 (例えば "CustomLineSeparator"、"CustomFieldSeparator"、"CustomTextQualifier" など) も使用する文字または文字列を表す 16 進数とともに含める必要があります。

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="FileSchema" nillable="true" type="FileSchema"/>
  <xs:complexType name="FileSchema">
    <xs:sequence>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        default="linesequential"
        name="Type"
        type="FileTypeEnum"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        default="UTF-8" name="Encoding" type="xs:string"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        name="RecordLength"
        type="xs:int"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        default="default"
        name="LineSeparator"
        type="LineSeparatorEnum"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"
        name="CustomLineSeparator"
        type="xs:string"/>
      <xs:element
        minOccurs="0"
        maxOccurs="1"

```

```
        default="comma"
        name="FieldSeparator"
        type="FieldSeparatorEnum"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="CustomFieldSeparator"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="none"
  name="TextQualifier"
  type="TextQualifierEnum"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="CustomTextQualifier"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="false"
  name="HasHeader"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="true"
  name="EnforceColumnCount"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Fields"
  type="ArrayOfFieldSchema"/>
</xs:sequence>
</xs:complexType>
<xs:simpleType name="FileTypeEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="linesequential"/>
    <xs:enumeration value="fixedwidth"/>
    <xs:enumeration value="delimited"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="LineSeparatorEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="default"/>
    <xs:enumeration value="windows"/>
    <xs:enumeration value="unix"/>
    <xs:enumeration value="mac"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
```



```
</xs:simpleType>
<xs:simpleType name="FieldSeparatorEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="comma"/>
    <xs:enumeration value="tab"/>
    <xs:enumeration value="space"/>
    <xs:enumeration value="semicolon"/>
    <xs:enumeration value="period"/>
    <xs:enumeration value="pipe"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="TextQualifierEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="none"/>
    <xs:enumeration value="single"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="custom"/>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="ArrayOfFieldSchema">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="unbounded"
      name="Field"
      nillable="true"
      type="FieldSchema"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="FieldSchema">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Name"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      default="string"
      name="Type"
      type="xs:string"/>
    <xs:element
      minOccurs="1"
      maxOccurs="1"
      name="Position"
      type="xs:int"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Length"
      type="xs:int"/>
  </xs:sequence>
</xs:complexType>
</xs:schema>
```

```
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="false"
  name="Trim"
  type="xs:boolean"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Locale"
  type="Locale"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  name="Pattern"
  type="xs:string"/>
<xs:element
  minOccurs="0"
  maxOccurs="1"
  default="none"
  name="Order"
  type="SortOrderEnum"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Locale">
  <xs:sequence>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Country"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Language"
      type="xs:string"/>
    <xs:element
      minOccurs="0"
      maxOccurs="1"
      name="Variant"
      type="xs:string"/>
  </xs:sequence>
</xs:complexType>
<xs:simpleType name="SortOrderEnum">
  <xs:restriction base="xs:string">
    <xs:enumeration value="none"/>
    <xs:enumeration value="ascending"/>
    <xs:enumeration value="descending"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

データフロー オプションの設定

以下では、Write to File ステージの実行時オプションをサポートするようにデータフローを設定する手順について説明します。

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. **[Write to File]** ステージを展開します。

以下のデータフロー オプションがエクスポートされています。

1. 文字エンコード
2. Field Separator
3. Text Qualifier
4. レコード長
5. 最初の行はヘッダ レコード
6. 選択した Write to File オプション名が、**[オプション名]** フィールドと **[オプション ラベル]** フィールドに表示されます。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。
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、**[選択ステージ]** オプションを選択します。
9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **[OK]** をクリックします。
12. 必要に応じて、オプションの追加を続けます。

13. 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。
14. データフローを保存してエクスポートします。

データフロー オプションのルール

1. 文字エンコーディング: 基盤の JVM で有効なすべての種類のエンコーディングが使用できます。このオプションを空にすることはできません。
2. フィールド区切り文字: 任意の 1 文字が区切り文字として使用できます。現時点では、16 進数や空白はサポートされていません。
3. *Text Qualifier*: 任意の 1 文字が修飾子として使用できます。16 進数はサポートされていません。
4. レコード長: 整数値のみが指定できます。空白や数値以外にすることはできません。
5. 開始レコード: 整数値のみが指定できます。数値以外にすることはできません。
6. 最大レコード数: 整数値のみが指定できます。数値以外にすることはできません。
7. 最初の行はヘッダレコード: true または false の boolean 値のみが指定できます。空白にすることはできません。

Write to Hadoop Sequence File

Write to Hadoop Sequence File ステージでは、データフローからの出力としてデータがシーケンシャル ファイルに書き出されます。シーケンシャル ファイルは、キー/値ペアで構成されるフラットファイルです。詳細については、wiki.apache.org/hadoop/SequenceFile を参照してください。

注: Write to Hadoop Sequence File ステージは、Hadoop 分散ファイル システム (HDFS) 上にある区切り記号付きの未圧縮シーケンシャル ファイルのみをサポートします。

関連するタスク: [Hadoop への接続 \(31ページ\)](#) : **Write to Hadoop Sequence File** ステージを使用できるようにするには、Hadoop ファイルサーバーへの接続を作成する必要があります。それが済むと、その接続を保存するための名前がサーバー名として表示されます。

[ファイル プロパティ] タブ

フィールド	説明
フィールド区切り文字	<p>区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは () 記号がフィールド区切り文字として使われています。</p> <pre>7200 13TH ST MIAMI FL 33144</pre> <p>フィールド区切り文字として定義できるのは次の文字です。</p> <ul style="list-style-type: none"> • スペース • タブ • カンマ • ピリオド (.) • セミコロン • パイプ () <p>これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。</p>
Text qualifier	<p>区切り記号付きファイル内のテキスト値を囲むのに使用する文字。</p> <p>例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。</p> <pre>"7200 13TH ST" "MIAMI" "FL" "33144"</pre> <p>テキスト修飾子として定義できるのは次の文字です。</p> <ul style="list-style-type: none"> • 一重引用符 (') • 二重引用符 (") <p>これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。</p>

[フィールド] タブ

[フィールド] タブでは、ファイルの各フィールドの名前、位置、およびタイプを定義します。詳細については、[出力シーケンシャル ファイルのフィールドの定義](#) (294ページ)

出力シーケンシャル ファイルのフィールドの定義

Write to Hadoop Sequence File ステージで、**【フィールド】** タブに、ファイル内のフィールドの名前、位置、タイプを定義します。**【ファイルプロパティ】** タブで入力ファイルを定義したら、フィールドを定義できます。

1. 入力データ、または既存のファイルから必要なフィールドを選択する場合は、**【クイック追加】** をクリックします。
 - a) 目的のフィールドを入力データから選択します。
 - b) **【OK】** をクリックします。
2. 新しいフィールドを追加する場合は、**【追加】** をクリックします。
 - a) フィールドの **【名前】** を入力します。
 - b) フィールドの **【タイプ】** を選択します。このステージでは、以下のデータ タイプがサポートされています。

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。

注: Parquet ファイルでは、datetime と time のデータタイプは String としてマッピングされます。RC ファイルでは、datetime データタイプは timestamp としてマッピングされます。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。指数表記すると、値の範囲は、 $-1.79769313486232E+308 \sim 1.79769313486232E+308$ 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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。

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。

注：RC、Avro、Parquet Hive ファイルに対し、bigdecimal データタイプは、有効桁数が 38 で小数点以下桁数が 10 の decimal データタイプに変換されます。

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)。

注：RC ファイルでは、long データタイプは bigint データタイプとしてマッピングされます。

string 文字シーケンス。

c) **[位置]** フィールドで、レコード内のこのフィールドの位置を入力します。

例えば、この入力ファイルで、AddressLine1 は位置 1、City は位置 2、StateProvince は位置 3、PostalCode は位置 4 です。

```
"AddressLine1"|"City"|"StateProvince"|"PostalCode"
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
"One Global View"|"Troy"|"NY"|"12180"
```

- 既存ファイルを上書きする場合は、**[再生成]** をクリックして既存ファイルからスキーマを取得してから、それを変更します。
このステージへの入力データに含まれる最初の 50 件のレコードに基づいて、スキーマが生成されます。
- フィールドの文字列の先頭と末尾から余分なスペース文字を削除するには、**[空白をトリム]** チェック ボックスを選択します。
- キーの生成について、オプションを 1 つ指定します。

自動生成 Hadoop クラスタによって、キーが自動生成されます。自動生成の場合、グリッド内のすべてのフィールドは値フィールドと見なされます。キーのデータタイプは long です。

ユーザ定義 デフォルトでは、グリッドの先頭のフィールドがキーフィールドとして選択されます。そのフィールドがキーフィールドであることを示すアイコンが表示されます。キーフィールドとして別のフィールドを選択することができます。

出力ファイルのフィールドを定義したら、フィールドのコンテンツとレイアウトを編集できます。

オプション名	説明
追加	フィールドを出力に追加します。現在のレイアウトの末尾にフィールドを追加できます。また、既存の位置にフィールドを挿入することもできます。その場合、残りのフィールドの位置は適宜調整されます。
変更	フィールドの名前とタイプを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されたフィールドの順序を変更します。

Write to Hive File

Write to Hive File ステージでは、データフロー入力を指定された出力 **Hive** ファイルに書き込みます。

サポートされている ORC、RC、Parquet、Avro のいずれかの **Hive** ファイル形式を出力ファイルとして選択できます。

関連するタスク:

