pitney bowes

Spectrum™ Technology Platform

バージョン 12.0

データ品質ガイド

目次

8 - ステージ リファレンス

9 - ISO 国コードとモジュール サポート

1 - はじめに

このセクションの構成

データ品質の概要

データ品質には、組織で使用されるデータの精度、適時性、完全性、および一貫性が使用目的に適していることの確認が伴います。Spectrum™ Technology Platformは、以下の機能を提供することによってデータ品質イニシアチブをサポートします。

パーシング

パーシングは、フィールド内にある連続した入力文字を解析して複数のフィールドに分割する処理です。例えば、Name というフィールドに "John A.Smith" という値が格納されている場合、この値を分割して、FirstName フィールドに "John"、MiddleName フィールドに "A"、LastName フィールドに "Smith" を格納することができます。

正規化

正規化機能は、同じタイプのデータを受け取って、同じ形式に格納します。正規化できるデータのタイプとしては、電話番号、日付、名前、住所、ID 番号などがあります。例えば、電話番号は、かっこ、ピリオド、ダッシュなどの数字以外の文字を除去して書式設定できます。

正規化されたデータは書式に一貫性のないデータより正確にマッチングされるので、マッチングまたは重複除去アクティビティを実行する前に、データを正規化する必要があります。

マッチング

マッチングは、目的にとって重要な何らかの方法で相互に関連付けられるレコードを識別するプロセスです。例えば、顧客データから冗長な情報を除去しようとする場合、同じ顧客に対する重複レコードを特定できます。または、マーケティング資料が同じ住所に重複して送られないようにする場合は、同じ世帯に住む顧客のレコードを特定できます。

重複除去

重複除去機能は、1 つのエンティティを表していながら何らかの理由で、場合によっては若干異なるデータで、システムに複数回入力されたレコードを特定します。例えば、異なる部門が同じベンダーに対して異なるベンダー ID を使用してベンダー情報を個別にシステムに入力している場合があります。Spectrum™ Technology Platformを使用すると、このようなレコードをベンダーごとに 1 つのレコードに統合できます。

例外レコードの確認

自動的な方法では確実に処理できないデータがあり、熟練したデータ管理責任者が確認しなければならない場合があります。手動確認が必要になる場合があるレコードの例を次に示します。

- 住所検証エラー
- ジオコーディング エラー
- 信頼性の低いマッチング
- マージ/統合の結果

Business Steward モジュールは、例外レコードを特定および解決することのできる機能の集合です。

2 - パーシング

このセクションの構成

パーシングの概要

パーシングは、フィールド内にある連続した入力文字を解析して複数のフィールドに分割する処理です。例えば、Name というフィールドに "John A.Smith" という値が格納されている場合、この値を分割して、FirstName フィールドに "John"、MiddleName フィールドに "A"、LastName フィールドに "Smith" を格納することができます。

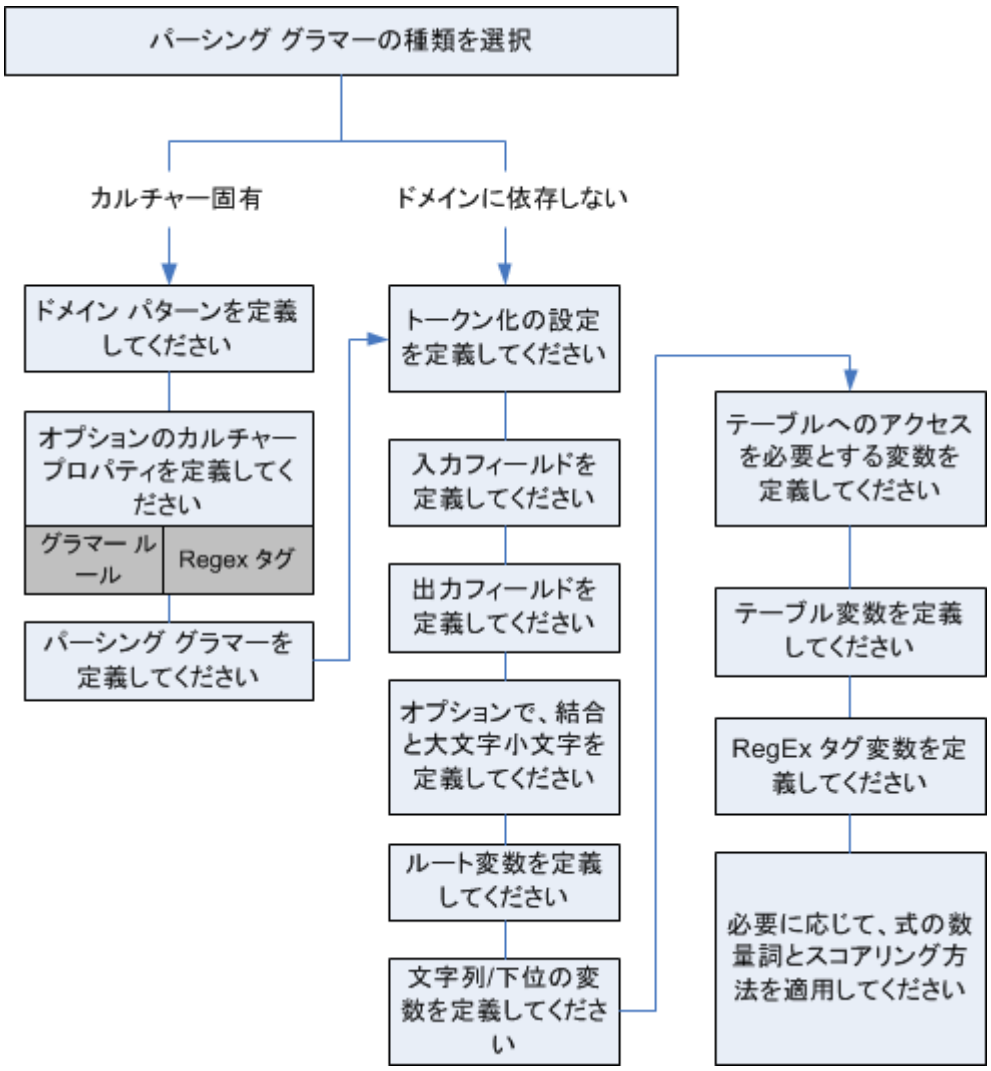
パーシングを行うデータフローを作成するには、Open Parser ステージを使用します。Open Parser では、グラマーと呼ばれるパーシング ルールを記述できます。グラマーは、連続した文字列を *ドメイン パターン*と呼ばれる一連の名前付き要素にマップする一連の式です。ドメイン パターンとは、名前、住所、顧客番号のようなデータ構造として表したい入力データの中の、連続した 1 つ以上のトークンです。ドメイン パターンは、入力データからパース可能な任意の数のトークンで構成できます。ドメイン パターンは、<root> 式としてパーシング グラマーに表現されます。入力データには、そのようなトークンが使用しにくいフォーマット、または混在フォーマットで含まれることがしばしばあります。例:

- 入力データには名と姓に分離したい名前が単一のフィールドに含まれている。
- 入力データには複数のカルチャーの住所が含まれており、特定のカルチャーの住所データのみを抽出したい。
- 入力データには電子メール アドレスが埋め込まれた自由形式テキストが含まれており、電子メール アドレスを抽出して個人データとマッチングし、データベースに格納したい。

グラマーには、*カルチャー固有*のものと *ドメインに依存しない*ものの 2 種類があります。カルチャー固有のパーシング グラマーは、カルチャーや言語 (英語、カナダ英語、スペイン語、メキシコ スペイン語など)、および特定のタイプのデータ (電話番号、個人名など) に関連付けられています。Open Parser ステージがカルチャー固有のパーシングを実行するように設定されている場合、各カルチャーのパーシング グラマーが各レコードに適用されます。最高のパーサー スコアを取得したグラマー (または最初にスコアが 100 になったグラマー) の結果が返されます。また、カルチャー固有のパーシング グラマーでは入力レコードの CultureCode フィールド内の値を使用し、カルチャーのパーシング グラマーに含まれているカルチャー設定に従ってデータを処理できます。カルチャー固有のパーシング グラマーは、親からプロパティを継承できます。ドメインに依存しないパーシング グラマーは、言語にも特定のタイプのデータにも関連付けられていません。ドメインに依存しないパーシング グラマーは、親からプロパティを継承せず、入力データの CultureCode 情報を無視します。

Open Parser は、入力フィールド内の連続した文字列を解析し、*トークン化*という処理によって連続したトークンに分類します。トークン化するとは、入力文字列のセクションをスペースやハイフン、その他の区切り文字列 (トークン化文字とも呼ばれます) でそれぞれのトークンに区切り、分類することです。その後、トークンは指定した出力フィールド内に配置されます。

次の図に、パーシング グラマーを作成する処理を示します。



データフローでのドメインに依存しないパーシング グラマーの定義

ドメインに依存しないパーシング グラマーをデータフローで定義するには、次の手順に従います。

1. Enterprise Designer で、データフローに Open Parser ステージを追加します。
2. キャンバス上の [Open parser] ステージをダブルクリックします。

3. **[ルール]** タブの **[ドメインに依存しないグラマーを定義]** をクリックします。
4. グラマーエディタを使用して、グラマー ルールを作成します。コマンドと変数をテキスト ボックスに入力するか、**[コマンド]** タブに表示されるコマンドを使用することができます。詳細については、グラマー（27ページ）を参照してください。
5. パーシング グラマーのテキスト文字列の切り取り、コピー、貼り付け、検索、および置換を行うには、グラマーエディタ内で右クリックして、該当するコマンドを選択します。
6. 作成したパーシング グラマーをチェックするには、**[検証]** をクリックします。

   検証機能により、エラーが発生した行と列、エラーの説明、エラーの原因となったコマンド名または値など、グラマー構文のエラーが表示されます。

7. **[プレビュー]** タブをクリックして、パーシング グラマーをテストします。
8. パーシング グラマーの作成を終了したら、**[OK]** をクリックします。

カルチャー固有のパーシング

カルチャー固有のパーシング グラマーの定義

カルチャー固有のパーシング グラマーを使用すると、さまざまな言語やカルチャーに対して異なるパーシング ルールを指定できます。これにより、1 つの Open Parser ステージで異なる国のデータ（例えば、米国の電話番号と英国の電話番号）をパースできます。デフォルトでは、それぞれの入力レコードが、Open Parser ステージで指定された順に各カルチャーのパーシング グラマーを使用してパースされます。また、入力レコードで特定のカルチャーのパーシング グラマーを使用する場合は、そのレコードに [CultureCode] フィールドを追加することもできます。詳細については、**レコードへのパーシング カルチャーの割り当て**（12ページ）を参照してください。

> **注：** ドメインに依存しないパーシング グラマーを作成する場合は、「**データフローでのドメインに依存しないパーシング グラマーの定義**（9ページ）」を参照してください。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[ドメイン]** タブをクリックします。
3. **[追加]** をクリックします。
4. **[名前]** フィールドにドメイン名を入力します。
5. **[説明]** フィールドにドメイン名の説明を入力します。

6. 新しい空のドメインを作成する場合は、**[OK]** をクリックします。別のドメインを基に新しいドメインを作成する場合は、以下の操作を実行します。
   a) 別のドメインに基づいて新しいドメインを作成する場合は、**[他のドメインをテンプレートとして使用]** を選択します。
   b) リストにあるドメインを選択します。次のステップで **[OK]** をクリックすると、新しいドメインが作成されます。新しいドメインには、選択したドメイン テンプレートで定義済みのカルチャー固有のパーシング グラマーがすべて含まれます。
   c) **[OK]** をクリックします。
7. グローバル カルチャーのパーシング グラマーを定義します。グローバル カルチャーは、デフォルトのカルチャーであり、カルチャー固有のパーシング グラマーが定義されていないカルチャーを持つレコードをパースするために使用されます。
   a) **[グラマー]** タブで、作成した新しいドメインを選択します。
   b) ドメインをテンプレートから作成した場合、カルチャーが既にリストに含まれていることがあります。
      - カルチャーがリストにある場合は、**[グローバル カルチャー]** を選択し、**[編集]** をクリックします。
      - カルチャーがリストにない場合は、**[追加]** をクリックし、**[グローバル カルチャー]** を選択して **[OK]** をクリックします。
   c) **[グラマー]** タブで、グローバル カルチャーのパーシング グラマーを記述します。**[コマンド]**、**[グラマー ルール]**、および**[RegEx タグ]** のタブを使用して、定義済みのパーシング グラマー要素を挿入できます。定義済みの要素を入力するには、要素を挿入する場所にカーソルを置き、追加する要素をダブルクリックします。

      [コマンド] タブにパーシング コマンドが表示されます。使用可能なコマンドについては、「グラマー（27ページ）」を参照してください。

      [グラマー ルール] タブに [カルチャー プロパティ] ダイアログ ボックスで作成したグラマー ルールが表示されます。グラマー ルールの作成の詳細については、カルチャーのグラマー ルールの定義（32ページ）を参照してください。

      [Regex タグ] タブに [カルチャー プロパティ] ダイアログ ボックスで作成した Regex タグが表示されます。Regex タグの作成の詳細については、カルチャー RegEx タグの定義（33ページ）を参照してください。

   d) 作成したグラマー構文をチェックするには、**[検証]** をクリックします。パーシング グラマーの検証機能により、グラマー構文のエラーが表示されます。表示には、発生したエラー、エラーが発生した行と列、エラーが発生したコマンド、グラマー ルール、または Regex タグが含まれます。
   e) サンプル データを使用してグラマーの結果をテストするには、**[プレビュー]** タブをクリックします。**[入力データ]** に、パースするサンプル データを入力します。1 行につき 1 つの

レコードを入力します。その後、**[プレビュー]** ボタンをクリックします。パースされた出力フィールドが **[結果]** グリッドに表示されます。出力フィールドについては、出力（295ページ）を参照してください。トレースについては、最終パース結果のトレース（38ページ）を参照してください。期待した結果が得られない場合は、**[グラマー]** タブをクリックし、期待した結果が得られるまでパーシング グラマーの編集と代表的な入力データのテストを続行します。

f) グローバル カルチャーのパーシング グラマーの定義が完了したら、**[OK]** をクリックします。

8. 目的のカルチャーごとにカルチャー固有のグラマーを定義します。カルチャー固有のグラマーを追加するには、**[追加]** をクリックし、グローバル カルチャーの場合と同じ手順でグラマーを定義します。手順を繰り返してカルチャーを必要なだけ追加します。
9. カルチャー固有のパーシング グラマーの追加が完了したら、**[OK]** をクリックします。

これで、作成したドメインとカルチャーを Open Parser ステージで使用してパーシングを実行できるようになります。

レコードへのパーシング カルチャーの割り当て

カルチャー固有のパーシング グラマーを使用するように Open Parser ステージを設定すると、カルチャーごとのパーシング グラマーが Open Parser ステージでのカルチャーのリスト順に各入力レコードに適用されます。ただし、特定のカルチャーのパーシング グラマーをレコードに適用する場合は、CultureCode という名前のフィールドを追加できます。このフィールドには、以下の表に示すサポートされているカルチャー コードの 1 つが含まれている必要があります。

カルチャー コード

カルチャー コードは、小文字の言語コード 2 文字と大文字の国または地域コード 2 文字で構成されます。例えば、スペイン語 (メキシコ) であれば "es-MX"、英語 (米国) であれば "en-US" となります。2 文字の言語コードが存在しない場合は、3 文字のコードが使用されます。例えば、ウズベク語 (ウズベキスタン、キリル文字) は "uz-Cyrl-UZ" となります。言語は、2 桁の小文字の言語コードによってのみ指定されます。例えば、"fr" はフランス語の中立的カルチャーを指定し、"de" はドイツ語の中立的カルチャーを指定します。

> **注 :** 異なるパターンに従うカルチャー名が 2 つあります。カルチャー "zh-Hans" (簡体字中国語) と "zh-Hant" (繁体字中国語) は、中立的カルチャーです。カルチャー名は現在の標準を表すものなので、古い名前である "zh-CHS" および "zh-CHT" を使用する理由がないのであれば、こちらを使用してください。

以下の表に、サポートされているカルチャー コードを示します。

言語 (カルチャー/地域)	カルチャーコード
グローバル カルチャー	Global Culture
アフリカーンス語	af
アフリカーンス語 (南アフリカ)	af-ZA
アルバニア語	sq
アルバニア語 (アルバニア)	sq-AL
アラビア文字	ar
アラビア語 (アルジェリア)	ar-DZ
アラビア語 (バーレーン)	ar-BH
アラビア語 (エジプト)	ar-EG
アラビア語 (イラク)	ar-IQ
アラビア語 (ヨルダン)	ar-JO
アラビア語 (クウェート)	ar-KW
アラビア語 (レバノン)	ar-LB
アラビア語 (リビア)	ar-LY

言語 (カルチャー/地域)	カルチャーコード
アラビア語 (モロッコ)	ar-MA
アラビア語 (オマーン)	ar-OM
アラビア語 (カタール)	ar-QA
アラビア語 (サウジアラビア)	ar-SA
アラビア語 (シリア)	ar-SY
アラビア語 (チュニジア)	ar-TN
アラビア語 (U.A.E.)	ar-AE
アラビア語 (イエメン)	ar-YE
アルメニア語	hy
アルメニア語 (アルメニア)	hy-AM
アゼルバイジャン語	az
アゼルバイジャン語 (アゼルバイジャン、キリル文字)	az-Cyrl-AZ
アゼルバイジャン語 (アゼルバイジャン、ラテン文字)	az-Latn-AZ
バスク語	eu

言語 (カルチャー/地域)	カルチャーコード
バスク語 (バスク)	eu-ES
ベラルーシ語	be
ベラルーシ語 (ベラルーシ)	be-BY
ブルガリア語	bg
ブルガリア語 (ブルガリア)	bg-BG
カタロニア語	ca
カタロニア語 (カタロニア)	ca-ES
中国語	zh
中国語（香港特別行政区、中国）	zh-HK
中国語 (マカオ特別行政区)	zh-MO
中国語 (中国)	zh-CN
中国語 (簡体字)	zh-Hans
中国語（シンガポール）	zh-SG
中国語（台湾）	zh-TW

言語 (カルチャー/地域)	カルチャーコード
中国語 (繁体字)	zh-Hant
クロアチア語	hr
クロアチア語 (クロアチア)	hr-HR
チェコ語	cs
チェコ語 (チェコ共和国)	cs-CZ
デンマーク語	da
デンマーク語 (デンマーク)	da-DK
ディベヒ語	dv
ディベヒ語 (モルディブ)	dv-MV
オランダ語	nl
オランダ語（ベルギー）	nl-BE
オランダ語 (オランダ)	nl-NL
英語	en
英語（オーストラリア）	en-AU

言語 (カルチャー/地域)	カルチャーコード
英語 (ベリーズ)	en-BZ
英語（カナダ）	en-CA
英語（カリブ諸国）	en-029
英語（アイルランド）	en-IE
英語（ジャマイカ）	en-JM
英語（ニュージーランド）	en-NZ
英語 (フィリピン)	en-PH
英語（南アフリカ）	en-ZA
英語 (トリニダード・トバゴ)	en-TT
英語 (英国)	en-GB
英語（米国）	en-US
英語 (ジンバブエ)	en-ZW
エストニア語	et
エストニア語 (エストニア)	et-EE

言語 (カルチャー/地域)	カルチャーコード
フェロー語	fo
フェロー語 (フェロー諸島)	fo-FO
ペルシア語	fa
ペルシア語 (イラン)	fa-IR
フィンランド語	fi
フィンランド語 (フィンランド)	fi-FI
フランス語	fr
フランス語（ベルギー）	fr-BE
フランス語（カナダ）	fr-CA
フランス語 (フランス)	fr-FR
フランス語（ルクセンブルク）	fr-LU
フランス語 (モナコ)	fr-MC
フランス語（スイス）	fr-CH
ガリシア語	gl

言語 (カルチャー/地域)	カルチャーコード
ガリシア語 (スペイン)	gl-ES
グルジア語	ka
グルジア語 (グルジア)	ka-GE
ドイツ語	de
ドイツ語（オーストリア）	de-AT
ドイツ語 (ドイツ)	de-DE
ドイツ語（リヒテンシュタイン）	de-LI
ドイツ語（ルクセンブルク）	de-LU
ドイツ語（スイス）	de-CH
ギリシャ文字	el
ギリシャ語 (ギリシャ)	el-GR
グジャラート語	gu
グジャラート語 (インド)	gu-IN
ヘブライ語	he

言語 (カルチャー/地域)	カルチャーコード
ヘブライ語 (イスラエル)	he-IL
ヒンディー語	hi
ヒンディー語 (インド)	hi-IN
ハンガリー語	hu
ハンガリー語 (ハンガリー)	hu-HU
アイスランド語	is
アイスランド語 (アイスランド)	is-IS
インドネシア語	id
インドネシア語 (インドネシア)	id-ID
イタリア語	it
イタリア語 (イタリア)	it-IT
イタリア語（スイス）	it-CH
日本語	ja
日本語 (日本)	ja-JP

言語 (カルチャー/地域)	カルチャーコード
カンナダ語	kn
カンナダ語 (インド)	kn-IN
カザフ語	kk
カザフ語 (カザフスタン)	kk-KZ
コンカニ語	kok
コンカニ語 (インド)	kok-IN
韓国語	ko
韓国語 (韓国)	ko-KR
キルギス語	ky
キルギス語 (キルギスタン)	ky-KG
ラトビア語	lv
ラトビア語 (ラトビア)	lv-LV
リトアニア語	lt
リトアニア語 (リトアニア)	lt-LT

言語 (カルチャー/地域)	カルチャーコード
マケドニア語	mk
マケドニア語 (マケドニア旧ユーゴスラビア共和国)	mk-MK
マレー語	ms
マレー語 (ブルネイ・ダルサラーム)	ms-BN
マレー語 (マレーシア)	ms-MY
マラーティー語	mr
マラーティー語 (インド)	mr-IN
モンゴル語	mn
モンゴル語 (モンゴル)	mn-MN
ノルウェー語	no
ノルウェー語 (ブークモール、ノルウェー)	nb-NO
ノルウェー語 (ニーノシュク、ノルウェー)	nn-NO
ポーランド語	pl
ポーランド語 (ポーランド)	pl-PL

言語 (カルチャー/地域)	カルチャーコード
ポルトガル語	pt
ポルトガル語 (ブラジル)	pt-BR
ポルトガル語 (ポルトガル)	pt-PT
パンジャブ語	pa
パンジャブ語 (インド)	pa-IN
ルーマニア語	ro
ルーマニア語 (ルーマニア)	ro-RO
ロシア語	ru
ロシア語 (ロシア)	ru-RU
サンスクリット語	SA
サンスクリット語 (インド)	sa-IN
セルビア語	sr
セルビア語 (セルビア、キリル文字)	sr-Cyrl-CS
セルビア語 (セルビア、ラテン文字)	sr-Latn-CS

言語 (カルチャー/地域)	カルチャーコード
スロバキア語	sk
スロバキア語 (スロバキア)	sk-SK
スロベニア語	sl
スロベニア語 (スロベニア)	sl-SI
スペイン語	es
スペイン語（アルゼンチン）	es-AR
スペイン語（ボリビア）	es-BO
スペイン語（チリ）	es-CL
スペイン語（コロンビア）	es-CO
スペイン語（コスタリカ）	es-CR
スペイン語 (ドミニカ共和国)	es-DO
スペイン語（エクアドル）	es-EC
スペイン語 (エルサルバドル)	es-SV
スペイン語（グアテマラ）	es-GT

言語 (カルチャー/地域)	カルチャーコード
スペイン語 (ホンジュラス)	es-HN
スペイン語 (メキシコ)	es-MX
スペイン語 (ニカラグア)	es-NI
スペイン語 (パナマ)	es-PA
スペイン語（パラグアイ）	es-PY
スペイン語（ペルー）	es-PE
スペイン語 (プエルトリコ)	es-PR
スペイン語 (スペイン)	es-ES
スペイン語 (スペイン、トラディショナル ソート)	es-ES_tradnl
スペイン語（ウルグアイ）	es-UY
スペイン語（ベネズエラ）	es-VE
スワヒリ語	sw
スワヒリ語 (ケニア)	sw-KE
スウェーデン語	sv

言語 (カルチャー/地域)	カルチャーコード
スウェーデン語 (フィンランド)	sv-FI
スウェーデン語 (スウェーデン)	sv-SE
シリア語	syr
シリア語 (シリア)	syr-SY
タミル語	ta
タミル語 (インド)	ta-IN
タタール語	tt
タタール語 (ロシア)	tt-RU
テルグ語	te
テルグ語 (インド)	te-IN
タイ語	th
タイ語 (タイ)	th-TH
トルコ語	tr
トルコ語 (トルコ)	tr-TR

言語 (カルチャー/地域)	カルチャーコード
ウクライナ語	uk
ウクライナ語 (ウクライナ)	uk-UA
ウルドゥー語	ur
ウルドゥー語 (パキスタン)	ur-PK
ウズベク語	uz
ウズベク語 (ウズベキスタン、キリル文字)	uz-Cyrl-UZ
ウズベク語 (ウズベキスタン、ラテン文字)	uz-Latn-UZ
ベトナム語	vi
ベトナム語 (ベトナム)	vi-VN

グラマー

有効なパーシング グラマーは、次のものを含みます。

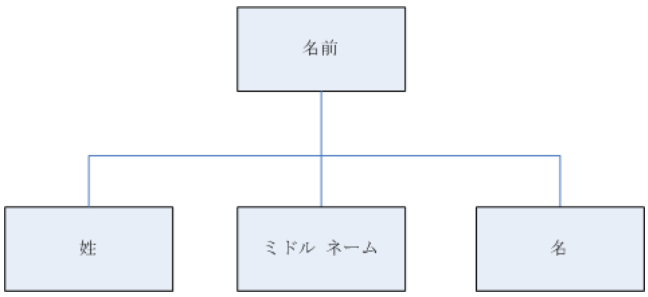
- トークンのシーケンス、つまりドメイン パターンをルール変数として定義するルート変数。
- 有効な文字セット、およびその文字セットの出現によってドメイン パターンの一部と認識されるシーケンスを定義するルール変数。詳細については、**Rule セクションのコマンド**（30ページ）を参照してください。
- パースする入力フィールド。入力フィールドは、パースするソース データ レコード内のフィールドを指定します。

- パースされた結果データ用の出力フィールド。出力フィールドは、パースされた各結果トークンをどこに格納するかを定義します。

有効なパーシング グラマーには、次の目的で、その他のオプション コマンドも含まれます。

- パーシングする入力データのトークン化に使用される文字。トークン化文字は、トークンの開始と終了を決定する、スペースやハイフンなどの文字です。デフォルトのトークン化文字はスペースです。トークン化文字は、連続する文字を一連のトークンに分解する主要な方法です。トークン化コマンドをNONEに設定して、フィールドをトークン化しないこともできます。トークン化を None に設定する場合、グラマー ルールのルール定義内にスペースを含めなければなりません。
- 入力データのトークンに対する大文字と小文字の区別オプション。
- マッチング トークンを区切るための結合文字。
- テーブル内のトークンのマッチング
- テーブル内の複合トークンのマッチング
- Regex タグの定義
- 引用符で囲まれたリテラル文字列
- 式の数量詞 (オプション)。式の数量詞の詳細については、**Rule セクションのコマンド**（30ページ）および**式の数量詞: Greedy、Reluctant、および Possessive な動作**を参照してください。
- グループ化、コメント化、および割り当てのためのその他のインジケータ (オプション)。グループ化した式の詳細については、**グループ化演算子 ( )**を参照してください。

パーシング グラマーのルール変数は、ドメイン パターン内に連続する文字またはトークンの階層化ツリー構造を形成します。<FirstName>、<MiddleName>、および <LastName> というトークンを含む名前入力データに基づいてドメイン パターンを定義するパーシング グラマーを作成できます。



次の入力データを使用します。

```
Joseph Arnold Cowers
```

ドメイン パターン内の 3 つのトークンとして、このデータ文字列を表すことができます。

```
<root> = <FirstName><MiddleName><LastName>;
```

このドメイン パターンのルール変数は次のようになります。

```
<FirstName> = <given>;
<MiddleName> = <given>;
<LastName> = @Table("Family Names");
<given> = @RegEx("[A-Za-z]+");
```

この簡単なサンプル グラマーに基づいて、Open Parser はスペースでトークン化を行い、トークン Joseph を名として解釈します。最初のトークン内の文字は、[A-Za- z]+ という定義にマッチし、トークンが定義済みのシーケンス内にあるからです。必要に応じて、式に別の式を続けることができます。

例

```
<variable> = "some leading string" <variable2>;

<variable2> = @Table ("given") @RegEx("[0-9]+");
```

グラマー ルールとは、変数が 1 つまたは複数の式に等しい文法的な文です。各グラマー ルールは次の形式になります。

```
<rule> = expression [| expression...];
```

グラマー ルールは次のルールに従わなければなりません。

- <root> は特殊な変数名で、ドメイン パターンを定義しているため、グラマーの中で最初に実行されるルールです。<root> は、グラマー内の他のルールから参照できません。
- <rule> 変数は、自分自身を直接的または間接的に参照できません。ルール A がルール B を参照し、ルール B がルール C を参照し、ルール C がルール A を参照する場合、循環参照が作成されます。循環参照は許可されていません。
- <rule> 変数は、1 つまたは複数の式と同等です。
- 各 expression 式は、OR で分離されます。OR は、パイプ文字 (|) で表されます。
- 式は一度に 1 つずつ調べられます。最初にマッチする expression が選択されます。それ以降の式は調べられません。
- 変数名は、英字、数字、アンダースコア (_)、およびハイフン (-) で構成することができます。変数の名前は、任意の有効な文字で開始することができます。指定した出力フィールド名がこの形式に準拠していない場合、エイリアス機能を使用して、変数名を出力フィールドにマップしてください。

式は次のいずれかのタイプになります。

- 別の変数
- 単一引用符または二重引用符で囲まれた 1 つ以上の文字からなる文字列です。例:

  ```
  "McDonald" 'McDonald' "O'Hara" 'O\'Hara' 'D"har' "D\"har"
  ```

- テーブル

- CompoundTable
- RegEx コマンド

コマンドのメタ文字

Open Parser は、%Tokenize コマンドおよび @RegEx コマンドで、Java Regex 文字クラスのメタ文字の標準セットをサポートしています。メタ文字とは、パターン マッチングにおいて特殊な意味を持つ文字です。サポートされているメタ文字には次のものがあります。

```
([{\^-$|]})?*+.
```

メタ文字を通常の文字として強制的に扱うには、次の 2 つの方法があります。

- メタ文字の前にバックスラッシュを付ける
- \Q (引用符の開始) と \E (引用符の終了) で囲む

%Tokenize は、Java 正規表現全体ではなく、Java 正規表現の文字クラスのルールに従います。

一般的に、文字セットでは次の文字が予約されています。

- '[' および ']' は、別のセットを示します。
- '-' は、他の 2 つの文字間にある場合、1 つのメタ文字です。
- '^' は、セット内の最初の文字である場合、1 つのメタ文字です。
- '&&' は、他の 2 つの文字間にある場合、複数のメタ文字です。
- '\' は、続く文字がリテラルであることを意味します。

ある文字をメタ文字として扱うかどうかに疑問がある場合や、文字をリテラルとして扱いたい場合は、バックスラッシュを使用してその文字をエスケープします。

Header セクションのコマンド

ここでは、Header セクションのコマンドについて説明します。一部のコマンドはオプションです。オプションのコマンドについては、デフォルトの値または動作を明記します。

- **Tokenize** (オプション)
- **Tokenize (None)**
- **InputField** (入力フィールドが使用されていない場合は必須)
- **InputFields** (入力フィールドが使用されていない場合は必須)
- **OutputFields** (必須)
- **IgnoreCase** (オプション)
- **Join** (オプション)

Rule セクションのコマンド

Rule セクションのコマンドには次のものがあります。

- **RegEx**
- テーブル
- **CompoundTable**
- トークン
- スコアリング
- **RuleID**
- **<root> 変数**
- **rule|rule**
- グループ化演算子 ( )
- **Min/Max 回数演算子 {min,max}**
- **Exact 回数演算子 {exact}**
- 代入演算子 (=)
- **OR 演算子 (|)**
- **End-of-Rule 演算子 (;)**
- コメント演算子 (!)
- **0 または 1 回の出現を表す数量詞 (?)**
- **0 回以上の出現を表す数量詞 (*)**
- **1 回以上の出現を表す数量詞 (+)**
- **式の数量詞: Greedy、Reluctant、および Possessive な動作**

カルチャー

カルチャーは、カルチャー固有のパーシング グラマーを構成するための主要な概念です。カルチャーを使用すると、さまざまなカルチャーや言語に対して異なるパーシング ルールを作成できます。カルチャーは次の階層になっています。

- **グローバル カルチャー**: グローバル カルチャーはカルチャーに依存せず、言語にとらわれません。すべてのカルチャーおよび言語にわたるパーシング グラマーを作成するには、グローバル カルチャーを使用してください。
  - **言語**: 言語は、特定のカルチャー/地域ではなく、言語に関連付けられています。例: 英語。
    - **カルチャー/地域**: カルチャー/地域は、言語と国または地域に関連付けられています。例: 英国の英語、米国の英語。

カルチャー階層において、カルチャー/地域の親は言語であり、言語の親はグローバル カルチャーです。

カルチャー/地域は、親の言語のプロパティを継承します。言語は、グローバル カルチャーのプロパティを継承します。したがって、ある言語を共有する複数の国々で使用するため、その言語でパーシング グラマーを定義することができます。その後、基本となる言語カルチャーと同じ言語を共有しているが、固有の住所指定、名前指定、あるいは他の国または地域的な違いを持つ特定の国または地域用に特殊化されたパーシング グラマーで、その言語のグラマー ルールをオーバーライドすることができます。

カルチャーの継承を使用して、カルチャー コードが割り当てられているが、そのカルチャー コード用のグラマー ルールが定義されていない入力レコードをパースすることもできます。この場合、Open Parser は、グラマー ルールが割り当てられている言語コードを検索します。そのような言語コードが存在しない場合、Open Parser はグローバル カルチャー内で割り当て済みグラマー ルールを検索します。

ドメインエディタは、言語コードとカルチャー コードを組み合わせて使用し、言語とカルチャー/地域をそれぞれ表します。

カルチャーのグラマー ルールの定義

カルチャーのグラマー ルールを使用すると、グローバル カルチャーのパーシング グラマーの一部を、カルチャーや言語に特有の文字列、コマンド、または式で置き換えることができます。グラマー ルールを定義することで、グローバル カルチャーのパーシング グラマーの一部をレコードのカルチャーや言語に基づいてカスタマイズできます。これは、カルチャーごとにまったく別のパーシング グラマーを作成せず、代わりにグローバル カルチャーのグラマーを使用してカルチャーごとにグローバル カルチャー グラマーの特定の部分のみをカスタマイズする場合に役立ちます。

このトピックでは、カルチャーのグラマー ルールを作成する方法について説明します。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[カルチャー]** タブをクリックします。

   サポートされている全カルチャーの一覧は、レコードへのパーシング カルチャーの割り当て（12ページ）を参照してください。

3. グラマー ルールを追加するカルチャーを選択し、**[プロパティ]** をクリックします。
4. **[グラマー ルール]** タブをクリックします。表示される情報には、選択したカルチャーおよび関連付けられたソース カルチャー用に定義されたグラマー ルール名、定義済みのグラマー ルールの値、説明などがあります。
5. **[追加]** をクリックします。
6. **[名前]** フィールドにグラマー ルールの名前を入力します。
7. **[説明]** フィールドにグラマー ルールの説明を入力します。
8. **[値]** フィールドにグラマー ルールを入力します。

グラマー ルールは、任意の有効な変数、文字列、コマンド、またはグループ化した式です。詳細については、グラマー（27ページ）を参照してください。

9. テキスト ボックスの値をスクロールせずに表示するには、**[右端での折り返しを有効にする]**を選択します。
10. **[OK]** をクリックします。

入力したグラマー ルールの値が検証されます。値にグラマー構文のエラーが含まれる場合は、発生したエラーの説明、エラーが発生した行と列、エラーが発生したコマンド、グラマー ルール、または Regex タグに関するメッセージが表示されます。

グラマー ルールの例

欧米の名前をパースするグラマーがあります。おそらくパターンの構造はすべてのカルチャーで同じであり(<FirstName><MiddleName><LastName>)、ルールの多くは同じパターンまたはテーブルにマッチする可能性があります。ただし、姓についてカルチャー固有のテーブルもあるので、レコードのカルチャー コードに基づいて適切なテーブルを使用することもできます。

そのために、グローバル カルチャー内の <LastName> 要素をカルチャー固有のテーブルへの参照で置き換えた、各カルチャーのグラマー ルールを定義することができます。例えば、オランダの姓のテーブルがある場合、オランダ (nl) のカルチャーのグラマー ルールを次のように作成します。

[名前]: LastName
[説明]: オランダの姓
[値]: `@Table("Dutch Last Names");`

カルチャー RegEx タグの定義

このトピックでは、カルチャー固有のパーシング グラマーの定義時にカルチャー Regex タグを定義する方法について説明します。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[カルチャー]** タブをクリックします。サポートされているカルチャーの一覧が **[カルチャー]** タブに表示されます。サポートされている全カルチャーの一覧は、レコードへのパーシング カルチャーの割り当て（12ページ）を参照してください。
3. リストからカルチャーを選択し、**[プロパティ]** をクリックします。**[カルチャー プロパティ]** ダイアログ ボックスが表示されます。
4. **[Regex タグ]** タブをクリックします。表示される情報には、選択したカルチャーおよび関連付けられたソース カルチャー用に定義された Regex タグ名や、Regex タグの値、説明などがあります。

5. **[追加]** または **[変更]** をクリックします。
6. **[名前]** テキスト ボックスに Regex タグの名前を入力します。

   選択したカルチャーに既に存在する名前を入力した場合は、警告アイコンが点滅します。別の名前を入力するか、このダイアログ ボックスを閉じて、既存の Regex タグを削除してから、**[追加]** を再度クリックしてください。

7. **[説明]** テキスト ボックスに Regex タグの説明を入力します。
8. **[値]** テキスト ボックスに Regex タグの値を入力します。

   値には任意の有効な正規表現を指定できますが、空の文字列とのマッチングはできません。

   ドメインエディタにはいくつかの Regex タグが定義済みで、これを使用してカルチャー プロパティを定義できます。パーシング グラマーのトークン化文字の定義にも、これらの Regex タグを使用できます。

   定義済みの Regex タグを変更するか、それをコピーして独自の Regex タグを作成することができます。オーバーライド プロパティを使用して、特定の言語用に特殊な Regex タグを作成することもできます。

   - 文字: 任意の言語の任意の文字。この Regex タグには、キリル文字のスクリプト、アジア言語のスクリプト、タイ語のスクリプトなど、使用するスクリプトの違いによって、複数の言語用のオーバーライドが含まれます。
   - 小文字: 対応する大文字がある小文字。
   - 番号: 任意のスクリプトの任意の数字。
   - 句読文字: 任意の句読文字。
   - 大文字: 対応する小文字がある大文字。
   - 空白: 任意の空白または非表示の区切り文字。

9. **[OK]** をクリックします。

カルチャーのインポートとエクスポート

カルチャーは作成するだけでなく、どこか別の場所に作成したカルチャーをインポートしたり、ドメインエディタ で作成したカルチャーをエクスポートしたりすることもできます。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[カルチャー]** タブをクリックします。
3. **[インポート]** または **[エクスポート]** をクリックします。
4. 以下のいずれかの方法を実行します。

   - カルチャーをインポートする場合は、その場所に移動してカルチャーを選択します。**[開く]** をクリックします。インポートしたカルチャーがドメイン エディタに表示されます。

- カルチャーをエクスポートする場合は、エクスポートしたカルチャーを保存したい場所に移動して選択します。**[保存]** をクリックします。エクスポートしたカルチャーが保存され、ドメイン エディタに戻ります。

ドメイン

ドメインの追加

ドメインは、名前、住所、電話番号のデータのようなデータのタイプを表します。ドメインは、入力データ内の連続する 1 つ以上のトークンを表すパターンで構成されます。このパターンは、一般的にはパースに必要で、1 つ以上のカルチャーに関連付けられます。

このトピックでは、カルチャー固有のパーシング グラマーの定義時に ドメインエディター でドメインを追加する方法について説明します。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[ドメイン]** タブをクリックします。
3. **[追加]** をクリックします。
4. **[名前]** フィールドにドメイン名を入力します。
5. **[説明]** フィールドにドメイン名の説明を入力します。
6. 新しい空のドメインを作成する場合は、**[OK]** をクリックします。別のドメインを基に新しいドメインを作成する場合は、以下の操作を実行します。
   a) 別のドメインに基づいて新しいドメインを作成する場合は、**[他のドメインをテンプレートとして使用]** を選択します。
   b) リストにあるドメインを選択します。次のステップで **[OK]** をクリックすると、新しいドメインが作成されます。新しいドメインには、選択したドメイン テンプレートで定義済みのカルチャー固有のパーシング グラマーがすべて含まれます。
   c) **[OK]** をクリックします。

ドメインの変更

ドメインは、名前、住所、電話番号のデータのようなデータのタイプを表します。ドメインは、入力データ内の連続する 1 つ以上のトークンを表すパターンで構成されます。このパターンは、一般的にはパースに必要で、1 つ以上のカルチャーに関連付けられます。

このトピックでは、ドメインを変更する方法について説明します。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[ドメイン]** タブをクリックします。

3. リストでドメインを選択し、**[変更]** をクリックします。**[ドメインの変更]** ダイアログ ボックスが表示されます。
4. 説明情報を変更します。
5. ドメインの説明のみを変更する場合は、**[OK]** をクリックします。テンプレート ドメインを更新し、その変更内容を変更中のドメインに追加する場合は、次の手順を続行します。
6. **[他のドメインをテンプレートとして使用]** を選択して、ドメイン テンプレートに対して行われた変更を継承します。
7. リストからドメイン パターン テンプレートを選択します。次のステップで **[OK]** をクリックすると、ドメイン パターンが変更されます。変更されたドメイン パターンには、選択したドメイン パターン テンプレートで定義済みのカルチャー固有のパーシング グラマーがすべて含まれます。選択したドメイン パターン内のパーシング グラマーが、ドメイン パターン テンプレートのパーシング グラマーで上書きされます。
8. **[OK]** をクリックします。

作業の結果を確認するには、次の手順に従います。

1. **NameParsing** というドメイン パターンを作成し、グローバル カルチャー、en、および en-US 用のパーシング グラマーを定義します。
2. **NameParsing2** というドメイン パターンを作成し、**NameParsing** をドメイン パターン テンプレートとして使用します。**NameParsing2** は正確なコピーとして作成され、グローバル カルチャー、en、および en-US 用のパーシング グラマーを含みます。
3. グローバル カルチャー グラマーのグラマー ルールの一部を変更することによって、**NameParsing** のカルチャー固有のパーシング グラマーを変更し、en-CA を新しいカルチャーとして追加します。
4. [ドメイン] タブで **NameParsing2** を選択して [変更] をクリックし、**NameParsing** をドメイン パターン テンプレートとして再度使用します。

結果は次のとおりです。

- グローバル カルチャー パーシング グラマーが更新されます (変更を加えていた場合は、その変更が上書きされます)。
- カルチャー en および en-US は変更されません (ただし、これらがターゲット ドメインで変更されている場合は、en および en-US は NameParsing のバージョンに戻ります)。
- en-CA 用のカルチャー固有のグラマーが追加されます。

ドメインの削除

ドメインは、名前、住所、電話番号のデータのようなデータのタイプを表します。ドメインは、入力データ内の連続する 1 つ以上のトークンを表すパターンで構成されます。このパターンは、一般的にはパースに必要で、1 つ以上のカルチャーに関連付けられます。

このトピックでは、ドメインを削除する方法について説明します。

1. Enterprise Designer で、**[ツール] > [Open Parser ドメインエディタ]** を選択します。
2. **[ドメイン]** タブをクリックします。
3. リストにあるドメインを選択します。
4. **[削除]** をクリックします。

   ドメインが1つ以上のカルチャー固有のパーシング グラマーと関連付けられている場合、ドメインの削除に対する確認を求めるメッセージが表示されます。このドメインにカルチャー固有のパーシング グラマーが関連付けられていない場合、選択したドメインの削除を確認するメッセージが表示されます。

5. **[はい]** をクリックします。ドメインと、このドメインに関連付けられているカルチャー固有のパーシング グラマーが削除されます。

ドメインのインポートとエクスポート

ドメインは作成するだけでなく、どこか別の場所に作成したドメインをインポートしたり、ドメインエディタ で作成したドメインをエクスポートしたりすることもできます。

1. **[ドメイン]** タブをクリックします。**[ドメイン]** タブが表示されます。
2. **[インポート]** または **[エクスポート]** をクリックします。
3. 以下のいずれかの方法を実行します。

   - ドメインをインポートする場合は、その場所に移動してドメイン名を選択します。**[開く]** をクリックします。インポートしたドメインがドメイン エディタに表示されます。
   - ドメインをエクスポートする場合は、エクスポートしたドメインを保存したい場所に移動して選択します。**[保存]** をクリックします。エクスポートしたドメインが保存され、ドメインエディタに戻ります。

パーシング結果の分析

最終パース結果のトレース

Open Parser のトレース詳細機能は、入力フィールドがトークンごとに出力フィールドの値にどのようにパースされたかを視覚的に表示します。トレース機能により、マッチング結果、非マッチング結果、および中間結果が表示されます。

最終パース結果には、パーシング グラマー ツリーと結果出力が表示されます。マッチング処理の結果のみを確認したい場合は、このビューを使用してください。これはデフォルトのビューです。

1. Enterprise Designer で、パーシング結果をトレースする Open Parser ステージが含まれているデータフローを開きます。
2. キャンバス上の [Open Parser] ステージをダブルクリックします。
3. **[プレビュー]** タブをクリックします。
4. パースするサンプル データを入力し、**[プレビュー]** ボタンをクリックします。
5. [トレース] 列で、**[ここをクリック...]** リンクをクリックするとトレース図が表示されます。

   パーシング グラマーのツリー ビューには、選択したオプションに応じて、次の要素が 1 つまたは複数表示されます。

   - <root> 変数。ツリーの最上位ノードは、<root> 変数です。
   - <root> 変数で定義された式。第 2 レベルのノードは、<root> 変数で定義された式です。<root> 式は、出力フィールドの名前も定義します。
   - 第 2 レベル ノードの変数定義。第 3 レベル ノードとその下の各レベルは、各 <root> 式の定義です。他の変数、エイリアス、またはルール定義を式の定義にすることができます。
   - 出力となる値とトークン。ツリーの最下位ノードには、パーシング グラマーで連続した各トークンに割り当てられた値が表示されます。
   - パーシング グラマーの関連要素のパーサー スコア。パーサー スコアは、ルート式の下位から上位に決定されます。例えば、ある式パターンの重みが 80 で、上位のルールの重みが 75 の場合、上位の式の最終的なスコアは、子のスコアと上位のスコアから作成されます。この例では、60% になります。
   - 見やすくなるよう、改行なしの空白文字 (上向きの括弧) として **[入力データ]** テキスト ボックスに表示される空白文字。トークンとして使用されない区切り文字は、灰色で表示されます。

6. **[情報]** フィールドで **[最終パース結果]** を選択します。

注： パーシング イベントをステップ スルーするには、パーシング イベントのステップ スルー（39ページ）を参照してください。

7. **[詳細レベル]** リストで、いずれかのオプションを選択します。
   - **マッチしない式を非表示**。マッチまたは非マッチ結果となるブランチを表示します。マッチにならないルート式のブランチは、省略記号ボタンとして表示されます。マッチにならないブランチを見たい場合は、省略記号ボタンをダブルクリックしてください。
   - **マッチしないルート式を非表示**。マッチまたは非マッチ結果を含むルート式のすべてのブランチを表示します。その他のルート式は表示されません。
   - **すべてのルートを表示**。すべてのルート式を表示します。ルートにマッチ結果がない場合、そのルート式の表示は省略記号ボタンを使用して折りたたまれます。
   - **すべての式を表示**。ルート式とすべてのブランチを表示します。ルート式は省略記号ボタンとして表示されなくなります。代わりに、ブランチ内の各式のルールが表示されます。

   マッチしない式を非表示にする詳細レベル ビューを選択し、現在表示されていないルート式を選択した場合、トレース詳細によって、詳細レベルの選択がルート式の最小数を示すリスト項目に変更されますが、ルート式は引き続き表示されます。
8. **[スコアを表示]** をクリックすると、ルート式、変数式、マッチング結果、および非マッチング結果のパーサー スコアが表示されます。
9. **[ズーム]** フィールドで、ツリー ビューのサイズを選択します。
10. **[ルート節]** のリストで、いずれかのオプションを選択して、ルート式ツリーのそのブランチを表示します。

    トレース図の式ブランチをクリックすると、**[ルート節]** リストが更新され、選択された節が表示されます。省略記号をダブルクリックすると、折りたたまれた式が表示されます。
11. 必要な操作が終了したら、**[OK]** をクリックします。**[OK]** をクリックすると、詳細レベル、スコア表示、およびズーム コントロールの設定が保存されます。

パーシング イベントのステップ スルー

Open Parser のトレース詳細ビューでは、マッチング処理でのイベントごとのステップを示す図を表示できます。マッチング処理のトラブルシューティングを行っていて、各トークンがどのように評価されるか、パーシング グラマーのトークン化、および 1 トークンごとのマッチング結果を確認したい場合は、このビューを使用してください。

1. Enterprise Designer で、パーシング結果をトレースする Open Parser ステージが含まれているデータフローを開きます。
2. キャンバス上の [Open Parser] ステージをダブルクリックします。

3. **[プレビュー]** タブをクリックします。
4. パースするサンプル データを入力し、**[プレビュー]** ボタンをクリックします。
5. [トレース] 列で、**[ここをクリック...]** リンクをクリックするとトレース図が表示されます。

   パーシング グラマーのツリー ビューには、選択したオプションに応じて、次の要素が 1 つまたは複数表示されます。

   - <root> 変数。ツリーの最上位ノードは、<root> 変数です。
   - <root> 変数で定義された式。第 2 レベルのノードは、<root> 変数で定義された式です。<root> 式は、出力フィールドの名前も定義します。
   - 第 2 レベル ノードの変数定義。第 3 レベル ノードとその下の各レベルは、各 <root> 式の定義です。他の変数、エイリアス、またはルール定義を式の定義にすることができます。
   - 出力となる値とトークン。ツリーの最下位ノードには、パーシング グラマーで連続した各トークンに割り当てられた値が表示されます。
   - パーシング グラマーの関連要素のパーサー スコア。パーサー スコアは、ルート式の下位から上位に決定されます。例えば、ある式パターンの重みが 80 で、上位のルールの重みが 75 の場合、上位の式の最終的なスコアは、子のスコアと上位のスコアから作成されます。この例では、60% になります。
   - 見やすくなるよう、改行なしの空白文字 (上向きの括弧) として **[入力データ]** テキスト ボックスに表示される空白文字。トークンとして使用されない区切り文字は、灰色で表示されます。

6. マッチと非マッチは、トレース図で次のように色分けされます。

   - 緑色のボックスは、最終的な成功結果の一部であるマッチを示します。
   - 赤色のボックスは、非マッチを示します。
   - 黄色のボックスは、イベントのステップ スルー時に最終的にはロールバックされる中間マッチを示します。中間マッチは、[パーシング イベントのステップ スルー] にのみ表示されます。
   - 灰色のボックスは、そのトークンを別の式用に解放するためロールバックされた中間マッチを示します。中間マッチは、[パーシング イベントのステップ スルー] にのみ表示されます。

7. **[情報]** リストで **[パーシング イベントのステップ スルー]** を選択します。
8. **[詳細レベル]** リストで、いずれかのオプションを選択します。

   - **マッチしない式を非表示**。マッチまたは非マッチ結果となるブランチを表示します。マッチにならないルート式のブランチは、省略記号ボタンとして表示されます。マッチにならないブランチを見たい場合は、省略記号ボタンをダブルクリックしてください。
   - **マッチしないルート式を非表示**。マッチまたは非マッチ結果を含むルート式のすべてのブランチを表示します。その他のルート式は表示されません。
   - **すべてのルートを表示**。すべてのルート式を表示します。ルートにマッチ結果がない場合、そのルート式の表示は省略記号ボタンを使用して折りたたまれます。

- **すべての式を表示**。ルート式とすべてのブランチを表示します。ルート式は省略記号ボタンとして表示されなくなります。代わりに、ブランチ内の各式のルールが表示されます。

マッチしない式を非表示にする詳細レベル ビューを選択し、現在表示されていないルート式を選択した場合、トレース詳細によって、詳細レベルの選択がルート式の最小数を示すリスト項目に変更されますが、ルート式は引き続き表示されます。

9. **[スコアを表示]** をクリックすると、ルート式、変数式、マッチング結果、および非マッチング結果のパーサー スコアが表示されます。
10. **[ズーム]** フィールドで、ツリー ビューのサイズを選択します。
11. **[ルート節]** のリストで、いずれかのオプションを選択して、ルート式ツリーのそのブランチを表示します。

    トレース図の式ブランチをクリックすると、**[ルート節]** リストが更新され、選択された節が表示されます。省略記号をダブルクリックすると、折りたたまれた式が表示されます。

12. デフォルトでは、**[選択されたノードまで自動的にステップ]** チェック ボックスが選択されます。このチェック ボックスが選択されている状態で **[再生]** ボタンをクリックすると、イベントが最初から実行され、選択したノードまたはそのいずれかの子で発生した最初のイベントで停止します。停止せずにすべてのイベントを再生するには、**[再生]** ボタンをクリックする前にこのチェック ボックスをクリアします。
13. **[遅延再生 (秒)]** フィールドで、再生の速度を制御するための遅延を指定します。
14. **[再生]** ボタンをクリックして、パーシング イベントの実行を開始します。
15. 必要な操作が終了したら、**[OK]** をクリックします。

個人名の分割

名前データがすべて 1 つのフィールドに格納されている場合、姓、名、尊称など、名前の各パートごとのフィールドに名前を分割することができます。これらのパースされた名前要素は、名前のマッチング、名前の正規化、複数レコード名の統合など、他の自動化処理に使用できます。

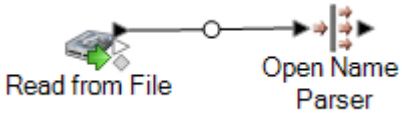
1. 以下のテーブルをまだ Spectrum™ Technology Platform サーバーにロードしていない場合は、ロードします。

   - Open Parser Base
   - Open Parser Enhanced Names

   Data Normalization モジュールのデータベース ロード ユーティリティを使用して、これらのテーブルをロードします。テーブルのロード手順については、『*インストール ガイド*』を参照してください。

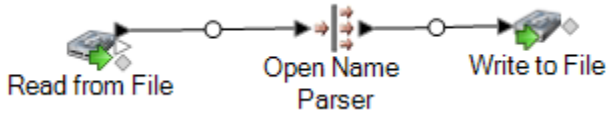
2. Enterprise Designer で、新しいデータフローを作成します。
3. ソース ステージをキャンバスにドラッグします。
4. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
5. Open Name Parser ステージをキャンバス上にドラッグし、ソース ステージに接続します。

   例えば、Read from File ステージを使用する場合、データフローは次のようになります。



6. シンク ステージをキャンバス上にドラッグし、Open Name Parser ステージを接続します。

   例えば、Write to File シンク ステージを使用する場合、データフローは次のようになります。



7. シンク ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。

個人名を構成パートにパースし、名前の各パートをそれぞれのフィールドに配置できるデータフローが作成されました。

パーシング用データフロー テンプレート

英語の名前の分割

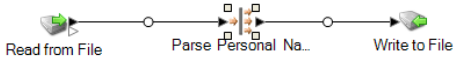
このデータフロー テンプレートは、個人名 (例えば "John P. Smith") を受け取り、これをパースして名、ミドル ネーム、姓に分割し、性別データを追加する方法を示します。

ビジネス シナリオ

ある保険会社に勤務するあなたは、見込み客の性別に基づいて個人向けの保険料見積もりを送付しようとしています。入力データには名前データがフルネームで記録されていますが、このデー

タをパースして [First]、[Middle]、および [Last] の各名前フィールドに貼り付ける必要があります。また、個人の性別を入力データから判別することも必要です。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[個人名をパース]** を選択します。

このデータフローでは、次のものが必要となります。

- Universal Name モジュール
- Open Parser Base テーブル
- Open Parser Enhanced Names テーブル

このデータフローでは、データをファイルから読み取り、Open Name Parser ステージで処理します。Open Name Parser は Universal Naming モジュールの一部です。このデータフローでは、各名前に対して次の処理が行われます。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前が含まれています。

Open Name Parser

Open Name Parser では、名前フィールドをチェックして、Spectrum™ Technology Platform 名前データベース ファイルに格納されている名前データと比較します。この比較結果に基づいて、名前データが [First]、[Middle]、[Last] の各名前フィールドに分割されます。

Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも [FirstName]、[MiddleName]、[LastName]、[EntityType]、[GenderCode]、および [GenderDeterminationSource] の各フィールドがあります。

アラブ系の名前の分割

このテンプレートでは、西洋式のアラブ系の名前をコンポーネントにパースする方法を示します。パーシング ルールに従って **[名前]** フィールド内の各トークンを分割し、それらを **[Kunya]**、**[Ism]**、**[Laqab]**、**[Nasab]**、および **[Nisba]** の 5 つのフィールドにコピーします。これらの出力フィールドは、アラブ系の名前を構成する 5 つの要素を表します。詳細については、ビジネス シナリオで説明します。

ビジネス シナリオ

ある銀行では、アラビア語を母語とする顧客に提供するカスタマー サービスをより充実させるために、アラブ系の氏名のしくみを理解しようと取り組んでいます。複数の顧客から、利用明細に記載された氏名の表記が正しくないと苦情が寄せられました。あなたが所属するマーケティング グループでは、顧客との連絡を密にするために、アラビア語を母語とする顧客を対象としたマーケティング キャンペーンと電話サポートの強化を進めています。

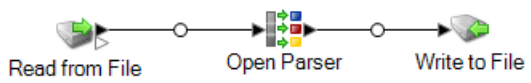
アラブ系の氏名表記を理解するために、インターネット上に資料を探したところ、次のサイトにアラブ系の名前について解説されていました。

- **en.wikipedia.org/wiki/Arabic_names**
- **heraldry.sca.org/laurel/names/arabic-naming2.htm**

アラブ系の氏名の表記規則は、Ism、Kunya、Nasab、Laqab、および Nisba という名前要素で構成されます。

- Ism は、アラブ系のメインの名前であり、個人の "名" に相当します。
- 多くのケースで、個人の最初に生まれた男児の名前を示す Kunya が Ism の代わりに使われます。
- Nasab は、父系の血統を表す名前です。息子を意味する "イブン" または "ビン" または娘を意味する "ビント" という単語を使って、個人の血統を示します。
- Laqab は、個人の特徴を表すために使用されます。例えば、"al-Rashid" は正直者、正しく導かれた者を意味し、"al-Jamil" は美しいことを意味します。
- Nisba は、個人の職業、出身地、または一族 (部族、家系など) を表します。この名前は、数世代の家族に受け継がれます。アラブ系の名前要素のなかで、Nisba は西洋式の "姓" に最も近いものと言えるでしょう。例えば、"al-Filistin" はパレスティナ人を意味します。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ParseArabicNames]** を選択します。このデータフローでは、Data Normalization モジュールが必要です。

このデータフローでは、データをファイルから読み取り、Open Parser ステージで処理します。入力ファイルの各データ行に対し、このデータフローで以下の操作が行われます。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前が含まれています。

Open Parser

このステージでは、ドメインエディタで作成されたカルチャー固有ドメイン グラマーを使うか、ドメインに依存しないグラマーを使うかを定義します。ドメインエディタで作成したカルチャー固有のパーシング グラマーは、カルチャーとドメインに関連付けられた、検証済みのパーシング グラマーです。Open Parser で作成したカルチャーに依存しないパーシング グラマーは、カルチャーとドメインに関連付けられていない、検証済みのパーシング グラマーです。

このテンプレートでは、パーシング グラマーはドメインに依存しないグラマーとして定義されています。

Open Parser ステージでは、次のようにコマンドと式が定義されたパーシング グラマーを使用します。

- `%Tokenize` は、スペース文字 (`\s`) に設定されています。これは、Open Parser が入力フィールドをトークンに分割するときにスペース文字で区切ることを意味します。例えば、Abu Mohammed al-Rahim ibn Salamah という名前に含まれるトークンは、Abu、Mohammed、al-Rahim、ibn、Salamah の 5 つです。
- `%InputField` は、**[名前]** フィールドから入力データを取得してパースするように設定されています。
- `%OutputFields` は、パースしたデータを、**[Kunya]**、**[Ism]**、**[Laqab]**、**[Nasab]**、**[Nisba]** の 5 つのフィールドにコピーするように設定されています。
- `<root>` 式には、アラブ系の名前のパターンが以下のように定義されています。
- 0 または 1 つの **Kunya** が含まれる
- 厳密に 1 つまたは 2 つの **Ism** が含まれる
- 0 または 1 つの **Laqab** が含まれる
- 0 または 1 つの **Nasab** が含まれる
- 0 以上の **Nisba** が含まれる

ドメインを定義するルール変数では、必須の `OutputFields` コマンドで定義された出力フィールドと同じ名前を使う必要があります。

パーシング グラマーでは、正規表現と式の数量詞を組み合わせて使い、アラブ系の名前パターンを作成します。パーシング グラマーでは次の特殊な文字を使用します。

- "?" 文字は、その位置に正規表現が 0 回または 1 回出現できることを意味します。
- "*" 文字は、その位置に正規表現が 0 回以上出現できることを意味します。
- ";" 文字は、ルールの終了を意味します。

[コマンド] タブでは、パーシング グラマーで使用できるその他の特殊な記号にマウス ポインターを重ねると、その記号の説明が表示されます。

デフォルトでは、数量詞は Greedy な動作を行います。Greedy な動作とは、式が、マッチを成立させつつ、できる限り多くのトークンを消費しようとすることを意味します。この動作は、'?' を付加することによって Reluctant なマッチングに、'+' を付加することによって Possessive なマッチングに変更することができます。Reluctant なマッチングとは、式が、マッチを成立させつつ、できる限り少なくトークンを消費しようとすることを意味します。Possessive なマッチングとは、マッチの成立が妨げられる場合であっても、できる限り多くのトークンを消費しようとすることを意味します。

パーシング グラマーをテストするには、[プレビュー] タブをクリックします。以下に示す名前を **[名前]** フィールドに入力し、**[プレビュー]** をクリックします。

Name	Kunya	Ism	Laqab	Nasab	Nisba
Abu Karim Muhammad al-Jamil ibn Nidal ibn Abdulaziz al-Filistini	Abu Karim	Muhammad	al-Jamil	ibn Nidal ibn Abdulaziz	al-Filistini
Layla bint Zuhayr ibn Yazid al-Nahdiyah		Layla		bint Zuhayr ibn Yazid	al-Nahdiyah
Yazid ibn Abi Hakim		Yazid		ibn Abi Hakim	
Abu Bishr al-Yaman ibn Abi al-Yaman al-Bandaniji	Abu Bishr	al-Yaman		ibn Abi	al-Yaman al-Bandaniji
Abu al-Tayyib ʿAbd al-Rahim ibn Ahmad al-Harrani	Abu al-Tayyib	ʿAbd	al-Rahim	ibn Ahmad	al-Harrani
Ahmad ibn Sa'id al-Bahili		Ahmad		ibn Sa'id	al-Bahili
Abu al-ʿAbbas Muhammad ibn Ya'qub ibn Yusuf al-Asamm al-Naysaburi	Abu al-ʿAbbas	Muhammad		ibn Ya'qub ibn Yusuf	al-Asamm al-Naysaburi
Abu al-Qasim Mansur ibn al-Zabriqan ibn Salamah al-Namari	Abu al-Qasim	Mansur		ibn al-Zabriqan ibn Salamah	al-Namari
ʿUbayd ibn Mu'awiyah ibn Zayd ibn Thabit ibn al-Dahhak		ʿUbayd		ibn Mu'awiyah ibn Zayd ibn Thabit ibn al-Dahhak	
Umm Ja'far Zubaydah	Umm Ja'far	Zubaydah			

また、他の有効または無効な名前を入力して、入力データがどのようにパースされるのかを確認することもできます。

トレース機能を使うと、最終的なパース結果やパーシング過程をグラフィカルな表示で確認できます。**[トレース]** 列のリンクをクリックして、そのデータ行の [トレース詳細] を表示します。

Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも **[Kunya]**、**[Ism]**、**[Laqab]**、**[Nasab]**、**[Nisba]** の各フィールドが含まれます。

中国の名前の分割

このテンプレートでは、中国系の名前をコンポーネントにパースする方法を示します。パーシング ルールに従って **[名前]** フィールド内の各トークンを分割し、それらを **[LastName]** フィールドと **[FirstName]** フィールドにコピーします。

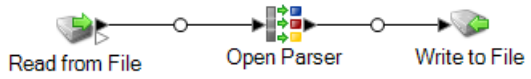
ビジネス シナリオ

ある投資サービス企業では、中国語を母語とする顧客を対象として、さまざまな送付物に中国語のテキストを掲載することを検討しています。

中国系の氏名表記を理解するために、インターネット上で資料を検索したところ、中国系の氏名の構成を解説した以下のサイトが見つかりました。

en.wikipedia.org/wiki/Chinese_names

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ParseChineseNames]** を選択します。このデータフローでは、Data Normalization モジュールが必要です。

このデータフローでは、データをファイルから読み取り、Open Parser ステージで処理します。入力ファイルの各データ行に以下の操作を行います。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前が含まれています。

Open Parser

このステージでは、ドメインエディタで作成されたカルチャー固有ドメイン グラマーを使うか、ドメインに依存しないグラマーを使うかを定義します。ドメインエディタで作成したカルチャー固有のパーシング グラマーは、カルチャーとドメインに関連付けられた、検証済みのパーシング グラマーです。Open Parser で作成したカルチャーに依存しないパーシング グラマーは、カルチャーとドメインに関連付けられていない、検証済みのパーシング グラマーです。

このテンプレートでは、パーシング グラマーはドメインに依存しないグラマーとして定義されています。

Open Parser ステージでは、次のようにコマンドと式が定義されたパーシング グラマーを使用します。

- `%Tokenize` は、None に設定されています。`Tokenize` を `None` に設定する場合、パーシング グラマー ルールは、ルール定義内に空白などのトークン区切り文字を含む必要があります。
- `%InputField` は、**[名前]** フィールドから入力データを取得してパースするように設定されています。

- `%OutputFields` は、パースしたデータを **[LastName]** フィールドと **[FirstName]** フィールドにコピーするように設定されています。

`<root>` 式には、中国系の名前のパターンが以下のように定義されています。

- **LastName** は 1 回含まれる
- **FirstName** は 1 ～ 3 回含まれる

ドメインを定義するルール変数では、必須の `OutputFields` コマンドで定義された出力フィールドと同じ名前を使う必要があります。

`CJKCharacter` ルール変数は、中国/日本/韓国(CJK) の文字パターンを定義します。この文字パターンは、符号や記号ではない文字のみで定義されています。ルールは以下のとおりです。

```
<CJKCharacter> = @RegEx("([\p{InCJKUnifiedIdeographs}&&\p{L}])");
```

- 正規表現 `\p{InX}` は、特定のカルチャーに対応する Unicode ブロックを指定するために使われます。ここで、`X` はカルチャーを表します。この例では、カルチャーは CJKUnifiedIdeographs です。
- 正規表現では、マッチングの対象とする文字のセットを文字クラスと呼びます。例えば、[aeiou] は母音のみを含む文字クラスです。文字クラスは、他の文字クラスに含めたり、結合演算子 (暗黙的) や交差演算子 (&&) を使って構築したりすることができます。結合演算子は、1 つ以上のオペランド クラスに含まれるすべての文字を含むクラスを表します。交差演算子は、交差する Unicode ブロックの重なり合う部分に含まれるすべての文字を含むクラスを表します。
- 正規表現 `\p{L}` は、文字だけを含む Unicode ブロックを示すために使用します。

パーシング グラマーをテストするには、[プレビュー] タブをクリックします。以下に示す名前を **[名前]** フィールドに入力し、**[プレビュー]** をクリックします。

Name	FirstName	LastName
王若琳	若琳	王
刘耕宏	耕宏	刘
许惠欣	惠欣	许
蔡依林	依林	蔡
水沫	沫	水
蕭亞軒	亞軒	蕭
郑元畅	元畅	郑
林依晨	依晨	林

また、他の有効または無効な名前を入力して、入力データがどのようにパースされるのかを確認することもできます。

トレース機能を使うと、最終的なパース結果やパーシング過程をグラフィカルな表示で確認できます。**[トレース]** 列のリンクをクリックして、そのデータ行の [トレース詳細] を表示します。

Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも **[LastName]** フィールドと **[FirstName]** フィールドがあります。マッチ結果リストのマッチ結果を選択し、**[削除]** をクリックします。

スペインとドイツの名前の分割

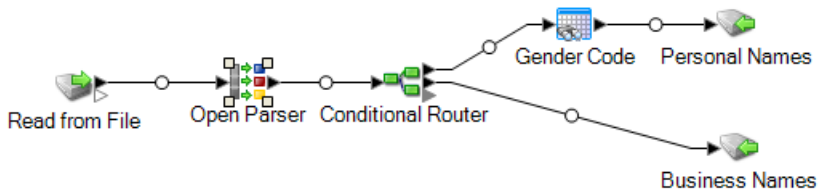
このテンプレートでは、スペイン系とドイツ系の名前など、カルチャーが混在する名前を要素にパースする方法を示します。パーシング ルールを使って **[名前]** フィールドの名前を各トークンに分割し、個人名および企業名パーシング グラマーで定義されたフィールドにコピーします。このパーシング グラマーの詳細については、**[ツール] > [Open Parser ドメインエディタ]** を選択し、**[個人および企業名]** ドメインと **[ドイツ語 (de)]** または **[スペイン語 (es)]** カルチャーのいずれかを選択します。

また、このテンプレートでは、テーブル管理ツールに収められたテーブル データを使って個人名に性別コードを適用します。テーブル管理の詳細については、**[ツール] > [テーブル管理]** を参照してください。

ビジネス シナリオ

ブリュッセルに本社を置く、ある製薬会社は、ドイツとスペインの営業拠点を統合しました。会社では、カルチャーの混合したデータベースに名前データを格納するために、これらの 2 つのカルチャーで使われる名前の派生形を分析する必要があります。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ParseSpanish&GermanNames]** を選択します。このデータフローでは、Data Normalization モジュールが必要です。

このデータフローでは、データをファイルから読み取り、Open Parser ステージで処理します。入力ファイルの各データ行に以下の操作を行います。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前と、各名前の CultureCode 情報が記録されています。CultureCode 情報は、入力する名前のカルチャーが "ドイツ (de)" であるか "スペイン (es)" であるかを示します。

Open Name Parser

Open Name Parser では、名前フィールドをチェックして、Spectrum™ Technology Platform 名前データベース ファイルに格納されている名前データと比較します。この比較結果に基づいて、名前データが [First]、[Middle]、[Last] の各名前フィールドに分割されます。

Conditional Router

このステージでは、入力をチェックして、個人名であれば Gender Codes ステージに渡し、企業名であれば Business Names ステージに渡します。

