Informatica® Cloud Data Integration

REST V2 コネクタ

Informatica Cloud Data Integration REST V2 コネクタ
2022 年 10 月

© 著作権 Informatica LLC 2016, 2022

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

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

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

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

特許については、https://www.informatica.com/legal/patents.html を参照してください。

免責: 本文書は、一切の保証を伴わない「現状渡し」で提供されるものとし、Informatica LLC は他社の権利の非侵害、市場性および特定の目的への適合性の黙示の保証などを含めて、一切の明示的および黙示的保証の責任を負いません。Informatica LLC では、本ソフトウェアまたはドキュメントに誤りのないことを保証していません。本ソフトウェアまたはドキュメントに記載されている情報には、技術的に不正確な記述や誤植が含まれる場合があります。本ソフトウェアまたはドキュメントの情報は、予告なしに変更されることがあります。

NOTICES

この Informatica 製品（以下「ソフトウェア」）には、Progress Software Corporation（以下「DataDirect」）の事業子会社である DataDirect Technologies からの特定のドライバ（以下「DataDirect ドライバ」）が含まれています。DataDirect ドライバには、次の用語および条件が適用されます。

1. DataDirect ドライバは、特定物として現存するままの状態で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。
2. DataDirect または第三者は、予見の有無を問わず発生した ODBC ドライバの使用に関するいかなる直接的、間接的、偶発的、特別、あるいは結果的損害に対して責任を負わないものとします。本制限事項は、すべての訴訟原因に適用されます。訴訟原因には、契約違反、保証違反、過失、厳格責任、詐称、その他の不法行為を含みますが、これらに限るものではありません。

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

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

発行日: 2022-12-04

目次

はじめに. 5
Informatica のリソース. 5
Informatica マニュアル. 5
Informatica Intelligent Cloud Services Web サイト. 5
Informatica Intelligent Cloud Services コミュニティ. 5
Informatica Intelligent Cloud Services マーケットプレース. 5
データ統合コネクタのドキュメント. 6
Informatica ナレッジベース. 6
Informatica Intelligent Cloud Services Trust Center. 6
Informatica グローバルカスタマサポート. 6

第 1 章 : REST V2 コネクタについて. 7

REST V2 コネクタアセット. 8
XML オブジェクト. 8
REST V2 コネクタの管理. 10
安全な通信. 10
サーバーレスランタイム環境での安全な通信. 11
サーバーレスランタイム環境の Swagger ファイル. 12
ランタイム環境に関するガイドライン. 12

第 2 章 : REST V2 の接続. 14

REST V2 接続のプロパティ. 14
OAuth 2.0 クライアント資格情報認証. 16
OAuth 2.0 認証コード認証. 18
JWT ベアラートークン認証. 21
REST V2 接続についてのルールおよびガイドライン. 23

第 3 章 : REST V2 操作. 25

REST V2 ソース操作. 25
Request Message Editor を使用した要求の設定. 25
要求メッセージの入力値のパラメータ化. 26
ソーストランスフォーメーションでのフィールドマッピング. 27
パック済みフィールドの作成. 27
REST V2 ミッドストリーム操作. 28
REST V2 ターゲット操作. 29
ターゲットトランスフォーメーションのフィールドマッピング. 30
配列要素を含む要求の設定 30
複数のソースグループを使用した要求の作成. 30
単一のソースグループを使用した要求の作成. 31
REST エンドポイント URL へのファイルのアップロード. 32

応答ヘッダーの解析. 33
セキュリティ定義の解析. 34
REST V2 操作についてのルールおよびガイドライン. 35

第 4 章 : REST V2 コネクタを使用したマッピングとマッピングタスク. . 38
マッピングにおける REST ソーストランスフォーメーション. 38
詳細ソースプロパティ. 38
REST ソースマッピングの例. 41
マッピングにおける REST ミッドストリームトランスフォーメーション. 43
詳細ミッドストリームプロパティ. 44
ミッドストリームトランスフォーメーションマッピングの例. 45
マッピングにおける REST V2 ターゲット. 48
詳細ターゲットプロパティ. 48
入力設定プロパティ. 49
REST ターゲットマッピングの例. 49

付録 A : Swagger オブジェクト. 52

索引. 54

はじめに

*REST V2 コネクタ*を使用すると、データ統合を使って、REST API をサポートする Web サービスに対して読み書きを行う方法を学べます。データ統合で、REST V2 接続を作成する方法や、マッピングとマッピングタスクを開発および実行する方法を学習できます。

Informatica のリソース

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

Informatica マニュアル

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

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

Informatica Intelligent Cloud Services Web サイト

Informatica Intelligent Cloud Services Web サイト（http://www.informatica.com/cloud）にアクセスできます。このサイトには、Informatica Cloud 統合サービスに関する情報が含まれます。

Informatica Intelligent Cloud Services コミュニティ

Informatica Intelligent Cloud Services コミュニティを使用して、技術的な問題について議論し、解決します。また、技術的なヒント、マニュアルの更新情報、FAQ（よくある質問）への答えを得ることもできます。

次の Informatica Intelligent Cloud Services コミュニティにアクセスします。

https://network.informatica.com/community/informatica-network/products/cloud-integration

開発者は、次の Cloud 開発者コミュニティで詳細情報を確認したり、ヒントを共有したりできます。

https://network.informatica.com/community/informatica-network/products/cloud-integration/cloud-developers

Informatica Intelligent Cloud Services マーケットプレース

Informatica マーケットプレイスにアクセスすると、データ統合コネクタ、テンプレート、およびマップレットを試用したり購入したりできます。

https://marketplace.informatica.com/

データ統合コネクタのドキュメント

データ統合コネクタのドキュメントには、マニュアルポータルからアクセスできます。マニュアルポータルを利用するには、https://docs.informatica.com にアクセスしてください。

Informatica ナレッジベース

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

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

Informatica Intelligent Cloud Services Trust Center

Informatica Intelligent Cloud Services Trust Center は、Informatica のセキュリティポリシーおよびリアルタイムでのシステムの可用性について情報を提供します。

Trust Center（https://www.informatica.com/trust-center.html）にアクセスします。

Informatica Intelligent Cloud Services Trust Center にサブスクライブして、アップグレード、メンテナンス、およびインシデントの通知を受信します。Informatica Intelligent Cloud Services Status ページには、すべての Informatica Cloud 製品の実稼働ステータスが表示されます。メンテナンスの更新はすべてこのページに送信され、停止中は最新の情報が表示されます。更新と停止の通知がされるようにするには、Informatica Intelligent Cloud Services の 1 つのコンポーネントまたはすべてのコンポーネントについて更新の受信をサブスクライブします。すべてのコンポーネントにサブスクライブするのが、更新を逃さないようにするための最良の方法です。

登録するには、https://status.informatica.com/に移動し、**[更新を購読登録]** をクリックします。その後、電子メール、SMS テキストメッセージ、Webhook、RSS フィードとして、またはこの 4 つを任意に組み合わせて送信された通知を受信することを選択ができます。

Informatica グローバルカスタマサポート

電話またはオンラインでカスタマサポートセンターに連絡できます。

オンラインサポートについては、Informatica Intelligent Cloud Services の **[サポート要求の送信]** をクリックしてください。またオンラインサポートを使用して問題を記録することもできます。オンラインサポートを利用するには、ログインが必要です。https://network.informatica.com/welcome でログイン要求できます。

Informatica グローバルカスタマサポートの電話番号は、Informatica の Web サイト https://www.informatica.com/services-and-training/support-services/contact-us.html に掲載されています。

第 1 章

REST V2 コネクタについて

REST アーキテクチャで構築されている Web サービスアプリケーションと対話するには、REST V2 コネクタを使用します。REST V2 コネクタは、ソーストランスフォーメーション、ターゲットトランスフォーメーション、または Web サービストランスフォーメーションのミッドストリームで使用できます。

マッピングで REST V2 コネクタミッドストリームを利用して、Web サービスアプリケーションに単一または複数の要求を送信し、応答データを処理できます。マッピングパイプラインで複数のトランスフォーメーションから取得したデータを渡して、処理することもできます。マッピングを詳細モードに切り替えて、高度な機能を有効にするためのトランスフォーメーションと関数を含めることができます。

マッピングで REST V2 コネクタミッドストリームを使用するには、まず、Web サービスアプリケーションで実行する必要がある操作用のビジネスサービスを作成します。次に、マッピングの Web サービストランスフォーメーションミッドストリームでそのビジネスサービスの関連付けを行い、Web サービスアプリケーションからデータを読み取ったり、Web サービスアプリケーションにデータを書き込んだりできるようにします。例えば、REST V2 コネクタをミッドストリームトランスフォーメーションとして使用して、Web サービスアプリケーションで PUT 操作を実行させることができます。ターゲットトランスフォーメーションとミッドストリームトランスフォーメーションでは、フィールド名に特殊文字を使用できます。

ソーストランスフォーメーション、ターゲットトランスフォーメーション、およびミッドストリームトランスフォーメーションにおいて、次の REST メソッドを使用できます。

- GET
- PUT
- POST
- DELETE
- OPTIONS
- HEAD
- PATCH

REST V2 接続を作成できたら、次のいずれかの REST 認証タイプを設定できます。

- 標準
  - 基本
  - OAuth 1.0
- OAuth 2.0 クライアント資格情報
- OAuth 2.0 認証コード
- JWT Bearer トークン