[Hadoop への接続](#) (31ページ) : **Write to Hive File** ステージを使用できるようにするには、**Hadoop** ファイルサーバーへの接続を作成する必要があります。それが済むと、その接続を保存するための名前がサーバー名として表示されます。

[ファイル プロパティ] タブ

表 6 : 共通ファイル プロパティ

フィールド	説明
サーバ名	<p>[ファイル名] フィールドで選択したファイルが Hadoop システム上にあることを示します。Hadoop システム上のファイルを選択すると、[サーバー名] には、ファイル サーバーの作成時に Management Console で指定されたとおりのファイル サーバーの名前が反映されます。</p>
ファイル名	<p>省略記号ボタン (...) をクリックし、指定済みの Hadoop ファイル サーバーに作成する出力 Hive ファイルを参照します。このステージの出力データは、選択したファイルに書き込まれます。</p> <p>注：このステージでこのファイルを使用する前に、Management Console で Hadoop ファイル サーバーへの接続を作成しておく必要があります。</p>
ファイル タイプ:	<p>サポートされている以下の4つの Hive ファイル フォーマットのいずれかを選択します。</p> <ul style="list-style-type: none"> • ORC • RC • Parquet • Avro

表 7 : ORC ファイル プロパティ

フィールド	説明
バッファ サイズ	<p>ORC ファイルへの書き込み中に割り当てるバッファ サイズを定義します。キロバイト単位で指定します。</p> <p>注：デフォルトのバッファ サイズは 256 KB です。</p>
ストライプ サイズ	<p>ORC ファイルへの書き込み中に作成するストライプのサイズを定義します。メガバイト単位で指定します。</p> <p>注：デフォルトのストライプ サイズは 64 MB です。</p>

フィールド	説明
行インデックス幅	2つの連続した行インデックスの間に書き込む行の数を定義します。 注：デフォルトの行インデックス幅は 10000 行です。
圧縮タイプ	ORC ファイルへの書き込み中に使用する圧縮タイプを定義します。使用できる圧縮タイプは、 ZLIB と SNAPPY です。 注：デフォルトの圧縮タイプは ZLIB です。
パディング	ORC ファイルへの書き込み中に、HDFS ブロックの境界をまたぐストライプを最小限にするため、ストライプをパディングするかどうかを指定します。 注：デフォルトで、 [パディング] チェックボックスは選択されています。
プレビュー	データフローが少なくとも一度実行され、データが選択されたファイルに書き込まれた後で、書き込まれたファイルの最初の 50 件のレコードが取得され、 [プレビュー] グリッドに表示されます。

表 8 : RC ファイル プロパティ

フィールド	説明
バッファ サイズ	RC ファイルへの書き込み中に割り当てるバッファ サイズを定義します。キロバイト単位で指定します。 注：デフォルトのバッファ サイズは 256 KB です。
ブロック サイズ	RC ファイルへの書き込み中に作成するブロックのサイズを定義します。メガバイト単位で指定します。 注：デフォルトのブロック サイズは 64 MB です。
圧縮タイプ	RC ファイルへの書き込み中に使用する圧縮タイプを定義します。使用できる圧縮タイプは、 NONE と DEFLATE です。 注：デフォルトの圧縮タイプは NONE です。

フィールド	説明
プレビュー	<p>データフローが少なくとも一度実行され、データが選択されたファイルに書き込まれた後で、書き込まれたファイルの最初の 50 件のレコードが取得され、[プレビュー] グリッドに表示されます。</p> <p>[フィールド] タブでは、必須フィールドの順序とデータタイプを定義します。</p> <p>注： RC ファイル タイプの場合は、[プレビュー] をクリックして [プレビュー] グリッドを表示する前に、出力ファイルのメタデータを定義する必要があります。</p>

表 9 : Parquet ファイル プロパティ

フィールド	説明
圧縮タイプ	<p>PARQUET ファイルへの書き込み中に使用する圧縮タイプを定義します。使用できる圧縮タイプは、UNCOMPRESSED、GZIP、および SNAPPY です。</p> <p>注： デフォルトの圧縮タイプは UNCOMPRESSED です。</p>
ブロック サイズ	<p>PARQUET ファイルへの書き込み中に作成するブロックのサイズを定義します。メガバイト単位で指定します。</p> <p>注： デフォルトのブロック サイズは 128 MB です。</p>
ページ サイズ	<p>圧縮用のページサイズです。読み込み時には、各ページを個別に解凍できます。キロバイト単位で指定します。</p> <p>注： デフォルトのページサイズは 1024 KB です。</p>
辞書を有効にする	<p>辞書エンコーディングを有効/無効にします。</p> <p>重要： 辞書ページ サイズを有効にするには、辞書が有効である必要があります。</p> <p>注： デフォルトは true です。</p>

フィールド	説明
Dictionary Page size(辞書ページサイズ)	<p>辞書エンコーディングを使用する場合は、1つの行グループの1つの列につき1つの辞書ページがあります。辞書ページサイズは、ページサイズと同じように機能します。キロバイト単位で指定します。</p> <p>注：デフォルトの辞書ページサイズは 1024 KB です。</p>
Writer バージョン	<p>Parquet は、PARQUET_1_0 と PARQUET_2_0 の 2 つの Writer API バージョンをサポートします。</p> <p>注：デフォルトは PARQUET_1_0 です。</p>
プレビュー	<p>データフローが少なくとも一度実行され、データが選択されたファイルに書き込まれた後で、書き込まれたファイルの最初の 50 件のレコードが取得され、[プレビュー] グリッドに表示されます。</p>

表 10 : Avro ファイル プロパティ

フィールド	説明
同期間隔 (バイト単位)	<p>各ブロックに書き込まれる、およその非圧縮バイト数を指定します。有効な値は、32 ~ 2^30 です。ただし、同期間隔は 2K ~ 2M の間にすることが推奨されます。</p> <p>注：デフォルトの同期間隔は 16000 です。</p>
圧縮	<p>Avro ファイルへの書き込み中に使用する圧縮タイプを定義します。使用できる圧縮タイプは、NONE、SNAPPY、および DEFLATE です。[DEFLATE] 圧縮を選択する場合は、圧縮レベルを選択するための追加のオプションが表示されます (以下を参照)。</p> <p>注：デフォルトの圧縮タイプは NONE です。</p>
圧縮レベル	<p>このフィールドは、上記の [比較] フィールドで [DEFLATE] オプションを選択した場合に表示されます。</p> <p>0 ~ 9 の値が選択可能で、0 は圧縮なしを意味します。圧縮レベルは 1 ~ 9 の順に高くなり、それに伴ってデータ圧縮にかかる時間も増加します。</p> <p>注：デフォルトの圧縮レベルは 1 です。</p>

フィールド	説明
プレビュー	データフローが少なくとも一度実行され、データが選択されたファイルに書き込まれた後、書き込まれたファイルの最初の 50 件のレコードが取得され、このグリッドに表示されます。

[フィールド] タブ

[フィールド] タブでは、このステージのソース ファイルに存在し、出力ファイルへの書き込みに選択するフィールドの名前とタイプを定義します。

詳細については、「[Hive ファイル書き込みのためのフィールドの定義 \(301ページ\)](#)」を参照してください。

[実行時] タブ

[実行時] タブは、設定済みの Hadoop ファイル サーバーに存在する同じ名前のファイルの [上書き] オプションを提供します。[上書き] チェックボックスをチェックした場合、データフローの実行時に同じ Hadoop ファイル サーバーに存在する同じ名前のファイルは新しい出力 Hive ファイルで上書きされます。

デフォルトで、[上書き] チェックボックスはチェックされていません。

注：[上書き] を選択しない場合、データフロー実行中に書き込むファイルの名前が同じ Hadoop ファイル サーバー上の既存のファイルと重複すると例外が生成されます。

Hive ファイル書き込みのためのフィールドの定義

Write to Hive File ステージの [フィールド] タブには、ステージへの入力データに含まれるフィールドのスキーマ名とデータタイプが一覧表示されます。

- 入力データ、または既存のファイルから必要なフィールドを選択する場合は、**[クイック追加]** をクリックします。
 - 目的のフィールドを入力データから選択します。
 - [OK]** をクリックします。
- 新しいフィールドを追加する場合は、**[追加]** をクリックします。
 - フィールドの **[名前]** を入力します。
 - フィールドの **[タイプ]** を選択します。このステージでは、以下のデータ タイプがサポートされています。

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。
注: Parquet ファイルでは、datetime と time のデータタイプは String としてマッピングされます。RC ファイルでは、datetime データタイプは timestamp としてマッピングされます。
- 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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{-23}) \times 2^{127}$ 。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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- 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。
注: RC、Avro、Parquet Hive ファイルに対し、bigdecimal データタイプは、有効桁数が 38 で小数点以下桁数が 10 の decimal データタイプに変換されます。
- long** 正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
注: RC ファイルでは、long データタイプは bigint データタイプとしてマッピングされます。
- string** 文字シーケンス。

- c) **[位置]** フィールドで、レコード内のこのフィールドの位置を入力します。

例えば、この入力ファイルで、AddressLine1 は位置 1、City は位置 2、StateProvince は位置 3、PostalCode は位置 4 です。

```
"AddressLine1"|"City"|"StateProvince"|"PostalCode"
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
"One Global View"|"Troy"|"NY"|"12180"
```