性別コード

キャンバスでこのステージをダブルクリックしてから、**[変更]** をクリックして Table Lookup ルール オプションを表示します。

[分類] オプションでは、ソースの値をキーとして使用し、対応する値をテーブル エントリから [デスティネーション] リストで選択したフィールドにコピーします。このテンプレートでは、**[フィールド全体]** が選択され、**[ソース]** が **[FirstName]** フィールドを使用するように設定されています。Table Lookup はフィールド全体を 1 つの文字列として扱い、文字列全体を分類できる場合はレコードにフラグを設定します。

[デスティネーション] は **[GenderCode]** フィールドに設定され、**Gender Codes** テーブルに格納された検索語を使って男性名と女性名の分類が行われます。Table Lookup は、入力データに含まれる語が見つからない場合に "不明 (Unknown)" を意味する値 **U** を設定します。この機能をより詳しく理解するには、**[ツール] > [テーブル管理]** を選択し、Gender Codes テーブルを選択してください。

Write to File

このテンプレートには、2 つの Write to File ステージが含まれています。1 つは個人名用、もう 1 つは企業名用です。入力フィールドだけでなく、個人名出力ファイルにも **[Name]**、**[TitleOfRespect]**、**[FirstName]**、**[MiddleName]**、**[LastName]**、**[PaternalLastName]**、**[MaternalLastName]**、**[MaturitySuffix]**、**[GenderCode]**、**[CultureUsed]**、**[ParserScore]** の各フィールドが格納されます。

企業名出力ファイルには、**[Name]**、**[FirmName]**、**[FirmSuffix]**、**[CulureUsed]**、**[ParserScore]** の各フィールドが格納されます。

電子メールアドレスの分割

このテンプレートでは、電子メール アドレスをコンポーネントにパースする方法を示します。パーシング ルールに従って **[電子メール]** フィールド内の各トークンを分割し、それらを **[Local-Part]**、**[DomainName]**、**[DomainExtension]** の 3 つのフィールドにコピーするように設

定されています。**Local-Part** は、電子メール アドレスのドメイン名部分を表し、**DomainName** は電子メール アドレスのドメイン名を表します。**DomainExtension** は、電子メール アドレスのドメイン拡張子を表します。例えば、pb.com の場合、"pb" はドメイン名、"com" はドメイン拡張子です。

オープンなパーシング作業に役立つパブリック ドメイン情報の入手先として、インターネットはとても便利です。この例では、電子メールのフォーマット情報をインターネット上の複数のソースから取得し、テーブル管理ツールにインポートしてドメイン値のテーブルを作成しました。このテンプレートで実行するドメイン拡張子の操作は、この方法が非常に役立つことを具体的に示すものです。

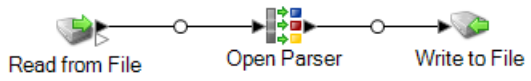
また、このテンプレートを利用すると、テーブル管理ツールにロードしたテーブル データを使って、テーブル検索をパーシング作業の一環として効果的に実行する方法を習得できます。

ビジネス シナリオ

ある保険会社が、電子メールによるマーケティング キャンペーンを初めて実施するために準備しています。顧客の電子メール アドレスを格納したデータベースがあり、ここに収められた電子メール アドレスが有効な SMTP フォーマットになっていることを確認する必要があります。

このデータフローを作成する前に、テーブル管理ツールを使って有効なドメイン名拡張子のテーブルをロードして、検証プロセスの一環としてドメイン名拡張子を検索できるようにする必要があります。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ParseEmail]** を選択します。このデータフローでは、Data Normalization モジュールが必要です。

このデータフローでは、データをファイルから読み取り、Open Parser ステージで処理します。入力ファイルの各データ行に対し、このデータフローで以下の操作が行われます。

ドメイン拡張子テーブルの作成

最初の作業として、テーブル管理ツールを使って、電子メール アドレスに含まれるドメイン拡張子が有効かどうかをチェックするために使う Open Parser テーブルを作成します。

1. **[ツール]** メニューの **[テーブル管理]** を選択します。
2. **[タイプ]** リストで **[Open Parser]** を選択します。
3. **[新規]** をクリックします。

4. **[ユーザ定義テーブルの追加]** ダイアログ ボックスで［テーブル名］フィールドに **"EmailDomains"** と入力し、**[コピー元]** リストで **"None"** が選択されていることを確認してから、**[OK]** をクリックします。
5. **[名前]** リストに **"EmailDomains"** が表示されている状態で、**[インポート]** をクリックします。
6. **[インポート]** ダイアログ ボックスで **[参照]** をクリックし、テーブルのソース ファイルを選択します。デフォルトの場所は次のとおりです。`<drive>:\Program Files\Pitney Bowes\Spectrum\server\modules\coretemplates\data\ Email_Domains.txt`テーブル管理ツールには、インポート ファイルに含まれるテキストがプレビュー表示されます。
7. **[OK]** をクリックします。テーブル管理ツールにソース ファイルがインポートされ、インターネット ドメイン拡張子のリストが表示されます。
8. **[閉じる]** をクリックします。`EmailDomains` テーブルが作成されます。次に、ParseEmail テンプレートを使ってデータフローを作成します。

Read from File

このステージでは、パースする電子メール アドレスが記録されているファイルの名前、格納場所、およびレイアウトを識別します。

Open Parser

Open Parser ステージでは、次のようにコマンドと式が定義されたパーシング グラマーを使用します。

- `%Tokenize` は、None に設定されています。`Tokenize` を `None` に設定する場合、パーシング グラマー ルールは、ルール定義内に空白などのトークン区切り文字を含む必要があります。
- `%InputField` は、**[Email_Address]** フィールドから入力データを取得してパースするように設定されています。
- `%OutputFields` は、パースしたデータを **[Local-Part]**、**[DomainName]**、**[DomainExtension]** の 3 つのフィールドにコピーするように設定されています。
- root 式は、パースするトークンのパターンを定義します。

```
<root> = <Local-Part>"@"<DomainName>"."<DomainExtension>;
```

ドメインを定義するルール変数では、必須の `OutputFields` コマンドで定義された出力フィールドと同じ名前を使う必要があります。

- パーシング グラマーの残りの部分では、各ルール変数を式で定義します。

```
<Local-Part> = (<alphanum> ".")* <alphanum> | (<alphanum> "_")*
<alphanum> ;
<DomainName> = (<alphanum> ".")? <alphanum>;
```

```
<DomainExtension> = @Table("EmailDomains")* "."? @Table("EmailDomains");
<alphanum>=@RegEx("[A-Za-z0-9]+");
```

<Local-Part> 変数は、<alphanum> 変数、ピリオド文字、および他の <alphanum> 変数をを含む文字列として定義されます。

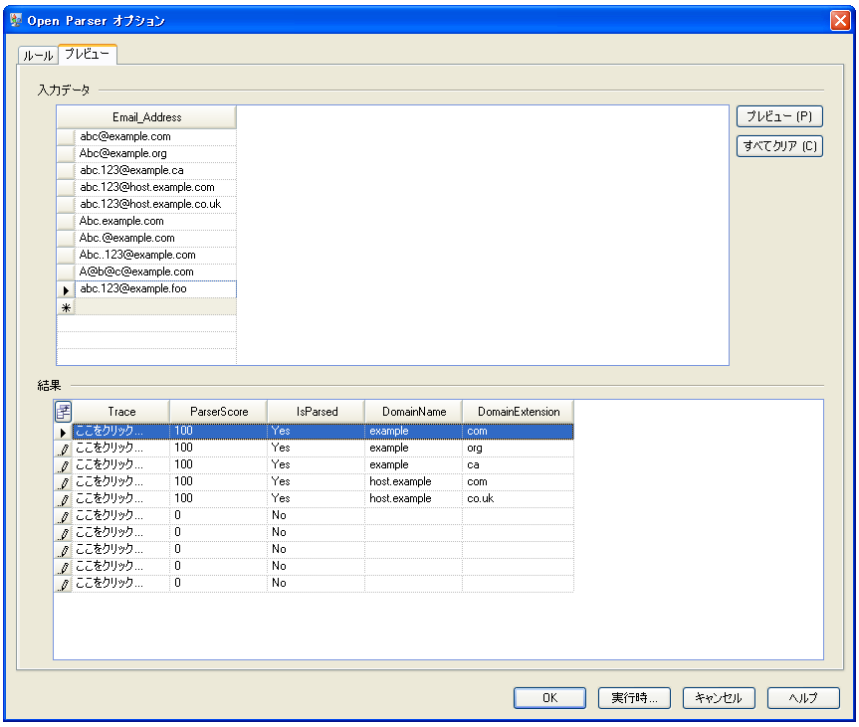
<alphanum> 変数定義は A ～ Z、a ～ z、および 0 ～ 9 の文字からなる文字列を意味する正規表現です。<alphanum> 変数は、このパーシング グラマーの全体を通して使用され、パーシング グラマーの最後の行で定義されます。

パーシング グラマーでは、正規表現とリテラル文字を組み合わせて使い、電子メール アドレスのパターンを作成します。このパーシング グラマーでは、二重引用符で囲まれた文字は、リテラル文字、検索に使うテーブルの名前、または正規表現として扱われます。パーシング グラマーでは次の特殊な文字を使用します。

- "+" 文字は、その位置に正規表現が 1 回以上出現できることを意味します。
- "?" 文字は、その位置に正規表現が 0 回または 1 回出現できることを意味します。
- "|" 文字は、変数に OR 条件が適用されることを意味します。
- ";" 文字は、ルールの終了を意味します。

[コマンド] タブでは、パーシング グラマーで使用できるその他の特殊な記号にマウス ポインターを重ねると、その記号の説明が表示されます。

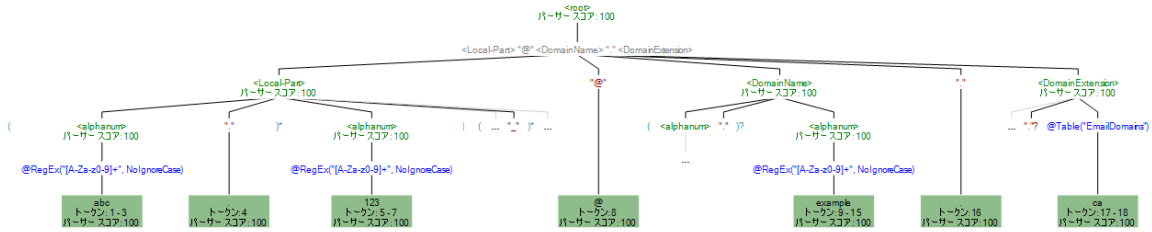
パーシング グラマーをテストするには、[プレビュー] タブをクリックします。以下に示した電子メール アドレスを **[電子メール アドレス]** フィールドに入力してから、**[プレビュー]** をクリックします。



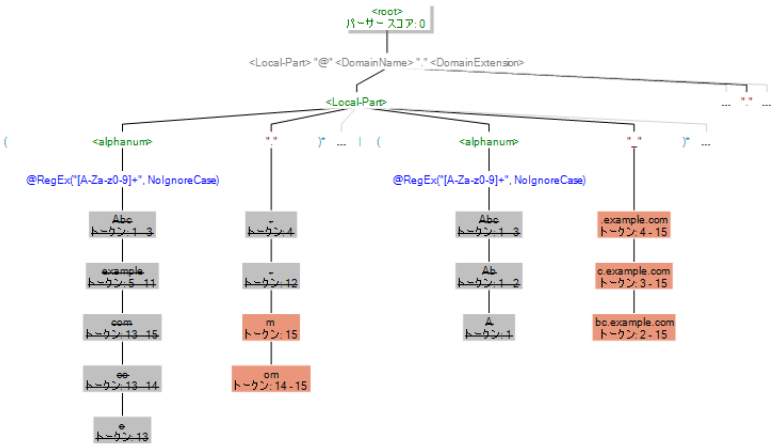
また、他の電子メール アドレスを入力して、入力データがどのようにパースされるのかを確認することもできます。

トレース機能を使うと、最終的なパース結果やパーシング過程をグラフィカルな表示で確認できます。**[トレース]** 列のリンクをクリックして、そのデータ行の [トレース詳細] を表示します。

[トレース詳細] にマッチング結果が表示されます。マッチしたトークンをパーシング グラマーで各式と比較します。



また、トレース機能は、非マッチ結果を参照する目的にも使えます。次の図は、非マッチ結果を示しています。マッチしたトークンをパーシング グラマーで各式と比較します。この入力データ (`Abc.example.com`) がマッチしなかった理由は、マッチすると認識されるために必要なトークンが揃っていないからです。つまり、Local- Part トークンと Domain トークンを区切る @ 文字が欠けています。



Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも **[Local-Part]**、**[DomainName]**、**[DomainExtension]**、**[IsParsed]**、**[ParserScore]** の各フィールドがあります。

米国の電話番号を分割

このテンプレートでは、米国の電話番号をコンポーネントにパースする方法を示します。パーシング ルールに従って **[PhoneNumber]** フィールド内の各トークンを分割し、それらを **[CountryCode]**、**[AreaCode]**、**[Exchange]**、**[Number]** の 4 つのフィールドにコピーします。

ビジネス シナリオ

サービス提供エリアを広げつつある、あるワイヤレス サービス プロバイダは、ユーザからかかってくる電話番号データを分析するプロジェクトを発足しました。

以下のデータフローは、このビジネス シナリオの解決策を示しています。

このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ParseUSPhoneNumbers]** を選択します。このデータフローでは、Data Normalization モジュールが必要です。

このデータフローでは、データをファイルから読み取り、Open Parser ステージで処理します。入力ファイルの各データ行に以下の操作を行います。

Read from File

このステージでは、パースする電話番号が記録されているファイルの名前、格納場所、およびレイアウトを識別します。

Open Parser

このステージでは、ドメインエディタで作成されたカルチャー固有ドメイン グラマーを使うか、ドメインに依存しないグラマーを使うかを定義します。ドメインエディタで作成したカルチャー固有のパーシング グラマーは、カルチャーとドメインに関連付けられた、検証済みのパーシング グラマーです。Open Parser で作成したカルチャーに依存しないパーシング グラマーは、カルチャーとドメインに関連付けられていない、検証済みのパーシング グラマーです。

このテンプレートでは、パーシング グラマーはドメインに依存しないグラマーとして定義されています。

Open Parser ステージでは、次のようにコマンドと式が定義されたパーシング グラマーを使用します。

- `%Tokenize` は、None に設定されています。`Tokenize` を `None` に設定する場合、パーシング グラマー ルールは、ルール定義内に空白などのトークン区切り文字を含む必要があります。
- `%InputField` は、**[PhoneNumber]** フィールドから入力データを取得してパースするように設定されています。
- `%OutputFields` は、パースしたデータを **[CountryCode]**、**[AreaCode]**、**[Exchange]**、**[Number]** の 4 つのフィールドにコピーするように設定されています。
- `<root>` 式では、パースするトークンのパターンが定義され、OR 文 (|) が使用されます。以下のような電話番号が有効です。
- **[CountryCode]**、**[AreaCode]**、**[Exchange]**、および **[Number]** または
- **[AreaCode]**、**[Exchange]**、および **[Number]** または
- **[Exchange]** および **[Number]**

パーシング グラマーでは、正規表現とリテラル文字を組み合わせて使い、電話番号のパターンを作成します。このパーシング グラマーでは、二重引用符に囲まれた文字は、リテラル文字列または正規表現として扱われます。

`<root>` コマンドで使われるプラス記号 (+) は、引用符で囲まれているのでリテラル文字として定義されます。一重引用符または二重引用符は、リテラル文字を示すために使用できます。プラス記号を引用符で囲まずに使うと、その前にある式が 1 回以上繰り返して出現できるという意味になります。

電話番号ドメイン ルールは、以下の文字パターンにマッチするように定義されています。

- "+" 文字の 0 回または 1 回の出現。
- **CountryCode** ルール。これは 0 ～ 9 の 1 文字を意味します。

- 左丸括弧、ハイフン、またはスペース文字の 0 回または 1 回の出現。これらの文字のいずれかが 2 つ並んで出現すると、非マッチとなります。つまり、無効な電話番号と判定されます。
- **AreaCode** ルール。これは 0 ～ 9 の数字が 3 つ並ぶことを意味します。
- 左丸括弧、ハイフン、またはスペース文字の 0 回または 1 回の出現。これらの文字のいずれかが 2 つ並んで出現すると、非マッチとなります。つまり、無効な電話番号と判定されます。
- **Exchange** ルール。これは 0 ～ 9 の数字が 3 つ並ぶことを意味します。
- 左丸括弧、ハイフン、またはスペース文字の 0 回または 1 回の出現。これらの文字のいずれかが 2 つ並んで出現すると、非マッチとなります。つまり、無効な電話番号と判定されます。
- **Number** ルール。これは 0 ～ 9 の数字が 4 つ並ぶことを意味します。

ドメインを定義するルール変数では、必須の `OutputFields` コマンドで定義された出力フィールドと同じ名前を使う必要があります。

正規表現と式の数量詞

パーシング グラマーでは、正規表現と式の数量詞を組み合わせて使い、米国の電話番号のパターンを作成します。電話番号を分割パーシング グラマーでは次の特殊な文字を使用します。

- "?" 文字は、その位置に正規表現が 0 回または 1 回出現できることを意味します。
- (|) 文字は、OR 条件を示します。
- ";" 文字は、ルールの終了を意味します。

[コマンド] タブでは、パーシング グラマーで使用できるその他の特殊な記号にマウス ポインターを重ねると、その記号の説明が表示されます。

[プレビュー] タブの使用

パーシング グラマーをテストするには、[プレビュー] タブをクリックします。以下に示す電話番号を **[PhoneNumber]** フィールドに入力し、**[プレビュー]** をクリックします。

PhoneNumber	CountryCode	AreaCode	Exchange	Number
1(410)286-7334	1	410	286	7334
14042867534	1	404	286	7534
(410)286-7256		410	286	7256
301-868-9999		301	868	9999
1-222-458-7799	1	222	458	7799
+1(410)286-7334	1	410	286	7334
901 888 9990		901	888	9990
1 410 888 2345	1	410	888	2345
234-4567			234	4567
234 6789			234	6789

また、他の有効または無効な電話番号を入力して、入力データがどのようにパースされるのかを確認することもできます。

トレース機能を使うと、最終的なパース結果やパーシング過程をグラフィカルな表示で確認できます。**[トレース]** 列のリンクをクリックして、そのデータ行の [トレース詳細] を表示します。

Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも **[CountryCode]**、**[AreaCode]**、**[Exchange]**、**[Number]** の各フィールドが含まれます。

3 - 正規化

このセクションの構成

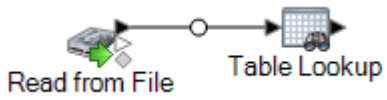
語の正規化

用語の使用に一貫性がないと、データ品質の問題が生じ、パーシングや検索などが困難になることがあります。使用に一貫性がない語をデータ内から探して正規化するデータフローを作成できます。例えば、データの企業名に "Incorporated"、"Inc."、"Inc" という語が含まれている場合、1 つの形式 (例えば "Inc.") に正規化するデータフローを作成できます。

> **注：** この手順を実行する前に、データに適用する正規化された語を格納した Data Normalization モジュール データベースを管理者がインストールする必要があります。データベースのインストール手順については、『*インストール ガイド*』を参照してください。

1. Enterprise Designer で、新しいデータフローを作成します。
2. ソース ステージをキャンバスにドラッグします。
3. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
4. Table Lookup ステージをキャンバス上にドラッグし、ソース ステージに接続します。

   例えば、Read from File ソース ステージを使用する場合、データフローは次のようになります。

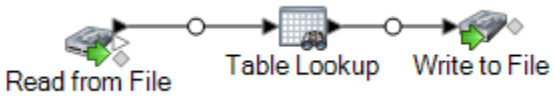


5. キャンバス上の Table Lookup ステージをダブルクリックします。
6. Table Lookup のオプションを指定するには、ルールを作成します。複数のルールを作成して、ルールを適用する順序を指定することができます。**[追加]** をクリックして、ルールを作成します。
7. **[アクション]** フィールドで、デフォルトの **[正規化]** オプションをオンのままにします。
8. フィールド全体が正規化したい語の場合は、**[オン]** フィールドで **[フィールド全体]** が選択された状態のままにします。フィールド内の個々の語を正規化したい場合は、**[フィールド内の個々の語]** を選択します。
9. **[ソース]** フィールドで、正規化するフィールドを選択します。
10. **[デスティネーション]** フィールドで、正規化された語を格納するフィールドを選択します。同じフィールドをソース フィールドとして指定した場合、ソース フィールドの値が正規化された語で置き換えられます。
11. **[テーブル]** フィールドで、正規化された語を格納するテーブルを選択します。

> **注：** 必要なテーブルが表示されていない場合は、システム管理者に連絡してください。Data Normalization モジュール データベースをロードする必要があります。

12. **[テーブル エントリが見つからなかった場合、デスティネーションの値として次の値を設定する]** フィールドで、**[ソースの値]** を選択します。
13. **[OK]** をクリックします。
14. その他のフィールドの値を正規化する場合は、追加のルールを定義します。ルールの定義が完了したら、**[OK]** をクリックします。
15. シンク ステージをキャンバス上にドラッグし、Table Lookup ステージに接続します。

    例えば、Write to File シンク ステージを使用する場合、データフローは次のようになります。



16. シンク ステージをダブルクリックして設定します。

    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

これでデータフローが語を正規化するようになりました。

個人名の正規化

この手順では、個人名データ (例: "John P.Smith") を受け取り、同じ名前の共通ニックネームを識別して、冗長レコードの統合に使用できる名前の正規化バージョンを作成するデータフローの作成方法を示します。

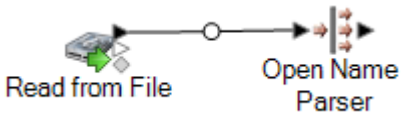
> **注：** 始める前に、入力データに個人のフル ネームが含まれる "Name" という名前のフィールドがあることを確認してください。

1. 以下のテーブルをまだ Spectrum™ Technology Platform サーバーにロードしていない場合は、ロードします。
   - Open Parser Base
   - Open Parser Enhanced Names

   Data Normalization モジュールのデータベース ロード ユーティリティを使用して、これらのテーブルをロードします。テーブルのロード手順については、『*インストール ガイド*』を参照してください。

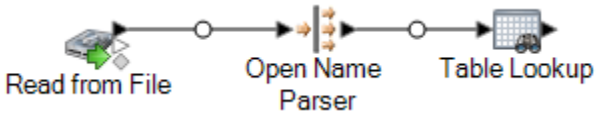
2. Enterprise Designer で、新しいデータフローを作成します。
3. ソース ステージをキャンバスにドラッグします。
4. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
5. Open Name Parser ステージをキャンバス上にドラッグし、ソース ステージに接続します。

   例えば、Read from File ステージを使用する場合、データフローは次のようになります。



6. Table Lookup ステージをキャンバスにドラッグし、Open Name Parser 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。

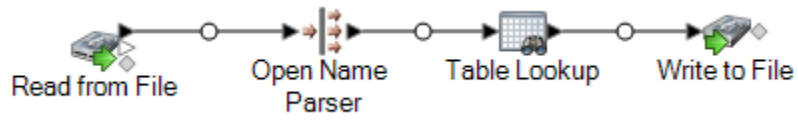


7. キャンバス上の Table Lookup ステージをダブルクリックします。
8. **[ソース]** フィールドで **FirstName** を選択します。
9. **[デスティネーション]** フィールドで **FirstName** を選択します。

   ソースとデスティネーションの両方に同じフィールドを指定することにより、フィールドは名前の正規化されたバージョンで更新されます。

10. **[テーブル]** フィールドで **NickNames.xml** を選択します。
11. **[OK]** をクリックします。
12. **[OK]** を再度クリックし、**[Table Lookup オプション]** ウィンドウを閉じます。
13. シンク ステージをキャンバスにドラッグし、Table Lookup ステージに接続します。

    例えば、Write to File シンクを使用した場合、データフローは次のようになります。



14. シンク ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。

個人名を受け取り、名を正規化して、ニックネームを名前の標準バージョンで置き換えるデータフローができました。

正規化用テンプレート

個人名の形式化

このデータフロー テンプレートは、個人名 (例えば "John P. Smith") を受け取り、同じ名前の共通のニックネームを識別し、その名前の正規化バージョンを作成する方法を示します。正規化バージョンは、冗長なレコードの整理統合に利用できます。また、性別データに基づいて敬称データを追加する方法も示します。

ビジネス シナリオ

ある NPO が、ガラ イベントの招待状を発送しようとしています。入力データには名前がフルネームで記録されていますが、これを [First]、[Middle]、[Last] の各名前フィールドにパースし、さらに [Title of Respect] フィールドを追加して招待状をフォーマルな体裁にすることを考えています。また、名前データにニックネームがある場合は、それをもっとフォーマルな表記の名前に置き換える必要があります。

以下のデータフローは、このビジネス シナリオの解決策を示しています。

このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[StandardizePersonalNames]** を選択します。このデータフローでは、Data Normalization モジュールと Universal Name モジュールが必要です。

入力ファイルの各データ行に以下の操作を行います。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前が含まれています。

Name Parser

このテンプレートでは、Name Parser ステージを Parse Personal Name という名前で呼びます。Parse Personal Name ステージでは、名前フィールドがチェックされ、Spectrum™ Technology Platform 名前データベース ファイルに格納されている名前データと比較されます。この比較結果に基づいて、名前データが [First]、[Middle]、[Last] の各名前フィールドに分割され、エンティティ タイプと性別がすべての名前に割り当てられます。また、名前データに加えて、パターン認識も使用されます。

このテンプレートでは、Parse Personal Name ステージは次のように設定されています。

- [個人名をパース] が選択され、[企業名をパース] が選択されていません。このようにオプションを設定すると、名が評価されて性別、順序、および句読文字が判別されますが、企業名の評価は実行されません。
- [性別判定ソース] はデフォルトに設定されています。たいていのケースで、デフォルトは性別の判定に最適な設定であり、さまざまな名前に対応できます。ただし、特定のカルチャーに属する名前を処理する場合は、そのカルチャーを選択します。特定のカルチャーを選択すると、名前の性別が適切に判定される確率が高くなります。例えば、デフォルトのままで設定した場合、Jean という名前は女性と判定されます。しかし、[フランス系] を選択した場合は、男性と判定されます。
- 順序は [正順序] に設定されています。名前フィールドの順序は、[敬称]、[名]、[ミドル ネーム]、[姓]、[接尾語] となります。
- [ピリオドを残す] は選択されていません。名前データに含まれる句読文字は残されません。

Transformer

このテンプレートでは、Transformer ステージを Assign Titles と呼びます。Assign Titles ステージでは、カスタム スクリプトを使って、Parse Personal Name ステージからのデータ ストリーム出力中に各行を検索し、**GenderCode** 値に基づいて **TitleOfRespect** 値を割り当てます。

使用するカスタム スクリプトを以下に示します。

```
if (row.get('TitleOfRespect') == '')
{
 if (row.get('GenderCode') == 'M')
  row.set('TitleOfRespect', 'Mr')
 if (row.get('GenderCode') == 'F')
  row.set('TitleOfRespect', 'Ms')
```

Assign Titles ステージでは、`GenderCode` フィールドで **M** を検出するたびに **TitleOfRespect** の値を `Mr` に設定します。Assign Titles ステージでは、`GenderCode` フィールドで **F** を検出するたびに **TitleOfRespect** の値を `Ms` に設定します。

正規化

このテンプレートでは、正規化ステージを Standardize Nicknames と呼びます。Standardize Nicknames ステージでは、Nicknames.xml データベースに名を検索し、その名前にフォーマルな表記があればそれでニックネームを置き換えます。例えば、Tommy という名前は Thomas で置き換えます。

Write to File

このテンプレートには、1 つの Write to File が含まれます。入力フィールドだけでなく、出力ファイルにも [TitleOfRespect]、[FirstName]、[MiddleName]、[LastName]、[EntityType]、[GenderCode]、および [GenderDeterminationSource] の各フィールドがあります。

4 - マッチング

このセクションの構成

マッチングの用語

平均スコア	すべての重複の平均マッチ スコア。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。
ベースライン	マッチした他の結果と比較するために選択したマッチ結果。
候補グループ	Candidate Finder が割り当てた ID でグループ化されたサスペクトまたは候補のレコード。サスペクト (そのグループの最初のレコード) は入力ソースから読み込まれたレコードですが、候補は通常、SQL クエリを使用してデータベースから取り出されたレコードです。
候補レコード	マッチ グループまたは候補グループ内にあるサスペクト以外のすべてのレコード。
ドロップ	重複の減少。
詳細マッチ レコード	マッチ ステージによって処理されたレコードに合致した 1 つのレコードのこと。各レコードは、そのマッチ グループまたは候補グループと出力コレクションの情報と共に、そのレコードがサスペクト、ユニーク、または重複のいずれであったかという情報を提供します。候補レコードは、なぜその入力レコードがサスペクトにマッチしたか、またはマッチしなかったかという情報を提供します。
重複コレクション	サスペクトとその重複レコードがコレクション番号によってグループ化されたもの。ユニーク レコードは常にコレクション番号 0 とされます。
重複レコード	マッチ グループ内で他のレコードとマッチしたレコード。サスペクトまたは候補となります。
Express マッチ	サスペクトと候補が指定されたフィールド内の内容に正確にマッチした場合に作成されるもののこと。通常は ExpressMatchKey が Match Key Generator によって提供されます。式マッチがそれ以上の処理がされず終了した場合、サスペクトと候補は重複していると判断できます。
入力レコード	マッチング ソート実行前のマッチング ステージにおけるレコードの順序。
Interflow Match	2 つの入力レコード ストリーム間で類似データ レコードのマッチを検出するマッチング ステージ。最初のレコード ストリームはサスペクト レコードのソースで、2 番目のストリームは候補レコードのソースです。
Intraflow Match	1 つの入力ストリーム内の類似データ レコードのマッチを検出するマッチング ステージ。
リフト	重複の増加。
マッチ グループ	(グループ化方法) マッチ キーまたはスライディング ウィンドウでグループ化されたレコード。

マッチ結果	(またはリソース バンドル) ステージで生成されたファイルの論理的グループ。このデータはステージの実行毎に保持され、ディスクに保存されます。次の実行では、前回の結果の上書きまたは変更はしません。MAT では、設定情報と同様にサマリと詳細結果に関する情報を提供するために使用されます。
マッチ結果リスト	MAT が現在の分析セッションで分析できる、1 種類のマッチ結果のリスト。
マッチ結果タイプ	マッチ結果の中身を示します。MAT はマッチ結果タイプを使用して、データの使用方法を決定します。
マッチャー ステージ	マッチング ルーチンを実行するキャンバス上のステージ。マッチャー ステージは Interflow Match、Intraflow Match、および Transactional Match です。
マッチでなくなったもの	以前は重複またはサスペクトとされていたが、現在はユニークとなっているレコード。
新しいマッチ	以前はユニークだったが、現在は重複またはサスペクトとされているレコード。
スライディング ウィンドウ	スライディング ウィンドウ マッチング方法は、ウィンドウというあらかじめ定められたバッファ サイズを、対応するデータ行で連続して埋めていきます。各行はウィンドウに追加され、既にウィンドウ内にあるアイテムと比較されます。
サスペクト レコード	マッチ グループまたは候補グループ内の候補と照合されるドライバ レコード。
Transactional Match	Candidate Finder または外部アプリケーションが返す候補レコードとサスペクト レコードを照合するマッチング ステージ。
ユニーク レコード	マッチ グループで他のレコードにマッチしないサスペクトまたは候補レコード。マッチ グループ内に 1 つしか存在していないレコードであれば、サスペクトは自動的にユニーク レコードとなります。

マッチ キーの定義に関するテクニック

効果的であると同時に実用的でもあるマッチングを行うには、精度とパフォーマンスのバランスを取る必要があります。マッチングに最高の精度を求めるなら、レコードを他のすべてのレコードと逐一付き合わせる必要がありますが、処理するレコードが多いと耐え難いほどのパフォーマンス低下を招くため、実用的ではありません。マッチング処理に関わるレコードの数を制限するため、マッチングする可能性が高いレコードだけを比較の対象とするのが賢明です。そうするには、マッチ キーを使用します。マッチ キーとは、ユーザが指定したアルゴリズムで各レコードに

対して生成される値です。レコードから値がアルゴリズムに渡され、マッチ キー値が生成されます。この値は、レコードに新しいフィールドとして保存されます。

例えば、次のような入力レコードがあり、

名 - Fred
姓 - Mertz
郵便番号 - 21114-1687
性別コード - M

次のようなレコードのデータを組み合わせてマッチ キーを生成するマッチ キー ルールを定義したとします。

入力フィールド	開始位置	長さ
郵便番号	1	5
郵便番号	7	4
姓	1	5
名	1	5
性別コード	1	1

次のようなキーになります。

211141687MertzFredM

マッチ キーが同じレコードは、同じマッチ グループにまとめられます。マッチング処理では、さらにグループ内のレコードが比較され、マッチングが判定されます。

Interflow Match または Intraflow Match を使ってレコードをマッチングする場合は、マッチ キーを生成するために、Match Key Generator ステージを使用します。Transactional Match を使ってレコードをマッチングする場合は、Candidate Finder ステージを使ってマッチ グループを作成します。

> **注：** 以下に説明するガイドラインは、Match Key Generator キーと Candidate Finder クエリの両方に当てはまります。Candidate Finder では、このガイドラインは SELECT 文の定義方法に対して適用してください。

マッチ グループのサイズとパフォーマンス

マッチ キーによってマッチ グループのサイズが決定されます。これは、データフローのパフォーマンスが決定されるということでもあります。マッチ グループのサイズが倍になると、実行時間も倍に伸びます。例えば、マッチングの可能性があるレコードをグループに 20 個含めるマッチ キーを定義した場合、10 個しかレコードを含めないマッチ キーを使う場合と比較して、処理に倍の時間がかかります。マッチ キー ルールを引き締めると、マッチ グループに含まれるレコードが少なくなり、マッチングするレコードを除外してしまう恐れが大きくなります。マッチ キー ルールを緩めると、マッチングするレコードがグループから除外される可能性は低くなりますが、グループのサイズは大きくなります。データに適したバランスを取るには、実際に処理するデータとよく似たデータを使ってさまざまなマッチ キー ルールをテストする必要があります。

密度

マッチ キーを設計する際に重要なのは、データの密度を考慮することです。密度とは、データがマッチ グループ間に分散する度合いを意味します。パフォーマンスは、実行しなければならない比較の回数で決まるので、小数の大きなマッチ グループを生成するマッチ キーは、大量の小さなマッチ グループを生成するマッチ キーと比べてパフォーマンスを低下させます。

この因果関係を具体的に理解するために、マッチングしたい名前と住所のレコードが 100 万件あると想定します。マッチ キーとして、郵便番号の先頭 3 バイトと姓の頭文字を使うと仮定します。レコードが全米から集められた場合は、このマッチ キーは十分な数のマッチ グループを生成し、実用に耐えるパフォーマンスを示すと予想できます。しかし、レコードがすべてニューヨーク州のものだとしたらどうでしょうか。郵便番号はどれも "100" で始まるので、最大でもマッチ グループは 26 個しか生成されません。マッチ グループは大きくなり、平均で約 38,000 のレコードが含まれる計算になります。

マッチ グループごとに発生する比較の最大回数は、以下の数式で求められます。

```
N * (N-1) / 2
```

ここで `N` は、マッチ グループに含まれるレコードの数です。

つまり、26 個のマッチ グループのそれぞれに 38,000 のレコードがあると、実行される比較の最大回数は約 187 億回に達します。計算の過程はこのようになります。

最初に、マッチ グループごとの比較の最大回数を計算します。

```
38,000 * (38,000-1) / 2 = 721,981,000
```

次に、この計算結果にマッチ グループの数を掛けます。

```
721,981,000 * 26 = 18,771,506,000
```

仮に、郵便番号の先頭 3 バイトに 100 とおりの値があるとしたら、生成されるマッチ グループは 2,600 個になり、それぞれに含まれるレコード数は平均 380 になるでしょう。この場合、比較の最大回数は 1 億 8,700 万回で、100 分の 1 に減ります。レコードがすべてニューヨーク州のデータであれば、郵便番号の先頭 4 バイト、ないしは 5 バイトをマッチ キーに使うことを検討するの

が良いでしょう。生成されるマッチ グループの数が増え、比較の総数が減ります。若干のマッチング漏れが生じるでしょうが、それと引き換えに実行時間が大幅に短縮されます。

実際には、この例で使用したようなマッチ キーは、データに偏りがあるため、等しいサイズでマッチ グループを生成しません。例えば、姓が "S" で始まる人は、"X" で始まる人より多いでしょう。このような事情から、最も大きなマッチ グループをなるべく小さくすることに気を配る必要があります。レコードが 100,000 個あるマッチ グループは、レコードが 10,000 個のマッチ グループの 10 倍の大きさですが、比較の回数は 100 倍であり、時間も 100 倍かかります。例えば、郵便番号の 5 バイトと AddressLine1 フィールドの 6 バイトをマッチ キーに使うとします。最初の印象では、かなり上等なマッチ キーが得られそうです。問題は、私書箱の住所です。ほとんどのマッチ グループは実用的な大きさに収まりますが、10002PO BOX のようなキーで若干の非常に大きなマッチ グループが生成されます。このような大きなマッチ グループを分割するには、マッチ キーを修正して、私書箱番号の先頭 2 桁を含めます。

マッチ キーをマッチ ルールに合わせる

最高精度の結果を得るには、使用するマッチ ルールと相性の良いマッチ キーを設計する必要があります。そうするには、マッチ ルールをどう定義するか十分に検討することが求められます。

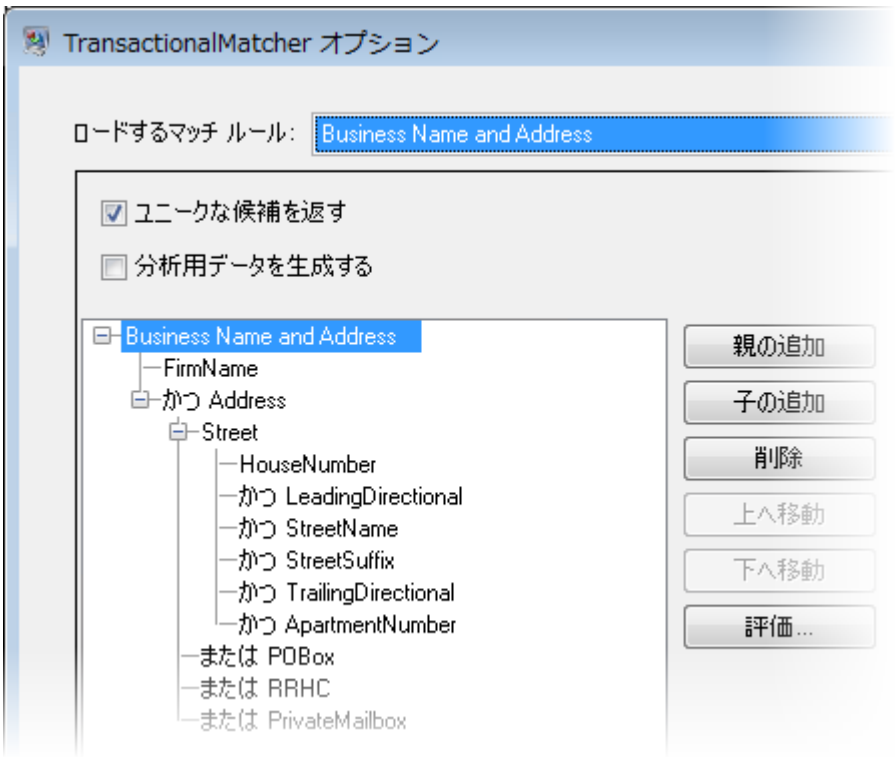
- マッチ キーには、マッチ ルールで正確なマッチングを得るために必要なフィールドが含まれる必要があります。
- マッチ キーでは、マッチ ルールで使用されるものと同じ種類のアルゴリズムを使用します。例えば、発音に基づくアルゴリズムを使うマッチ ルールと組み合わせるマッチ キーであれば、発音に基づくアルゴリズムを使うように設計します。
- マッチ キーの作成には、マッチ ルールで使われるすべてのフィールドの値を使います。
- マッチ キーで使われる 1 つ以上のフィールドにデータの欠落があった場合、マッチ キーにどのような影響が現れるか考慮してください。例えば、ミドルネームの頭文字をマッチ キーの一部に使用し、データに John A.Smith のレコードと John Smith のレコードが含まれるとします。マッチ ルールを設定して、ミドルネームの頭文字フィールドに値がなければ無視することにしました。こうすると、先ほどの 2 つのレコードはマッチ ルールによって一致とみなされます。ただし、マッチ キーはミドルネームの頭文字を使うので、2 つのレコードは別のマッチ グループに分かれてしまい、互いに比較されません。そのため、マッチ ルールの意図した結果になりません。

マッチ ルール

各マッチング ステージ (Interflow Match、Intraflow Match、および Transactional Match) では、マッチ ルールを設定する必要があります。マッチ ルールは、あるレコードが他のレコードと一致

するかどうかを判断する条件を定義します。マッチ ルールでは、比較するフィールド、フィールドの比較方法、および複雑なマッチング ルール向けの比較の階層を指定します。

比較の階層を作成すると、ネストした Boolean マッチ ルールを作成できます。例えば、以下のマッチ ルールがあるとします。



この例では、マッチ ルールは、企業名と住所に基づいてレコードを照合します。マッチ ルールの最初の要素は [FirmName] フィールドです。この要素は、レコードどうしが一致するには [FirmName] フィールドの値が一致する必要があることを意味します。2 番目の要素は住所を評価します。この要素の前に論理演算子 "and" があり、レコードどうしが一致するには [FirmName] と [Address] の両方が一致する必要があることを意味します。マッチ ルールの [Address] 部分は、ストリート、PO Box、地方配送路/幹線請負契約 (RRHC)、および私書箱の 4 種類の住所を評価する子ルールで構成されます。子 [Street] は、データフロー フィールドの [HouseNumber]、[LeadingDirectional]、[StreetName]、[StreetSuffix]、[TrailingDirectional]、および [ApartmentNumber] を調べます。これらがすべて一致すると、親ルール "Street" とその親ルール "Address" はすべて "true" と評価されます。"Street" ルールが "true" と評価されない場合は、[POBox] フィールド、[RRHC]、[PrivateMailbox] フィールドの順に評価されます。これら 3 つのいずれかが一致すれば、親 Address 要素は一致します。

マッチ ルールの作成

マッチ ルールを Interflow Match、Intraflow Match、および Transactional Match で使用して、1 つのレコードが別のレコードとマッチするかどうかを判断する基準を定義します。マッチ ルールでは、比較するフィールド、フィールドの比較方法、および複雑なマッチング ルール向けの比較の階層を指定します。

マッチ ルールは、Interflow Match、Intraflow Match、および Transactional Match で作成できます。Enterprise Designer のマッチ ルール管理ツールでもマッチ ルールを作成できます。マッチ ルール管理ツールでルールを作成すると、あらゆるデータフローで、そして他のユーザがそのルールを使用できるようになります。マッチャー ステージのいずれかでマッチ ルールを作成すると、そのルールはそのステージでのみ使用可能になります。ただし、**[保存]** ボタンをクリックしてルールを保存すれば、他のステージおよびユーザがそのルールを使用できるようになります。

1. Enterprise Designer を開きます。
2. 以下のいずれかの方法を実行します。
   - Interflow Match、Intraflow Match、または Transactional Match でマッチ ルールを定義する場合は、マッチ ルールを定義するマッチ ステージをダブルクリックします。**[ロードするマッチ ルール]** フィールドで、定義済みのマッチ ルールを作成の出発点として選択します。空のマッチ ルールで開始する場合は、**[新規]** をクリックします。
   - マッチ ルール管理ツールでマッチ ルールを定義する場合は、**[ツール] > [マッチ ルール管理]** の順に選択します。既存のルールを独自ルールの作成の出発点として使用する場合は、**[コピー元]** ボックスをオンにして、出発点として使用するルールを選択します。
3. マッチ ルールで使用するデータフロー フィールドと、マッチ ルール階層を指定します。
   a) **[親の追加]** をクリックします。
   b) 親の名前を入力します。名前は一意であり、フィールドであってはなりません。階層の最初の親は、**[ロードするマッチ ルール]** フィールドのマッチ ルール名として使用されます。作成したすべてのカスタム マッチ ルールと変更した定義済みルールは、名前の前に "Custom" という語が付加されて保存されます。
   c) **[子の追加]** をクリックします。ルール階層内にドロップダウン メニューが表示されます。親に追加するフィールドを選択します。

      注 : 親の下のすべての子は、同じ論理演算子を使用する必要があります。フィールド間で異なる論理演算子を使用する場合は、まず中間の親を作成する必要があります。

   d) この操作を繰り返してマッチング階層を完成させます。
4. 親オプションを定義します。親ノードを選択すると、ルール階層の右側に親オプションが表示されます。

a) **[真でない場合に一致]** をクリックして、親の論理演算子を AND から AND NOT に変更します。このオプションをオンにすると、この親で定義されているロジックに一致しないレコードのみがマッチします。

> **注：[真でない場合に一致]** オプションをオンにすると、**[マッチング方法]** オプションが無効になります。詳細については、否定のマッチ条件（81ページ）を参照してください。

b) **[マッチング方法]** フィールドで、親がマッチするかどうかを判断する方法を指定します。次のいずれかです。

すべて真	すべての子がマッチすると確認された場合にマッチすると見なします。この方法を選択すると、子の間に "AND" コネクタが作成されます。
いずれかが真	少なくとも 1 つの子がマッチすると確認された場合にマッチすると見なします。この方法を選択すると、子の間に "OR" コネクタが作成されます。
しきい値で判断	親のスコアがそのしきい値に等しいか、それよりも大きい場合にマッチすると見なします。このオプションを選択すると、**[しきい値]** スライダーが表示されます。このスライダーを使ってしきい値を指定します。スコアリング方法によって、使用する論理コネクタが決定されます。親のしきい値を子のしきい値より高くすることはできません。

> **注：** ここで設定したしきい値は、実行時に [データフロー オプション] ダイアログ ボックスを使って変更できます。**[編集] > [データフロー オプション]** に移動し、**[追加]**をクリックします。ステージを展開し、**[最上位しきい値]** をクリックして、**[デフォルト値]** フィールドにしきい値を入力します。

c) **[欠落データ]** フィールドで、フィールド内の空白データのスコアの計算方法を指定します。次のいずれかです。

空白を無視	フィールドに空白データが入っている場合、そのフィールドを無視します。
0 としてカウント	フィールドに空白データが入っている場合、そのフィールドのスコアを 0 とします。
100 としてカウント	フィールドに空白データが入っている場合、そのフィールドのスコアを 100 とします。
空白を比較	サスペクトと候補のフィールドのスコアを、両方のフィールドに空白データが含まれる場合は 100、それ以外の場合は 0 とします。

d) **[スコアリング方法]** フィールドで、マッチング スコアの決定に使用する方法を選択します。次のいずれかです。

加重平均	各子の重みを使用して、平均マッチ スコアを決定します。
平均	各子の平均スコアを使用して、親のスコアを決定します。
最大値	子の最も高いスコアを使用して、親のスコアを決定します。
最小値	子の最も低いスコアを使用して、親のスコアを決定します。
ベクトル和	それぞれの子のベクトル和を使用して、親のスコアを決定します。計算式は次のとおりです。 sqrt(a^2 + b^2 + c^2) / sqrt(n)。ここで、a、b、および c は 3 つの子のスコアであり、n は子の数です。

次の表に、マッチング方法とスコアリング方法の論理的な関係、およびそれぞれの組み合わせによってマッチング処理時に使用されるロジックがどのように変わるかを示します。

表 1 : マッチング方法とスコアリング方法の表

スコアリング方法	マッチング方法			コメント
	いずれかが真	すべて真	しきい値で判断	
加重平均	不可	AND	AND	マッチング方法として [すべて真] または [しきい値で判断] が選択されている場合にのみ使用可能です。
平均	不可	AND	AND	
ベクトル和	不可	AND	AND	
最大値	OR	不可	OR	マッチング方法として [いずれかが真] または [しきい値で判断] が選択されている場合にのみ使用可能です。
最小値	OR	不可	OR	

5. 子オプションを定義します。子を選択すると、ルール階層の右側に子オプションが表示されます。
   a) 選択された子レコード フィールドを入力ファイル内のフィールドにマッピングするには、**[候補フィールド]** オプションをオンにします。
   b) 2 つのレコードの間で異なるフィールドを互いにマッチングするには、**[クロスマッチ対象]** オプションをオンにして、ドロップダウン リストから 1 つ以上の項目を選択します。マッ

チ ルール管理ツールを使用してマッチ ルールを作成または編集する場合は、ドロップダウン リストはなく、各フィールド名をカンマで区切って入力する必要があります。

c) **[真でない場合に一致]** をクリックして、論理演算子を AND から NOT に変更します。このオプションをオンにすると、レコードがこの子で定義されているロジックに一致しない場合のみ、マッチ ルールが真として評価されます。

例えば、複数のアカウントに関連付けられている個人を特定したい場合、名前は一致するがアカウント番号は一致しないマッチ ルールを作成することができます。アカウント番号をマッチングする子に対して **[真でない場合に一致]** オプションを使用します。

d) **[欠落データ]** フィールドで、フィールド内の空白データのスコアの計算方法を指定します。次のいずれかです。

空白を無視	フィールドに空白データが入っている場合、そのフィールドを無視します。
0 としてカウント	フィールドに空白データが入っている場合、そのフィールドのスコアを 0 とします。
100 としてカウント	フィールドに空白データが入っている場合、そのフィールドのスコアを 100 とします。
空白を比較	サスペクトと候補のフィールドのスコアを、両方のフィールドに空白データが含まれる場合は 100、それ以外の場合は 0 とします。

e) **[しきい値]** フィールドで、マッチと判断されるために個々のフィールド レベルで満たす必要があるしきい値を指定します。

f) **[スコアリング方法]** フィールドで、マッチング スコアの決定に使用する方法を選択します。次のいずれかです。

加重平均	各アルゴリズムの重みを使用して、平均マッチ スコアを決定します。
平均	各アルゴリズムの平均スコアを使用して、マッチ スコアを決定します。
最大値	最も高いアルゴリズムのスコアを使用して、マッチ スコアを決定します。
最小値	最も低いアルゴリズムのスコアを使用して、マッチ スコアを決定します。
ベクトル和	各アルゴリズムのスコアのベクトル和を使用して、マッチ スコアを決定します。このスコアリング方法は、最終的なマッチ スコアで比例的に表される 1 つ以上のアルゴリズムでより高いマッチ スコアが求められる場合に便利です。最終的なスコアの計算に使用される式は、次のとおりです。 sqrt(a^2 + b^2 + c^2) / sqrt(n)。ここで、a、b、および c は 3 つの異なるアルゴリズムのスコアであり、n は使用するアルゴリズムの数です。

g) フィールドの値がマッチするかどうかを確認するために使用する 1 つ以上のアルゴリズムを選択します。次のいずれかを選択します。

Acronym (頭字語)
頭字語データを探し、企業名がその頭字語にマッチするかどうかを判定します。あるいは、各単語の最初の文字を使用して頭字語を作成します。例: Internal Revenue Service とその頭字語 IRS はマッチすると判断され、マッチ スコア 100 が返されます。

Character Frequency (文字出現回数)
文字列に含まれるすべての文字の出現回数を個別に確認し、2 つの文字列における全体的な出現回数を比較します。

Daitch-Mokotoff Soundex
発音が同じでも綴りが異なるスラブ語およびイディッシュ語の姓をより正確にマッチングできるようにする音声アルゴリズム。コード化された名前の長さは 6 桁で、1 つの名前に対して可能性のある複数のエンコーディングを返すことができます。このオプションは、ドイツ系またはスラブ系の姓を処理する際の Soundex の制限に対応するために作成されました。

Date
日付のフォーマットにかかわらず、入力レコードに含まれる日付を比較します。[オプション] 列の [編集] をクリックして、次のオプションを指定します。

- **月を必須とする**: 年のみで構成される日付のマッチングを防ぎます。
- **日を必須とする**: 月と年のみで構成される日付のマッチングを防ぎます。
- **MM/DD という転置形式もマッチとみなす**: 月と日が数値形式で指定されている場合に、サスペクトの月と候補の月、サスペクトの日と候補の日という標準的な比較に加えて、サスペクトの月と候補の日、サスペクトの日と候補の月の比較を行います。
- **MM/DD/YYYY ではなく DD/MM/YYYY 形式を優先する**: 月と日がともに数値形式で指定されており、両者が文脈からは識別できない場合の日付のパーシングに使用します。例えば、数字が 5 と 13 の場合、パーサーは自動的に 5 を月に、13 を日に割り当てます。1 年に 12 カ月しかないためです。しかし、数字が 5 と 12 (または、ともに 12 以下の任意の 2 つの数字) の場合、パーサーは最初の数字が月であるとみなします。このオプションを選択することにより、パーサーが最初の数字を月ではなく日として読むように指定することができます。
- **範囲オプション — 全体**: マッチング日付間の最大日数を設定できます。例えば、全体範囲として 35 日と入力し、候補日付が 2000/12/31 の場合、サスペクト日付 2001/02/05 はマッチですが、サスペクト日付 2001/02/06 はマッチではありません。全体範囲として 1 日と入力し、候補日付が 2000/01 の場合、サスペクト日付 1999 はマッチですが (1999/12/31 と比較します)、サスペクト日付 2001/01 はマッチではありません。

- **範囲オプション — 年**: 月、日とは別に、マッチング日付間の年数を設定できます。例えば、年の範囲として 3 年と入力し、候補日付が 2000/01/31 の場合、サスペクト日付 2003/01/31 はマッチですが、サスペクト日付 2003/02 はマッチではありません。同様に、候補日付が 2000 の場合、サスペクト日付 2003/03 は月の不整合がなく、3 年の範囲内にあるので、マッチとなります。
- **範囲オプション — 月**: 年、日とは別に、マッチング日付間の月数を設定できます。例えば、月の範囲として 4 と入力し、候補日付が 2000/01/01 の場合、サスペクト日付 2000/05 は日の不整合がなく、4 カ月の範囲内にあるのでマッチですが、サスペクト日付 2000/05/02 は日の不整合があるためマッチではありません。
- **範囲オプション — 日**: 年、月とは別に、マッチング日付間の日数を設定できます。例えば、日の範囲として 5 と入力し、候補日付が 2000/01/01 の場合、サスペクト日付 2000/01 は日の不整合がないのでマッチですが、サスペクト日付 1999/12/27 は月の不整合があるためマッチではありません。

Double Metaphone	文字の発音表記に基づいて、2 つの文字列間の類似性を判断します。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。
Edit Distance (編集距離)	1 つの文字列をもう 1 つの文字列に変換するために必要な削除、挿入、または置換の数に基づいて、2 つの文字列間の類似性を判断します。
Euclidean Distance	結合された語のベクトル空間を次元として使用して、2 つの文字列間の類似性の尺度を提供します。2 つの整数の最大公約数も判断します。1 組の正の整数から、小さい方の数と 2 つの数の差で構成される新しい 1 組の数字を形成します。2 つの数字が等しくなるまでこのプロセスを繰り返します。その数字が元のペアの最大公約数となります。例えば、21 は 252 と 105 の最大公約数です (252 = 12 × 21、105 = 5 × 21)。したがって、252 − 105 = (12 − 5) × 21 = 147 なので、147 と 105 の最大公約数も 21 です。
Exact Match (完全一致)	2 つの文字列が同じであるかどうかを判断します。
Initials (頭文字)	パースした個人名の頭文字のマッチングに使用します。
Jaro-Winkler Distance (Jaro-Winker 距離)	1 つの文字列をもう 1 つの文字列に変換するために要する文字の置換の数に基づいて、2 つの文字列間の類似性を判断します。このオプションは、個人名などの短い文字列用に開発されました。

Keyboard Distance (キーボード距離)	キーボード上のキーの位置によって重みを付け、1 つの文字列をもう一方の文字列に変換するために必要な削除、挿入、または置換の数に基づいて、2 つの文字列間の類似性を判断します。[オプション] 列の [編集] をクリックして、使用しているキーボードのタイプを QWERTY (米国)、QWERTZ (オーストリアおよびドイツ)、または AZERTY (フランス) の中から指定します。
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。
Kullback-Liebler Distance	2 つの文字列内の単語の分散の相違に基づいて、2 つの文字列間の類似性を判断します。
Metaphone	文字の発音表記に基づいて、2 つの英語の文字列間の類似性を判断します。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。
Metaphone (スペイン語)	文字の発音表記に基づいて、2 つの文字列間の類似性を判断します。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。
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。
Name Variant (名前の派生形)	2 つの名前が互いの派生形であるかどうかを判断します。このアルゴリズムは、2 つの名前が互いの派生形である場合はマッチ スコア 100 を返し、互いの派生形でない場合はマッチ スコア 0 を返します。例えば、JOHN は JAKE の派生形なので、マッチ スコア 100 が返されます。JOHN は HENRY の派生形ではないので、マッチ スコア 0 が返されます。[オプション] 列の **[編集]** をクリックして、[Name Variant (名前の派生形)] オプションを選択します。詳細については、**Name Variant Finder**（329ページ）を参照してください。
NGram Distance	後続の語の蓋然性を前の n 個の語に基づいて計算します。この語には、音素、シラブル、文字、単語、基本対、または任意の組み合わせの文字が含まれます。このアルゴリズムには NGram のサイズを入力するオプションがあります。サイズのデフォルト値は 2 です。

NGram Similarity

2 つの文字列の類似度を、音素、シラブル、文字、単語、または基本対の最長の共通部分系列の長さに基づいて求めます。

このアルゴリズムには、以下のオプションが含まれています。

- **Ngram サイズ**: NGram のサイズを入力します。デフォルト値は 2 です。
- **ノイズ文字の除去**: 句読文字を空白で置き換える場合はこのチェックボックスをオンにします。
- **空白の除去**: 単語をマージする場合はこのチェック ボックスをオンにします。

Numeric String (数値文字列)

住所行の数値属性を文字から分離して、住所行を比較します。例えば、住所文字列 1234 Main Street Apt 567 では、文字列の数値属性 (1234567) がパースされ、残りの文字列値 (Main Street Apt) とは異なる処理が行われます。このアルゴリズムでは、まず数値アルゴリズムを使用して、文字列の中の数値データが照合されます。数値データのマッチが 100 の場合、Edit Distance と Character Frequency を使用して、英字データが照合されます。最終的なマッチ スコアは次のように計算されます。

```
(numericScore + (EditDistanceScore +
CharacterFrequencyScore) / 2) / 2
```

例えば、以下の 2 つの住所は次のように計算され、マッチ スコアは 95.5 になります。

123 Main St Apt 567
123 Maon St Apt 567

数値スコア = 100
Edit Distance (編集距離) = 91
Character Frequency (文字出現回数) = 91

91 + 91 = 182
182/2 = 91
100 + 91 = 191
191/2 = 95.5

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 Smath" 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 Smath" は、このアルゴリズムによってどちらも "JANSNATH" というインデックスが付けられているからです。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。Soundex と異なり、いくつかのマルチキャラクタ N グラムを処理し、相対母音位置を維持します。

> **注：** このアルゴリズムは、英字以外の文字を処理しません。英字以外の文字を含むレコードの処理は失敗します。

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。

Soundex 文字の発音表記に基づいて、2 つの文字列間の類似性を判断します。

SubString (部分文字列) 1 つの文字列が別の文字列内に出現するかどうかを判断します。

Syllable Alignment 音声情報と編集距離ベースの計算を組み合わせます。比較される文字列を対応する音節シーケンスに変換し、1 つの音節シーケンスを他の音節シーケンスに変換するのに必要な編集の数を計算します。

次の表に、選択した親のスコアリング方法と、使用できるアルゴリズム数との論理的関係を示します。

表 2 : マッチング アルゴリズムとスコアリング方法の表

スコアリング方法	アルゴリズム	
	単一	複数
加重平均	不可	はい
平均	不可	はい
最大値	はい	はい

スコアリング方法	アルゴリズム	
	単一	複数
最小値	不可	はい
ベクトル和	不可	はい

6. Interflow Match、Intraflow Match、または Transactional Match でルールを定義し、そのルールを他のステージや他のユーザと共有する場合は、ウィンドウの上部にある **[保存]** ボタンをクリックします。

否定のマッチ条件

マッチ条件とは、2 つのレコードがマッチするとみなすためにどのフィールドをマッチングするかを表す文です。しかし、2 つのフィールドが一致 *しない* 場合に 2 つのレコードがマッチするとみなす条件を定義したい場合もあります。*否定*として知られるこの手法は、マッチ ルール内の条件のロジックを反転させるものです。

例えば、コール センターの顧客サポート レコードがあり、複数のアカウントに関連してコール センターに問い合わせをした顧客を特定したいとします。つまり、複数のアカウントに関連付けられている個人を特定します。複数のアカウントを所有する顧客を特定するには、名前は一致するがアカウント番号は一致 *しない* レコードをマッチさせることが必要です。この場合は、アカウント番号のマッチ条件に対して否定を用いることになります。

否定を用いるには、マッチ ルールの定義時に **[真でない場合に一致]** チェック ボックスをオンにします。このオプションは、マッチ ルール内の親 (条件のグループ) と子 (個々の条件) の両方に適用できます。親に適用する場合と子に適用する場合で、このオプションの効果は少し異なります。**[真でない場合に一致]** オプションを親に適用すると、マッチング方法のオプションが次のように実質的に反転します。

- マッチング方法が **[すべて真]** である場合は、"いずれかが偽" という条件になります。このマッチ ルールは、親の下に偽と評価される子が少なくとも 1 つ存在する場合のみ親を偽と評価し、レコードをマッチとみなします。**[真でない場合に一致]** オプションが有効であるため、偽と評価される場合にマッチすることになります。
- マッチング方法が **[いずれかが真]** である場合は、"いずれも真でない" という条件になります。このマッチ ルールは、真と評価される子が 1 つもない場合のみ、レコードをマッチとみなします。いずれかの子が真である場合は親が真となりますが、 **[真でない場合に一致]** オプションが有効であるため、真と評価される場合はマッチとはみなされません。真と評価される子が 1 つもなく、親が "真でない" と評価される場合のみ、このルールにマッチします。
- マッチング方法が **[しきい値で判断]** である場合は、指定されたしきい値以上のレコードをマッチとみなすという条件が、しきい値未満のレコードをマッチとみなすという条件に変わります。

指定された値よりもしきい値が低いレコードは偽と評価されますが、**[真でない場合に一致]** が有効であることから、これがマッチとみなされます。

[真でない場合に一致] オプションは、マッチ ルール内の子要素に適用する場合の方が容易に理解できます。この場合は単純に、アルゴリズムによってマッチと *みなされない* 場合に 2 つのレコードはマッチすると判断されます。

マッチ ルールのテスト

マッチ ルールを定義した後、その結果を確認するためにマッチ ルールをテストできます。マッチ ルールをテストするには、マッチ ルール評価を使用して、少量のサンプル データに対するマッチ ルールの効果を調べます。

1. Enterprise Designer でデータフローを開きます。
2. テストするマッチ ルールを含むステージをダブルクリックします。

   マッチ ルールは、Interflow Match、Intraflow Match、および Transactional Match で使用されます。

3. マッチ ルールの階層で、テストするノードを選択し、**[評価]** をクリックします。
4. **[インポート]** タブで、テスト データ (サスペクトと最大 10 件の候補) を入力します。テスト データの入力方法は 2 つあります。

   - テスト データを手動で入力するには、**[サスペクト]** の下にサスペクト レコードと、**[候補]** の下に最大 10 件の候補を入力します。レコードを入力したら、**[エクスポート]** をクリックしてレコードをファイルに保存できます。それ以降は、データを手動で再入力する代わりに、このファイルをインポートすることができます。
   - テスト データをファイルからインポートするには、**[インポート]** をクリックし、サンプル レコードを含むファイルを選択します。区切り記号付きファイルは、区切り文字としてカンマ、パイプ、またはタブを使用し、**[候補]** の下に表示されたフィールド名と一致するヘッダ フィールドを持つヘッダ レコードを含む必要があります。Household 入力のヘッダ レコードのサンプルを以下に示します。

     ```
     Name,AddressLine1,City,StateProvince
     ```

5. これらのいずれかの方法を使用してルールを評価します。

   - **[現在のルール]** をクリックします。これにより、**[マッチ ルール]** タブで定義されたルールが実行されます。結果は、一度に 1 つのサスペクトと候補のペアに対して表示されます。結果を循環して表示するには、矢印ボタンをクリックします。フィールドのスコアとアルゴリズムが、マッチ ルール制御と同様のツリー形式で表示されます。結果は、XML フォーマットでエクスポートすることもできます。

注 : マッチ ルールを変更し、その変更内容をステージのマッチ ルールに適用する場合は、**[保存]** をクリックします。

- **[すべてのアルゴリズム]** をクリックします。この設定を使うと、マッチ ルールが無視され、すべてのアルゴリズムが各フィールドに実行されてサスペクトと候補のペアが評価されます。結果は、一度に 1 つのサスペクトおよび候補のペアについて表示され、矢印ボタンを使って循環して表示できます。

  マッチ ルールまたは入力を変更したときに結果が自動的に更新されるようにするには、**[自動更新]** チェック ボックスを選択します。この機能を **[すべてのアルゴリズム]** オプションを選択して使うと、入力を変更したときにのみ、結果が更新されます。

[スコア] の下に表示される結果は、次のように色分けされます。

- 緑 — 実行結果がマッチだったルール。
- 赤 — 実行結果がマッチでなかったルール。
- 灰色 — 無視されたルール。
- 青 — ルールに含まれる個別のアルゴリズムの結果。

評価結果を XML フォーマットでエクスポートするには、**[エクスポート]** をクリックします。

マッチ ルールの共有

モジュール、ステージ、データフロー、およびユーザ間で共有できるマッチ ルールを作成できます。マッチ ルールを共有すると、マッチ ルールを 1 回だけ作成し、その後は必要なときにいつでも参照できるので、データフローの開発が容易になります。また、同じ機能を実行することを目的としたマッチ ルールのデータフロー間での一貫性の確保が容易になります。

マッチ ルール リポジトリに保存されているマッチ ルールは、管理ユーティリティを使用すると .mr ファイルまたは .json ファイルとしてエクスポートできます。これらのルールをその他の Spectrum インストールと共有しようとする場合や、Spectrum Big Data Quality SDK で利用する場合には、この機能が特に便利です。

注 : デフォルトでは、.mr ファイルとしてマッチ ルールがエクスポートされます。

- Interflow Match、Intraflow Match、または Transactional Match で作成したマッチ ルールを共有するには、ステージのオプション ウィンドウ上部にある **[保存]** ボタンをクリックします。
- マッチ ルール管理ツールでルールを作成すると、そのルールは自動的にデータフローですべてのユーザが使用できるようになります。マッチ ルール管理ツールを表示するには、Enterprise Designer で **[ツール] > [マッチ ルール管理]** を選択します。

共有マッチ ルールの表示

Enterprise Designer では、Spectrum™ Technology Platform システムで使用可能な共有マッチ ルールをすべて参照できます。これらのマッチ ルールをデータフローの Interflow Match、Intraflow Match、および Transactional Match ステージで使用して、マッチングを実行できます。

マッチ ルール リポジトリ内のマッチ ルールを参照するには、以下の手順に従います。

1. Enterprise Designer を開きます。
2. **[ツール] > [マッチ ルール管理]** を選択します。
3. 表示するルールを選択し、**[表示]** をクリックします。

JSON オブジェクトとしてのカスタム マッチ ルールの作成

マッチ ルールは、データフローのオプションとしてエクスポーズすると、実行時に設定して渡すことができます。これによって、複数のマシン間でマッチ ルールを共有でき、また、既存のマッチ ルールを JSON 形式のマッチ ルール文字列でオーバーライドできます。 また、プロセス フローまたは Job Executor コマンドライン ツールからジョブを呼び出すときに、ステージ オプションを設定することもできます。

MatchRule と MatchInfo のスキーマは、以下のフォルダにあります。

`<Spectrum Location>\server\modules\jsonSchemas\matcher` という命名規則に従います。

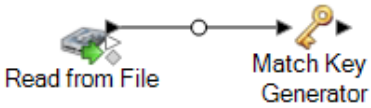
1. マッチ ルールが含まれるデータフローを保存し、エクスポーズします。
2. マッチ ルールを使用するデータフローを開きます。
3. 編集 > データフローオプション に移動します。
4. **[データフロー オプションをステージにマッピングします]** テーブルで、マッチ ルールを使用するマッチング ステージをクリックし、**[カスタム マッチ ルール]** ボックスをオンにします。
5. オプション: **[オプション ラベル]** フィールドのマッチ ルール名を "カスタム マッチ ルール" から目的の名前に変更します。
6. **[OK]** を 2 回クリックします。

単一ソースからのレコードのマッチング

この手順では、Intraflow Match ステージを使用して、単一のデータ ソース (ファイルやデータベース テーブルなど) から、ユーザが指定したマッチング条件に基づいて相互に関連のあるレコードのグループを識別する方法を説明します。このデータフローは、レコードをコレクションにグループ化し、そのコレクションを出力ファイルに書き込みます。

1. Enterprise Designer で、新しいデータフローを作成します。
2. ソース ステージをキャンバスにドラッグします。
3. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
4. Match Key Generator ステージをキャンバスにドラッグし、ソース ステージに接続します。

   例えば、Read from File ソース ステージを使用している場合は、データフローは次のようになります。

   Match Key Generator は、レコードごとに非ユニーク キーを作成します。この非ユニーク キーは、潜在的な重複レコードのグループを特定するためにマッチング ステージで使用できます。マッチ キーを使用すると、レコードをマッチ キー別にグループ化し、各グループ内でのみレコードを比較できるので、マッチング プロセスが促進されます。

5. Match Key Generator をダブルクリックします。
6. **[追加]** をクリックします。
7. 各レコードのマッチ キーを生成するために使用するルールを定義します。

表 3 : Match Key Generator のオプション

オプション名	説明 / 有効な値
アルゴリズム	

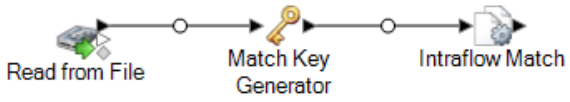
オプション名	説明 / 有効な値
	マッチ キーの生成に使用するアルゴリズムを指定します。次のいずれかです。 **Consonant (子音)** 指定されたフィールドを、子音を削除して返します。 **Double Metaphone** 文字の発音表記に基づくコードを返します。Double Metaphone は Metaphone アルゴリズムの改良版で、さまざまな言語に多数存在する不規則性を考慮しています。 **Koeln** ドイツ語で発音される名前に、音声によってインデックスを付けます。同じ発音を持つ名前を同じ表現にエンコードできるので、綴りに小さな相違があっても、マッチさせることができます。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。

オプション名	説明 / 有効な値
	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。 **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** 選択されているフィールドの指定部分を返します。
フィールド名	選択したアルゴリズムを適用してマッチ キーを生成するフィールドを指定します。例えば、LastName というフィールドを選択し、Soundex アルゴリズムを選択した場合、Soundex アルゴリズムが LastName フィールドのデータに適用されて、マッチ キーが生成されます。
開始位置	指定したフィールド内での開始位置を指定します。すべてのアルゴリズムで開始位置を指定できるとは限りません。
長さ	開始位置から含める文字の数を指定します。すべてのアルゴリズムで長さを指定できるとは限りません。
ノイズ文字の削除	ハイフン、空白、その他の特殊文字等、英数字以外の文字を入力フィールドからすべて削除します。
ソート入力	入力フィールド内の文字または語をすべてアルファベット順にソートします。 **文字** ユニーク ID を作成する前に、入力フィールドの文字値をソートします。 **語** ユニーク ID を作成する前に、入力フィールドの各語値をソートします。