次のメディアタイプを使用してデータを処理できます。

- application/xml

- application/json
- application/x-www-form-urlencoded
- JSON サブタイプ。例: application/**hal+**json
- JSON カスタムタイプ。例: application/**vnd.ds.abc.v1+**json
- 拡張 JSON MIME タイプ。例: application/vnd.ds.abc.v1+json;**version=q1**
- text/xml

Swagger 定義のコンシューム属性値とプロデュース属性値に複数の MIME タイプがある場合、REST V2 コネクタは、ペイロードがリスト内の最初の MIME タイプであると見なします。

REST V2 コネクタアセット

データ統合でアセットを作成し、Rest V2 コネクタを使用してデータを統合できます。

次のデータ統合アセットでは、ソース、ターゲット、ミッドストリームのトランスフォーメーションタイプを使用できます。

- マッピング
- マッピングタスク

マッピングまたはマッピングタスクを実行すると、<Secure Agent のインストールディレクトリ>/apps/Data_Integration_Server/logs ディレクトリにセッションログが保存されます。

アセットとトランスフォーメーションの設定に関する詳細については、データ統合のドキュメントの「*マッピング*」、「*トランスフォーメーション*」、および「*タスク*」を参照してください。

XML オブジェクト

XML 属性、ラッパーオブジェクト、名前空間オブジェクトなどの XML オブジェクトを設定できます。

XML オブジェクトの属性のみ定義できます。Swagger 仕様ファイルで XML 要素の属性は定義できません。配列要素にインライン定義がある場合のみ、単純型配列要素に対してラッパーオブジェクトを使用できます。ソーストランスフォーメーション、ミッドストリームトランスフォーメーション、およびターゲットトランスフォーメーションに対する応答内の修飾されている名前空間オブジェクトと修飾されていない名前空間オブジェクトを使用します。

XML 属性の Swagger 定義:

```
"XMLAttrArray_Request##body##Employee##dep##Mangr" : {
"properties" : {
"desg" : {
"type" : "string",
"xml": { "attribute": true }
}, "mid" : { "type" : "string" },
```

XML 属性の例:

```
<dep>
<Mangr desg="Director">
<mid>em09</mid>
<mname>mg</mname>
</Mangr>
<id>did</id>
<name>dname</name>
</dep>
```

XML 属性は必ず文字列型になります。Swagger 定義に数値型の属性が含まれていても、REST V2 コネクタは常に、文字列型の属性として処理します。生成される XSD では、XML 属性のデータ型は文字列になります。

ラッパーオブジェクトの Swagger 定義:

```
"books" : {
"type" : "array",
"items" : { "type" : "string" },
"xml": {
"wrapped" : "true", "name": "books-array"
}
}
```

ラッパーオブジェクトの例:

```
<books-array>
<books>1</books>
<books>2</books>
</books-array>
```

名前空間オブジェクトの Swagger 定義:

```
root" : {
"properties" : {
"table" : {
"$ref" : "#/definitions/NSResReq_Request##body##root##table"
}
},
"xml":{
"prefix": "h",
"namespace": "http://www.w3.org/TR/html4/"
}
}
```

名前空間オブジェクトの例:

```
<h:root xmlns:h="http://www.w3.org/TR/html4/">
<table>
<tr>
<td>Apples</td>
<td>Bananas</td>
</tr>
</table>
</h:root>
```

注: Swagger 定義に複数の名前空間が含まれていないことを確認してください。

REST V2 コネクタの管理

組織の管理者が次のタスクを実行すると、ユーザーは REST V2 コネクタを使用できます。

- 64 ビットマシンに Secure Agent をインストールする。
- Secure Agent をホスティングしているマシンのメモリサイズが、2048 MB 以上であることを確認する。
- 最新バージョンの XToolkit API を Secure Agent ディレクトリにデプロイする。
- Data Transformation パッケージを Secure Agent マシンにインストールする。
- UDTforHierarchy パッケージを Secure Agent マシンにインストールする。
- Secure Agent マシンで、Data Transformation パッケージ、UDTforHierarchy パッケージ、saas-xmetadataread パッケージにライセンスを提供する。

安全な通信

TLS 認証を設定すると、REST API との安全な一方向または双方向通信を確立できます。

Secure Agent は、TLS 経由で REST API との安全な通信を確立します。一方向の SSL でも双方向の SSL でも使用できます。証明書を設定する前に、証明書が.jks 形式であることを確認します。

一方向の SSL の使用

一方向の SSL を使用するには、次の手順を実行します。

- サーバー証明書を<Secure Agent installation directory>\jre\lib\security\cacerts ファイルにインポートします。次のコマンドを使用します。
  keytool -importkeystore -srckeystore *<PathtoCert>*\clientSSL.p12 -srcstoretype *<PKCS12 証明書のタイプ>* -destkeystore <Informatica agent Installation location\jdk\jre\lib\security\cacerts -deststoretype JKS
- トラストストアのファイル名とトラストストアのパスワードの JVM オプションを追加します。
  - 管理者>［ランタイム環境］をクリックして、エージェントを選択します。
  - **［システム構成の詳細］** の［タイプ］で［DTM］を選択します。
  - 次の JVM オプションを追加します。
    - JVMOption1=-Djavax.net.ssl.trustStore=*<..jks トラストストアファイルの絶対パス>*

- JVMOption2=-Djavax.net.ssl.trustStorePassword=<*トラストストアのパスワード*>

接続プロパティ［**トラストストアファイル名**］および［**トラストストアのパスワード**］にトラストファイルの名前とトラストストアのパスワードを指定することもできます。

双方向の SSL の使用

双方向の SSL を使用するには、最初に一方向 SSL の手順を実行する必要があります。それから、次の手順を実行します。

- キーストアのファイルとキーストアのパスワードの JVM オプションを追加します。
  - 管理者>［ランタイム環境］をクリックして、エージェントを選択します。
  - ［**システム構成の詳細**］の［タイプ］で［DTM］を選択します。
  - 次の JVM オプションを設定します。
    - JVMOption3=-Djavax.net.ssl.keyStore=<*.jks キーストアファイルの絶対パス*>
    - JVMOption4=-Djavax.net.ssl.keyStorePassword=<*キーストアのパスワード*>

  接続プロパティ［**キーストアファイル名**］および［**キーストアのパスワード**］にキーファイルの名前とキーストアのパスワードを指定することもできます。

Secure Agent は、次の順序で証明書を処理します。

キーストア

1. 接続属性
2. JVM プロパティ

トラストストア

1. 接続属性
2. JVM プロパティ
3. <Secure Agent のインストールディレクトリ>\jdk\jre\lib\security\cacerts にインポートされた証明書

サーバーレスランタイム環境での安全な通信

サーバーレスランタイム環境では、TLS 認証を設定すると、REST API との安全な一方向または双方向通信を確立できます。

証明書を設定する前に、証明書が.jks 形式であることを確認します。

サーバーレスランタイム環境を使用して安全な REST V2 接続を設定するには、次の前提条件タスクを完了して、TLS 証明書をサーバーレスランタイムの場所に追加します。

1. AWS のサーバーレスエージェント設定用に次の構造を作成します: <Supplementary file location>/serverless_agent_config
2. 一方向の安全な通信の場合はトラストストア証明書を、双方向の安全な通信の場合はトラストストアとキーストアの証明書を AWS アカウントの次の場所にある Amazon S3 バケットにそれぞれ追加します: <Supplementary file location>/serverless_agent_config/SSL
3. 次のコードスニペットをテキストエディタにコピーします。

```
version: 1
agent:
  agentAutoApply:
    general:
      sslStore:
        - fileCopy:
            sourcePath: SSL/<RESTV2_trustStore_cert_name>.jks
```

```
    - fileCopy:
        sourcePath: SSL/<RESTV2_keyStore_cert_name>.jks
```

ここで、ソースパスは AWS の証明書ファイルのディレクトリパスです。

4. 構文とインデントが有効であることを確認してから、ファイルを serverlessUserAgentConfig.yml として次の AWS の場所に保存します: <Supplementary file location>/serverless_agent_config .yml ファイルが実行されると、SSL 証明書が AWS の場所からサーバーレスエージェントディレクトリにコピーされます。
5. REST V2 接続のプロパティで、**[トラストストア]** フィールドと **[キーストア]** フィールドのサーバーレスエージェントディレクトリで次の証明書パスを指定します: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks

サーバーレスランタイム環境の Swagger ファイル

サーバーレスランタイム環境では、次の 2 つのいずれかの方法で Swagger ファイルを設定できます。

- Swagger ファイルのパブリックホスト URL を **Swagger ファイルパス**接続プロパティで指定します。URL は、それ以上の認証とリダイレクトを要求せずにファイルのコンテンツを返す必要があることを確認します。
- Swagger ファイルをサーバーレスエージェントディレクトリに配置します。

サーバーレスランタイム環境で Swagger ファイルを設定するには、次の前提条件タスクを完了して、Swagger ファイルをサーバーレスランタイムの場所に追加します。

1. AWS のサーバーレスエージェント設定用に次の構造を作成します: <Supplementary file location>/serverless_agent_config
2. AWS アカウントの次の場所にある Amazon S3 バケットに Swagger 仕様ファイルを追加します: <Supplementary file location>/serverless_agent_config/restv2
   a. 次のコードスニペットをテキストエディタにコピーします。

```
version: 1
agent:
  dataIntegrationServer:
    AutoApply:
      restv2:
        swaggers:
          - fileCopy:
              sourcePath: restv2/<swagger_file_name1>.json
          - fileCopy:
              sourcePath: restv2/<swagger_file_name2>.json
```