- 既存ファイルを上書きする場合は、**[再生成]** をクリックして既存ファイルからスキーマを取得後、それを変更します。
これによって ORC と Parquet の出力ファイルの場合は、既存ファイルのメタデータに基づくスキーマが生成されます。RC 出力ファイルの場合は、既存フィールドを上書きするには、フィールドを明示的に追加する必要があります。

[名前]列には、入力データのさまざまな列の名前が表示されます。**[タイプ]**列には、入力データの各フィールドのデータタイプが表示されます。

注: *Parquet* ファイルタイプの場合は、もう1つの列である **[Null 可]** によって、フィールドに Null が設定可能かどうかを示されます。特定のフィールドのこのチェックボックスをオンにすることによって、そのフィールドに Null を設定可能にできます。それ以外の場合はこのチェックボックスをオフにします。

- 出力される列の名前、データタイプ、順序は、次のボタンを使用して変更できます。

オプション名	説明
追加	フィールドを出力に追加します。
変更	選択されているフィールドの名前とデータタイプを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されているフィールドの出力時の並び順を変更します。

- [OK]** をクリックします。

Write to NoSQL DB

Write to NoSQL DB では、データフローの出力を NoSQL データベースに書き込みます。このステージは、MongoDB と Couchbase のデータベース タイプをサポートします。

注： Write to NoSQLDB の複数の実行時インスタンスを使用することで、パフォーマンスを大きく改善できます。複数の実行時インスタンスを指定するには、**[実行時]** ボタンをクリックします。

[全般] タブ

フィールド	説明
接続	<p>ドロップダウンリストから必要なデータベース接続を選択します。表示されるオプションは、Management Console に定義されている接続によって異なります。</p> <p>新しい接続を追加するには、NoSQL への接続 (62ページ) を参照してください。</p> <p>既存の接続を変更するには、Management Console の [データ ソース] ページの接続リストから接続を選択して開き、必要な更新を行い、[保存] ボタンをクリックします。</p>
テーブル/ビュー	<p>書き込み先のコレクションの名前を指定します。</p> <p>[テーブル/ビュー] ドロップボックスにコレクション名を入力し、[テーブルの作成] をクリックすることによって、NoSQL データベースに新しいコレクションを作成できます。</p> <p>注： Couchbase の場合は、ビューではなくバケットに書き込むため、[テーブル/ビュー] ドロップダウンと [テーブルの作成] ボタンは無効になります。また、[プレビュー] ボタンも無効です。</p>
NULL 値を無視	<p>このオプションを有効にすると、NULL 値を持つフィールドはすべて無視されます。</p> <p>注： このオプションを有効にしない場合、NULL 値を持つフィールドもすべてデータベースに書き込まれます。</p>
プレビュー	<p>選択したテーブルのレコードを表示します。</p> <p>注： MongoDB データ ソースでは、[Where] フィールドに 1 つ以上の where 句が入力されている場合に [プレビュー] をクリックすると、フィルタリングされたレコードが表示されます。where 句が入力されていない場合は、すべてのレコードがプレビューされます。</p> <p>注： Couchbase データ ソースでは、[プレビュー] をクリックすると、キーを格納する追加の <code>_id</code> フィールドも表示されます。レコードに既に <code>_id</code> フィールドが存在する場合は、フィールドのプレビュー時に追加の <code>_id</code> フィールドによって上書きされます。</p>

フィールド	説明
すべて展開	プレビュー ツリーの項目を展開します。
すべて折りたたむ	プレビュー ツリーの項目を折りたたみます。

[フィールド] タブ

[フィールド] タブでは、データベースに書き込むデータを選択できます。詳細については、「[NoSQL データベースのフィールドの定義 \(305ページ\)](#)」を参照してください。

NoSQL データベースのフィールドの定義

Write to NoSQL DB ステージで、**[フィールド]** タブに、前のステージから取得したフィールドの名前とタイプを定義します。

1. **[フィールド]** タブで、**[フィールドの再生成]** をクリックします。

すると、前のステージから引き渡されるフィールドが表示されます。

データは、Fieldname (datatype) という形式で表示されます。

注: Couchbase の場合は、レコードに **_id** フィールドがあれば、そのフィールドがデータベースにレコードを書き込むためのキーとして使用されます。それ以外の場合は、レコード用のキーが自動生成され、データベースに書き込まれます。

2. フィールドの名前とタイプを変更するには、フィールドをハイライト表示し、**[変更]** をクリックします。
3. **[名前]** フィールドで、追加するフィールドを選択するか、フィールドの名前を入力します。
4. **[タイプ]** フィールドで、データに対して数学的な操作を行う予定がない場合は、データ タイプを文字列のままにしておくことができます。ただし、そのような操作を行う予定がある場合は、適切なデータタイプを選択します。ファイルに含まれる文字列データは、データフローでの適切なデータ操作を可能にするデータタイプに変換されます。

このステージでは、以下のデータタイプがサポートされています。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823E+38 \sim 3.402823E+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
- long** 正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
- string** 文字シーケンス。

5. **[フィールド]** タブでは、データベースに書き込む各フィールドを個別に選択するか、**[すべて選択]** をクリックしてすべてのフィールドを選択することができます。
6. オプションとして、**[実行時]** タブでバッチサイズが設定できます。バッチサイズは、一度にデータベースに書き込まれるレコード数を表します。
7. **[OK]** をクリックします。

NoSQL DB データフロー オプション

以下では、NoSQL DB の実行時オプションをサポートするようにデータフローを設定する手順について説明します。

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. NoSQLDB ステージを展開します。
6. データフロー オプションは、次の表に示すようにエクスポートされます。

データベース	読み込み	書き込み
Mongo DB	接続	接続
	テーブル	テーブル
Couchbase DB	接続	接続

データベース	読み込み	書き込み
	ビュー	
	設計ドキュメント名	

選択した NoSQL DB オプション名が、**[オプション名]** フィールドと **[オプション ラベル]** フィールドに表示されます。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。

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、**[選択ステージ]** オプションを選択します。
9. 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。
10. 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トする前にしか変更できません。サービスをエクスポートした後は、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](#)を参照してください。

11. **[OK]** をクリックします。
12. 必要に応じて、オプションの追加を続けます。
13. 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。
14. データフローを保存してエクスポートします。

Write to Spreadsheet

Write to Spreadsheet は、Excel スプレッドシートへのデータをデータフローからの出力のように書き出します。

[\[ファイル プロパティ\] タブ](#)

[ファイル プロパティ] タブには、データフローから書き出すスプレッドシートとデータを指定するためのオプションがあります。

フィールド名	説明
サーバ名	[ファイル名] フィールドで指定したファイルが Spectrum™ Technology Platform サーバ上にあることを示します。
ファイル名	ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、ファイルを参照して選択します。 重要： なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。
書き込みモード	データフローに出力する目的でスプレッドシート内のデータを書き出すよう指定します。次のいずれかのオプションを使用して実行時にスプレッドシートを作成できます。オプションは次のとおりです。 作成または上書き 新規のファイルを作成し、データフローが実行されるたびに出力ファイル内の既存のデータを置き換えます。 挿入 データフローの出力をマップ エリアに追加し、そこに既存のデータを後ろへ移します。 追加 データフローの出力をファイルの最後に追加します。ファイルの既存のデータは消去しません。
シート名	データフローに書き出すデータについてのスプレッドシートのシート名を指定します。
書き出し開始位置	書き出すデータの開始位置を行と列の組 (A1 や B2) または列で指定します。 [挿入] オプションでは、行と列の両方を指定する必要があります。 [追加] オプションでは行の値は無視されるので、列のみを指定できます。
先頭行を列ヘッダにする	ファイルの先頭行の内容がデータではなくヘッダ情報であると指定します。

[フィールド] タブ

[フィールド] タブでは、ファイル内の各フィールドの列、位置、タイプ、Nullable 値を定義します。詳細については、以下を参照してください。

[出力ファイルのフィールドを定義する \(309ページ\)](#)

出力ファイルのフィールドを定義する

Write to Spreadsheet の **[フィールド]** タブでファイルの各フィールドの名前、位置、およびデータタイプを定義します。**[ファイルプロパティ]** タブで出力ファイルを定義したら、フィールドを定義できます。**[Nullable]** オプションがオンで **[Name]** フィールドに NULL 値が含まれている場合、データフローはスプレッドシートに NULL 値を書き込みます。

出力ファイルにヘッダレコードが含まれている場合は、**[再生成]** をクリックするだけでフィールドを簡単に定義できます。

フィールドの位置、長さ、およびデータタイプにデフォルト値を設定してフィールドを定義するには、**[クイック追加]** をクリックし、追加するフィールドを選択します。

入力ファイルにヘッダレコードが含まれていない場合、またはフィールドを手動で定義したい場合は、以下のステップを続けます。

1. **[追加]** をクリックします。
2. **[名前]** フィールドで、追加したいフィールドを選択します。
3. **[タイプ]** フィールドで、データフローから入力されるフィールドのデータタイプを選択します。

Spectrum™ Technology Platform では、以下のデータタイプがサポートされています。

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。

bytearray バイトの配列 (リスト)。

注: bytearray は REST サービスの入力としてはサポートされていません。

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、 $2^{-1074} \sim (2 \cdot 2^{-52}) \times 2^{1023}$ 。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、 $2^{-149} \sim (2 \cdot 2^{23}) \times 2^{127}$ 。指数表記すると、値の範囲は、 $-3.402823\text{E}+38 \sim 3.402823\text{E}+38$ 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、 $-2^{31} (-2,147,483,648) \sim 2^{31}-1 (2,147,483,647)$ 。
long	正と負の整数を含む数値データ タイプ。値の範囲は、 $-2^{63} (-9,223,372,036,854,775,808) \sim 2^{63}-1 (9,223,372,036,854,775,807)$ 。
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。