8. ルールの定義が終了したら、**[OK]** をクリックします。
9. さらにマッチルールを追加する場合は、**[追加]**をクリックして追加します。追加するものがなくなったら、**[OK]** をクリックします。

10. Intraflow Match ステージをキャンバスにドラッグし、Match Key Generator ステージに接続します。

    例えば、Read from File ソース ステージを使用している場合は、データフローは次のようになります。



11. Intraflow Match をダブルクリックします。
12. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

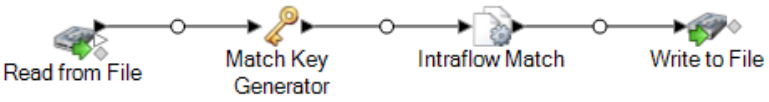
    注： Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

13. **[グループ化]** フィールドで、**[マッチ キー]** を選択します。

    同じマッチ キーを持つレコードがグループに配置されます。マッチ ルールがグループ内のレコードに適用されて、重複があるかどうかが確認されます。各レコードのマッチ キーは、この手順で先に設定した Generate Match Key ステージによって生成されます。

14. 他のオプションの変更の詳細については、マッチ ルールの作成（72ページ）を参照してください。
15. **[OK]** をクリックして Intraflow Match の設定を保存し、データフロー キャンバスに戻ります。
16. シンク ステージをキャンバスにドラッグし、マッチ キーを生成ステージに接続します。

    例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。



17. シンク ステージをダブルクリックして設定します。

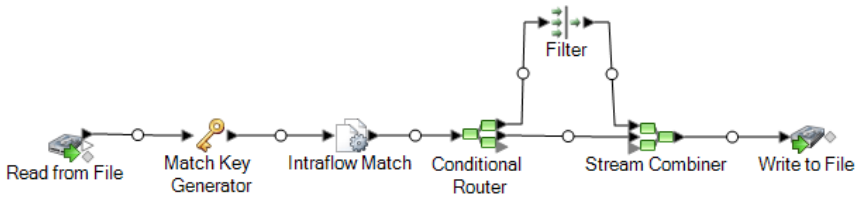
    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

1 つのソースのレコードをマッチングするデータフローが完成しました。

単一データ ソースのレコードのマッチングの例

クレジット カード会社のデータ管理責任者が、顧客データベースを分析し、複数のレコードに重複して現れる住所とその住所に住む顧客の氏名を洗い出して、同じ世帯に重複して送付されるクレジット カード契約勧誘のダイレクト メールの数を最小限にすることを検討しています。

この例では、単一の入力ファイルの情報を比較して世帯ごとに1つのレコードを含む出力ファイルを作成することによって、同じ世帯のメンバーを識別する方法を示します。



Read from File ステージは、世帯ごとのユニーク レコードと、同じ世帯のものである可能性があるレコードの両方を含むデータを読み込みます。入力ファイルには名前と住所が含まれます。

Match Key Generator は、重複の可能性があるレコードを意味する類似レコードに共通して与えられる非ユニーク キーであるマッチ キーを作成します。

Intraflow Match ステージは、同じマッチ キーを持つレコードを比較し、ユニークなレコードまたは同じ世帯の複数のレコードの1つとして各レコードをマークします。

Conditional Router は各世帯のレコードのコレクションであるレコードを Filter ステージに送信し、Filter ステージは世帯ごとに 1 レコードだけを残して他のすべてのレコードを除外し、Stream Combiner ステージに送ります。Conditional Router ステージはユニーク レコードも Stream Combiner に直接送ります。

最後に、Write to File ステージは世帯ごとに 1 レコードを含む出力ファイルを作成します。

ソース間でのレコードのマッチング

この手順では、Interflow Match ステージを使用して、あるソースに含まれるレコードが別のソースのレコードにマッチするかどうかを特定する方法を説明します。第 1 のソースにはサスペクト レコードが含まれ、第2のソースには候補レコードが含まれます。このデータフローでは、レコー

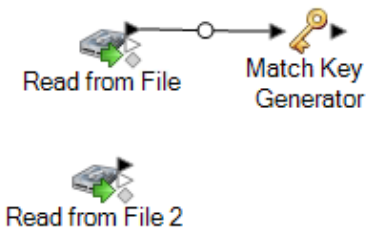
ドは別のソースのレコードにのみマッチングします。同じソース内でレコードがマッチングするかは確認されません。このデータフローは、レコードを一致するレコードのコレクションにグループ化し、そのコレクションを出力ファイルに書き込みます。

1. Enterprise Designer で、新しいデータフローを作成します。
2. 2 つのソース ステージをキャンバスにドラッグします。1 つのステージはサスペクト レコードのソースを参照するように設定し、もう 1 つのステージは候補レコードのソースを参照するように設定します。

   ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。

3. Match Key Generator ステージをキャンバスにドラッグし、ソース ステージの 1 つに接続します。

   例えば、Read from File ソース ステージを使用している場合は、データフローは次のようになります。



   Match Key Generator は、レコードごとに非ユニーク キーを作成します。この非ユニーク キーは、潜在的な重複レコードのグループを特定するためにマッチング ステージで使用できます。マッチ キーを使用すると、レコードをマッチ キー別にグループ化し、各グループ内でのみレコードを比較できるので、マッチング プロセスが促進されます。

   > **注：** 2 番目の Match Key Generator ステージは後で追加します。この段階では、キャンバスに必要なステージは 1 つだけです。

4. Match Key Generator ステージをダブルクリックします。
5. **[追加]** をクリックします。
6. 各レコードのマッチ キーを生成するために使用するルールを定義します。

表 4 : Match Key Generator のオプション

オプション名	説明 / 有効な値
アルゴリズム	

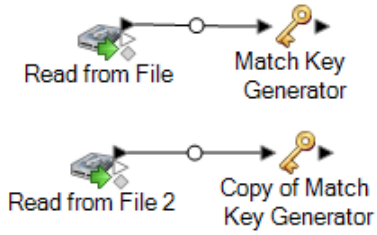
オプション名	説明 / 有効な値
	マッチ キーの生成に使用するアルゴリズムを指定します。次のいずれかです。
	Consonant (子音) 指定されたフィールドを、子音を削除して返します。
	Double Metaphone 文字の発音表記に基づくコードを返します。Double Metaphone は Metaphone アルゴリズムの改良版で、さまざまな言語に多数存在する不規則性を考慮しています。
	Koeln ドイツ語で発音される名前に、音声によってインデックスを付けます。同じ発音を持つ名前を同じ表現にエンコードできるので、綴りに小さな相違があっても、マッチさせることができます。結果は常に一連の数字です。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。

オプション名	説明 / 有効な値
	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。 **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** 選択されているフィールドの指定部分を返します。
フィールド名	選択したアルゴリズムを適用してマッチ キーを生成するフィールドを指定します。例えば、LastName というフィールドを選択し、Soundex アルゴリズムを選択した場合、Soundex アルゴリズムが LastName フィールドのデータに適用されて、マッチ キーが生成されます。
開始位置	指定したフィールド内での開始位置を指定します。すべてのアルゴリズムで開始位置を指定できるとは限りません。
長さ	開始位置から含める文字の数を指定します。すべてのアルゴリズムで長さを指定できるとは限りません。
ノイズ文字の削除	ハイフン、空白、その他の特殊文字等、英数字以外の文字を入力フィールドからすべて削除します。
ソート入力	入力フィールド内の文字または語をすべてアルファベット順にソートします。 **文字** ユニーク ID を作成する前に、入力フィールドの文字値をソートします。 **語** ユニーク ID を作成する前に、入力フィールドの各語値をソートします。

7. ルールの定義が終了したら、**[OK]** をクリックします。
8. キャンバスで Match Key Generator ステージを右クリックして、**[ステージのコピー]** を選択します。

9. キャンバスの空いている領域を右クリックし、**[貼り付け]** を選択します。
10. Match Key Generator のコピーを他のソース ステージに接続します。

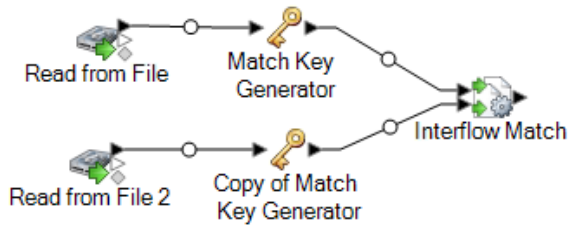
    例えば、Read from File 入力ステージを使用している場合は、データフローは次のようになります。



    データフローには、まったく同じルールを使用して各ソースのマッチ キーを生成する 2 つの Match Key Generator ステージが含まれるようになりました。このデータフローが正しく機能するには、同じ設定の Match Key Generator ステージを用意することが不可欠です。

11. Interflow Match ステージをキャンバスにドラッグし、各 Match Key Generator ステージをそれに接続します。

    例えば、Read from File 入力ステージを使用している場合は、データフローは次のようになります。



12. Interflow Match ステージをダブルクリックします。
13. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

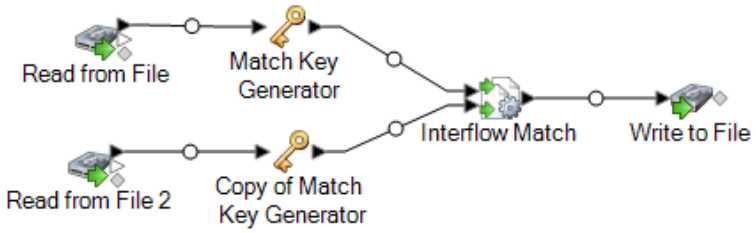
    > **注：** Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

14. **[グループ化]** フィールドで、**[マッチ キー]** を選択します。

同じマッチ キーを持つレコードがグループに配置されます。マッチ ルールがグループ内のレコードに適用されて、重複があるかどうかが確認されます。各レコードのマッチ キーは、この手順で先に設定した Generate Match Key ステージによって生成されます。

15. 他のオプションの変更の詳細については、マッチ ルールの作成（72ページ）を参照してください。
16. シンク ステージをキャンバスにドラッグし、Interflow Match ステージに接続します。

    例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。



17. シンク ステージをダブルクリックして設定します。

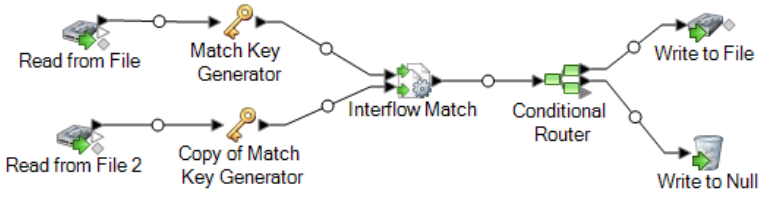
    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

2 つのデータ ソースのレコードをマッチングするデータフローが完成しました。

複数のソースからのレコードのマッチングの例

ダイレクト メールの会社で、メール拒否リストに載っているユーザを識別してダイレクト メールが送信されないようにします。1つのファイルには受取人のリストがあり、もう 1 つのファイル (禁止ファイル) にはダイレクト マーケティング メールの受け取りを拒否している人のリストがあります。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



Read from File ステージでメーリング リストからデータを読み込み、Read from File 2 ステージで禁止リストからデータを読み込みます。2 つの Match Key Generator ステージはまったく同じ設定なので、これらによって生成されるマッチ キーを Interflow Match で使用して、一致する可能性のあるグループを作成できます。Interflow Match はメーリング リストと禁止ファイルの両方にあるレコードを識別し、これらのレコードを重複としてマークします。Conditional Router は、ユニー

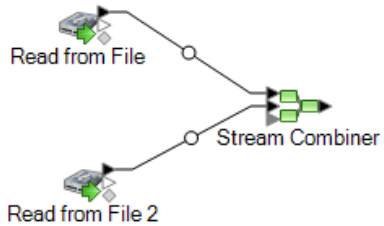
ク レコード (禁止リストで見つからなかったレコード) を Write to File に送信し、ファイルに書き込みます。Conditional Router ステージは他のすべてのレコードを Write to Null に送信して破棄します。

ソース間およびソース内でのレコードのマッチング

この手順では、Interflow Match ステージを使用して、あるファイルに含まれるレコードが別のファイルおよび同じファイルのレコードにマッチするかどうかを特定する方法を説明します。例えば、2 つのファイル (ファイル A とファイル B) があり、ファイル A 内にファイル B のレコードと一致するレコードがあるかどうかを確認すると同時に、ファイル A 内のレコードで、ファイル A 内の別のレコードに一致するものがあるかどうかも確認したいとします。これは、Stream Combiner と Intraflow Match ステージを併用して実現できます。

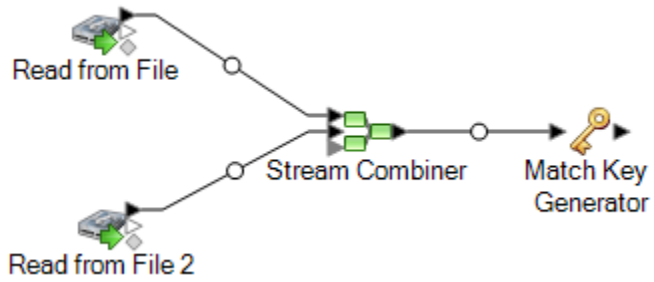
1. Enterprise Designer で、新しいデータフローを作成します。
2. ソース ステージをキャンバスにドラッグします。
3. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
4. 2 番目のソース ステージをキャンバスにドラッグし、2 番目のデータ ソースがデータフローに読み込まれるように構成します。
5. Stream Combiner ステージをキャンバス上にドラッグし、2 つのソース ステージをこのステージに接続します。

   例えば、データフローに 2 つの Read from File ステージがある場合、Stream Combiner を追加すると次のような構成になります。



6. Match Key Generator ステージをキャンバスにドラッグし、Stream Combiner ステージに接続します。

   例えば、データフローは次のような構成になります。



Match Key Generator は、レコードごとに非ユニーク キーを作成します。この非ユニーク キーは、潜在的な重複レコードのグループを特定するためにマッチング ステージで使用できます。マッチ キーを使用すると、レコードをマッチ キー別にグループ化し、各グループ内でのみレコードを比較できるので、マッチング プロセスが促進されます。

7. Match Key Generator をダブルクリックします。
8. **[追加]** をクリックします。
9. 各レコードのマッチ キーを生成するために使用するルールを定義します。

表 5 : Match Key Generator のオプション

オプション名	説明 / 有効な値
アルゴリズム	

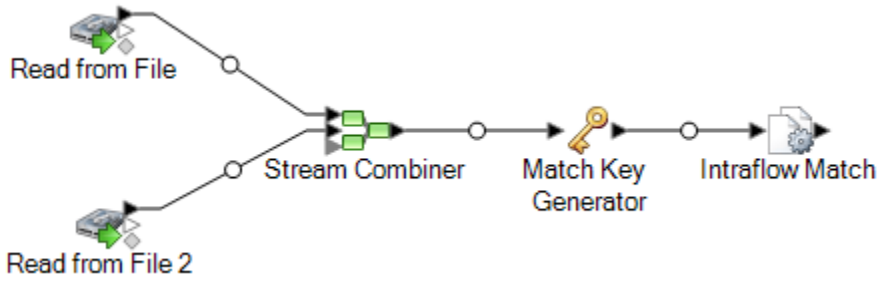
オプション名	説明 / 有効な値
	マッチ キーの生成に使用するアルゴリズムを指定します。次のいずれかです。
	Consonant (子音) 指定されたフィールドを、子音を削除して返します。
	Double Metaphone 文字の発音表記に基づくコードを返します。Double Metaphone は Metaphone アルゴリズムの改良版で、さまざまな言語に多数存在する不規則性を考慮しています。
	Koeln ドイツ語で発音される名前に、音声によってインデックスを付けます。同じ発音を持つ名前を同じ表現にエンコードできるので、綴りに小さな相違があっても、マッチさせることができます。結果は常に一連の数字です。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。

オプション名	説明 / 有効な値
	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。 **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** 選択されているフィールドの指定部分を返します。
フィールド名	選択したアルゴリズムを適用してマッチ キーを生成するフィールドを指定します。例えば、LastName というフィールドを選択し、Soundex アルゴリズムを選択した場合、Soundex アルゴリズムが LastName フィールドのデータに適用されて、マッチ キーが生成されます。
開始位置	指定したフィールド内での開始位置を指定します。すべてのアルゴリズムで開始位置を指定できるとは限りません。
長さ	開始位置から含める文字の数を指定します。すべてのアルゴリズムで長さを指定できるとは限りません。
ノイズ文字の削除	ハイフン、空白、その他の特殊文字等、英数字以外の文字を入力フィールドからすべて削除します。
ソート入力	入力フィールド内の文字または語をすべてアルファベット順にソートします。 **文字** ユニーク ID を作成する前に、入力フィールドの文字値をソートします。 **語** ユニーク ID を作成する前に、入力フィールドの各語値をソートします。

10. ルールの定義が終了したら、**[OK]** をクリックします。
11. さらにマッチルールを追加する場合は、**[追加]** をクリックして追加します。追加するものがなくなったら、**[OK]** をクリックします。

12. Intraflow Match ステージをキャンバスにドラッグし、Match Key Generator ステージに接続します。

    例えば、データフローは次のような構成になります。



13. Intraflow Match をダブルクリックします。
14. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

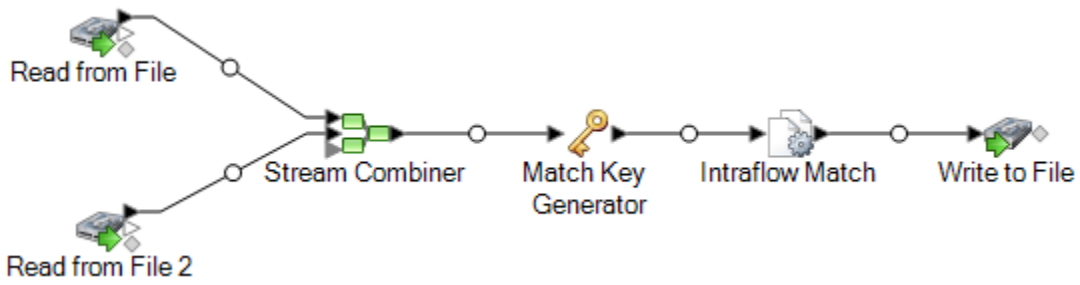
    > **注：** Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

15. **[グループ化]** フィールドで、**[マッチ キー]** を選択します。

    同じマッチ キーを持つレコードがグループに配置されます。マッチ ルールがグループ内のレコードに適用されて、重複があるかどうかが確認されます。各レコードのマッチ キーは、この手順で先に設定した Generate Match Key ステージによって生成されます。

16. 他のオプションの変更の詳細については、マッチ ルールの作成（72ページ）を参照してください。
17. **[OK]** をクリックして Intraflow Match の設定を保存し、データフロー キャンバスに戻ります。
18. シンク ステージをキャンバスにドラッグし、マッチ キーを生成ステージに接続します。

    例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。



19. シンク ステージをダブルクリックして設定します。

    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

データベースに対するレコードのマッチング

この手順では、サスペクト レコードがファイルやデータベースなどのソースからのもので、候補レコードが他の関係のないレコードを含むデータベースにある場合に、レコードをマッチングする方法について説明します。このデータフローは、各入力レコードについて、データベースでそのレコードの候補をクエリした後、Transactional Match ステージを使用してレコードをマッチングします。最後に、データフローはマッチング レコードのコレクションを出力ファイルに書き込みます。

> **注 :** Transactional Match は、サスペクト レコードと候補レコードの照合のみを行います。Intraflow Match のように、サスペクト レコードと他のサスペクト レコードとの照合は行いません。

1. Enterprise Designer で、新しいデータフローを作成します。
2. ソース ステージをキャンバスにドラッグします。
3. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
4. Candidate Finder ステージをキャンバスにドラッグし、それにソース ステージを接続します。

   例えば、Read from File ソース ステージを使用している場合は、データフローは次のようになります。

   Candidate Finder は、候補レコードを取得します。これらの候補レコードは、一連の潜在的なマッチとして、後のデータフローで Transactional Match によって評価されます。

5. キャンバスで Candidate Finder ステージをダブルクリックします。
6. **[接続]** フィールドで、クエリを実行して候補レコードを検索するデータベースを選択します。目的のデータベースがリストにない場合は、Management Console を開き、データベース接続を先に定義します。

7. [SQL] フィールドに、データフロー フィールドの 1 つの値に基づいて候補レコードを検索する SQL`SELECT`文を入力します。データフローのフィールドを参照するには、`${FieldName}`という形式を使用します。ここで、`FieldName` は参照するフィールドの名前です。

   例えば、LastName 列の値がデータフロー レコードの Customer_LastName フィールドと同じレコードをデータベースで検索する場合は、次のような SQL 文を作成します。

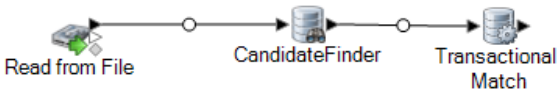
```
SELECT FirstName, LastName, Address, City, State, PostalCode
FROM Customer_Table
WHERE LastName = ${Customer_LastName};
```

8. **[フィールド マップ]** タブで、データベースの各列のデータが含まれるデータフローのフィールドを選択します。

   [選択フィールド] 列はデータベースの列のリストで、**[ステージ フィールド]** はデータフローのフィールドのリストです。

9. **[OK]** をクリックします。
10. Transactional Match ステージをキャンバスにドラッグし、Candidate Finder ステージをそれに接続します。

    例えば、Read from File 入力ステージを使用している場合は、データフローは次のようになります。



    Transactional Match では、サスペクト レコードを、Candidate Finder ステージから返される候補レコードと照合します。Transactional Match では、マッチング ルールを使用して、サスペクト レコードを、同じ候補グループ番号 (Candidate Finder で割り当てられる番号) を持つすべての候補レコードと比較して、重複を識別します。

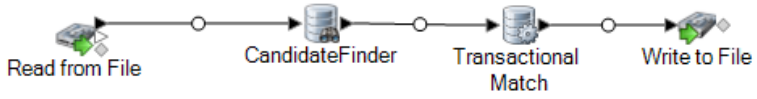
11. キャンバスで Transactional Match ステージをダブルクリックします。
12. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

    注： Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

13. 他のオプションの変更の詳細については、マッチ ルールの作成（72ページ）を参照してください。

14. Transactional Match ステージの設定が終了したら、**[OK]** をクリックします。
15. シンク ステージをキャンバスにドラッグし、Transactional Match ステージに接続します。

    例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。



16. シンク ステージをダブルクリックして設定します。

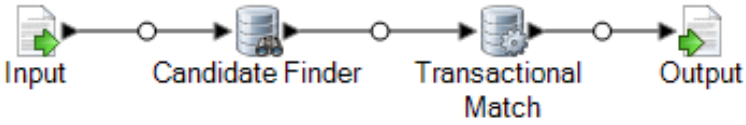
    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

2 つのデータ ソースのレコードをマッチングするデータフローが完成しました。

データベースに対するレコードのマッチングの例

ネット通販会社の営業部長は、オンラインの見込み客が既に購入履歴のある顧客なのか、新規の顧客なのかを確認したいと思っています。

以下のデータフロー サービスは、このビジネス シナリオの解決策を示しています。



このデータフローは、API 呼び出しまたは Web サービス呼び出しによって送られた見込み客データを評価するサービスです。顧客データベースの顧客データと照合してデータを評価し、見込み客が顧客かどうかを判定します。

Input ステージは、データフローが AddressLine1、City、Name、PostalCode、StateProvince の各入力フィールドを受け付けるように設定されています。[AddressLine1] と [Name] は、このテンプレートによるデータフロー処理のかなめとなるフィールドです。

Candidate Finder ステージで候補レコードを取得します。これらの候補レコードは、一連の潜在的なマッチとして、Transactional Match ステージで評価されます。

Transactional Match ステージは、サスペクト レコードを、Candidate Finder ステージから返された潜在的な候補レコードと照合します。Transactional Match では、マッチング ルールを使用して、サスペクト レコードを、同じ候補グループ番号 (Candidate Finder で割り当てられる番号) を持つすべての候補レコードと比較して、重複を識別します。この例では、Transactional Match は LastName と AddressLine1 を比較します。

Output ステージは、API または Web サービスの応答を使用してデータフローの結果を返します。

複数のマッチ ルールによるレコードのマッチング

サンプル データフローをダウンロード

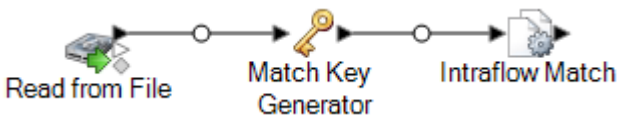
レコードのマッチングに複数のマッチング操作を使う必要がある場合、複数のマッチ キーを使うデータフローを作成し、その結果を組み合わせると実質的に複数の条件によるマッチングを行えます。例えば、レコードのマッチング条件として、

名前と住所の一致
OR
誕生日と政府発行 ID の一致をデータフローで使うとします。

このロジックでマッチングを実行するには、名前と住所のマッチングをあるステージで実行し、誕生日と政府発行 ID のマッチングを別のステージで実行してから、双方でマッチングしたレコードを 1 つのコレクションに収めるデータフローを作成します。

ここでは、2 つのマッチング ステージを通じてマッチングを処理するデータフローを設定する一般的な手順を説明します。この手順を具体的に説明するため、ここでは Intraflow Match ステージを使用しますが、同じテクニックは Interflow Match ステージにも応用できます。

1. Enterprise Designer で、新しいデータフローを作成します。
2. ソース ステージをキャンバスにドラッグします。
3. ソース ステージをダブルクリックして設定します。ソース ステージの設定手順については、『*データフロー デザイナー ガイド*』を参照してください。
4. 最初のマッチング処理を定義します。最初のマッチング処理の結果は、最初のマッチング条件を満たすレコード、つまりここでは名前と住所が条件を満たすレコードのコレクションになります。
   a) Match Key Generator ステージと Intraflow Match ステージをキャンバスにドラッグし、連結して以下のようなデータフローを作成します。



   a) Match Key Generator ステージでは、最初のマッチング処理に使うマッチ キーを定義します。

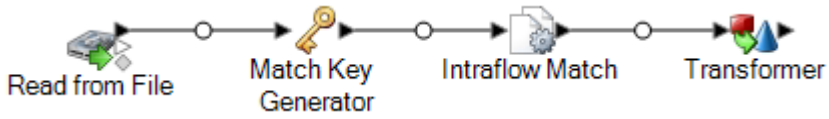
例えば、最初のマッチング処理で名前と住所のマッチングを確認する場合、姓と郵便番号が含まれるフィールドに基づいてマッチ キーを作成します。

b) Intraflow Match ステージで、最初のマッチング処理を行うマッチ ルールを定義します。

例えば、このマッチング ステージで名前と住所のマッチングを行うとします。

5. 最初のマッチング処理で得られたコレクション番号を別のフィールドに保存します。こうするのは、2 番目のマッチング処理で CollectionNumber フィールドが上書きされるからです。最初のマッチング処理の結果を保持するには、CollectionNumber フィールドの名前を変更する必要があります。

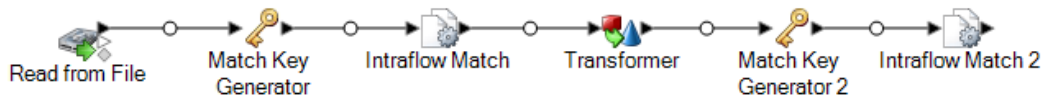
a) Transformer ステージをキャンバスにドラッグし、Intraflow Match ステージに連結して以下のようなデータフローを作成します。



b) Transformer ステージを設定して、CollectionNumber フィールドの名前を CollectionNumberPass1 に変更します。

6. 2 番目のマッチング処理を定義します。2 番目のマッチング処理の結果は、2 番目のマッチング条件を満たすレコード、つまりここでは誕生日と政府発行 ID が条件を満たすレコードのコレクションになります。

a) Match Key Generator ステージと Intraflow Match ステージをキャンバスにドラッグし、連結して以下のようなデータフローを作成します。



b) 2 番目の Match Key Generator ステージでは、2 番目のマッチング処理に使うマッチ キーを定義します。

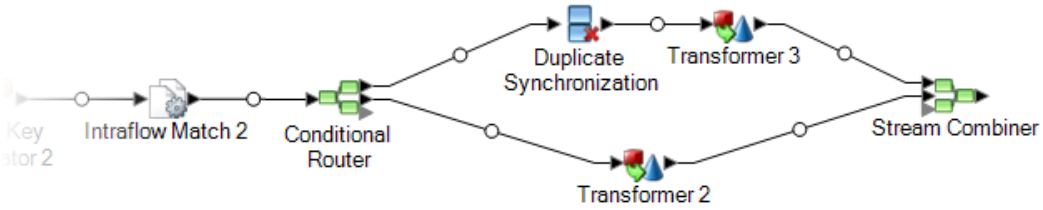
例えば、2 番目のマッチング処理で誕生日と政府発行 ID のマッチングを確認する場合、誕生日と政府発行 ID が含まれるフィールドに基づいてマッチ キーを作成します。

c) 2 番目の Intraflow Match ステージでは、2 番目のマッチング処理に使うマッチ キーを定義します。

例えば、このマッチング ステージで誕生日と政府発行 ID のマッチングを行うとします。

7. 2 番目のマッチング処理で重複レコードが見つかった場合、最初のマッチング処理でもそれが重複レコードとされていたかどうかを確認してください。

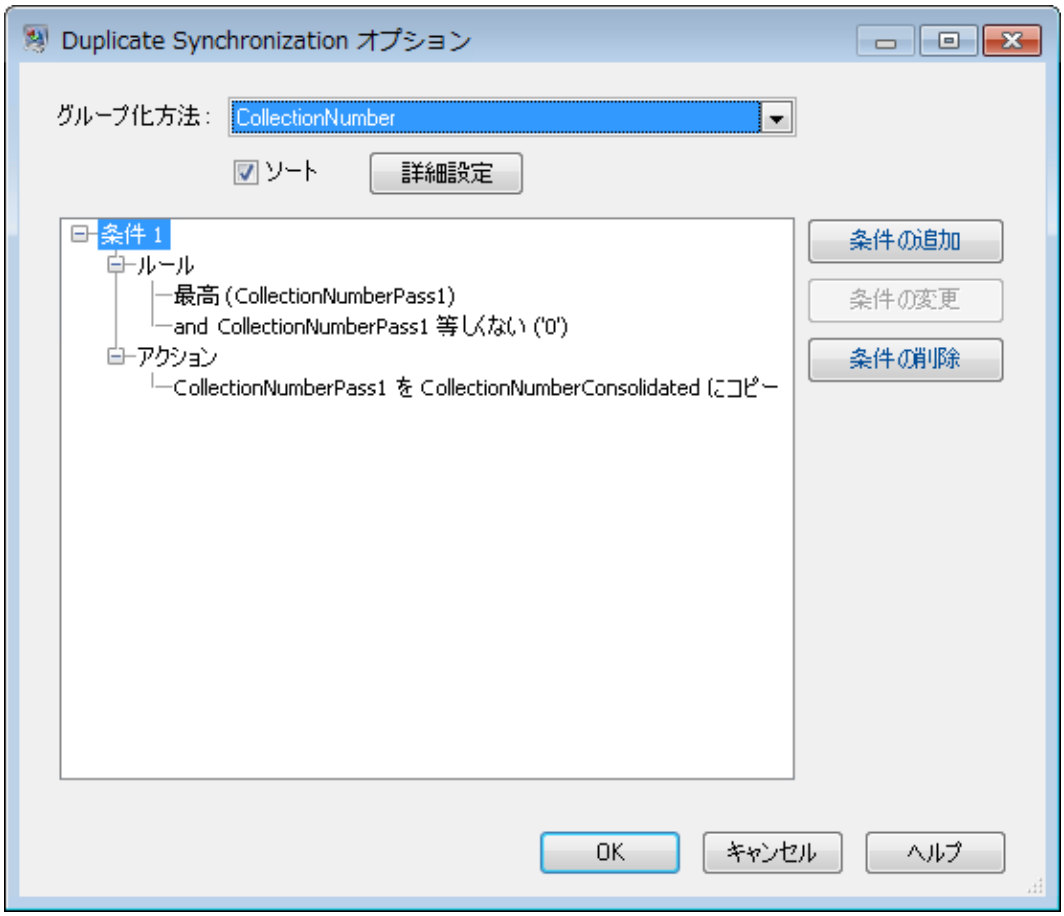
a) 2 番目の Intraflow Match ステージの後に、以下のデータフロー処理を作成します。



b) Conditional Router ステージを設定して、CollectionNumber フィールドが 0 ではないレコードを Duplicate Synchronization ステージに送ります。

これで、重複レコードが 2 番目のマッチング処理から Duplicate Synchronization ステージに渡されます。

c) Duplicate Synchronization ステージを設定して、レコードを CollectionNumer フィールドに従ってグループ化します。CollectionNumer フィールドには、2 番目のマッチング処理で得られたコレクション番号が含まれます。次に、各コレクションに含まれるレコードについて、最初のマッチング処理で重複レコードとされたものがないか確認します。該当するレコードがあれば、そのコレクション番号を CollectionNumberConsolidated という新しいフィールドにコピーします。この操作を行うには、Duplicate Synchronization を以下の手順で設定します。



d) Duplicate Synchronization ステージの次に位置する Transformer ステージでは、以下のスクリプトを使ってカスタム トランスフォームを作成します。

```
if (data['CollectionNumberConsolidated'] == null) {
 data['CollectionNumberConsolidated'] = data['CollectionNumber']
}
```

e) Conditional Router の直後にある Transformer (サンプル データフローの Transformer 2) では、トランスフォームを設定して CollectionNumberPass1 を CollectionNumberConsolidated にコピーします。

これで、ユニーク レコードが 2 番目のマッチング処理から取り出され、CollectionNumberPass1 が CollectionNumberConsolidated にコピーされます。

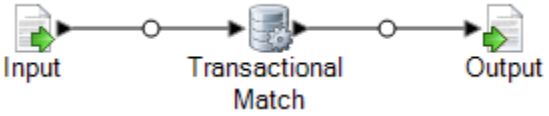
8. Stream Combiner の後、どちらかのマッチング処理でマッチングしたレコードのコレクションが完成します。CollectionNumberConsolidated フィールドを見れば、マッチング レコードかどうかがわかります。Stream Combiner ステージの後に、必要であればシンクや追加の処理を追加できます。

ユニバーサル マッチ サービスの作成

ユニバーサル マッチ サービスは、任意のマッチ ルールを使用してマッチングを実行し、任意の入力フィールドを受け付けることができるサービスです。このサービスは、マッチ ルール名を入力オプションとして受け取るので、使用するマッチ ルールを API 呼び出しまたは Web サービス要求で指定できます。入力スキーマは事前に定義されていないので、マッチングしたいレコードのタイプにふさわしいフィールドを任意に使用できます。ユニバーサル マッチング サービスを作成すると、マッチ ルールごとにサービスを用意する手間が省け、また、新しいマッチ ルールを追加する場合もサービスを追加する必要がありません。

以下に、ユニバーサル マッチング サービスを作成する手順と、Web サービス要求をユニバーサル マッチング サービスに送る方法の例を示します。

1. Enterprise Designer で、新しいサービス データフローを作成します。
2. Input ステージ、Transactional Match ステージ、および Output ステージをキャンバスにドラッグし、連結して以下のようなデータフローを作成します。



3. Transactional Match ステージをダブルクリックします。

4. **[ロードするマッチ ルール]** フィールドで、マッチ ルールを選択します。例えば、デフォルトの **[Household]** マッチ ルールを選択します。

   マッチ ルールはサービス要求で指定しますが、Transactional Match ステージではデータフローを有効にするためにデフォルトのマッチ ルールを設定する必要があります。マッチ ルールを選択しないと、データフローが検証で無効と判定され、エクスポーズできません。

5. **[OK]** をクリックします。
6. Output ステージをダブルクリックします。
7. フィールド `MatchRecordType` と`MatchScore` を選択してエクスポーズします。
8. **[OK]** をクリックします。

   注： Input ステージではフィールドをエクスポーズする必要はありません。入力フィールドは、ユーザ定義フィールドとしてサービス要求で指定するからです。

9. **[編集] > [データフロー オプション]** をクリックします。
10. **[追加]** をクリックします。
11. **[Transactional Match]** を展開し、**[Match Rule]** の横にあるボックスをオンにします。

    これでマッチ ルール オプションが実行オプションとしてエクスポーズされ、マッチ ルールをサービス要求で指定できるようになります。

12. **[OK]** をクリックし、再び **[OK]** をクリックして **[データフロー オプション]** ウィンドウを閉じます。
13. データフローを保存してエクスポーズします。

これで、ユニバーサル マッチング サービスは完成です。Enterprise Designer の [マッチ ルール管理] で定義したマッチ ルールを使って、マッチングの実行に使用できます。サービスを呼び出すときに、マッチ ルールを `MatchRule` オプションに指定し、入力フィールドをユーザ定義フィールドとして指定します。

例: ユニバーサル マッチング サービスの呼び出し

[マッチ ルール管理] で、マッチ ルールを `AddressAndBirthday` という名前で作成しました。このマッチ ルールは、Address フィールドと Birthday フィールドがあるレコードにマッチングします。ユニバーサル マッチング サービスを使って、SOAP Web サービス要求でこのルールに従ってマッチングを実行することを考えています。

これを行うには、SOAP 要求の MatchRule 要素に AddressAndBirthday を指定し、レコードのフィールドを user_fields 要素に指定します。

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:univ="http://www.pb.com/spectrum/services/UniversalMatchingService">

   <soapenv:Header/>
   <soapenv:Body>
      <univ:UniversalMatchingServiceRequest>
         <univ:options>

<univ:MatchRule>AddressAndBirthday</univ:MatchRule>
         </univ:options>
         <univ:Input>
            <univ:Row>
               <univ:user_fields>
                  <univ:user_field>
                     <univ:name>Name</univ:name>
                     <univ:value>Bob Smith</univ:value>
                  </univ:user_field>
                  <univ:user_field>
                     <univ:name>Address</univ:name>
                     <univ:value>4200 Parliament
Pl</univ:value>
                  </univ:user_field>
                  <univ:user_field>
                     <univ:name>Birthday</univ:name>
                     <univ:value>1973-6-15</univ:value>
                  </univ:user_field>
               </univ:user_fields>
            </univ:Row>
            <univ:Row>
               <univ:user_fields>
                  <univ:user_field>
                     <univ:name>Name</univ:name>
                      <univ:value>Robert M. Smith</univ:value>

                  </univ:user_field>
                  <univ:user_field>
                     <univ:name>Address</univ:name>
                     <univ:value>4200 Parliament
Pl</univ:value>
                  </univ:user_field>
                  <univ:user_field>
                     <univ:name>Birthday</univ:name>
                     <univ:value>1973-6-15</univ:value>
                  </univ:user_field>
               </univ:user_fields>
            </univ:Row>
            <univ:Row>
```

```
                <univ:user_fields>
                   <univ:user_field>
                      <univ:name>Name</univ:name>
                      <univ:value>Bob Smith</univ:value>
                   </univ:user_field>
                   <univ:user_field>
                      <univ:name>Address</univ:name>
                      <univ:value>424 Washington
Blvd</univ:value>
                   </univ:user_field>
                   <univ:user_field>
                      <univ:name>Birthday</univ:name>
                      <univ:value>1959-2-19</univ:value>
                   </univ:user_field>
                </univ:user_fields>
             </univ:Row>
          </univ:Input>
       </univ:UniversalMatchingServiceRequest>
    </soapenv:Body>
</soapenv:Envelope>
```

この要求を発行すると、以下のような応答が得られます。

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns3:UniversalMatchingServiceResponse
xmlns:ns2="http://spectrum.pb.com/"

xmlns:ns3="http://www.pb.com/spectrum/services/UniversalMatchingService">

          <ns3:Output>
             <ns3:Row>
                <ns3:MatchScore/>

<ns3:MatchRecordType>Suspect</ns3:MatchRecordType>
                <ns3:user_fields>
                   <ns3:user_field>
                      <ns3:name>Name</ns3:name>
                      <ns3:value>Bob Smith</ns3:value>
                   </ns3:user_field>
                   <ns3:user_field>
                      <ns3:name>Birthday</ns3:name>
                      <ns3:value>1973-6-15</ns3:value>
                   </ns3:user_field>
                   <ns3:user_field>
                      <ns3:name>Address</ns3:name>
                       <ns3:value>4200 Parliament Pl</ns3:value>

                   </ns3:user_field>
                </ns3:user_fields>
```

```
            </ns3:Row>
            <ns3:Row>
               <ns3:MatchScore>100</ns3:MatchScore>

<ns3:MatchRecordType>Duplicate</ns3:MatchRecordType>
               <ns3:user_fields>
                  <ns3:user_field>
                     <ns3:name>Name</ns3:name>
                     <ns3:value>Robert M. Smith</ns3:value>
                  </ns3:user_field>
                  <ns3:user_field>
                     <ns3:name>Birthday</ns3:name>
                     <ns3:value>1973-6-15</ns3:value>
                  </ns3:user_field>
                  <ns3:user_field>
                     <ns3:name>Address</ns3:name>
                      <ns3:value>4200 Parliament Pl</ns3:value>

                  </ns3:user_field>
               </ns3:user_fields>
            </ns3:Row>
         </ns3:Output>
      </ns3:UniversalMatchingServiceResponse>
   </soap:Body>
</soap:Envelope>
```

Express マッチ キーの使用

Express キー マッチは、比較の実行回数を減らし、Interflow Match または Intraflow Match ステージを使用するデータフローの実行速度を改善するのに便利なツールです。2つのレコードがExpress キーに完全一致する場合、候補は 100% マッチと見なされ、それ以上の照合は試行されません。2 つのレコードが Express キー値にマッチしない場合は、ルール ベースの方法で比較されます。ただし、あいまいな Express キーを使うと、誤検出のマッチが多数返されます。

1. Enterprise Designer でデータフローを開きます。
2. Match Key Generator ステージをダブルクリックします。
3. **[Express マッチ キーを生成]** ボックスをオンにします。
4. **[追加]** をクリックします。
5. 以下のフィールドに必要な情報を入力します。

表 6 : Match Key Generator のオプション

オプション名	説明 / 有効な値
アルゴリズム	

オプション名	説明 / 有効な値
	マッチ キーの生成に使用するアルゴリズムを指定します。次のいずれかです。
	Consonant (子音) 指定されたフィールドを、子音を削除して返します。
	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。
	KoeIn 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。

オプション名	説明 / 有効な値
	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。 **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** 選択されているフィールドの指定部分を返します。
フィールド名	選択したアルゴリズムを適用してマッチ キーを生成するフィールドを指定します。例えば、LastName というフィールドを選択し、Soundex アルゴリズムを選択した場合、Soundex アルゴリズムが LastName フィールドのデータに適用されて、マッチ キーが生成されます。
開始位置	指定したフィールド内での開始位置を指定します。すべてのアルゴリズムで開始位置を指定できるとは限りません。
長さ	開始位置から含める文字の数を指定します。すべてのアルゴリズムで長さを指定できるとは限りません。
ノイズ文字の削除	ハイフン、空白、その他の特殊文字等、英数字以外の文字を入力フィールドからすべて削除します。
ソート入力	入力フィールド内の文字または語をすべてアルファベット順にソートします。 **文字** ユニーク ID を作成する前に、入力フィールドの文字値をソートします。 **語** ユニーク ID を作成する前に、入力フィールドの各語値をソートします。

6. **[OK]** をクリックします。
7. Express マッチ キーの生成で使用するフィールドやアルゴリズムを追加で指定する場合は **[追加]** をクリックし、それ以外の場合は **[OK]** をクリックします。

8. キャンバスで Interflow Match または Intraflow Match ステージをダブルクリックします。
9. **[Express マッチ有効]** オプションを選択し、**[ExpressMatchKey]** フィールドを選択します。

   このフィールドには、Match Key Generator によって生成された Express マッチ キーが含まれます。

10. **[OK]** をクリックします。
11. データフローを保存して実行します。

Express キーを使った比較で候補がマッチしたかどうかを確認するには、**[ExpressKeyIdentified]** フィールドの値をチェックします。"Y" はマッチしたことを示し、"N" はマッチしなかったことを示します。サスペクト レコードは **[ExpressKeyIdentified]** 値が常に "N" であることに注意してください。

マッチ結果の分析

Enterprise Designer のマッチ分析ツールは、同じタイプの 1 つ以上のマッチング ステージの結果を表示します。 このツールは、データフローでのマッチ結果のサマリを提供します。また、このツールを使用すると、マッチ結果をレコードごとに表示することもできます。 この情報を使用すると、マッチ ルールをトラブルシューティングしたり、希望どおりの結果が得られるように微調整したりできます。

マッチ分析ツールには以下の機能があります。

- マッチ サマリ結果: 単一のマッチ結果または 2 つのマッチ結果の比較によるサマリ レコード数を表示します。
- リフト/ドロップ グラフ: 棒グラフを使用して、マッチの増減を表示します。
- マッチ ルール: 単一のマッチ結果に使用するマッチ ルール、または 2 つのマッチ結果を比較するときにマッチ ルールに加えられる変更を表示します。
- マッチ詳細結果: 単一のマッチ結果または 2 つのマッチ結果の比較によるレコード処理の詳細情報を表示します。

マッチ結果のサマリの表示

マッチ分析ツールは、重複レコードの数や平均マッチ スコアなど、データフローのマッチング プロセスに関するサマリ情報を表示できます。単一のジョブの結果を表示したり、複数のジョブの結果を比較したりできます。

1. Enterprise Designer で、分析するデータフローを開きます。
2. 分析するマッチングがある各 Interflow Match、Intraflow Match、または Transactional Match ステージでは、ステージをダブルクリックし、**[分析用データを生成する]** チェックボックスをオンにします。

   重要： **[分析用データを生成する]** オプションを有効にすると、パフォーマンスが低下します。このオプションは、マッチ分析ツールの終了後にオフにする必要があります。

3. **[実行] > [現在のフローを実行]** を選択します。

   注： 最適な結果を得るには、レコードの生成数が 100,000 件以下のデータを使用します。マッチ結果が多くなると、マッチ分析ツールのパフォーマンスが低下します。

4. データフローの実行が終了したら、**[ツール] > [マッチ分析]** を選択します。

   [マッチ結果を表示] ダイアログ ボックスに、マッチ分析ツールに表示されたマッチ結果を持つデータフローの一覧が表示されます。分析対象のジョブが一覧に表示されない場合は、データフローを開き、マッチング ステージの **[分析用データを生成する]** チェック ボックスがオンになっていることを確認します。

   ヒント： データフローが大量にあり、データフローをフィルタリングする場合は、**[次のジョブのみ表示]** ドロップダウン リストからフィルタ オプションを選択します。

5. 表示するデータフローの横にある "+" アイコンをクリックして展開します。
6. データフローの下に、データフローの各マッチャー ステージのエントリが1つあります。結果を表示するステージを選択し、**[追加]** をクリックします。

   マッチ分析ツールが Enterprise Designer ウィンドウの下部に表示されます。

7. マッチャーの結果を横に並べて別のマッチャーの結果と比較する場合:
   a) **[追加]** をクリックします。
   b) 結果を比較するマッチャーを選択します。
   c) **[追加]** をクリックします。
   d) データフローの一覧で、追加したばかりのマッチャーを選択し、**[比較]** をクリックします。

[サマリ] タブに、ジョブのマッチング統計情報が表示されます。表示される情報は、データフローで使用されるマッチング ステージのタイプによって異なります。

Intraflow Match の場合、以下のサマリ情報が表示されます。

入力レコード	マッチャー ステージで処理されるレコードの合計数。
ユニーク レコード	マッチ グループで他のレコードにマッチしないサスペクトまたは候補レコード。マッチ グループ内に1つしか存在していないレコードであれば、サスペクトは自動的にユニーク レコードとなります。

マッチ グループ	(グループ化方法) マッチ キーまたはスライディング ウィンドウでグループ化されたレコード。
重複コレクション	サスペクトとその重複レコードがコレクション番号によってグループ化されたもの。ユニーク レコードは常にコレクション番号 0 とされます。
Express マッチ	サスペクトと候補が指定されたフィールド内の内容に正確にマッチした場合に作成されるもののこと。通常は ExpressMatchKey が Match Key Generator によって提供されます。式マッチがそれ以上の処理がされず終了した場合、サスペクトと候補は重複していると判断できます。
平均スコア	すべての重複の平均マッチ スコア。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。

Interflow Match の場合、以下のサマリ情報が表示されます。

重複コレクション	サスペクトとその重複レコードがコレクション番号によってグループ化されたもの。ユニーク レコードは常にコレクション番号 0 とされます。
Express マッチ	サスペクトと候補が指定されたフィールド内の内容に正確にマッチした場合に作成されるもののこと。通常は ExpressMatchKey が Match Key Generator によって提供されます。式マッチがそれ以上の処理がされず終了した場合、サスペクトと候補は重複していると判断できます。
平均スコア	すべての重複の平均マッチ スコア。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。
入力サスペクト	マッチャーが他のレコードとの照合を試みた入力ストリーム内のレコードの数。
[サスペクトと重複]	少なくとも 1 つの候補レコードと一致した入力サスペクトの数。
[ユニーク サスペクト]	どの候補レコードとも一致しなかった入力サスペクトの数。
[サスペクトと候補]	マッチ グループ内に候補レコードが少なくとも 1 つある、つまり照合の試みが少なくとも 1 回は行われた入力サスペクトの数。
[候補がないサスペクト]	マッチ グループ内に候補レコードがない、つまり照合の試みが行われなかった入力サスペクトの数。

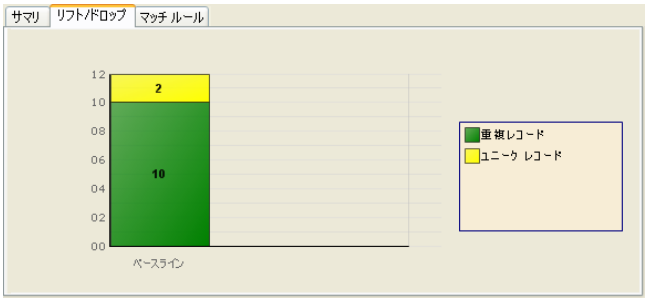
Transactional Match の場合、以下のサマリ情報が表示されます。

平均スコア	すべての重複の平均マッチ スコア。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。
入力サスペクト	マッチャーが他のレコードとの照合を試みた入力ストリーム内のレコードの数。
[サスペクトと重複]	少なくとも 1 つの候補レコードと一致した入力サスペクトの数。
[ユニーク サスペクト]	どの候補レコードとも一致しなかった入力サスペクトの数。
[サスペクトと候補]	マッチ グループ内に候補レコードが少なくとも 1 つある、つまり照合の試みが少なくとも 1 回は行われた入力サスペクトの数。

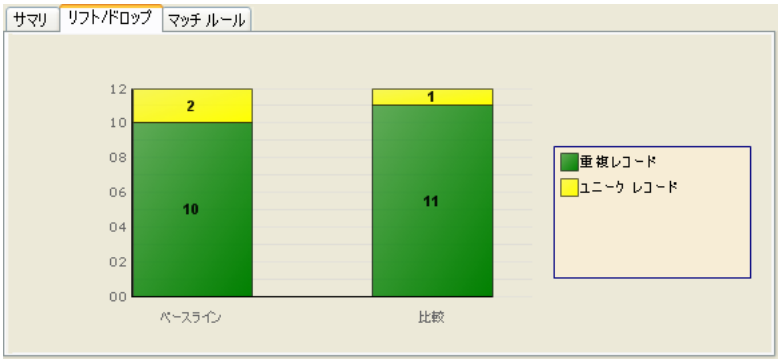
[候補がないサスペクト] マッチ グループ内に候補レコードがない、つまり照合の試みが行われなかった入力サスペクトの数。

マッチ分析ツールの**[リフト/ドロップ]**タブには、選択したベースライン結果と、オプションで選択した比較結果について、重複レコードとユニーク レコードの数が棒グラフで表示されます。*リフト*は、重複レコード数の増加を意味します。*ドロップ*は、重複レコード数の減少を意味します。ユニーク レコードは黄色で、重複レコードは緑で示されます。

ベースライン ジョブのみを選択している場合、グラフにはそのジョブの結果が表示されます。



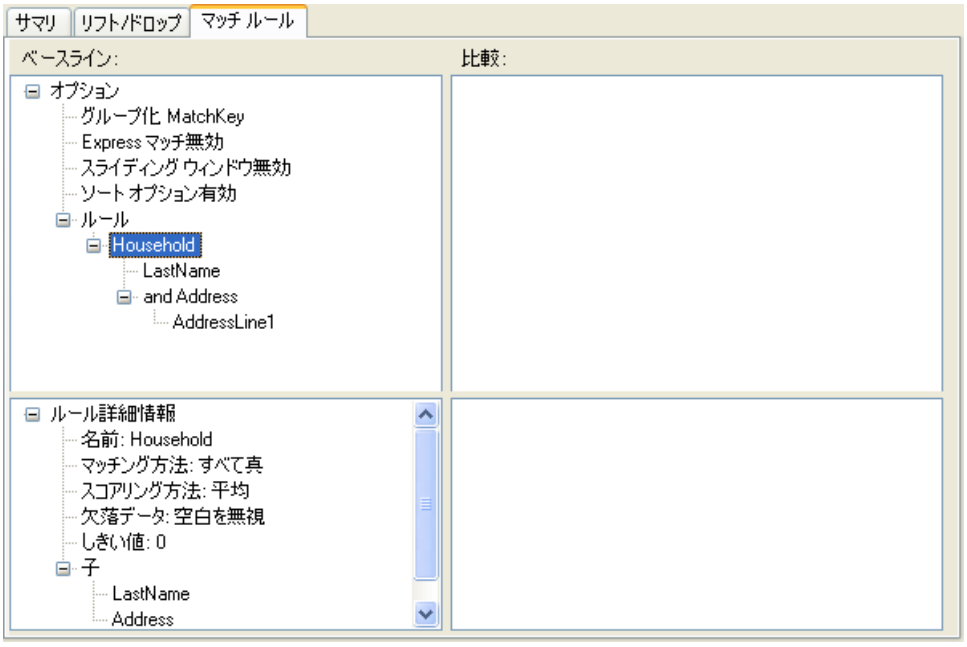
ベースライン ジョブと比較ジョブの両方を選択している場合は、その両方のジョブを表すグラフが横に並んで表示されます。



マッチ分析ツールの **[マッチ ルール]** タブには、単一のマッチ結果に使用するマッチ ルール、または 2 つのマッチ結果を比較するときにマッチ ルールに加える変更が表示されます。

マッチ ルールは階層構造で表示されます。この階層構造はマッチ ルールを作成したステージの階層構造と同じです。ルール階層には、[オプション] と [ルール] の 2 つのノードがあります。[オプション] ノードには、選択したマッチ結果のステージ設定が表示されます。[ルール] ノードには、選択したマッチ結果のマッチ ルールが表示されます。

ルール詳細情報を表示するには、階層内のノードを選択します。



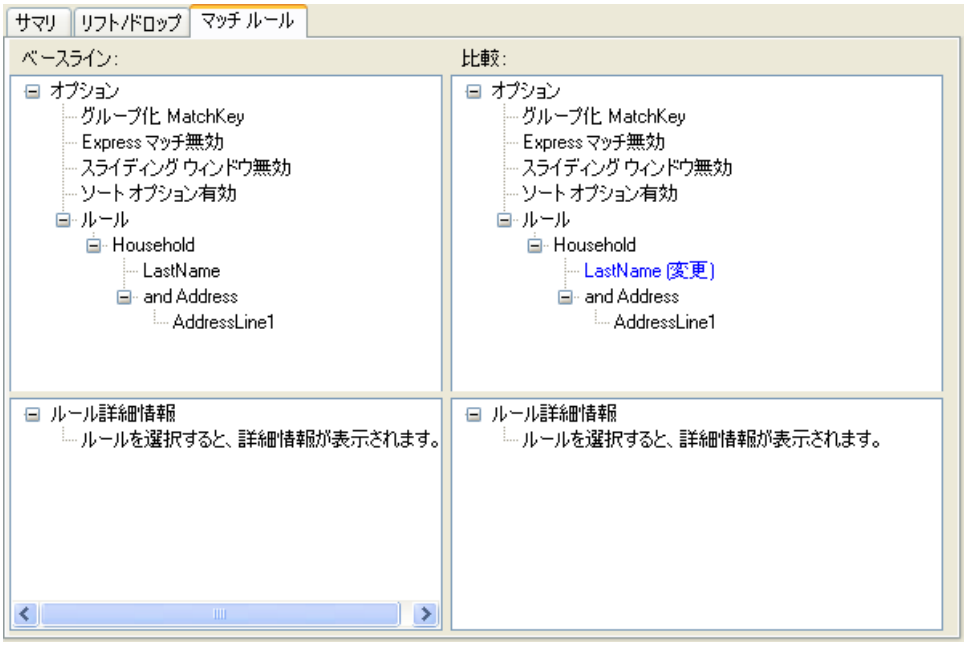
複数のジョブ間でマッチ ルールを比較する場合、ベースライン マッチ結果と比較マッチ結果との相違点は、以下のように色分けして表示されます。

青 比較マッチ結果のマッチ ルールが変更されたことを示します。

緑 比較マッチ結果のマッチ ルールが追加されたことを示します。

赤 比較マッチ結果のマッチ ルールが省略されたことを示します。

例:



レコードレベルのマッチ結果の表示

詳細表示には、マッチ結果セットのマッチ レコードに関する情報が詳しく表示されます。

詳細結果を表示するには:

1. マッチ分析ツールで、ベースライン ジョブと、オプションで比較ジョブを指定します。
2. **[詳細]** をクリックします。

   [表示] ドロップダウン リストで選択したビューに基づいて、ベースライン マッチ結果が表示されます。 以下の表に、各マッチ ステージ タイプで表示される列を示します。

表 7 : 表示される詳細結果データ

詳細関連の結果	Intraflow	Interflow	Transactional
入力レコード番号	X	X	X
マッチ グループ	X	X	
Express キー	X	X	
Express キー ドライバ レコード	X	X	
コレクション番号	X	X	X
マッチ レコード タイプ	X	X	X
ルールで使用するフィールド	X	X	X
全体 (トップレベル) ルール スコア	X		
候補グループ		X	X

詳細関連の結果	Intraflow	Interflow	Transactional
マッチ スコア マッチ結果リストのマッチ結果を選択し、**[削除]** をクリックします。		X	X

マッチ率グラフについては、マッチ率グラフ（125ページ）を参照してください。

3. **[分析]** フィールドで、以下のいずれかを選択します。

   ベースライン ベースライン実行のマッチ結果を表示します。

   [比較] 比較実行のマッチ結果を表示します。

4. **[表示]** リストから次のいずれかの値を選択し、**[更新]** をクリックします。 ベースライン結果を分析する場合は、以下のオプションを使用できます。

   - [サスペクトと候補]: (すべてのマッチャー) サスペクト レコードと、各サスペクトとの照合を試みる候補レコードをすべて表示します。[サスペクトと候補]
   - [サスペクトと重複]: (すべてのマッチャー) すべてのサスペクト レコードと、各サスペクトと照合された候補レコードを表示します。[サスペクトと重複]
   - [サスペクトと Express マッチ]: (Express マッチ キーが有効時の Interflow Match と Intraflow Match) Express マッチ キーに基づいて照合されるサスペクト レコードと候補レコードを表示します。[サスペクトと Express マッチ]
   - [重複コレクション]: (Intraflow と Interflow) すべての候補コレクションをコレクション番号別に表示します。[重複コレクション]
   - [マッチ グループ]: (Intraflow と Interflow) レコードをマッチ グループ別に表示します。[マッチ グループ]
   - [候補グループ]: (Transactional Match) レコードを候補グループ別に表示します。
   - [ユニーク サスペクト]: (Interflow および Transactional Match) どの候補レコードともマッチしないサスペクト レコードをすべて表示します。
   - [ユニーク レコード]: (Intraflow) マッチしないレコードをすべて表示します。
   - [候補がないサスペクト]: (Interflow および Transactional Match) 照合する候補がないサスペクトをすべて表示します。
   - [すべてのレコード]: 処理されたレコードをマッチング ステージ別にすべて表示します。

   比較結果を分析する場合は、以下の表示オプションを使用できます。

   - [新しいマッチ]: (Intraflow) 新しいマッチと、それに関連するサスペクトをすべて表示します。このビューでは、新しい重複のあるサスペクトと、新しいサスペクトが 1 つのビューに表示されます。

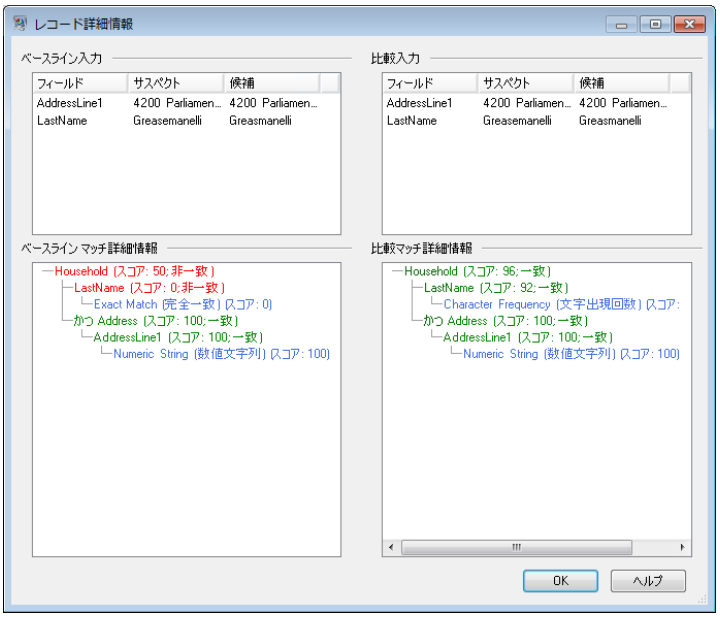
- [新しくマッチしたサスペクト]: (Interflow および Transactional Match) ベースラインには重複を持たないが、比較には少なくとも 1 つの重複を持つサスペクトを表示します。
- [新しいユニーク サスペクト]: (Interflow および Transactional Match) ベースラインには重複を持つが、比較には重複を持たないサスペクトを表示します。
- [マッチでなくなったもの]: (Intraflow) マッチでなくなったものをすべて表示します。 このビューでは、重複がなくなったサスペクトと、サスペクトでなくなったものが 1 つのビューに表示されます。
- [新しい重複のあるサスペクト]: (すべてのマッチャー) ベースラインでサスペクトで、比較でもサスペクトのままのレコードに対して新しい重複となるレコードを表示します。
- [重複がなくなったサスペクト]: (すべてのマッチャー) ベースラインでサスペクトで、比較でもサスペクトのままのレコードに対して重複でなくなったレコードを表示します。
- [新しいサスペクト]: (Intraflow) 比較マッチ結果ではサスペクトであるが、ベースラインではサスペクトでなかったレコードを表示します。
- [サスペクトでなくなったもの] (Intraflow) 比較結果ではサスペクトでないが、ベースラインではサスペクトであったレコードを表示します。

5. サスペクト レコードを展開して、その候補を表示します。
6. 候補レコードを選択し、**[詳細]** をクリックします。

   注 : このオプションは、Intraflow Match ステージでスライディング ウィンドウが有効になっている場合は使用できません。

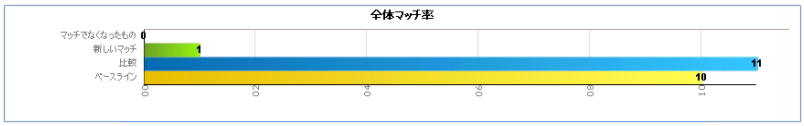
[レコード詳細情報] ウィンドウには、フィールドレベルのデータと、各マッチ ルールに対するレコードのマッチ スコアが表示されます。 ベースライン ジョブと比較ジョブの両方の実行を指定した場合は、その両方についてレコードの結果を確認できます。

- [ベースライン入力] — マッチで使用するフィールド レベルのデータがサスペクトと候補の両方について表示されます。
- [ベースライン マッチ詳細情報] — マッチ ルールの各ノードのスコアリング情報が表示されます。
- [比較入力] — マッチで使用するフィールド レベルのデータがサスペクトと候補の両方について表示されます。
- [比較マッチ詳細情報] — マッチ ルールの各ノードのスコアリング情報が表示されます。 緑のテキストは、ルール内でマッチするノードを表します。 赤のテキストは、ルール内でマッチしないノードを表します。



マッチ率グラフ

マッチ率グラフは、詳細ビューにマッチ情報をグラフィカルに表示します。



Intraflow Match の場合、1 つのグラフにマッチ全体が表示されます。

- [ベースライン マッチ]: ベースライン結果の総マッチ数。
- [比較マッチ]: 比較結果の総マッチ数。
- [新しいマッチ]: ベースライン結果ではユニークであったが、比較結果ではサスペクトまたは重複になるレコードの総数。
- [マッチでなくなったもの]: ベースライン結果ではサスペクトまたは重複であったが、比較結果ではユニークになるレコードの総数。

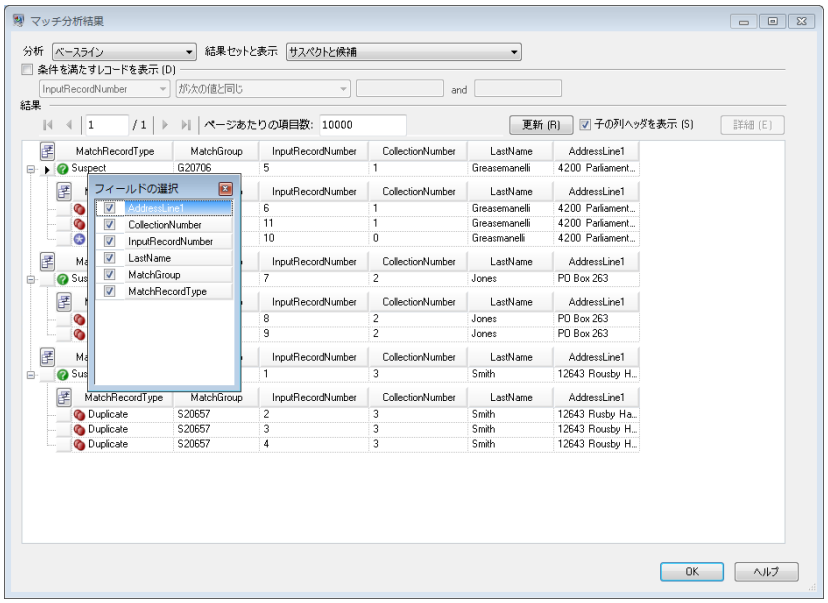
Interflow および Transactional Match の場合、2 つのグラフが表示されます。

- [全体マッチ率]
- [ベースライン マッチ]: ベースライン結果の総マッチ数。
- [比較マッチ]: 比較結果の総マッチ数。
- [新しいマッチ]: ベースライン結果ではユニークであったが、比較結果ではサスペクトまたは重複になるレコードの総数。
- [マッチでなくなったもの]: ベースライン結果ではサスペクトまたは重複であったが、比較結果ではユニークになるレコードの総数。

- [サスペクト マッチ率]
- [ベースライン マッチ]: ベースラインでユニークでなかったサスペクトの総数。
- [比較マッチ]: 比較でユニークでなかったサスペクトの総数。
- [新しいマッチ]: ベースラインではユニークであったが、比較結果ではマッチになるサスペクトの総数。
- [マッチでなくなったもの]: ベースラインではマッチであったが、比較結果ではユニークになるサスペクトの総数。

[フィールドの選択] の使用

[フィールドの選択] アイコン  をクリックして、選択した列をマッチ分析結果に表示します。 [フィールドの選択] は、親レベルと子レベルに表示されます。 表示する列は、親と子で別々に選択できます。



レコードのフィルタリング

[条件を満たすレコードを表示] チェック ボックスを使用して、表示する詳細マッチ レコードをフィルタリングします。 いくつかの演算子に基づいてレコードをフィルタリングして、ユーザが提供する値を、各詳細マッチ レコードの特定のフィールド内のデータと比較できます。

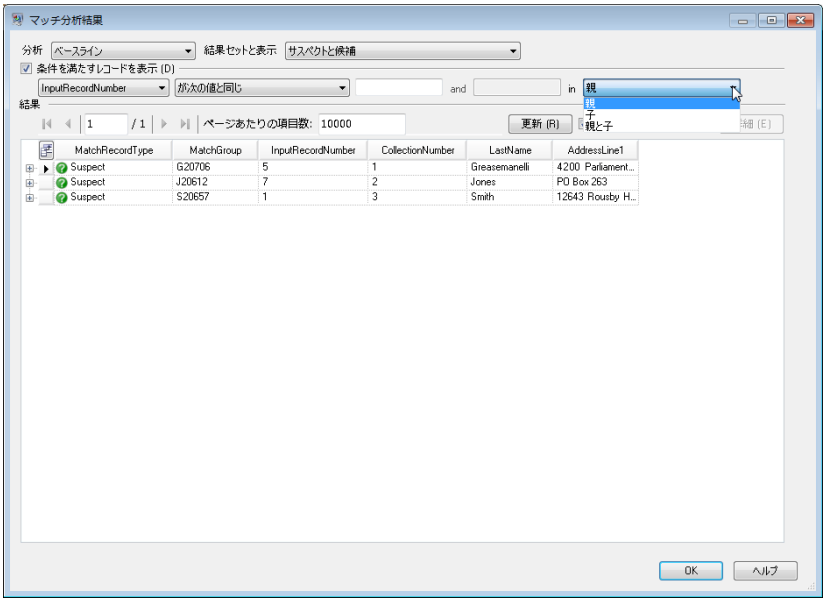
次の演算子を選択できます。

- 文字列タイプ フィールド (マッチグループ、マッチレコードタイプ、任意のマッチング データ)
- が次の文字を含む
- が次の値の間
- が次の値と同じ

- が次の値と異なる
- が次の文字から始まる
- 数値タイプ フィールド (コレクション番号、入力レコード番号、マッチスコア)
- が次の値の間
- が次の値と同じ
- が次の値と異なる
- が次の値より大きい
- が次の値以上
- が次の値より小さい
- が次の値以下

レコードをフィルタリングするには

1. [マッチ分析結果] ビューから、ベースラインまたは比較マッチ結果を選択し、**[更新]** をクリックします。
2. **[条件を満たすレコードを表示]** チェック ボックスを選択します。



3. [フィールド] リスト ボックスから、フィールドを選択します。
4. 演算子を選択します。
5. 選択した演算子タイプの値を入力します。**[が次の値の間]** を選択する場合は、値の範囲を入力します。
6. サスペクト ビューでフィルタリングする場合は、次のフィルタリングを実行できます。
   - [親] — 親 (サスペクト) のみでフィルタリングします。すべての子が返されます。
   - [子] — フィルタリング範囲に含まれない子をすべて除外します。 親 (サスペクト) ノードが返されます。

- [親と子] — 親 (サスペクト) でフィルタリングします。このとき、親が返される場合は、その子でフィルタリングします。

7. **[更新]** をクリックします。 オプションと値の範囲に含まれるレコードが表示されます。 選択したオプションと値の範囲内にレコードが含まれない場合は、該当するレコードがないことを示すメッセージが表示されます。

マッチ ルールの変更の分析

Enterprise Designer のマッチ分析ツールを使用すると、マッチ ルールの変更がデータフローのマッチ結果に与える効果を詳細に表示できます。 この操作を行うには、データフローを実行し、変更を行った後、データフローを再度実行して、マッチ分析ツールに結果を表示します。 この手順は、マッチ結果に与える効果の表示方法を示しています。