   ここで、ソースパスは AWS の Swagger ファイルのディレクトリパスです。
3. 構文とインデントが有効であることを確認してから、ファイルを serverlessUserAgentConfig.yml として次の AWS の場所に保存します: <Supplementary file location>/serverless_agent_config .yml ファイルが実行されると、SSL 証明書が AWS の場所からサーバーレスエージェントディレクトリにコピーされます。
4. REST V2 接続プロパティで、**[Swagger ファイルパス]** フィールドのサーバーレスエージェントディレクトリの次の Swagger パスを指定します: /home/cldagnt/SystemAgent/serverless/configurations/restv2/<swagger_file_name>.json

ランタイム環境に関するガイドライン

異なるランタイム環境でマッピングを実行する場合は、次の制限を考慮してください。

- Hosted Agent およびサーバーレスランタイム環境の場合、プロキシサーバーを使用して Informatica Intelligent Cloud Services に接続することはできません。

- Hosted Agent の場合、S wagger 仕様ファイル URL はパブリック URL である必要があり、さらに認証とリダイレクトを求めることなく、ファイルのコンテンツを返す必要があります。
- Hosted Agent の場合、CA によって署名されたカスタムサーバー証明書を必要とする、Informatica 証明書トラストストアの一部ではない REST API エンドポイントに接続できません。
- Hosted Agent の場合、JWT ベアラートークン認証を設定することはできません。

第 2 章

REST V2 の接続

REST V2 接続を作成し、web サービスアプリケーションを呼び出します。

接続を作成する際は、Swagger 仕様ファイルと認証方法を必要に応じて指定します。次の認証方法を指定できます。

- 標準
- OAuth 2.0 クライアント資格情報
- OAuth 2.0 認証コード
- JWT Bearer トークン

OAuth 2.0 クライアント資格情報または OAuth 2.0 認証コードの認証を使用して、認証プロセスを行いために仮想プライベートクラウドネットワーク内の認証サーバーに接続できます。

REST V2 コネクタは swagger 仕様バージョン 2.0 をサポートしています。swagger 仕様ファイルには、操作 ID、パスパラメータ、クエリパラメータ、ヘッダフィールド、要求の詳細、および応答の詳細が含まれます。

REST V2 接続は **［接続］** ページで作成します。

REST エンドポイントで swagger 仕様を使用していない場合、管理者から swagger 仕様ファイルを生成できます。**［管理者］** > **［Swagger ファイル］** をクリックして、Swagger 仕様ファイルを生成します。

Swagger ファイルを作成するパラメータの詳細については、データ統合のマニュアルの「*接続*」を参照してください。

REST V2 接続のプロパティ

REST V2 接続をセットアップする際には、接続プロパティを設定する必要があります。

次の表に、標準認証タイプ接続における REST V2 接続のプロパティを示します。

接続プロパティ	説明
ランタイム環境	タスクを実行するランタイム環境の名前。 Secure Agent、Hosted Agent、またはサーバーレスランタイム環境を指定します。
認証タイプ	コネクタが Web サーバーアプリケーションにログインする際に認証が必要な場合に使用される認証メソッドを選択します。デフォルトは［なし］です。

接続プロパティ	説明
認証ユーザー ID	基本認証を選択したときに Web サービスアプリケーションにログインするためのユーザー名。 ダイジェスト認証は適用されません。
認証パスワード	基本認証を選択したときにユーザー名に関連付けられたパスワード。 ダイジェスト認証は適用されません。
OAuth コンシューマキー	Web サービスアプリケーションに関連付けられるクライアントキー。 認証タイプが［OAuth］の場合にのみ必要です。
OAuth コンシューマシークレット	Web サービスアプリケーションに接続するためのクライアントパスワード。 認証タイプが［OAuth］の場合にのみ必要です。
OAuth トークン	Web サービスアプリケーションに接続するためのアクセストークン。 認証タイプが［OAuth］の場合にのみ必要です。
OAuth トークンシークレット	OAuth トークンに関連付けられるパスワード。 認証タイプが［OAuth］の場合にのみ必要です。
Swagger ファイルパス	Swagger 仕様ファイルのファイル名を含む絶対パスまたはホストされた URL。ホストされた URL は、それ以上の認証とリダイレクトを要求せずにファイルのコンテンツを返す必要があります。 Swagger 仕様ファイルの絶対パスを指定する場合、Swagger 仕様ファイルは Secure Agent をホストするマシン上にある必要があります。ユーザーは、フォルダーと仕様ファイルの読み取り権限を持っている必要があります。例: `C:\swagger\sampleSwagger.json`
トラストストアファイルパス	REST API との一方向または双方向の安全な接続を確立するための TLS 証明書を含むトラストストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 トラストストアファイル名とパスワードを JVM オプションとして設定することもできますし、証明書を次のディレクトリにインポートすることもできます。 `<Secure Agent のインストールディレクトリ>\jre\lib\security\cacerts` サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでトラストストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
トラストストアのパスワード	SSL 証明書を含むトラストストアファイルのパスワード。 トラストストアのパスワードを JVM オプションとして設定することもできます。
キーストアファイル名	REST API との双方向の安全な接続を確立するために必要なキーと証明書を含むキーストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 キーストアファイル名と格納場所を JVM オプションとして設定することもできます。また、証明書を任意のディレクトリにインポートすることもできます。 サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでキーストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
キーストアのパスワード	通信を安全に行うために必要なキーストアファイルのパスワード。 キーストアのパスワードを JVM オプションとして設定することもできます。

接続プロパティ	説明
プロキシタイプ	プロキシのタイプ。以下のいずれかのオプションを選択することができます。 - プロキシなし。エージェントレベルまたは接続レベルで設定されたプロキシサーバーをバイパスします。 - プラットフォームプロキシ。エージェントレベルで設定されたプロキシが考慮されます。 - カスタムプロキシ。接続レベルで設定されたプロキシが考慮されます。
プロキシ設定	プロキシ設定形式: `<host>:<port>` 認証されたプロキシサーバーを設定することはできません。
詳細フィールド	Secure Agent が REST エンドポイントに接続するときに使用する引数を入力します。次の引数をセミコロン（;）で区切って指定できます。 `ConnectionTimeout`。REST エンドポイントからの応答を取得するための待機時間（ミリ秒）。接続タイムアウトを過ぎると、接続は終了します。デフォルトはエンドポイント API で定義されているタイムアウトです。 **注**: REST V2 接続タイムアウトとエンドポイント API タイムアウトの両方が定義されている場合、接続は定義されている最短のタイムアウトで終了します。 `connectiondelaytime`。REST エンドポイントに要求を送信するための遅延時間（ミリ秒）。デフォルトは 10000 です。 `retryattempts`。応答で 400 および 500 シリーズのエラーコードが返された場合に接続が試行された回数。デフォルトは 3 です。再試行を無効にするには 0 を指定します。 `qualifiedSchema`。選択したスキーマが修飾されているかどうかを指定します。デフォルトは false です。 例: `connectiondelaytime:10000;retryattempts:5`

OAuth 2.0 クライアント資格情報認証

以下の表に、「OAuth 2.0 - クライアント資格情報」認証タイプ接続の REST V2 接続プロパティを示します。

接続プロパティ	説明
アクセストークン URL	アプリケーションで設定されているアクセストークン URL。
クライアント ID	アプリケーションのクライアント ID。
クライアントシークレット	アプリケーションのクライアントシークレット。
スコープ	API エンドポイントでカスタムスコープが定義されている場合に、アクセス制御を指定します。スコープ属性をスペースで区切って入力します。以下に例を示します。 `root_readonly root_readwrite manage_app_users`
アクセストークンパラメータ	アクセストークン URL で使用する追加パラメータ。パラメータは JSON 形式で定義する必要があります。以下に例を示します。 `[{"Name":"resource","Value":"https://<serverName>"}]`

接続プロパティ	説明
クライアント認証	認証のためにクライアント ID およびクライアントシークレットを送信するオプションを、要求本文または要求ヘッダーのいずれかから選択します。デフォルトは、**[本文でクライアント資格情報を送信する]** です。
アクセストークンの生成	上のフィールドで指定された情報に基づいて、アクセストークンを生成します。
アクセストークン	アクセストークンの値を入力するか、**[アクセストークンの生成]** をクリックして、アクセストークンの値を指定します。 プロキシサーバーを介してアクセストークンの生成呼び出しを渡すには、Secure Agent レベルで認証されていないプロキシサーバーを設定する必要があります。REST V2 接続レベルのプロキシ設定は、アクセストークンの生成呼び出しには適用されません。
Swagger ファイルパス	Swagger 仕様ファイルのファイル名を含む絶対パスまたはホストされた URL。ホストされた URL は、それ以上の認証とリダイレクトを要求せずにファイルのコンテンツを返す必要があります。 Swagger 仕様ファイルの絶対パスを指定する場合、Swagger 仕様ファイルは Secure Agent をホストするマシン上にある必要があります。ユーザーは、フォルダーと仕様ファイルの読み取り権限を持っている必要があります。例: `C:\swagger\sampleSwagger.json`
トラストストアファイルパス	REST API との一方向または双方向の安全な接続を確立するための TLS 証明書を含むトラストストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 トラストストアファイル名とパスワードを JVM オプションとして設定することもできますし、証明書を次のディレクトリにインポートすることもできます。 `<Secure Agent のインストールディレクトリ>\jre\lib\security\cacerts` サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでトラストストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
トラストストアのパスワード	SSL 証明書を含むトラストストアファイルのパスワード。 トラストストアのパスワードを JVM オプションとして設定することもできます。
キーストアファイル名	REST API との双方向の安全な接続を確立するために必要なキーと証明書を含むキーストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 キーストアファイル名と格納場所を JVM オプションとして設定することもできます。また、証明書を任意のディレクトリにインポートすることもできます。 サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでキーストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
キーストアのパスワード	通信を安全に行うために必要なキーストアファイルのパスワード。 キーストアのパスワードを JVM オプションとして設定することもできます。
プロキシタイプ	プロキシのタイプ。以下のいずれかのオプションを選択することができます。 - プロキシなし: エージェントレベルまたは接続レベルで設定されたプロキシサーバーをバイパスします。 - プラットフォームプロキシ: エージェントレベルで設定されたプロキシが考慮されます。 - カスタムプロキシ: 接続レベルで設定されたプロキシが考慮されます。