4. [追加] をクリックします。

出力ファイルのフィールドを定義したら、フィールドのコンテンツとレイアウトを編集できます。

オプション名	説明
追加	フィールドを出力に追加します。現在のレイアウトの末尾にフィールドを追加できます。また、既存の位置にフィールドを挿入することもできます。その場合、残りのフィールドは Write to Spreadsheet によって適宜調整されます。
変更	フィールドの名前とタイプを変更します。
削除	選択されたフィールドを出力から削除します。
上へ移動/下へ移動	選択されたフィールドの順序を変更します。

Write to Variable Format File

Write to Variable Format File は、さまざまなレイアウトのレコードをファイルに書き出します。可変フォーマット ファイルには、次の特性があります。

- ファイル内のレコードは、異なるフィールド、および異なる数のフィールドを持つことができる。
- 各レコードには、レコードのタイプを識別するタグ (通常は数字) を含める必要がある。
- 階層的な関連性をサポートしている。

可変フォーマット ファイルの例

次の例は、2人の顧客 Joe Smith と Anne Johnson の当座預金口座の取引に関する情報を含む、可変フォーマット ファイルを示しています。この例のファイルは、フィールド区切り文字としてカンマを使用する区切り記号付きファイルです。

```
001   Joe,Smith,M,100 Main St,555-234-1290
100   CHK12904567,12/2/2007,6/1/2012,CHK
200   1000567,1/5/2012,Fashion Shoes,323.12
001   Anne,Johnson,F,1202 Lake St,555-222-4932
100   CHK238193875,1/21/2001,4/12/2012,CHK
200   1000232,3/5/2012,Blue Goose Grocery,132.11
200   1000232,3/8/2012,Trailway Bikes,540.00
```

各レコードの先頭フィールドには、レコードのタイプ、およびレコードのフォーマットを識別するタグが含まれています。

- 001: Customer レコード
- 100: Account レコード
- 200: Account transaction レコード

区切り記号付きファイルにおいて、上の例に示すように、タグ値 (001、100、200) をレコード先頭の固定長バイトに含めることは一般的です。

各レコードには独自のフォーマットがあります。

- 001: FirstName,LastName,Gender,Address,PhoneNumber
- 100: AccountID,DateOpened,ExpirationDate,TypeOfAccount
- 200: TransactionID,DateOfTransaction,Vendor,Amount

レコードフォーマット 100 (アカウント レコード) は、前のレコード 001 の子で、レコードフォーマット 200 (アカウント トランザクション レコード) は、前のレコード 100 (アカウント レコード) の子です。したがって、例のファイルでは、Joe Smith のアカウント CHK12904567 に Fashion Shoes で 2012/1/5 に数量 323.12 のトランザクションが発生していました。同様に、Anne Johnson のアカウント CHK238193875 には、Blue Goose Grocery で 2012/3/5 に 1 つ、Trailway Bikes で 2012/3/8 に 1 つのトランザクションが発生していました。

[ファイル プロパティ] タブ

オプション名	説明
サーバ名	<p>入力として選択したファイルが Enterprise Designer を実行しているコンピュータ上にあるか、Spectrum™ Technology Platform サーバー上にあるかを示します。ローカルコンピュータ上のファイルを選択した場合、サーバー名はマイコンピュータになります。サーバー上のファイルを選択した場合、サーバー名は Spectrum™ Technology Platform になります。</p>
ファイル名	<p>ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。</p> <p>注： なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。</p>
ルート タグ名	<p>他のレコード タイプの親であるレコードに対して使用するタグ。例えば、3つのレコード タイプ 001、100、200 があり、レコード タイプ 100 および 200 がレコード タイプ 001 の子である場合、001 はルート タグです。</p>
固定長タグを使用	<p>レコード タグを配置する各レコードの先頭に固定長スペースを割り当てるかどうかを指定します。この例では、固定長フィールドに、タグ 001、100、200 があるファイルを示しています。</p> <pre>001 Joe, Smith, M, 100 Main St, 555-234-1290 100 CHK12904567, 12/2/2007, 6/1/2012, CHK 200 1000567, 1/5/2012, Mike's Shoes, 323.12</pre>
タグ長	<p>[固定長タグを使用] ボックスをオンにした場合、各レコードの先頭でタグに対して割り当てるスペースの数を指定します。例えば、7 を指定した場合は、各レコードの最初の 7 つのスペースがタグに対して予約されます。指定する値は、最も長いタグ名の文字の数以上でなければなりません。タグ名については「可変フォーマット ファイルのタグ名 (320 ページ)」を参照してください。</p> <p>指定された値より長い名前を持つフィールドを [フィールド] タブに追加した場合、[タグ長] フィールドの値は自動的に増えます。</p> <p>最大タグ長は 1024 です。</p>

オプション名

説明

数値タグの接頭辞の削除

タグをファイルに書き出す前に、フィールド名の "NumericTag_" の部分を削除します。接頭辞 "NumericTag_" は、数字で始まるすべてのタグ名に対して、**Read from Variable Format File** ステージによってタグ名に追加されます。これは、タグ名がレコードのデータを含むリストデータフローフィールドの名前として使用され、データフローフィールド名の先頭を数字にすることができないためです。例えば、タグ 100 はリストフィールド名 "NumericTag_100" に変更されます。このオプションを有効にすると、このフィールドは "NumericTag_100" ではなく "100" というタグを持つレコードとして出力ファイルに書き出されます。

オプション名	説明
文字エンコーディング	テキスト ファイルのエンコーディング。次のいずれかを選択します。
CP1252	このエンコーディングは Windows-1252 文字セット、または単純に Windows 文字セットとも呼ばれています。これは ISO-8859-1 の上位クラスであり、128 ~ 159 のコード範囲を使用して、ISO-8859-1 文字セットに含まれていない追加の文字を表示します。
UTF-8	すべての Unicode 文字をサポートし、かつ ASCII との下位互換性があります。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。
UTF-16	すべての Unicode 文字をサポートします。しかし、ASCII との下位互換性はありません。UTF の詳細については、 unicode.org/faq/utf_bom.html を参照してください。
US-ASCII	英語のアルファベット順に従う文字エンコーディング。
UTF-16BE	ビッグエンディアン UTF-16 エンコーディング (下位アドレスが上位バイトとなるようにシリアル化)。
UTF-16LE	リトルエンディアン UTF-16 エンコーディング (下位アドレスが下位バイトとなるようにシリアル化)。
ISO-8859-1	主として西ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-1 とも呼ばれます。
ISO-8859-3	主として南ヨーロッパの言語で使われる ASCII 文字エンコーディング。Latin-3 とも呼ばれます。
ISO-8859-9	主としてトルコ語で使われる ASCII 文字エンコーディング。Latin-5 とも呼ばれます。
CP850	西ヨーロッパの言語を書くための ASCII コード ページ。
CP500	西ヨーロッパの言語を書くための EBCDIC コード ページ。
Shift_JIS	日本語のための文字エンコーディング。
MS932	NEC 特殊文字、NEC 選定 IBM 拡張文字、IBM 拡張文字を含めた Microsoft の拡張版 Shift_JIS 文字コード。
CP1047	Latin-1 文字セット全体を含む EBCDIC コード ページ。

オプション名	説明
フィールド区切り文字	<p>区切り記号付きファイル内のフィールドを区切るのに使用する文字を指定します。例えば、次のレコードでは () 記号がフィールド区切り文字として使われています。</p> <pre data-bbox="560 451 1421 514">7200 13TH ST MIAMI FL 33144</pre> <p>フィールド区切り文字として定義できるのは次の文字です。</p> <ul data-bbox="560 577 706 808" style="list-style-type: none">• スペース• タブ• カンマ• ピリオド (.)• セミコロン• パイプ () <p>これ以外の文字がフィールド区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字を区切り文字として選択してください。</p>
タグ区切り文字	<p>区切り記号付きファイル内の各レコードの識別フィールドを区切るために、タグフィールドの後ろに配置する文字を指定します。タグ区切り文字は 1 文字でなければなりません。</p> <p>デフォルトでは、以下の文字がタグ区切り文字として選択できます。</p> <ul data-bbox="560 1155 706 1386" style="list-style-type: none">• スペース• タブ• カンマ• ピリオド (.)• セミコロン• パイプ () <p>これ以外の文字がタグ区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックして、カスタムなタグ区切り文字を追加して選択します。</p> <p>注：デフォルトで、[レコード区切り文字] は [フィールド区切り文字] として選択された文字と同じです。このフィールドを有効にして別の文字を選択するには、[フィールド区切り文字と同じ] チェックボックスをオフにします。</p>
[フィールド区切り文字と同じ]	<p>タグ区切り文字がフィールド区切り文字と同じかどうかを示します。別の文字をタグ区切り文字として選択するには、このチェックボックスをオフにします。</p> <p>注：デフォルトで、このチェックボックスはオフで、[タグ区切り文字] フィールドは無効になっています。</p>

オプション名

説明

Text qualifier

区切り記号付きファイル内のテキスト値を囲むのに使用する文字。

例えば、次のレコードでは二重引用符 (") がテキスト修飾子として使われています。

```
"7200 13TH ST"|"MIAMI"|"FL"|"33144"
```