重要： マッチ結果を比較する場合、ベースライン実行と比較実行で同じ入力データを使用する必要があります。 異なる入力データを使用すると、本来の結果が得られない可能性があります。 正確な比較を行うには、以下の点に従います。

- 同じ入力ファイルまたはテーブルを使用します。
- マッチング ステージの前にデータを同じ方法でソートします。
- Transactional Match を使用するときは、同じ Candidate Finder クエリを使用します。

1. Enterprise Designer で、分析するデータフローを開きます。
2. 分析するマッチングがある各 Interflow Match、Intraflow Match、または Transactional Match ステージでは、ステージをダブルクリックし、**[分析用データを生成する]** チェックボックスをオンにします。

   重要： **[分析用データを生成する]** オプションを有効にすると、パフォーマンスが低下します。このオプションは、マッチ分析ツールの終了後にオフにする必要があります。

3. **[実行] > [現在のフローを実行]** を選択します。

   注： 最適な結果を得るには、レコードの生成数が 100,000 件以下のデータを使用します。マッチ結果が多くなると、マッチ分析ツールのパフォーマンスが低下します。

4. データフローのマッチャー ステージで、マッチ ルールを希望どおりに変更した後、データフローを再度実行します。

   例えば、しきい値を増やした場合の効果をテストするには、しきい値を変更した後、データフローを再度実行します。

5. データフローの実行が終了したら、**[ツール] > [マッチ分析]** を選択します。

[マッチ結果を表示] ダイアログ ボックスに、マッチ分析ツールに表示されたマッチ結果を持つデータフローの一覧が表示されます。分析対象のジョブが一覧に表示されない場合は、データフローを開き、マッチング ステージの **[分析用データを生成する]** チェック ボックスがオンになっていることを確認します。

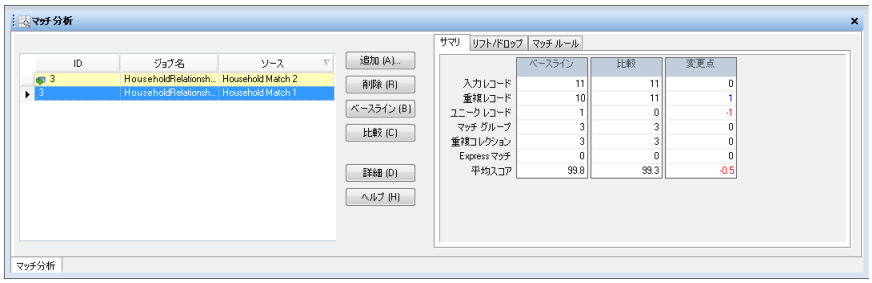
ヒント： データフローが大量にあり、データフローをフィルタリングする場合は、**[次のジョブのみ表示]** ドロップダウン リストからフィルタ オプションを選択します。

6. [マッチ分析] ウィンドウの左側に、マッチャー ステージの一覧があります。マッチャー ステージでは 1 回の実行で 1 つ使用します。比較のベースラインとして使用する、実行のマッチャー ステージを選択し、**[ベースライン]** をクリックします。次に、そのベースラインと比較する実行を選択し、**[比較]** をクリックします。

これで、サマリ マッチ結果を比較できます。例えば、重複レコードの合計数や、マッチ ルールに対する各レコードの評価方法を示すレコードレベルの詳細情報を比較できます。

マッチ結果の比較の例

例えば、HouseholdRelationshipsAnalysis というジョブを実行するとします。Household Match 2 ステージに加える変更の効果を確認します。 最初に、元の設定を使用してジョブを実行します。次に、Household Match 2 ステージでマッチ ルールを変更した後、ジョブを再度実行します。 マッチ分析ツールで、元の設定を持つジョブ ID 10 の実行をベースラインに設定します。 ジョブ ID 13 は、変更後のマッチ ルールを使用した実行です。 **[比較]** をクリックすると、変更後のマッチ ルール (ジョブ ID 13) では元のマッチ ルールより重複レコードの数が 1 つ増加し、ユニーク レコードの数が 1 つ減少しているのがわかります。



マッチ結果の追加

マッチ分析ツールのオープン中にジョブを実行し、マッチ結果リストが空の場合、マッチ結果はリストに自動的に追加されます。マッチ結果が追加された後、マッチ分析ツールは、同じマッチ タイプ (Interflow Match、Intraflow Match、または Transactional Match) のマッチ結果のみを追加します。

マッチ分析ツールで現在選択されているタイプとは異なるタイプのマッチ結果を分析する場合は、以下の手順に従います。

1. マッチ結果リストのすべてのマッチ結果を選択し、**[削除]** をクリックします。
2. 異なるマッチング ステージを使用するジョブを Server Explorer から開くか、ジョブが既に開かれている場合は、キャンバスの上のタブをクリックします。
3. ジョブを実行します。

ジョブの実行が完了したら、最後のジョブ インスタンスのマッチ結果がマッチ結果リストに追加されます。

マッチ結果の削除

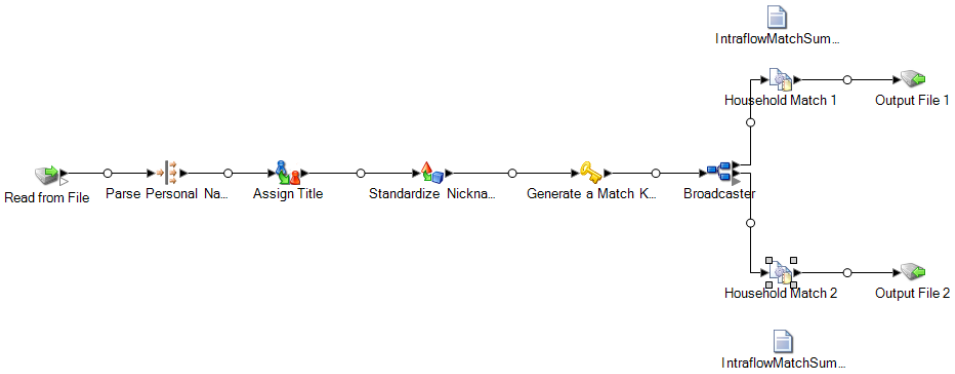
マッチ結果リストからマッチ結果を削除するには、マッチ結果リストでマッチ結果を選択し、**[削除]** をクリックします。

マッチ結果リストと [サマリ] タブは、次のように更新されます。

- 削除するマッチ結果がベースラインまたは比較マッチ結果のどちらでもない場合、マッチ結果が削除されます。[サマリ] タブに変更はありません。
- 削除するマッチ結果がベースラインとして設定されていた場合、次に古いマッチ結果が新しいベースラインとして設定され、[サマリ] タブが更新されて、新しいベースライン データのみが表示されます。
- 削除するマッチ結果が比較マッチ結果として設定されていた場合、[サマリ] タブが更新されて、既存のベースライン データのみが表示されます。
- 削除するマッチ結果がマッチ結果リストに表示されている 2 つのマッチ結果のどちらかの場合、残ったマッチ結果が新しいベースラインとして設定され、[サマリ] タブが更新されて、新しいベースライン データのみが表示されます。

例: マッチ分析の使用

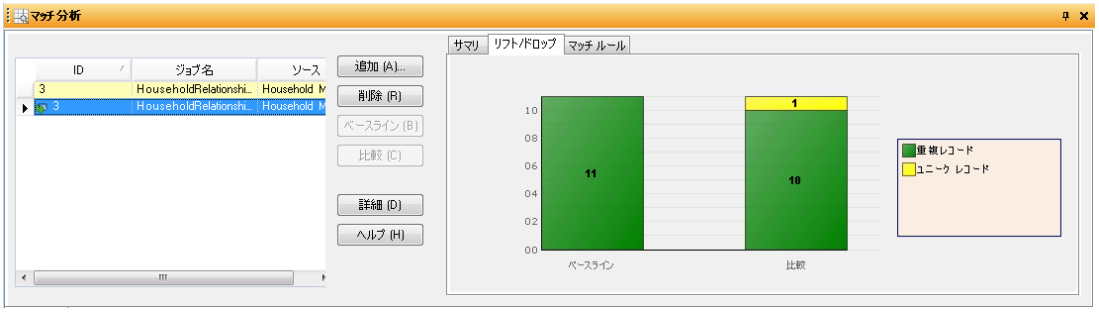
この例では、マッチ分析ツールを使用して 2 種類のマッチのリフト/ドロップ率を比較する方法を示します。マッチャーを通じてデータを送信する前に、Broadcaster を使ってデータを 2 つのストリームに分割します。その後、それぞれのストリームを Intraflow Match ステージを通じて送信します。各データ ストリームには、同一の処理済みデータが含まれます。各 Intraflow Match ステージでは、異なるマッチング アルゴリズムを使い、さまざまなマッチのリフト/ドロップに使えるマッチ分析データが生成されます。



このサンプル データフローは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[HouseholdRelationshipsAnalysis]** を選択します。このデータフローでは、Advanced Matching モジュール、Data Normalization モジュール、および Universal Name モジュールが必要です。また、このデータフローでは、Table Lookup Core データベースと Open Parser Base テーブルをロードする必要もあります。

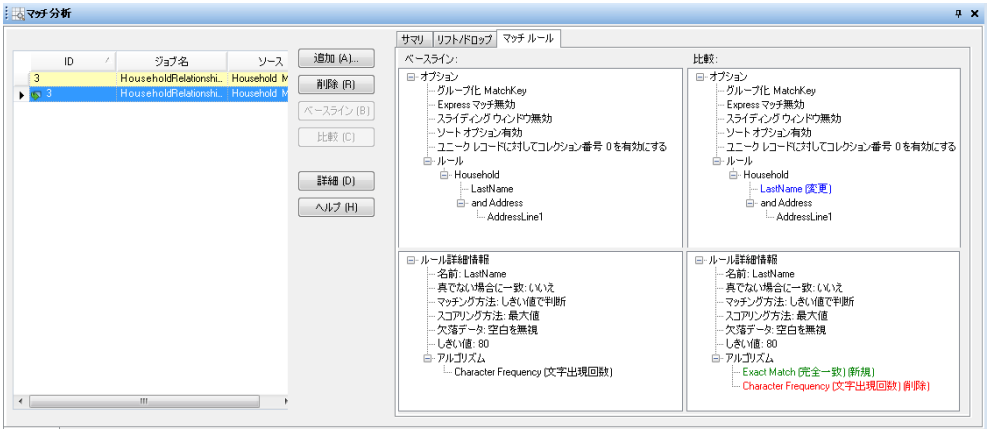
この例を使用するには:

1. データフローを実行します。
2. **[ツール] > [マッチ分析]** を選択します。
3. **[マッチ結果を表示]** ウィンドウで **[HouseholdRelationshipAnalysis]** を展開し、[ソース] リストから **[Household Match 1]** と **[Household Match 2]** を選択してから **[追加]** をクリックします。
4. [マッチ結果] リストで **[Household Match 1]** を選択し、**[比較]** をクリックします。サマリ結果が表示されます。
5. **[リフト/ドロップ]** タブをクリックします。リフト/ドロップ グラフが表示されます。



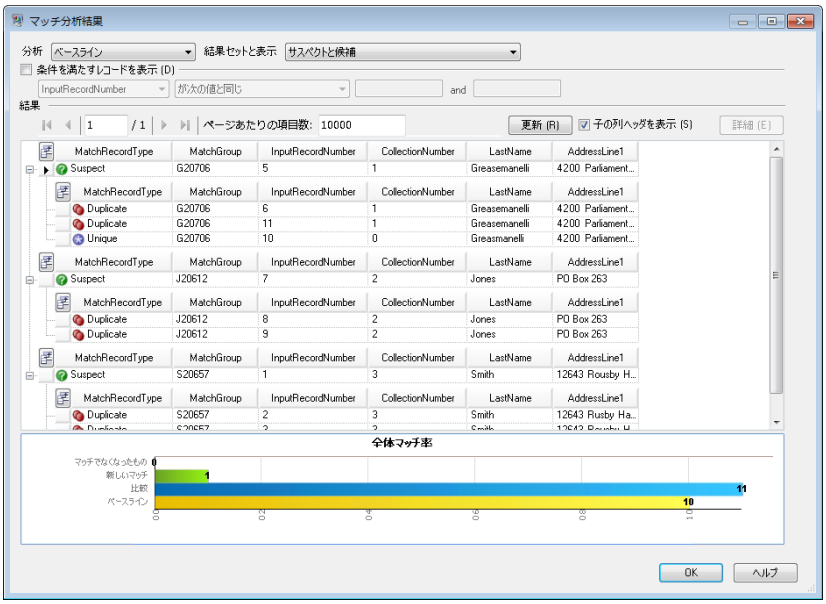
このグラフには、異なるマッチ ルールを使って生成された重複レコードとユニーク レコードとの相違が示されます。

6. **[マッチ ルール]** タブをクリックします。マッチ ルールの比較が表示されます。



このタブからは、アルゴリズムが変更されたことが読み取れます。[Character Frequency (文字出現回数)] がオフになり、[Exact Match (完全一致)] が追加されています。

7. **[詳細]** をクリックします。
8. 表示リストで **[重複コレクション]** を選択してから、**[更新]** をクリックします。
9. 各 **[CollectionNumber]** を展開して、重複コレクションごとに [サスペクト] のレコードと [重複] のレコードを表示します。



10. 詳細ビューのコレクションを、作成した出力ファイルと比較します。

マッチング用データフロー テンプレート

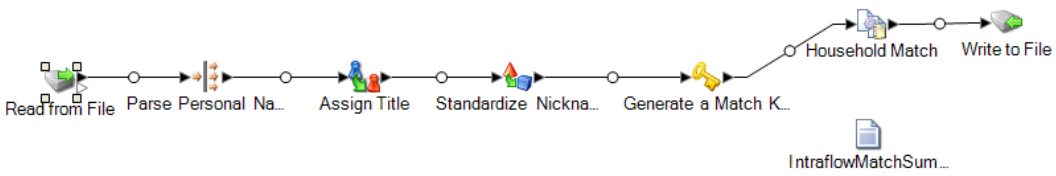
同世帯の個人を識別

このデータフロー テンプレートは、単一の入力ファイル内で情報を比較し、世帯コレクションの出力ファイルを作成することによって、同じ世帯のメンバーを特定する方法を示します。

ビジネス シナリオ

クレジットカード会社のデータ管理責任者が、顧客データベースを分析し、複数のレコードに重複して現れる住所とその住所に住む顧客の氏名を洗い出して、同じ住所に重複して送付される郵便物とクレジットカード契約勧誘の数を最小限にすることを検討しています。

以下のデータフローは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[HouseholdRelationships]** を選択します。このデータフローでは、Advanced Matching モジュール、Data Normalization モジュール、および Universal Name モジュールが必要です。

このデータフローでは、入力ファイルに含まれる各レコードに対して次の処理が行われます。

Read from File

このステージでは、パースする名前が記録されているファイルの名前、格納場所、およびレイアウトを識別します。ファイルには、男性と女性の両方の名前が含まれています。

Open Name Parser

Open Name Parser では、名前フィールドをチェックして、Spectrum™ Technology Platform名前データベース ファイルに格納されている名前データと比較します。この比較結果に基づいて、名

前データが [First]、[Middle]、[Last] の各名前フィールドに分割され、エンティティ タイプと性別がすべての名前に割り当てられます。また、名前データに加えて、パターン認識も使用されます。

Standardize Nicknames

このテンプレートでは、Table Lookup ステージを Standardize Nicknames と呼びます。Standardize Nicknames ステージでは、Nicknames.xml データベースに名を検索し、その名前にフォーマルな表記があればそれでニックネームを置き換えます。例えば、Tommy という名前は Thomas で置き換えます。

Transformer

このテンプレートでは、Transformer ステージを Assign Titles と呼びます。Assign Titles ステージでは、カスタム スクリプトを使って、Parse Personal Name ステージからのデータ ストリーム出力中に各行を検索し、**GenderCode** 値に基づいて **TitleOfRespect** 値を割り当てます。

使用するカスタム スクリプトを以下に示します。

```
if (row.get('TitleOfRespect') == '')
{
 if (row.get('GenderCode') == 'M')
  row.set('TitleOfRespect', 'Mr')
 if (row.get('GenderCode') == 'F')
  row.set('TitleOfRespect', 'Ms')
```

Assign Titles ステージでは、`GenderCode` フィールドで **M** を検出するたびに **TitleOfRespect** の値を `Mr` に設定します。Assign Titles ステージでは、`GenderCode` フィールドで **F** を検出するたびに **TitleOfRespect** の値を `Ms` に設定します。

Match Key Generator

Match Key Generator では、アルゴリズムと入力ソース フィールドで構成されるユーザ定義ルールを処理して、マッチ キー フィールドを生成します。マッチ キーは、重複の可能性があるレコードを意味する類似レコードに共通して与えられる非ユニーク キーです。マッチ キーは、同じマッチ キーを含むレコードのみを比較するために簡易的なマッチングに使われます。マッチ キーは、入力フィールドで構成されます。指定する入力フィールドごとに、そのフィールドで実行されるアルゴリズムが選択されます。その後、各フィールドの結果を連結して、単一のマッチ キー フィールドが作成されます。

このテンプレートでは、SubString (LastName (1:3)) と SubString (PostalCode (1:5)) の 2 つのマッチ キー フィールドが定義されています。

例えば、次のような入力住所があり、

FirstName - Fred

LastName - Mertz

PostalCode - 21114-1687

次のルールが指定されている場合、

入力フィールド	開始位置	長さ
LastName	1	3
PostalCode	1	5

上記のルールと入力データを使うと、次のようなキーになります。

Mer21114

Household Match

このデータフロー テンプレートでは、Intraflow Match ステージを Household Match と呼びます。このステージでは、単一の入力ストリーム内の類似するデータ レコード間でマッチを検出します。また、マッチしたレコードは、名前/住所以外の情報で修飾できます。このマッチング エンジンを使うと、他のステージで定義または作成されたフィールドに基づいて階層型のルールを作成できます。

マッチを検出するレコード ストリームのほかに、比較の対象とするフィールド、スコアの計算方法、および正しいマッチングと見なす全般的な条件も設定されます。

このテンプレートでは、LastName および AddressLine1 の比較を実行するカスタム マッチング ルールを作成します。Interflow Summary Report のデータを生成するために、**[分析用データを生成する]** チェック ボックスを選択します。

マッチング階層を作成する際は、以下のガイドラインに従ってください。

- 親ノード名は一意な名前でなければなりません。フィールドにすることはできません。
- 子フィールドは、Spectrum™ Technology Platformのデータ タイプ フィールドである必要があります。つまり、1つ以上のコンポーネントを通じて使用できるフィールドでなければなりません。
- 親の下のすべての子は、同じ論理演算子を使用する必要があります。コネクタを結合するには、まず中間の親ノードを作成する必要があります。
- 親ノードのしきい値を子のしきい値より低くすることはできません。
- 親ノードにしきい値がなくてもかまいません。

Write to File

テンプレートには 1 つの Write to File ステージがあり、このステージで住所を世帯のコレクションとして表示するテキスト ファイルを作成します。

Intraflow Summary Report

このテンプレートには、Intraflow Match Summary Report が含まれます。ジョブを実行した後で、**[実行の詳細]** ウィンドウ内で [レポート] を展開し、**[IntraflowMatchSummary]** をクリックします。

Intraflow Match Summary Report には、処理済みレコードの統計情報が表示され、レコード数と全体のマッチング スコアがグラフィカルな棒グラフで示されます。

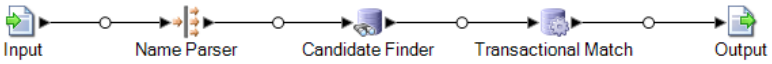
見込み客が顧客かどうかの判定

このデータフロー テンプレートでは、入力ファイル内の見込み客データを顧客データベース内の顧客データと比較して、既に顧客である見込み客を洗い出す方法を示します。これはサービス データフローです。つまり、このデータフローは API または Web サービスを介してアクセスできます。

ビジネス シナリオ

ネット通販会社の営業部長は、オンラインの見込み客が既に購入履歴のある顧客なのか、新規の顧客なのかを確認したいと思っています。

以下のデータフロー サービスは、このビジネス シナリオの解決策を示しています。



このデータフロー テンプレートは Enterprise Designer で使用できます。**[ファイル] > [新規作成] > [データフロー] > [テンプレートから作成]** に移動し、**[ProspectMatching]** を選択します。このデータフローでは、Advanced Matching モジュールと Universal Name モジュールが必要です。

このデータフローでは、入力ファイルに含まれる各レコードに対して次の処理を行います。

入力

このテンプレートで選択される入力フィールドは、[AddressLine1]、[City]、[Name]、[PostalCode]、および [StateProvince] です。[AddressLine1] と [Name] は、このテンプレートによるデータフロー処理のかなめとなるフィールドです。

Name Parser

このテンプレートでは、Name Parser ステージを Parse Personal Name という名前で呼びます。Parse Personal Name ステージでは、名前フィールドがチェックされ、Spectrum™ Technology Platform名前データベース ファイルに格納されている名前データと比較されます。この比較結果に基づいて、名前データが [First]、[Middle]、[Last] の各名前フィールドに分割され、エンティティ タイプと性別がすべての名前に割り当てられます。また、名前データに加えて、パターン認識も使用されます。

このテンプレートでは、Parse Personal Name ステージは次のように設定されています。

- [個人名をパース] が選択され、[企業名をパース] が選択されていません。このようにオプションを設定すると、名が評価されて性別、順序、および句読文字が判別されますが、企業名の評価は実行されません。
- [性別判定ソース] はデフォルトに設定されています。たいていのケースで、デフォルトは性別の判定に最適な設定であり、さまざまな名前に対応できます。ただし、特定のカルチャーに属する名前を処理する場合は、そのカルチャーを選択します。特定のカルチャーを選択すると、名前の性別が適切に判定される確率が高くなります。例えば、デフォルトのままで設定した場合、Jean という名前は女性と判定されます。しかし、[フランス系] を選択した場合は、男性と判定されます。
- 順序は [正順序] に設定されています。名前フィールドの順序は、[敬称]、[名]、[ミドル ネーム]、[姓]、[接尾語] となります。
- [ピリオドを残す] は選択されていません。名前データに含まれる句読文字は残されません。

Candidate Finder

Candidate Finder ステージは、Transactional Match ステージと組み合わせて使用します。

Candidate Finder ステージで候補レコードを取得します。これらの候補レコードは、一連の潜在的なマッチとして、Transactional Match ステージで評価されます。また、Candidate Finder では、データの書式によって、サスペクト レコード、候補レコード、またはその両方のレコードの名前や住所のパーシングが必要となる場合もあります。

Candidate Finder を設定する際に、クエリの実行に使用するデータベース接続を選択します。Management Console で設定されたいずれかの接続を選択できます。リストにないデータベースに接続するには、Management Console でそのデータベースへの接続を設定してから Candidate Finder をいったん閉じ、再び開いて接続リストの内容を更新します。

SQL クエリを定義するために、Candidate Finder オプション ビューのテキスト ボックスに有効な SQL SELECT 文を入力できます。例えば、Customer_Table というテーブルがデータベース内にあり、以下の列が含まれると仮定します。

Customer_Table

Cust_Name
Cust_Address
Cust_City
Cust_State
Cust_Zip

> **注：** 有効な SQL SELECT 文を入力できますが、`Select *` はこのコントロールには無効です。

データベースからこれらの行をすべて取得するには、次のようなクエリを作成します。

```
select Cust_Name, Cust_Address, Cust_City, Cust_State, Cust_Zip from
Customer_Table;
```

ただし、トランザクションをデータベース内のすべての行にマッチさせようとするユーザはほとんどいないでしょう。必要な候補レコードのみを返すには、置換変数を使って WHERE 節を追加します。置換変数を指定すると、Candidate Selection エンジンによって変数がサスペクト レコードの実データで置換されます。

置換変数を使うには、**${FieldName}** のようにフィールド名をカッコで囲み、その直前にドル記号を付けます。例えば、次のクエリを実行すると、サスペクト レコード内の PostalCode の値に Cust_Zip の値が一致するレコードのみが返されます。

```
select Cust_Name, Cust_Address, Cust_City, Cust_State,Cust_Zip
from Customer_Table
where Cust_Zip = ${PostalCode};
```

次に必要な作業は、データベース内の列名が [Component Field] の名前に正確に一致しない場合にそのデータベース列をステージ フィールドにマッピングすることです。名前が正確に一致する列は、対応するステージフィールドに自動的にマッピングされます。選択フィールド (データベース内の列) を使って、ステージ フィールド (データフローで定義されたフィールド名) にマッピングする必要があります。

ここでも、上記の例と同じ Customer_Table を想定します。

Customer_Table
Cust_Name
Cust_Address
Cust_City
Cust_State
Cust_Zip

これらのレコードをデータベースから取得するには、列名を、Transactional Match ステージおよびデータフロー内の他のステージで使われるフィールド名にマッピングする必要があります。例えば、Cust_Address は AddressLine1 にマッピングし、Cust_Zip は PostalCode にマッピングします。

1. Candidate Finder オプション ビューの **[選択フィールド]** の下でドロップダウン リストを選択します。次に、データベース列 Cust_Zip を選択します。
2. **[ステージ フィールド]** の下でドロップダウン リストを選択します。次に、マッピングするフィールドを選択します。

例えば、Cust_Zip を Postal Code にマッピングするには、最初に選択フィールドで Cust_Zip を選択してから、対応するステージ フィールドの行で PostalCode を選択します。

このフィールド マッピング方法のほかに、特殊な書式を SQL クエリで使う方法でも同様のマッピングを実行できます。具体的には、列名に続けてステージ フィールドの名前をカッコで囲んで指定するという書式を使います。この方法を使うと、選択フィールドが、対応するステージフィールドに自動的にマッピングされます。

この代替方法で前の例のクエリを書き直すと、以下のようになります。

```
select Cust_Name {Name}, Cust_Address {AddressLine1},
 Cust_City {City}, Cust_State {StateProvince},
 Cust_Zip {PostalCode}
from Customer
where Cust_Zip = ${PostalCode};
```

Transactional Match

Transactional Match ステージは、Candidate Finder ステージと組み合わせて使用します。

Transactional Match ステージでは、サスペクト レコードを、Candidate Finder ステージから返される潜在的な候補レコードと照合できます。

Transactional Match では、マッチング ルールを使用して、サスペクト レコードを、同じ候補グループ番号 (Candidate Finder で割り当てられる番号) を持つすべての候補レコードと比較して、重複を識別します。候補レコードが重複の場合は、コレクション番号が割り当てられ、そのマッチ レコード タイプに重複が設定され、その候補レコードが書き出されます。グループ内のマッチしない候補にはコレクション番号 0 が割り当てられ、そのラベルにユニークが設定され、その候補が書き出されます。

このテンプレートでは、LastName および AddressLine1 の比較を実行するカスタム マッチング ルールを作成します。

マッチング階層を作成する際は、以下のガイドラインに従ってください。

- 親ノード名は一意な名前でなければなりません。フィールドにすることはできません。
- 子フィールドは、Spectrum™ Technology Platformのデータ タイプ フィールドである必要があります。つまり、1 つ以上のステージを通じて使用できるフィールドでなければなりません。
- 親の下のすべての子は、同じ論理演算子を使用する必要があります。コネクタを結合するには、まず中間の親ノードを作成する必要があります。
- 親ノードのしきい値を子のしきい値より低くすることはできません。
- 親ノードにしきい値がなくてもかまいません。

出力

このテンプレートはサービスなので、すべての使用可能なフィールドを出力に送信します。必要に応じて出力を制限できます。

5 - 重複除去

このセクションの構成

重複レコードのフィルタリング

重複レコードを除去する最も簡単な方法は、マッチング ステージの後のデータフローに Filter ステージを追加することです。Filter ステージは、指定されている設定に基づいて重複レコードのコレクションからレコードを削除します。

1. Enterprise Designer で、マッチングによって重複レコードを特定するデータフローを作成します。

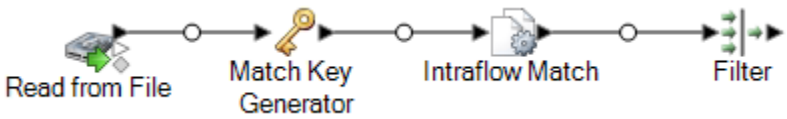
   類似性のあるレコード (同じアカウント番号または名前を含むレコードなど) を特定する必要があるので、重複除外の最初のステップはマッチングです。レコードのマッチングを行うデータフローの作成方法については、以下のトピックを参照してください。

   単一ソースからのレコードのマッチング（85ページ）
   ソース間でのレコードのマッチング（90ページ）
   データベースに対するレコードのマッチング（103ページ）

   注 : 作成する必要があるデータフローは、データを読み込み、Interflow Match、Intraflow Match、または Transactional Match ステージでマッチングを実行するところまでだけです。ここまでのデータフローを作成したら、引き続き以下の手順を実行します。

2. データを読み込んでレコードのマッチングを行うデータフローを定義した後、Filter ステージをキャンバスにドラッグし、マッチングを実行するステージ (Interflow Match、Intraflow Match、または Transactional Match) に接続します。

   例えば、ファイルからデータを読み込み、Intraflow Match でマッチングを実行するデータフローの場合は、Filter ステージを追加した後は次のようになります。

3. キャンバスで Filter ステージをダブルクリックします。
4. **[グループ化]** フィールドで、**[コレクション番号]** を選択します。
5. **[返される重複レコード数の上限値]** オプションはオンのままにし、値を 1 に設定します。これらはデフォルトの設定です。
6. 各コレクションの最初のレコードを保持するか、または各コレクションで保持するレコードを選択するルールを定義するかを決定します。各コレクションの最初のレコードを保持する場合は、このステップをスキップします。ルールを定義する場合は、ルール ツリーで **[ルール]** を選択し、以下の手順に従います。

a) **[ルールの追加]** をクリックします。

各グループのレコードが評価されて、ここで定義したルールを満たすかどうかが確認されます。ルールを満たすレコードは保持され、それ以外のレコードは破棄されます。

b) 各グループで保持するレコードを特定するルールを定義します。

以下のオプションを使用してルールを定義します。

オプション	説明
フィールド名	レコードをフィルタリングするかどうかを判断するために値を評価するデータフロー フィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 100 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに(バイト数が)最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

c) **[OK]** をクリックします。

1 つのルールを含む Filter を設定しました。必要に応じて、さらにルールを追加できます。

7. **[OK]** をクリックして **[Filter オプション]** ウィンドを閉じます。
8. シンク ステージをキャンバスにドラッグし、Filter ステージに接続します。

例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。

9. シンク ステージをダブルクリックして設定します。

シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

各重複グループでルールに一致するレコードを特定して他のすべてのレコードを除去し、重複除外されたデータを含む出力ファイルを作成するデータフローができました。

Best of Breed レコードの作成

データから重複レコードを除去するには、重複レコードのグループのデータを 1 つの "Best of Breed" レコードに結合できます。各重複レコードに同じタイプのデータ (電話番号や名前など) が含まれていて、各レコードの最善のデータを残されるレコードで維持したい場合、この方法が有効です。

この手順では、重複レコードを Best of Breed レコードに結合するデータフローを作成する方法について説明します。

1. Enterprise Designer で、マッチングによって重複レコードを特定するデータフローを作成します。

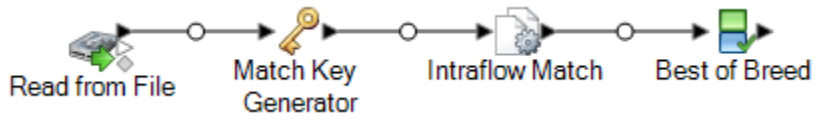
   類似性のあるレコード (同じアカウント番号または名前を含むレコードなど) を特定する必要があるので、重複除外の最初のステップはマッチングです。レコードのマッチングを行うデータフローの作成方法については、以下のトピックを参照してください。

   単一ソースからのレコードのマッチング（85ページ）
   ソース間でのレコードのマッチング（90ページ）
   データベースに対するレコードのマッチング（103ページ）

   注 : 作成する必要があるデータフローは、データを読み込み、Interflow Match、Intraflow Match、または Transactional Match ステージでマッチングを実行するところまでだけです。ここまでのデータフローを作成したら、引き続き以下の手順を実行します。

2. データを読み込んでレコードのマッチングを行うデータフローを定義した後、Best of Breed ステージをキャンバスにドラッグし、マッチングを実行するステージ (Interflow Match、Intraflow Match、または Transactional Match) に接続します。

   例えば、ファイルからデータを読み込み、Intraflow Match でマッチングを実行するデータフローの場合は、Best of Breed ステージを追加した後は次のようになります。



3. キャンバスで Best of Breed ステージをダブルクリックします。
4. **[グループ化]** フィールドで、**[コレクション番号]** を選択します。
5. **[Best of Breed 設定]** の条件ツリーで **[ルール]** を選択します。
6. **[ルールの追加]** をクリックします。

各グループのレコードが評価されて、ここで定義したルールを満たすかどうかが確認されます。レコードがルールと一致する場合、ルールと関連付けられているアクションの構成方法により、データを Best of Breed レコードにコピーできます。アクションは後で定義します。

7. 重複レコードのデータが Best of Breed レコードにコピーされるために重複レコードが満たす必要のあるルールを定義します。

   以下のオプションを使用してルールを定義します。

オプション	説明
フィールド名	条件を満たし、関連付けられているアクションを実行する必要があるかどうかを判断するために値を評価するデータフロー フィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が100のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに(バイト数が)最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

8. **[OK]** をクリックします。
9. ツリーの **[アクション]** ノードをクリックします。
10. **[アクションの追加]** をクリックします。
11. レコードがルールで定義されている条件を満たす場合に Best of Breed レコードにコピーするデータを指定します。

オプション	説明
ソース タイプ	Best of Breed レコードにコピーするデータのタイプを指定します。次のいずれかを選択します。 **フィールド** Best of Breed レコードにフィールドの値をコピーする場合は、このオプションを選択します。 **文字列** Best of Breed レコードに定数値をコピーする場合は、このオプションを選択します。
ソース データ	Best of Breed レコードにコピーするデータを指定します。[ソース タイプ] が [フィールド] の場合は、デスティネーション フィールドに値をコピーするフィールドを選択します。[ソース タイプ] が [文字列] の場合は、デスティネーション フィールドにコピーする定数値を指定します。

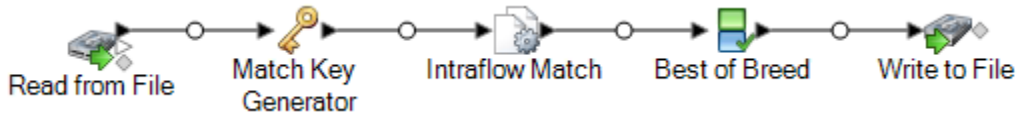
オプション	説明
出力先	**[ソース データ]** フィールドで指定したデータのコピー先となる、Best of Breed レコード内のフィールドを指定します。
[ソース データを蓄積]	**[ソース データ]** フィールドのデータが数値の場合は、このオプションを有効にしてすべての重複レコードのソース データを集約し、その合計値を Best of Breed レコードにコピーできます。 例えば、グループ内に 3 つの重複レコードがあり、Deposits フィールドに以下の値が含まれていたとします。 100.00 20.00 5.00 この場合、3 つすべての値が集約されて 125.00 になり、その合計値が Best of Breed レコードの Deposits フィールドにコピーされます。

12. **[OK]** をクリックします。

    1 つのルールと 1 つのアクションを含む Best of Breed を構成しました。必要に応じて、さらにルールとアクションを追加できます。

13. **[OK]** をクリックして **[Best of Breed オプション]** ウィンドウを閉じます。
14. シンク ステージをキャンバスにドラッグし、Best of Breed ステージに接続します。

    例えば、Write to File シンク ステージを使用した場合、データフローは次のようになります。



15. シンク ステージをダブルクリックして設定します。

    シンク ステージの設定方法については、『*データフロー デザイナー ガイド*』を参照してください。

一致するレコードを特定し、コレクション内のレコードを単一の Best of Breed レコードに結合するデータフローができました。

6 - 例外レコード

このセクションの構成

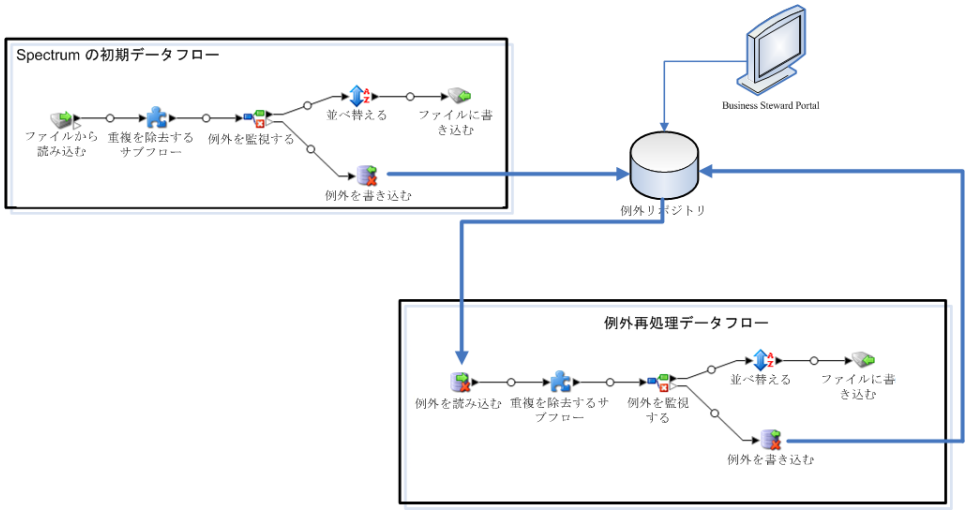
例外を処理するデータフローの設計

Business Steward モジュールのライセンスを供与すると、データフローに例外管理プロセスを含めることができます。例外管理処理の基本的な構成要素には次のものがあります。

- レコードの重複除外、住所検証、ジオコーディングなど、データ品質処理を実行する初期データフロー。
- 処理できなかったレコードを特定する Exception Monitor ステージ。
- Write Exceptions ステージは、Exception Monitor ステージで特定された例外レコードを取得して、手動による確認のために例外リポジトリに書き出します。
- ブラウザベースのツールである Business Steward Portal を使用して、例外レコードを確認および編集することができます。編集したレコードは承認済みとしてマークされ、再処理できるようになります。
- 例外再処理ジョブは、Read Exceptions ステージを使用して、承認済みのレコードを例外リポジトリからジョブに読み込みます。このジョブは、通常は元のデータフローと同じロジックを使用して、訂正されたレコードの再処理を試みます。Exception Monitor ステージで、再度、例外の有無が確認されます。Write Exceptions ステージにより、例外は例外リポジトリに送り返され、追加の確認が行われます。

  注： データフローでは Exception Monitor ステージと Write Exceptions ステージの間に別のステージを配置しないでください。これを行うと、Business Steward Portal での例外の設定に影響が生じます。

基本的な例外管理の実装を説明するシナリオ例を示します。



この例には 2 つのデータフローがあります。1 つは、入力レコードの郵便番号データを評価する初期データフロー、もう 1 つは、編集された例外を取得して、そのレコードに現在は有効な郵便番号データが含まれているかどうかを検証する例外再処理ジョブです。

どちらのデータフローにも Exception Monitor ステージがあります。このステージには、手動による確認のためにレコードをルーティングすべきかどうかの判断に使用する条件が含まれます。この条件は、`PostalCode is empty`など、1 つ以上の式で構成されます。この式は、郵便番号の含まれていないレコードを例外と見なし、Write Exceptions ステージにルーティングして、例外リポジトリに書き出すということを意味します。詳細については、**Exception Monitor**（241ページ）を参照してください。

Exception Monitor が例外として特定したレコードは、Write Exceptions ステージを使用して、例外リポジトリにルーティングされます。データ スチュワードが、例外レコードの確認および変更用のブラウザベースのツールである Business Steward Portal を使用して、リポジトリ内の例外を確認します。この例の場合、データ スチュワードは Business Steward Portal の例外エディターを使用して、例外レコードに手動で郵便番号を追加し、"承認済み" とマークできます。

Business Steward Portal で "承認済み" とマークされたレコードは、Spectrum™ Technology Platform データフローに再度読み込むことができます。この処理は、Read Exceptions ステージを使用して行われます。まだ例外となるレコードがある場合は、データ スチュワードによる確認のため、再度例外リポジトリに書き出されます。

ユーザの状況に最適なアプローチを判断するには、次の点を考慮してください。

- **例外レコードをどのように特定するか**。Exception Monitor ステージでは、任意のフィールドの値、またはフィールドの組み合わせを評価して、レコードが例外かどうかを判断することができます。現在データフローから得ている結果を分析して、例外をどのように特定するかを決定する必要があります。明確に検証済みまたはエラーになったレコードではなく、中程度のデータ品質のレコードを特定したい場合があります。
- **元のデータフローで使用したものと同じロジックを使用して、編集済みおよび承認済みのレコードを再処理するか**。その場合は、サブフローを使用して、再利用可能なビジネス ロジックを作成することができます。例えば、住所検証を実行する初期データフローと、訂正済みのレコードを再処理して訂正を検証する例外再処理ジョブで、サブフローを使用することができます。すると、2 つのデータフロー間で異なるソース ステージとシンク ステージを使用することができます。初期データフローには、処理するデータを顧客データベースから取得する Read from DB ステージが含まれることがあります。例外再処理ジョブには、例外リポジトリから編集済みおよび承認済みの例外レコードを取得する、Read Exceptions ステージが含まれます。
- **訂正済みおよび承認済みの例外をあらかじめ定められたスケジュールで再処理するか**。その場合は、Management Console のスケジューリングを使用して、再処理ジョブをスケジュールすることができます。

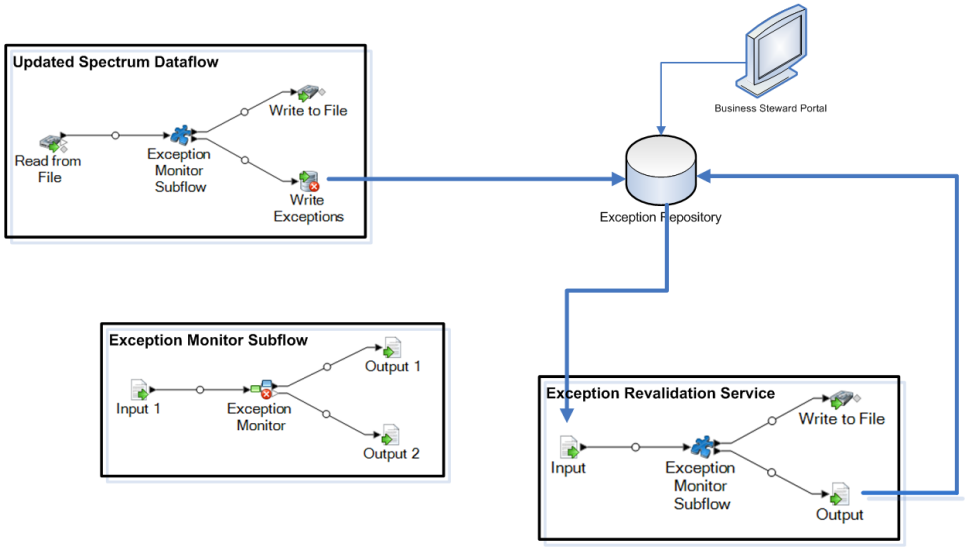
リアルタイム再検証用データフローの設計

例外管理をデータフローで使用している場合、例外レコードを Business Steward Portal で修正した後で、再検証機能を使ってそのレコードに検証プロセスを再実行できます。こうすると、変更によってレコードが正常に処理されるようになるかどうかをリアルタイムで確認できます。Read Exceptions バッチ ジョブが再び実行されて結果が出るのを待つ必要はありません。

再検証の環境は、主に以下の要素で構成されます。

- エクスポーズされたサブフローを再利用するか、またはそれを含むジョブまたはサービス。入力ソース、入力を処理するサブフロー ステージ、Write Exceptions ステージ、正常に処理されたレコードを格納する出力シンクも含まれる必要があります。
- Exception Monitor ステージを含むエクスポーズされたサブフロー。このステージは、再検証サービスをポイントし、再検証に関する設定 (再検証されたレコードを再処理するか承認するかなど) が済んでいるものです。
- エクスポーズされたサブフローを再利用するか、それを含むエクスポーズされたサービス。このサービスは、Business Steward Portal で編集、保存され、再検証のために送信されたレコードを処理します。

再検証の実装について具体的に説明するシナリオ例を示します。



次の例では、ジョブ、サブフロー、サービスの 3 つのデータフローを使用します。ジョブは、入力データを使ってサブフローを実行します。サブフローには Exception Monitor ステージが含まれ、そこでは、レコードを手動による確認に回すかが決定されます。この例では、[PostalCode] フィールドに値がないレコードを例外とみなし、Write Exceptions ステージに送ります。この例

外は、Business Steward Portal に現れるものです。このフィールドに値があれば、レコードは Write to File ステージに送られます。

> **注：** データフローが Best of Breed 機能も使うように設定されている場合は、再検証 Exception Monitor ステージ/サブフローとサービス自体に CollectionRecordType フィールドを手動で追加してエクスポーズする必要があります。Best of Breed 機能の詳細については、**[例外データの書き出し]** オプションと**Best of Breed レコードの作成**（271ページ）を参照してください。

Exception Monitor ステージを構成するときに設計した例外再検証サービスは、Business Steward Portal 例外エディターで 1 つ以上の例外レコードを編集し、**[再検証して保存]** をクリックすると呼び出されます。ジョブと同様に、同じビジネス ロジックでレコードを再処理する例外監視サブフローがサービスに含まれます。レコードが Exception Monitor ステージで設定された条件を 1 つでも満たさないと、例外がリポジトリで更新されます。レコードが Exception Monitor ステージで設定された条件を満たすと、1 つまたは 2 つのアクションが実行されます。どのアクションが実行されるかは、[再検証後のアクション] フィールドでの選択によって決まります。

- **レコードを再処理** — レコードがリポジトリから削除され、再処理されます。
- **レコードを承認** — レコードに承認済みのマーキングをし、リポジトリに戻します。

次の手順に従い、リアルタイムの再検証シナリオを作成して使用します。

1. ジョブまたはサービス データフローを開くか、作成します。これには、Exception Monitor ステージ、入力ソース (Read from File ステージや Input ステージなど)、出力シンク (Write to File ステージや Output ステージなど)、Write Exceptions ステージが含まれる必要があります。
2. Exception Monitor ステージをサブフローに変換し、入力フィールドと出力フィールドを処理データフローの各フィールドに対応付けます。ExceptionMetadata フィールドは、入力ソースだけでなく、Write Exceptions ステージをジョブに埋め込む出力ステージにも忘れずに含めてください。ジョブやサービスから使用できるようにサブフローをエクスポーズします。
3. Input ステージ、手順 2 で作成したサブフロー、Output ステージ、出力シンク (Write to File ステージ、Write to DB ステージなど) が含まれるサービスを作成します。入力フィールドと出力フィールドを初期データフローの各フィールドに対応付けます。ExceptionMetadata フィールドを Input ステージだけでなく Output ステージにも含めることを忘れないでください。サブフローで使用できるようにサービスをエクスポーズします。
4. サブフローに戻り、Exception Monitor ステージの [構成] タブを開きます。手順 3 で作成した再検証サービスを選択し、再検証後に実行するアクションを指定します。サブフローを保存し、もう一度エクスポーズします。
5. サービスに戻ります。メッセージが表示され、サブフローが変更されたため、サービスの内容が更新されることが通知されます。**[OK]** をクリックしてから、サービスを保存し、もう一度エクスポーズします。

6. 初期ジョブまたはサービスに戻ります。メッセージが表示され、サブフローが変更されたため、サービスの内容が更新されることが通知されます。**[OK]** をクリックしてから、データフローを保存します。
7. ジョブを実行します。

   注： 初期ジョブまたはサービスを以前に実行したことがある場合でも、再検証の資格があるレコードをリポジトリに格納する再検証シナリオを作成した後で、そのジョブまたはサービスを実行しなければなりません。例外エディター内で [再検証して保存] ボタンがアクティブになっていれば、それはそのレコードが再検証の資格があるという意味です。

7 - 検索テーブル

このセクションの構成

検索テーブルの概要

検索テーブルは、キー/値のペアで構成されるテーブルです。Spectrum™ Technology Platform のステージはデータを正規化するとき、このテーブルを使用してトークンの置き換えを行います。Advanced Transformer、Open Parser、および Table Lookup で使用される検索テーブルの内容を変更するには、Enterprise Designer のテーブル管理ツールを使用します。

Data Normalization モジュールのテーブル

Advanced Transformer のテーブル

Advanced Transformer は、次のテーブルを使用して語を識別します。新しいテーブルの作成、または既存のテーブルの変更には、テーブル管理を使用してください。詳細については、**検索テーブルの概要**（158ページ）を参照してください。

- 航空略語
- すべての頭字語と頭文字語
- 企業名の略語
- カナダの準州の略号
- コンピューティング/IT 略語
- 区切り記号
- ドイツ企業
- Fortune 1000
- 地理的方向指示の略号
- Global Sentry のノイズ語
- Global Sentry の認可国
- 政府機関の略語
- IATA 航空会社コード
- IATA 航空会社コード (国)
- 法律略語
- 医学略語

- 医療機関の頭字語
- 軍事略語
- ニックネーム
- セカンダリ ユニットの略号
- セカンダリ ユニットの逆順序
- シンガポールの略語
- スペイン語の略語
- スペイン語の方向指示の略号
- スペイン語のストリート接尾語の略号
- 州名の略号
- 州名の逆順序
- ストリート接尾語の略号
- ストリート接尾語の逆順序
- 子会社と親会社
- 米国陸軍の頭字語
- 米国海軍の頭字語

Open Parser のテーブル

Open Parser は、次のテーブルを使用して語を識別します。新しいテーブルの作成、または既存のテーブルの変更には、テーブル管理を使用してください。詳細については、**検索テーブルの概要**（158ページ）を参照してください。

基本テーブル

基本テーブルは、Data Normalization モジュールのインストール パッケージに同梱されています。

- アカウント説明
- 企業
- 企業接続詞
- 企業接頭語
- 企業接尾語
- 企業名称
- 接続詞
- 姓の前に付ける敬称
- 姓
- 一般接尾語
- ドイツ企業

- 名
- 世代接尾語
- スペイン語の名
- スペイン語の姓
- 敬称

Core Name のテーブル

Core Names のテーブルは、Data Normalization モジュールのインストール パッケージでは提供されず、別途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。

Core Names のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- 拡張姓
- 拡張名

企業名のテーブル

企業名のテーブルは、Data Normalization モジュールのインストール パッケージでは提供されず、別途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。

企業名のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- 企業 - 南北アメリカ大陸
- 企業 - アジア太平洋地区
- 企業 - EMEA
- 企業冠詞
- 企業接続詞

Arabic Plus Pack のテーブル

Arabic Plus Pack のテーブルは、Data Normalization モジュールのインストール パッケージでは提供されず、別途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。

Arabic Plus Pack のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- アラビア語の姓 (アラビア文字)

- アラビア語の姓 (ローマ字)
- アラビア語の名 (アラビア文字)
- アラビア語の名 (ローマ字)

Asian Plus Pack のテーブル

Asian Plus Pack のテーブルは、Data Normalization モジュールのインストール パッケージでは提供されず、別途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。

Asian Plus Pack のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- 中国語の姓 (ネイティブ)
- 中国語の姓 (ローマ字)
- 中国語の名 (ネイティブ)
- 中国語の名 (ローマ字)
- 韓国語の姓 (ネイティブ)
- 韓国語の姓 (ローマ字)
- 韓国語の名 (ネイティブ)
- 韓国語の名 (ローマ字)
- 日本語の姓 (かな)
- 日本語の姓 (漢字)
- 日本語の姓 (ローマ字)
- 日本語の名 (かな)
- 日本語の名 (漢字)
- 日本語の名 (ローマ字)

Table Lookup のテーブル

Table Lookup は、次のテーブルを使用して語を識別します。新しいテーブルの作成、または既存のテーブルの変更には、テーブル管理を使用してください。詳細については、**検索テーブルの概要**（158ページ）を参照してください。

基本テーブル

基本テーブルは、Data Normalization モジュールのインストール パッケージに同梱されています。

- 航空略語
- すべての頭字語と頭文字語

- 企業名の略語
- カナダの準州の略号
- コンピューティング/IT 略語
- EU の頭字語
- Fortune 1000
- フランスの略語
- フランスの Arrondissement (郡) と Department (県) 番号の対応
- フランスの Commune (コミューン) と郵便番号の対応
- フランスの Department (県) と Region (地域圏) の対応
- フランスの Department (県) 番号と Department の対応
- 性別コード
- 地理的方向指示の略号
- ドイツの頭字語
- ドイツの都市と州コードの対応
- ドイツの市外局番と都市の対応
- ドイツの都市と州コードの対応
- ドイツの州の略語
- Global Sentry の認可国
- 政府機関の略語
- IATA 航空会社コード
- IATA 航空会社コード (国)
- 法律略語
- 医学略語
- 医療機関の頭字語
- 軍事略語
- ニックネーム
- セカンダリ ユニットの略号
- セカンダリ ユニットの逆順序
- シンガポールの略語
- スペイン語の略語
- スペイン語の方向指示の略号
- スペイン語のストリート接尾語の略号
- 州名の略号
- 州名の逆順序
- ストリート接尾語の略号
- ストリート接尾語の逆順序
- 子会社と親会社

- 英国の都市と郵便番号エリアの対応
- 英国の市外局番のプレフィックス
- 英国の市外局番と都市の対応
- 英国の郵便番号エリアと都市の対応
- 米国陸軍の頭字語
- 米国海軍の頭字語
- ZREPLACE (フランスの住所検証用の SAP モジュールで使用)

Core Names

Core Names のテーブルを使用するには、ライセンスが別途必要です。詳細については、営業担当者に問い合わせてください。

Core Names のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- 拡張姓エスニシティ
- 拡張性別コード
- 拡張名エスニシティ

Arabic Plus Pack

Arabic Plus Pack のテーブルを使用するには、ライセンスが別途必要です。詳細については、営業担当者に問い合わせてください。

Arabic Plus Pack のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- アラビア語の姓エスニシティ (アラビア文字)
- アラビア語の姓エスニシティ (ローマ字)
- アラビア語の性別コード (アラビア語)
- アラビア語の性別コード (ローマ字)
- アラビア語の名エスニシティ (アラビア文字)
- アラビア語の名エスニシティ (ローマ字)

Asian Plus Pack

Asian Plus Pack のテーブルを使用するには、ライセンスが別途必要です。詳細については、営業担当者に問い合わせてください。

Asian Plus Pack のテーブルは、Data Normalization モジュールのデータベース ロード ユーティリティを使用してロードしなければなりません。手順については、『*Spectrum™ Technology Platform インストール ガイド*』を参照してください。

- CJK の姓エスニシティ (ネイティブ)
- CJK の姓エスニシティ (ローマ字)
- CJK の名エスニシティ (ネイティブ)
- CJK の名エスニシティ (ローマ字)
- 日本語の性別コード (かな)
- 日本語の性別コード (漢字)
- 日本語の性別コード (ローマ字)

Universal Name モジュールのテーブル

Name Variant Finder テーブル

Name Variant Finder ステージでは、次のテーブルを使用します。テーブルごとに個別のライセンスが必要です。

- Arabic Plus Pack: `g1-cdq-cjki-arabic-<date>.jar`
- Asian Plus Pack - 中国語:`g1-cdq-cjki-chinese-<date>.jar`
- Asian Plus Pack - 日本語:`g1-cdq-cjki-japanese-<date>.jar`
- Asian Plus Pack - 韓国語: `g1-cdq-cjki-korean-<date>.jar`
- Core Names Database: `g1-cdq-nomino-base-<date>.jar`

Open Name Parser テーブル

Open Name Parser は、次のテーブルを使用して語を識別します。新しいテーブルの作成、または既存のテーブルの変更には、テーブル管理を使用してください。詳細については、**検索テーブルの概要**（158ページ）を参照してください。

基本テーブル

基本テーブルは、Universal Name モジュールのインストール パッケージに同梱されています。

- アカウント説明

- 企業接続詞
- 接続詞
- 姓の前に付ける敬称
- 姓
- 一般接尾語
- 名
- 世代接尾語
- スペイン語の名
- スペイン語の姓
- 敬称

Core Name のテーブル

Core Name のテーブルは、Universal Name モジュールのインストール パッケージには同梱されていないため、追加のライセンスが必要です。

- 拡張姓
- 拡張名

企業名のテーブル

以下の企業名のテーブルは、Universal Name モジュールのインストール パッケージに同梱されています。

- アカウント説明
- 企業
- 企業冠詞
- 企業接続詞
- 企業接頭語
- 企業接尾語
- 企業名称
- 接続詞

以下の企業名テーブルは、Universal Name モジュールのインストール パッケージには同梱されていないため、追加のライセンスが必要です。

- 企業 - 南北アメリカ大陸
- 企業 - アジア太平洋地区
- 企業 - EMEA

Asian Plus Pack のテーブル

Asian Plus Pack のテーブルは、Universal Name モジュールのインストール パッケージには同梱されていないため、追加のライセンスが必要です。

- 日本語の姓 (かな)

- 日本語の姓 (漢字)
- 日本語の姓 (ローマ字)
- 日本語の名 (かな)
- 日本語の名 (漢字)
- 日本語の名 (ローマ字)
- 日本語の敬称

検索テーブルの内容の表示

Enterprise Designer のテーブル管理を使用すると、検索テーブルの内容を表示できます。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. **[タイプ]** フィールドで、表示する検索テーブルがあるステージを選択します。
3. **[名前]** フィールドで、表示するテーブルを選択します。
4. 以下のオプションを使用して、テーブルの表示方法を変更できます。

オプション	説明
特定の語を検索する	**[次で開始]** フィールドに、検索する語を入力し、**[更新]** をクリックします。
テーブルのページを移動	**[更新]** ボタンの右側にある進む/戻るのアイコンをクリックします。
ページあたりの表示項目数を変更する	**[ページあたりの項目数]** フィールドの値を変更します。
Table Lookup テーブル内にある各正規化語の検索語をすべて表示する	**[次の順で表示]** フィールドで、**[正規化語 (グループ化)]** を選択します。このオプションは Table Lookup テーブルでのみ使用できます。

検索テーブルへの語の追加

検索テーブルに含まれていない語がデータ内にあることがわかり、その語を検索テーブルに追加する場合は、以下の手順に従います。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. **[タイプ]** フィールドで、変更する検索テーブルがあるステージを選択します。
3. **[名前]** フィールドで、語を追加するテーブルを選択します。
4. **[追加]** をクリックします。
5. **[検索語]** フィールドに、データ内に存在する語を入力します。これが検索キーとして使用されます。
6. Table Lookup テーブルの場合は、**[正規化語]** フィールドに、データフロー内の検索語を置き換える語を入力します。

   例えば、PB という語を Pitney Bowes に変更する場合は、PB を検索語として、Pitney Bowes を正規化語として入力します。

7. この語が既に Table Lookup テーブルに存在し、ステップ 5 で入力した値で置き換えたい場合は、**[既存の語を上書き]** チェック ボックスを選択します。
8. **[追加]** をクリックします。

検索テーブルからの語の削除

検索テーブルから語を削除するには

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. 語を選択し、**[削除]** をクリックします。
3. **[はい]** をクリックしてテーブルの語を削除します。

正規化された形式の語の変更

語の正規化のために Table Lookup で使用されるテーブルの場合、正規化された形式の語を変更できます。例えば、PB および PB Software という検索語を持つテーブルがあり、正規化された語が Pitney Bowes であって、正規化された形式を Pitney Bowes Inc, に変更する場合は、以下の手順に従ってこの操作を行うことができます。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. **[タイプ]** フィールドで、**[Table Lookup]** を選択します。
3. **[名前]** フィールドで、変更するテーブルを選択します。
4. 変更する語を選択し、**[変更]** をクリックします。

   ヒント： 1 つの正規化語について複数の検索語が存在する場合は、**[次の順で表示]** フィールドで **[正規化語 (グループ化)]** を選択し、グループを選択して、**[変更]** をクリックすると、新しい正規化語を使用するようにすべての検索語を容易に変更できます。

5. **[正規化語]** フィールドに新しい値を入力します。
6. **[OK]** をクリックします。

テーブルのカスタマイズを元に戻す

テーブルに変更を加えた場合、テーブルを元の状態に戻すことができます。テーブルのカスタマイズを元に戻すには、次の手順に従います。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. 元に戻すテーブルを選択します。
3. **[元に戻す]** をクリックします。

   [元に戻す] ウィンドウが表示されます。このウィンドウには、追加、削除、および変更された語のリストが表示されます。

4. 元に戻したい各テーブルエントリの **[元に戻す]** チェック ボックスを選択します。**[すべて選択]** または **[すべて非選択]** をクリックして、すべての **[元に戻す]** チェック ボックスを選択またはクリアすることもできます。
5. **[OK]** をクリックします。

検索テーブルの作成

Advanced Matching モジュール、Data Normalization モジュール、および Universal Name モジュールには、語を置き換えたり正規化したりする幅広い処理に使用できるさまざまなテーブルが付属します。ただし、これらのテーブルがニーズに合わない場合は、独自に検索語のテーブルを作成して Advanced Transformer、Open Parser、または Table Lookup で使用できます。テーブルを作成するには、以下の手順に従います。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. **[タイプ]** フィールドで、検索テーブルを作成するステージを選択します。
3. **[新規]** をクリックします。**[テーブルの追加]** ダイアログ ボックスが表示されます。
4. **[テーブル名]** フィールドに、新しいテーブルの名前を入力します。
5. 選択したタイプの新しい空のテーブルが必要な場合は、**[コピー元]** は **なし** に設定したままにしてください。既存のテーブルから新しいテーブルを設定する場合は**[コピー元]** リストからテーブル名を選択します。
6. **[OK]** をクリックします。

新規テーブルへのテーブル項目の追加については、**検索テーブルへの語の追加**（167ページ）を参照してください。

データのインポート

検索テーブルへのデータのインポート

ファイルからデータを検索テーブルにインポートして、Advanced Transformer、Open Parser、または Table Lookup で使用できます。ファイルから検索テーブルにデータをインポートするためには、ファイルが以下の要件を満たしている必要があります。

- UTF-8 エンコードであること。
- 区切り記号付きのファイルであること。サポートされている区切り文字がカンマ (,)、セミコロン (;)、パイプ (|)、タブ (\t) であること。
- 区切り記号が埋め込まれたフィールドが二重引用符で開始および終了していること ("1,a"、"2,b"、"3,c" など)。

- 二重引用符で開始および終了しているフィールド内のリテラル引用符が 2 つの引用符を持つこと ("2"" feet" など)。

データをファイルから検索テーブルにインポートするには

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. データのインポート先となるテーブルを選択します。または、新しいテーブルを作成します。テーブル作成の手順については、**検索テーブルの作成**（169ページ）を参照してください。
3. **[インポート]** をクリックします。
4. **[参照]** をクリックして、インポートするデータが含まれているファイルを選択します。
5. **[開く]** をクリックします。ファイルのプレビューに、インポートされたファイル内のデータのプレビューが表示されます。
6. ユーザ定義テーブルから列を選択して、既存テーブルの列にマップすることができます。例えば、インポートするユーザ定義テーブルに 2 つの列があるものとします。column1 と column2 です。列のリストには column1 と column2 が表示されます。column2 を選択して検索語にマップし、column1 を選択して正規化語にマップすることができます。
7. **[新しい語のみインポート]** を選択してユーザ定義テーブルから新規レコードのみをインポートするか、**[既存の語を上書き]** を選択して選択した列のすべてのレコードをインポートします。
8. **[OK]** をクリックします。

高度なインポートの使用

高度なインポート機能を使用すると、Advanced Transformer、Table Lookup、および Open Parser で使用される検索テーブルにデータを選択的にインポートできます。以下のような場合に高度なインポートを使用します。

- 区切り記号付きのユーザ定義ファイルで選択した列から、語を抽出する。
- 区切り記号付きのユーザ定義ファイルで選択した列から、1 語からなる単語 (トークン) を抽出する。トークンを抽出すると、ファイル内の指定の列にその語が出現する回数を識別し、関連語のグループを作成して、テーブルに追加することができます。

インポートするデータが含まれているファイルは、以下の要件を満たしている必要があります。

- UTF-8 エンコードであること。
- 区切り記号付きのファイルであること。サポートされている区切り文字がカンマ (,)、セミコロン (;)、パイプ (|)、タブ (\t) であること。
- 区切り記号が埋め込まれたフィールドが二重引用符で開始および終了していること ("1,a"、"2,b"、"3,c" など)。
- 二重引用符で開始および終了しているフィールド内のリテラル引用符が 2 つの引用符を持つこと ("2"" feet" など)。

1. Enterprise Designer で、**[ツール] > [テーブル管理]** を選択します。
2. データのインポート先となるテーブルを選択します。
3. **[高度なインポート]** を選択します。
4. **[参照]** をクリックして、インポートするファイルを選択します。
5. **[開く]** をクリックします。
6. **[列]** リストからテーブル列を選択します。サンプル データによって、ユーザ定義テーブルに表示される各語の出現頻度が示されます。頻度は、既存のテーブルにまだ存在しない語に関してのみ表示されます。
7. 語を 1 語からなる単語として表示するには、**[1 語からなる単語に分割]** を選択します。
8. Advanced Transformer テーブルおよび Open Parser テーブルの場合
   a) 左側のリストから語を選択します。
   b) 右側のリストに語を追加するには、右矢印をクリックします。選択した語をテーブル リストから削除するには、左矢印をクリックします。
   c) **[OK]** をクリックして、変更をテーブルに保存します。
9. Table Lookup テーブルの場合
   a)  をクリックして、テーブル グループを追加します。
   b) **[新規]** をクリックします。
   c) 新しい語を入力し、**[追加]** をクリックします。語の追加を終了するまで続行し、**[閉じる]** をクリックします。
   d) リストから語を選択し、[追加] をクリックします。語の追加を終了するまで続行し、**[閉じる]** をクリックします。新しい語が右側の語のリストに追加されます。
   e) 選択したグループに語を追加するには、左側で語を選択し、右矢印をクリックします。語をいずれかのグループから削除するには、左矢印をクリックします。
   f) 語を変更するには、右側のリストから語を選択して、  をクリックします。
   g) 語を削除するには、右側のリストから語を選択して、  をクリックします。
   h) **[OK]** をクリックして、変更をテーブルに保存します。

8 - ステージ リファレンス

このセクションの構成

Advanced Matching モジュール

Advanced Matching モジュール

Advanced Matching モジュールは、複数の入力ファイル間や、複数の入力ファイル内でレコードを照合します。また、Advanced Matching モジュールでは、名前、住所、名前と住所、名前/住所以外のフィールド (社会保障番号、生年月日等) 等、さまざまなフィールドの照合も可能です。

コンポーネント

Advanced Matching モジュールは、次のステージで構成されます。

- **Best-of-Breed** — このステージでは、テンプレート レコードを選択して、そのレコードから複合的な最良のレコードを作成することで、重複クラスタから最良の組み合わせレコードを選択します。このレコードは、サバイバ レコードとして返されます。
- **Candidate Finder** — このステージでは、候補レコードを取得します。これらの候補レコードは、一連の潜在的なマッチとして、Transactional Match ステージで評価されます。Candidate Finder は Transactional Match と併用します。
- **Duplicate Synchronization** — このステージでは、レコードのコレクションからフィールドを指定して、そのフィールドをコレクション内のすべてのレコードの対応するフィールドにコピー (配置) できます。
- **Filter** — このステージでは、重複レコードのコレクションから削除するレコードと、そのコレクションに残すレコードを選別する条件を指定できます。この条件によって絞り込まれたレコードは、下流でさらに処理されるか、出力ファイルに書き込まれます。
- **Interflow Match** — このステージでは、複数の入力ストリーム内の類似するデータ レコード間でマッチを特定します。
- **Intraflow Match** — このステージでは、単一の入力ストリーム内の類似するデータ レコード間でマッチを特定します。
- **マッチ分析** — マッチ分析では、マッチング ステージからの結果を分析および比較して、マッチング処理の結果を詳細に把握できます。また、マッチ分析は、マッチング結果の向上に役立てることもできます。
- **Match Key Generator** — このステージでは、すべての類似レコードで共有する非ユニーク キーを作成します。同じキーを共有するレコードは、グループ化して比較できます。
- **Private Match** — このステージでは、機密情報を漏洩することなく、2 つのエンティティでデータセットを比較し、共通のレコードまたは顧客を特定することができます。暗号化機能を使用

することにより、マッチング実行時のセキュリティが維持され、復号化機能によって、マッチしたデータの出力が表示されます。

- **Transactional Match —** このステージでは、Candidate Finder ステージでクエリを実行してサスペクト トランザクションをデータベースと照合し、可能性がある候補レコードを返します。
- **Write to Search Index —** このステージでは、ステージへの入力データに基づいて全文インデックスを作成できます。このデータを専用検索インデックスに取り込むと、Candidate Finder からのインデックスに対する検索を行うときに応答がより迅速に得られます。

Best of Breed

Best of Breed は、重複レコードのコレクションから選択する最良のデータを使用して新しい統合レコードを作成することで、重複レコードを統合します。この "スーパー" レコードは、最良の組み合わせレコードと呼ばれます。処理対象レコードの選択で使用するルールを定義します。処理が完了すると、最良の組み合わせレコードがシステムに保持されます。

オプション

以下の表に、Best of Breed のオプションを示します。

オプション名	説明/有効値
グループ化方法	単一の Best of Breed レコードに結合するレコードのグループを作成するのに使用するフィールドを指定し、グループごとに Best of Breed レコードを 1 つ作成します。データフローで以前、マッチング ステージを使用していた場合は、CollectionNumber フィールドを選択して、そのマッチング ステージで作成されたコレクションをグループとして使用する必要があります。ただし、別のフィールドによってレコードをグループ化する場合は、そのフィールドを選択します。例えば、AccountNumber フィールドの値が同じであるすべてのレコードを 1 つの Best of Breed レコードに結合する場合は、AccountNumber を選択します。
ソート	**[グループ化]** フィールドにフィールドを指定する場合は、このボックスをオンにして、選択したフィールドの値でレコードをソートします。このオプションは、デフォルトで有効になっています。

オプション名	説明/有効値

詳細設定	ソート パフォーマンス オプションを指定するには、このボタンをクリックします。デフォルトでは、Management Console で指定されたソート パフォーマンス オプション (システム用のデフォルトのパフォーマンス オプション) が有効になります。システムのデフォルトのパフォーマンス オプションよりも優先する場合は、**[ソート パフォーマンス オプションをオーバーライド]** ボックスをオンにし、以下のフィールドに値を指定します。
	
	メモリ内レコードの上限値
	ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。
	
	注： 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。
	
	一時ファイルの最大数
	ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する一時ファイル数の増減がパフォーマンスに与える影響を確認する必要があります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。
	
	`(NumberOfRecords × 2) ÷ InMemoryRecordLimit = NumberOfTempFiles`
	
	一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。
	
	圧縮を有効にする
	一時ファイルをディスクに書き込むときに圧縮します。
	
	注： 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。それでも一般には、次の式で妥当なソート パフォーマンスが得られます。
	
	`(InMemoryRecordLimit × MaxNumberOfTempFiles ÷ 2) >= TotalNumberOfRecords`

オプション名	説明/有効値
元のレコードを保持	Best of Breed レコードと共にコレクション内のすべてのレコードを保持するには、このオプションを選択します。Best of Breed レコードのみが必要な場合は、このオプションをオフにします。
最初のレコードを使用	コレクションの最初のレコードをテンプレート レコードとして自動的に選択するには、このオプションを選択します。テンプレート レコードは、Best of Breed レコードを作成するときのベースとなるレコードです。
テンプレート レコードを定義	テンプレート レコードを選択するルールを定義する場合は、このオプションを選択します。詳細については、テンプレート レコード ルールの定義（176ページ）を参照してください。