接続プロパティ	説明
プロキシ設定	プロキシ設定形式: `<host>:<port>` 認証されたプロキシサーバーを設定することはできません。
詳細フィールド	Secure Agent が REST エンドポイントに接続するときに使用する引数を入力します。次の引数をセミコロン（;）で区切って指定できます。 `ConnectionTimeout`: REST エンドポイントからの応答を取得するための待機時間（ミリ秒）。接続タイムアウトを過ぎると、接続は終了します。デフォルトはエンドポイント API で定義されているタイムアウトです。 **注**: REST V2 接続タイムアウトとエンドポイント API タイムアウトの両方が定義されている場合、接続は定義されている最短のタイムアウトで終了します。 `connectiondelaytime`: REST エンドポイントに応答を送信するための遅延時間（ミリ秒）。デフォルトは 10000 です。 `retryattempts`: 応答で 400 および 500 シリーズのエラーコードが返された場合に接続が試行された回数。デフォルトは 3 です。再試行を無効にするには 0 を指定します。 `qualifiedSchema`: 選択したスキーマが修飾されているかどうかを指定します。デフォルトは false です。 例: `connectiondelaytime:10000;retryattempts:5`

OAuth 2.0 認証コード認証

認証コードの認証を使用するには、アプリケーションで次の Informatica リダイレクト URL を登録する必要があります。

`https://`*`<組織の Informatica クラウドホスティング設備>`*`/ma/proxy/oauthcallback`

アクセストークンの有効期限が切れ、応答でエラーコード 400、401 および 403 が返された場合に、顧客のファイアウォールの外側にある Informatica リダイレクト URL からエンドポイントに接続し、新しいアクセストークンの取得を試みます。

以下の表に、「OAuth 2.0 - 認証コード」認証タイプ接続の REST V2 接続プロパティを示します。

接続プロパティ	説明
認証トークン URL	アプリケーションで設定されている認証サーバー URL。
アクセストークン URL	アプリケーションで設定されているアクセストークン URL。
クライアント ID	アプリケーションのクライアント ID。
クライアントシークレット	アプリケーションのクライアントシークレット。
スコープ	API エンドポイントでカスタムスコープが定義されている場合に、アクセス制御を指定します。スコープ属性をスペースで区切って入力します。以下に例を示します。 `root_readonly root_readwrite manage_app_users`

接続プロパティ	説明
アクセストークンパラメータ	アクセストークン URL で使用する追加パラメータ。パラメータは JSON 形式で定義する必要があります。以下に例を示します。 `[{"Name":"resource","Value":"https://<serverName>"}]`
認証コードパラメータ	認証トークン URL で使用する追加パラメータ。パラメータは JSON 形式で定義する必要があります。以下に例を示します。 `[{"Name":"max_age","Value":60},{"Name":"state","Value":"test"}]`
クライアント認証	認証のためにクライアント ID およびクライアントシークレットを送信するオプションを、要求本文または要求ヘッダーのいずれかから選択します。デフォルトは、**[本文でクライアント資格情報を送信する]** です。
アクセストークンの生成	上のフィールドで指定された情報に基づいて、アクセストークンを生成し、トークンをリフレッシュします。
アクセストークン	アクセストークンの値を入力するか、**[アクセストークンの生成]** をクリックして、アクセストークンの値を指定します。 プロキシサーバーを介してアクセストークンの生成呼び出しを渡すには、Secure Agent レベルで認証されていないプロキシサーバーを設定する必要があります。REST V2 接続レベルのプロキシ設定は、アクセストークンの生成呼び出しには適用されません。
リフレッシュトークン	リフレッシュトークンの値を入力するか、**[アクセストークンの生成]** をクリックして、リフレッシュトークンの値を指定します。アクセストークンが有効でないか、有効期限切れの場合、Secure Agent は、リフレッシュトークンを使用して新しいアクセストークンを取得します。 リフレッシュトークンが期限切れの場合は、有効なリフレッシュトークンを指定するか、**[アクセストークンの生成]** をクリックして新しいリフレッシュトークンを生成します。
Swagger ファイルパス	Swagger 仕様ファイルのファイル名を含む絶対パスまたはホストされた URL。ホストされた URL は、それ以上の認証とリダイレクトを要求せずにファイルのコンテンツを返す必要があります。 Swagger 仕様ファイルの絶対パスを指定する場合、Swagger 仕様ファイルは Secure Agent をホストするマシン上にある必要があります。ユーザーは、フォルダーと仕様ファイルの読み取り権限を持っている必要があります。例: `C:\swagger\sampleSwagger.json`
トラストストアファイルパス	REST API との一方向または双方向の安全な接続を確立するための TLS 証明書を含むトラストストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 トラストストアファイル名とパスワードを JVM オプションとして設定することもできますし、証明書を次のディレクトリにインポートすることもできます。 `<Secure Agent のインストールディレクトリ>\jre\lib\security\cacerts` サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでトラストストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
トラストストアのパスワード	SSL 証明書を含むトラストストアファイルのパスワード。 トラストストアのパスワードを JVM オプションとして設定することもできます。

接続プロパティ	説明
キーストアファイル名	REST API との双方向の安全な接続を確立するために必要なキーと証明書を含むキーストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 キーストアファイル名と格納場所を JVM オプションとして設定することもできます。また、証明書を任意のディレクトリにインポートすることもできます。 サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでキーストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
キーストアのパスワード	通信を安全に行うために必要なキーストアファイルのパスワード。 キーストアのパスワードを JVM オプションとして設定することもできます。
プロキシタイプ	プロキシのタイプ。以下のいずれかのオプションを選択することができます。 - プロキシなし: エージェントレベルまたは接続レベルで設定されたプロキシサーバーをバイパスします。 - プラットフォームプロキシ: エージェントレベルで設定されたプロキシが考慮されます。 - カスタムプロキシ: 接続レベルで設定されたプロキシが考慮されます。
プロキシ設定	プロキシ設定形式: `<host>:<port>` 認証されたプロキシサーバーを設定することはできません。
詳細フィールド	Secure Agent が REST エンドポイントに接続するときに使用する引数を入力します。次の引数をセミコロン（;）で区切って指定できます。 `ConnectionTimeout`: REST エンドポイントからの応答を取得するための待機時間（ミリ秒）。接続タイムアウトを過ぎると、接続は終了します。デフォルトはエンドポイント API で定義されているタイムアウトです。 **注:** REST V2 接続タイムアウトとエンドポイント API タイムアウトの両方が定義されている場合、接続は定義されている最短のタイムアウトで終了します。 `connectiondelaytime`: REST エンドポイントに応答を送信するための遅延時間（ミリ秒）。デフォルトは 10000 です。 `retryattempts`: 応答で 400 および 500 シリーズのエラーコードが返された場合に接続が試行された回数。デフォルトは 3 です。再試行を無効にするには 0 を指定します。 `qualifiedSchema`: 選択したスキーマが修飾されているかどうかを指定します。デフォルトは false です。 例: `connectiondelaytime:10000;retryattempts:5`

JWT ベアラートークン認証

REST V2 接続をセットアップする際には、接続プロパティを設定する必要があります。

次の表では、JWT ベアラートークン認証を使用する場合の REST V2 接続のプロパティについて説明します。

接続プロパティ	説明

JWT ヘッダー	JSON 形式の JWT ヘッダー。
	サンプル:
	{
	"alg":"RS256",
	"kid":"xxyyzz"
	}
	HS256 および RS256 アルゴリズムを設定できます。

JWT ペイロード	JSON 形式の JWT ペイロード。
	サンプル:
	{
	"iss":"abc",
	"sub":"678",
	"aud":"https://api.box.com/oauth2/token",
	"box_sub_type":"enterprise",
	"exp":"120",
	"jti":"3ee9364e"
	}
	exp として表される有効期限は、秒単位の相対時間です。有効期限は、トークン発行者の時間（iat）から UTC 形式で計算されます。
	ペイロードに iat が定義されており、有効期限に達すると、マッピングとアクセストークンの生成が失敗します。新しいアクセストークンを生成するには、ペイロードに有効な iat を指定する必要があります。
	iat がペイロードで定義されていない場合、有効期限は現在のタイムスタンプから計算されます。
	有効期限を文字列値として渡すには、値を二重引用符で囲みます。以下に例を示します。
	"exp":"120",
	有効期限を整数値として渡すには、値を二重引用符で囲まないでください。以下に例を示します。
	"exp":120,

認証サーバー	アプリケーションで設定されているアクセストークン URL。
認証の詳細プロパティ	アクセストークン URL で使用する追加パラメータ。パラメータは JSON 形式で定義する必要があります。以下に例を示します。 [\{"Name":"client_id","Value":"abc"},\{"Name":"client_secret","Value":"abc"}]