テキスト修飾子として定義できるのは次の文字です。

- 一重引用符 (')
- 二重引用符 (")

これ以外の文字がテキスト修飾子として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をテキスト修飾子として選択してください。

レコード区切り文字

順次ファイルまたは区切り記号付きファイル内のレコードを区切るのに使用する文字を指定します。**[デフォルトの EOL を使用]** チェック ボックスをオンにすると、このフィールドは使用できません。

使用できるレコード区切り文字の設定は次のとおりです。

Unix (U+000A)	改行 (LF) 文字でレコードを区切ります。これは Unix システムの標準のレコード区切り文字です。
Macintosh (U+000D)	復帰 (CR) 文字でレコードを区切ります。これは Macintosh システムの標準のレコード区切り文字です。
Windows (U+000D U+000A)	復帰改行 (CR+LF) でレコードを区切ります。これは Windows システムの標準のレコード区切り文字です。

これ以外の文字がレコード区切り文字として使われているファイルについては、省略記号ボタン (...) をクリックし、別の文字をレコード区切り文字として選択してください。

デフォルトの EOL を使用

Spectrum™ Technology Platform サーバーが実行されているオペレーティングシステムのデフォルトの行末 (EOL) 文字をファイルのレコード区切り文字として使用します。

ファイルの EOL 文字がサーバーのオペレーティングシステムで使われているデフォルトの EOL 文字と異なる場合は、このオプションをオンにしないでください。例えば、ファイルで Windows の EOL が使われていて、サーバーの動作プラットフォームが Linux の場合は、このオプションをオンにしないでください。代わりに、**[レコード区切り文字]** フィールドで **[Windows]** オプションを選択します。

[フィールド] タブ

[フィールド] タブは、出力ファイルに含めるデータフローのフィールドを制御します。

オプション名	説明
追加	<p>クリックして、出力にフィールドを追加します。</p> <p>Write to Variable Format File で使用するデータフロー フィールドの作成については、可変フォーマットファイルへのフラットデータの書き出し (318ページ) を参照してください。</p>
変更	<p>クリックして、タグの名前を変更します。このボタンが有効になるのは、タグが選択されている場合のみです。[固定長タグを使用] オプションを [ファイル プロパティ] タブで有効にすると、それより長いタグ名を入力した場合にタグ長が自動的に調整されます。</p> <p>注：このボタンの使用によるルート タグ名の変更は、[ファイル プロパティ] タブの [ルート タグ名] フィールドの値の変更と同じ効果を持ちます。</p>
削除	<p>選択されたフィールドを出力から削除します。リストフィールドを削除する場合、子フィールドもすべて削除されます。子フィールドを削除する場合、選択された子フィールドのみがリスト フィールドから削除されます。</p>
XX すべて削除	<p>すべてのフィールドを出力から削除します。</p>
上へ移動/下へ移動	<p>選択されたフィールドの順序を変更します。</p>

[実行時] タブ

オプション名	説明
ファイル名	<p>ここには、[ファイル プロパティ] タブで定義したファイルが表示されます。</p>

オプション名	説明
複数ファイルの生成	<p>1つのファイルにすべてのレコードを書き込むのではなく、レコードをいくつかのファイルに書き込む場合は、このオプションを選択します。各レコードを書き込むファイルは、レコード自体に指定されます。各レコードには、レコードを書き込むファイルのファイル名または完全ファイルパスのいずれかを指定するフィールドが含まれている必要があります。例えば、(各種グループの)各社の株価をすべてのクライアントに個別に送信する場合、この機能を使用すると、各社の株価を個別のファイルに書き込んで、それらのファイルを各クライアントに送信できます。【複数ファイルの生成】オプションを有効にする場合は、Spectrum™ Technology Platform サーバー上またはFTPサーバー上のいずれかにある出力ファイルを指定する必要があります。FTPサーバー上のファイルにデータを書き込む場合は、Management Console を使用してファイルサーバーへの接続を定義する必要があります。</p> <p>注: 【ファイルパスフィールド】で選択する列のレコードは、ソート順になっている必要があります。レコードにファイル名または完全ファイルパスのいずれかが含まれている場合は、この機能を使用します。</p>
ファイルパスフィールド	<p>レコードを書き込むファイルのパス(ファイル名または完全ファイルパスのいずれか)を含むフィールドを選択します。ルートタグに直接マップされる単純タイプ要素のみが【ファイルパスフィールド】にリストされることに注意してください。このフィールドは、【複数ファイルの生成】を選択している場合にのみ有効になります。</p>
書き込みモード	<p>データフローの出力をファイルの最後に追加するか、ファイル内の既存のデータを削除した後に出力を書き込むかを指定します。</p> <p>Overwrite データフローが実行されるごとに、出力ファイル内の既存のデータを置き換えます。</p> <p>追加 データフローの出力をファイルの最後に追加します。ファイルの既存のデータは消去しません。</p>

可変フォーマット ファイルへのフラット データの書き出し

Spectrum™ Technology Platform データフローでは、各レコードは同じフィールドを持ちます。ただし、可変フォーマット ファイルでは、すべてのレコードが同じフィールドを含むわけではありません。データフローのフラットデータを可変フォーマット ファイルに書き出すには、データフロー内の各レコードを分割し、各レコードのフィールドを可変フォーマット ファイルで使用するレコードタイプに対応するリスト フィールドにグループ化します。リスト フィールドは、フィールドのコレクションです。例えば、フィールド `FirstName`、`LastName`、`Gender`、`Address`、および `Phone` は、`AccountOwner` というリスト フィールドにグループ化できます。

フラットデータを可変フォーマットファイルに書き出すには、Aggregator ステージを使用して、可変フォーマットファイルに書き出すレコードタイプに対応するリストフィールドにフィールドをグループ化します。これを行うには、次の手順を実行します。

1. データフロー内の Write to Variable Format File ステージより上流の任意の場所に Aggregator ステージを配置します。
2. Aggregator ステージをダブルクリックして、オプション ウィンドウを開きます。
3. **[グループ化方法]** を選択し、**[追加]** をクリックします。
4. **[グループ化方法]** フィールドで、関連データの識別に使用できる一意の識別子を含むフィールドを選択します。このフィールドの値は、フラットデータ内のレコード間で一意でなければなりません。例えば、アカウント番号、社会保障番号、または電話番号などです。

注：選択したフィールドはソートされている必要があります。ソートされていない場合は、Sorter ステージを使用して、レコードをそのフィールドでソートします。

5. **[OK]** をクリックします。
6. **[出力リスト]** を選択し、**[追加]** をクリックします。
各出力リストは、可変フォーマットファイル内の1つのレコードタイプを表します。
7. **[新しいデータタイプ]** を選択し、**[タイプ名]** フィールドに、このデータタイプに含める情報のタイプを指定します。これが可変フォーマットファイル内のレコードタイプになります。例えば、このデータタイプがアカウントトランザクションに関連するレコードを含む場合、タイプに "AccountTransaction" という名前を付けることができます。
8. **[名前]** フィールドに、このフィールドに付与する名前を入力します。これは、**[タイプ名]** フィールドで指定した名前と同じ名前にすることができます。
9. **[OK]** をクリックします。
10. 作成したデータタイプを選択し、**[追加]** をクリックします。
11. **[既存フィールド]** オプションをオンのままにして、このデータタイプに含めるいずれかのフィールドを選択し、**[OK]** をクリックします。これが可変フォーマットファイル内のレコードタイプになることに注意してください。この操作を繰り返して、その他のフィールドをこのレコードタイプに追加します。
12. 可変フォーマットファイルに含めるレコードタイプごとに、追加の出力リストを作成します。完了したら、**[OK]** をクリックして Aggregator オプションを閉じます。

Aggregator ステージから出力されるフィールドが、可変フォーマットファイル出力に含めるレコードタイプに対応するリストフィールドにグループ化されるようになります。

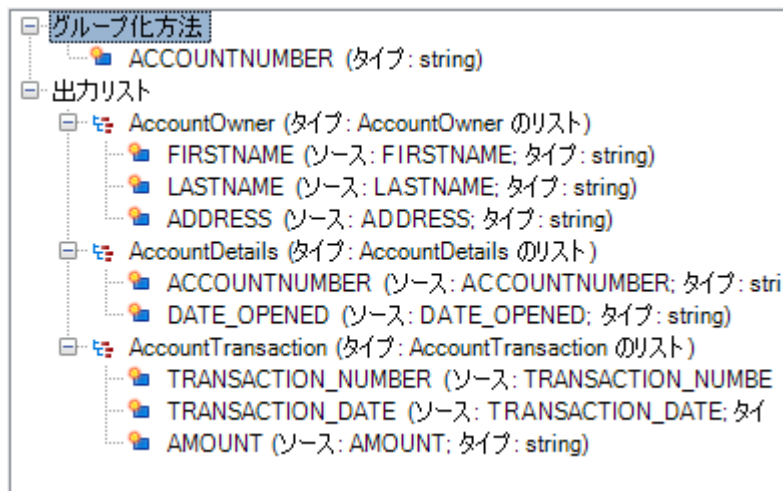
例えば、次のフラットデータがあるとします。

```
FIRSTNAME, LASTNAME, ADDRESS, ACCOUNTNUMBER, DATE_OPENED, TRANSACTION_NUMBER, TRANSACTION_DATE, AMOUNT  
Joe, Smith, 100 Main St, CHK12904567, 12/2/2007, 1000567, 1/5/2012, 323.12
```