テンプレート レコード ルールの定義

Best of Breed の処理では、Best of Breed レコードの作成に使用するコレクション内のレコードがテンプレート レコードになります。テンプレート レコードは、Best of Breed レコードを作成するときの出発点として使用され、定義している Best of Breed 設定に基づいて修正されます。Best of Breed ステージでは、テンプレート レコードを自動的に選択することも、テンプレート レコードを選択するルールを定義することもできます。このトピックでは、テンプレート レコードを選択するルールの定義方法について説明します。

テンプレート ルールを作成するには、フィールド名、演算子、値タイプ、および値を指定します。テンプレート レコード オプションの例を以下に示します。

フィールド名: MatchScore
フィールド タイプ: 数値
演算子: 等しい
値タイプ: 文字列
値: 100

このテンプレート ルールを適用すると、マッチ スコアが値 100 に等しいレコードがコレクションから選択されます。

以下の手順は、Best of Breed ステージでテンプレート レコード ルールを定義する方法を示しています。

1. Best of Breed ステージの **[テンプレート レコード設定]** の下で、**[テンプレート レコードを定義]** オプションを選択します。
2. ツリー内の **[ルール]** をクリックします。

3. **[ルールの追加]** をクリックします。
4. 以下のフィールドに必要な情報を入力します。

オプション	説明
フィールド名	レコードをテンプレート レコードにする必要があるかどうかを判断するために値を評価するデータフロー フィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が100のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに(バイト数が)最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

5. **[OK]** をクリックします。
6. ルールを追加で指定する場合は、**[ルールの追加]** をクリックします。

   その他のルールを追加する場合は、各ルールの間で使用する論理演算子を選択する必要があります。新しいルールと前のルールの両方に合格しないと、レコードをテンプレート レコードとして選択できないように設定する場合は、**[And]** を選択します。前のルールまたは新しいルールのいずれか一方に合格すれば、レコードをテンプレート レコードとして選択できるように設定する場合は、**[Or]** を選択します。

これで、テンプレート レコードの選択で使用するルールを設定できました。Best of Breed 設定を行って、Best of Breed ステージの設定を完了します。

Best of Breed ルールとアクションの定義

Best of Breed ルールとアクションは、コレクション内の重複レコードのどのフィールドを Best of Breed レコードにコピーするかを決定します。ルールはレコード内の値をテストします。レコードがルールに合格すると、レコードのデータがテンプレート レコードにコピーされます。アクションは、コピーするデータと、データを受け取るテンプレート レコード内のフィールドを定義します。ルールとアクションがすべて実行されると、テンプレート レコードは Best of Breed レコードになります。

ルールとアクションは条件としてグループ化し、複数の条件を使用できます。これにより、以下の操作が可能です。

1. Best of Breed ステージの **[Best of Breed 設定]** の下で、ツリー内の **[ルール]** ノードをクリックできます。
2. **[ルールの追加]** をクリックします。
3. 以下のフィールドに必要な情報を入力します。

オプション	説明
フィールド名	条件を満たし、関連付けられているアクションを実行する必要があるかどうかを判断するために値を評価するデータフロー フィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が100のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに(バイト数が)最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

4. **[OK]** をクリックします。
5. この条件のルールを追加で指定する場合は、**[ルールの追加]** をクリックします。

   その他のルールを追加する場合は、各ルールの間で使用する論理演算子を選択する必要があります。新しいルールと前のルールの両方に合格しないと、条件を満たしていると判断せず、関連付けられているアクションを実行できないように設定する場合は、**[And]** を選択します。前のルールまたは新しいルールのいずれか一方に合格すれば、条件を満たしていると判断するように設定する場合は、**[Or]** を選択します。
6. ツリーの **[アクション]** ノードをクリックします。
7. **[アクションの追加]** をクリックします。
8. 以下のフィールドに必要な情報を入力します。

オプション	説明
ソース タイプ	Best of Breed レコードにコピーするデータのタイプを指定します。次のいずれかを選択します。 **フィールド** Best of Breed レコードにフィールドの値をコピーする場合は、このオプションを選択します。 **文字列** Best of Breed レコードに定数値をコピーする場合は、このオプションを選択します。

オプション	説明
ソース データ	Best of Breed レコードにコピーするデータを指定します。[ソース タイプ] が [フィールド] の場合は、デスティネーション フィールドに値をコピーするフィールドを選択します。[ソース タイプ] が [文字列] の場合は、デスティネーション フィールドにコピーする定数値を指定します。
出力先	**[ソース データ]** フィールドで指定したデータのコピー先となる、Best of Breed レコード内のフィールドを指定します。
[ソース データを蓄積]	**[ソース データ]** フィールドのデータが数値の場合は、このオプションを有効にしてすべての重複レコードのソース データを集約し、その合計値を Best of Breed レコードにコピーできます。 例えば、グループ内に 3 つの重複レコードがあり、Deposits フィールドに以下の値が含まれていたとします。 100.00 20.00 5.00 この場合、3 つすべての値が集約されて 125.00 になり、その合計値が Best of Breed レコードの Deposits フィールドにコピーされます。

9. **[OK]** をクリックします。
10. この条件で実行するアクションを追加で指定する場合は、**[アクションの追加]** をクリックして上記の手順を繰り返します。
11. 他の条件を追加するには、ツリー内のルート条件をクリックし、**[条件の追加]** をクリックします。

Best of Breed ルールとアクションの例

この Best of Breed ルールを適用すると、マッチ スコアが値 100 に等しいレコードが選択されます。続いて、選択されたフィールドに対応する顧客番号データが、Best of Breed レコード上の AccountNumber フィールドにコピーされます。

ルール
フィールド名: MatchScore
フィールド タイプ: 数値
演算子: 等しい
値タイプ: 文字列
値: 100

アクション

ソース タイプ: フィールド
ソース データ: AccountNumber
出力先: AccountNumber

出力

表 8 : Best of Breed 出力

フィールド名	書式	説明/有効値
CollectionRecordType	String	重複するレコードのコレクションからテンプレートと Best of Breed レコードを識別します。有効な値を次に示します。 **Primary** レコードは、コレクションの選択されたテンプレート レコードです。 **Secondary** レコードは、コレクションの選択されたテンプレート レコードではありません。 **BestOfBreed** レコードは、コレクションの新規作成 Best of Breed レコードです。

Candidate Finder

Candidate Finder は、一連の潜在的なマッチを形成する候補レコードを取得します。 データベース検索は Transactional Match と連携し、検索インデックス検索は Transactional Match とは別に機能します。 また、Candidate Finder では、データの書式によっては、サスペクト レコード、候補レコード、またはその両方のレコードの名前や住所のパーシングが必要となる場合もあります。

また、Candidate Finder ではフルテキスト インデックス検索も可能で、さまざまな検索タイプ (次で始まる、次を含む、すべて含む、いずれかを含む、いずれも含まない、あいまい、パターン、直線距離、範囲、ワイルドカード) と条件 (すべて真、いずれかが真、いずれも真でない) を使用して、文字やテキストの単純および複雑な検索条件を容易に定義できます。

データベース オプション

[Candidate Finder] ダイアログを使って、潜在的なマッチ候補をデータベースから取得する SQL 文を定義したり、データベースから選択した列をデータフローで定義したフィールド名にマッピングすることができます。

表 9 : Candidate Finder データベース オプション

オプション名	説明/有効値
Finder タイプ	データベースの選択
接続	候補レコードが含まれるデータベースを選択します。 Management Console で設定されたいずれかの接続を選択できます。 リストにないデータベースに接続するには、Management Console でそのデータベースへの接続を設定してから Candidate Finder をいったん閉じ、再び開いて接続リストの内容を更新します。 **注 :** Enterprise Designer の [データフロー オプション] 機能を使用すると、接続名を実行時に公開して設定できます。
SQL 文	**SQL クエリの定義**（185ページ）の説明に従って、SQL 文をテキスト ボックスに入力します。
[フィールド マップ] タブ	**データベース列をステージ フィールドにマッピングする**（186ページ）の説明に従って、フィールド マッピング設定を選択します。
[プレビュー] タブ	サンプルのマッチ キーを入力して SQL SELECT 文またはインデックス クエリをテストするには、このタブをクリックします。

SQL クエリの定義

有効な SQL SELECT 文を **[Candidate Finder オプション]** ダイアログ ボックスのテキスト ボックスに入力できます。

注 : `Select *` は無効です。

例えば、Customer_Table というテーブルがデータベース内にあり、以下の列が含まれると仮定します。

- Customer_Table
- Cust_Name
- Cust_Address
- Cust_City
- Cust_State
- Cust_Zip

データベースからこれらの行をすべて取得するには、次のようなクエリを作成します。

```
SELECT Cust_Name, Cust_Address, Cust_City, Cust_State, Cust_Zip from
Customer_Table;
```

データベース内のすべての行にマッチするようなトランザクションを実行することは、実際にはほとんどないでしょう。 必要な候補レコードのみを返すには、置換変数を使って WHERE 節を追加します。 置換変数は、変数をサスペクト レコードの実データで置換するように Candidate Selection エンジンに指示するために使用できる特殊な表記です。

置換変数を使うには、${FieldName} のようにフィールド名をカッコで囲み、その直前にドル記号を付けます。 例えば、次のクエリを実行すると、サスペクト レコード内の PostalCode の値に Cust_Zip の値が一致するレコードのみが返されます。

```
SELECT Cust_Name, Cust_Address, Cust_City, Cust_State,Cust_Zip
FROM Customer_Table
WHERE Cust_Zip = ${PostalCode};
```

SQL 2000 では、データ タイプは Candidate Finder で使われるタイプと同一である必要があります。 JDBC ドライバは、WHERE 節で使用する Candidate Finder 入力変数 (Ex: ${MatchKey}) を nVarChar(4000) データ タイプに設定します。 データベース内でデータが VarChar データ タイプに設定されている場合、SQL Server はデータベースのインデックスを無視します。 インデックスが無視されると、パフォーマンスが低下します。 したがって、SQL 2000 では以下のクエリを使ってください。

```
SELECT Cust_Name, Cust_Address, Cust_City, Cust_State,Cust_Zip
FROM Customer_Table
WHERE Cust_Zip = CAST(${PostalCode} AS VARCHAR(255));
```

データベース列をステージ フィールドにマッピングする

データベース内の列名が [Component Field] 名に完全に一致する場合、列名は対応するステージ フィールドに自動的にマッピングされます。 名前が正確に一致しない場合は、選択フィールド (データベースの列) を使ってステージ フィールド (データフローで定義されたフィールド名) にマッピングする必要があります。

例えば、Customer_Table という名前のテーブルに次の列があるとします。

- Cust_Name
- Cust_Address
- Cust_City
- Cust_State
- Cust_Zip

これらのレコードをデータベースから取得するには、列名を、Transactional Match およびデータフロー内の他のコンポーネントで使われるフィールド名にマッピングする必要があります。 例え

ば、Cust_Address は AddressLine1 にマッピングし、Cust_Zip は PostalCode にマッピングします。

1. **[Candidate Finder オプション]** ダイアログの **[選択フィールド]** の下でドロップダウン リストを選択します。 次に、データベース列 Cust_Zip を選択します。
2. **[ステージ フィールド]** の下でドロップダウン リストを選択します。 次に、マッピングするフィールドを選択します。

例えば、Cust_Zip を Postal Code にマッピングするには、最初に選択フィールドで Cust_Zip を選択してから、対応するステージ フィールドの行で PostalCode を選択します。

別のフィールド マッピング方法

SQL クエリで特殊な表記を使ってマッピングを実行できます。列名の後に、マッピング先のフィールド名をカッコで囲んで記述します。 このように表記すると、選択フィールドは対応するステージ フィールドに自動的にマッピングされます。

例を次に示します。

```
select Cust_Name {Name}, Cust_Address {AddressLine1},
    Cust_City {City}, Cust_State {StateProvince},
    Cust_Zip {PostalCode}
from Customer
where Cust_Zip = ${PostalCode};
```

実行時の接続名の設定

接続名は、データフロー オプションとしてエクスポーズされている場合は、実行時に設定して引き渡すことができます。これによって、異なる接続名を使用してデータフローが実行できます。

1. Enterprise Designer で、Candidate Finder ステージを使用しているデータフローを開きます。
2. そのデータフローを保存してエクスポーズします。
3. **[編集] > [データフロー オプション]** に移動します。
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]** テーブルで、Candidate Finder を展開し、必要に応じてオプションを編集します。編集するオプションのチェック ボックスをオンにしてから、**[デフォルト値]** ドロップダウンの値を変更します。
5. オプション: **[オプション ラベル]** フィールドで、オプションの名前を変更します。
6. **[OK]** を 2 回クリックします。

検索インデックス オプション

Candidate Finder ダイアログでは、検索インデックス内で入力フィールド値を照合する簡易検索を実行したり、検索インデックスからマッチング候補を取得するマッチング ルールを作成するために高度な検索を実行することができます。

簡易検索インデックス オプション

表 10 : Candidate Finder オプション

オプション名	説明 / 有効な値
Finder タイプ	検索インデックスを選択します。
名前	Enterprise Designer の Advanced Matching 展開済みステージの下で **Write to Search Index** ステージを使用して作成された適切なインデックスを選択します。
開始レコード	検索結果の最初のレコード番号を入力します。デフォルト値は 1 です。
結果の最大数	インデックス検索で返す応答の最大数を入力します。デフォルトは 10 です。
マッチ数を返す	マッチ総数を返します。例えば、上の [結果の最大数] フィールドをデフォルトの "10" のままにしている場合、結果は 10 件しか返されません。しかし、このチェック ボックスをオンにすれば、処理中に得られたマッチ総数が出力フィールド TotalMatchCount に返されます。
インデックス検索タイプ	実行するインデックス検索のタイプを決定します。**[簡易検索]** を選択します。
インデックス フィールド	簡易検索による照合に使うインデックス フィールドを選択します。
入力フィールド	簡易検索による照合に使う入力フィールドを選択します。

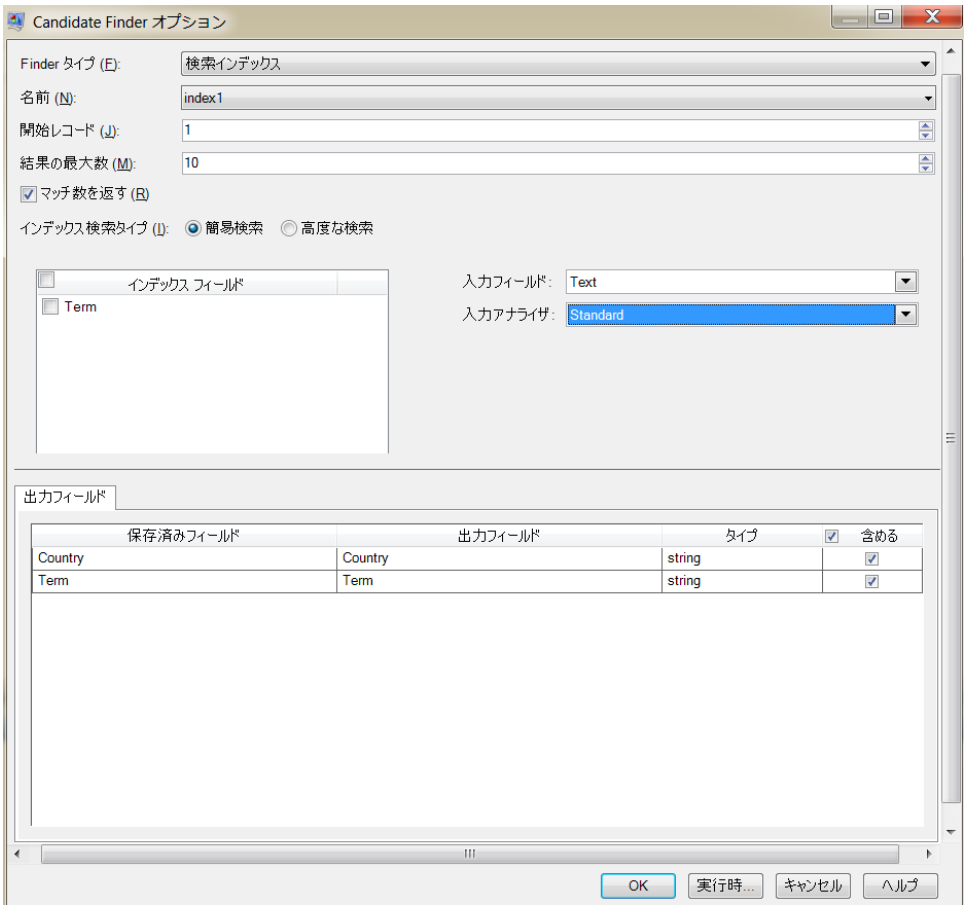
オプション名	説明 / 有効な値
入力アナライザ	

オプション名	説明 / 有効な値
	入力文字列のトークン化に使用するアナライザを指定します。次のいずれかです。 • [標準] — 空白分析とストップ ワード分析のスーパーセットを含むグラマーベースのトークン化機能を提供します。英単語を分割する英語の句読文字を理解すると、(ストップ ワード分析で) 無視する単語がわかり、小文字比較を実行することで、大文字と小文字を区別しない検索を技術的に実行できます例えば、文字列 "Pitney Bowes Software" の場合、3 つのトークン "Pitney"、"Bowes"、および "Software" が返されます。 • [空白] — トークンを空白で区切ります。英語テキストの単語分割点を空白と改行に基づいて理解するという点で、どちらかというと、標準分析のサブセットです。 • [ストップ ワード] — "the"、"and"、"a" などの冠詞を削除して、インデックスのサイズを減らし、パフォーマンスを高めます。 • [キーワード]—データのストリームから 1 つのトークンを作成します。例えば、文字列 "Pitney Bowes Software" の場合、1 つのトークン "Pitney Bowes Software" だけが返されます。 • [ロシア語] — ロシア語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"and"、"I"、"you" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [ドイツ語] — ドイツ語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [デンマーク語] — デンマーク語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"at"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [オランダ語] — オランダ語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [フィンランド語]—フィンランド語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"is"、"and"、"of" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [フランス語] — フランス語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [ハンガリー語] — ハンガリー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [イタリア語] — イタリア語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [ノルウェー語] — ノルウェー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [ポルトガル語] — ポルトガル語のインデックスと先行入力サービスをサポート

オプション名	説明 / 有効な値
	します。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [スペイン語] — スペイン語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [スウェーデン語]—スウェーデン語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。 • [ヒンディー語] — ヒンディー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"by"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
[出力フィールド] タブ	**[含める]** ボックスをオンにすると、出力に含める必要がある保存済みフィールドを選択できます。 **注 :** 入力フィールドがデータフローの以前のステージのもので、検索インデックスの保存フィールド名と同じ名前を持つ場合、入力フィールドの値によって出力フィールドの値は上書きされます。

以下に、簡易インデックス検索を使用した Candidate Finder オプション ステージの完成例を示します。

- 検索インデックスの **[名前]** は "CF_Index"
- **[開始レコード]** は 100 で、検索結果は 100 番目のレコードから開始
- **[結果の最大数]** は 25 で、結果は 25 件しか返されない
- **[合計マッチ数を返す]** のオプションが選択されており、これには表示上限の 25 件だけでなくすべてのレコードが含まれる
- 簡易**インデックス検索タイプ**
- 入力フィールド "City" との照合に使用される "City" の **[インデックス フィールド]**
- インデックス フィールド "City" との照合に使用される "City" **[入力フィールド]**
- フィールドを照合するドイツ語の入力アナライザ
- すべてのフィールドが出力に返されることを示すフィールド マップ。



高度な検索インデックス オプション

表 11 : Candidate Finder オプション

オプション名	説明 / 有効な値
Finder タイプ	検索インデックスを選択します。
名前	Enterprise Designer の Advanced Matching 展開済みステージの下で **Write to Search Index** ステージを使用して作成された適切なインデックスを選択します。
開始レコード	検索結果の最初のレコード番号を入力します。デフォルト値は 1 です。
結果の最大数	インデックス検索で返す応答の最大数を入力します。デフォルトは 10 です。

オプション名	説明 / 有効な値
マッチ数を返す	マッチ総数を返します。例えば、上の [結果の最大数] フィールドをデフォルトの "10" のままにしている場合、結果は 10 件しか返されません。しかし、このチェック ボックスをオンにすれば、処理中に得られたマッチ総数が出力フィールド TotalMatchCount に返されます。
インデックス検索タイプ	実行するインデックス検索のタイプを決定します。**[高度な検索]** を選択します。
[親の追加] ボタン	親オプションにアクセスします。
親オプション — 名前	親の名前を入力します。
親オプション — 検索方法	親がマッチかどうかを判定する方法を指定します。次のいずれかです。 すべて真 — すべての子がマッチと判定される場合、親はマッチと見なされます。この方法を選択すると、子の間に "AND" コネクタが作成されます。 いずれかが真 — 少なくとも 1 つの子がマッチと判定される場合、親はマッチと見なされます。この方法を選択すると、子の間に "OR" コネクタが作成されます。 いずれも真でない — いずれの子もマッチと判定されない場合、親はマッチと見なされます。この方法を選択すると、子の間に "NOT" コネクタが作成されます。
[子の追加] ボタン	子オプションにアクセスします。
子オプション — インデックス フィールド	検索インデックスを作成するフィールドを選択します。
子オプション — 検索タイプ	入力データを検索するか、インデックス データと照合するかを判定する検索/マッチング条件を指定します。検索では大文字と小文字がすべて区別されます。

オプション名	説明 / 有効な値
語または句が次で始まる	検索インデックス フィールドに含まれるテキストが、入力フィールドに含まれるテキストで始まるかどうかを判定します。 例えば、入力フィールド内のテキスト "tech" は、"Technical"、"Technology"、"Technologies"、"Technician"、"National University of Technical Sciences" などを含む検索インデックス フィールドとのマッチと見なされます。同様に、入力フィールド内の語句 "DEF Sof" は、"ABC DEF Software"、"DEF Software"、および "DEF Software India" を含む検索インデックス フィールドとのマッチと見なされますが、"Software DEF" や "DEF ABC Software" を含む検索インデックス フィールドとのマッチとは見なされません。
含む	検索インデックス フィールドに入力フィールドのデータが含まれるかどうかを判定します。この検索タイプでは、検索インデックス フィールドを検索するときに入力フィールドの単語の並びを考慮します。例えば、入力フィールドのデータが "Pitney" と "Pitney Bowes" の場合、これらの単語は検索インデックス フィールド "Pitney Bowes Software Inc." に含まれていると見なされます。
すべて含む	入力フィールドの英数字がすべて検索インデックス フィールドに含まれているかどうかを判定します。この検索タイプでは、検索インデックス フィールドを検索するときに入力フィールドの単語の並びを考慮しません。
いずれかを含む	入力フィールドのいずれかの英数字が検索インデックス フィールドに含まれているかどうかを判定します。
いずれも含まない	入力フィールドの英数字から成る単語がいずれも検索インデックス フィールドに含まれていないかどうかを判定します。

オプション名	説明 / 有効な値
あいまい	ある単語を別の単語に変換するために必要な削除、挿入、または置換の数に基づいて、英数字から成る 2 つの単語間の類似性を判断します。 **[編集の最大数]** パラメータを使用して、正しいマッチと見なすために許容する編集回数の制限を設定します。 • 0 — 削除、挿入、または置換を許容しません。入力フィールドのデータと検索インデックス フィールドのデータは同じである必要があります。 • 1 — 削除、挿入、または置換を 1 回だけ許容します。例えば、"Barton" を含む入力フィールドは、"Carton" を含む検索インデックス フィールドと一致します。 • 2 — 削除、挿入、または置換を 2 回だけ許容します。例えば、"Barton" を含む入力フィールドは、"Martin" を含む検索インデックス フィールドと一致します。 [あいまい] 検索タイプは、1 つの単語の検索でのみ使用されます。**[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックス フィールドを比較するときに、フィールド内の最初の単語のみを考慮します。例えば、インデックス フィールドが "Pitney"、入力フィールドが "Pitney Bowes" の場合、これらのフィールドは "Bowes" のある/なしによってマッチと見なされません。ただし、このボックスをオンにすると、"Bowes" は無視され、最初の単語である "Pitney" のみが考慮されるため、2 つの単語はマッチと見なされます。
数値	入力フィールドの数字がすべて検索インデックス フィールドに含まれているかどうかを判定します。 [数値] 検索タイプは、1 つの単語の検索でのみ使用されます。 **[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックス フィールドを比較するときに、フィールド内の最初の単語のみを考慮します。
パターン	入力フィールドのテキスト パターンが検索条件のテキスト パターンと一致しているかどうかを判定します。さらに、**[パターン文字列]** フィールドで、テキスト パターンを変更できます。例えば、入力フィールドに "nlm" が含まれ、定義されているパターンが "a*b?c" の場合、この入力フィールド は、"Neelam"、"nelam"、"neelum"、"nilam" などの単語と一致します。 [パターン] 検索タイプは、1 つの単語の検索でのみ使用されます。**[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックス フィールドを比較するときに、フィールド内の最初の単語のみを考慮します。

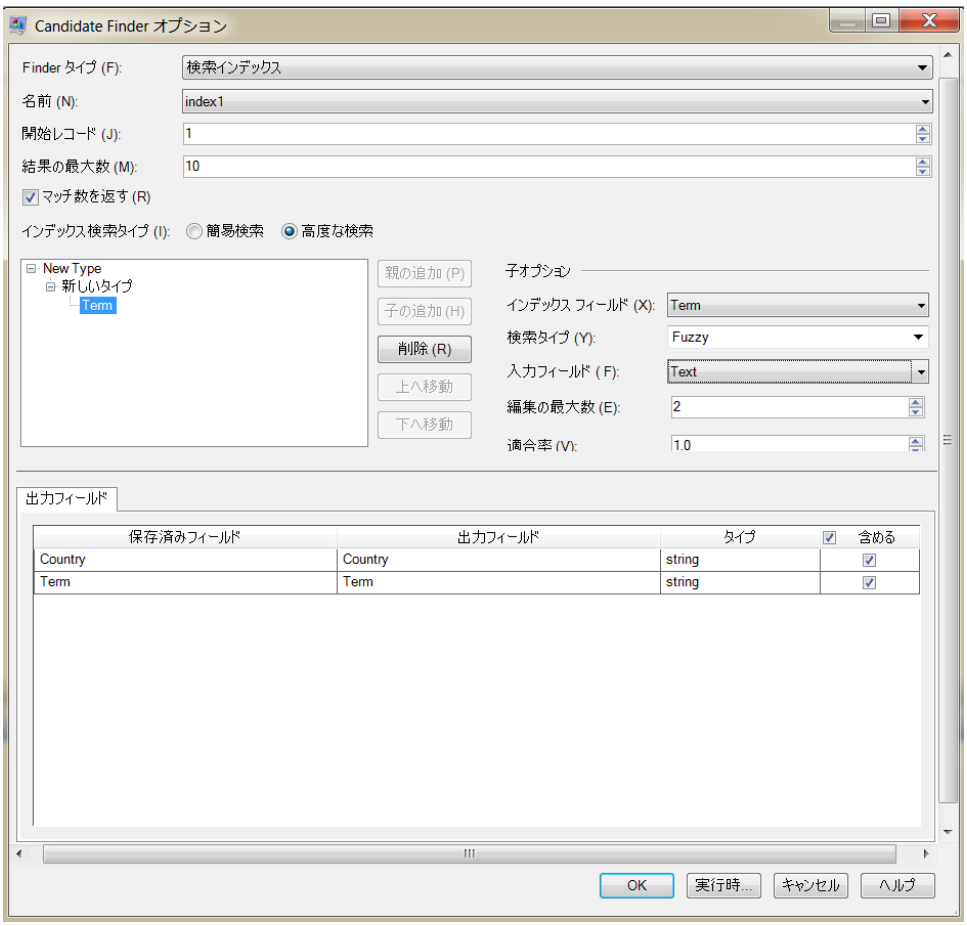
オプション名	説明 / 有効な値
近接度	入力フィールド内の各単語が一定の距離内にあるかどうかを判定します。 • インデックスで検索する入力を **[入力フィールド 1]** と **[入力フィールド 2]** に定義します。 • **[距離]** パラメータを使用して、[入力フィールド 1] と [入力フィールド 2] に指定した単語間の最大許容距離を指定します (この距離内にあれば一致と見なされます)。 例えば、この検索タイプを正しく使用すると、"Spectrum Technology Platform is a product of Pitney Bowes Software Inc." という文章を含む検索インデックス フィールドから互いに 10 単語以内の距離で "Spectrum" ([入力フィールド 1] の単語) と "Pitney" ([入力フィールド 2] の単語) を検索できます。 [直線距離] 検索タイプは、1 つの単語の検索でのみ使用されます。**[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックス フィールドを比較するときに、フィールド内の最初の単語のみを考慮します。
範囲	ある範囲内に含まれる語の包含的検索を実行します。範囲の指定には、下限フィールド (開始語) と上限フィールド (終了語) を使用します。検索インデックス フィールド内では、英数字から成るすべての単語は辞書の順序で配列されます。 • **[下限フィールド]** パラメータを使用して、開始語として使用するフィールドを選択します。 • **[上限フィールド]** パラメータを使用して、終了語として使用するフィールドを選択します。 例えば、20001 ([下限フィールド] で定義) ～ 20009 ([上限フィールド] で定義) までの郵便番号を検索すると、その範囲内の郵便番号を持つ住所がすべて返されます。 [範囲] 検索タイプは、1 つの単語の検索でのみ使用されます。**[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックス フィールドを比較するときに、フィールド内の最初の単語のみを考慮します。
ワイルドカード	1 つ以上の**ワイルドカード文字**を使用して検索します。 ワイルドカード文字を挿入する入力ファイル内での **[位置]** を選択します。 [ワイルドカード] 検索タイプは、1 つの単語の検索でのみ使用されます。**[余分な単語を無視]** をクリックすると、Candidate Finder は、入力フィールドとインデックスフィールドを比較するときに、フィールド内の最初の単語のみを考慮します。

オプション名	説明 / 有効な値
子オプション — 適合率	最大 100 までの数字を入力して、子フィールドの適合率を制御します。ブースト係数を高くすると、フィールドの適合性が高まります。例えば、[企業名] フィールドからの結果の適合性を他のフィールドからの結果の適合性よりも高くする場合は、**[インデックス フィールド名]** から "企業名" を選択し、"5" を入力します。 **注 :** ここで入力する数字は正の数でなければなりませんが、1 未満の数字、例えば "0.5" なら有効です。
空白を無視	クエリで、空の入力ファイル フィールドを考慮する場合は、このチェックボックスをオフにします。 **注 :** デフォルトでは、クエリによって空のフィールドが無視されます。
[出力フィールド] タブ	**[含める]** ボックスをオンにすると、出力に含める必要がある保存済みフィールドを選択できます。 **注 :** 入力フィールドがデータフローの以前のステージのもので、検索インデックスの保存フィールド名と同じ名前を持つ場合、入力フィールドの値によって出力フィールドの値は上書きされます。

以下に、高度なインデックス検索を使用した Candidate Finder オプション ステージの完成例を示します。

- 検索インデックスの **[名前]** は "CF_Index"
- **[開始レコード]** は 26 で、検索結果は 26 番目のレコードから開始
- **[結果の最大数]** は 10 で、結果は 10 件しか返されない
- **[合計マッチ数を返す]** のオプションが選択されており、これには表示上限の 10 件だけでなくすべてのレコードが含まれる
- 高度な**インデックス検索タイプ**
- **親**タイプ名 "State Match"
- 子タイプ名 "StateProvince" (インデックス フィールド名に基づく)
- **[あいまい]** 検索タイプで **[編集の最大数]** を 2 に設定 (編集回数が最大 2 回であれば正しいマッチと見なす)
- **[入力フィールド]** "StateProvince" を [インデックス フィールド] "StateProvince" の照合対象として使用
- 州データの適合性を高めるために **[適合率]** を 2.0 に設定

- InputKeyValue、AddressLine1、AddressLine2、StateProvince および PostalCode フィールドが出力に返されるが、FirmName または City フィールドは返されないことを示すフィールド マップ



実行時におけるオプションの設定

Candidate Finder の一部のオプションは、データフロー オプションとしてエクスポーズされている場合は、実行時に設定して引き渡すことができます。これによって、異なる設定を使用してデータフローが実行できます。Candidate Finder に対して使用可能なデータフロー オプションには、以下のものがあります。

- ConnectionName — 候補レコードを含むデータベース名。
- SearchIndexName — Candidate Finder データフローで使用される検索インデックス名。
- StartingRecord — 検索結果の最初のレコード番号。
- MaximumResults — インデックス検索で返す応答の最大数。
- ReturnMatchCount — マッチ総数。このフィールドは、MaximumResults フィールドには小さな値を入力したが、マッチ総数は知りたいという場合に便利です。

Candidate Finder オプションを実行時に定義するには

1. Enterprise Designer で、Candidate Finder ステージを使用しているデータフローを開きます。
2. そのデータフローを保存してエクスポーズします。
3. ［編集］ > ［データフローオプション］に移動します。
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]** テーブルで、Candidate Finder を展開し、必要に応じてオプションを編集します。編集するオプションのチェック ボックスをオンにしてから、**[デフォルト値]** ドロップダウンの値を変更します。
5. オプション: **[オプション ラベル]** フィールドで、オプションの名前を変更します。
6. **[OK]** を 2 回クリックします。

出力

表 12 : Candidate Finder 出力

フィールド名	書式	説明 / 有効な値
CandidateGroup	文字列	このフィールドは、サスペクト レコードとその候補のグループを表します。各サスペクト レコードには、CandidateGroup 番号が与えられます。サスペクトの候補には、同じ CandidateGroup 番号が与えられます。例えば、John Smith がサスペクト レコードで、候補レコードが John Smith と Jon Smth の場合、これらの 3 つのレコードはどれも同じ CandidateGroup 値を持ちます。
TotalMatchCount	文字列	このフィールドは、処理中に得られたマッチ総数を表します。
TransactionRecordType	文字列	次のいずれかです。 **Suspect** サスペクト レコードは、クエリへの入力として使われます。 **Candidate** 候補レコードは、クエリから結果として返されます。

Duplicate Synchronization

Duplicate Synchronization は、レコードのコレクションから、そのコレクション内のすべてのレコードの対応するフィールドにコピーするフィールドを指定します。フィールド データをコレクション内の別のレコードにコピーするときに従う必要があるルールを指定できます。処理が完了すると、コレクション内のレコードがすべて保持されます。

オプション

以下の表に、**Duplicate Synchronization** ステージのオプションを示します。

オプション名	説明 / 有効な値
グループ化方法	同期するレコードのグループを作成するのに使用するフィールドを指定します。データフローで以前、Interflow Match、Intraflow Match、Transactional Match などのマッチング ステージを使用していた場合は、CollectionNumber フィールドを選択して、そのマッチング ステージで作成されたコレクションをグループとして使用する必要があります。ただし、別のフィールドによってレコードをグループ化する場合は、そのフィールドを選択します。例えば、AccountNumber フィールドの値が同じであるレコードを同期する場合は、AccountNumber を選択します。
ソート	**[グループ化]** フィールドにフィールドを指定する場合は、このボックスをオンにして、選択したフィールドの値でレコードをソートします。このオプションは、デフォルトで有効になっています。

オプション名	説明 / 有効な値

詳細設定	ソート パフォーマンス オプションを指定するには、このボタンをクリックします。デフォルトでは、Management Console で指定されたソート パフォーマンス オプション (システム用のデフォルトのパフォーマンス オプション) が有効になります。システムのデフォルトのパフォーマンス オプションよりも優先する場合は、**[ソート パフォーマンス オプションをオーバーライド]** ボックスをオンにし、以下のフィールドに値を指定します。
	
	メモリ内レコードの上限値
	ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。
	
	注： 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。
	
	一時ファイルの最大数
	ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する一時ファイル数の増減がパフォーマンスに与える影響を確認する必要があります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。
	
	`(NumberOfRecords × 2) ÷ InMemoryRecordLimit = NumberOfTempFiles`
	
	一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。
	
	圧縮を有効にする
	一時ファイルをディスクに書き込むときに圧縮します。
	
	注： 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。それでも一般には、次の式で妥当なソート パフォーマンスが得られます。
	
	`(InMemoryRecordLimit × MaxNumberOfTempFiles ÷ 2) >= TotalNumberOfRecords`

ルール

Duplicate Synchronization ルールは、コレクション内の他のすべてのレコードにデータをコピーするレコードを決定します。

ルールを追加するには、ルール階層内のルールを選択し、**[ルールの追加]** をクリックします。

複数のルールを指定する場合は、各ルールの間で使用する論理演算子を選択する必要があります。新しいルールと前のルールの両方に合格しないと、条件を満たしていると判断しないように設定する場合は、**[And]** を選択します。前のルールまたは新しいルールのいずれか一方に合格すれば、条件を満たしていると判断するように設定する場合は、**[Or]** を選択します。

オプション	説明
フィールド名	レコードをフィルタリングするかどうかを判断するために値を評価するデータフローフィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 100 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに (バイト数が) 最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

アクション

アクションは、グループ内の他のレコードにコピーするフィールドを決定します。アクションを追加するには、Duplicate Synchronization 条件ツリー内のアクションを選択し、**[アクションの追加]** をクリックします。以下のオプションを使用してアクションを定義します。

オプション	説明
ソース タイプ	グループ内の他のレコードにコピーするデータのタイプを指定します。次のいずれかを選択します。 **フィールド** グループ内の他のレコードにフィールドの値をコピーする場合は、このオプションを選択します。 **文字列** グループ内の他のレコードに定数値をコピーする場合は、このオプションを選択します。
ソース データ	グループ内の他のレコードにコピーするデータを指定します。[ソース タイプ] が [フィールド] の場合は、グループ内の他のレコードに値をコピーするフィールドを選択します。[ソース タイプ] が [文字列] の場合は、グループ内の他のレコードにコピーする定数値を指定します。 **注：** ソース データが NULL 値である場合は、グループの他のレコードにコピーされません。他のレコードは元の値のままとなります。

オプション	説明
ディスティネーション	**[ソースデータ]**フィールドで指定したデータのコピー先となる、他のレコード内のフィールドを指定します。例えば、グループ内の他のすべてのレコードの AccountBalance フィールドにデータをコピーする場合は、AccountBalance を指定します。

Duplicate Synchronization ルールとアクションの例

この Duplicate Synchronization ルールとアクションは、マッチ スコアが 100 のレコードを選択し、グループ内の他のすべてのレコードに AccountNumber フィールドのアカウント番号をコピーします。

ルール
フィールド名: MatchScore
フィールド タイプ: 数値
演算子: 等しい
値タイプ: 文字列
値: 100

操作
ソース タイプ: フィールド
ソース データ: AccountNumber
デスティネーション: NewAccountNumber

Filter

Filter ステージでは、指定したルールに基づいて、レコードをレコードのグループに保持または削除します。

オプション

以下の表に、Filter ステージのオプションを示します。

オプション名	説明/有効値
グループ化方法	フィルタリングするレコードのグループを作成するのに使用するフィールドを指定します。Filter ステージでは、ステージの設定方法に応じて各グループのレコードを 1 つ以上保持します。データフローで以前、Interflow Match、Intraflow Match、Transactional Match などのマッチング ステージを使用していた場合は、CollectionNumber フィールドを選択して、そのマッチング ステージで作成されたコレクションをグループとして使用する必要があります。 ただし、別のフィールドによってレコードをグループ化する場合は、そのフィールドを選択します。 例えば、AccountNumber フィールドの値が同じであるレコードを残し、それ以外のレコードをフィルタリングしてすべて除外する場合は、AccountNumber を選択します。
ソート	**[グループ化]** フィールドにフィールドを指定する場合は、このボックスをオンにして、選択したフィールドの値でレコードをソートします。このオプションは、デフォルトで有効になっています。

オプション名	説明/有効値

詳細設定	ソート パフォーマンス オプションを指定するには、このボタンをクリックします。デフォルトでは、Management Console で指定されたソート パフォーマンス オプション (システム用のデフォルトのパフォーマンス オプション) が有効になります。システムのデフォルトのパフォーマンス オプションよりも優先する場合は、**[ソート パフォーマンス オプションをオーバーライド]** ボックスをオンにし、以下のフィールドに値を指定します。
	
	メモリ内レコードの上限値
	ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。
	
	注： 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。
	
	一時ファイルの最大数
	ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する一時ファイル数の増減がパフォーマンスに与える影響を確認する必要があります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。
	
	`(NumberOfRecords × 2) ÷ InMemoryRecordLimit = NumberOfTempFiles`
	
	一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。
	
	圧縮を有効にする
	一時ファイルをディスクに書き込むときに圧縮します。
	
	注： 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。それでも一般には、次の式で妥当なソート パフォーマンスが得られます。
	
	`(InMemoryRecordLimit × MaxNumberOfTempFiles ÷ 2) >= TotalNumberOfRecords`

オプション名	説明/有効値
返される重複レコード数の上限値	各グループから返されるレコードの最大数を指定します。 このオプションを 1 に設定すると、フィルタ ルールを定義して、グループごとに返す必要があるレコードを決定できます。 ルールを定義しないと、各コレクションの先頭のレコードが返されて、残りのレコードは破棄されます。 このモードでは、フィルタ ルールは保持するレコードを定義します。 例えば、グループ内でマッチ スコアが最も高いレコードを保持するルールを定義し、このオプションを 1 に設定すると、各グループ内でマッチ スコアが最も高いレコードが保持され、グループ内のそれ以外のレコードは破棄されます。 このオプションを 2 以上の値に設定すると、フィルタ ルールを指定できません。 **注:** 定義したルールの条件を満たすレコードがコレクションになければ、レコードはグループから返されません。
コレクションから重複を削除	フィルタ ルールを使用して、コレクションから削除するレコードを決定します。 コレクション内の残りのレコードは保持されます。 このオプションを選択するときは、ルールを定義する必要があります。 **注 :** グループ内にレコードが 1 つしかない場合は、フィルタ ルールは無視され、そのレコードが保持されます。

ルール オプション

フィルタ ルールでは、グループ内に保持または削除するレコードを決定します。 **[返される重複レコード数の上限値]** オプションを選択すると、ルールは保持するレコードを決定します。 **[コレクションから重複を削除]** オプションを選択すると、ルールはデータフローから削除するレコードを決定します。

ルールを追加するには、ルール階層内のルールを選択し、**[ルールの追加]** をクリックします。

複数のルールを指定する場合は、各ルールの間で使用する論理演算子を選択する必要があります。 新しいルールと前のルールの両方に合格しないと、条件を満たしていると判断しないように設定する場合は、**[And]** を選択します。 前のルールまたは新しいルールのいずれか一方に合格すれば、条件を満たしていると判断するように設定する場合は、**[Or]** を選択します。

> **注 :** Filter ステージでは条件を 1 つだけ使用できます。 フィルタ階層で [条件] を選択すると、ボタンがグレー表示になります。

オプション	説明
フィールド名	レコードをフィルタリングするかどうかを判断するために値を評価するデータフローフィールドの名前を指定します。
フィールド タイプ	フィールドのデータのタイプを指定します。次のいずれかです。 **数値以外** フィールドに数値以外のデータ (文字列データなど) が含まれる場合は、このオプションを選択します。 **数値** フィールドに数値データ (double、float など) が含まれる場合は、このオプションを選択します。

オプション	説明
演算子	フィールドの評価で使用する比較のタイプを指定します。次のいずれかです。 **含む** フィールドが指定された値を含むかどうかを確認します。例えば、"sailboat" には値 "boat" が含まれます。 **等しい** フィールドが指定された値に正確に一致するかどうかを確認します。 **次の値より大きい** フィールド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、数値のフィールドにのみ有効です。 **以上** フィールド値が指定された値に一致するか、またはそれよりも大きいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最高** フィールドの値をグループ内のすべてのレコードについてチェックし、最も大きな値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 100 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **空** フィールドに値がないことを確認します。 **空でない** フィールドに値が含まれているかどうかを確認します。 **次の値より小さい** フィールド値が指定された値よりも小さいかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **以下** フィールド値が指定された値以下であるかどうかを確認します。この演算子は、数値のフィールドにのみ有効です。 **最長** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに (バイト数が) 最長の値が格納されているレコードを特定します。例えば、グループ内に、"Mike" という値と "Michael" という値が含まれる場合、"Michael" という値が格納されているレコードが選択されます。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最低** グループ内のすべてのレコードのフィールドの値を比較し、フィールドに最も小さい値が格納されているレコードを特定します。例えば、グループ内でフィールドに値 10、20、30、および 100 が格納されている場合、フィールド値が 10 のレコードが選択されます。この演算子は、数値のフィールドにのみ有効です。最低値が複数のレコードにある場合は、1 つのレコードが選択されます。 **最多** フィールド値が、グループ内のレコードのこのフィールドにおいて最も多く出現する値であるかどうかを確認します。最多の値が複数存在する場合、アクションは実行されません。 **が等しくない** フィールド値が指定された値に一致しないことを確認します。

オプション	説明
値タイプ	フィールドの値と比較する値のタイプを指定します。次のいずれかです。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。 **フィールド** フィールドを別のデータフロー フィールドの値と比較する場合は、このオプションを選択します。 **文字列** フィールドを特定の値と比較する場合は、このオプションを選択します。
値	フィールドの値と比較する値を指定します。**[フィールド タイプ]** フィールドで [フィールド] を選択した場合は、データフロー フィールドを選択します。**[値タイプ]** フィールドで [文字列] を選択した場合は、比較で使用する値を入力します。 **注：** このオプションは、演算子として [最高]、[最低]、または [最長] を選択している場合は使用できません。

フィルタ ルールの例

このルールは、各グループで MatchScore フィールドの値が最も高いレコードを保持します。[演算子] が [最高] または [最低] の場合、**[値]** および **[値タイプ]** オプションは適用されないことに注意してください。

フィールド名 = MatchScore
フィールド タイプ = 数値
演算子 = 最高

このルールは、AccountNumber の値が "12345" のレコードを保持します。

フィールド名 = AccountNumber
フィールド タイプ = 数値
演算子 = が次に等しい
値タイプ = 文字列
値 = 12345

Interflow Match

Interflow Match は、2 つの入力レコード ストリーム内の類似するデータ レコード間でマッチを検出します。最初のレコード ストリームはサスペクト レコードのソースで、2 番目のストリームは候補レコードのソースです。

Interflow Match では、マッチ グループ条件 (マッチ キー等) を使用して、特定のサスペクト レコードと重複する可能性があるレコードのグループを識別します。

各候補はサスペクトと個々に照合され、マッチ ルールに基づいてスコアが付けられます。候補が重複の場合は、コレクション番号が割り当てられ、そのマッチ レコード タイプに重複が設定され、その候補が書き出されます。マッチしないユニークな候補を書き出すかどうかは、ユーザの選択に基づきます。Interflow Match によって、現在のマッチ グループ内の候補レコードがすべて使用されると、マッチするサスペクト レコードには、重複レコードに対応するコレクション番号が割り当てられます。また、マッチが見つからない場合は、サスペクトにコレクション番号 0 が割り当てられ、そのラベルにユニーク レコードが設定されます。

> **注：** Interflow Match は、サスペクト レコードと候補レコードの照合のみを行います。Intraflow Match のように、サスペクト レコードと他のサスペクト レコードとの照合は行いません。

重複の数を制限している場合に、現在のサスペクトでその制限を越えると、特定のサスペクトのマッチング処理が終了することがあります。

Express キーのマッチ結果を候補マッチ スコアに変換する方法は、マッチングのタイプ (Intraflow または Interflow) によって異なります。Interflow マッチングの場合、正常な Express キー マッチは常に 100 マッチスコアを候補に与えます。一方、Intraflow マッチングの場合、Express キー マッチの結果として候補に与えられるスコアは、その候補がマッチするレコードが他のサスペクトのマッチであるかどうかによって異なります。つまり、Express キーがサスペクトの重複であれば、マッチスコアは常に 100 になり、Express キーが他の候補の重複であれば、その候補のマッチスコア (必ずしも 100 ではない) が継承されます。

オプション

1. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

   > **注：** Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

2. マッチ キュー内のレコードをグループ化するのに使用するフィールドを選択するには、**[グループ化方法]** をクリックします。 Intraflow Match は、同じマッチ キューにある他のレコードに対してのみレコードのマッチングを試みます。
3. **[グループ化方法]** フィールドで選択したフィールドに基づいて、照合前に入力をソートするには、**[ソート]** ボックスを選択します。
4. ソート パフォーマンス オプションを追加で指定するには、**[詳細設定]** をクリックします。

メモリ内レコードの上限値 ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。

> **注：** 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。

一時ファイルの最大数 ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する一時ファイル数の増減がパフォーマンスに与える影響を確認する必要があります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。

```
(NumberOfRecords × 2) ÷ InMemoryRecordLimit =
NumberOfTempFiles
```

一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。

圧縮を有効にする 一時ファイルをディスクに書き込むときに圧縮します。

注： 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。それでも一般には、次の式で妥当なソート パフォーマンスが得られます。

```
(InMemoryRecordLimit × MaxNumberOfTempFiles ÷ 2) >=
TotalNumberOfRecords
```

5. **[Express マッチ有効]** をクリックして Express キー値の初期比較を実行し、2 つのレコードが一致すると判断できるかどうかを確認します。

   Express キー マッチは、実行する比較の回数を減らし、実行速度を改善するには便利なツールです。 あいまいな Express キーを使うと、誤検出のマッチが多数返されます。 Express キーは、MatchKeyGenerator で生成されるマッチ キーの一部として生成できます。 詳細については、**Match Key Generator**（223ページ）を参照してください。

   2 つのレコードが Express キーに正確にマッチする場合、候補は 100% の重複と見なされます。 2 つのレコードが Express キー値にマッチしない場合は、ルール ベースの方法で比較されます。

   Express キーを使った比較で候補がマッチしたかどうかを確認するには、**[ExpressKeyIdentified]** フィールドの値をチェックします。"Y" はマッチしたことを示し、"N" はマッチしなかったことを示します。 サスペクト レコードは **[ExpressKeyIdentified]** 値が常に "N" であることに注意してください。

6. **[最初のコレクション番号]** テキスト ボックスに、重複レコードのコレクション番号フィールドに割り当てる最初の番号を入力します。

   コレクション番号は、マッチ キュー内の各重複レコードを一意に識別するための値です。 ユニーク レコードには、コレクション番号 0 が割り当てられます。 各重複レコードには、**[最初のコレクション番号]** テキスト ボックスに指定された値で始まるコレクション番号が割り当てられます。

7. 次のオプションのいずれかを選択します。

オプション	説明
サスペクトをすべての候補と比較	このオプションを選択すると、マッチ グループに既に重複が見つかっている場合でも、サスペクトは同じマッチ グループ (グループ化オプション) のすべての候補と照合されます。 例: サスペクト - John Smith 候補 - Bill Jones 候補 - John Smith 候補 - John Smith この例では、サスペクト John Smith が両方の候補 John Smith と比較されます。 **[ユニークな候補を返す]** ボックスをオンにすると、マッチ グループ内のレコードのうち、ユニーク レコードと特定されたレコードが候補ポートから返されます。

オプション	説明
***n*検出後にサスペクトと候補の比較を停止**	このオプションを選択すると、サスペクトはマッチ グループ (グループ化オプション)内のすべての候補と比較されますが、ユーザが定義した数だけ重複が検出されると比較が停止します。 例えば、1 つの重複を検出した後、候補を停止し、次のデータを得たとします。 サスペクト - John Smith 候補 - Bill Jones 候補 - John Smith 候補 - John Smith この例では、サスペクト レコード John Smith は、最初の候補 John Smith が重複として特定されると、マッチ グループ内での比較を停止します。

8. **[分析用データを生成する]**をクリックしてマッチ結果を生成します。詳細については、マッチ結果の分析（117ページ）を参照してください。
9. **[ユニーク レコードにコレクション番号 0 を割り当て]**(デフォルトでオン)は、ユニーク レコードにコレクション番号としてゼロを割り当てます。このオプションをオフにすると、ユニーク レコードにゼロ以外のコレクション番号が生成されます。 ユニーク レコードのコレクション番号は、他のコレクション番号と連動して順に生成されます。 例えば、マッチング データフローで 5 つのレコードが検出され、最初の 3 つのレコードがユニークの場合、コレクション番号は下の最初のグループに示すように割り当てられます。 マッチング データフローで 5 つのレコードが検出され、最後の 2 つがユニークの場合、コレクション番号は下の 2 つ目のグループに示すように割り当てられます。

オプション	説明
コレクション番号	**レコード タイプ**
1	ユニーク
2	ユニーク
3	ユニーク
4	重複/サスペクト
4	重複/サスペクト

オプション	説明
コレクション番号	レコード タイプ
1	重複/サスペクト
1	重複/サスペクト
2	ユニーク
3	ユニーク
4	ユニーク

このボックスをオンのままにしておくと、データフローで検出されたユニーク レコードにデフォルトでゼロのコレクション番号が割り当てられます。

10. カスタム マッチング ルールを作成するには、マッチ ルールの作成（72ページ）を参照してください。
11. **[評価]**をクリックして、サスペクト レコードと候補レコードとの比較によるスコアを生成します。 詳細については、**Interflow Match**（212ページ）を参照してください。

出力

表 13 : Interflow Match 出力フィールド

フィールド名	説明 / 有効な値
CollectionNumber	重複レコードのコレクションを識別します。有効な値は 1 以上です。
ExpressMatchIdentified	マッチの取得に Express マッチ キーが使われたかどうかを示します。有効な値は Yes または No です。

フィールド名	説明 / 有効な値
HasDuplicates	レコードが別のレコードと重複するかどうかを示します。次のいずれかです。 **Y** レコードはサスペクト レコードであり、重複するレコードがあります。 **N** レコードはサスペクト レコードであり、重複するレコードはありません。 **D** レコードは候補レコードであり、サスペクト レコードと重複します。 **U** レコードは候補レコードですが、サスペクト レコードと重複しません。
InterflowSourceType	有効な値は input_port_0 または input_port_1 です。
MatchRecordType	コレクションに含まれるマッチ レコードのタイプを識別します。有効な値を次に示します。 **suspect** 他のレコードと重複する可能性があるとフラグが立てられた元の入力レコード。 **duplicate** 入力レコードと重複するレコード。 **unique** 重複がないレコード。
MatchScore	2 つのレコードの全体的なスコアを識別します。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。

注： Validate Address および Advanced Matching モジュールのステージでは、どちらも MatchScore フィールドを使用します。データフローの出力の MatchScore フィールドの値は、出力ステージに送られる前に最後に値を変更したステージによって決まります。データフローに Validate Address および Advanced Matching モジュールのステージが含まれ、各ステージの MatchScore 出力フィールドを確認したい場合は、Transformer ステージを使用して、MatchScore 値を他のフィールドにコピーしてください。例えば、Validate Address によって MatchScore という出力フィールドが作成され、Transformer ステージによって Validate Address の MatchScore フィールドが AddressMatchScore というフィールドにコピーされます。マッチャー ステージを実行すると、マッチャーから得た値が MatchScore フィールドに設定され、Validate Address から得た AddressMatchScore の値が引き渡されます。

Intraflow Match

Intraflow Match は、単一の入力ストリーム内の類似するデータ レコード間でマッチを検出します。データフローの他のステージで定義または作成されたフィールドに基づいて階層型のルールを作成できます。

オプション

1. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

   注： Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

2. マッチ キュー内のレコードをグループ化するのに使用するフィールドを選択するには、**[グループ化方法]** をクリックします。 Intraflow Match は、同じマッチ キューにある他のレコードに対してのみレコードのマッチングを試みます。
3. **[グループ化方法]** フィールドで選択したフィールドに基づいて、照合前に入力をソートするには、**[ソート]** ボックスを選択します。
4. ソート パフォーマンス オプションを追加で指定するには、**[詳細設定]** をクリックします。

メモリ内レコードの上限値 ソートでメモリ内に保持できるデータ行の最大数を指定します。この上限を越えると、ディスクにページングします。デフォルトでは、10,000 レコード未満のソートはメモリ内で行われ、10,000 レコードを越えるソートはディスク ソートとして実行されます。上限値は 100,000 レコードです。通常、メモリ内ソートはディスク ソートよりはるかに速いため、大部分のソートがメモリ内ソートとなり、大規模セットのみがディスクに書き出されるよう、この値を十分大きく設定してください。

注： 複数のジョブを同時並行で実行する環境では、**[メモリ内レコードの上限値]** の設定を増やすと、メモリ不足になる可能性が高くなります。

一時ファイルの最大数 ソート プロセスで使用できる一時ファイルの最大数を指定します。使用する一時ファイルの数を増やすと、パフォーマンスが向上する可能性があります。ただし、最適なファイル数は Spectrum™ Technology Platform を実行しているサーバーの構成に大きく依存します。さまざまな設定を試して、使用する

一時ファイル数の増減がパフォーマンスに与える影響を確認する必要があります。必要になる可能性がある一時ファイルの適切な数を計算するには、次の式を使用します。

```
(NumberOfRecords × 2) ÷ InMemoryRecordLimit =
NumberOfTempFiles
```

一時ファイルの最大数は 1,000 を超える値にはできないことに注意してください。

圧縮を有効にする 一時ファイルをディスクに書き込むときに圧縮します。

注 : 最適なソート パフォーマンスの設定は、サーバーのハードウェア構成によって異なります。それでも一般には、次の式で妥当なソート パフォーマンスが得られます。

```
(InMemoryRecordLimit × MaxNumberOfTempFiles ÷ 2) >=
TotalNumberOfRecords
```

5. **[Express マッチ有効]** をクリックして Express キー値の初期比較を実行し、2 つのレコードが一致すると判断できるかどうかを確認します。

   Express キーは、MatchKeyGenerator で生成されるマッチ キーの一部として生成できます。詳細については、**Match Key Generator**（223ページ）を参照してください。

6. **[最初のコレクション番号]** テキスト ボックスに、重複レコードのコレクション番号フィールドに割り当てる最初の番号を入力します。

   コレクション番号は、マッチ キュー内の各重複レコードを一意に識別するための値です。 ユニーク レコードには、コレクション番号 0 が割り当てられます。 各重複レコードには、**[最初のコレクション番号]** テキスト ボックスに指定された値で始まるコレクション番号が割り当てられます。

7. **[スライディング ウィンドウ]** をクリックして、このマッチング方法を有効にします。 スライディング ウィンドウの詳細については、**スライディング ウィンドウ マッチング方法**（221ページ） を参照してください。
8. **[分析用データを生成する]** をクリックしてマッチ結果を生成します。詳細については、**マッチ結果の分析**（117ページ）を参照してください。
9. **[ユニーク レコードにコレクション番号 0 を割り当て]** (デフォルトでオン) は、ユニーク レコードにコレクション番号としてゼロを割り当てます。このオプションをオフにすると、ユニーク レコードにゼロ以外のコレクション番号が生成されます。 ユニーク レコードのコレクション番号は、他のコレクション番号と連動して順に生成されます。 例えば、マッチング データフローで 5 つのレコードが検出され、最初の 3 つのレコードがユニークの場合、コレクション番号は下の最初のグループに示すように割り当てられます。 マッチング データフローで 5 つの

レコードが検出され、最後の 2 つがユニークの場合、コレクション番号は下の 2 つ目のグループに示すように割り当てられます。

オプション	説明
コレクション番号	**レコード タイプ**
1	ユニーク
2	ユニーク
3	ユニーク
4	重複/サスペクト
4	重複/サスペクト
コレクション番号	**レコード タイプ**
1	重複/サスペクト
1	重複/サスペクト
2	ユニーク
3	ユニーク
4	ユニーク

このボックスをオンのままにしておくと、データフローで検出されたユニーク レコードにデフォルトでゼロのコレクション番号が割り当てられます。

10. 他のオプションの変更の詳細については、マッチ ルールの作成（72ページ）を参照してください。
11. **[評価]** をクリックして、サスペクト レコードと候補レコードとの比較によるスコアを生成します。 詳細については、**Interflow Match**（212ページ）を参照してください。

デフォルトのマッチング方法

マッチャーは、ユーザが設定したグループ化方法 (マッチ グループ) を使用して、重複している可能性のあるレコードのグループを識別します。 次に、マッチャーはグループ内の各レコードを調べます。レコードが既存のサスペクトにマッチする場合、そのレコードはサスペクトの重複と見なされ、スコア、コレクション番号、およびマッチ レコード タイプ (重複) が割り当てられ、マッチから除外されます。 一方、レコードがマッチ グループ内の既存のサスペクトにマッチしなかった場合は、そのレコードが新しいサスペクトとなります。つまり、現在のマッチ グループに追加され、後続のレコードとマッチングできるようになります。 現在のマッチ グループ内のすべてのレコードを調べ終わると、マッチャーはすべてのサスペクトをマッチから除外し、マッチ レコード タイプにユニークというラベルを付け、コレクション番号 0 を割り当てます。 少なくとも 1 つの重複があるサスペクトのマッチ レコード タイプはサスペクトのままで、マッチした重複レコードと同じコレクション番号が割り当てられます。 最後に、マッチ グループ内のすべてのレコードが出力に書き出されます。 新しいマッチ グループが比較されます。

> **注：** デフォルトのマッチング方法では、同じマッチ グループ内のレコードのみが比較されます。

Express キーのマッチ結果を候補マッチ スコアに変換する方法は、マッチングのタイプ (Intraflow または Interflow) によって異なります。 Interflow マッチングの場合、正常な Express キー マッチは常に 100 マッチスコアを候補に与えます。 一方、Intraflow マッチングの場合、Express キー マッチの結果として候補に与えられるスコアは、その候補がマッチするレコードが他のサスペクトのマッチであるかどうかによって異なります。つまり、Express キーがサスペクトの重複であれば、マッチスコアは常に 100 になり、Express キーが他の候補の重複であれば、その候補のマッチスコア (必ずしも 100 ではない) が継承されます。

スライディング ウィンドウ マッチング方法

スライディング ウィンドウ アルゴリズムは、ウィンドウというあらかじめ定められたバッファ サイズを対応するデータ行で連続して埋めていくアルゴリズムです。 各行はウィンドウに追加され、既にウィンドウ内にある各項目と比較されます。 ある項目とのマッチと判断された場合、ドライバ レコード (ウィンドウに追加する新規項目) と候補 (既にウィンドウ内にある項目) の両方に同じグループ ID が付与されます。 ドライバ レコードがウィンドウ内に含まれるすべての項目と比較されるまで、この比較が続行されます。

新しいドライバをウィンドウに追加していくと、最終的にはあらかじめ定められた容量に達します。 その時点でウィンドウがスライドするため、スライディング ウィンドウという用語が使用されます。 スライディングとは、最新のドライバ レコードがウィンドウに追加されると、ウィンドウ バッファが削除され、ウィンドウ内の最古の項目が書き出されるということを意味しています。

出力

表 14 : Intraflow Match 出力

フィールド名	説明 / 有効な値
CollectionNumber	重複レコードのコレクションを識別します。有効な値は 1 以上です。
ExpressMatchIdentified	マッチの取得に Express マッチ キーが使われたかどうかを示します。有効な値は Yes または No です。
MatchRecordType	コレクションに含まれるマッチ レコードのタイプを識別します。有効な値を次に示します。 **suspect** 互いに重複する関連性にあるかどうかを確認するために、他のレコードと比較するレコード。1 コレクションにサスペクト レコードは 1 つのみ含まれます。 **duplicate** サスペクト レコードと重複するレコード。 **unique** 重複がないレコード。
MatchScore	2 つのレコードの全体的なスコアを識別します。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。

注 : Validate Address および Advanced Matching モジュールのステージでは、どちらも MatchScore フィールドを使用します。データフローの出力の MatchScore フィールドの値は、出力ステージに送られる前に最後に値を変更したステージによって決まります。データフローに Validate Address および Advanced Matching モジュールのステージが含まれ、各ステージの MatchScore 出力フィールドを確認したい場合は、Transformer ステージを使用して、MatchScore 値を他のフィールドにコピーしてください。例えば、Validate Address によって MatchScore という出力フィールドが作成され、Transformer ステージによって Validate Address の MatchScore フィールドが AddressMatchScore というフィールドにコピーされます。マッチャー ステージを実行すると、マッチャーから得た値が MatchScore フィールドに設定され、Validate Address から得た AddressMatchScore の値が 引き渡されます。

Match Key Generator

Match Key Generator は、レコードごとに非ユニーク キーを作成します。この非ユニーク キーは、潜在的な重複レコードのグループを特定するためにマッチング ステージで使用できます。マッチ キーを使用すると、レコードをマッチ キー別にグループ化し、各グループ内でのみレコードを比較できるので、マッチング プロセスが促進されます。

マッチ キーは、定義したルールを使用して作成され、入力フィールドから構成されます。指定する入力フィールドごとに、そのフィールドで実行されるアルゴリズムが選択されます。 その後、各アルゴリズムの結果を連結して、単一のマッチ キー フィールドが作成されます。

マッチ キーの作成に加え、後のデータフローの Intraflow Match ステージまたは Interflow Match ステージで使用する Express マッチ キーも作成できます。

複数のマッチ キーおよび Express マッチ キーを作成できます。

例えば、次のような入力レコードがあり、

名 - Fred
姓 - Mertz
郵便番号 - 21114-1687
性別コード - M

次のようなレコードのデータを組み合わせてマッチ キーを生成するマッチ キー ルールを定義したとします。

入力フィールド	開始位置	長さ
郵便番号	1	5
郵便番号	7	4
姓	1	5
名	1	5
性別コード	1	1

次のようなキーになります。

211141687MertzFredM

入力

入力はソース データの任意のフィールドです。

オプション

Match Key Generator のオプションを定義するには、**[追加]** ボタンをクリックします。**[マッチ キー フィールド]** ダイアログが表示されます。

> **注：** Enterprise Designer の [データフロー オプション] 機能を使用すると、Match Key Generator を実行時に公開して設定できます。

表 15 : Match Key Generator のオプション

オプション名	説明 / 有効な値
アルゴリズム	

オプション名	説明 / 有効な値
	マッチ キーの生成に使用するアルゴリズムを指定します。次のいずれかです。 **Consonant (子音)** 指定されたフィールドを、子音を削除して返します。 **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

オプション名	説明 / 有効な値
	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。 **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** 選択されているフィールドの指定部分を返します。
フィールド名	選択したアルゴリズムを適用してマッチ キーを生成するフィールドを指定します。例えば、LastName というフィールドを選択し、Soundex アルゴリズムを選択した場合、Soundex アルゴリズムが LastName フィールドのデータに適用されて、マッチ キーが生成されます。
開始位置	指定したフィールド内での開始位置を指定します。すべてのアルゴリズムで開始位置を指定できるとは限りません。
長さ	開始位置から含める文字の数を指定します。すべてのアルゴリズムで長さを指定できるとは限りません。
ノイズ文字の削除	ハイフン、空白、その他の特殊文字等、英数字以外の文字を入力フィールドからすべて削除します。
ソート入力	入力フィールド内の文字または語をすべてアルファベット順にソートします。 **文字** ユニーク ID を作成する前に、入力フィールドの文字値をソートします。 **語** ユニーク ID を作成する前に、入力フィールドの各語値をソートします。

複数のマッチ キー生成アルゴリズムを追加する場合は、**[上へ移動]** ボタンと **[下へ移動]** ボタンを使用して、アルゴリズムの適用順序を変更することができます。

Express マッチ キーの生成

[Express マッチ キーを生成] オプションを有効にし、**[追加]** をクリックして、後のデータフローの Intraflow Match ステージまたは Interflow Match ステージで使用する Express マッチ キーを定義します。

[Express マッチ キーを生成] オプションが有効で、**[Express マッチ キー有効]** オプションを下流の Interflow Match ステージまたは Intraflow Match ステージで選択すると、ここで作成した Express マッチ キーを使用して最初の照合が試行されます。2 つのレコードの Express マッチ キーが一致すると、レコードはマッチと見なされ、それ以上の処理は試行されません。レコードの Express マッチ キーが一致しない場合は、Interflow Match または Intraflow Match で定義されたマッチ ルールを使用してレコードの一致を調べます。

出力

表 16 : Match Key Generator 出力

フィールド名	説明/有効値
ExpressMatchKey	マッチ レベルを表す値。 Express マッチ キーが一致する場合、スコアは 100 です。 Express マッチ キーが一致しない場合、スコア 0 が返されます。
MatchKey	レコードを識別するために生成されるキー。

Private Match

Private Match では、機密情報を漏洩することなく、2 つのエンティティでデータセットを比較し、共通レコードを特定することができます。例えば、2 つの企業が共同マーケティング キャンペーンの立ち上げを企画しているとします。両社は、顧客情報を含むそれぞれ独自のデータベースを保有しており、ターゲットを絞ったキャンペーンを展開するために、両方の企業を利用する顧客を特定したいと考えています。しかし、データ セキュリティを確保し、また、プライバシー規則を遵守するために、データベースを互いに共有したり、第三者に提供してマッチングを行ってもらったりすることはしたくありません。Private Match 機能を利用すれば、セキュリティやプライバシーに関する規則に違反することなく、2 つのデータベースを互いにマッチングすることができます。

Private Match は、次の 3 つのモードのいずれかで使用されます。

- 暗号化モード — 最初のユーザが自分のデータを入力すると、インデックス フィールドとマッチ フィールドが抽出されて暗号化されます。公開鍵と、最初のユーザのデータを含む変位テーブルが、2人目のユーザ用に生成されます。また、最初のユーザが後で使用するための秘密鍵が生成されます。
- プライベートマッチ — 2人目のユーザが、自分のデータと、最初のユーザの暗号化データを入力し、公開鍵と変位テーブルを提供して、マッチングを行います。マッチしたデータを含むファイルが生成され、最初のユーザに送信されます。
- 復号化モード — 最初のユーザが、2人目のユーザの暗号化データを入力し、秘密鍵を提供して、両方のユーザのデータのマッチしたインデックスを含む出力を生成します。

暗号化機能 (暗号化モード) を使用することにより、マッチング実行時のセキュリティが維持され (プライベートマッチ)、復号化機能によって、マッチしたデータの出力が表示されます (復号化モード)。生成されてユーザ間で共有されるファイルはすべて暗号化され、読み取ることはできません。

入力

Private Match ステージの入力要件は、実行するタスクによって異なります。

- 暗号化モード — 最初のユーザのデータを含むファイルを、入力ポートに接続する必要があります。 テキスト ファイルやデータベースに加え、ほぼすべての種類のソース ファイルが使用できます。
- プライベートマッチモード — 2 人目のユーザのデータを含むファイルを、最初の入力ポートに接続する必要があります。こちらも、テキスト ファイルやデータベースに加えて、ほぼすべての種類のソース ファイルが使用できます。 最初のユーザによって生成された暗号化データを、2 つめの入力ポートに接続する必要があります。
- 復号化モード — 2 人目のユーザによって生成された出力ファイル。

オプション

Private Match ステージのオプションは、実行するタスクによって異なります。

> **注：** バージョン 10.x からバージョン 11.0 以降にアップグレードし、秘密鍵を作成している場合は、それらの鍵を再生成する必要があります。バージョン 10.x で生成されたキーが、バージョン 11.0 以降とは互換性を持たないためです。

暗号化モード

1. **[暗号化]** 操作を選択します。
2. ファイル内の各レコードに一意の ID を与える **[インデックス フィールド]** を選択します。一意の ID は数値である必要があります。
3. 2 人目のユーザのデータとのマッチングに使用する **[マッチ フィールド]** を選択します。