接続プロパティ	説明
トラストストアファイルパス	REST API との一方向または双方向の安全な接続を確立するための TLS 証明書を含むトラストストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 トラストストアファイル名とパスワードを JVM オプションとして設定することもできますし、証明書を次のディレクトリにインポートすることもできます。 `<Secure Agent のインストールディレクトリ>\jre\lib\security\cacerts` サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでトラストストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
トラストストアのパスワード	SSL 証明書を含むトラストストアファイルのパスワード。 トラストストアのパスワードを JVM オプションとして設定することもできます。
キーストアファイルパス	必須。REST API との双方向の安全な接続を確立するために必要なキーと証明書を含むキーストアファイルの絶対パス。各 Secure Agent マシンのランタイム環境で使用可能なディレクトリパスを指定します。 キーストアファイル名と格納場所を JVM オプションとして設定することもできます。また、証明書を任意のディレクトリにインポートすることもできます。 サーバーレスランタイム環境の場合、サーバーレスエージェントディレクトリでキーストアファイルパスを指定します。 例: /home/cldagnt/SystemAgent/serverless/configurations/ssl_store/<cert_name>.jks
キーストアのパスワード	必須。通信を安全に行うために必要なキーストアファイルのパスワード。 キーストアのパスワードを JVM オプションとして設定することもできます。
プライベートキーのエイリアス	必須。JWT ペイロードの署名に使用されるプライベートキーのエイリアス名。
プライベートキーのパスワード	必須。通信を安全に行うために必要なキーストアファイルのパスワード。プライベートキーのパスワードは、キーストアのパスワードと同じでなければなりません。
アクセストークン	アクセストークンの値を入力するか、**[アクセストークンの生成]** をクリックして、アクセストークンの値を指定します。 プロキシサーバーを介してアクセストークンの生成呼び出しを渡すには、Secure Agent レベルで認証されていないプロキシサーバーを設定する必要があります。REST V2 接続レベルのプロキシ設定は、アクセストークンの生成呼び出しには適用されません。
Swagger ファイルパス	Swagger 仕様ファイルのファイル名を含む絶対パスまたはホストされた URL。ホストされた URL は、それ以上の認証とリダイレクトを要求せずにファイルのコンテンツを返す必要があります。 Swagger 仕様ファイルの絶対パスを指定する場合、Swagger 仕様ファイルは Secure Agent をホストするマシン上にある必要があります。ユーザーは、フォルダーと仕様ファイルの読み取り権限を持っている必要があります。例: `C:\swagger\sampleSwagger.json`
プロキシタイプ	プロキシのタイプ。以下のいずれかのオプションを選択することができます。 - プロキシなし: エージェントレベルまたは接続レベルで設定されたプロキシサーバーをバイパスします。 - プラットフォームプロキシ: エージェントレベルで設定されたプロキシが考慮されます。 - カスタムプロキシ: 接続レベルで設定されたプロキシが考慮されます。

接続プロパティ	説明
プロキシ設定	プロキシ設定形式: `<host>:<port>` 認証されたプロキシサーバーを設定することはできません。
詳細フィールド	Secure Agent が REST エンドポイントに接続するときに使用する引数を入力します。次の引数をセミコロン（;）で区切って指定できます。 `ConnectionTimeout`: REST エンドポイントからの応答を取得するための待機時間（ミリ秒）。接続タイムアウトを過ぎると、接続は終了します。デフォルトはエンドポイント API で定義されているタイムアウトです。 **注**: REST V2 接続タイムアウトとエンドポイント API タイムアウトの両方が定義されている場合、接続は定義されている最短のタイムアウトで終了します。 `connectiondelaytime`: REST エンドポイントに応答を送信するための遅延時間（ミリ秒）。デフォルトは 10000 です。 `retryattempts`: 応答で 400 および 500 シリーズのエラーコードが返された場合に接続が試行された回数。デフォルトは 3 です。再試行を無効にするには 0 を指定します。 `qualifiedSchema`: 選択したスキーマが修飾されているかどうかを指定します。デフォルトは false です。 例: `connectiondelaytime:10000;retryattempts:5`

重要: JWT ヘッダーでの HS256 アルゴリズムのサポートをプレビューできます。評価目的でのプレビュー機能はサポートされていますが、保証対象外で本番環境には対応していません。非本番環境でのみ使用することをお勧めします。Informatica では、本番環境用に次のリリースでプレビュー機能を導入するつもりですが、市場や技術的な状況の変化に応じて導入しない場合もあります。詳細については、Informatica グローバルカスタマサポートにお問い合わせください。機能を使用するには、組織が適切なライセンスを所有している必要があります。

REST V2 接続についてのルールおよびガイドライン

REST V2 接続についてのルールとガイドラインは次のとおりです。

- 接続をテストすると、Secure Agent が次のパラメータを検証します。

  - ローカルの Swagger ファイルのパス、またはホストされた Swagger ファイルの URL。

  - Swagger ファイルの JSON 形式。

  ただし、接続をテストすると、Secure Agent はエンドポイント資格情報を検証しません。

- エージェントレベルまたは接続レベルでプロキシを設定できます。接続レベルでシステムプロキシおよびプロキシを定義する場合に優先されるプロキシ設定を理解するには、次の表を参照してください。

システムプロキシ	REST V2 接続属性			結果
	プロキシなし	プラットフォームプロキシ	カスタムプロキシ	
×	○	×	いいえ	プロキシを考慮しません。

×	×	○	×	プロキシを考慮しません。
×	×	×	○	カスタムプロキシを考慮します。
○	はい	×	いいえ	プロキシを考慮しません。
○	×	○	×	プラットフォームプロキシを考慮します。
○	×	×	○	カスタムプロキシを考慮します。

第 3 章

REST V2 操作

Web サービスアプリケーションからデータを読み取ったり、Web サービスアプリケーションにデータを書き込むには、**Mapping Designer** でマッピングを作成します。

マッピングを作成するには、REST V2 接続と操作を選択する必要があります。Secure Agent は、Web サービスアプリケーションに接続して、データにアクセスしたり、データをトランスフォームしたり、配信したりします。リレーショナルターゲットにデータを書き込む場合は、Secure Agent が Web サービスアプリケーションからフェッチした階層データをリレーショナルデータに変換します。リレーショナルソースからデータを読み取って、REST V2 ターゲットにデータを書き込む場合は、Secure Agent がソースからフェッチしたリレーショナルデータを階層データに変換します。

REST V2 コネクタは、ソーストランスフォーメーション、ターゲットトランスフォーメーション、ミッドストリームトランスフォーメーションで URL リダイレクトを指示する HTTP 300 番台の全ステータスコードを処理できます。

REST V2 コネクタを使用して、ソーストランスフォーメーションとミッドストリームトランスフォーメーションでページングを実行できます。

REST V2 ソース操作

Web サービスアプリケーションからデータを読み取るには、Mapping Designer でソーストランスフォーメーションを作成します。

マッピングでソーストランスフォーメーションに REST V2 接続を選択し、Web サービスアプリケーションからデータを読み取る場合、ソーストランスフォーメーションの［**ソース**］タブで操作を指定します。Secure Agent が、Swagger 仕様で指定されている操作 ID を表示します。

操作 ID を選択し、Request Message Editor で付属のサンプルテンプレートを利用して要求メッセージを作成し、その操作の詳細プロパティを設定します。リレーショナルターゲットに書き込む場合は、階層形式の入力フィールドを、リレーショナル形式の出力フィールドにマッピングすることによって、ソースデータのリレーショナル構造を定義します。マッピングを実行すると、Secure Agent は、Web サービスアプリケーションから、指定されている操作用にデータを取得します。

Request Message Editor を使用した要求の設定

ソーストランスフォーメーションを作成する際、Web サービスアプリケーションで実行する操作に対する XML 要求メッセージを設定します。

要求メッセージを作成するには、［**Request Message Editor**］を使用します。要求メッセージは XML 形式です。その操作用のサンプル要求メッセージを使用すると、要求メッセージをカスタマイズして、データフローに含めるデータを指定できます。

要求をカスタマイズするには、要求メッセージをサンプルテンプレートから［**Request Message Editor**］ペインにコピーします。ここで、XML メッセージを編集したり、要求に対して属性を追加したりできます。要求メッセージから不要なタグや空のタグを削除して、操作の失敗を回避します。

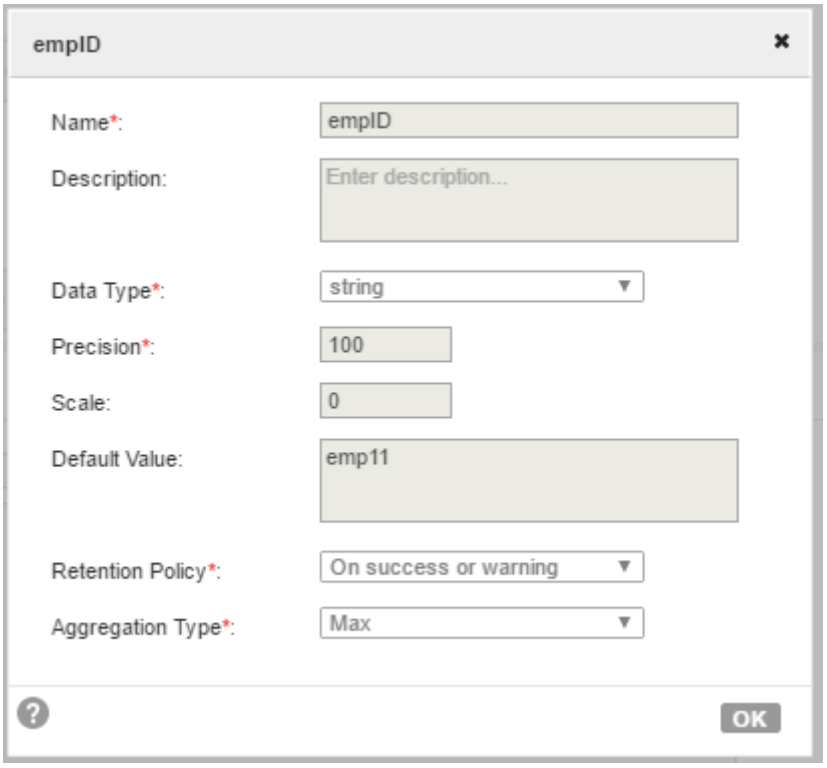
ヘッダーフィールド、パスパラメータ、およびクエリパラメータは、ソーストランスフォーメーション、ターゲットトランスフォーメーション、ミッドストリームトランスフォーメーションのフィールドとして表示されます。