このデータを、可変フォーマット ファイル内で次のように変換するものとします。

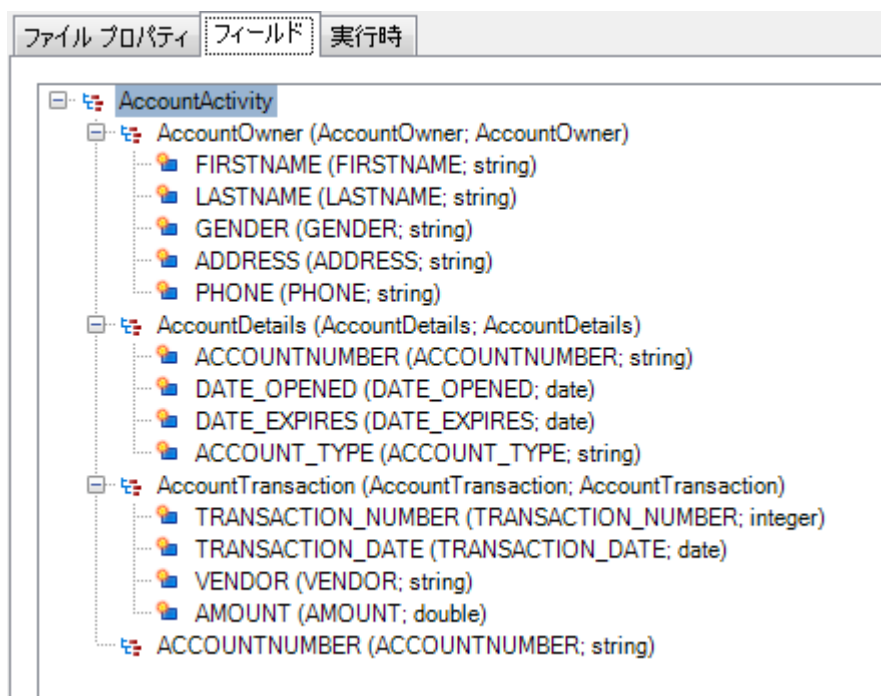
```
AccountOwner      Joe,Smith,100 Main St
AccountInformation CHK12904567,12/2/2007
Transaction       1000567,1/5/2012,323.12
```

この処理を行うには、次のように設定された Aggregator ステージを作成します。



可変フォーマット ファイルのタグ名

可変フォーマットファイルでは、出力ファイル内の各レコードがレコードタイプを示すタグを持ちます。Write To Variable Format File では、フィールド名が出力ファイルのタグ名として使用されます。例えば、次のフィールドを考えてみましょう。



これらのフィールドは、次のようにファイルに書き出されます。この例では、アカウントに 2 つの AccountTransaction レコードがあることに注意してください。

```
AccountOwner      Anne,Johnson,F,1202 Lake St,555-222-4932
AccountDetails    CHK238193875,1/21/2001,4/12/2012,CHK
AccountTransaction 1000232,3/5/2012,Blue Goose Grocery,132.11
AccountTransaction 1000232,3/8/2012,Trailway Bikes,540.00
```

注：文字列などの単純フィールドを含むリスト フィールドのみが、出力ファイルに書き出されます。リストフィールドが他のリストフィールドでのみ構成される場合は、出力ファイルに書き出されません。上の例では、AccountActivity タグを持つレコードは出力ファイルに書き出されません。なぜなら、AccountActivity は他のリスト フィールド (AccountOwner、AccountDetails、および AccountTransaction) でのみ構成されるからです。

Write to XML

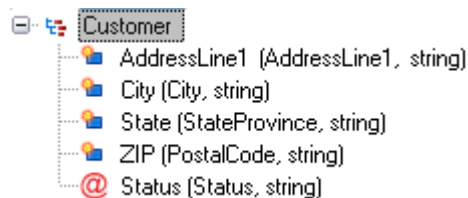
Write to XML ステージは、ジョブまたはサブフローの出力を XML ファイルに書き出します。

[ファイル プロパティ] タブ

フィールド名	説明
データ ファイル	出力 XML ファイルへのパスを指定します。省略記号ボタン (...) をクリックし、目的のファイルを見つけます。 注：なお、Spectrum™ Technology Platform サーバーを実行しているプラットフォームが Unix または Linux の場合、これらのプラットフォームでファイル名およびパスの大文字と小文字が区別されることに注意してください。
実ファイル	[フィールド] タブで指定された構造を表示します。要素をクリックし、[データ ファイル] フィールドで指定されたファイルがその要素を含む場合、データのプレビューが表示されます。プレビューには単純要素のデータのみ表示できることに注意してください。
スキーマのエクスポート	実ファイル ビューに表示されるスキーマを表す XSD ファイルを保存するには、このボタンをクリックします。スキーマ ファイルは指定した場所に即座に保存されます。

[フィールド] タブ

[フィールド] タブでは、出力 XML ファイルに含めるフィールドを定義します。フィールドを追加すると、そのフィールドがツリー構造に表示されます。ツリーには、XML ファイルに書き出される要素または属性の名前が表示されます。次の例で示すように、要素/属性名の後ろの括弧内は、データフロー フィールド名とそのデータ タイプです。



これは、4つの要素と1つの属性がXMLファイルに書き出されることを示します。属性は、赤い"@ "記号で示されます。

要素 **State** は **StateProvince** フィールドのデータを含み、文字列データであることに注意してください。同様に、要素 **ZIP** は **PostalCode** フィールドのデータを含み、文字列データです。XML ファイルは次のようになります。

```
<XmlRoot>
  <Customer Status="0">
    <AddressLine1>7713 Mullen Dr</AddressLine1>
    <City>Austin</City>
    <State>TX</State>
    <ZIP>78757-1346</ZIP>
  </Customer>
  <Customer Status="0">
    <AddressLine1>1825B Kramer Ln</AddressLine1>
    <City>Austin</City>
    <State>TX</State>
    <ZIP>78758-4260</ZIP>
  </Customer>
</XmlRoot>
```

注：ルート要素名 (この例では <XmlRoot>) が **【ファイル プロパティ】** タブで指定されています。

次の表は、**【フィールド】** タブのオプションについて説明したものです。

オプション名	説明
追加	フィールドを出力に追加します。

オプション名

説明

変更

オプション名

説明

フィールドを XML に書き出す方法を変更します。次のオプションを指定できます。

名前 このオプションは、単純フィールドを変更する場合に使用できます。データフロー フィールドを XML 要素または属性に書き出すかどうかを指定します。

要素 このオプションを選択すると、フィールドのデータが XML 要素に書き出されます。**[要素名]** フィールドで使用する要素名を指定します。

属性 フィールドのデータが親要素の属性に書き出されます。**[属性名]** フィールドで使用する属性名を指定します。

要素名/属性名 XML ファイルに書き出される要素または属性の名前を指定します。デフォルト名はデータフロー フィールド名です。

すべての子を次のタイプに変更する このオプションは、複合要素を変更する場合に使用できます。複合要素を含める XML のタイプを指定します。次のいずれかです。

変更なし 子タイプは、現在定義されている要素または属性のままです。各フィールドのタイプを個別に指定するには、フィールドを選択し、**[変更]** をクリックします。

要素 要素の単純フィールドがすべて XML 要素として書き込まれます。

属性 要素の単純フィールドがすべて XML 属性として書き込まれます。

ネームスペース 要素または属性に使用する XML ネームスペースを指定する場合は、ここで選択します。Write to XML ステージの **[フィールド]** タブで、ネームスペースを作成できます。

空のフィールドを含む null 値を持つ、またはデータを持たない XML 要素を出力ファイルに含めるには、このボックスをオンにします。このボックスをオンにしない場合、空の要素は出力に含まれません。

例えば、<City> という要素が定義されているが、City フィールドにデータを持たないレコードがある場合、**[空のフィールドを含む]** をオンにしていれば、XML 出力に以下が含まれます。

```
<City xs:nil="true"></City>
```

このボックスをオンにしない場合、<City> 要素は出力ファイルに書き出されません。

注：**[データフロー フィールド]** には、データが要素または属性に書き出されるフィールドが表示されます。これは、要素名または属性名を異な

オプション名	説明
	<p>る名前に変更した場合に、その要素または属性に含まれるフィールドのデータを引き続き確認できるようにするためです。</p>
削除	<p>選択されたフィールドを出力から削除します。リストフィールドを削除する場合、子フィールドもすべて削除されます。子フィールドを削除する場合、選択された子フィールドのみがリスト フィールドから削除されます。</p>
すべて削除	<p>すべてのフィールドを出力から削除します。</p>
上へ移動/下へ移動	<p>選択されたフィールドの順序を変更します。</p> <p>単純要素を複合要素内に移動することはできないことに注意してください。複合要素内の要素を変更する場合は、データフローの Aggregator ステージを変更して、複合要素に含めるデータフロー フィールドを含める必要があります。詳細については、「フラットデータからの複合XMLの作成 (328ページ)」を参照してください。</p>
再生成	<p>現在定義されているフィールドを、上流のチャンネルから Write to XML に出力されるフィールドで置き換えます。</p>