4. ジョブ実行時に生成される **[公開鍵]** ファイルのパスと名前を指定します。
5. ジョブ実行時に生成される **[キー ストア]** ファイルのパスと名前を指定します。
6. ジョブ実行時に生成される **[変位テーブル]** ファイルのパスと名前を指定します。
7. 2 人目のユーザに送信される出力ファイル内の暗号化データを含む **[出力列]** の名前を入力します。
8. **[OK]** をクリックします。

プライベートマッチ

1. **[プライベートマッチ]** 操作を選択します。
2. ファイル内の各レコードに一意の ID を与える **[インデックス フィールド]** を選択します。一意の ID は数値である必要があります。
3. 最初のユーザのデータとのマッチングに使用する **[マッチ フィールド]** を選択します。
4. 暗号化モードの手順のステップ 7 で最初のユーザが入力した、**[暗号化されたデータ フィールド]** の名前を入力します。
5. **[公開鍵]** ファイルに移動します。
6. **[変位テーブル]** ファイルに移動します。
7. 最初のユーザに送信される出力ファイル内の暗号化データを含む **[出力列]** の名前を入力します。

復号化モード

1. **[復号化]** 操作を選択します。
2. プライベートマッチ モードの手順のステップ 7 で入力した、**[暗号化されたデータ フィールド]** を選択します。
3. **[キー ストア]** ファイルに移動します。
4. 出力ファイル内の復号化データを含む **[出力列]** を選択します。このフィールド内のデータの形式は、次に示すように、最初のユーザのデータのマッチしたインデックスと、2 人目のユーザのデータのマッチしたインデックスを、カンマ文字 (,) で区切ったものとなります。

```
User1Data,User2Data
```

出力

Private Match ステージの出力要件は、実行するタスクによって異なります。

- 暗号化モード — 最初のユーザの暗号化データを含む、Write to File ステージによって生成されたファイル。
- プライベートマッチモード — マッチ結果に関する暗号化された情報を含む、Write to File ステージによって生成されたファイル。

- 復号化モード — 両方のユーザのデータのマッチしたインデックスを含む、Write to File ステージによって生成されたファイル。

Transactional Match

Transactional Match では、サスペクト レコードを、Candidate Finder ステージから返される候補レコードと照合します。Transactional Match では、マッチング ルールを使用して、サスペクト レコードを、同じ候補グループ番号 (Candidate Finder で割り当てられる番号) を持つすべての候補レコードと比較して、重複を識別します。候補レコードが重複の場合は、コレクション番号が割り当てられ、そのマッチ レコード タイプに重複が設定され、その候補レコードが書き出されます。グループ内のマッチしない候補にはコレクション番号 0 が割り当てられ、そのラベルにユニークが設定され、その候補が書き出されます。

> **注 :** Transactional Match は、サスペクト レコードと候補レコードの照合のみを行います。Intraflow Match のように、サスペクト レコードと他のサスペクト レコードとの照合は行いません。

Transactional Match は Candidate Finder と組み合わせて使用します。Candidate Finder の詳細については、**Candidate Finder**（184ページ）を参照してください。

オプション

1. **[ロードするマッチ ルール]** フィールドで、定義済みのいずれかのマッチ ルールを選択します。このマッチ ルールはそのまま使用することも、必要に応じて変更することもできます。 定義済みのいずれかのマッチ ルールを出発点として使用せずに、新しいマッチ ルールを作成する場合は、**[新規作成]** をクリックします。 カスタム ルールは、データフローで 1 つだけ使用できます。

   > **注 :** Enterprise Designer の [データフロー オプション] 機能を使用すると、マッチ ルールを実行時に公開して設定できます。

2. ステージからの出力にユニークな候補レコードを含める場合は、**[ユニークな候補を返す]** を選択します。
3. マッチ分析ツールを使用してデータフローの結果を分析する場合は、**[分析用データを生成する]** を選択します。 詳細については、**マッチ結果の分析**（117ページ）を参照してください。
4. 他のオプションの変更の詳細については、**マッチ ルールの作成**（72ページ）を参照してください。
5. **[評価]** をクリックして、サスペクト レコードと候補レコードとの比較によるスコアを生成します。 詳細については、**Interflow Match**（212ページ）を参照してください。

出力

表 17 : Transactional Match 出力

フィールド名	説明 / 有効な値
HasDuplicates	レコードが別のレコードと重複するかどうかを示します。次のいずれかです。 **Y** レコードはサスペクト レコードであり、重複するレコードがあります。 **N** レコードはサスペクト レコードであり、重複するレコードはありません。 **D** レコードは候補レコードであり、サスペクト レコードと重複します。 **U** レコードは候補レコードですが、サスペクト レコードと重複しません。
MatchRecordType	コレクションに含まれるマッチ レコードのタイプを識別します。有効な値を次に示します。 **Suspect** 他のレコードと重複する可能性があるとフラグが立てられた元の入力レコード。 **Duplicate** 入力レコードと重複するレコード。 **Unique** 重複がないレコード。
MatchScore	2 つのレコードの全体的なスコアを識別します。有効な値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。
MatchInfo.<RuleName>.IsMatch	ルールのマッチ状態を `True` または `False` で表示します。`True` は一致を表し、`False` は一致するものがなかったことを示します。この情報は、**[Transactional Match オプション]** 画面で **[詳細なマッチ情報を返す]** オプションを選択した場合に表示されます。 **注 :** このフィールドの値は、定義された親ルール設定に基づいて子ノードのマッチ状態から導出されます。

フィールド名	説明 / 有効な値
MatchInfo.<RuleName>.Score	ルールのマッチ スコアを表示します。とり得る値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。この情報は、**[Transactional Match オプション]** 画面で **[詳細なマッチ情報を返す]** オプションを選択した場合に表示されます。 **注：** このフィールドのスコアは、定義された親ルール設定に基づいて子ノードのスコアから導出されます。
MatchInfo.<RuleName>.<RuleNodeName>.IsMatch	ルール階層に含まれる各ノードのマッチ状態を `True` または `False` で表示します。`True` は一致を表し、`False` は一致するものがなかったことを示します。`RuleNodeName` は変数であり、実際のマッチ ルールに含まれる階層ノード名で置き換えられます。 **[Transactional Match オプション]** 画面で **[詳細なマッチ情報を返す]** オプションを選択した場合は、ルール階層に含まれる各ノードでこのフィールドが出力されます。
MatchInfo.<RuleName>.<RuleNodeName>.Score	ルール階層に含まれる各ノードのマッチ スコアを表示します。`RuleNodeName` は変数であり、実際のマッチ ルールに含まれる階層ノード名で置き換えられます。 **[Transactional Match オプション]** 画面で **[詳細なマッチ情報を返す]** オプションを選択した場合は、ルール階層に含まれる各ノードでこのフィールドが出力されます。 とり得る値は 0 ～ 100 です。0 は精度の低いマッチを意味し、100 は完全一致を意味します。

注： Validate Address および Advanced Matching モジュールのステージでは、どちらも MatchScore フィールドを使用します。データフローの出力の MatchScore フィールドの値は、出力ステージに送られる前に最後に値を変更したステージによって決まります。データフローに Validate Address および Advanced Matching モジュールのステージが含まれ、各ステージの MatchScore 出力フィールドを確認したい場合は、Transformer ステージを使用して、MatchScore 値を他のフィールドにコピーしてください。例えば、Validate Address によって MatchScore という出力フィールドが作成され、Transformer ステージによって Validate Address の MatchScore フィールドが AddressMatchScore というフィールドにコピーされます。マッチャー ステージを実行すると、マッチャーから得た値が MatchScore フィールドに設定され、Validate Address から得た AddressMatchScore の値が引き渡されます。

Write to Search Index

Write to Search Index では、ステージに入力されるデータに基づいてフルテキスト インデックスを作成できます。このデータを専用検索インデックスに取り込むと、他のSpectrum™ Technology Platformステージからのインデックスに対する検索を行うときに応答がより迅速に得られます。大量の自由形式テキスト データを検索または分類する場合や、インタラクティブなテキストベース クエリを大量にサポートする場合は、リレーショナル データベースよりフルテキスト検索インデックスが適しています。

Write to Search Index では、分析機能を使用して、入力テキストをトークンと呼ばれるインデックス要素に細分します。その後、それらのトークンから検索インデックス語を抽出します。使用する分析のタイプ (入力テキストをトークンに分割する方法) によって、後でそのテキストを検索する方法は異なります。例えば、*キーワード*分析は常に厳密なマッチングを実行して文字列全体を 1 つのトークンとしてトークン化しますが、*標準*分析は文字列を分割してトークンを作成します。*ストップ ワード*分析は、よりいっそう洗練化されており、"a" や "the" のような冠詞を文字列から削除したうえでトークン化を行うため、インデックス サイズが縮小されます。

検索インデックスではリアルタイムに近い機能がサポートされており、インデックスの更新はほぼ即座に行われます。検索インデックスを使用するステージを閉じてから作成し直す必要はありません。

コマンド ラインで実行可能な検索インデックス関連のタスクの詳細については、『管理ガイド』の「管理ユーティリティ」セクションの「検索インデックス」を参照してください。

オプション

1. Enterprise Designer で、キャンバスの Write to Search Index ステージをダブルクリックします。
2. インデックスの **[名前]** を入力します。
3. **[書き出しモード]** を選択します。インデックスを再生成する際に、新しいデータが既存のデータに与える影響について、オプションを使用して指定できます。
   - **作成または上書き** — 新しいデータが既存のデータを上書きし、既存のデータはインデックスに存在しなくなります。
   - **更新または追加** — 新しいデータが既存のデータを上書きし、以前は存在しなかった新しいデータがインデックスに追加されます。
   - **追加** — 新しいデータが既存のデータに追加され、既存のデータはそのまま残されます。
   - **削除** — 選択されたフィールドのデータが、検索インデックスから削除されます。
4. **キー フィールド**を、レコードに対して行う操作が**更新または追加**と**削除**のどちらかに基づいて選択します。

- **[作成または上書き]** モードの場合は、**キー フィールド**を (分散環境で使用される) Elastic 検索インデックスに対してユニークなものにする必要があります。このフィールドを空白のままにしている場合は、重複の有無に関係なく、すべてのレコードがインデックスに保存されます。ただし、このインデックス上では更新、追加、削除など、どのような書き込み操作も行えなくなります。次の表は、**キー フィールド**が Lucene および Elastic 検索インデックスに対して非ユニークである場合のインデックス作成動作を説明したものです。

書き込みモード	キー フィールド	Lucene 検索インデックス	Elastic 検索インデックス
作成または上書き	同じ**キー フィールド**を持つ重複レコード	すべてのレコードが保存されています。 **注:** 同じキー フィールドを持つ重複レコードは、更新操作を行うとすぐに上書きされます。	同じキー フィールドを持つすべての重複レコードが**上書き**されます。
更新または追加	同じ**キー フィールド**を持つ重複レコード	重複するものは上書きされます。	重複するものは上書きされます。

5. 検索インデックスの作成時に一括で確定するレコード数を指定するには、**[一括確定]** ボックスをオンにします。次に、**[バッチ サイズ]** フィールドにレコード数を入力します。デフォルト値は 5000 です。
6. 作成する **[分析]** を選択します。
   - [標準] — 空白分析とストップ ワード分析のスーパーセットを含むグラマーベースのトークン化機能を提供します。英単語を分割する英語の句読文字を理解すると、(ストップ ワード分析で) 無視する単語がわかり、小文字比較を実行することで、大文字と小文字を区別しない検索を技術的に実行できます例えば、文字列 "Pitney Bowes Software" の場合、3 つのトークン "pitney"、"bowes"、および "software" が返されます。標準分析とキーワード分析の比較については、**標準分析とキーワード分析**（239ページ）を参照してください。
   - [空白] — トークンを空白で区切ります。英語テキストの単語分割点を空白と改行に基づいて理解するという点で、どちらかというと、標準分析のサブセットです。
   - [ストップ ワード] — "the"、"and"、"a" などの冠詞を削除して、インデックスのサイズを減らし、パフォーマンスを高めます。
   - [キーワード] — データのストリームから 1 つのトークンを作成し、そのままの状態を維持します。例えば、文字列 "Pitney Bowes Software" の場合、1 つのトークン "Pitney Bowes

Software" だけが返されます。標準分析とキーワード分析の比較については、**標準分析とキーワード分析**（239ページ）を参照してください。

- [ロシア語] — ロシア語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"and"、"I"、"you" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [ドイツ語] — ドイツ語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [デンマーク語] — デンマーク語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"at"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [オランダ語] — オランダ語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [フィンランド語] — フィンランド語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"is"、"and"、"of" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [フランス語] — フランス語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [ハンガリー語] — ハンガリー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [イタリア語] — イタリア語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [ノルウェー語] — ノルウェー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [ポルトガル語] — ポルトガル語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [スペイン語] — スペイン語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。
- [スウェーデン語] — スウェーデン語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"the"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。

- [ヒンディー語] — ヒンディー語のインデックスと先行入力サービスをサポートします。多くのストップ ワードもサポートし、"by"、"and"、"a" などを除外してインデックスのサイズを減らし、パフォーマンスを高めます。

7. スキーマをサーバーから再ロードするには、**[スキーマを再ロード]** をクリックします。

   注 : フィールド名を変更するには、**[フィールド]** 列に新しい名前を直接入力します。ただし、**[ステージ フィールド]** の名前や **[タイプ]** は変更できません。

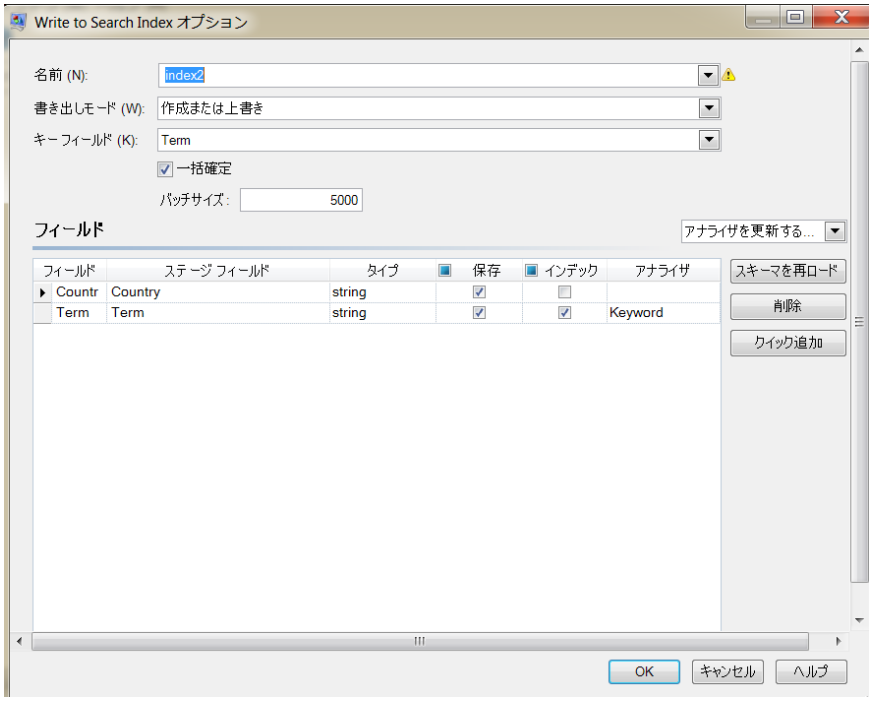
8. 入力ソースからのフィールドの追加/削除を選択的に行うには、**[クイック追加]** をクリックします。**[クイック追加]** ポップアップ ウィンドウに、入力ソースからのすべてのフィールドのリストが表示されます。追加するフィールドを選択し、**[OK]** をクリックします。
9. データを保存するフィールドを選択します。例えば、住所の入力ファイルを使用して、[郵便番号] フィールドのみにインデックスを作成し、残りのフィールド ([住所 1]、[都市]、[州] など) を保存すると、インデックス検索でマッチが見つかったときに住所全体が返されます。
10. 検索クエリのインデックスにデータを追加するフィールドを選択します。

    注 : 特定のフィールドを削除する場合は、それらのフィールドを選択し、**[削除]** をクリックします。

11. 必要に応じて、フィールドに適用する分析を変更して、[分析] フィールドで選択した項目とは別のものを使用してください。
12. **[OK]** をクリックします。

以下に、Write to Search Index オプション ステージの完成例を示します。

- "SearchIndex" という名前
- [作成または上書き] の書き込みモード
- 一括確定数は 2000 レコード
- 標準分析の使用
- 入力ファイルに含まれるフィールドの一覧
- インデックス データと共に保存されるフィールドの一覧この例の場合、保存されないのは AddressLine2 のみです。
- インデックスを構成するフィールドの一覧。この例の場合、インデックス化されないのは AddressLine2 のみです。
- [PostalCode] フィールドでのキーワード分析の使用



出力

Write to Search Index には、処理できず検索インデックスに追加できなかったレコードのデータを収集するための**エラー ポート**という 1 つの出力ポートがあります。エラー ポートからシンク ステージに送られるレコードは、形式に誤りがあると考えられます。

入力レコードの**キー フィールド**の値が空白または NULL である場合、レコードはエラー ポートに転送されます。スタンドアロン検索をサポートする検索インデックス エンジン (従来のインデックス エンジン) の場合、エラー ポートは更新と削除の操作に対してしか機能しません。一方、分散をサポートする検索インデックス エンジンの場合は、作成、更新、削除、追加の 4 つすべての操作に対して、形式に誤りのあるレコードがエラー ポートに収集されます。

検索インデックスの管理

検索インデックスの管理ツールを使用すると、以下の作業を実行できます。

- 既存の検索インデックス内でのフィールドの追加/削除
- 1 つ以上の検索インデックスの削除

既存の検索インデックスに対するフィールドの追加/削除を行うには:

1. **[ツール] > [検索インデックスの管理]** を選択します。既存の検索インデックスを示す **[検索インデックスの管理]** ポップアップ ウィンドウが表示されます。
2. 変更するインデックスを選択し、**[変更]** をクリックします。その検索インデックス内のすべてのフィールドのリストと詳細情報を示す **[インデックスの変更]** ページが表示されます。

3. フィールドを削除するには、リストからそのフィールドを選択し、**[フィールドを削除]** をクリックします。
4. リストに新しいフィールドを追加するには、**[フィールドの追加]** をクリックします。**[フィールドの追加]** ポップアップ ウィンドウが表示されます。このウィンドウでは、以下の詳細情報を追加できます。
   - **フィールド名**: フィールドの名前を入力します。
   - **タイプ**: フィールド タイプを選択します。選択肢として string、integer、long、float、double があります。
   - **インデックス**: 検索クエリ用のインデックスに追加する場合は、このチェック ボックスをオンにします。
   - **保存**: フィールドを保存するには、このチェック ボックスをオンにします。
   - **分析**: 選択肢のリストから、フィールドのインデックス作成に使用する分析を選択します。

     注： このオプションは、このフィールドのインデックス作成を選択した場合にのみ有効になります。

既存の検索インデックスを削除するには:

1. **[ツール] > [検索インデックスの管理]** を選択します。
2. 削除する検索インデックスを選択します。
3. **[削除]** をクリックします。
4. **[閉じる]** をクリックします。

注： 管理ユーティリティを使用して検索インデックスを削除することもできます。そのコマンドは `index delete --n IndexName` です。ここで、"IndexName" は削除するインデックスの名前です。

標準分析とキーワード分析

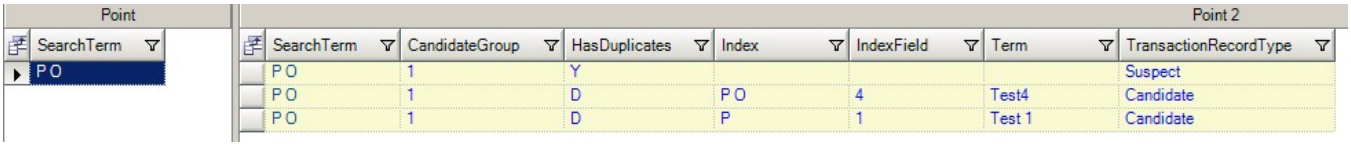
標準分析とキーワード分析のどちらかを選択する前に、これら 2 つの分析の動作に以下の違いがあることに注意する必要があります。

- **標準分析**: 文字列を分割してトークン化します。また、そのすべてのトークンを小文字に変換します。
- **キーワード分析**: 文字列全体をトークン化し、その状態を維持します (大文字/小文字の変更は行いません)。

例:

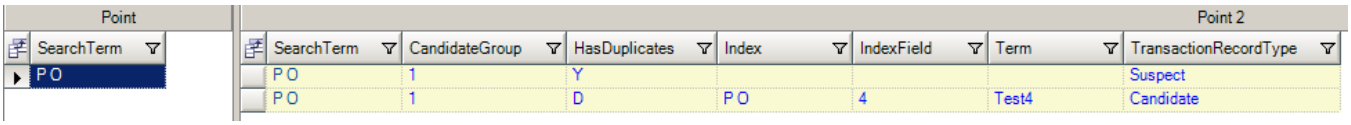
contains any (いずれかを含む) アルゴリズムの例を取り上げます。ここでは、入力が P O であるとします。この場合、**標準**分析は P O レコードと P レコードの両方を返しますが、**キーワード**分析は P O レコードのみを返します (以下を参照)。

標準分析による検索結果:



Point							Point 2
SearchTerm	SearchTerm	CandidateGroup	HasDuplicates	Index	IndexField	Term	TransactionRecordType
P O	P O	1	Y				Suspect
	P O	1	D	P O	4	Test4	Candidate
	P O	1	D	P	1	Test 1	Candidate

キーワード分析による検索結果:



Point							Point 2
SearchTerm	SearchTerm	CandidateGroup	HasDuplicates	Index	IndexField	Term	TransactionRecordType
P O	P O	1	Y				Suspect
	P O	1	D	P O	4	Test4	Candidate

Business Steward モジュール

Business Steward モジュール

Business Steward モジュールは、例外レコードを特定および解決することのできる機能の集合です。例外レコードとは、Spectrum™ Technology Platformで適切に処理できなかったレコードです。例外レコードは、データ スチュワード (data steward) が手動で確認する必要があります。例外の例としては、次のものがあります。

- 住所検証エラー
- ジオコーディング エラー
- 信頼性の低いマッチング
- マージ/統合の結果

Business Steward モジュールでは、ブラウザベースのツールを使用して例外レコードを確認できます。手動で修正され、承認された例外レコードは、Spectrum™ Technology Platformデータ品質処理に再度投入することができます。

Business Steward モジュール内で行われる操作の多くは、ユーザ アクティビティを記録する Management Console ツールである監査ログに反映されます。このログには、以下の操作が含まれます。

- Write Exceptions ステージでの例外の追加
- Read Exceptions ステージおよび Business Steward Portal Manage Exceptions ページでの例外の削除
- Business Steward Portal Exception Manager での例外の割り当て

- Business Steward Portal 例外エディターでの例外の取得
- Business Steward Portal 例外エディターでの例外の更新
- Business Steward Portal 例外エディターで [保存] と [再検証] のどちらかをクリックしたときの例外の再検証

監査ログの詳細については、Spectrum™ Technology Platformの Web UI 用管理ガイドを参照してください。

コンポーネント

Business Steward モジュールは、次の部分で構成されます。

- **Exception Monitor —** レコードを一連の条件と照合して、データ スチュワードによる手動での確認が必要なレコードかどうかを判断するステージです。レコードがこれらの条件を満たすと、このステージでは電子メールを送信して受取人に例外を通知できます。
- **Write Exceptions —** 例外レコードを例外リポジトリに書き出すステージです。例外リポジトリに書き出された例外レコードは、データ スチャワードが確認できる状態になります。
- **例外ダッシュボードと例外エディター —** 統計情報の要約とグラフをダッシュボード形式で表示するブラウザベースのツールです。これらの情報を使用して、データに発生した例外の種類を確認できます。また、例外レコードを変更し、その再処理を承認できるエディターも用意されています。
- **例外の管理 —** すべてのユーザの例外レコード アクティビティを確認および監視できるブラウザベースのツールです。
- **データ品質のパフォーマンス —** 例外レコードのトレンドに関する情報を提供するブラウザベースのツールです。主要パフォーマンス指標を特定し、所定の条件が満たされたら通知を送信できます。
- **Read Exceptions —** 例外リポジトリから例外レコードをデータフローに読む込むステージです。このステージでは、データ スチュワードが修正した例外レコードの再処理を行えます。

Exception Monitor

Exception Monitor ステージでは、レコードを一連の条件と照合して、データ スチュワードによる手動での確認が必要なレコードかどうかを判断します。Exception Monitor を使って、Spectrum™ Technology Platformが正しく処理できなかったレコードを手動による確認ツール (Business Steward Portal) に渡すことができます。

レコードの手動確認が必要かどうかを判断する条件を設定するだけでなく、それらの条件が所定の回数満たされた場合に 1 つ以上の電子メール アドレスに通知を送信するように Exception Monitor を設定することもできます。

例外処理の詳細については、**Business Steward モジュール**（240ページ）を参照してください。

入力

Exception Monitor は、任意のレコードを入力とします。入力データに "CollectionNumber" というフィールドがない場合、**[例外グループのすべてのレコードを返す]** オプションは無効になります。

> **注 :** Exception Monitor は、リストやジオメトリ オブジェクトなどの複合データを含むフィールドをモニタリングできません。

オプション

[条件] タブ

表 18 : Exception Monitor のオプション

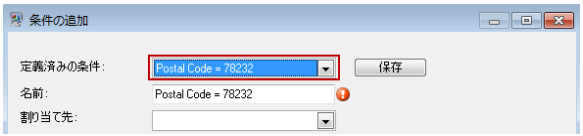
オプション名	説明
条件が満たされた時点で評価を停止する	1 つの条件が満たされた場合に、残りの条件に対してレコードの評価を続行するかどうかを指定します。このオプションを有効にすると、実行すべき評価の数が減少する可能性があるため、パフォーマンスが向上することがあります。ただし、必ずしもすべての条件を評価しない場合、Business Steward Portal に表示される例外レポートの完全性はある程度損なわれます。例えば、3 つの条件 (Address Completeness、Name Confidence、Geocode Confidence) を定義し、レコードが Address Completeness で定義された条件を満たしていて、このオプションが有効な場合、レコードは Name Confidence および Geocode Confidence に対して評価されません。Name Confidence 条件にマッチするためにこのレコードが例外と判定される場合でも、この情報は記録されません。レコードは、Address Completeness と Name Confidence 両方の問題があるとレポートされる代わりに、Address Completeness の問題のみがあるとレポートされます。

条件および例外の追加または変更

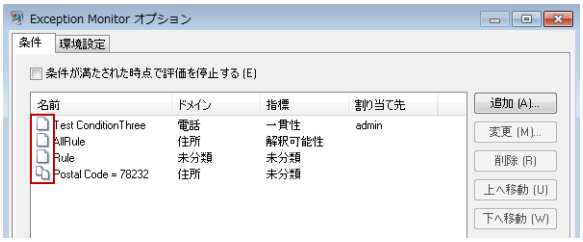
条件は、レコードが "例外" で手動による確認のためにルーティングする必要があるかどうかを判断するための基準を定義します。通常は、データフローの初期に自動処理がエラーになったり、信頼性が低いために、手動で確認する必要があるレコードを一貫して特定できる条件を定義するということです。

Exception Monitor ステージでは、[条件の追加] ダイアログ ボックスを使用して、定義済みの条件およびカスタム条件を作成できます。定義済みの条件は、すべてのデータフローで使用できますが、カスタム条件は、その条件を作成したデータフローでのみ使用できます。どちらのタイプも設定処理はほぼ同じですが、定義済みの条件を作成するには、フィールドに必要な情報を入力してから、以下の図の赤い枠で囲まれた **[保存]** をクリックして、条件を保存する必要があります。

カスタム条件を保存すると、**[定義済みの条件]** フィールドが "<カスタム条件>" から条件の名前に変わります。



定義済みの条件またはカスタム条件を作成した後、それらは、[Exception Monitor のオプション] ダイアログ ボックスの [条件] タブに表示されます。以下の図のように、条件名の横のアイコンは、その条件が定義済みの条件かカスタム条件かを表しています。2 つの文書のアイコンは定義済みの条件を表し、1 つの文書のアイコンはカスタム条件を表します。



1. **[Exception Monitor のオプション]** ウィンドウの [条件] タブで、**[追加]** をクリックして新しい条件を作成するか、**[変更]** をクリックして既存の条件を編集します。次のフィールドに必要な値を指定します。

   - **定義済みの条件** — 定義済みの条件を選択するか、ドロップダウンを "<カスタム条件>" のままにして新しい条件を作成します。
   - **名前** — 条件の名前。任意の名前を付けることができます。条件名は Business Steward Portal に表示されるので、わかりやすい名前を使用してください。例えば、"MatchScore<80" や "FailedDPV" などです。新しい条件の名前として既存の条件名の末尾に文字を追加する場合 (例えば、"FailedDPV" と "FailedDPV2")、既存の名前と一致する最後の文字 (この例では "V") を入力した時点で、既存の条件を上書きするかどうかを尋ねるメッセージが表示されます。このプロンプトに "はい" と答え、条件名を最後まで入力してから、**[OK]** または **[保存]** をクリックすると、[Exception Monitor のオプション] ダイアログ ボックスに両方の条件が表示されます。新しい条件の名前が既存の条件の名前と 100% 同じでない限り、既存の条件は上書きされません。

- **割り当て先** — この条件を満たす例外レコードを割り当てる必要があるユーザを選択します。このフィールドでユーザを選択していない場合、例外レコードは、ジョブを実行したユーザに自動的に割り当てられます。
- **データ ドメイン** — (オプション) この条件によって評価するデータの種類を指定します。データに発生した例外の種類を表示するレポート目的でのみ使用されます。例えば、条件によって住所検証の成功/失敗を評価する場合、データ ドメインは "住所" です。条件によってジオコーディング操作の成功/失敗を評価する場合、データ ドメインは "空間" になります。独自のデータ ドメインを指定するか、定義済みの以下のドメインのいずれかを選択することができます。
  - Account — セールス アカウントに関連付けられた企業または組織の名前をチェックします。
  - Address — 完全な郵送先住所や郵便番号などの住所データをチェックします。
  - Asset — 物理的資産、不動産、人材、その他のアセットなど、企業の資産に関するデータをチェックします。
  - Date — 日付データをチェックします。
  - Email — 電子メール データをチェックします。
  - Financial — 通貨、証券などに関連するデータをチェックします。
  - Name — 名や姓などの個人名データをチェックします。
  - Phone — 電話番号データをチェックします。
  - Product — 材料、部品、商品などに関するデータをチェックします。
  - Spatial — 洪水発生地帯、海岸線、家屋、販売区域など定義済みの地勢を表す点、ポリゴン、または線データをチェックします。
  - SSN — 米国社会保障番号データをチェックします。
  - Uncategorized — この条件を分類しない場合は、このオプションを選択します。
- **データ品質指標** — (オプション) この条件が評価する指標を指定します。データに発生した例外の種類を表示するレポート目的でのみ使用されます。例えば、レコードの完全性 (すべての住所に郵便番号が含まれているかなど) を評価するための条件を設計する場合、データ品質指標として "完全性" を指定することができます。独自の指標を指定するか、定義済みの以下の指標のいずれかを選択することができます。
  - Accuracy — データを信頼できるソースに対して検証できるかどうかを評価します。例えば、郵政当局のデータを使用して住所を検証することはできません。正確ではないので、例外と見なされます。
  - Completeness — データに不可欠な属性が欠落しているかどうかを評価します。例えば、郵便番号が欠落している住所や、連絡先が欠落しているアカウントなどです。
  - Consistency — データが複数のシステム間で一貫しているかどうかを評価します。例えば、顧客データ システムでは M と F という性別コードが使用されているが、処理しているデータの性別コードが 0 と 1 の場合、データには一貫性の問題があると見なされます。

- Interpretability — データが他のシステムによって解釈可能なデータ構造に正しくパースされているかどうかを評価します。例えば、社会保障番号に含めることができるのは数値データだけです。このデータに xxx-xx-xxxx などの文字が含まれる場合、データには解釈可能性の問題があると見なされます。
- Recency — データが最新かどうかを評価します。例えば、ある個人が引っ越したが、システム内の住所が古いままの場合、データには最新の問題があると見なされます。
- Uncategorized — この条件を分類しない場合は、このオプションを選択します。
- Uniqueness — 重複データがあるかどうかを評価します。データフローが重複データを統合できなかった場合、そのレコードは例外と見なされます。

2. 少なくとも 1 つの式を条件に追加する必要があります。式とは、フィールドの値をチェックする論理文です。式を追加するには、**[追加]** をクリックします。既存の式を変更するには、**[変更]** をクリックします。次のフィールドに必要な値を指定します。
   - **Expression Builder で作成した式** — このオプションは、基本式を作成する場合に選択します。
   - **カスタム式** — このオプションは、Groovy スクリプトを使用して式を記述する場合に選択します。ネストされた評価など、より複雑なロジックを使用する必要がある場合は、カスタム式を使用します。詳細については、**Exception Monitor でのカスタム式の使用**（246ページ）を参照してください。
   - この条件に対して他の式が既に定義されている場合、**[論理演算子]** フィールドで演算子を選択することができます。次のいずれかです。
     - And — 前にある式に加えてこの式も真でなければ、条件は真になりません。
     - Or — 前にある式が真でなくてもこの式が真であれば、条件は真になります。
   - Expression Builder を使用して式を作成する場合、次のフィールドを利用できます。
     - **フィールド名** — この式で評価するフィールドを選択します。Exception Monitor ステージより上流のステージに基づいて、利用可能なフィールドの一覧が表示されます。
     - **演算子** — 評価で使用する演算子を選択します。
     - **値** — [演算子] フィールドで選択した演算子を使用して式でチェックする値を指定します。
3. **[追加]** をクリックして式を追加します。式の追加が終了したら、**[閉じる]** をクリックします。
4. 式の評価順序を変更するには、**[上へ移動]** ボタンと **[下へ移動]** ボタンを使用します。
5. この条件を特定の回数満たした場合に、Exception Monitor で 1 つ以上の電子メール アドレスにメッセージを送信するには、**[通知]** タブをクリックします。送信される電子メールには Business Steward Portal の例外エディター内の失敗レコードへのリンクが含まれ、例外エディターでは正しいデータを手動で入力できます。通知を設定しない場合は、手順 11 に進みます。特定の電子メール アドレスでの通知の受信を停止するには、そのアドレスを、[条件の変更] ダイアログ ボックスで [通知] タブの **[通知送信先]** 行にある受取人一覧から削除します。

注： Enterprise Designer 内で通知を正しく使用できるようにするには、Management Console で通知を設定する必要があります。通知の設定については、*『管理ガイド』*を参照してください。

6. 通知を送信する必要がある電子メール アドレスを入力します。複数のアドレスを入力する場合は、各アドレスをカンマ、空白スペース、またはセミコロンで区切ります。
7. 通知を送信するタイミングを指定します。条件を満たしたらすぐに通知を送信することも、条件を特定の回数満たしたら送信することもできます。最大値は 1,000,000 回です。
8. 初回の電子メールの後、指定された電子メール アドレスに確認メッセージを送信する場合は、**[次の日数後にリマインダーを送信]** ボックスをチェックします。
9. 初回の電子メールの後、何日経過したら確認電子メールを送信するかを入力します。
10. 初回の確認電子メールの後に続けて毎日確認メッセージを送信する場合は、**[毎日リマインダーを送信]** をクリックします。
11. 通知用電子メールの **[件名]** を入力します。
12. この条件を再利用できるように定義済みの条件として保存するには、**[保存]** をクリックします。既存の条件を変更して [保存] をクリックすると、既存の条件を上書きするかどうかを尋ねるメッセージが表示されます。定義済みの条件を上書きすると、その条件を使用しているすべてのデータフローで変更が有効になるので注意してください。
13. 式に対する作業が完了したら、**[OK]** をクリックします。
14. 必要に応じて追加の条件を追加または変更します。
15. 条件の評価順序を変更するには、**[上へ移動]** ボタンと **[下へ移動]** ボタンを使用します。条件の評価順序が重要になるのは、**[条件が満たされた時点で評価を停止する]** オプションを有効にした場合のみです。詳細については、**[構成] タブ**（248ページ）を参照してください。
16. 作業が完了したら、**[OK]** をクリックします。

条件または式の削除

- 条件を削除するには、Exception Monitor を開き、削除する条件を選択して、**[削除]** をクリックします。条件を削除すると、条件内の式もすべて削除されることに注意してください。
- 式を削除するには、その式を含む条件を開き、式を選択して、**[削除]** をクリックします。

Exception Monitor でのカスタム式の使用

Groovy スクリプト言語を使用して式を作成することにより、Exception Monitor によるレコードのルーティング方法を制御する独自のカスタム式を記述することができます。

Groovy スクリプトの使用

Groovy の詳細については、**groovy-lang.org** を参照してください。

Exception Monitor ステージで使用される Groovy 式は、レコードを例外とみなし、手動による確認にルーティングするべきかどうかを示す Boolean 値 (真または偽) になる必要があります。例外レコードは例外ポートにルーティングされます。

例えば、バリデーションの確信レベルが 85 未満のレコードを確認する必要がある場合、スクリプトは次のようになります。

```
data['Confidence']<85
```

Exception Monitor は、Confidence フィールドの値を条件と照合して、レコードの送信先出力ポートを決定します。

フィールドに単一の値があるかどうかを確認する

次の式は、Status フィールドの値が 'F' の場合に真として評価されます。完全一致が要求されるため、フィールドの値が 'f' の場合は真として評価されません。

```
return data['Status'] == 'F';
```

フィールドに複数の値があるかどうかを確認する

次の式は、Status フィールドの値が 'F' または 'f' の場合に真として評価されます。

```
boolean returnValue = false;
if (data['Status'] == 'F' || data['Status'] == 'f')
{
 returnValue = true;
}
return returnValue;
```

フィールドの長さを評価する

次の式は、PostalCode フィールドの文字数が 6 文字以上の場合に真として評価されます。

```
return data['PostalCode'].length() > 5;
```

フィールド値に含まれる文字を確認する

次の式は、PostalCode フィールドにダッシュが含まれている場合に真として評価されます。

```
boolean returnValue = false;
if (data['PostalCode'].indexOf('-') != -1)
```

```
{
 returnValue = true;
}
return returnValue;
```

一般的な誤り

次に、スクリプトを使用するときの一般的な誤りを示します。

次の式は誤りです。PostalCode (列名) は、単一引用符または二重引用符で囲む必要があります。

```
return data[PostalCode];
```

次の式は誤りです。列が指定されていません。

```
return data[];
```

次の式は誤りです。row.set() は Boolean 値を返しません。row.set() は常に偽として評価されます。また、row.set() は、PostalCode フィールドを 88989 に変更します。

```
return row.set('PostalCode', '88989');
```

フィールドの値の設定には単一引用符を使用し、フィールドの値の確認には二重引用符を使用します。

[構成] タブ

表 19 : Exception Monitor のオプション

オプション名	説明
Exception Monitor を無効にする	Exception Monitor をオンまたはオフにします。Exception Monitor を無効にした場合、レコードはステージを通過するだけで、アクションは発生しません。これは、実際にはデータフローから Exception Monitor を削除することと似ています。
例外上限に達したらジョブを停止する	指定した数のレコードが例外条件を満たした場合にジョブの実行を停止するかどうかを指定します。
例外レコードの最大数	**[例外上限に達したらジョブを停止する]** が選択されている場合、このフィールドを使用して、ジョブの実行を停止するまでに許可する例外レコードの最大数を指定します。例えば、100 を指定した場合、101 番目の例外レコードが発生するとジョブは停止します。

オプション名	説明
報告のみで例外を生成しない	例外条件を満たすレコードを追跡し、その統計情報を Business Steward Portal の [データ品質のパフォーマンス]ページにレポートしますが、レコードの例外は作成しません。
例外グループのすべてのレコードを返す	例外レコードだけでなく、例外レコードのグループに属するすべてのレコードを返すかどうかを指定します。例えば、あるマッチ グループ (MatchKey に基づく) に 4 つのレコードがあるとします。1 つはサスペクト レコード、1 つはスコア 90 の重複レコード、残り 2 つはスコア 80 と 83 のユニーク レコードです。MatchScore が 80 ～ 89 のレコードを例外とする条件がある場合、デフォルトでは、マッチ スコア 80 と 83 のレコードだけが例外ポートに送信されます。しかし、このオプションが有効な場合は、4 つのレコードすべてが例外ポートに送信されます。 データ スチュワードが例外レコードをグループ内の他のレコードと比較できるようにする場合は、このオプションを有効にしてください。グループ内のすべてのレコードを比較することにより、データ スチュワードは、例外レコードの処理方法に関して、より詳細な情報に基づく決定を行うことができます。たとえば、マッチングを行う際に、データ スチュワードはすべての候補を確認して、例外レコードが他の重複レコードかどうかを判断することができます。 **注 :** 入力データに "CollectionNumber" というフィールドがない場合、このオプションは無効になります。
グループ化方法	**[例外グループのすべてのレコードを返す]** を選択した場合は、レコードをグループ化するフィールドを選択します。 **注 :** 入力フィールド "CollectionNumber" は、「グループ化方法」機能に対して有効な選択肢ではないため、このリストには表示されません。
サービスを再検証	このデータフローからレコードを再検証するときに実行するサービスを選択します。
再検証後のアクション	再検証されたレコードを再処理するか、承認するかを指定します。

オプション名	説明
マッチング フィールドで例外レコードをマッチング	マッチングフィールドを使用して、入力レコードをリポジトリ内の例外レコードとマッチングします。以前に例外を生成したが、現在は入力内で訂正されているレコードが入力に含まれている場合、このオプションを有効にします。 入力レコードは条件と照合され、次にリポジトリ内の既存の例外レコードとマッチングされます。入力レコードが条件を満たし、例外レコードとマッチする場合、その例外レコードはリポジトリから削除されます。入力レコードが条件を満たさず、例外レコードとマッチする場合、その例外レコードは更新され、リポジトリ内に維持されます。また、リポジトリ内に重複がある場合、更新されるのは、データフローごとに 1 つのマッチした例外のみです。そのデータフローのそれ以外の例外は削除されます。
マッチング フィールド	リポジトリ内の例外フィールドとマッチングするキーの作成に使用するすべての入力フィールドのリストを示します。**[マッチング フィールドで例外レコードをマッチング]** チェック ボックスをオンした場合は、少なくとも 1 つのマッチング フィールドを定義する必要があります。

出力

Exception Monitor は、2 つのポートにレコードを返します。1 つのポートには、Exception Monitor ステージで定義したいずれの条件も満たさないレコードが含まれます。もう一方のポートである例外ポートには、1 つ以上の例外条件にマッチするすべてのレコードが含まれます。**[例外グループのすべてのレコードを返す]** オプションが有効な場合、例外ポートに非例外レコードも含まれることがあります。Exception Monitor がレコード内にフィールドを追加したり、フィールドを変更したりすることはありません。

Read Exceptions

Read Exceptions は、入力として例外リポジトリのレコードをデータフローに読み込むステージです(例外リポジトリの詳細については、**Business Steward モジュール**（240ページ）を参照してください)。

> **注：** Read Exceptions によってデータフローに読み込まれたレコードは、リポジトリから削除されます。

入力

Read Exceptions により、例外リポジトリからデータが読み込まれます。データフロー内の別のステージから入力を取得することはありません。

> **注：** Business Steward Portal で "承認済み" とマークされたレコードのみがデータフローに読み込まれます。

オプション

Read Exceptions ステージには次のオプションがあります。

[全般] タブ

[一般] タブのオプションで、データフローに読み込む例外レコードを指定します。

[Filter] オプションを使用すると、次の条件を使用して、例外リポジトリからレコードのサブセットを選択することができます。

- **ユーザ:** データフローに読み込む例外を生成したデータフローを実行したユーザ。
- **データフロー名:** データフローに読み込む例外を生成したデータフローの名前。
- **ステージ ラベル:** Enterprise Designer でデータフローに表示される、Exception Monitor ステージのラベル。この条件は、例外を生成したデータフローに複数の Exception Monitor ステージが含まれ、それらの Exception Monitor ステージのいずれかからの例外のみを読み込みたい場合に役立ちます。
- **開始日:** データフローに読み込む最古のレコードの日時。例外レコードの日付は、例外レコードが最後に変更された日付です。
- **終了日:** データフローに読み込む最新のレコードの日時。例外レコードの日付は、例外レコードが最後に変更された日付です。

[フィールド] リストには、データフローに読み込まれるフィールドが表示されます。デフォルトではすべてのフィールドが含まれますが、**[含める]** 列のチェック ボックスをクリアすることにより、そのフィールドを除外できます。

[プレビュー] リストには、**[Filter]** で指定した条件を満たすレコードが表示されます。

> **注：** プレビューには、Business Steward Portal で "承認済み" とマークされ、フィルタ条件を満たすレコードのみが表示されます。

[ソート] タブ

フィールド値に基づいて入力レコードをソートするには、[ソート] タブを使用します。

- **追加**: ソートの基となるフィールドを追加します。
- **[フィールド名]** 列: ソートの基となるフィールドの名前が表示されます。ドロップダウン ボタンをクリックして、フィールドを選択することができます。
- **[順序]** 列: 昇順でソートするか降順でソートするかを指定します。
- **[上へ]** および **[下へ]**: ソートの順序を変更します。レコードは、リストの先頭のフィールドによってまずソートされ、次にリストの 2 番目のフィールドによってソートされます。

- **[削除]**: ソート フィールドを削除します。

[実行時] タブ

- **[開始レコード]:** データフローに読み込む最初のレコードのリポジトリ内の位置を指定します。例えば、リポジトリ内の最初の 99 レコードをスキップする場合は、100 と指定します。**[一般]** タブで指定した条件にマッチする場合、100 番目のレコードがリポジトリに読み込まれる最初のレコードになります。レコードの位置は、Business Steward Portal 内のレコードの順序によって決定されます。
- **[すべてのレコード]: [一般]** タブで指定した検索条件にマッチするすべてのレコードを読み込む場合は、このオプションを選択します。
- **[最大レコード数]:** データフローに読み込むレコード数を制限する場合は、このオプションを選択します。例えば、選択条件にマッチする最初の 1,000 レコードのみを読み込みたい場合は、このオプションを選択して、1000 と指定します。

出力

Read Exceptions ステージは、承認済みかつRead Exceptions オプションで指定した選択条件にマッチする例外リポジトリ内のレコードを返します。Read Exceptions は、レコードのフィールドに加え、Business Steward Portal でそのレコードに対して最後に行われた変更を説明するフィールドを返します。

表 20 : Read Exceptions の出力

フィールド名	説明
Exception.Comment	例外の解決者が入力したコメント。例えば、コメントによって、Business Steward Portal でそのレコードに対して行った変更を説明することができます。
Exception.LastModifiedBy	Business Steward Portal で最後にレコードを変更したユーザ。
Exception.LastModifiedMilliseconds	Business Steward Portal で最後にレコードが変更された時刻。時刻は 1970/01/01 0:00 GMT からのミリ秒で表されます。これは、Java プログラミング言語で時間を計算する標準的な方法です。この値を使用してデータの比較を実行したり、トランスフォームを作成してこの値を必要な任意の日付形式に変換することができます。

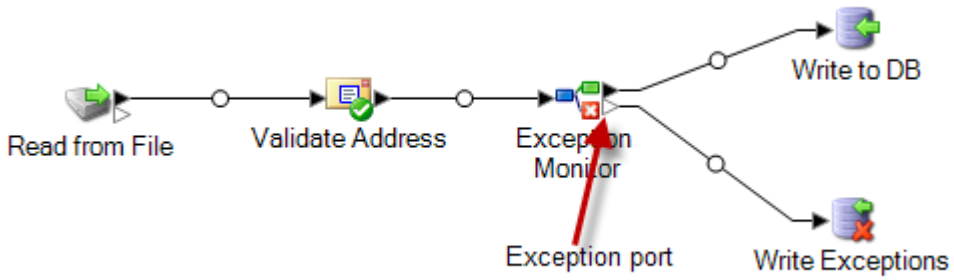
フィールド名	説明
Exception.LastModifiedString	Business Steward Portal で最後にレコードが変更された時刻。このフィールドは、Exception.LastModifiedMilliseconds フィールドよりも理解しやすい表現で日付を表します。時刻は次の形式で表されます。 `Thu Feb 17 13:34:32 CST 2011`

Write Exceptions

Write Exceptions は、Exception Monitor ステージが例外として特定したレコードを取得して、例外リポジトリに書き出すステージです。例外リポジトリに書き出されたレコードは、Business Steward Portal を使用して確認および編集できます。

入力

Write Exceptions ステージは、Exception Monitor ステージの例外ポートからレコードを取得して、例外リポジトリに書き出します。Write Exceptions ステージは、Exception Monitor ステージの例外ポートの下流に配置する必要があります。例外ポートは Exception Monitor ステージの 1 番下の出力ポートです。



オプション

Write Exceptions ステージでは、例外リポジトリにどのフィールドのデータを返すかを設定できます。表示されるフィールドは、データフローの上流にあるステージによって異なります。例えば、データフローに Validate Address ステージがある場合は、AddressLine1、AddressLine2、City、PostalCode 等のフィールドが Write Exceptions ステージに表示されます。デフォルトでは、これらのフィールドがすべて選択されます。例外リポジトリに返さないフィールドがある場合は、

そのフィールドのボックスをオフにします。フィールドの順序は、Write Exceptions ステージに入力された順序によって決まります。行を選択し、画面右側の矢印で上下に移動させることによって、フィールドを並べ替えることができます。ここで選択した順序は、Business Steward Portal のすべてのユーザに適用されますが、各ユーザは Portal 内で、自分の好みに合わせてフィールドを並べ替えることができます。

> **注：** 例外の識別には、Write Exceptions ステージ名ではなく、データフロー名と Exception Monitor ステージ名が使用されます。したがって、例外レコードが存在する状態で既存のデータフローを更新する場合は (フィールド順序の変更を除きます)、それらのレコードを処理するか、データフローと Exception Monitor ステージに新しい名前を付ける必要があります。同じ方法で両方の種類の例外を後処理することはできないためです。

データのマッチングが通常の規則で行えない場合は、**[Portal での Best Of Breed レコードの作成を許可する]** を選択します。この機能は、グループ内で選択されているレコードをコピーし、それを重複レコードの代わりに処理に使用します。このオプションは、マッチング ジョブから生成されたグループ化済み例外レコードに対してのみ使用可能です。このレコードは、CandidateGroup フィールドまたは CollectionNumber フィールドがあるかどうかで識別されます。このオプションを使用すると、“CollectionRecordType” という読み取り専用のフィールドが例外レコードに追加されます。このフィールドは、リストの最後に表示されます。このフィールドについてはすべてのオプションが無効になることに注意してください。

> **注：** データフローが**再検証**も使うように設定されている場合は、Exception Monitor ステージ/サブフローとサービス自体に CollectionRecordType フィールドを手動で追加してエクスポーズする必要があります。

Best of Breed レコードを Business Steward Portal 例外エディターの [重複を解決] ビューに追加した後で、このフィールドは "BestOfBreed" に設定されます。BestOfBreed レコードの作成を選択した場合、Business Steward Portal でそれらのレコードに **[すべて承認]** オプションを使うことはできません。Business Steward Portal における BestOfBreed レコードの詳細については、**この説明**を参照してください。

入力フィールドをリポジトリには格納したいが、Business Steward Portal には表示したくないという場合があり得ます。例えば、フィールドに機密データが含まれる場合や、単に Portal の表示を簡潔にしたい場合が考えられます。**[表示を許可]** ボックスをオンにして、選択されているフィールドのうち、例外リポジトリに渡された後で表示するフィールドを指定することもできます。デフォルトでは、すべてのフィールドが表示されます。Portal に表示したくないフィールドがあれば、そのチェックボックスをオフにします。

また、選択しているフィールドのうち、例外リポジトリに渡された後に Portal で編集可能にするフィールドを指定できます。デフォルトでは、Write Exceptions に読み込まれるすべてのフィールドで **[編集を許可]** 列がチェックされています。読み取り専用状態で例外リポジトリに返すフィールドがある場合は、そのフィールドのボックスをオフにします。

最後に、**[検索]** 機能を使って、問題になっているデータが含まれるフィールドに検索を割り当てることができます。Business Steward 構成ツールで定義された検索のリストから選択するか、検索の名前を手動で入力することができます。検索の詳細については、**検索**（279ページ）を参照してください。

注： 検索は、文字列タイプのフィールドにのみ割り当てることができます。

出力

Write Exceptions は、データフローに出力を返しません。例外レコードを例外リポジトリに書き出します。

Business Steward Portal

Business Steward Portal の概要

Business Steward Portal とは

Business Steward Portal は、自動処理に失敗したり、十分な信頼性で処理されなかったレコードを確認、変更、承認するためのツールです。Business Steward Portal を使用して、正しいデータや追加データをレコードに手動で入力します。例えば、顧客レコードが住所検証処理に失敗した場合に、検索ツールを使用して調査を行うことによって顧客の住所を特定し、正しい住所を含むようにレコードを変更することができます。変更したレコードは、設定に応じて、Spectrum™ Technology Platform で承認および再処理したり、別のデータ検証または強化処理に送ったり、データベースに書き出したりすることができます。Portal を使用して、元のレコードにはなかった情報を追加することも可能です。

さらに、Business Steward Portal は、データ ドメイン (名前、住所、空間など) や、データがエラーになったデータ品質指標 (完全性、精度、一貫性など) といった、例外処理をトリガするデータの種類を理解するためのサマリ グラフも提供します。

また、Business Steward Portal の [例外の管理] ウィンドウでは、あるユーザから別のユーザへのレコードの再割り当てなど、例外レコードに関するアクティビティを表示および管理できます。最後に、Business Steward Portal の [データ品質] 画面には、データフローおよびステージ全体のトレンドに関する情報が表示されます。

例外処理の詳細については、**Business Steward モジュール**（240ページ）を参照してください。

Business Steward Portal へのアクセス

Business Steward Portal を開くには、**[スタート] > [すべてのプログラム] > [Pitney Bowes] > [Spectrum Technology Platform] > [サーバー] > [Welcome ページ]** に移動します。**[Spectrum**

データ品質] を選択し、**[Business Steward Portal]** を選択して、**[Business Steward Portal を開く]** をクリックします。

また、次の手順で開くこともできます。

1. Web ブラウザを開いて、http://<サーバー名>:8080/bsmportal に移動します。

   例を次に示します。

   http://myserver:8080/bsmportal

   サーバー名とポートがわからない場合は、Spectrum™ Technology Platform 管理者に問い合わせてください。

2. Spectrum™ Technology Platform にログインします。ログインできない場合は、Spectrum™ Technology Platform 管理者に問い合わせてください。

Business Steward Portal メニュー

Business Steward Portal メニューは、次に示すように、4 つのオプションで構成されます。このメニューからヘルプ システムにアクセスすることもできます。

- **[ダッシュボード]** — 自分に割り当てられた例外のステータスが表示されます。表示権限があれば、他のユーザに割り当てられた例外のステータスも表示されます。
- **[エディター]** — 例外レコードを再処理のために確認、編集、承認します。
- **[管理]** — 表示権限があれば、自分自身または他のユーザに例外を割り当てることができます。削除権限があれば、例外レコードをリポジトリから完全に削除することができます。
- **[データ品質]** — 表示権限があれば、このページにアクセスして統計情報を表示し、例外レコードの主要パフォーマンス指標を設定できます。表示権限がない場合は、このページにアクセスできません。
- [ユーザ] ドロップダウン — Business Steward Portal のヘルプ システム、Spectrum Technology Platform のすべてのドキュメント、[プロファイル] ページにアクセスします。このページでは、優先言語を設定し、優先カルチャーを示す国を指定できるほか、通知用メール アドレスの設定や、パスワードの変更を行うことができます。

[ダッシュボード] ページ

例外ダッシュボードには、自分や他のユーザが保有する例外レコードのステータスの概要データが表示されます(ただし、変更権限がなければ他人のデータを表示することはできません)。これには、以下が含まれます。

- 例外の総数
- 承認済みの例外と未承認の例外の数や割合
- 選択したユーザの日、週、月ごとの進捗
- 例外レコード承認のデータフロー別およびデータフロー内ステージ別の進捗

注： Management Console の Business Steward の設定ページで進捗の追跡をオフにすると、ユーザおよびデータフローの進捗グラフが表示されなくなります。

1. [ダッシュボード] タブで、以下で強調表示された最初のドロップダウン ボックスから例外アクティビティを表示するユーザを選択します。リストには現在例外が割り当てられているユーザのみが表示されることに注意してください。

2. 日、週、月の期間を選択するには、**[設定]** の歯車アイコン (以下で強調表示された 2 つ目のドロップダウン ボックス) をクリックします。
3. データフローのリスト、または複数のデータフローのサブセットを絞り込むには、**[フィルタ]** フィールドに検索条件を入力します。検索語は、データフロー名に含まれている必要があります。

[エディター] ページ

例外エディターでは、例外レコードを手動で確認、変更、承認できます。手動確認の目標は、特に Spectrum™ Technology Platform が自動データフロー処理の一環として訂正できなかった場合に、不正なデータや欠落しているデータを特定し、それを更新して承認することです。そうすれば例外レコードを再検証して、承認または再処理することができます。

また例外エディターは、重複する例外レコードを解決したり、レコードの編集、承認、再実行に役立つ情報を検索ツールで調べたりするためにも使用できます。

例外エディターの内容のカスタマイズ

複数の方法で、例外エディターの表示内容をカスタマイズできます。**[選択オプション]** を使用して、特定のユーザ、データフロー、ジョブ ID などに対するレコードを返すことができます。**[Field Filter]** ツールを使用して、特定の条件を満たすレコードを表示し、条件を満たさないレコードを非表示にすることができます。

[ページあたりの項目数] ツールを使用している状態で、選択オプションや Field Filter を適用して表示される例外レコードのリストを絞った場合は、結果は最初の画面に表示されているものだけでなく、複数のページにわたる可能性があることを忘れないでください。例えば、ページあたりの項目数を 10 に設定し、Field Filter を適用して特定の郵便番号のレコードだけを返すようにしたとします。最初の画面では 10 件しかレコードが返されていないように見えるかもしれませんが、それは 1 度に表示するレコード数を 10 件までと設定したためで、実際には結果は複数のページにわたる可能性があります。

選択オプションの使用

例外グリッドに表示されるレコードを絞り込むには、選択オプション ウィンドウのドロップダウン フィールドで条件を選択します。

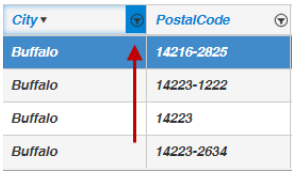
ユーザ	必須データフローに割り当てられたユーザの ID。この情報は、変更権限がある場合のみ表示可能です。
データフロー名	必須例外レコードを生成したデータフローの名前。
ステージ ラベル	必須データフロー内の Exception Monitor ステージに付与される、ユーザが定義した名前。データフローに複数の Exception Monitor ステージが含まれる場合、この情報は特に役立ちます。データフローの作成者が各 Exception Monitor ステージに意味のあるラベルを付与した場合、例外レコードを生成した Exception Monitor を特定することができます。デフォルトのラベルは "Exception Monitor" です。
ジョブ ID	必須システムがジョブに割り当てる数字の識別子。ジョブを実行するたびに、新しいジョブ ID が割り当てられます。
ステータス	これはオプションです。レコードの承認ステータス。
日付	これはオプションです。データフローが実行された日付 (およびオプションとして時刻)。時刻を入力するには、日付に続けて時刻を入力します。
データ ドメイン	これはオプションです。例外となったデータの種類。データ ドメインの例には、名前、住所、電話番号などがあります。この情報は、編集する必要があるレコード内のフィールドを特定するのに役立ちます。
品質指標	これはオプションです。レコードがエラーになった品質指標。品質指標の例には、精度、完全性、一意性などがあります。この情報は、レコードが例外と識別された理由を判断するのに役立ちます。

Field Filter の使用

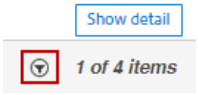
選択オプションを作成し、例外レコードが読み込まれたら、フィールド フィルタリングを使用して関心のあるレコードのみを表示することができます。デフォルトでは、Business Steward Portal には一度に 1 つのSpectrum™ Technology Platformデータフローのレコードのみが表示されます。特定のフィールド内で一定の条件を満たすレコードのみを表示するように、レコード リストをさらにフィルタリングすることができます。作成されたフィルタは自動的に保存され、データフローを次に例外エディターで開く際には、そのフィルタが適用されます。

注： Field Filter ツールを使用する前に、選択オプションを適用する必要があります。

フィルタは複数のフィールドに対して作成可能ですが、各フィールドに対して作成できるフィルタは 1 つだけです。フィールドに既にフィルタが適用されている場合、矢印の背景色は青色になります。



フィルタは、テーブル形式で表示している場合のみ作成可能です。ただし、テーブル形式表示で定義されたフィルタは、フォーム形式表示にも反映されます。テーブル形式表示と同様に、フォームの最下部近くにあるインジケータによって、レコードにフィルタが適用されていることが示されます。



レコードの一覧をフィルタリングするには、次の手順に従います。

1. **[フィルタ]**ボタンをクリックします。各列の見出しの横にフィルタ アイコンが表示されます。
2. データにフィルタを適用するフィールドのフィルタ アイコンをクリックします。
3. フィールドのデータ タイプに適した演算子を選択し、その後に値を指定します。

が次の値と同じ	指定した値と完全に同じ値を持つレコードを探します。数値またはテキスト値を指定できます。例えば、MatchScore の値が 82、または LastName の値が "Smith" のレコードを検索できます。
が次に等しくない	指定した値以外の値を持つレコードを探します。数値またはテキスト値を指定できます。例えば、MatchScore の値が 100 以外、または LastName が "Smith" 以外のレコードを検索できます。
が次の文字から始まる	選択したフィールドが特定の値で始まるレコードを探します。例えば、LastName フィールドで "Van" をフィルタリングする場合、"Van

	Buren"、Vandenburg"、または "Van Dyck" という値を持つレコードが表示されます。
が次の文字を含む	選択したフィールド内のいずれかの位置に指定した値を含むレコードを探します。例えば、AddressLine1 フィールドで "South" をフィルタリングする場合、"12 South Ave."、"9889 Southport St."、"600 South Shore Dr."、および "4089 5th St. South" という値を持つレコードが表示されます。
が次を含まない	選択したフィールド内のどの位置にも指定した値が含まれないレコードを探します。例えば、AddressLine1 フィールドで "South" をフィルタリングする場合、"12 South Ave."、"9889 Southport St."、"600 South Shore Dr."、および "4089 5th St. South" という値を持つレコードは表示されません。
が次の値で終わる	選択したフィールドが特定の値で終わるレコードを探します。例えば、City フィールドが "burg" で終わるレコードをフィルタリングする場合、"Gettysburg"、"Fredricksburg"、および "Blacksburg" という値を持つレコードが表示されます。
が次の値より大きい	指定した値より大きい数値を持つレコードを探します。
が次の値以上	指定した値以上の数値を持つレコードを探します。例えば、50 を指定した場合、選択したフィールドの値が 50 以上のレコードが表示されます。
が次の値より小さい	指定した値より小さい数値を持つレコードを探します。
が次の値以下	指定した値以下の数値を持つレコードを探します。例えば、50 を指定した場合、選択したフィールドの値が 50 以下のレコードが表示されます。
次の値より後	指定した値よりも後の日付または時間を持つレコードを探します。
次の値と同じかそれより後	指定した値と同じかそれよりも後の日付または時間を持つレコードを探します。
次の値より前	指定した値よりも前の日付または時間を持つレコードを探します。
次の値と同じかそれより前	指定した値と同じかそれよりも前の日付または時間を持つレコードを探します。

4. **[フィルタ]** ボタンをクリックして条件を適用します。そのフィールドに対する条件にデータが一致するレコードのみが表示されます。
5. フィルタアイコンをもう一度クリックしてから、**[クリア]**をクリックすると、フィルタが削除されます。**[フィルタ]** ボタンをクリックすると、すべてのフィルタが削除されます。この操作は、テーブル形式とフォーム形式のどちらで表示している場合でも実行できます。

レコードの表示

例外レコードは、2 通りの形式で表示できます。デフォルトのビューは、テーブル形式です。1 ページあたり最大 100 件の例外レコードが表示可能です。リストをスクロールして、任意の順序でレコードを編集できます。複数のレコードを編集し、すべての編集内容を一度に保存できます。各レコードに対する編集内容を個別に保存する必要はありません。このビューを使用する場合は、1 ページに表示するレコード数を、画面最下部のドロップダウンで指定できます。

例外レコードを表示するためのもう 1 つのビューは、フォーム形式です。このビューでは、複数のレコードを同時に編集することはできません。また、各レコードに対する編集内容は、個別に保存する必要があります。複数のレコードに対する編集内容を一度に保存することはできません。

レコードの詳細の表示

どちらの形式で表示しているかに関わらず、例外グリッドには、レコードのすべてのフィールドと、その承認ステータス、例外タイプ、そしてレコードに追加されているコメントがあればそれが表示されます。テーブル形式表示では、レコードの左端にある矢印をクリックすることによって、レコードに関するその他の詳細情報が表示できます。フォーム形式表示では、**[詳細を表示]** をクリックします。これらの操作によって、以下の情報を表示する **[詳細]** タブが開きます。

ジョブ ID
システムがジョブに割り当てる数字の識別子。ジョブを実行するたびに、新しいジョブ ID が割り当てられます。

グループ化方法
例外レコード グループ内のすべてのレコードを返すようデータフローが設定されていた場合、レコードをグループ化しているフィールドが表示されます。このオプションは、重複レコードを識別するデータフローや、レコードを世帯にグループ化するデータフローなど、マッチングを実行するデータフローにのみ適用されます。

例外の時刻
Exception Monitor がレコードを例外として識別した日時。

レコード タイプ
選択したレコードのタイプ。次のいずれかです。

- **E** — 例外
- **GE** — 例外のグループ
- **N** — Best of Breed

ステータス
選択したレコードのステータス。次のいずれかです。

- 新規
- 解決済み (レコードが作成後に少なくとも 1 度編集されている場合のみ、あり得ます)

[条件] タブをクリックすると、以下の情報が表示されます。

条件
レコードを例外として識別した条件の名前。条件名は、データフローの設定者によって定義されます。

ドメイン	例外となったデータの種類。データ ドメインの例には、名前、住所、電話番号などがあります。この情報は、編集する必要があるレコード内のフィールドを特定するのに役立ちます。
指標	レコードがエラーになった品質指標。品質指標の例には、精度、完全性、一意性などがあります。この情報は、レコードが例外と識別された理由を判断するのに役立ちます。

レコードが変更される度に、レコードを変更したユーザ、変更日時、レコードが割り当てられたユーザの名前、レコードのコメント フィールドに入力されたデータ (存在する場合) といった特定の情報がシステムに保持されます。例外グリッドに表示されるレコードには、最新の変更と最新コメント (存在する場合) が反映されます。一方、**[履歴]** タブには、レコードが例外として最初にリポジトリに追加されてから、レコードを表示しているその時点までの、レコードの存続期間を通して、以下の情報が表示されます。

バージョン	変更のリビジョン番号。
最終変更者	変更を行ったユーザ。
割り当て先	改訂時点でこの例外レコードが割り当てられているユーザ。
日時	変更が保存された日付および時刻。
コメント	変更を行ったユーザが入力したコメント (任意)。

フィールドの並べ替え

テーブル形式で表示している場合は、列ヘッダの任意の箇所をクリックすることによって、特定のフィールドに基づいてレコードの表示順序を変更することができます。例えば、州のアルファベット順でレコードを表示する場合は、[州]列ヘッダをクリックします。最初のクリックで昇順、2回目のクリックで降順に並べ替えられます。3回目のクリックでは並べ替え順序がクリアされ、レコードは並べ替える前の順序に戻ります。

フィールドの設定

[ビューの設定] ボタン ([ユーザ] ドロップダウンの下の画面右側にある歯車ボタン) をクリックして変更を加えることにより、表示するフィールドを選択して、表示順序を変更することができます。その変更はユーザ名とデータフロー名に基づいてサーバーに保存されるため、後でデータフローを開いたときも同じ設定が適用されます。同様に、ここで行う変更は、フォーム形式表示で例外レコードを編集するときに表示される項目にも影響を与えます。[ビューの設定] の以下の機能を使用して、例外エディターに表示されるフィールドをカスタマイズします。

[ビューの設定] におけるフィールドの検索

[検索] ボックスにフィールド名の全体または一部を入力すると、使用可能なフィールドのリストが動的に更新されます。検索では、大文字と小文字は区別されません。

フィールドの非表示

例外レコードに表示したくないフィールドがあれば、**[ビューの設定]** をクリックし、非表示にするフィールドの選択を解除します。表示されるリストでは、例外グリッドでの表示と同じ順序で項目が並びます。

フィールドの順序の変更

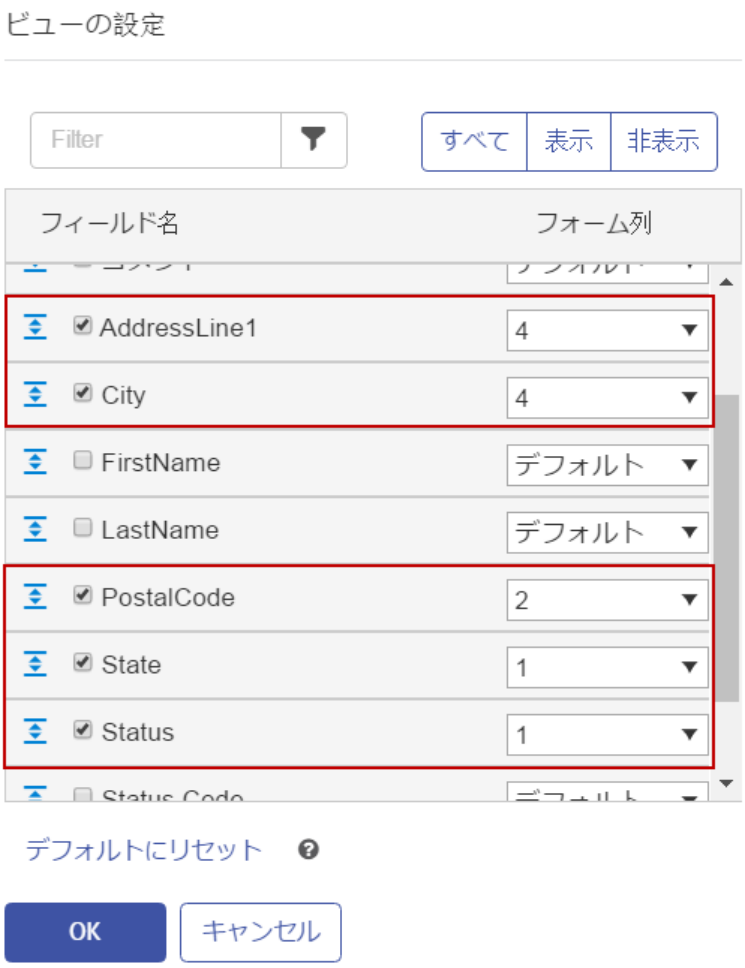
ドラッグアンドドロップ操作によってフィールドを任意の順序に並べ替えることにより、フィールドの表示順序を変更できます。ただし、検索結果が表示されている場合は、フィールドを並べ替えることはできません。フィールドを再び並べ替えられるようにするには、検索ウィンドウをクリアして、[すべて] を選択する必要があります。