要求メッセージの入力値のパラメータ化

入出力パラメータを使用して、要求 XML の入力値を表現することができます。

Mapping Designer で入出力パラメータを設定します。［**マッピング設計**］ページで、パラメータパネルを開いて、入出力パラメータを設定します。

次の図は、設定された `empID` 入力パラメータ値を示しています。



パラメータを設定したら、要求メッセージ内で$$Name 形式のパラメータ名を使用します。XML は、パラメータ化されたオブジェクトから取得されたフィールド値を使用します。

例えば、`CouchDB_INPUT` 操作に対する XML 要求で、パラメータ化された `empID` を使用するとします。

次のサンプル要求は、XML 要求で指定できるパラメータ化値を示しています。

```
<!--1 or more repetitions:-->
<proc:CouchDB_INPUT xmlns:proc="http://xml.schemas/infa/procedure/">
  <!--Optional:-->
  <CouchDB>
    <!--1 or more repetitions:-->
    <emp>
      $$empID
    </emp>
  </CouchDB>
</proc:CouchDB_INPUT>
```

マッピングでの入出力パラメータ値の設定

差分データロードを管理するタスクが実行されるたびに変更可能な変数値を保持する入出力パラメータを使用できます。

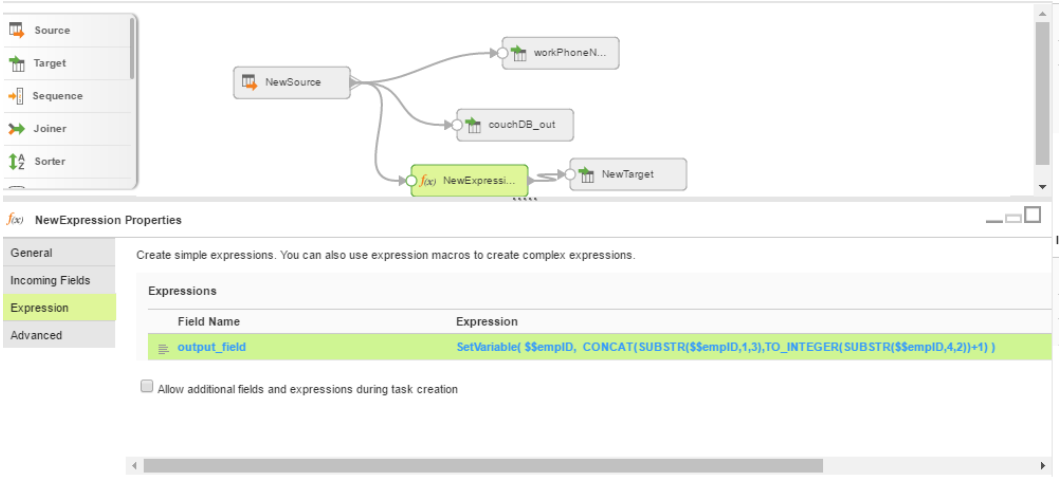
タスクが実行されるたびにパラメータを更新する場合は、マッピングで、入出力パラメータ化されたフィールドを複合式に追加します。入出力パラメータは、タスク手順を格納した値用プレースホルダーのような働きをします。データ統合は、実行時、指定された設定に基づいてパラメータを評価します。

例えば、次のパラメータ化された値を式トランスフォーメーションで使用できます。

```
SetVariable( $$empID, CONCAT(SUBSTR($$empID,1,3),TO_INTEGER(SUBSTR($$empID,4,2))+1) )
```

この例では、SetVariable 関数は、マッピングタスクが実行されるたびに、パラメータ値を設定します。$$empID パラメータにはデフォルト値を設定します。タスクが実行されるたびに$$empID 値を 1 ずつ更新します。

次の図は、Mapping Designer で入出力パラメータに値を割り当てる、式トランスフォーメーションで使用されるロジックを示しています。



タスクが実行されると、Secure Agent により、式内で指定されているロジックに従って、入出力パラメータが更新されます。

ソーストランスフォーメーションでのフィールドマッピング

応答メッセージ形式は、Swagger 仕様ファイル内のサービス応答定義に従います。階層構造の応答フィールドを、出力グループおよびフィールドのリレーショナル構造にマッピングできます。

ソースの操作を設定し、要求メッセージを指定した後、必要なグループとフィールドを出力に含めるように、階層データからリレーショナル形式を作成します。

このためには、出力フィールドとして含める要素を応答構造で選択します。Secure Agent は、実行時に、階層構造の XML 応答をリレーショナルグループに変換します。

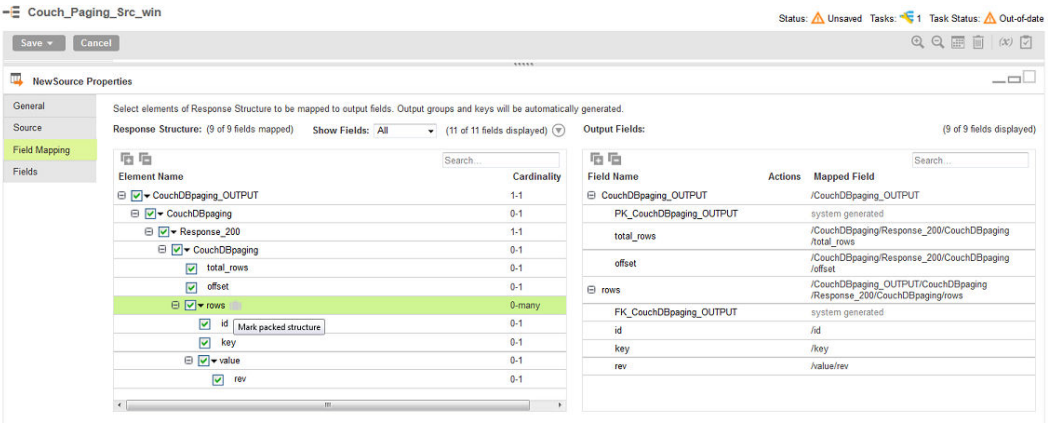
パック済みフィールドの作成

階層要素を解析する必要がない場合は、複数の要素を 1 つのフィールドにパックできます。また、ターゲットに書き込む際に、配列要素をパックすることもできます。

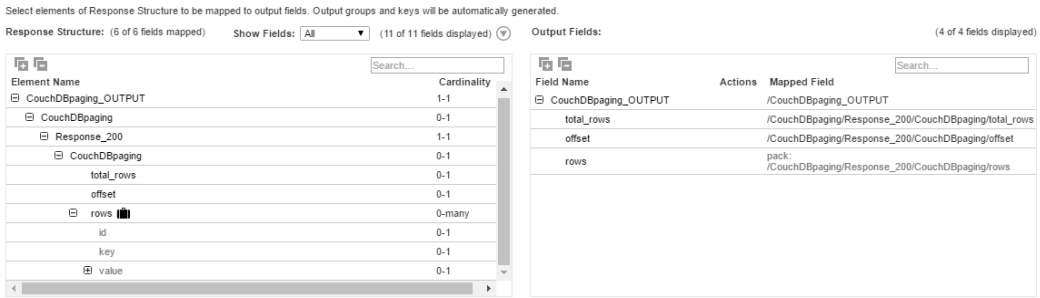
[フィールドマッピング] タブでは、要素の隣にパックアイコンが表示されます。要素の隣にあるパックアイコンを選択すると、エージェントは実行時、その要素とその子要素を単一の XML 文字列にパックします。フィールドマッピングにおいて、パック済みフィールドの子要素に当たるフィールドは選択できません。

パックアイコンをクリックすれば、フィールドのパックを解除できます。

複数のフィールドをパックするには、行要素の隣に表示されるパックアイコンをクリックします。



次の図は、バック済み行要素を示しています。



次のサンプルは、タスク実行後のターゲットファイル内の rows フィールドの出力を示しています。

```
<infa_packed>
<rows>
<id>52835a523b9b6816ec81e9585b09c987</id>
<key>52835a523b9b6816ec81e9585b09c987</key>
<value>
<rev>1-c7fe1b7b899a9ce50319a47d675af735</rev>
</value>
</rows>
</infa_packed>
```