[実行時] タブ

オプション名	説明
複数ファイルの生成	<p>1つのファイルにすべてのレコードを書き込むのではなく、レコードをいくつかのファイルに書き込む場合は、このオプションを選択します。各レコードを書き込むファイルは、レコード自体に指定されます。各レコードには、レコードを書き込むファイルのファイル名または完全ファイルパスのいずれかを指定するフィールドが含まれている必要があります。例えば、(各種グループの) 各社の株価をすべてのクライアントに個別に送信する場合、この機能を使用すると、各社の株価を個別のファイルに書き込んで、それらのファイルを各クライアントに送信できます。【複数ファイルの生成】オプションを有効にする場合は、Spectrum™ Technology Platform サーバー上またはFTPサーバー上のいずれかにある出力ファイルを指定する必要があります。FTPサーバー上のファイルにデータを書き込む場合は、Management Consoleを使用してファイルサーバーへの接続を定義する必要があります。</p> <p>注：【ファイルパスフィールド】で選択する列のレコードは、ソート順になっている必要があります。レコードにファイル名または完全ファイルパスのいずれかが含まれている場合は、この機能を使用します。</p>
ファイルパスフィールド	<p>レコードを書き込むファイルのパス (ファイル名または完全ファイルパスのいずれか) を含むフィールドを選択します。ルートに直接マップされる単純タイプ要素のみが【ファイルパスフィールド】にリストされることに注意してください。このフィールドは、【複数ファイルの生成】を選択している場合にのみ有効になります。</p>
実行時にスキーマを生成	<p>実行時に XSD を生成し、XML ファイルにスキーマへの noNamespaceSchemaLocation 参照を挿入するには、このオプションを選択します。属性 noNamespaceSchemaLocation の値は、XML ファイル内の XSD ファイル名です。データフローの編集時にスキーマをエクスポートする場合は、出力 XML ファイル内の XSD への参照がなくなるため、XSD への参照を手動で追加する必要があります。</p>
スキーマパス	<p>出力 XML ファイルのスキーマを含む XSD ファイルを保存するパスを指定します。省略記号ボタン (...) をクリックし、必要なファイルを参照して選択します。データフローを実行すると、指定した場所にスキーマファイルが保存されます。</p>

XML 出力ファイルでのネームスペースの使用

名前空間を使用すると、各要素または属性を XML 名前空間に割り当てることによって、出力内に重複する要素名および属性名を含めることができます。

1. Enterprise Designer で、データフローを開きます。
2. キャンバスで Write to XML ステージをダブルクリックします。

3. **[フィールド]** タブをクリックします。
4. 1つ以上のネームスペースを定義します。
 - a) **[接頭辞]** 列に、要素または属性をネームスペースに関連付けるために使用する接頭辞を入力します。
 - b) **[名前空間]** 列で、ネームスペースの URL を指定します。
 - c) これを繰り返して、出力XMLファイルに使用するすべてのネームスペースを定義します。
5. 1つ以上の要素または属性をネームスペースに関連付けます。
 - a) **[フィールド]** タブで、ネームスペースに関連付ける要素または属性を選択して **[変更]** をクリックするか、**[追加]** をクリックして新しい要素または属性を作成します。
 - b) **[名前空間]** フィールドで、要素または属性に関連付けるネームスペースの接頭辞を選択します。
 - c) **[OK]** をクリックします。

フラット データからの複合 XML の作成

多くの場合、データフローは、単純 XML 要素として XML に書き出されるフラット フィールドを含むレコードを生成します。フラット フィールドを複合 XML 要素に構成して階層データを生成する場合は、1つ以上の **Aggregator** ステージを使用してこれを行うことができます。

例えば、先頭行がヘッダ レコードである次のフラット データがあるとします。

```
addressline1,age,city,country,gender,name,number,postalcode,stateprovince,type
1253 Summer St.,43,Boston,United States,M,Sam,019922,02110,MA,Savings
```

以下に示すように、住所に関連するデータ フィールドと、アカウントに関連するフィールドを、`<Address>` および `<Account>` という複合 XML 要素にグループ化できます。

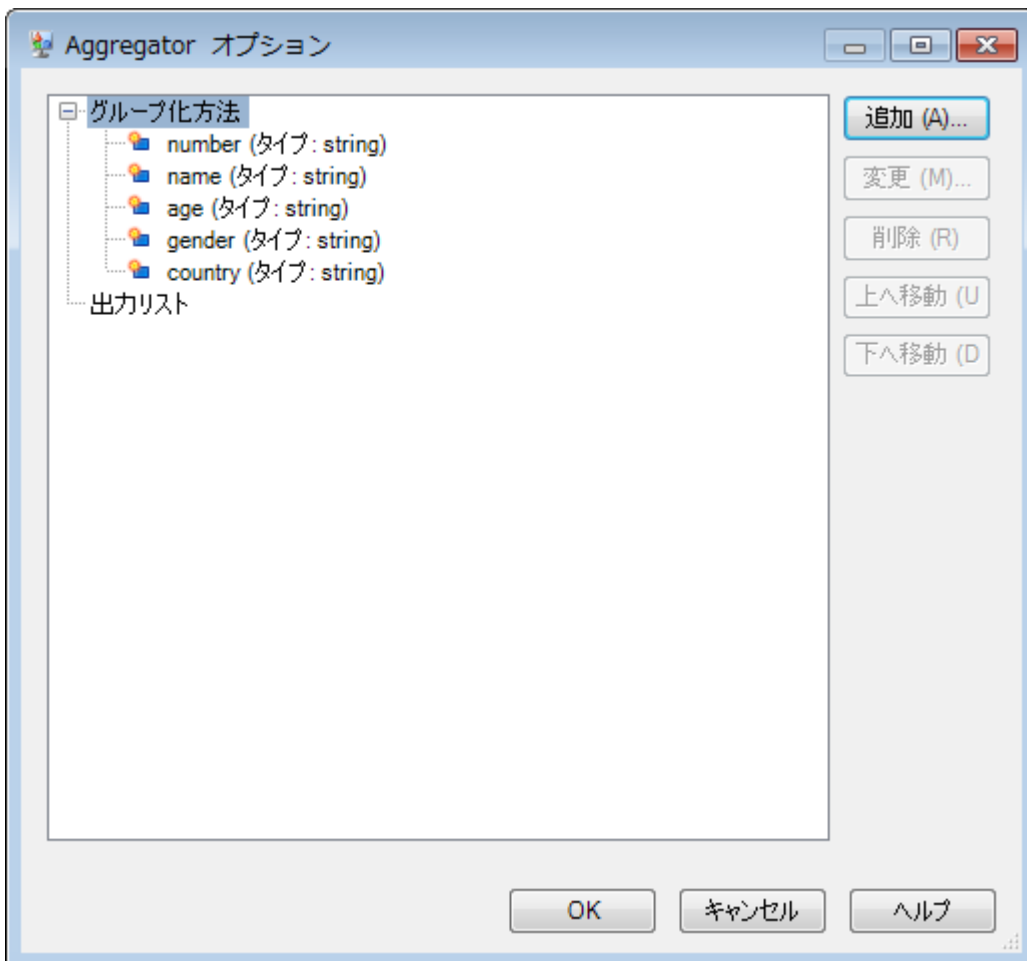
```
<CustomerRecord>
  <name>Sam</name>
  <age>43</age>
  <gender>M</gender>
  <country>United States</country>
  <Address>
    <addressline1>1253 Summer St.</addressline1>
    <city>Boston</city>
    <stateprovince>MA</stateprovince>
    <postalcode>02110</postalcode>
  </Address>
  <Account>
    <number>019922</number>
    <type>Savings</type>
```



```
</Account>
</CustomerRecord>
```

1. 複合要素を作成するデータフロー内の位置に **Aggregator** ステージを追加します。
2. **Aggregator** ステージをダブルクリックして、ステージのオプションを開きます。
3. **[グループ化方法]** を選択し、**[追加]** をクリックします。
4. アカウント番号など、各レコードに対して一意の値を含むフィールドを選択して、**[OK]** をクリックします。
5. 他の単純フィールドも通過させる場合は、**[グループ化方法]** を選択し、再度 **[追加]** をクリックして、含める単純フィールドをすべて追加します。

例えば、この場合は、各レコードに 5 つの単純フィールド **number**、**name**、**age**、**gender**、および **country** があります。