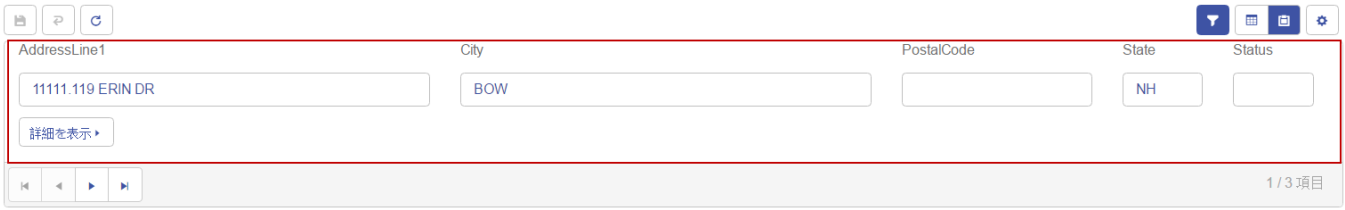
> **注：** テーブル形式で表示している場合は、例外エディターで直接、列見出しをドラッグ アンド ドロップすることによって、フィールドの表示順序を変更できます。[ビューの設定] からこれを行う必要はありません。

フォーム形式レイアウトの設定

フォーム形式表示では、表示する列を選択するだけでなく、列幅を指定することにより、フィールドのデフォルトのレイアウトを上書きできます。列幅の指定に関する唯一のルールとして、表示するすべてのフィールドの列幅の合計を 12 にする必要があります。例えば、1 行に 4 つのフィールドを並べる場合は、各フィールドの列幅を "3" に指定することができます。同様に、フィールドを 2 つだけ含める場合は、各フィールドの列幅を "6" と入力することができます。また、1 行に住所フィールドを 5 つ含める場合は、合計が 12 になるように、各フィールドに対して列の数を指定する必要があります。以下の最初の画像では、5 つのフィールドの設定の例を示しています。2 つ目の画像では、その設定がどのようにレンダリングされるかの例を示しています。

> **注：** 任意のフィールドやフィールドの組み合わせに対してデフォルトのレイアウトを上書きすると、上書きされたフィールドはウィンドウサイズに応じて調整されなくなりますが、デフォルト値のフィールドは、引き続き、ブラウザのサイズに基づき調整されます。





例外レコードの編集

例外レコードを編集する目的は、レコードを訂正または補強して承認し、正しく処理できるようにすることです。例外レコードの編集では、他の Spectrum Technology Platform サービスを使用したり、マップやインターネット、会社の他の情報システムなどの外部リソースに問い合わせたりすることが必要となる場合があります。

レコードを確認した後に、例外エディターで直接それらを編集および承認できます。テーブル形式で表示している場合は複数のレコードを一度に編集できますが、フォーム形式で表示している場合は一度に 1 つのレコードしか編集できません。

読み取り専用フィールドは編集できません。読み取り専用フィールドを編集可能にする場合は、そのデータフローとジョブ ID に関連付けられたすべての例外レコードを削除し、Write Exceptions ステージでフィールドを適切に構成した後でデータフローを再度実行する必要があります。この操作を行うと、編集可能なフィールドを持つ新しい例外レコードが作成されます。また、データタイプが一致しない値を含むようにフィールドを編集することはできません。例えば、数値データ タイプのフィールドに、文字を含めることはできません。

テーブル形式表示

テーブル形式で表示された単一のレコードのフィールドを編集するには

1. 編集するフィールドをクリックして、そのフィールドの値を変更します。読み取り専用フィールドはグレー表示されます。フィールドを右クリックすると、切り取り、コピー、および貼り付けオプションにアクセスできます。フィールドを編集すると、そのフィールドの左上隅に緑色の三角形が表示されます。この視覚的ヒントにより、フィールドの値が変更されているがまだ保存されていないことが一目でわかります。
2. 変更したレコードの **[承認済み]** ボックスをオンにします。これにより、Spectrum™ Technology Platform で処理可能としてレコードがマークされます。
3. 変更を取り消す必要が生じた場合は、取り消しを行うレコードを選択して、**[変更の取り消し]** ボタンをクリックします。
4. レコードの編集を終えたら、**[保存]** ボタンをクリックします。

テーブル形式で表示された複数のレコードのフィールドを編集するには

1. Ctrl キーまたは Shift キーを押しながら、選択されたすべてのレコードにおいて変更するフィールドをクリックします。例えば、"L.A." というインスタンスをすべて "Los Angeles" に変更する場合は、Ctrl キーを押して、あるレコードの [City] フィールドの中をクリックします。続いて、Ctrl キーまたは Shift キーを押したまま、変更する他のレコードの同じフィールドの中をクリックします。
2. フィールドの値を必要に応じて変更します。これらのフィールドを編集できますが、ここで行う変更は、これらのフィールドの以前の値がそれぞれ異なっている場合も含め、選択したすべてのレコードに適用されるので注意してください。同様に、複数のレコードを編集しているときに、フィールドのデータをクリアすると、そのデータは選択しているすべてのレコードでクリアされます。
3. 変更したレコードの **[承認済み]** ボックスをオンにします。これにより、Spectrum™ Technology Platform で処理可能としてレコードがマークされます。または、**[すべて承認]** ボタンをクリックすると、エディターに表示されている単純な例外レコードがすべて承認されます。([すべて承認] 機能は、マッチングしている例外レコードには適用されません。)
4. 変更を取り消す必要が生じた場合は、取り消しを行うレコードを選択して、**[変更の取り消し]** ボタンをクリックします。
5. レコードの編集を終えたら、**[保存]** ボタンをクリックします。レコードに対する変更が、例外リポジトリに保存され、表示が更新されます。再検証ワークフローをまだ定義していない場合は、例外リストが再度読み込まれます。ただし、編集したレコードのいくつかは、検索/フィル

タ条件にもう一致しなくなったために、更新後のリストには表示されない可能性があります。再検証ワークフローを*定義済み*の場合は、編集したレコードが有効になり、リポジトリから完全に削除されたことによって、更新後のリストには表示されない可能性があります。

6. 画面最下部のナビゲーション ボタンを使用して、前後の例外レコード ページに移動できます。また、これらのボタンを使用して、最初または最後の例外レコードに直接移動することもできます。

フォーム形式表示

フォーム形式で表示されたレコードを編集するには、以下の手順に従います。

1. **[フォーム形式で表示]** をクリックします。セット内の最初のレコードが表示されます。
2. 編集するフィールドをクリックし、フィールドの値を変更します。読み取り専用フィールドはグレー表示されます。フィールドを右クリックすると、切り取り、コピー、および貼り付けオプションにアクセスできます。フィールドを編集すると、フィールドの枠が緑色に変わります。この視覚的ヒントにより、フィールドの値が変更されているがまだ保存されていないことが一目でわかります。
3. **[コメント]** 列には変更に関するコメントを追加できます。コメントは他のユーザも見ることができ、レコードに対する変更を追跡するのに使用できます。
4. レコードを有効にするために必要な変更を確実に行ったら、**[承認済み]** ボックスをオンにします。これにより、Spectrum™ Technology Platformで処理可能としてレコードがマークされます。
5. 加えた変更を取り消す必要が生じた場合は、**[変更の取り消し]** ボタンをクリックします。
6. **[保存]** をクリックします。レコードに対する変更が、例外リポジトリに保存され、表示が更新されます。変更を加えたレコードが表示されるか、変更したレコードがもう使用できない場合や検索/フィルタ条件にもう一致しない場合は、リストの次のレコードが表示されます。
7. 画面最下部のナビゲーション ボタンを使用して、前後の例外レコードに移動できます。また、これらのボタンを使用して、最初または最後の例外レコードに直接移動することもできます。

重複レコードの解決

レコードが別のレコードの重複かどうかをSpectrum™ Technology Platformが確実に判断できない場合は、重複解決例外が発生します。重複レコードの解決方法は 2 つあります。

> **注：** 重複レコードは、テーブル形式に対する [重複を解決] の機能によってのみ解決可能です。ただし、これらのレコードをフォーム形式で編集することはできます。

1 つのアプローチは、重複レコードをコレクションにグループ化することです。レコードを承認すると、データの各コレクション内の重複レコードを除外する統合処理によって処理することができます。

もう 1 つのアプローチは、例えばストリート名のスペルを訂正するなど、レコードを重複として認識しやすくなるよう編集することです。レコードを承認すると、Spectrum™ Technology Platform

はマッチングおよび統合処理によってレコードを再処理します。レコードを正しく訂正した場合、Spectrum™ Technology Platformはそのレコードを重複として認識できるようになります。

重複レコードは、Best of Breed レコードを作成することによっても解決できます。このテクニックでは、レコード コレクションを管理し、コレクションのレコードから 1 つを選んで編集して、元のレコードと重複レコードの両方のフィールドを含めることで、他の 2 つのアプローチも取り入れています。このレコードは、Best of Breed レコードと呼ばれます。

レコードを別のレコードの重複とする

重複レコードは、Business Steward Portal にレコード グループとして表示されます。レコードを重複レコードと同じグループに移動することによって、別のレコードの重複とすることができます。

レコードを重複とするには、次の手順に従います。

1. 作業対象とするレコードを選択し、**[重複を解決]** をクリックします。

   [重複の解決] ビューに重複レコードが表示されます。レコードは、次のマッチ レコード タイプを含むコレクションまたは候補グループにグループ化されています。

suspect	互いに重複する関連性にあるかどうかを確認するために、他のレコードと比較するレコード。1 コレクションにサスペクト レコードは 1 つのみ含まれます。
duplicate	サスペクト レコードと重複するレコード。
unique	重複がないレコード。

   MatchRecordType 列を見ると、レコードのタイプを判断することができます。

2. 必要に応じて、個々のレコードを訂正します。詳細については、**例外レコードの編集**（264ページ）を参照してください。あるいは、グループ間でレコードをドラッグ アンド ドロップすることができます。セルのデータ タイプが同じならば、1 つのセルのコンテンツを別のセルにドラッグ アンド ドロップすることもできます (つまり、数値フィールドのコンテンツをテキスト フィールドにドラッグすることはできません)。
3. CollectionNumber フィールドまたは CandidateGroup フィールドに、レコードの移動先とするグループの番号を入力します。レコードが、グループ内の他のレコードの重複となります。

   MatchRecordType の値が "サスペクト" のレコードは、別の重複のコレクションに移動できない場合があります。

   > **注 :** レコードは、例外を生成したデータフローで使用されるマッチング ロジックのタイプによって、CollectionNumber フィールドまたは CandidateGroup フィールドでグループ化されます。マッチングに関する追加情報が必要な場合は、Spectrum™ Technology Platform管理者に問い合わせてください。

4. レコードの変更が終了したら、**[承認済み]** ボックスをチェックします。これは、レコードをSpectrum™ Technology Platformで再処理できるようになったことを示します。
5. 変更を保存するには、**[保存]** ボタンをクリックします。

重複レコードの新しいグループの作成

互いに重複とするレコードの新しいグループを作成できる場合があります。新しいグループを作成できない場合もあります。新しいグループを作成できるかどうかは、例外レコードを生成したSpectrum™ Technology Platform の処理のタイプによって決定されます。

1. 作業対象とするレコードを選択し、**[重複を解決]** をクリックします。

   [重複の解決] ビューに重複レコードが表示されます。レコードは、次のマッチ レコード タイプを含むコレクションまたは候補グループにグループ化されています。

suspect	互いに重複する関連性にあるかどうかを確認するために、他のレコードと比較するレコード。1 コレクションにサスペクト レコードは 1 つのみ含まれます。
duplicate	サスペクト レコードと重複するレコード。
unique	重複がないレコード。

   MatchRecordType 列を見ると、レコードのタイプを判断することができます。

2. 必要に応じて、個々のレコードを訂正します。詳細については、**例外レコードの編集**（264ページ）を参照してください。
3. 新しいコレクションに配置するレコードを選択し、**[新しいコレクション]** をクリックします。新しいコレクションには、一意なコレクション番号が自動的に付与されます。選択したレコードは、サスペクト レコードになります。

   > **注：[新しいコレクション]** ボタンが表示されていない場合は、作業中のレコードに対して新しいコレクションを作成することはできません。新しいコレクションを作成できるのは、例外を生成したデータフローが Interlfow Match または Intraflow Match ステージを含む場合のみで、Transactional Match ステージを含む場合は作成できません。これらのマッチング ステージに関する詳細な情報については、Spectrum™ Technology Platform 管理者に問い合わせてください。

4. 新しいコレクションの番号をレコードの CollectionNumber フィールドに入力することによって、レコードをコレクションに追加します。
5. レコードの変更が終了したら、**[承認済み]** ボックスをチェックします。これは、レコードをSpectrum™ Technology Platform で再処理できるようになったことを示します。
6. 変更を保存するには、**[保存]** ボタンをクリックします。

レコードをユニークにする

重複レコードをユニーク レコードに変更するには

1. MatchRecordType フィールドに "Unique" と入力します。
2. レコードの変更が終了したら、**[承認済み]** ボックスをチェックします。これは、レコードを Spectrum™ Technology Platform で再処理できるようになったことを示します。
3. 変更を保存するには、**[保存]** ボタンをクリックします。

重複解決時に自動調整されるフィールド

Business Steward Portal の重複解決ビューでレコードを変更すると、レコードの新しい性質を反映するよう、一部のフィールドが自動的に調整されます。

表 21 : Interflow Match または Intraflow Match で処理されるレコード

操作	フィールドに自動適用される値
レコードを 1 つのコレクションから別のコレクションに移動	**レコードを重複コレクションに移動する場合** • MatchRecordType: 重複 • MatchScore: 100 • HasDuplicates: D (このフィールドはデータフローに Interflow Match ステージが含まれる場合にのみ存在します) **重複レコードをユニーク レコードのコレクション (コレクション 0) に移動する場合** • MatchRecordType: ユニーク • MatchScore: 変更なし • HasDuplicates: U (このフィールドはデータフローに Interflow Match ステージが含まれる場合にのみ存在します) **サスペクト レコードをユニーク レコードのコレクション (コレクション 0) に移動する場合** • MatchRecordType: ユニーク • MatchScore: 0 • HasDuplicates: N (このフィールドはデータフローに Interflow Match ステージが含まれる場合にのみ存在します)

操作	フィールドに自動適用される値
新しいコレクションの作成	• MatchRecordType: サスペクト • MatchScore: 値なし • HasDuplicates: Y (このフィールドはデータフローに Interflow Match ステージが含まれる場合にのみ存在します) **注 :** Interflow Match ステージを含むデータフローから得られたレコードの場合、InterflowSourceType フィールドの値が "input_port_0" のレコードのみがサスペクト レコードになることができます。

表 22 : Transactional Match で処理されるレコード

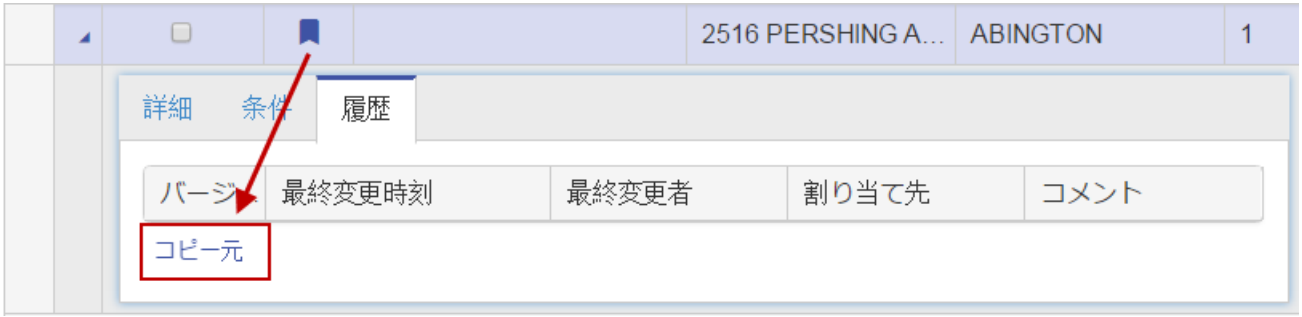
操作	フィールドに自動適用される値
MatchRecordType を重複に変更	• HasDuplicates: D • MatchScore: 100
MatchRecordType をユニークに変更	• HasDuplicates: U • MatchScore: 変更なし
HasDuplicates を D に変更	• MatchRecordType: 重複 • MatchScore: 100
HasDuplicates を U に変更	• MatchRecordType: ユニーク • MatchScore: 変更なし
HasDuplicates を Y に変更	• MatchRecordType: サスペクト • MatchScore: 空白
HasDuplicates を N に変更	• MatchRecordType: サスペクト • MatchScore: 空白

Best of Breed レコードの作成

Best of Breed レコードを作成するには、以下の手順に従います。

> **注：** Best of Breed レコードは、非ゼロ コレクションにのみ追加できます。また、Best of Breed レコードは、グループ内の各コレクションに対して作成する必要があります。そうしなければ、グループ内のすべてのレコードを承認できません。

1. コレクション内に Best of Breed レコードを作成するには、そのコレクションからレコードを選択し、**[Best of Breed レコードの追加]** ボタン () をクリックします。選択したレコードのコピーが作成され、例外エディター内で元のレコードの直後に配置されます。この Best of Breed レコードを保存すると、位置がレコード コレクションの最後に移動します。Best of Breed レコードの履歴を確認すれば、コピー元のレコードへのリンクがあることがわかります。



> **注：** この操作を取り消すには、Best of Breed レコードを選択してから **[Best of Breed レコードの削除]** ボタン () をクリックします。

2. 複数のフィールドを Best of Breed レコードに結合するには、重複レコードから複数のレコードをドラッグして Best of Breed レコードにドロップします。この環境でドラッグ アンド ドロップ操作を行う際は、以下の制限事項があることに注意してください。

- レコード間でドラッグしたセルを "編集" モードにすることはできません (そのセルをドロップした先のセルは "編集" モードにすることができます)。
- セルを別のセルにドロップした場合、そのセルが同じ行にあるか別の行にあるかに関係なく、ドロップ先のセルはドロップしたセルの値を受け取ります。
- Boolean 型のセルはドラッグ アンド ドロップで操作できません。
- 文字列型以外のセルは文字列型のセル (日付フィールドなど) にドラッグできますが、文字列型のセルを文字列型以外のセルにドロップできません。
- 検索値 (コンビネーション ボックス エディターなど) を使用するセルに別のセルをドロップすることはできませんが、このようなセルから値を他の文字列型のセルにドラッグすることはできます。
- 読み取り専用のセルに他のセルをドロップすることはできません。

Best of Breed 機能の詳細については、**[例外データの書き出し]** オプションを参照してください。

検索ツールの使用

Business Steward Portal の例外エディターには、例外レコードを編集および承認し、正常に再実行するために役立つ情報の検索を支援する、検索ツールがあります。ツールには、Bing Maps や Spectrum™ Technology Platform でライセンス取得したサービスが含まれます。アイコンの中央にプラス記号 (+) が付いているツールはプレミアム ツールであり、使用には料金が発生します。

> **注：** これらの検索ツールは、最初に、Management Console の Business Steward の設定ページで設定する必要があります。設定ページでツールが選択されていないと、このページにツールが表示されません。また、Management Console でどのツールを選択するかに関わらず、ここでユーザが表示できるのは、そのユーザが表示権限を持つサービスのみです。

Bing Maps の使用

Bing Maps は、住所の位置を地図上に表示し、地図のズームや移動を行うためのコントロールを提供します。また、地図をクリックすることにより、住所を取得することができます。

> **注：** Bing Maps 検索ツールは Microsoft によって提供されており、このサービスを利用するにはインターネットに接続する必要があります。

1. 調査するレコードをクリックします。
2. レコード テーブルの下の **[検索ツール]** をクリックして、ビューを展開します。
3. **[サービス]** フィールドで、**[Bing Map]** を選択します。
4. 検索に使用するフィールドを、**[フィールド名]** 列から選択します。例えば、地図上で住所を検索するために、AddressLine1 と City が選択できます。地図上に都市のみを表示する場合は、City と StateProvince のみを選択することができます。
5. 従来の地図を表示する場合は "道路"、航空写真を表示する場合は "バーズ アイ"、どちらが適切であるかを Bing に判断してもらう場合は "自動" を選択します。
6. **[Go]** をクリックします。緯度と経度を含む結果が、**[結果]** ボックスと地図上にピンの形で表示されます。回転ボタンをクリックすると、表示が 90 度回転します。コンパス上の矢印をクリックすると、選択された方向に少しずつ中心が移動します。縮尺を変更するには、[拡大] と [縮小] のボタンを使用します。また、ピンを新しい場所にドラッグして動かすと、地図情報が動的に更新されます。
7. 他の建物の住所を取得するには、地図上をクリックします。[バーズ アイ] ビューに切り替えると、建物が検索しやすい場合があります。

最初の地図検索を終えた時点で、別の例外レコードを選択して [Go] ボタンをクリックすると、それに応じて地図が更新されます。

Spectrum サービス検索ツールの使用

Pitney Bowes サービス検索ツールには、ValidateAddress や GetPostalCodes など、ライセンスを付与されているすべてのサービスが含まれます。これらのサービスを例外エディターで使用して、修正または増補しようとしている例外データを検索して検証できます。

この機能を使用する場合、役割セキュリティのプラットフォーム グループの下で、サービスに対する表示権限を保有している場合のみ、サービスが表示されることに注意してください。同様に、サービスを実行するには、サービスの実行権限が必要です。ただし、これらの権限は、セキュアエンティティ オーバーライドを使用して変更可能です。トップレベルの権限とオーバーライドを組み合わせて使用することで、管理者は BSM Portal Service ドロップダウンにおいて、特定のユーザや役割がアクセス可能なサービスのリストを管理できます。

1. 検索するデータを含むレコードを選択します。
2. 例外エディターで、**[検索ツール]** をクリックします。
3. **[サービス]** フィールドで、ValidateAddress や GetCandidateAddresses など、使用するサービスを選択します。
4. 例外レコードに含まれるフィールドが、そのサービスにおいて異なる名前で使用されている場合は、**[サービス フィールド]** タブで、サービス入力フィールドと例外フィールドのマッピングを行います。例えば、ValidateAddress を使用しており、例外レコードに AddressLine1 フィールドがなくその代わりに AddrLine1 フィールドが含まれている場合、[AddressLine1] 行の [例外フィールド] 列で、"AddrLine1" を選択します。サービスを実行する前に、少なくとも 1 つの入力フィールドをマッピングする必要があります。

   注： Business Steward Portal では、例外レコードを同じフィールド名、ステージ名、データフロー名でマッピングしている限り、ユーザが作成したサービス フィールドから例外フィールドへのマッピングを記憶します。例えば、例外レコードに "AddrLine1" という名前のフィールドがあり、そのフィールドを "AddressLine1" にマッピングすると、"AddrLine1" を含み、同じデータフローの同じステージによって作成されたレコードを操作している限り、このマッピングが記憶されます。

5. サービス出力フィールドについて、ステップ 4 を繰り返します。この操作は必須ではありませんが、サービスデータを適用する前に少なくとも 1 つの出力フィールドをマッピングしておく必要があります。読み取り専用の例外フィールドをサービス出力フィールドとマッピングすることはできないので注意してください。
6. データベース リソースを選択し、Management Console で設定されたサービス オプションを表示および変更するには、**[オプション]** タブをクリックします。特定のオプションの用途がわからない場合は、そのオプションにカーソルを合わせると説明が表示されます。ここで加えた変更は、例外レコードが同一のユーザ、データフロー、ステージで使用される場合に維持されます。

   注： 使用しているサービスがデータベースを必要とする場合は、Management Console でデータベース リソースを設定しておく必要があります。例えば、Validate Address を

使用して米国のレコードを確認する場合は、Management Console で米国データベースを設定しておく必要があります。

7. **[サービスの実行]** をクリックします。更新されたレコードが、レコードの成功を示すステータス コードとともに **[結果]** タブに表示されます。例外レコードとマッピングされているフィールドにはアスタリスクが表示されます。
8. 結果のレコードを選択し、**[サービス データの適用]** をクリックして、データを例外レコードのマッピング先フィールドに転送します。
9. 更新されたレコードを再処理する場合は、そのレコードの **[承認済み]** チェック ボックスをクリックし、**[保存]** をクリックします。

[管理] ページ

[管理] ページにおいて、表示と変更の権限を持つユーザは、すべての割り当て先の例外レコード アクティビティを確認および管理できます。また、このページでは、例外レコードをあるユーザから別のユーザに再割り当てすることもできます。削除権限を持つ場合は、データフロー名とジョブ ID に基づいて、例外レコードのグループ全体をシステムから削除可能です。

例外レコードのアクティビティの確認

[例外の管理] ページの **[ステータス]** セクションには、割り当てまたはデータフロー名ごとに例外レコードのアクティビティが表示されます。画面右上隅近くの [割り当て] ボタンまたは [データフロー] ボタンをクリックすることにより、どちらを表示するかを指定できます。[割り当て] は、各ユーザに割り当てられている例外レコードの数、承認されたレコードの数および残りのレコードの数を表示します。また、例外レコードも含まれているグループ内に存在する非例外レコードの数も表示されるようになりました。非例外データが表示されるのは、例外レコードも含まれるグループ内に非例外レコードがあり、このデータの表示を妨げるセキュリティ オーバーライドがない場合に限られます。

[データフロー] は、各データフローに対して承認されたレコードの数と割合を表示します。複数の Exception Monitor ステージを含むデータフローの場合、それらのステージ別に、この情報の詳細を得ることができます。例えば、データフローに 2 つの Exception Monitor ステージが含まれ、"NameExceptions" はデータフローの全例外の 37% を分担し、"DataExceptions" は残りの 63% を分担するとします。このような場合、各**データフロー**の状態は、リポジトリ内のすべてのデータフローに関連するものであることから、全体として計算されますが、各**ステージ**の状態は、選択したデータフローの例外の総数に基づいて計算されます。

さらに、[フィルタ] 行に検索条件を入力することによって、表示された情報をフィルタリングできます。リストには、入力した文字に一致するデータフローまたは割り当て先が動的に自動設定されます。

このページの **[進捗]** セクションでは、ダッシュボードに個別の進捗状況が表示されているすべてのユーザまたはすべてのデータフローの累積的な進捗が表示されます。ユーザの進捗を表示する

には画面上部の [割り当て] を、データフローの進捗を表示するには [データフロー] を選択します。スケールを使用して、スケールの測定単位 (日、週、月) と、表示するユニット数を選択します。

例外レコードの割り当て

[例外の管理] ページ (**[管理]** の下) の **[割り当て]** セクションでは、あるユーザから別のユーザに例外レコードを再割り当てできます。

1. 別のユーザに例外レコードを割り当てるユーザを、**[ユーザ]** フィールドで選択します。
2. ユーザに属する例外レコードをすべて再割り当てする場合は、手順 4 に進みます。ユーザの例外レコードを一部だけ再割り当てする場合は、次のうち該当するフィールドに値を設定します。
   - データフロー名 — 例外レコードを生成するデータフローの名前。
   - ステージ ラベル — 例外レコードを生成するステージの名前。
   - ジョブ ID — 例外レコードを含むジョブに割り当てられる ID。
   - データ ドメイン — Exception Monitor で割り当てる**データ**の種類。
   - 品質指標 — Exception Monitor で割り当てる**指標**の種類。
   - 開始日/時刻 — 例外レコードが作成された日付範囲の開始日。
   - 終了日/時刻 — 例外レコードが作成された日付範囲の終了日。
   - 承認ステータス — 例外レコードが承認済みかどうか。
3. 少なくとも **[ユーザ]**、**[データフロー名]**、**[ステージ ラベル]** の各フィールドを選択した後、例外フィールドの値に基づいてフィルタをさらに調整できます。
   a) [Field Filter を追加] アイコンをクリックします。
   b) **[フィールド名]** 列で、フィルタリングを実行するフィールドを選択します。
   c) **[演算子]** 列で、次のいずれかを選択します。

が次の値と同じ	指定した値と完全に同じ値を持つレコードを探します。数値またはテキスト値を指定できます。例えば、MatchScore の値が 82、または LastName の値が "Smith" のレコードを検索できます。
が次に等しくない	指定した値以外の値を持つレコードを探します。数値またはテキスト値を指定できます。例えば、MatchScore の値が 100 以外、または LastName が "Smith" 以外のレコードを検索できます。
が次の値より大きい	指定した値より大きい数値を持つレコードを探します。
が次の値以上	指定した値以上の数値を持つレコードを探します。例えば、50 を指定した場合、選択したフィールドの値が 50 以上のレコードが表示されます。
が次の値より小さい	指定した値より小さい数値を持つレコードを探します。

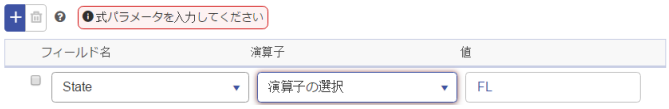
が次の値以下	指定した値以下の数値を持つレコードを探します。例えば、50 を指定した場合、選択したフィールドの値が 50 以下のレコードが表示されます。
が次の文字を含む	選択したフィールド内のいずれかの位置に指定した値を含むレコードを探します。例えば、AddressLine1 フィールドで "South" をフィルタリングする場合、"12 South Ave."、"9889 Southport St."、"600 South Shore Dr."、および "4089 5th St. South" という値を持つレコードが表示されます。
が次を含まない	選択したフィールド内のどの位置にも指定した値が含まれないレコードを探します。例えば、"が次を含まない" を選択して AddressLine1 フィールドで "South" をフィルタリングする場合、"12 South Ave."、"9889 Southport St."、"600 South Shore Dr."、および "4089 5th St. South" という値を持つレコードは表示されません。
が次の文字から始まる	選択したフィールドが特定の値で始まるレコードを探します。例えば、LastName フィールドで "Van" をフィルタリングする場合、"Van Buren"、Vandenburg"、または "Van Dyck" という値を持つレコードが表示されます。
が次の値で終わる	選択したフィールドが特定の値で終わるレコードを探します。例えば、City フィールドが "burg" で終わるレコードをフィルタリングする場合、"Gettysburg"、"Fredricksburg"、および "Blacksburg" という値を持つレコードが表示されます。

d) **[フィールド値]** 列にフィルタリング条件として使用する値を入力します。

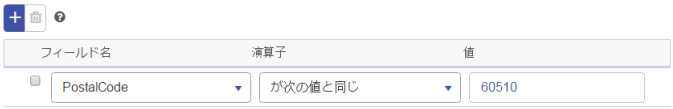
> **注 :** 検索値は大文字と小文字が区別されます。つまり、SMITH を検索すると、"smith" や "Smith" ではなく、すべて大文字の "SMITH" という値を持つレコードのみが返されます。

e) 複数のフィールドに基づいてフィルタリングを行うには、[Field Filter を追加] アイコンを再度クリックして、複数のフィルタを追加します。例えば、LastName の値が "SMITH" で State の値が "NY" のレコードをすべて取得したい場合は、LastName フィールド用と State フィールド用に 2 つのフィルタを使用することができます。

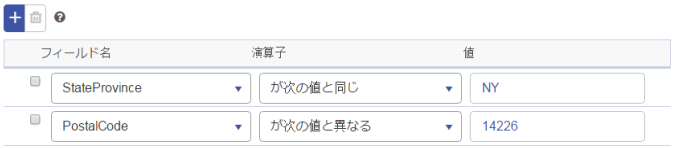
この例では、State フィールドの値が "FL" のレコードがすべて返されます。



この例では、PostalCode の値が 60510 ではないレコードがすべて返されます。



この例では、StateProvince の値が "NY" で、郵便番号が 14226 以外のレコードがすべて返されます。



4. **[再割り当て]** をクリックします。
5. これはオプションです。再割り当てする例外の数を選択します。得られた例外のすべてまたは一部を、新しいユーザに割り当てることができます。例えば、上限として "10" を入力すると、条件を満たす最初の 10 件のレコードのみが再割り当てされ、条件を満たす残りのレコードは再割り当てされません。
6. **[再割り当て]** ドロップダウンで、別のユーザを選択します。
7. **[確認]** をクリックします。

例外レコードの削除

例外レコードをリポジトリから削除したい場合もあります。例えば、システムのテストで使用して残ったレコードや処理後に例外であると誤って判断されたレコードがある場合や、承認済みのレコードを先に処理して削除してから同じジョブを再実行したい場合が挙げられます。例外レコードの削除は、[管理] ページ (**[管理]** の下) の **[完全削除]** セクションで行うことができます。

[削除] をクリックする前に、**[データフロー名]** と **[ジョブ ID]** の両方のフィールドで選択を行う必要があります。ただし、[ジョブ ID] フィールドで "すべて" を選択すると、選択したデータフローによるすべてのジョブ実行から例外レコードを削除できます。すべてのパフォーマンス データを削除するには、**[データ品質レポート データを削除]** をクリックします。このオプションを選択しない場合、ジョブの例外レコードはリポジトリから削除されますが、パフォーマンス データは [データ品質] ページにその後も表示されます。

[データ品質] 画面

Business Steward Portal の [データ品質] 画面には、例外レコードのトレンドに関する情報が表示されます。

> **注：** Management Console の [Business Steward の設定] 画面でデータ品質レポートを無効にすると、この画面は無効になります。

トレンドの特定

[データ品質トレンド] ページには、次の統計情報がデータフロー別およびドメイン別に表示されます。

- 処理された状況の総数
- 例外状況の総数
- 正しく処理された状況の割合
- 30 日間におけるデータのトレンド

データフローを選択すると情報がさらにステージ別に分割され、ドメインを選択すると指標別に分割されます。ここに表示される値は、過去 30 日間にわたって実行された Exception Monitor ステージがあるデータフローと各ドメインや指標に由来するものです。

- 情報をドメイン別に表示している場合は、**[データフロー名]** を選択すると特定のデータフローの情報を表示できます。また、**[ステージ ラベル]** を選択すると、特定のステージの情報を表示できます。ステージに基づいて結果にフィルタリングも行う場合は、単一のデータフローを選択する必要があることに注意してください。そうしなければ、すべてのデータフローからデータが表示されます。
- 情報をデータフロー別に表示している場合は、**[ドメイン]** を選択すると特定のドメインの情報を表示できます。また、**[指標]** を選択すると、特定の指標の情報を表示できます。それ以外の場合は、すべてのドメインのデータが表示されます。
- データをどこまでさかのぼるかを指定する **[スケール]** の期間を選択します。任意の日数、週数、月数を選択できます。
- 結果をステージ別に分割するには、データフローを選択します。
- 結果をデータ品質指標別に分割するには、ドメインを選択します。

データの異なる部分を表示するために選択を行うと、そのデータの状態を反映してグラフの内容が変わります。例えば、ドメイン データを参照する場合、特定のドメイン ("住所" など) 別にデータを絞り込んだ後で、特定の指標 ("完全性" など) 別にさらに絞り込むことができます。別の選択を行うたび、グラフの内容がそれに応じて更新されます。

Business Steward 構成

はじめに

Business Steward 構成の概要

Business Steward 構成は、書き込み権限を持つユーザに次の機能を提供します。

- 検索 — 特定のデータセットを修正に使用する例外レコード修正機能を提供します。

- ドメイン — 評価するデータの種類を指定できます。
- メトリクス — データを測定する方法を指定できます。
- 通知 — 特定のドメインまたはメトリクスに関連付けられた例外が一定の件数に達した時点で 1 つ以上の電子メール アドレスにメッセージを送信できます。

Business Steward 構成ツールへのアクセス

Business Steward 構成ツールにアクセスするには

1. Web ブラウザで次の URL を表示します。

   http://*server*:*port*/managementconsole

   ここで *server* は、 Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* はSpectrum™ Technology Platformが使用する HTTP ポートです。デフォルトの HTTP ポートは 8080 です。

2. 有効なユーザ名とパスワードを入力します。
3. **[リソース]** ボタンをクリックします。
4. **[Business Steward 構成]** を選択します。

検索

検索ツールを使用すると、例外エディターでレコードを更新する際に、特定のフィールドの値のリストから値を選択することができます。この機能は、変更したいフィールドにデータを含むレコードが複数ある場合に、特に便利です。例えば、銀行データを含む例外レコードがいくつかあるとします。このようなデータには、例えば、そのレコードに関連付けられた口座の種類を表すコード (1 = 当座預金、2 = 貯蓄、3 = 金融市場など) で構成されるフィールドを含めることができます。これらの例外レコードには住所の [国] フィールドに国名ではなく ISO コードが含まれており、住所の検証が不可能になっていると仮定します。これを修正するには、ISO コードと国名のペアを提供する検索を作成し、例外エディターで修正を実行します。例外エディターで ISO コードをリストから選択すると、その ISO コードに対応する国名がフィールドに設定されます。

このツールを使うもう 1 つの利点は、修正に使えるオプションを制限できることです。修正しようとしてかえってエラーを招く可能性を減らすことができます。先程と同様に、国名の誤りまたは欠落があると仮定します。例外レコードごとに国名を手動で入力するよう要求するのではなく、国名のリストを提供する検索を作成することで、国名のスペルの間違いを排除し、修正後に正しく検証できるレコードになる可能性を高めることができます。

検索プロセスの概要

検索プロセスは、3 つのステップから構成されます。例外を調査し、それらに共通して見られる問題点 ([国] フィールドに無効なデータがある、など) を特定した後で、検索を実行します。

- 誤ったデータを上書きする正確なデータの値ペアまたは値 / ラベル ペアを使って、検索を作成します。
- 例外レコードを生成しているデータフローで Write Exceptions ステージを使って、作成した検索に問題のフィールドを指定し、データフローを再実行します。
- 問題となっているフィールドの誤ったデータを検索から提供される正しいデータで上書きし、Business Steward Portal の例外エディターで例外レコードを修正します。

検索の作成

検索は値のペアまたは値とラベルのペアで構成され、データフロー内で例外レコードが生成される原因となっているデータを置き換える値を指定できます。値は問題のデータと置き換えるために使用し、ラベルは Exception Editor でレコードを修正するために検索テーブルを使うときにリストから選択する項目として使用します。

> **注：** 検索テーブルに値だけを含めた場合は、その値がラベルとしても使用されます。

検索の各項目を設定するには、情報を手動で入力するか、外部ソースからコピーして **[多数を追加]** ダイアログ ボックスに貼り付けます。情報がカンマ、タブ、またはセミコロンを区切り文字とする 1 つまたは 2 つの列で記述されている限り、スプレッドシートやテキスト ファイルに限らずほとんどあらゆる形式のファイルを外部ソースに使用できます。

> **注：** **[多数を追加]** 機能を使ってから **[保存]** をクリックすると、その検索に設定した既存の値ペアまたは値 / ラベル ペアは削除されます。ただし、[多数を追加] 機能を使った後で、値ペアや値 / ラベル ペアを手動で追加できます。

1. **[検索を追加]** ボタンをクリックします。
2. 新しい検索の名前をテキスト ボックスに入力します。
3. 値 / ラベル ペアを手動で追加するには
   - **[検索値を追加]** ボタンをクリックします。
   - 検索に使う値またはラベル (またはその両方) を入力します。

   [多数を追加] 機能を使うには
   - **[多数を追加]** ボタンをクリックして、ダイアログ ボックスを開きます。
   - 使用するデータに適切な列と区切り文字を選択します。データを Microsoft Excel から貼り付ける場合は、区切り文字としてタブを使用します。区切り文字の選択を誤ると、行全体が 1 つの値またはラベルとしてインポートされ、最初の列として設定されます。
   - すべてのコンテンツの値、区切り文字、またはラベルを入力するか、別のアプリケーションから貼り付けます。

   すべて値ペアまたは値 / ラベル ペアを追加した後で、[値] 列または [ラベル] 列を昇順または降順に並べ替えることができます。いったん並べ替えたリストは、並べ替える前の状態に戻せないことに注意してください (逆順に並べ替えたり、他の列で並べ替えることは可能です)。

4. ステップ 3 を必要に応じて繰り返します。
5. **[保存]** をクリックします。

検索の割り当て

検索を作成した後で、データフローの Write Exceptions ステージで問題となっているデータがあるフィールドに割り当てる必要があります。

1. Enterprise Designer で、例外レコードを生成しているデータフローを開きます。
2. Write Exceptions ステージを開きます。
3. 問題となっているデータがあるフィールドの **[検索名]** 列で、ドロップダウン リストから正しいデータを含む検索を選択し、**[OK]** をクリックします。
4. データフローを保存して再実行します。

レコードの作成

検索を作成し、データフローのフィールドに割り当てた後で、Business Steward Portal で例外レコードを修正する必要があります。

1. 例外エディターで、例外レコードを生成しているデータフローを選択します。
2. 問題となっている最初のレコードで、検索を割り当てたフィールドをクリックします。
3. そのフィールドのドロップダウン ボタンをクリックし、目的のレコードのラベルを選択します。このラベルが必ずしも値と一致するとは限りません。例えば、フィールドの値を "California" にしたい場合に "CA" と記されたラベルをクリックするようなケースがあります。
4. 問題となっているレコードのすべてについて、ステップ 3 を繰り返します。
5. 変更した例外を保存します。

検索の変更または削除

1. [検索] 画面を開きます。
2. 目的の検索の横にあるボックスをチェックします。
3. **[検索を編集]** ボタンをクリックし、必要に応じて検索を変更し、**[保存]** をクリックします。

   または

   [検索を削除] ボタンをクリックします。

ドメイン

ドメインは、評価対象となるデータの種類を指定します。データに発生した例外の種類を表示するレポート目的で使用されます。例えば、条件によって住所検証の成功/失敗を評価する場合、データ ドメインは "住所" です。条件によってジオコーディング操作の成功/失敗を評価する場合、データ ドメインは "空間" になります。

> **注：** ここで確立するドメインは、Business Steward 構成と Exception Monitor ステージの両方でデフォルト オプションとして使用されます。

以下に示す定義済みのドメインから 1 つを選択するか、**[項目を追加]** ボタンをクリックして必要なフィールドに値を入力し、独自のドメインを指定できます。また、ドメインを選択し、**[項目を編集]** ボタンをクリックして必要な変更を加えることでドメインを編集できます。検索データを **[フィルタ]** フィールドに入力すれば、表示されるドメインの一覧をフィルタすることもできます。結果は動的に更新されます。

- Account — セールス アカウントに関連付けられた企業または組織の名前をチェックします。
- Address — 完全な郵送先住所や郵便番号などの住所データをチェックします。
- Asset — 物理的資産、不動産、人材、その他のアセットなど、企業の資産に関するデータをチェックします。
- Date — 日付データをチェックします。
- Email — 電子メール データをチェックします。
- Financial — 通貨、証券などに関連するデータをチェックします。
- Name — 名や姓などの個人名データをチェックします。
- Phone — 電話番号データをチェックします。
- Product — 材料、部品、商品などに関するデータをチェックします。
- Spatial — 洪水発生地帯、海岸線、家屋、販売区域など定義済みの地勢を表す点、ポリゴン、または線データをチェックします。
- SSN — 米国社会保障番号データをチェックします。
- Uncategorized — この条件を分類しない場合は、このオプションを選択します。

メトリクス

メトリクスは、データを測定する方法を指定します。データに発生した例外の種類を表示するレポート目的で使用されます。例えば、レコードの完全性(すべての住所に郵便番号が含まれているかなど) を評価するための条件を設計する場合、データ品質指標として "完全性" を指定することができます。

> **注：** ここで確立するメトリクスは、Business Steward 構成と Exception Monitor ステージの両方でデフォルト オプションとして使用されます。

以下に示す定義済みのメトリクスから 1 つを選択するか、**[項目を追加]** ボタンをクリックして必要なフィールドに値を入力し、独自のメトリクスを指定できます。また、メトリクスを選択し、**[項目を編集]** ボタンをクリックして必要な変更を加えることでメトリクスを編集できます。検索データを **[フィルタ]** フィールドに入力すれば、表示されるメトリクスの一覧をフィルタすることもできます。結果は動的に更新されます。

- Accuracy — データを信頼できるソースに対して検証できるかどうかを評価します。例えば、郵政当局のデータを使用して住所を検証することはできません。正確ではないので、例外と見なされます。

- Completeness — データに不可欠な属性が欠落しているかどうかを評価します。例えば、郵便番号が欠落している住所や、連絡先が欠落しているアカウントなどです。
- Consistency — データが複数のシステム間で一貫しているかどうかを評価します。例えば、顧客データ システムでは M と F という性別コードが使用されているが、処理しているデータの性別コードが 0 と 1 の場合、データには一貫性の問題があると見なされます。
- Interpretability — データが他のシステムによって解釈可能なデータ構造に正しくパースされているかどうかを評価します。例えば、社会保障番号に含めることができるのは数値データだけです。このデータに xxx-xx-xxxx などの文字が含まれる場合、データには解釈可能性の問題があると見なされます。
- Recency — データが最新かどうかを評価します。例えば、ある個人が引っ越したが、システム内の住所が古いままの場合、データには最新の問題があると見なされます。
- Uncategorized — この条件を分類しない場合は、このオプションを選択します。
- Uniqueness — 重複データがあるかどうかを評価します。データフローが重複データを統合できなかった場合、そのレコードは例外と見なされます。

通知

通知機能を使うと、特定のドメインまたはメトリクスに関連付けられた例外が一定の件数に達した時点で 1 つ以上の電子メール アドレスにメッセージを送信できます。送信される電子メールには Business Steward Portal の例外エディター内の失敗レコードへのリンクが含まれ、例外エディターでは正しいデータを手動で入力できます。特定の電子メール アドレスでの通知の受信を停止するには、そのアドレスを、[ドメインを編集] ページの [通知送信先] 行にある受取人一覧から削除します。

> **注：** Business Steward 構成内で通知を正しく使用できるようにするには、Management Console で通知を設定する必要があります。通知の設定については、『管理ガイド』を参照してください。

1. Business Steward 構成で、**[ドメイン]** ページまたは **[メトリック]** ページを開きます。
2. 通知対象に追加するドメインまたはメトリックを選択し、**[項目を編集]** ボタンをクリックします。
3. 通知を送信する電子メール アドレスをドロップダウン リスト (Management Console で設定されたアドレスが含まれている) から選択するか、新しいアドレスを入力します。
4. 通知をトリガーする例外レコードの数を選択します。
5. 通知の件名に使うテキストを入力します。
6. 通知の本文に使うメッセージを入力します。例外に関する重要な情報を差し替えるため、メッセージ中で変数を使うことができます。使用できるのは次の変数です。
   - ${jobID} — 例外レコードを生成したジョブの ID 番号。
   - ${jobName} — 例外レコードを生成したジョブの名前。
   - ${userName} — 例外レコードを生成したジョブのユーザ名。

- ${stageLabel} — 例外レコードを生成したデータフロー ステージの名前。
- ${link} — 特定のデータフローのレコードが表示される、Business Steward Portal のエディター ページへのリンク。

7. リマインダー メッセージを送信する場合は **[リマインダーを送信]** ボックスをチェックし、リマインダーを送信するまでの経過日数を選択します。
8. リマインダー通知の件名に使うテキストを入力します。
9. リマインダー通知の本文に使うメッセージを入力します。リマインダーには次の変数も使用できます。

   - ${Count} — 特定のデータフローまたはステージで生成された未解決の例外の数。

10. 例外が解決されるまでリマインダーを毎日送信する場合は、**[リマインダーを毎日送信]** ボックスをチェックします。

Business Steward の設定

はじめに

Business Steward の設定の概要

Business Steward の設定は、書き込み権限を持つユーザに次の機能を提供します。

- オプション — 監査ログと進行状況の追跡に関する設定を指定します。
- データ品質レポート — 合否条件と KPI の追跡に関する設定を指定します。
- 検索ツール サービス — Buisness Steward Portal 例外エディタの検索ツールに関する設定を指定します。

Business Steward の設定へのアクセス

Business Steward の設定にアクセスするには

1. Web ブラウザで次の URL を表示します。

   http://*server*:*port*/managementconsole

   ここで *server* は、 Spectrum™ Technology Platform サーバーの名前または IP アドレスで、*port* はSpectrum™ Technology Platformが使用する HTTP ポートです。デフォルトの HTTP ポートは 8080 です。

2. 有効なユーザ名とパスワードを入力します。
3. **[リソース]** ボタンをクリックします。
4. **[Business Steward の設定]** を選択します。

オプション

1. **[監査ログの作成]** をクリックすると、例外レコードの作成、読み取り、更新、または削除が行われるとBusiness Steward モジュールによってログに記録が残されます。
2. **[進捗の追跡]** をクリックすると、Business Steward Portal で行った例外レコードの承認が追跡されます。このオプションをオフにすると、Business Steward Portal のダッシュボードに進捗グラフが表示されなくなります。

データ品質レポート

1. **[データ品質レポート]** をクリックすると、Exception Monitor ステージで合否条件を追跡するようになります。このオプションをオフにすると、Business Steward Portal の [データ品質] 画面にデータが表示されなくなります。同様に、全てのデータフローの Exception Monitor ステージのにおいて、[レポートのみ] フィールドが無効になります。
2. **[保持]** ドロップダウンで、このデータを保持する期間を月数で指定します。

主要パフォーマンス指標の設定

[データ品質レポート] タブの **[KPI の設定]** セクションでは、主要パフォーマンス指標 (KPI) をデータに指定し、それらの KPI が特定の条件を満たした場合の通知を割り当てることができます。

1. **[KPI の追加]** をクリックします。
2. 主要パフォーマンス指標の **[名前]** を入力します。この名前は Spectrum™ Technology Platform サーバー上で一意である必要があります。
3. この主要パフォーマンス指標で使用するデータ品質の **[指標]** を 1 つ選択します。ここで指標を選択しない場合、この主要パフォーマンス指標は、すべての**指標**に関連付けられます。
4. この主要パフォーマンス指標で使用する **[データフロー]** の名前を選択します。ここで名前を選択しない場合、この主要パフォーマンス指標は、すべての Business Steward Module データフローに関連付けられます。
5. この主要パフォーマンス指標で使用する **[ステージ ラベル]** を選択します。ここでステージ ラベルを選択しない場合、この主要パフォーマンス指標は、データフロー内のすべての Business Steward Module ステージに関連付けられます。
6. この主要パフォーマンス指標で使用するデータの **[ドメイン]** を選択します。ここでデータ ドメインを選択しない場合、この主要パフォーマンス指標は、すべての**ドメイン**に関連付けられます。ここでドメインを選択すると、[条件] フィールドが無効になることに注意してください。
7. 主要パフォーマンス指標で使用する **[条件]** を選択します。ここで条件を選択しない場合、この主要パフォーマンス指標は、デフォルトで "すべて" に設定されます。条件を選択するには、最初に [ドメイン] フィールドで "すべて" を選択する必要があることに注意してください。条件が選択されていると、[ドメイン] フィールドは無効になります。

8. **[KPI 期間]** を選択して、Business Steward モジュールでデータを監視して通知を送信する期間を指定します。例えば、"1" と "毎月" を選択した場合、基準とするしきい値または分散から前月比で例外の割合が増加していると、KPI 通知が送信されます。
9. **[相違]** または **[しきい値]** の割合を指定します。分散値は、前の期間からの例外レコードの失敗の増加率を表します。しきい値は、通知を送信するときの失敗の割合を表します。この値は 1 以上である必要があります。
10. これらの条件を満たした場合に送信する必要がある通知の **[受取人]** の電子メール アドレスをリストから選択するか、入力します。可能な場合、電子メールアドレスを入力し始めると、このフィールドは自動的に設定されます。複数のアドレスを設定する場合に、各アドレスをカンマ、セミコロン、またはその他の句読文字で区切る必要はありません。
11. 通知電子メールで使用する **[件名]** を入力します。
12. これらの条件を満たしたときに通知で送信する **[メッセージ]** を入力します。
13. **[OK]** をクリックします。他の既存の KPI の中に新しい KPI が表示されます。KPI は、データを含む任意の列でソートできます。
14. **[保存]** をクリックします。

KPI の変更や削除を行うには、**[選択した KPI を編集]** または **[選択した KPI を削除]** をクリックします。

検索ツール サービス

1. Buisness Steward Portal 例外エディターで使用する検索ツール サービスを選択します。使用可能なサービスはユーザの権限によって異なり、Spectrum Technology Platform で使用するためのライセンスを得ているモジュールとサービスによって選び出されます。**[フィルタ]** を使ってフィルタ条件を指定すると、サービスのリストが絞り込まれます。
2. **[プレミアム]** をクリックすると、これらのサービス (Dun & Bradstreet サービスなど) の使用に別途料金が課されることをユーザに示すことができます。

Data Normalization モジュール

Data Normalization モジュール

Data Normalization モジュールでは、レコード内の語を調べ、その語が好ましい形式であるかどうかを確認します。

- **Advanced Transformer —** このステージは、一連のデータをスキャンして分割し、抽出データと非抽出データを既存のフィールドまたは新しいフィールドに配置します。
- **Open Parser —** このステージは、世界のさまざまなカルチャーからの入力データを、シンプルかつ強力なパーシング グラマーでパースします。このグラマーを使用すると、ドメイン パターンを表す一連の式を入力データのパース用に定義できます。また、Open Parser は、統計データを収集してパーシング マッチにスコアを付けるため、パーシング グラマーの効果を調べるのも容易です。
- **Table Lookup —** このステージは、語を評価し、その語とその語の妥当性確認済みの形式とを比較します。語が適切な形式でない場合は、その語を標準バージョンに置き換えます。Table Lookup には、単語のフルスペルを省略形に変更する、単語の省略形をフルスペルに変更する、ニックネームを正しい名前に変更する、ミススペルを訂正する機能等があります。
- **Transliterator —** Transliterator は、ラテン文字など、異なるスクリプト (用字系) の間で文字列を変換します。

コンポーネント

Data Normalization モジュールは、次の要素で構成されます。

- **Advanced Transformer —** このステージは、一連のデータをスキャンして分割し、抽出データと非抽出データを既存のフィールドまたは新しいフィールドに配置します。
- **Open Parser —** このステージは、世界のさまざまなカルチャーからの入力データを、シンプルかつ強力なパーシング グラマーでパースします。このグラマーを使用すると、ドメイン パターンを表す一連の式を入力データのパース用に定義できます。また、Open Parser は、統計データを収集してパーシング マッチにスコアを付けるため、パーシング グラマーの効果を調べるのも容易です。
- **Table Lookup —** このステージは、語を評価し、その語とその語の妥当性確認済みの形式とを比較します。語が適切な形式でない場合は、その語を標準バージョンに置き換えます。Table Lookup には、単語のフルスペルを省略形に変更する、単語の省略形をフルスペルに変更する、ニックネームを正しい名前に変更する、ミススペルを訂正する機能等があります。
- **Transliterator —** Transliterator は、ラテン文字など、異なるスクリプト (用字系) の間で文字列を変換します。

Advanced Transformer

Advanced Transformer ジョブは、テーブルまたは正規表現を使用して、一連のデータ列をスキャンして複数のフィールドに分割します。このコンポーネントは、特定の語、あるいは語の右側または左側から指定した数だけ単語を抽出します。抽出データと非抽出データを既存のフィールドまたは新しいフィールドに配置できます。

例えば、次の住所フィールドから一組の情報を抽出し、その情報を別のフィールドに配置するとします。

2300 BIRCH RD STE 100

これを行うには、語 STE と語 STE の右側にあるすべての単語を抽出する Advanced Transformer を作成することができます、分離されるフィールド：

2300 BIRCH RD

入力

Advanced Transformer は、データ フロー内の定義済みの入力フィールドを使用します。

オプション

Advanced Transformer オプションは、任意の Spectrum™ Technology Platform クライアントによって、または実行時に、データフロー オプションを使用してステージ レベルで設定できます。

オプションの設定

Advanced Transformer のオプションを指定するには、ルールを作成します。複数のルールを作成して、ルールを適用する順序を指定することができます。ルールを作成するには

1. キャンバスで、Advanced Transformer のインスタンスをダブルクリックします。[Advanced Transformer オプション] ダイアログが表示されます。
2. 実行時インスタンスの番号を選択し、**[OK]** をクリックします。実行時インスタンス オプションを使用して、ステージの複数同時インスタンスを実行し、パフォーマンスを向上できるように、データフローを設定します。
3. **[追加]** ボタンをクリックします。[Advanced Transformer ルール オプション] ダイアログが表示されます。

   注： 複数の Transformer ルールを追加する場合は、**[上へ移動]** ボタンと **[下へ移動]** ボタンを使用して、ルールの適用順序を変更することができます。

4. 実行する変換動作のタイプを選択し、**[OK]** をクリックします。以下の表にオプションの一覧を示します。

表 23 : Advanced Transformer オプション

オプション	説明
ソース	スキャンおよび分割を評価するためのソース入力フィールドを指定します。

オプション	説明
次を使用して抽出	**[テーブル データ]** または **[正規表現]** を選択します。 **<ドライブ>:\Program Files\Pitney Bowes\Spectrum\server\modules\advancedtransformer\data** にある XML テーブルを使用してスキャンおよび分割を行う場合は、**[テーブル データ]** を選択します。各オプションの詳細については、後述のテーブル データ オプションを参照してください。 正規表現を使用してスキャンおよび分割を行う場合は、**[正規表現]** を選択します。正規表現には、データを分割するための追加オプションが多数あります。あらかじめパッケージ化された正規表現のいずれかをリストから選択して使用するか、Regex 構文を使用して独自の正規表現を作成することができます。 例えば、最初の数値が見つかったときにデータを分割することができます。"John Smith 123 Main St." では、"John Smith" が 1 つのフィールドに格納され、"123 Main St." は別のフィールドに格納されます。各オプションの詳細については、後述の正規表現オプションを参照してください。
テーブル データ オプション	
非抽出データ	変換したデータを格納する出力フィールドを指定します。元の値を置換する場合は、[ソース] ドロップダウン ボックスで指定したものと同じフィールドを [デスティネーション] フィールドで指定します。 [デスティネーション] フィールドに新しいフィールド名を入力することもできます。新しいフィールド名を入力した場合、そのフィールド名は、Advanced Transformer の下流にあるデータフローのステージに表示されるようになります。
抽出データ	抽出したデータを格納する出力フィールドを指定します。 [抽出データ] フィールドに新しいフィールド名を入力することができます。新しいフィールド名を入力した場合、そのフィールド名は、Advanced Transformer の下流にあるデータフローのステージに表示されるようになります。

オプション	説明
トークン化する文字	トークン化する特殊文字を指定します。トークン化は、語を分離する処理です。例えば、"Smith, John" というデータの入ったフィールドがある場合に、カンマをトークン化するものとします。このデータは、次の語に分離されます。 • Smith • , • John 語が分離されたら、スキャンしてカンマを抽出することによってデータを分割し、正規化するデータとして "Smith" と "John" を明確に識別することができます。
テーブル	フィールドの分割の基礎となる語を含むテーブルを指定します。テーブルの一覧は、**Advanced Transformer のテーブル**（158ページ）を参照してください。テーブルの作成または変更については、**検索テーブルの概要**（158ページ）を参照してください。
複数の単語からなる語を検索	指定された文字列内で複数の単語からなる語を検索できるようにするには、このチェック ボックスを選択します。例: 入力文字列 = "Cedar Rapids 52401"、ビジネス ルール = Cedar Rapids = US というエントリを含むテーブルに基づいて文字列内の "Cedar Rapids" を識別する、出力 = "Cedar Rapids" の存在を識別し、**[City]** などの新しいフィールドに語を格納するものとします。 複数の単語からなる語の検索では、最初にマッチが出現した時点で検索は停止します。 **注：** このオプションを選択すると、パフォーマンスに悪影響が生じる場合があります。

オプション	説明
抽出	実行する抽出のタイプを指定します。次のいずれかです。 **語を抽出** 選択したテーブルによって識別される語を抽出します。 **語の右側の N 語** 語の右側の N 語を抽出します。抽出する語数を指定します。例えば、識別された語の右側の 2 語を抽出する場合、2 を指定します。 **語の左側の N 語** 語の左側の N 語を抽出します。抽出する語数を指定します。例えば、識別された語の左側の 2 語を抽出する場合、2 を指定します。 語の右側または左側の語を抽出する場合、デスティネーション データまたは抽出データにその語自体を含めるかどうかを指定できます。例えば、 2300 BIRCH RD STE 100 というフィールドがあり、"STE 100" を抽出して、抽出データで指定したフィールドに格納する場合、抽出データ フィールドにこの語を含めることにします。すると、"STE" という略語と "100" という語が格納されます。 デスティネーション データも抽出データも選択しない場合、その語は格納されず、破棄されます。
正規表現オプション	
正規表現	あらかじめパッケージ化された正規表現をリストから選択するか、テキスト ボックスに独自の正規表現を作成します。Advanced Transformer は、標準の Regex 構文をサポートしています。 Java 2 プラットフォームには java.util.regex というパッケージが含まれており、正規表現を使用することができます。詳細については、**java.sun.com/docs/books/tutorial/essential/regex/index.html** を参照してください。
[省略記号] ボタン	新しい正規表現を追加または削除するには、このボタンをクリックします。
グループに追加	定義済みの Regex 式を選択するか、新しい Regex 式を入力したら、**[グループに追加]** をクリックして Regex グループを抽出し、検出されたすべての Regex グループと共に、式全体を [グループ] リストに配置します。

オプション	説明
グループ	この列には、選択した正規表現グループの正規表現が表示されます。 例えば、Date Regex 式を選択した場合、テキスト ボックスに次の式が表示されます。(1[012]{1,2}|0?[1-9])[-/.]([12][0-9]|3[01]{1,2}|0?[1-9])[-/.](([0-9]{4}))この Regex 式には 3 つの部分があり、式全体と各部分を異なる出力フィールドに送ることができます。式全体がソース フィールドで検索され、ソース フィールドでマッチが見つかった場合は、関連する部分が割り当てられた出力フィールドに移されます。ソース フィールドが "On 06/12/14" で、Date 式を適用する場合、データ全体 (つまり "06/12/14") を DATE フィールドに、"12" を MONTH フィールドに、"14" を DAY フィールドに、"2006" を YEAR フィールドに格納するよう割り当てます。日付が検索され、見つかった場合は、適切な情報が該当する出力フィールドに移されます。 ソース フィールド: "On 12/14/2006" DATE: "12/14/2006" MONTH: "12" DAY: "14" YEAR: "2006"
[出力フィールド] フィールド	出力フィールドを選択するためのプルダウン メニュー。

実行時におけるオプションの設定

Advanced Transformer のルールは、データフローのオプションとしてエクスポーズされている場合、実行時に設定して渡すことができます。これにより、既存の設定を JSON 形式の文字列によってオーバーライドできます。また、プロセス フローまたは Job Executor コマンドライン ツールからジョブを呼び出すときに、ステージ オプションを設定することもできます。

AdvancedTransformerRules のスキーマは、次のフォルダにあります。

```
<Spectrum Location>\server\modules\jsonSchemas\advancedTransformer
```

Advanced Transformer のルールを実行時に定義するには

1. Enterprise Designer で、Advanced Transformer ステージを使用しているデータフローを開きます。
2. そのデータフローを保存してエクスポーズします。
3. **[編集] > [データフロー オプション]** に移動します。
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]** テーブルで、Advanced Transformer を展開します。AdvancedTransformerRules のチェックボックスをオンにします。
5. オプション: **[オプション ラベル]** フィールドで、オプションの名前を変更します。
6. **[OK]** を 2 回クリックします。

出力

Advanced Transformer は、新しい出力フィールドを作成しません。定義したフィールドが出力に書き出されるだけです。

Open Parser

Open Parser は、世界のさまざまなカルチャーからの入力データを、シンプルかつ強力なパーシング グラマーでパースします。このグラマーを使用すると、ドメイン パターンを表す一連の式を入力データのパース用に定義できます。また、Open Parser は、統計データを収集してパーシング マッチにスコアを付けるため、パーシング グラマーの効果を調べるのも容易です。

Open Name Parser を使用して、次のことができます。

- ドメインエディタで定義したドメイン固有およびカルチャー固有のパーシング グラマーを使用して、入力データをパースします。
- ドメインエディタで使用できるものと同じ、シンプルかつ強力なパーシング グラマーを使用して Open Parser で定義した、ドメインに依存しないパーシング グラマーを使用して、入力データをパースします。
- データフロー オプションで定義したドメインに依存しないパーシング グラマーを実行時に使用して、入力データをパースします。
- ターゲット入力データファイルを使用してジョブを実行する前に、パーシング グラマーをプレビューして、サンプル入力データがどのようにパースされるかをテストします。
- パーシング グラマー結果をトレースして、定義した式にトークンがどれくらいマッチしたか、またはマッチしなかったかを確認し、マッチング処理に関する理解を深めます。

入力

Open Parser は、パーサー グラマーで定義した入力フィールドを受け入れます。詳細については、**Header セクションのコマンド**（30ページ）を参照してください。

カルチャー固有のパーシングを実行する場合、オプションで入力データに [CultureCode] フィールドを含めて、特定のカルチャーのパーシング グラマーをレコードに対して使用できます。[CultureCode] フィールドを省略した場合やこのフィールドが空の場合は、Open Parser ステージのリストにある各カルチャーが、指定されている順に適用されます。パーサー スコアが最も高いカルチャー、またはスコアが 100 になった最初のカルチャーの結果が返されます。CultureCode フィールドの詳細については、**レコードへのパーシング カルチャーの割り当て**（12ページ）を参照してください。

オプション

以下の表に、Open Parser ステージのオプションを示します。

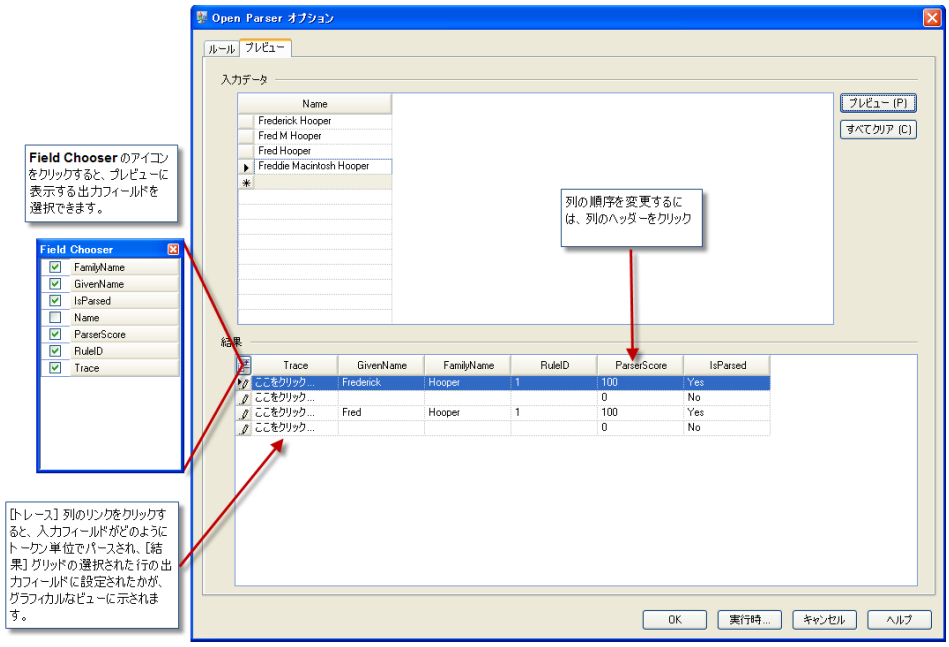
[ルール] タブ

オプション	説明
カルチャー固有のドメイン グラマーを使用	Enterprise Designer の Open Parser ドメインエディタ ツールで既に定義されている言語およびドメイン固有のパーシング グラマーの使用を指定します。ドメインの定義の詳細については、カルチャー固有のパーシング グラマーの定義（10ページ）を参照してください。 このオプションを選択した場合は、以下のオプションも表示されます。 **ドメイン** 使用するパーシング グラマーを指定します。 **カルチャー** パースするデータの言語またはカルチャーを指定します。カルチャーを追加するには、**[追加]** ボタンをクリックします。Open Parser が各カルチャーに関するデータのパースを試みる順序を変更するには、**[上へ移動]** および **[下へ移動]** ボタンを使用します。カルチャーの詳細については、カルチャー固有のパーシング グラマーの定義（10ページ）を参照してください。 **複数のパース済みレコードを返す** 入力を正しくパースしたカルチャーごとに Open Parser がレコードを返すようにするには、このオプションを有効にします。このボックスをチェックしない場合、Open Parser は、カルチャーにかかわらず、パーサー スコア 100 を達成した最初のレコードの結果を返します。パーサー スコア 100 のレコードにヒットすることなく、すべてのカルチャーが実行された場合、Open Parser は 100 に最も近いスコアのレコードを返します。複数のカルチャーが、100 未満の同じ高スコアのレコードを返す場合、ステップ 4 で設定した順序によって、どのカルチャーのレコードを返すかが決定されます。
ドメインに依存しないグラマーを定義	入力データの言語またはドメインを考慮せずに適用されるパーシング グラマーを定義する場合、このオプションを選択します。このオプションを選択すると、グラマー エディターが表示され、Enterprise Designer の Open Parser ドメインエディタ を使用せずに Open Parser ステージで直接パーシング グラマーを定義できます。 **注：** また、ドメインに依存しないグラマーを実行時に定義することもできます。詳細については、ここをクリックしてください。

[プレビュー] タブ

実用的なパーシング グラマーは、繰り返し作業によって作成されます。プレビューは、入力に対してさまざまなテストを行い、パーシング グラマーによって期待した結果が得られることを確認するのに役立ちます。

入力フィールドにテスト値を入力して、**[プレビュー]** をクリックします。



パースされた出力フィールドが **[結果]** グリッドに表示されます。出力フィールドについては、出力（295ページ）を参照してください。トレースについては、最終パース結果のトレース（38ページ）を参照してください。期待した結果が得られない場合は、**[ルール]** タブをクリックし、期待した結果が得られるまでパーシング グラマーの編集と入力データのテストを続行します。

出力

表 24 : Open Parser 出力

フィールド名	説明 / 有効な値
<入力フィールド>	パーシング グラマーで定義された元の入力フィールド。
<出力フィールド...>	パーシング グラマーで定義された出力フィールド。

フィールド名	説明 / 有効な値
CultureCode	入力データに含まれるカルチャー コード。サポートされている全カルチャー コードの一覧は、**レコードへのパーシング カルチャーの割り当て**（12ページ）を参照してください。
CultureUsedtoParseSelect a match results in the Match Results List and then click **Remove**.	各出力レコードのパースに使用されるカルチャー コードの値。この値は、カルチャー固有のパーシング グラマーとのマッチに基づいています。
IsParsed	出力レコードがパースされたかどうかを示します。有効な値は Yes または No です。
ParserScoreSelect a match results in the Match Results List and then click **Remove**.	合計平均スコアを示します。ParserScore の値は、パーシング グラマーでの定義に従って、0 ～ 100 の間になります。マッチが返されない場合は、0 が返されます。 詳細については、**スコアリング**を参照してください。
Trace	[結果] グリッドで選択された行に関して、パーシング グラマー内の各トークンが出力フィールドにどのようにパースされたかを視覚的に表示するには、このコントロールをクリックします。

Table Lookup

Table Lookup ステージは、語とその語の妥当性確認済みの形式とを照合して正規化し、正規化したバージョンを適用します。この評価は、正規化する語をテーブルから検索して実行されます。

例:

	名	姓
ソース入力:	Bill	Smith

正規化された出力:	William	Smith

正規化、特定、分類という 3 つのタイプのアクションを実行できます。

正規化アクションの実行時に語が見つかると、Table Lookup は、フィールド全体またはそのフィールド内の個々の語を正規化語に置き換えます。この置き換えは、フィールドに複数の単語が含まれている場合も行われます。Table Lookup には、単語のフルスペルを省略形に変更する、単語の省略形をフルスペルに変更する、ニックネームを正しい名前に変更する、ミススペルを訂正する機能等があります。

特定アクションの実行時に語が見つかると、Table Lookup は、そのレコードに対して、正規化可能な語を含むレコードとしてのフラグを設定しますが、アクションは実行しません。

分類アクションの実行時に語が見つかると、Table Lookup は、ソースの値をキーとして使用して、対応する値をテーブル エントリから、選択されたフィールドにコピーします。どのソース語もマッチしない場合、**[分類]** では指定のデフォルト値が使用されます。