REST V2 ミッドストリーム操作

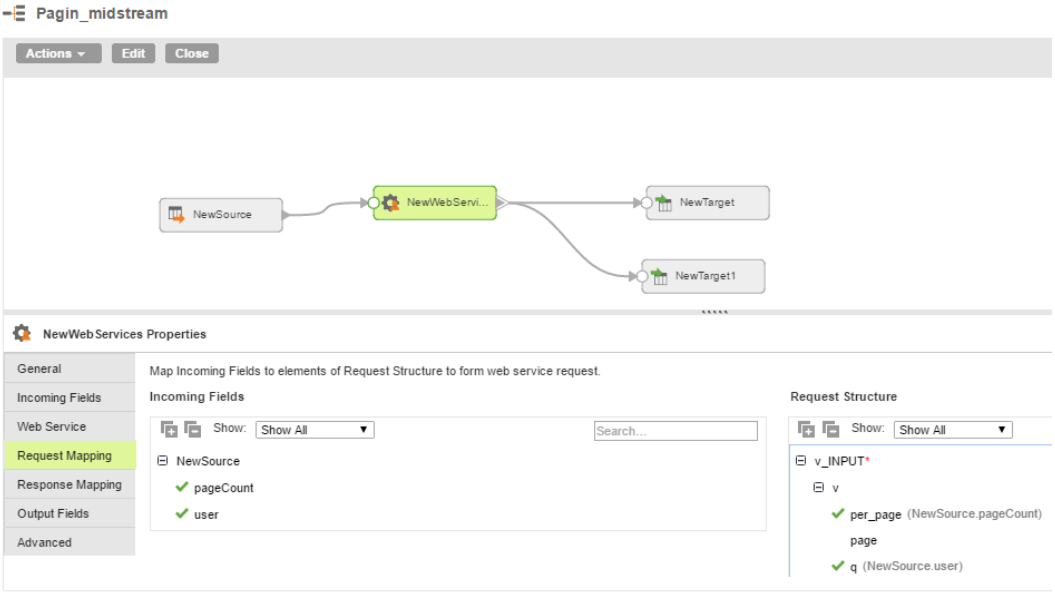
階層データに対して操作を実行するために、Web サービスアプリケーションと対話する REST V2 ミッドストリームトランスフォーメーションを作成します。

REST V2 コネクタミッドストリームを使用する場合、ビジネスサービスを作成し、そのビジネスサービスに対して REST V2 接続、および必要な REST 読み取り操作または REST 書き込み操作を関連付けます。ビジネスサービスを使用して、Mapping Designer で Web サービストランスフォーメーションに操作を追加することもできます。

注: ビジネスサービスを設定する際、**[操作]** リストに表示される値は、Swagger 仕様ファイルに指定されている操作 ID に基づきます。Swagger 仕様ファイル内の操作 ID が **[操作]** リストにない場合は、Informatica グローバルサポートにお問い合わせください。

Web サービストランスフォーメーションは、Web サービスクライアントとして Web サービスアプリケーションに接続し、データへのアクセス、データの変換、配信を行います。Mapping Designer で Web サービストランスフォーメーションを使用して、Web サービス要求を作成し、Web サービス応答を解析します。

次の図は、マッピングにおける REST V2 ミッドストリームトランスフォーメーションを示しています。



Web サービストランスフォーメーションの［応答マッピング］タブ上で、要求により、指定された REST 操作用の応答構造が生成されるので、ここから、出力に必要とされる要素を選択できます。出力グループと出力キーは自動的に生成されます。

デフォルトでは、応答構造にはフォールト出力グループが表示されます。フォールトグループには、要求 XML、エラーコード、およびエラーメッセージが含まれます。フォールトグループは、新しいビジネスサービスに対して表示されます。旧バージョンのコネクタで作成したビジネスサービスにフォールトグループをマッピングする場合は、新しいビジネスサービスを作成し、これを再度インポートする必要があります。

REST V2 ターゲット操作

Web サービスアプリケーションにデータを書き込むには、Mapping Designer でターゲットトランスフォーメーションを作成します。マッピングでターゲットトランスフォーメーションに対して REST V2 接続を選択すると、Secure Agent が、その接続の Swagger 仕様に指定されている操作 ID を表示します。その接続に設定されているオブジェクトに基づいて操作を選択する必要があります。リレーショナル形式のデータを Web サービスアプリケーションに書き込む場合は、Secure Agent は、Web サービスアプリケーションに書き込む前にリレーショナルデータを階層形式に変換します。

REST V2 ターゲットを設定する際、**［ターゲット］**タブで**［操作］**を選択すると、有効な操作一覧が表示されます。Secure Agent は、**［ターゲット］**タブで選択された操作の要求メッセージ構造に基づいてターゲットフィールドを作成します。

Web サービスアプリケーションにデータを書き込むと、Secure Agent は、ソースからのすべてのマッピング済み入力フィールドを含めます。マッピングを実行する前に、Web サービスアプリケーションのターゲットオブジェクトにデータを書き込むための複数の入力ソースを追加して、入力ソース間にプライマリキーと外部キーのリレーションを定義できます。ターゲットトランスフォーメーションの［フィールドマッピング］タブに、各入力ソースが別々のグループとして表示されます。［フィールドマッピング］タブでキーリレーションを定義できます。

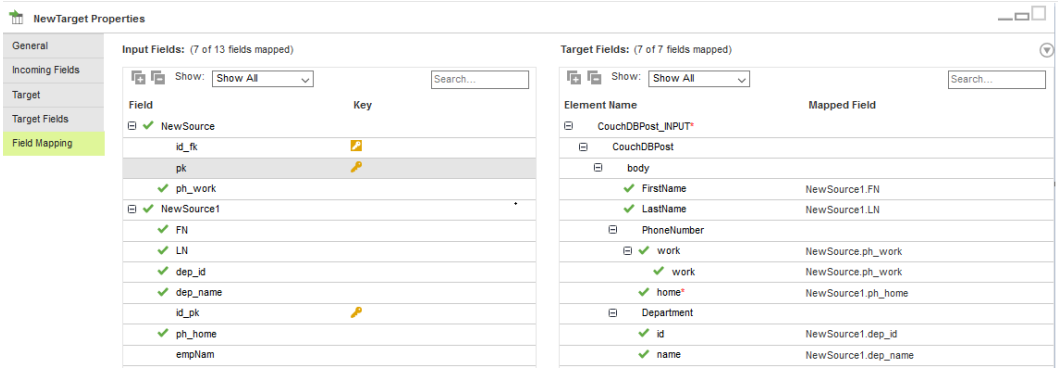
ターゲットトランスフォーメーションのフィールドマッピング

リレーショナル構造にあるフィールドを Web サービスアプリケーションで使用される階層構造にマッピングできます。

ターゲットトランスフォーメーションの［**フィールドマッピング**］タブで、［**ターゲットフィールド**］内のフィールドは階層形式で表示されます。ターゲットフィールドは、ターゲットトランスフォーメーション用に選択された操作の要求メッセージ構造によって決まります。

各ソースオブジェクトは、［**入力フィールド**］領域にグループとして表示されます。［入力フィールド］領域でフィールドを選択し、フィールドを要求にマッピングできます。入力フィールドに複数の入力グループが含まれている場合、グループを要求内の対応するノードにマッピングします。要求に必要なすべてのフィールドをマッピングしてください。

次の図は、従業員データを更新するために、REST V2 ターゲットにソースファイルからマッピングされた入力フィールドの例です。



配列要素を含む要求の設定

次のいずれかの方法により、配列要素を含む REST V2 ターゲットと REST V2 ミッドストリームトランスフォーメーションの要求を作成できます。

- 複数のソースグループを使用した要求の作成
- 単一のソースグループを使用した要求の作成

注: 名前のない配列を要求本文に設定することはできません。

複数のソースグループを使用した要求の作成

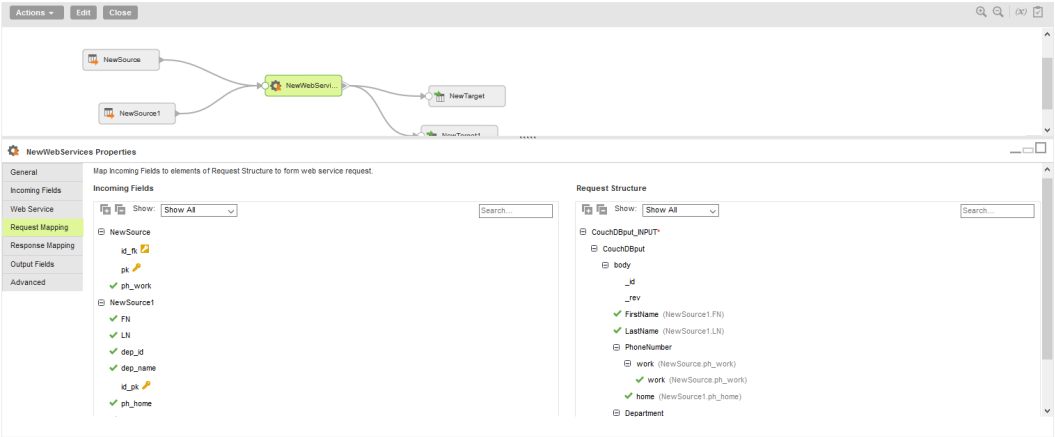
例えば、要求メッセージに、従業員の仕事用の電話番号など、複数回発生する要素が含まれる場合、この要素に対する入力フィールドは複数のソースグループから発生することになります。

複数のリレーショナルソースに対して 1 つの要求を作成するには、外部キーを使用して、複数の入力フィールドにおけるソースグループを、その親ソースグループと関連付けます。