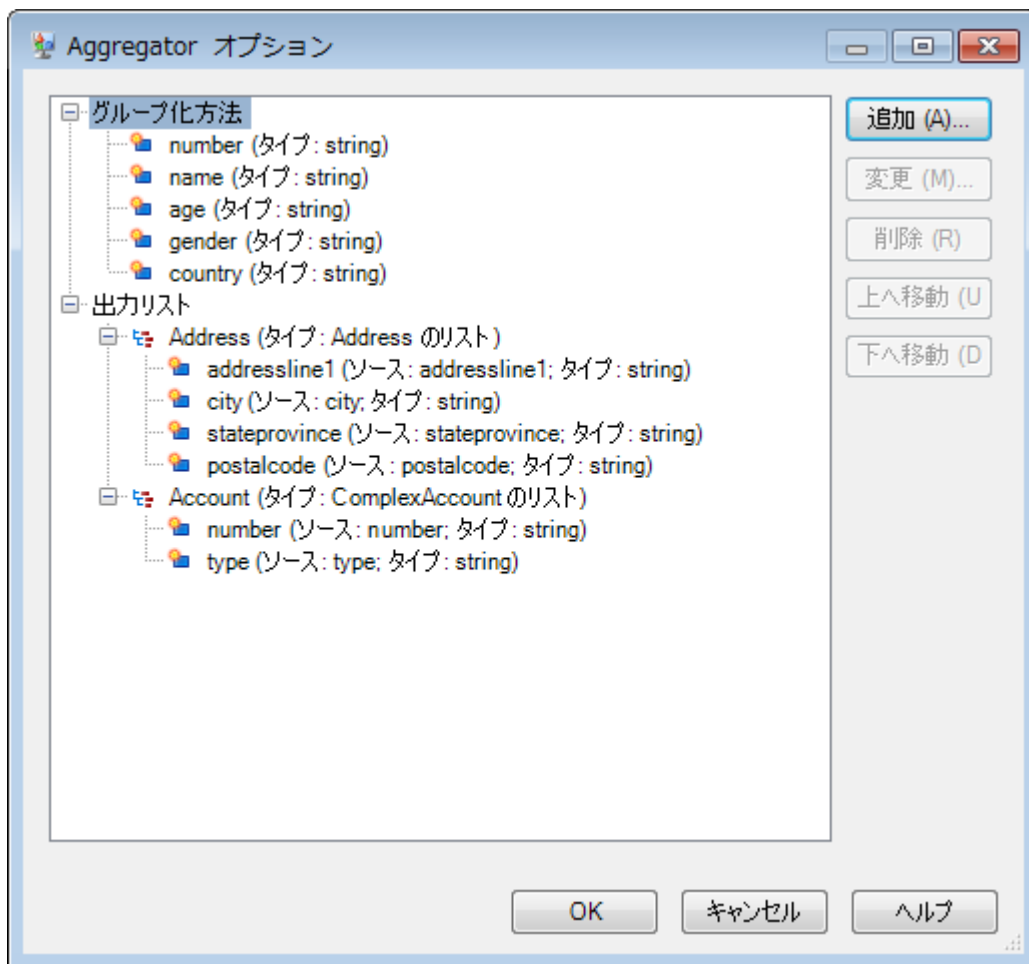


6. **[出力リスト]** を選択し、**[追加]** をクリックします。
7. **[新しいデータ タイプ]** を選択します。これによって、新しい複合要素が定義されます。この複合要素が含むデータの種類の説明を入力します。例えば、複合 XML 要素を作成しているの

で、"Complex" と入力することができます。データ タイプは、任意の名前にすることができます。

8. **[名前]** フィールドに、フィールドに対して使用する名前を入力します。これが XML 要素名にもなります。
9. **[OK]** をクリックします。
10. 作成したフィールドを選択し、**[追加]** をクリックします。
11. **[既存フィールド]** がオンの状態で、子フィールドとして複合要素に追加するフィールドを選択し、**[OK]** をクリックします。
12. 前の 2 つのステップを繰り返して、その他のフィールドを複合要素に追加します。
13. 必要に応じて、その他の複合フィールドを追加します。

完了したら、各レコードに含める各単純フィールドおよび複合フィールドを一覧表示する Aggregator ステージができています。例:



14. **[OK]** をクリックします。

日付および数字パターン

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
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

文字	説明	例
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
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

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
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
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

文字	説明	例
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 1.234×10^3 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	<p>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。</p> <pre>0000</pre> <p>次のように表されます。</p> <pre>0027</pre>
#	<p>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。</p> <pre>####</pre> <p>次のように表されます。</p> <pre>27</pre>

記号	説明
.	<p>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。</p>
-	<p>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。</p>
,	<p>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。</p> <p>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 1,000,0000 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。</p> <pre>#,##,###,#### #####,### ##,###,####</pre>
E	<p>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 (335ページ) を参照してください。</p>
;	<p>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 (334ページ) を参照してください。</p>
%	<p>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。</p> <pre>##%</pre> <p>次のような結果が生成されます。</p> <pre>35%</pre>
¤	<p>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。</p>

記号	説明
'	<p>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。</p> <pre data-bbox="552 420 649 451">' '# '#'</pre> <p>123 は次のように表記されます。</p> <pre data-bbox="552 514 649 546">#123"</pre> <p>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。</p> <pre data-bbox="552 609 730 640"># o' 'clock"</pre>

6 - 設定

このセクションの構成

Oracle LogMiner の設定

339

Oracle LogMiner の設定

Oracle LogMiner は、Oracle データベース用に作成されたログに対して Spectrum からクエリを実行し、アクセスできるようにするバックエンド ユーティリティです。

このユーティリティによって Spectrum™ は、**DB Change Data Reader** ステージの一部として、Oracle データソースのログを読み取り、そのテーブル列に加えられた変更を追跡することができます。

サポートされていないデータタイプとテーブル ストレージ属性

Oracle LogMiner は、以下のデータタイプとテーブル ストレージ属性をサポートしません。

- **BFILE** データタイプ
- 単純な抽象データタイプと、入れ子になった抽象データタイプ (**ADT**)
- コレクション (入れ子になったテーブルと **VARRAY**)
- オブジェクト参照
- 圧縮が有効になっているテーブル
- **SecureFiles**

サポートされているデータベースと、**REDO** ログ ファイル バージョン

LogMiner は、Oracle データベースのバージョン 8.1 以降で動作します。

LogMiner を使用して、Oracle 8.0 データベースからの **REDO** ログ ファイルを分析することもできます。ただし、取得される情報は、使用しているデータベースのバージョンではなく、ログのバージョンに依存します。

例えば、LogMiner を最適な方法で使用するために、サプリメンタル ロギングが有効である場合は、Oracle9i の **REDO** ログ ファイルを拡張して追加情報をキャプチャすることができます。Oracle の古いバージョンで作成された **REDO** ログ ファイルには追加情報がなく、そのために、LogMiner でサポートされる操作とデータタイプに制約が生じる場合があります。

SQL*Loader の制約

Spectrum CDC は、SQL*Loader ユーティリティによって Oracle テーブルに読み込まれたデータをキャプチャできます。ただし、次の制約があります。

1. データは、従来型のパスによってロードする必要があります。Oracle LogMiner がダイレクトパスロードをサポートしないため、Spectrum CDC は、ダイレクトパスロードによってロードされたデータをキャプチャできません。
2. ロード メソッドは、INSERT、APPEND、または REPLACE である必要があります。

TRUNCATE はサポートされていません。TRUNCATE コマンドは、SQL*Loader による TRUNCATE TABLE DDL コマンドの発行を引き起こすためです。Spectrum CDC 機能は、この DDL をキャプチャしないため、TRUNCATE TABLE DDL コマンドの使用によって生じる行の削除は、キャプチャされません。

必要なユーザ権限

以下の表に、Oracle CDC ユーザに必要な最小限のシステム権限を示します。

システム権限	Oracle バージョン
ALTER ANY TABLE	ALL
CONNECT	ALL
LOCK ANY TABLE	ALL
SELECT ANY TRANSACTION	10g またはそれ以降

以下の表に、Oracle CDC ユーザに必要な最小限のオブジェクト権限を示します。

オブジェクト名	権限
Source Tables	LOCK ANY TABLE OR SELECT
PUBLIC.V\$DATABASE	SELECT
PUBLIC.V\$LOGMNR_CONTENTS	SELECT
SYS.DBMS_LOGMNR	EXECUTE
SYS.DBMS_LOGMNR_D	EXECUTE

Oracle LogMiner の詳細については、[こちら](#)を参照してください。

7 - パフォーマンスの最適化

このセクションの構成

最適なフェッチ サイズの決定

342

最適なフェッチ サイズの決定

Read from DB ステージでは、**Read from DB** ステージと **Write to Null** ステージ間で実行時間を記録することで、最適なフェッチ サイズが計算されます。

Spectrum™ Data Integration モジュールを伴う独自アプリケーションを使用して、さまざまなフェッチ サイズの値でテスト ジョブの実行時間をテストするようにしてください。

1. Enterprise Designer でジョブを作成する
2. **[Read from DB]** ステージをキャンバス上にドラッグします。
3. **[Write to Null]** ステージをキャンバス上にドラッグします。
4. 2つのステージ間にチャンネルを作成します。
5. **[Read from DB]** ステージをダブルクリックして、テスト データが含まれているテーブルからデータを読み取るように設定します。
 - a) **[全般]** タブの **[接続]** フィールドで、テスト データが含まれているデータベースを選択します。
 - b) **[SQL の作成...]** をクリックして、選択されているスキーマとデータの読み取り元テーブルを使用する SQL クエリを作成します。
最適なテスト値が得られるように、選択されているテーブルに 1000 レコード以上あることを確認してください。
 - c) **[実行時]** タブで、**[フェッチ サイズ]** チェックボックスをオンにします。
 - d) 付随するフィールドに、1つのインスタンスで読み取るレコード数を入力します。
Spectrum™ Technology Platform は、1000 までのフェッチ サイズによって最適に動作するようにテストされています。
 - e) **[OK]** をクリックします。
6. ジョブを保存します。
7. ジョブを実行します。
[実行の詳細] ウィンドウが表示されます。
8. **[更新]** をクリックします。
9. **[開始]** および **[終了]** の時刻を書き留めます。
10. 徐々にフェッチ サイズを大きくしながらステップ 7～9 を繰り返して、サーバーにとって最適なフェッチ サイズを特定します。

このようにして、お使いの環境で最適なパフォーマンスが得られるフェッチ サイズが明らかになります。

著作権に関する通知

© 2019 Pitney Bowes. All rights reserved. MapInfo および Group 1 Software は Pitney Bowes Software Inc. の商標です。その他のマークおよび商標はすべて、それぞれの所有者の資産です。

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ます。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