入力

表 25 : Table Lookup 入力フィールド

フィールド名	説明 / 有効な値
Source	スキャンおよび分割を評価するためのソース入力フィールドを指定します。
StandardizationTable	**Table Lookup のテーブル**（161ページ）に一覧表示されているテーブルの 1 つ。

オプション

Table Lookup オプションは、任意の Spectrum™ Technology Platform クライアントによって、または実行時に、データフロー オプションを使用してステージ レベルで設定できます。

オプションの設定

Table Lookup のオプションを指定するには、ルールを作成します。複数のルールを作成して、ルールを適用する順序を指定することができます。ルールを作成するには、Table Lookup ステージを開き、**[追加]** をクリックして、次のフィールドを完成させます。

注： 複数の Table Lookup ルールを追加する場合は、**[上へ移動]** ボタンと **[下へ移動]** ボタンを使用して、ルールの適用順序を変更することができます。

オプション	説明
操作	ソース フィールドに対して実行するアクションのタイプを指定します。次のいずれかです。 **正規化** 検索テーブル内に見つかった正規化された語に一致するようにフィールド内のデータを変更します。フィールドに複数の語が含まれている場合、検索テーブルに見つかった語のみが正規化された語で置き換えられます。フィールド内のその他のデータは変更されません。 **特定** そのレコードに対して正規化可能な語を含むレコードとしてのフラグを設定しますが、フィールド内のデータに対するアクションは実行しません。フィールドを正規化できる場合は値 Yes を、正規化できない場合は値 No を持つ出力フィールド **StandardizedTermIdentified** がレコードに追加されます。 **分類** ソースの値をキーとして使用して、対応する値をテーブル エントリから [デスティネーション] リストで選択したフィールドにコピーします。これにより、レコードの分類に使用できる新しいフィールドがデータ内に作成されます。

オプション	説明
対象	フィールド全体を検索語として使用するのか、フィールド内の語ごとに検索テーブルを検索するのかを指定します。次のいずれかです。 **フィールド全体** フィールド全体を1つの語として扱い、結果として以下のようになります。 • **[正規化]** のアクションを選択した場合、Table Lookup はフィールド全体を1つの文字列として扱い、文字列全体を使用してフィールドを正規化します。例えば、"International Business Machines" は "IBM" に変更されます。 • **[特定]** のアクションを選択した場合、Table Lookup はフィールド全体を1つの文字列として扱い、その文字列全体を正規化できる場合はレコードにフラグを設定します。 • **[分類]** のアクションを選択した場合、Table Lookup はフィールド全体を 1 つの文字列として扱い、文字列全体を分類できる場合はレコードにフラグを設定します。 **フィールド内の個々の語** フィールド内の個々の語を独自の語として扱い、結果として以下のようになります。 • **[正規化]** のアクションを選択した場合、Table Lookup はフィールドをパースし、そのフィールド内の個々の語を正規化します。例えば、"Bill Mike Smith" は "William Michael Smith" に変更されます。 • **[特定]** のアクションを選択した場合、Table Lookup はフィールドをパースし、そのフィールド内に正規化できる単一の語があればレコードにフラグを設定します。 • **[分類]** のアクションを選択した場合、**[正規化]** と異なり、テーブルマッチがない場合、**[分類]** はソース語をコピーしません。どのソース語もマッチしない場合、**[分類]** では指定のデフォルト値が使用されます。**[正規化]** と異なり、**[分類]** はそのテーブル値のみを返し、**[ソース]** からは何も返しません。どのソース語もマッチしない場合、**[分類]** では指定のデフォルト値が使用されます。
ソース	検索する語が含まれている適切なフィールドを指定します。
ディスティネーション	テーブル検索によって返される語が書き込まれるフィールドを指定します。 値を置き換える場合は、**[ソース]** リストで指定したものと同じフィールドを **[デスティネーション]** リストで指定します。作成したいフィールドの名前を入力して、新しいフィールドを作成することもできます。 **[特定]** のアクションを選択している場合、**[デスティネーション]** フィールドは使用できません。

オプション	説明
テーブル	データフロー内のデータにマッチする語の検索に使用するテーブルを指定します。 編集可能なテーブルの一覧は、**Table Lookup のテーブル**（161ページ）を参照してください。テーブルの作成または変更については、**検索テーブルの概要**（158ページ）を参照してください。
複数の単語からなる語を検索	指定された文字列内での複数の単語の検索を有効にします。例: 入力文字列: "Major General John Smith" ビジネス ルール: エントリーを含むテーブルに基づいて文字列内の "Major General" を識別する 出力: "Major General" を "Maj.Gen." で置き換える 複数の単語からなる語の検索では、最初にマッチが出現した時点で検索は停止します。 **[対象]** が **[フィールド全体]** に設定されている場合、このオプションは無効になります。 **注：** このオプションを選択すると、パフォーマンスに悪影響が生じる場合があります。
テーブル エントリが見つからなかった場合、デスティネーションの値として次の値を設定する	マッチングする語が検索テーブル内に見つからなかった場合にデスティネーション フィールドに入れる値を指定します。次のいずれかです。 **ソースの値** ソース フィールドの値をデスティネーション フィールドに入れます。 **その他** 指定の値をデスティネーション フィールドに入れます。

実行時におけるオプションの設定

Table Lookup は、データフローのオプションとしてエクスポーズされている場合、実行時に設定して渡すことができます。これにより、既存の設定を JSON 形式の文字列によってオーバーライドできます。また、プロセス フローまたは Job Executor コマンドライン ツールからジョブを呼び出すときに、ステージ オプションを設定することもできます。

LookupRule のスキーマは、次のフォルダにあります。

```
<Spectrum Location>\server\modules\jsonSchemas\tableLookup
```

Table Lookup のルールを実行時に定義するには

1. Enterprise Designer で、Table Lookup ステージを使用しているデータフローを開きます。

2. そのデータフローを保存してエクスポーズします。
3. 編集 > データフローオプション に移動します。
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]** テーブルで、Table Lookup を展開します。LookupRule のチェックボックスをオンにします。
5. オプション: **[オプション ラベル]** フィールドで、オプションの名前を変更します。
6. **[OK]** を 2 回クリックします。

出力

表 26 : Table Lookup の出力

フィールド名	説明 / 有効な値
StandardizedTermIdentified	正規化可能な語がフィールドに含まれているかどうかを示します。**[フィールド全体]** オプションまたは [フィールド内の個々の語] オプションを選択した場合にのみ出力されます。 **Yes** レコードは正規化可能な語を含みます。 **No** レコードは正規化可能な語を含みません。

Transliterator

Transliterator は、ラテン文字など、異なるスクリプト (用字系) の間で文字列を変換します。例:

ソース	書き直し
キャンパス	kyanpasu
Αλφαβητικός Κατάλογος	Alphabētikós Katálogos
биологическом	biologichyeskom

書き直しは翻訳でないことを心に留めておくことが重要です。書き直しは、基盤となる語句を翻訳せずに、異なる用字系の間で文字を変換することを意味します。

> **注：** 標準の書き直し方法は、多くの場合、変換先の用字系における特定言語の発音ルールに従いません。

Transliterator ステージでは、以下の用字系がサポートされています。一般に、Transliterator ステージは、UNGEGN のラテン文字化方式に関する作業部会 (Working Group on Romanization Systems) のガイドラインに従います。詳細については、**www.eki.ee/wgrs** を参照してください。

アラビア文字 アラビア語、イラン語、ウルドゥー語など、複数のアジアおよびアフリカの言語で使われる用字系。

キリル文字 ロシア語などのスラブ系の言語を含む、東ヨーロッパおよびアジアの言語で使われる用字系。基本キリル文字セットに関しては、Transliterator ステージは一般に ISO 9 に従います。

ギリシャ文字 ギリシャ語で使われる用字系。

半角/全角 Transliterator ステージは、幅の狭い半角用字系と幅の広い全角用字系の間の変換を行えます。例えば、これは半角です。ｱﾙｱﾉﾘｳ。これは全角です。アルアノリウ。

ハングル 韓国語で使われる用字系です。Transliterator ステージは、韓国文化観光部の書き直し規則に従います。詳細については、**The National Institute of the Korean Language (韓国国立国語院)** の Web サイトを参照してください。

片仮名 日本語を書くために使うことができる複数の用字系の 1 つ。Transliterator ステージでは、ヘボン式と少しだけ異なる用字系が使われます。ヘボン式では、ZI (ジ) と DI (ヂ) は、どちらも "ji" で表記され、ZU (ズ) と DU (ヅ) はどちらも "zu" で表記されます。DI の代わりに "dji"、DU の代わりに "dzu" を使えば、書き直しの可逆性が少し改善されます。片仮名書き直しは可逆です。

ラテン文字 英語など、ヨーロッパの多くの言語で使われる用字系。

Transliterator は、Data Normalization モジュールのコンポーネントです。他のステージの一覧については、「**Data Normalization モジュール**（286ページ）」を参照してください。

書き直しの概念

用字系の書き直しには、一般に望ましいとされる品質要件がいくつか存在します。適切な書き直しに求められる要件は次のとおりです。

- 完了
- 予見性
- 発音可能性

- 明白性

これらの品質要件が同時に満たされることはまれなので、Transliterator ステージは、これらの要件のバランスを取ろうとします。

完了

変換元用字系における正しい形式の文字シーケンスが、変換先用字系の文字シーケンスにすべて書き直される、という要件です。

予見性

文字そのものが与えられれば (その用字系で記述される言語に関する知識がなくても)、比較的少数のルールに基づいて書き直しが可能である、という要件です。これで書き直しを機械的に実行できるようになります。

発音可能性

書き直しは、その発音に気を配らずにただ文字をマッピングするだけなら、ほとんど意味がありません。例えば、"αβγδεζηθ..." を "abcdefgh..." にマッピングするだけだと、完備性と明白性は満たされるかもしれませんが、発音できません。

標準の書き直し方法は、多くの場合、変換先の用字系における特定言語の発音ルールに従いません。例えば、日本語のヘボン式では (フランス語、ドイツ語、またはスペイン語とは対照的に) 英語の音価を持つ"j" が使われますが、使われる母音は標準的な英語の音声を持ちません。書き直し方法の中には、場合によって適切な発音を得るために特別な知識が必要とされるものもあります。例えば、日本語の訓令式では "tu" が "tsu" と発音されます。これは同じ用字系内に異なる言語が存在するような状況と似ています。例えば、単語 Gewalt がドイツ語由来の単語であるという知識があれば、"w" は "v" と発音されるでしょう。

場合によって、書き直しは慣習による影響を大きく受けます。例えば、現代ギリシャ語の文字 "ベータ" (β) の音声は "v" に似ていますが、変換によっては、(biology のように) そのまま b が使われることがあります。この場合、ユーザは書き直し後の単語内の "b" が "ベータ" (β) に対応し、現代ギリシャ語では "v" と発音されることを知っている必要があります。発音の予見性を高めるために文脈によって文字が異なる書き直しを受けることもあります。例えば、ギリシャ語の文字シーケンス "ガンマ ガンマ" (γγ) は "ng" と発音され、最初の "ガンマ" は "n" と文字化できます。

> **注：** 一般に、ラテン語の用字系を他の用字系に書き直すとき予見性の高い結果を得るために、英語のテキストで音声は生成されません。英語の発音は単語内の文字から簡単に予見できないからです。例えば、単語の grove、move、love はいずれも "ove" で終わっていますが、発音は大きく異なります。

明白性

変換先用字系の書き直し結果から変換元用字系のテキストを常に復元できる、という要件です。例えば、Elláda を変換前の Ελλάδα に復元できなければなりません。ただし、複数文字の書き直しでは曖昧性が生じることがあります。例えば、ギリシャ語の文字 "プサイ" (ψ) は ps にマッピングされますが、ps は文字シーケンス "パイ シグマ" (πσ) から生成される場合もあります。"パイ" (π) は p にマッピングされ、"シグマ" (σ) は s にマッピングされるからです。

この曖昧性の問題に対処するために、Transliterator はアポストロフィ (') を用いて文字シーケンスの曖昧性を解消しています。この手順に従うと、ギリシャ文字の "パイ シグマ" (πσ) は p's にマッピングされます。日本語では、変換先用字系で曖昧な文字シーケンスが単一の文字から生成されない限り、変換の曖昧性解消のためにアポストロフィ (') が用いられます。例えば、man'ichi と manichi を区別するためにこの手順が用いられます。

> **注：** 変換先用字系の文字の中には、限られた文脈以外であまり目にすることがないものがあります。例えば、日本語の "kya" (キャ) などに現れる小文字の "ya" (ャ) は、通常、単独では使われません。このような文字を扱うために、Transliterator はチルダ (~) を使います。例えば、入力が "~ya" なら単独の小文字 "ya" (ャ) が生成されます。ギリシャ文字への書き直しで、入力が "a~s" なら単語末尾でもギリシャ語のファイナルでない "シグマ" (ασ) が生成されます。同様に、入力が "~sa" なら単語末尾でなくてもファイナルな "シグマ" (ςα) が生成されます。

一般的な用字系の変換では、可逆性を得るための共通の技法として、通常なら区別できないような文字の間に特別なアクセント記号を用いることが行われています。例えば、以下は完全に可逆なラテン文字にマッピングされるギリシャ語テキストです。

入力

フィールド名	説明
任意の文字列フィールド	Transliterator ステージでは、任意の文字列フィールドを書き直すことができます。Transliterator ステージのオプションで、どのフィールドを書き直すかを指定できます。

フィールド名	説明
TransliteratorID	Transliterator ステージのオプションで指定されたデフォルトの書き直しをオーバーライドします。このフィールドは、レコードごとに異なる書き直しを指定したいときに使用してください。 次のいずれかです。 **Arabic-Latin** アラビア文字からラテン文字。 **Cyrillic-Latin** キリル文字からラテン文字。 **Greek-Latin** ギリシャ文字からラテン文字。 **Hangul-Latin** ハングル文字からラテン文字。 **Katakana-Latin** 片仮名からラテン文字。 **Latin-Arabic** ラテン文字からアラビア文字。 **Latin-Cyrillic** ラテン文字からキリル文字。 **Latin-Greek** ラテン文字からギリシャ文字。 **Latin-Hangul** ラテン文字からハングル文字。 **Latin-Katakana** ラテン文字から片仮名。 **Fullwidth-Halfwidth** 全角から半角。 **Halfwidth-Fullwidth** 半角から全角。

オプション

表 27 : Transliterator のオプション

オプション	説明/有効値
変換元	書き直したいフィールドで使われている用字系。サポートされている用字系については、**Transliterator**（301ページ）を参照してください。 **注 :** Transliterator ステージは、すべての用字系の間の書き直しをサポートしているわけではありません。**[変換元]** フィールドと **[変換先]** フィールドは、選択内容に応じて自動的に妥当な値を反映するようになっています。

オプション	説明/有効値
変換先	フィールドの変換先の用字系。サポートされている用字系については、**Transliterator**（301ページ）を参照してください。 **注：** Transliterator ステージは、すべての用字系の間の書き直しをサポートしているわけではありません。**[変換元]** フィールドと **[変換先]** フィールドは、選択内容に応じて自動的に妥当な値を反映するようになっています。
スワップ ボタン 	スワップ ボタンをクリックすると、**[変換元]** フィールドと **[変換先]** フィールドの間で言語が交換されます。
翻字するフィールド	書き直したいフィールドを指定します。
アクセント記号を削除	フィールドからアクセント記号を削除する場合は、このチェック ボックスをオンにします。このオプションは、デフォルトでオフのままになっています。

出力

指定したフィールドが Transliterator ステージにより書き直されます。それ以外の出力は生成されません。

Universal Name モジュール

Universal Name モジュール

正規化をきわめて正確に実行するには、一連のデータ列を複数のフィールドに分割する必要がある場合があります。Spectrum™ Technology Platformは、個人名、企業名、およびその他多くの語や略語をパースする高度なパーシング機能を備えています。また、スキャンおよび抽出操作のベースとして使用するカスタム語のリストを独自に作成することもできます。

コンポーネント

Universal Name モジュールを構成するコンポーネントを以下に示します。

- **Name Parser (廃止)** — 名前データ フィールドにある個人名、企業名、またはその他の名称を構成要素に分解します。パースされたこれらの名前要素は、名前のマッチング、名前の正規化、複数レコード名の統合など、他の自動化処理に使用できます。

  重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

- **Name Variant Finder** — 名前データベースにクエリを発行して名前の派生形を取得するために、姓または名のモードで使用されるコンポーネントです。Name Variant Finder は、性別やカルチャーで結果の件数を絞り込むことができます。Spectrum™ Technology Platformには基本名前ファイルが付属します。他のカルチャーのアドオン名前ファイルも用意されています。アドオン名前ファイルを展開するには、その JAR ファイルを`modules/tables/ext`フォルダにコピーします。アラブの名前は、アドオンとして使用できます。このファイルのデータは、Nomino, Inc. によって作成されました。
- **Open Name Parser** — Open Name Parser は、名前データ フィールドにある個人名、企業名、またはその他の名称を構成要素に分解しますパースされたこれらの名前要素は、名前のマッチング、名前の正規化、複数レコード名の統合など、他の自動化処理に使用できます。

Name Parser (廃止)

重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

Name Parser は、名前データ フィールドにある個人名、企業名、またはその他の名称を構成要素に分解します。パーシングの処理では、各要素が名前全体に対して持つ関連性が、機能、形式、および構文を基準に解釈されます。パースされたこれらの名前要素は、名前のマッチング、名前の正規化、複数レコード名の統合など、他の自動化処理に使用できます。

名前のパーシングは、次の手順で行われます。

- 名前が担う機能を示すために、そのエンティティ タイプを特定します。名前エンティティ タイプは、個人名と企業名の 2 つのグループに分かれます。それぞれのグループには、さらに複数のサブグループがあります。
- パーシングに使う構文を把握するために、名前の形式を特定します。個人名は、通常、自然な (署名) 順序または逆の順序に従います。企業名は、通常、階層型の順序に従います。
- 名前を構成する各要素が名前全体に占める構文上の関連性を識別するために、要素を特定してラベル付けします。個人名の構文は、敬称、名、ミドルネーム、姓、接尾語、アカウントを示す用語、その他の個人名要素で構成されます。企業名の構文は、メインのテキスト、重要性の低い用語、接頭語、接頭語の対象となる語、接尾語、その他の企業名要素で構成されます。

- 名前の性別を特定します。指定したカルチャーの情報に基づいて、性別が判断されます。たとえば、"Jean" はフランスでは男性名ですが、米国では女性名です。処理する名前がフランスに由来することがわかっていれば、フランスを性別判断カルチャーとして指定できます。Name Parser では、First Name テーブルと Compound First Names テーブルのデータを使って性別を判断します。名前がテーブルに見つからない場合は、敬称が名前に含まれていれば、Title テーブルを使って性別を判断します。それ以外の場合は、性別は不明とされます。

  注： 入力レコードのフィールドに、サポートされるカルチャーの一つが既に設定されている場合は、入力で GenderDeterminationSource フィールドを事前に設定することで、GUI 上の Gender Determination Source を上書きできます。

- パーサーで行われたパーシング処理の正確性の確信レベルを示すパーシング スコアを割り振ります。

入力

重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

表 28 : Name Parser 入力

フィールド名	説明 / 有効な値
GenderDeterminationSource	名前データのカルチャーは性別の判断に使います。デフォルトは異文化ルールに従います。例えば、Jean は一般に女性の名前で、デフォルトではそのように認識されますが、フランス語を選択した場合は男性の名前と認識されます。以下に、オプションを各カルチャーの代表的な国と共に示します。以下の国のリストは網羅的でありません。 **SLAVIC** ボスニア、ポーランド、アルバニア。 **ARMENIAN** アルメニア。 **DEFAULT** ブルガリア、ケイマン諸島、アイルランド、米国、イギリス。 **FRENCH** フランス。 **SCANDINAVIAN** デンマーク、フィンランド、アイスランド、ノルウェイ、スウェーデン。 **GERMANIC** オーストリア、ドイツ、ルクセンブルク、スイス、オランダ。 **GREEK** ギリシャ。 **HUNGARIAN** ハンガリー。 **ITALIAN** イタリア。 **PORTUGUESE** ポルトガル。 **ROMANIA** ルーマニア。 **HISPANIC** スペイン。 **ARABIC** チュニジア。 GenderDeterminationSource は Name Variant Finder でも使われ、取得する名前の派生形がカルチャーで制限されます。詳細については、**Name Variant Finder**（329ページ）を参照してください。
Name	パースしたい名前。このフィールドは必須です。

オプション

重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

Name Parser のオプションを指定するには、キャンバスで Name Parser のインスタンスをダブルクリックします。[Name Parser オプション] ダイアログが表示されます。

表 29 : Name Parser のオプション

オプション	説明
個人名をパース	個人名をパースするとき、これをオンにします。
結合名を複数のレコードに分割 マッチ結果リストのマッチ結果を選択し、**[削除]** をクリックします。	このボックスをクリックして、複数の人物を含む結合名、例えば、`Bill & Sally Smith` を複数レコードに分離します。 結合名が 2 つの名前レコードに分離されると、パーサーのレコード ID 出力フィールドが生成されます。分離された名前の各ペアは同じパーサー レコード ID で識別されます。
性別判定ソース マッチ結果リストのマッチ結果を選択し、**[削除]** をクリックします。	Name Parser の名前による性別判定方法を決定します。たいていのケースで、デフォルトは性別の判定に最適な設定であり、さまざまな名前に対応できます。ただし、特定のカルチャーに属する名前を処理する場合は、そのカルチャーを選択します。特定のカルチャーを選択すると、名前の性別が適切に判定される確率が高くなります。例えば、デフォルトのままで設定した場合、Jean という名前は女性と判定されます。フランス語を選択すると、男性と判定されます。 **注 :** カルチャーを選択しても名前がそのカルチャーで見つからないときは、さまざまなカルチャーのデータを記載したっでフォルト カルチャーが使われます。
順序	入力レコードの名前フィールドの順序を指定します。次のいずれかです。 **正順序** 名前フィールドの順序は、[敬称]、[名]、[ミドル ネーム]、[姓]、[接尾語] となります。 **逆順序** 姓が最初の順序になります。 **混在** 正順序と逆順序を組み合わせた順序になります。
ピリオドを残す	パースされる個人名のフィールドに句読文字を残します。
企業名をパース	企業名をパースするとき、これをオンにします。

オプション	説明
ピリオドを残す	企業名をパースするとき句読文字を残すにはこのチェックボックスをオンにします。
ユーザ定義テーブル	いずれかのユーザ定義テーブルをクリックして、各種パーサー テーブルの現在の値に値を追加します。これでビジネス環境に適合するようにテーブルをカスタマイズできます。[設定] をクリックして、追加したい値の入っている XML ファイルを選択します。ユーザ定義テーブルの詳細については、**Name Parser ユーザ定義テーブルの変更**（311ページ）を参照してください。

Name Parser ユーザ定義テーブルの変更

重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

Name Parser テーブル内の値を追加、変更、および削除することで、それらをビジネス環境に応じてカスタマイズできます。

Name Parser のユーザ定義テーブルは XML ファイルで、デフォルトでは `<Drive>:\Program Files\Pitney Bowes\Spectrum\server\modules\parser\data` フォルダにあります。Spectrum™ Technology Platform には次のユーザ定義テーブルが含まれます。

UserAccountDescriptions.xml

表 30 : UserAccountDescriptions.xml の列

列名	説明 / 有効な値
LookupValue	アカウント説明内でよく見つかる検索語。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>
```

```
                LookupValue
                ART
                AND

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            A/C
            ACCOUNT
            EXP

    </added-entries>
</table-data>
```

UserCompanyPrepositions.xml

表 31 : UserCompanyPrepositions.xml の列

列名	説明 / 有効な値
LookupValue	会社名によく使われる接頭語 (例えば、 "of" または "on")。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                AROUND
                NEAR

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            ABOUT
            AFTER
            ACROSS
```

```
    </added-entries>
</table-data>
```

UserCompanySuffixes.xml

表 32 : UserCompanySuffixes.xml の列

列名	説明 / 有効な値
LookupValue	会社名によく使われる接尾語。例としては、"Inc." や "Co." があります。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                SANDY
                CLUE

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            LTD
            LLC
            CO
            INC

    </added-entries>
</table-data>
```

UserCompanyTerms.xml

表 33 : UserCompanyTerms.xml の列

列名	説明 / 有効な値
LookupValue	会社名によく使われる一般語。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                MARY
                BLUE

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            ARC
            ARCADE
            ASSEMBLY
            ARIZONA

    </added-entries>
</table-data>
```

UserCompoundFirstNames.xml

このテーブルにはユーザ定義の複合ファースト ネームが入っています。複合名は 2 つの単語からなる名前です。

表 34 : UserCompoundFirstNames.xml の列

列名	説明 / 有効な値
FirstName	複合ファースト ネーム。最大 2 語。大文字と小文字は区別されません。
カルチャー	この FirstName/Gender の組み合わせが適用されるカルチャー。GenderDeterminationSource 入力フィールドで有効な任意の値を使用できます。詳細については、入力（308ページ）を参照してください。
性別	性別は一般にこの FirstName/Culture の組み合わせと関係します。次のいずれかです。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。このフィールドを空白にすると、未知と仮定されます。
頻度	今回のリリースでは使われていません。この列は空白でかまいません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                    FirstName
                    ANN MARIE
                    BILLY JOE

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Frequency
                KAREN SUE|0.126
                BILLY JOE|0.421

        </deleted-entry-group>
        <deleted-entry-group>
```

```
                FirstName|Gender|Culture
                JEAN ANN|M|DEFAULT
                JEAN CLUADE|F|FRENCH

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            FirstName|Gender|Culture
            JOHN Henry|M|DEFAULT
            A'SHA A'MAR|F|ARABIC
            BILLY JO|A|DEFAULT

    </added-entries>
</table-data>
```

UserConjunctions.xml

このテーブルはユーザ定義の接続詞 ("and"、"or"、"&" など) のリストです。

表 35 : UserConjunctions.xml の列

列名	説明 / 有効な値
LookupValue	任意の接続値。一語でなければなりません。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                FIND
                CARE
                %

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            &
            AND
            OR
```

```
    </added-entries>
</table-data>
```

UserFirstNames.xml

表 36 : UserFirstNames.xml の列

列名	説明 / 有効な値
FirstName	このテーブルの行に記述されるファースト ネーム。大文字と小文字は区別されません。
性別	性別は一般にこの FirstName/Culture の組み合わせと関係します。次のいずれかです。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。このフィールドを空白にすると、未知と仮定されます。
カルチャー	この FirstName/Gender の組み合わせが適用されるカルチャー。GenderDeterminationSource 入力フィールドで有効な任意の値を使用できます。詳細については、入力（308ページ）を参照してください。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                FirstName
                AADEL
                AADIL

        </deleted-entry-group>
        <deleted-entry-group>
```

```
                FirstName
                A'SACE
                A'BOCKETT

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Gender|Culture
                ALII|M|DEFAULT
                AISHA|F|ARABIC

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Gender
                JOHE|M

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            FirstName|Gender|Culture
            JOHE|M|DEFAULT
            A'SHAN|F|ARABIC

    </added-entries>
</table-data>
```

UserGeneralSuffixes.xml

このテーブルは個人名で使われる世代接尾語でないユーザ定義の接尾語 ("MD"、"PhD" など) のリストです。

表 37 : UserGeneralSuffixes.xml の列

列名	説明 / 有効な値
LookupValue	個人名によく使われ、世代接尾語でない接尾語。一語でなければなりません。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
```

```
        <deleted-entry-group>

                LookupValue
                AND
                WILL
                TUNA

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            ACCOUNTANT
            ATTORNEY
            ANALYST
            ASSISTANT

    </added-entries>
</table-data>
```

UserLastNamePrefixes.xml

このテーブルは個人名のラスト ネームで使われるユーザ定義の頭字語 ("Van"、"De"、"La" など) のリストです。

表 38 : UserLastNamePrefixes.xml の列

列名	説明 / 有効な値
LookupValue	個人名のラスト ネームで使われる頭字語。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                DO
                RUN
                ANIMAL

        </deleted-entry-group>
```

```
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            D'
            DA
            DEN
            DEL

    </added-entries>
</table-data>
```

UserLastNames.xml

表 39 : UserLastNames.xml の列

列名	説明 / 有効な値
LastName	このテーブルの行に記述されるラスト ネーム。大文字と小文字は区別されません。
性別	性別は一般にこの FirstName/Culture の組み合わせと関係します。次のいずれかです。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。このフィールドを空白にすると、未知と仮定されます。
カルチャー	この FirstName/Gender の組み合わせが適用されるカルチャー。GenderDeterminationSource 入力フィールドで有効な任意の値を使用できます。詳細については、入力（308ページ）を参照してください。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>
```

```
                LastName
                Rusod
                AADIL

        </deleted-entry-group>
        <deleted-entry-group>

                LastName
                KAASEEY
                JOIEN

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LastName|Culture|Gender
            SMITH|ENGLISH|A
            WILSON|ENGLISH|A
            JONES|ENGLISH|A

    </added-entries>
</table-data>
```

UserMaturitySuffixes.xml

このテーブルは、個人名で使われるユーザ定義の家族接尾語 ("Jr."、"Sr." など) のリストです。

表 40 : UserMaturitySuffixes.xml の列

列名	説明 / 有効な値
LookupValue	個人名で使われる家族接尾語。任意の一語です。大文字と小文字は区別されません。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                I
                V
                18
                VI
```

```
        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue
            I
            II
            III

    </added-entries>
</table-data>
```

UserTitles.xml

このテーブルは、個人名で使われるユーザ定義の肩書き ("Mr." or "Ms."など) のリストです。

表 41 : UserTitles.xml の列

列名	説明 / 有効な値
LookupValue	個人名で使われる肩書き。任意の一語です。大文字と小文字は区別されません。
性別	性別は一般にこの肩書きと関係します。次のいずれかです。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。このフィールドを空白にすると、未知と仮定されます。

入力例:

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                LookupValue
                Belt
                Friend
                Thursday
```

```
                Red

        </deleted-entry-group>
    </deleted-entries>
    <added-entries delimiter-character="|">

            LookupValue|Gender
            Mrs|F
            Mr|M
            Most|F

    </added-entries>
</table-data>
```

サンプルのユーザ定義テーブル

次の最初の図は、サンプルのテーブル UserFirstNames.xml と、ユーザ定義テーブルを変更するときの構文です。

```
<table-data>
    <deleted-entries delimiter-character="|">
        <deleted-entry-group>

                FirstName
                AADEL
                AADIL

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Frequency
                A'SACE|0.126
                A'BECKETT|0.421

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Gender|Culture|VariantGroup
                ALI|M|DEFAULT|GROUP88
                AISHA|F|ARABIC|GROUP43

        </deleted-entry-group>
        <deleted-entry-group>

                FirstName|Gender
                JOHN|M

        </deleted-entry-group>
    </deleted-entries>
```

```
    <added-entries delimiter-character="|">

            FirstName|Gender|Culture
            JOHN|M|DEFAULT
            A'SHA|F|ARABIC
            JAMES|M|DEFAULT

    </added-entries>
</table-data>
```

出力

重要： Name Parser ステージは廃止され、今後のリリースではサポートされない可能性があります。名前のパーシングには Open Name Parser を使用してください。

表 42 : Name Parser の出力

フィールド名	書式	説明 / 有効な値
AccountDescription	String	名前の一部であるアカウント説明。例えば、"Mary Jones Account # 12345" で、アカウント説明は "Account#12345"。
EntityType	String	名前の種類を示唆します。次のいずれかです。 **Firm** 名前または会社名。 **Personal** この名前は、個人の名前です。
会社名関係のフィールド		
FirmModifier.1.Object	String	会社名の接頭語の最初のオブジェクト。例えば、会社名 "Pratt & Whitney Division of United Technologies" で、接頭語の最初のオブジェクトは "United Technologies" です。
FirmModifier.1.Preposition	String	会社名で使われる最初の接頭語。例えば、会社名 "Pratt & Whitney Division of United Technologies" で、"of" が最初の接頭語です。

フィールド名	書式	説明 / 有効な値
FirmModifier.2.Object	String	会社名の接頭語の 2 番目のオブジェクト。例えば、会社名 "Church of Our Lady of Lourdes" で、接頭語の 2 番目のオブジェクトは "Lourdes" です。
FirmModifier.2.Preposition	String	会社名の 2 番目の接頭語。例えば、会社名 "Church of Our Lady of Lourdes" で、2 番目の接頭語は 2 番目の "of" です。
FirmName	String	会社名。例えば、"Pitney Bowes, Inc."。
FirmPrimary	String	会社名の基本部分。例えば、"Pitney Bowes"。
FirmSuffix	String	会社名の接尾語。例えば、"Co."、"Inc."
個人名に関係するフィールド		
FirstName	String	個人のファースト ネーム。
FirstNameVariantGroup	String	ファースト ネームが属す類似の名前のグループを示す識別番号。例えば、Muhammad、Mohammed、Mehmet はいずれも同じ名前派生グループに属します。実際のグループ ID はアドオン データの読み込み時に設定されます。 このフィールドには Name Variant Group 機能を実際に購入するまで値は設定されません。
GenderCode	String	個人名の性別はファースト ネームの分析時に決定されます。次のいずれかです。 **A** 曖昧。この名前は男性と女性のどちらの名前でもあります。例えば、Pat。 **F** 女性。この名前は女性の名前です。 **M** 男性。この名前は男性の名前です。 **U** 不明。この名前は、性別テーブルでは見つからない可能性が高いものです。

フィールド名	書式	説明 / 有効な値
GenderDeterminationSource	String	名前の性別判定に使われるカルチャー。この名前が性別テーブルで見つからなければ、このフィールドは空白です。
GeneralSuffix	String	個人名の一般/職業接尾語。例えば、 MD PhD 。
LastName	String	個人名のラスト ネーム。
MaturitySuffix	String	個人の世代/家族接尾語。例えば、Jr.または Sr.。
MiddleName	String	個人のミドル ネーム。
NameScore	String	パーシング処理の品質を表す 0 ～ 100 のスコア。0 は低品質、100 は高品質を表します。
ParserRecordID	String	各入力レコードの識別番号。
TitleOfRespect	String	個人の肩書き Mr.、Mrs.、Dr.、Rev.など。
結合名関係のフィールド		
PersonalName.2.FirstName	String	結合名の 2 番目の人物のファースト ネーム。結合名は、例えば、"John and Jane Smith"。
PersonalName.2.FirstNameVariantGroup	String	結合名の 2 番目の人物のファースト ネームと類似の名前のグループを示す識別番号。例えば、Muhammad、Mohammed、Mehmet はいずれも同じ名前派生グループに属します。実際のグループ ID はアドオン データの読み込み時に設定されます。 このフィールドには Name Variant Group 機能を実際に購入するまで値は設定されません。

フィールド名	書式	説明 / 有効な値
PersonalName.2.GenderCode	String	結合名の 2 番目の人物の性別は、Name Parser によるファースト ネームの分析で決定されます。結合名は、例えば、"John and Jane Smith"。次のいずれか: **A** 曖昧。この名前は男性と女性のどちらの名前でもあります。例えば、Pat。 **F** 女性。この名前は女性の名前です。 **M** 男性。この名前は男性の名前です。 **U** 不明。この名前は、性別テーブルでは見つからない可能性が高いものです。
PersonalName.2.GenderDeterminationSource	String	結合名の 2 番目の人物の性別を判定するカルチャー。結合名は、例えば、"John and Jane Smith"。
PersonalName.2.GeneralSuffix	String	結合名の 2 番目の人物の一般/職業接尾語。結合名は、例えば、"John and Jane Smith"。一般接尾語は、例えば、MD、PhD。
PersonalName.2.LastName	String	結合名の 2 番目の人物のラスト ネーム。結合名は、例えば、"John and Jane Smith"。
PersonalName.2.MaturitySuffix	String	結合名の 2 番目の人物の世代/家族接尾語。結合名は、例えば、"John and Jane Smith"。世代接尾語は、例えば、Jr。と Sr.。
PersonalName.2.MiddleName	String	結合名の 2 番目の人物のミドル ネーム。結合名は、例えば、"John and Jane Smith"。
PersonalName.2.TitleOfRespect	String	結合名の 2 番目の名前に対応する肩書き。例えば、"Mr.and Mrs.Smith" 結合名。肩書きの対応の例は、"Mr., Mrs., and Dr."。
PersonalName.3.FirstName	String	結合名の 3 番目の人物のファーストネーム。例えば、"Mr.& Mrs.John Smith & Dr.Mary Jones" は結合名。

フィールド名	書式	説明 / 有効な値
PersonalName.3.FirstNameVariantGroup	String	結合名の2番目の人物のファースト ネームと類似の名前のグループを示す識別番号。例えば、Muhammad、Mohammed、Mehmetはいずれも同じ名前派生グループに属します。実際のグループ ID はアドオン データの読み込み時に設定されます。 このフィールドには Name Variant Group 機能を実際に購入するまで値は設定されません。
PersonalName.3.GenderCode	String	結合名の 3 番目の人物の性別は、Name Parser のファースト ネームによる性判定で決定されます。結合名の例は "Mr.& Mrs.John Smith & Adam Jones"。次のいずれかです。 **A** 曖昧。この名前は男性と女性のどちらの名前でもあります。例えば、Pat。 **F** 女性。この名前は女性の名前です。 **M** 男性。この名前は男性の名前です。 **U** 不明。この名前は、性別テーブルでは見つからない可能性が高いものです。
PersonalName.3.GenderDeterminationSource	String	結合名の 3 番目の人物の性別判定に使われるカルチャー。"Mr.& Mrs.John Smith & Adam Jones"。
PersonalName.3.GeneralSuffix	String	結合名の3番目の人物の一般/職業接尾語。結合名の例は "Mr.& Mrs.John Smith & Adam Jones PhD." Examples of general suffixes are MD and PhD.
PersonalName.3.LastName	String	結合名の3番目の人物のラスト ネーム。例えば、"Mr.& Mrs.John Smith & Dr.Mary Jones" は結合名。
PersonalName.3.MaturitySuffix	String	結合名の3番目の人物の世代/家族接尾語。結合名の例は "Mr.& Mrs.John Smith & Adam Jones Sr."。世代接尾語の例は、Jr.と Sr.。
PersonalName.3.MiddleName	String	結合名の3番目の人物のミドル ネーム。例えば、"Mr.& Mrs.John Smith & Dr.Mary Jones" は結合名。

フィールド名	書式	説明 / 有効な値
PersonalName.3.TitleOfRespect	String	結合名の 3 番目の人物の肩書き。例えば、"Mr.& Mrs.John Smith & Dr.Mary Jones" は結合名。肩書きの対応の例は、"Mr., Mrs., and Dr."。

Name Variant Finder

Name Variant Finder は、名モードまたは姓モードで動作し、データベースに問い合わせて名前の派生形を取得します。例えば、"John" と "Jon" は名前 "Johnathan" の派生形です。Name Variant Finder の実行にはアドオン辞書が必要です。アドオン辞書は、Universal Name モジュール、Data Normalization モジュール、または Advanced Matching モジュールのデータベース ロード ユーティリティを使ってインストールできます。オプションであるこれらのカルチャー固有の辞書の入手方法については、営業担当者にお問い合わせください。

入力

表 43 : Name Variant Finder の入力フィールド

フィールド名	説明 / 有効な値
FirstName	名前が姓名の名の場合、派生形を見つけるために使う名前。
LastName	名前が姓の場合、派生形を見つけるために使う名前。
GenderCode	FirstName フィールドの名前の性別。次のいずれかです。 **注 :** 性別コードは、ラスト ネームではなく、ファースト ネームにのみ適用されます。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。

フィールド名	説明 / 有効な値
Ethnicity	カルチャーは通常 FirstName または LastName のフィールドの名前と関係します。名前のエスニシティがわかっていなくても、Name Parser または Open Parser ステージでこのフィールドに値を設定できます。 **注：** このフィールドは、以前は GenderDeterminationSource と呼ばれていました。

オプション

表 44 : Name Variant Finder のオプション

オプション	説明
名	ファースト ネームで派生形を探します。
姓	ラスト ネームで派生形を探します。
性別コード	性別がレコードの GenderCode フィールドに指定されている場合にのみ名前の派生形を返します。GenderCode フィールドについては、入力（329ページ）を参照してください。
民族性	カルチャーがレコードの Ethnicity フィールドに指定されている場合にのみ名前の派生形を返します。Ethnicity フィールドについては、入力（329ページ）を参照してください。
ローマ字表記	名前をローマ字で返します。ローマ字の名前とは、非ラテンの用字系をラテンの用字系に書き換えたものです。例えば、韓国名아진をローマ字で書くと Achin になります。
ネイティブ	名前を、そのカルチャーの本来の用字系で書いたものを返します。例えば、韓国名はハングルで返されます。

オプション	説明
かな	**[ネイティブ]** を選択した場合、このオプションを選択して日本語の名前をかなで返すように設定できます。かなは、平仮名と片仮名の用字系で構成されます。 **注 :** 日本語の名前の派生形を検索するには、Asian Plus Pack データベースのライセンスを取得する必要があります。詳細については、セールスエグゼクティブに問い合わせてください。
漢字	**[ネイティブ]** を選択した場合、このオプションを選択して日本語の名前を漢字で返すように設定できます。漢字は、日本語で使われる用字系の 1 つです。 **注 :** 日本語の名前の派生形を検索するには、Asian Plus Pack データベースのライセンスを取得する必要があります。詳細については、セールスエグゼクティブに問い合わせてください。

出力

表 45 : Name Variant Finder の出力

フィールド名	書式	説明 / 有効な値
CandidateGroup	String	入力名とその名前の派生形のグループを表します。各入力名には、CandidateGroup 番号が与えられます。その名前の派生形にも同じ CandidateGroup 番号が与えられます。
Ethnicity	String	Core Names およびアドオン辞書によって決定される名前のカルチャー。 **注 :** このフィールドは、以前は GenderDeterminationSource と呼ばれていました。
FirstName	String	個人の名。

フィールド名	書式	説明 / 有効な値
GenderCode	String	Core Names およびアドオン辞書によって決定される名前の性別。次のいずれかです。 **M** この名前は男性の名前です。 **F** この名前は女性の名前です。 **A** 曖昧。この名前は男性でも女性でもあります。 **U** 不明。この名前の性別はわかりません。
LastName	String	個人名のラスト ネーム。
TransactionalRecordType	String	名前がマッチング処理でどう使われたかを示します。次のいずれかです。 **Suspect** サスペクト レコードは、クエリへの入力として使われます。 **Candidate** 候補レコードは、クエリから結果として返されます。

Open Name Parser

OpenNameParser は、名前データ フィールドにある個人名、企業名、またはその他の名称を構成要素に分解します。パースされたこれらの名前要素は、名前のマッチング、名前の正規化、複数レコード名の統合など、他の自動化処理に使用できます。

OpenNameParser は、次の処理を行います。

- 名前が担う機能を示すために、その名前のタイプを特定します。名前エンティティ タイプは、個人名と企業名の 2 つのグループに分かれます。それぞれのグループには、さらに複数のサブグループがあります。
- パーシングに使う構文を把握するために、名前の形式を特定します。個人名は、通常、自然な (署名) 順序または逆の順序に従います。企業名は、通常、階層型の順序に従います。
- 名前を構成する各要素が名前全体に占める構文上の関連性を識別するために、要素を特定してラベル付けします。個人名の構文は、敬称、名、ミドルネーム、姓、接尾語、アカウントを示す用語、その他の個人名要素で構成されます。企業名の構文は、企業名や接尾語などで構成されます。

- 結合された個人名と企業名をパースし、それらを 1 つのレコードとして残すか、複数のレコードに分割します。例えば、結合された名前は、 "Mr.and Mrs.John Smith" や "Baltimore Gas & Electric dba Constellation Energy"です。
- 出力をレコードまたはリストとしてパースします。
- Open Parser ドメインエディタを使用して、Open Name Parser 詳細オプションで使用できる新しいドメインを作成できます。
- パーシングによる訂正の信頼度を示すパーシング スコアを割り当てます。

入力

表 46 : Open Name Parser の入力

フィールド名	説明
CultureCode	入力された名前データのカルチャー。オプションは次のとおりです。 **Null (empty)** グローバル カルチャー (デフォルト)。 **de** ドイツ語。 **es** スペイン語。 **ja** 日本語。 **注 :** Open Parser ドメインエディタを使用して独自のドメインを追加した場合、そのドメインのカルチャーとカルチャー コードも有効になります。
Name	パースしたい名前。このフィールドは必須です。

オプション

Open Name Parser のオプションは、Spectrum™ Technology Platform の任意のクライアントによってステージ レベルで設定するか、データフロー オプションを使用して実行時に設定できます。

パーシング オプション

次の表に、名前のパーシングを制御するオプションを示します。

表 47 : Open Name Parser パーシング オプション

オプション名	説明
個人名をパース	個人名をパースするかどうかを指定します。 **正順序** 名前フィールドの順序は、[敬称]、[名]、[ミドル ネーム]、[姓]、[接尾語] となります。 **逆順序** 姓が最初の順序になります。 **両方** 正順序と逆順序を組み合わせた順序になります。
結合名	結合名をパースするかどうかを指定します。
結合名を複数のレコードに分割	`Bill & Sally Smith` など、複数の人物を含む結合名を複数のレコードに分割するかどうかを指定します。 Unique ID Generator ステージを使用して、各分割レコードの ID を作成します。
企業名をパース	企業名をパースするかどうかを指定します。
結果をリストに出力	パース済み名前要素をリスト形式で返すかどうかを指定します。
しきい値の設定	パフォーマンスと品質のバランスをとる方法を指定します。パフォーマンスを上げると、品質出力が下がります。同様に、品質を上げると、パフォーマンスが下がります。このしきい値を満たすと、レコードに対して他の処理は実行されません。 デフォルト値は 100 です。

カルチャー オプション

次の表に、名前カルチャーを制御するオプションを示します。

表 48 : Open Name Parser カルチャー オプション

オプション名	説明
カルチャー	パーシング グラマーに含めるカルチャーを指定します。デフォルトでは、グローバル カルチャーが選択されます。 **注 :** Open Parser ドメインエディタを使用して独自のドメインを追加した場合、そのドメインのカルチャーとカルチャー コードもここに表示されます。 **[上へ]** ボタンと **[下へ]** ボタンをクリックして、カルチャーを実行する順序を設定します。

詳細設定オプション

次の表に、名前パーシング用の詳細オプションを示します。

表 49 : Open Name Parser の詳細オプション

オプション	説明
詳細オプション	[ドメイン] ドロップダウンを使用して、各名前の適切なドメインを選択します。 **[上へ]** ボタンと **[下へ]** ボタンをクリックして、パーサーを実行する順序を設定します。**[ショートカットしきい値]** フィールドに設定された数字よりもスコアの高い最初のドメインに対して結果が返されます。そのしきい値に達しているドメインがない場合は、スコアの最も高いドメインに対する結果が返されます。複数のドメインが同時にしきい値に達している場合は、最初に実行された (ここで設定された順序によって決まる) ドメインが優先され、その結果が返されます。 **注 :** Open Parser ドメインエディタを使用して独自のドメインを追加した場合は、そのドメインもここに表示されます。

実行時におけるオプションの設定

Open Name Parser のオプションは、データフロー オプションとしてエクスポーズされている場合は実行時に設定して引き渡すことができます。これにより、既存の設定をJSON形式の名前パーシング文字列でオーバーライドできます。また、プロセス フローまたは Job Executor コマンドライン ツールからジョブを呼び出すときに、ステージ オプションを設定することもできます。

Open Name Parser のオプションを実行時に定義するには

1. Enterprise Designer で、Open Name Parser ステージを使用するデータフローを開きます。
2. そのデータフローを保存してエクスポーズします。
3. 編集 > データフローオプション に移動します。
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]** テーブルで、Open Name Parser を展開し、必要に応じてオプションを編集します。編集するオプションのチェック ボックスをオンにしてから、**[デフォルト値]** ドロップダウンの値を変更します。
5. オプション: **[オプション ラベル]** フィールドで、オプションの名前を変更します。
6. **[OK]** を 2 回クリックします。

出力

表 50 : Open Name Parser の出力

フィールド名	書式	説明
AccountDescription	String	名前の一部であるアカウント説明。例えば、"Mary Jones Account # 12345" で、アカウント説明は "Account#12345"。
Names	String	パース済み要素のリストを含む階層フィールド。このフィールドは、[パーシング オプション] の **[結果をリストに出力]** ボックスをチェックしている場合に返されます。
会社名関係のフィールド		
FirmConjunction	String	"d/b/a" (doing business as)、"o/a" (operating as)、"t/a" (trading as) などの略語を含む企業の名前を示します。

フィールド名	書式	説明
FirmName	String	会社名。例えば、"Pitney Bowes"。
FirmSuffix	String	会社名の接尾語。例えば、"Co."、"Inc."
IsFirm	String	名前が、個人名ではなく、企業名であることを示します。
個人名に関係するフィールド		
Conjunction	String	名前に、"and"、"or"、"&" などの接続詞が含まれることを示します。
CultureCode	String	入力データに含まれるカルチャー コード。
CultureCodeUsedToParse	String	データのパースに使用されたカルチャー固有のグラマーを特定します。 **Null (empty)** グローバル カルチャー (デフォルト)。 **de** ドイツ語。 **es** スペイン語。 **ja** 日本語。 **注：** Open Parser ドメインエディタを使用して独自のドメインを追加した場合、そのドメインのカルチャーとカルチャーコードもこのフィールドに表示されます。
FirstName	String	個人のファースト ネーム。
GeneralSuffix	String	個人名の一般/職業接尾語。例えば、 MD PhD 。
IsParsed	String	出力レコードがパースされたかどうかを示します。値は True または False です。

フィールド名	書式	説明
IsPersonal	String	名前が企業名ではなく、個人名であるかどうかを示します。値は True または False です。
IsReverseOrder	String	入力名が逆順序であるかどうかを示します。値は True または False です。
LastName	String	個人名のラスト ネーム。父方の姓が含まれます。
LeadingData	String	名前の前に付けられる、名前以外の情報。
MaturitySuffix	String	個人の世代/家族接尾語。例えば、Jr.または Sr.。
MiddleName	String	個人のミドル ネーム。
Name.	String	入力に指定された個人名または企業名。
NameScore	String	各名前の既知および不明トークンの平均スコアを示します。NameScore の値は、パーシング グラマーでの定義に従って、0 ～ 100 の間になります。マッチが返されない場合は、0 が返されます。
SecondaryLastName	String	スペイン語のパーシング グラマーでは、その人の母の姓。
TitleOfRespect	String	"Mr."、"Mrs."、"Dr." など、名前の前に付けられる情報。
TrailingData	String	名前の後に付けられる、名前以外の情報。
結合名関係のフィールド		

フィールド名	書式	説明
Conjunction2	String	結合されている 2 番目の名前に、"and"、"or"、"&" などの接続詞が含まれることを示します。
Conjunction3	String	結合されている 3 番目の名前に、"and"、"or"、"&" などの接続詞が含まれることを示します。
FirmName2	String	結合されている 2 番目の企業名。例えば、Baltimore Gas & Electric dba Constellation Energy。
FirmSuffix2	String	結合されている 2 番目の企業の接尾語。
FirstName2	String	結合されている FirstName の 2 番目の名
FirstName3	String	結合されている FirstName の 3 番目の名。
GeneralSuffix2	String	結合されている 2 番目の名前の一般/職業接尾語。例えば、 MD PhD 。
GeneralSuffix3	String	結合されている 3 番目の名前の一般/職業接尾語。例えば、 MD PhD 。
IsConjoined	String	入力名が結合名であることを示します。結合名は、例えば、"John and Jane Smith"。
LastName2	String	結合されている LastName の 2 番目の姓。
LastName3	String	結合されている LastName の 3 番目の姓。
MaturitySuffix2	String	結合されている 2 番目の名前の世代/家族接尾語。例えば、Jr.または Sr.。

フィールド名	書式	説明
MaturitySuffix3	String	結合されている 3 番目の名前の世代/家族接尾語。例えば、Jr.または Sr.。
MiddleName2	String	結合されている MiddleName の 2 番目の名。
MiddleName3	String	結合されている MiddleName の 3 番目の名。
TitleOfRespect2	String	"Mr."、"Mrs."、"Dr." など、結合されている 2 番目の名前の前に付けられる情報。
TitleOfRespect3	String	"Mr."、"Mrs."、"Dr." など、結合されている 3 番目の名前の前に付けられる情報。

Open Name Parser サマリ レポート

Open Name Parser サマリ レポートには、入力レコードの合計数や、名前データが含まれないレコードの合計数など、ジョブに関するサマリ統計と、パーシング統計が一覧表示されます。レポートの使用方法については、『*Spectrum™ Technology Platform データフロー デザイナー ガイド*』を参照してください。

一般的な結果

- **入力レコードの合計数** — 入力ファイル内のレコードの数。
- **名前データが含まれないレコードの合計数** — 入力ファイル内のレコードのうち、パースする名前データが含まれていなかったレコードの数。
- **パースされた名前の合計数** — 入力ファイル内の名前のうち、パースされた名前の数。
- **レコードの合計数** — 処理されたレコードの合計数。
- **名前パーシング スコアの最小値** — 入力ファイル内の名前に与えられたパーシング スコアの最小値。
- **名前パーシング スコアの最大値** — 入力ファイル内の名前に与えられたパーシング スコアの最大値。
- **名前パーシング スコアの平均値** — 入力ファイル内のパースされたすべてのレコード間でのパーシング スコアの平均値。

個人名のパーシング結果

- **記述された個人名レコードの数** — 入力ファイル内の個人名の数。
- **結合名からパースされた名前の数** — 結合名が含まるレコードからパースされた名前の数。例えば、入力ファイルに、2 つの結合名を持つレコードが 5 つ、3 つの結合名を持つレコードが 7 つある場合、このフィールドの値は、(5 x 2) + (7 x 3) = 31 になります。
- **結合名を 2 つ持つレコード** — 結合名を 2 つ含む入力レコードの数。
- **結合名を 3 つ持つレコード**— 結合名を 3 つ含む入力レコードの数。
- **敬称を持つ名前の数** — パースされた名前のうち、敬称を含む名前の数。
- **世代接尾語を持つ名前の数** — パースされた名前のうち、世代接尾語を含む名前の数。
- **一般接尾語を持つ名前の数** — パースされた名前のうち、一般接尾語を含む名前の数。
- **アカウント説明を含む名前の数** — パースされた名前のうち、アカウント説明を含む名前の数。
- **逆順序の名前の合計数**—パースされた名前のうち、逆順序の名前の数 (出力フィールド isReversed が "True")。

企業名のパーシング結果

- **記述されたビジネス名レコードの数** — 入力ファイル内の企業名の数。
- **企業接尾語を持つ名前の数** — パースされた名前のうち、企業接尾語を含む名前の数。
- **アカウント説明を含む名前の数** — アカウント説明を含む入力レコードの数。
- **DBA レコードの合計数** — 略語 Doing Business As (DBA) を含み、出力フィールド isPersonal と isFirm の両方が "True" となる入力レコードの数。

9 - ISO 国コードとモジュール サポート

このセクションの構成

ISO 国コードとモジュール サポート

この表に、各国の ISO コードと、各国の住所作成、ジオコーディング、およびルーティングをサポートするモジュールを示します。

Enterprise Geocoding モジュールにアフリカ (30 か国)、中東 (8 か国)、ラテン アメリカ (20 か国) のデータベースが含まれていることに注意してください。これらのデータベースは、国別のジオコーディング データベースがない、各地域の比較的小さな国をカバーします。[サポートされるモジュール] 列は、これらのアフリカ、中東、ラテン アメリカ データベースに含まれる国を示しています。

また、Geocode Address World データベースは、すべての国について地図上の限定的な郵便ジオコーディング (ストリート レベルではない) を提供します。

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Afghanistan	AF	AFG	Universal Addressing モジュール
Aland Islands	AX	ALA	Universal Addressing モジュール
Albania	AL または SQ (Routing)	ALB	Universal Addressing モジュール Enterprise Geocoding モジュール Enterprise Routing モジュール
Algeria	DZ	DZA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
American Samoa	AS	ASM	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Andorra	AD	AND	Enterprise Geocoding モジュール(アンドラは、スペインのジオコーダで扱われています)。 Universal Addressing モジュール GeoComplete モジュール
Angola	AO	AGO	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Anguilla	AI	AIA	Universal Addressing モジュール
Antarctica	AQ	ATA	Universal Addressing モジュール
Antigua And Barbuda	AG	ATG	Universal Addressing モジュール
Argentina	AR	ARG	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール
Armenia	AM	ARM	Universal Addressing モジュール
Aruba	AW	ABW	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Australia	AU	AUS	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール
Austria	AT	AUT	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Azerbaijan	AZ	AZE	Universal Addressing モジュール
Bahamas	BS	BHS	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール
Bahrain	BH	BHR	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール
Bangladesh	BD	BGD	Universal Addressing モジュール
Barbados	BB	BRB	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Belarus	BY	BLR	Universal Addressing モジュール Enterprise Routing モジュール
Belgium	BE	BEL	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Belize	BZ	BLZ	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Benin	BJ	BEN	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Bermuda	BM	BMU	Universal Addressing モジュール Enterprise Routing モジュール
Bhutan	BT	BTN	Universal Addressing モジュール
Bolivia, Plurinational State Of	BO	BOL	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Bonaire, Saint Eustatius And Saba	BQ	BES	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Bosnia And Herzegovina	BA	BIH	Universal Addressing モジュール Enterprise Routing モジュール Enterprise Geocoding モジュール
Botswana	BW	BWA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Bouvet Island	BV	BVT	Universal Addressing モジュール
Brazil	BR	BRA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
British Indian Ocean Territory	IO	IOT	Universal Addressing モジュール
Brunei Darussalam	BN	BRN	Enterprise Geocoding モジュール Universal Addressing モジュール
Bulgaria	BG	BGR	Universal Addressing モジュール
Burkina Faso	BF	BFA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Burundi	BI	BDI	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Cambodia	KH	KHM	Universal Addressing モジュール
Cameroon	CM	CMR	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Canada	CA	CAN	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Cape Verde	CV	CPV	Universal Addressing モジュール
Cayman Islands	KY	CYM	Universal Addressing モジュール
Central African Republic	CF	CAF	Universal Addressing モジュール
Chad	TD	TCD	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Chile	CL	CHL	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール GeoComplete モジュール
China	CN または zh_CN (Routing)	CHN	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール
Christmas Island	CX	CXR	Universal Addressing モジュール
Cocos (Keeling) Islands	CC	CCK	Universal Addressing モジュール
Colombia	CO	COL	Universal Addressing モジュール
Comoros	KM	COM	Universal Addressing モジュール
Congo	CG	COG	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Congo, The Democratic Republic Of The	CD	COD	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Cook Islands	CK	COK	Universal Addressing モジュール
Costa Rica	CR	CRI	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Côte d'Ivoire	CI	CIV	Universal Addressing モジュール
Croatia	HR	HRV	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール
Cuba	CU	CUB	Enterprise Geocoding モジュール (ラテン アメリカ) Enterprise Routing モジュール Universal Addressing モジュール
Curacao	CW	CUW	Universal Addressing モジュール
Cyprus	CY	CYP	Enterprise Geocoding モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Czech Republic	CZ または CS (Routing)	CZE	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール GeoComplete モジュール
Denmark	DK	DNK	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Djibouti	DJ	DJI	Universal Addressing モジュール
Dominica	DM	DMA	Universal Addressing モジュール
Dominican Republic	DO	DOM	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Ecuador	EC	ECU	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Egypt	EG	EGY	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
El Salvador	SV	SLV	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Equatorial Guinea	GQ	GNQ	Universal Addressing モジュール
Eritrea	ER	ERI	Universal Addressing モジュール
Estonia	EE	EST	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール
Ethiopia	ET	ETH	Universal Addressing モジュール
Falkland Islands (Malvinas)	FK	FLK	Universal Addressing モジュール
Faroe Islands	FO	FRO	Universal Addressing モジュール
Fiji	FJ	FJI	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Finland	FI	FIN	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
France	FR	FRA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
French Guiana	GF	GUF	Enterprise Geocoding モジュール (*フランス領ギアナは、フランスのジオコーダで扱われています*)。 Universal Addressing モジュール
French Polynesia	PF	PYF	Universal Addressing モジュール
French Southern Territories	TF	ATF	Universal Addressing モジュール
Gabon	GA	GAB	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Gambia	GM	GMB	Universal Addressing モジュール
Georgia	GE	GEO	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Germany	DE	DEU	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Ghana	GH	GHA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール
Gibraltar	GI	GIB	Enterprise Geocoding モジュール (*ジブラルタルは、スペインのジオコーダで扱われています*)。 Universal Addressing モジュール
Greece	GR	GRC	Enterprise Geocoding モジュール Universal Addressing モジュール
Greenland	GL	GRL	Universal Addressing モジュール
Grenada	GD	GRD	Universal Addressing モジュール
Guadeloupe	GP	GLP	Enterprise Geocoding モジュール (*グアドループは、フランスのジオコーダで扱われています*)。 Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Guam	GU	GUM	Universal Addressing モジュール
Guatemala	GT	GTM	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Guernsey	GG	GGY	Universal Addressing モジュール
Guinea	GN	GIN	Universal Addressing モジュール
Guinea-Bissau	GW	GNB	Universal Addressing モジュール
Guyana	GY	GUY	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Haiti	HT	HTI	Universal Addressing モジュール
Heard Island and McDonald Islands	HM	HMD	Universal Addressing モジュール
Holy See (Vatican City State)	VA	VAT	Enterprise Geocoding モジュール (バチカンは、イタリアのジオコーダで扱われています)。 Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Honduras	HN	HND	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Hong Kong	HK	HKG	Enterprise Geocoding モジュール Universal Addressing モジュール
Hungary	HU	HUN	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Iceland	IS	ISL	Enterprise Geocoding モジュール Universal Addressing モジュール
India	IN	IND	Enterprise Geocoding モジュール Universal Addressing モジュール
Indonesia	ID	IDN	Enterprise Geocoding モジュール Universal Addressing モジュール
Iran, Islamic Republic Of	IR	IRN	Universal Addressing モジュール
Iraq	IQ	IRQ	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Ireland	IE	IRL	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Isle Of Man	IM	IMN	Universal Addressing モジュール
Israel	IL	ISR	Universal Addressing モジュール Enterprise Routing モジュール
Italy	IT	ITA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Jamaica	JM	JAM	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Japan	JP	JPN	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Jersey	JE	JEY	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Jordan	JO	JOR	Universal Addressing モジュール Enterprise Geocoding モジュール (中東) Enterprise Routing モジュール
Kazakhstan	KZ	KAZ	Universal Addressing モジュール
Kenya	KE	KEN	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール
Kiribati	KI	KIR	Universal Addressing モジュール
Korea, Democratic People's Republic Of	KP	PRK	Universal Addressing モジュール
Korea, Republic Of	KR	KOR	Universal Addressing モジュール
Kosovo	KS	KOS	Enterprise Geocoding モジュール Universal Addressing モジュール GeoComplete モジュール
Kuwait	KW	KWT	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Kyrgyzstan	KG	KGZ	Universal Addressing モジュール
Lao People's Democratic Republic	LA	LAO	Universal Addressing モジュール
Latvia	LV	LVA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール
Lebanon	LB	LBN	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール
Lesotho	LS	LSO	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール
Liberia	LR	LBR	Universal Addressing モジュール
Libyan Arab Jamahiriya	LY	LBY	Universal Addressing モジュール
Liechtenstein	LI	LIE	Enterprise Geocoding モジュール (リヒテンシュタインは、スイスのジオコーダで扱われています)。 Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Lithuania	LT	LTU	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール
Luxembourg	LU	LUX	Enterprise Geocoding モジュール (*ルクセンブルクは、ベルギーのジオコーダで扱われています*)。 Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Macao	MO	MAC	Enterprise Geocoding モジュール Universal Addressing モジュール
Macedonia, Former Yugoslav Republic Of	MK	MKD	Enterprise Geocoding モジュール Universal Addressing モジュール
Madagascar	MG	MDG	Universal Addressing モジュール
Malawi	MW	MWI	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Malaysia	MY	MYS	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Maldives	MV	MDV	Universal Addressing モジュール
Mali	ML	MLI	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Malta	ML	MLT	Enterprise Geocoding モジュール Universal Addressing モジュール
Marshall Islands	MH	MHL	Universal Addressing モジュール
Martinique	MQ	MTQ	Enterprise Geocoding モジュール (マルティニークは、フランスのジオコーダで扱われています)。 Universal Addressing モジュール
Mauritania	MR	MRT	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Mauritius	MU	MUS	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Mayotte	YT	MYT	Enterprise Geocoding モジュール (マヨットは、フランスのジオコーダで扱われています)。 Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Mexico	MX	MEX	Enterprise Geocoding モジュール Universal Addressing モジュール
Micronesia, Federated States Of	FM	FSM	Universal Addressing モジュール
Moldova, Republic Of	MD	MDA	Universal Addressing モジュール Enterprise Routing モジュール
Monaco	MC	MCO	Enterprise Geocoding モジュール (*モナコはフランスのジオコーダで扱われています*)。 Universal Addressing モジュール
Mongolia	MN	MNG	Universal Addressing モジュール
Montenegro	ME	MNE	Enterprise Geocoding モジュール Universal Addressing モジュール
Montserrat	MS	MSR	Universal Addressing モジュール
Morocco	MA	MAR	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Mozambique	MZ	MOZ	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール
Myanmar	MM	MMR	Universal Addressing モジュール
Namibia	NA	NAM	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Nauru	NR	NRU	Universal Addressing モジュール
Nepal	NP	NPL	Universal Addressing モジュール
Netherlands	NL	NLD	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
New Caledonia	NC	NCL	Universal Addressing モジュール
New Zealand	NZ	NZL	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Nicaragua	NI	NIC	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Niger	NE	NER	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Nigeria	NG	NGA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Niue	NU	NIU	Universal Addressing モジュール
Norfolk Island	NF	NFK	Universal Addressing モジュール
Northern Mariana Islands	MP	MNP	Universal Addressing モジュール
Norway	NO	NOR	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Oman	OM	OMN	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Pakistan	PK	PAK	Universal Addressing モジュール
Palau	PW	PLW	Universal Addressing モジュール
Palestinian Territory, Occupied	PS	PSE	Universal Addressing モジュール
Panama	PA	PAN	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Papua New Guinea	PG	PNG	Universal Addressing モジュール
Paraguay	PY	PRY	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Peru	PE	PER	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Philippines	PH	PHL	Enterprise Geocoding モジュール Universal Addressing モジュール Enterprise Routing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Pitcairn	PN	PCN	Universal Addressing モジュール
Poland	PL	POL	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Portugal	PT	PRT	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Puerto Rico	PR	PRI	Universal Addressing モジュール
Qatar	QA	QAT	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール
Reunion	RE	REU	Enterprise Geocoding モジュール (*レユニオンはフランスのジオコーダで扱われています*)。 Universal Addressing モジュール
Romania	RO	ROU	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Russian Federation	RU	RUS	Enterprise Geocoding モジュール Universal Addressing モジュール GeoComplete モジュール
Rwanda	RW	RWA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Saint Barthelemy	BL	BLM	Universal Addressing モジュール
Saint Helena, Ascension and Tristan Da Cunha	SH	SHE	Universal Addressing モジュール
Saint Kitts and Nevis	KN	KNA	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Saint Lucia	LC	LCA	Universal Addressing モジュール
Saint Martin (French Part)	MF	MAF	Universal Addressing モジュール
Saint Pierre and Miquelon	PM	SPM	Universal Addressing モジュール
Saint Vincent and the Grenadines	VC	VCT	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Samoa	WS	WSM	Universal Addressing モジュール
San Marino	SM	SMR	Enterprise Geocoding モジュール (サンマリノは、イタリアのジオコーダで扱われています)。 Universal Addressing モジュール
Sao Tome and Principe	ST	STP	Universal Addressing モジュール
Saudi Arabia	SA	SAU	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール
Senegal	SN	SEN	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Serbia	RS	SRB	Enterprise Geocoding モジュール Universal Addressing モジュール
Seychelles	SC	SYC	Universal Addressing モジュール
Sierra Leone	SL	SLE	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Singapore	SG	SGP	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Sint Maarten (Dutch Part)	SX	SXM	Universal Addressing モジュール
Slovakia	SK	SVK	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Slovenia	SI	SVN	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Solomon Islands	SB	SLB	Universal Addressing モジュール
Somalia	SO	SOM	Universal Addressing モジュール
South Africa	ZA	ZAF	Enterprise Geocoding モジュール Universal Addressing モジュール GeoComplete モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
South Georgia And The South Sandwich Islands	GS	SGS	Enterprise Geocoding モジュール Universal Addressing モジュール
South Sudan	SS	SSD	Universal Addressing モジュール
Spain	ES	ESP	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Sri Lanka	LK	LKA	Universal Addressing モジュール
Sudan	SD	SDN	Universal Addressing モジュール
Suriname	SR	SUR	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Svalbard And Jan Mayen	SJ	SJM	Universal Addressing モジュール
Swaziland	SZ	SWZ	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Sweden	SE	SWE	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Switzerland	CH	CHE	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
Syrian Arab Republic	SY	SYR	Universal Addressing モジュール
Taiwan, Province of China	TW または zh_TW (Routing)	TWN	Universal Addressing モジュール Enterprise Routing モジュール
Tajikistan	TJ	TJK	Universal Addressing モジュール
Tanzania, United Republic Of	TZ	TZA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール Enterprise Routing モジュール
Thailand	TH	THA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Timor-Leste	TL	TLS	Universal Addressing モジュール
Togo	TG	TGO	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Tokelau	TK	TKL	Universal Addressing モジュール
Tonga	TO	TON	Universal Addressing モジュール
Trinidad and Tobago	TT	TTO	Enterprise Geocoding モジュール (ラテン アメリカ) Universal Addressing モジュール
Tunisia	TN	TUN	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Turkey	TR	TUR	Enterprise Geocoding モジュール Universal Addressing モジュール GeoComplete モジュール
Turkmenistan	TM	TKM	Universal Addressing モジュール
Turks And Caicos Islands	TC	TCA	Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Tuvalu	TV	TUV	Universal Addressing モジュール
Uganda	UG	UGA	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Ukraine	UA	UKR	Enterprise Geocoding モジュール Universal Addressing モジュール
United Arab Emirates	AE	ARE	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール
United Kingdom	GB	GBR	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
United States	US	USA	Enterprise Geocoding モジュール Enterprise Routing モジュール Universal Addressing モジュール GeoComplete モジュール
United States Minor Outlying Islands	UM	UMI	Universal Addressing モジュール
Uruguay	UY	URY	Enterprise Geocoding モジュール Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Uzbekistan	UZ	UZB	Universal Addressing モジュール
Vanuatu	VU	VUT	Universal Addressing モジュール
Venezuela, Bolivarian Republic Of	VE	VEN	Enterprise Geocoding モジュール Universal Addressing モジュール
Viet Nam	VN	VNM	Universal Addressing モジュール
Virgin Islands, British	VG	VGB	Universal Addressing モジュール
Virgin Islands, U.S.	VI	VIR	Universal Addressing モジュール
Wallis and Futuna	WF	WLF	Universal Addressing モジュール
Western Sahara	EH	ESH	Universal Addressing モジュール
Yemen	YE	YEM	Enterprise Geocoding モジュール (中東) Universal Addressing モジュール

ISO 国名	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	サポートされるモジュール
Zambia	ZM	ZMB	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール
Zimbabwe	ZW	ZWE	Enterprise Geocoding モジュール (アフリカ) Universal Addressing モジュール

著作権に関する通知

© 2017 Pitney Bowes Software Inc. 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、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

© 2017 Pitney Bowes Software Inc.
All rights reserved