複数発生する要素に対する親ソースグループを判断するには、次の手順を実行します。

- 親グループは、隣接した複数発生する親要素である必要があります。
- 複数発生する親要素がない場合、複数発生する要素を、ルート要素にマッピングされたソースグループにマッピングします。

例えば、次の図は、複数発生要素 work の、リレーショナルデータソースから REST V2 ターゲットへのマッピングを示したものです。

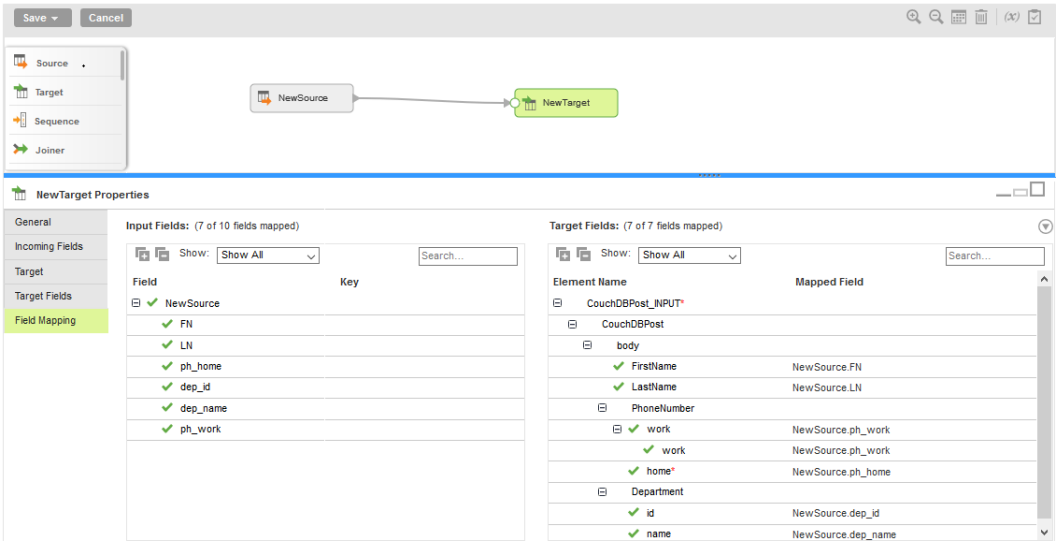


単一のソースグループを使用した要求の作成

ミッドストリームトランスフォーメーションまたは REST V2 ターゲットに対して、非正規化ソースを使って要求を作成するには、次の手順を実行します。

1. すべてのフィールドを単一のソースファイルに含めます。複数発生する要素は、非正規化ソースファイル内の最後のフィールドにくるようにします。
2. 管理者>［ランタイム環境］をクリックして、エージェントを選択します。
3. **［カスタム構成の詳細］**で新しいプロパティを追加します。
4. **［サービス］**で［データ統合サービス］、**［タイプ］**で［Tomcat］を選択します。
5. プロパティ名を ALLOW_DUPLICATE_VALUES と指定します。
6. 値を true に設定します。
7. **［完了］**をクリックして変更を保存します。
8. 対応するソースをターゲットフィールドにマッピングします。

例えば、次の図では、work フィールドが REST V2 ターゲットの配列要素にマッピングされています。



REST エンドポイント URL へのファイルのアップロード

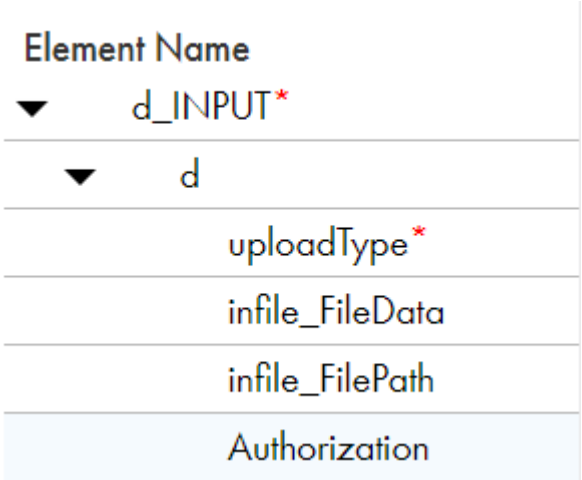
REST API 呼び出しの一部として REST エンドポイント URL にファイルをアップロードできます。

入力としてファイルを REST V2 コネクタに渡すには、要求の content-type を multipart/form-data または binary に設定する必要があります。swagger 定義で formData タイプの入力パラメータを使用する場合、ファイル境界がアップロードしたファイルのコンテンツに追加され、ファイルの開始と終了を示します。swagger 定義で binary タイプの入力パラメータを使用する場合、コンテンツはファイル境界を持たずに生成されます。

file タイプの formData パラメータを swagger の入力パラメータの 1 つとして定義する必要があります。以下に例を示します。

```
{ "name":"file",
"in":"formData",
"description":"file to upload",
"required":false,
"type":"file" }
```

次の図は、サンプルの REST V2 オブジェクト階層を示しています。



入力としてファイルを渡すには、次のいずれかの方法を使用します。

- xxx_FilePath 入力フィールドには、ソース内の値として完全なファイルパスを指定します。すべてのファイル形式で使用されます。
- xxx_FileData 入力フィールドには、ソース内のファイルの Base64 エンコード値を渡します。.pdf、.jpg、.xls、.doc などのファイル形式で使用されます。Base64 エンコード値の長さは 65535 文字以内である必要があります。
- xxx_FileData 入力フィールドには、ソース内の文字列としてファイル値を渡します。.txt、.JSON、.xml などのプレーンテキストファイル形式で使用されます。ファイルサイズは 65535 文字以内である必要があります。

次の図は、バイナリファイルをアップロードするためのサンプルの Swagger ファイルを示しています。

```
"paths": {
                "/drive/v3/files/": {
                        "post": {
                                "tags": ["d"],
                                "summary": null,
                                "description": null,
                                "operationId": "d",
                                "produces": ["application/json"],
                                "consumes": ["binary"],
                                "parameters": [
         {
            "name":"p1",
            "in":"formData",
            "description":"file to upload",
            "required":false,
            "type":"file"
         }
```

応答ヘッダーの解析

REST API エンドポイントが送信した応答のヘッダーコンテンツを解析できます。

ヘッダーコンテンツを解析するには、Swagger ファイルの headers タグを使用します。Set-Cookie タグを使用すると、header タグの一部として Cookie コンテンツを解析することもできます。REST V2 コネクタが、Swagger ファイルに定義されているフィールドを解析します。Web サービスで使用される、ヘッダーまたは Cookie 用の追加フィールドは、それぞれ OtherResponseHeaders フィールドと OtherCookies フィールドの下に表示されます。

応答ヘッダーの定義

```
"responses" : {
....
"headers": {
"places": {
"type" : "string",
"description": "calls per hour allowed by the user"
},
"sessionId": {
"type": "integer",
"description": "date in UTC when token expires"
},
"Set-Cookie/activeSession": {
```

```
"type" : "string",
"description": "cookies sample"
},
"Set-Cookie/PhoneId": {
"type": "integer",
"description": "cookies sample"
}
}
```

注: Set-Cookie タグは大文字小文字を区別します。

セキュリティ定義の解析

REST API エンドポイントで操作を実行するときに、セキュリティ定義を設定できます。

Swagger ファイルの securityDefinitions 入力パラメータを定義して、操作内でセキュリティ定義を呼び出します。securityDefinitions パラメータでは、セキュリティ定義タイプ basic および apiKey のみを使用できます。

securityDefinitions の定義

```
...
"security": [{"type": [],"id": [],"basic_key": []}]
}
}
},
"securityDefinitions": {
"type": {
"type": "apiKey",
"name": "type",
"in": "query",
"default":"json"
},
"id": {
"type": "apiKey",
"name": "id",
"in": "query"
},
"basic_key": {
"type": "basic",
```

```
"name": "basicAuth",
"in": "header"
}
}
```

OAuth バージョン 2.0 を使用する Web サービスに接続する場合は、要求メッセージでカスタムヘッダーフィールドまたはクエリパラメータとして認証の詳細の値を指定してください。

REST V2 操作についてのルールおよびガイドライン

REST V2 操作についてのルールとガイドラインは次のとおりです。

- 要求メッセージを検証することはできません。
- ヘッダーフィールド、パスパラメータ、クエリパラメータに関連付けられた要素は、ソーストランスフォーメーション、ターゲットトランスフォーメーション、ミッドストリームトランスフォーメーションの入力フィールドとして表示されます。
- 認証が有効なプロキシサーバーを設定することはできません。
- 要求および応答構造の xml 要素を解析するには、要素が親要素で囲まれていることを確認してください。以下に例を示します。
  `<parentElement><childElement>xyz</childElement></parentElement>`
- Integer、Decimal、Long データ型のみ使用できます。Swagger 仕様ファイル内の Long データ型の値が含まれるフィールドでは、タイプを Normal に設定し、形式を int64 に設定します。
- ソーストランスフォーメーションではパック済みフィールドのみ使用できます。
- JSON ペイロードのフィールド名では拡張 ASCII 文字と I18N 文字は使用できません。
- クエリパラメータと操作名のフィールド名で特殊文字を使用することはできません。
- 配列型のフィールドで特殊文字を使用することはできません。
- メディアタイプ application/xml または application/x-www-form-urlencoded を使用してデータを処理する場合、ヘッダー、本文、クエリパラメータ、および入力フィールドで&、>、<などの特殊文字を使用しないでください。これらの特殊文字は、ターゲット API でエンコードされた形式に変換されます。例えば、&は＆amp としてエンコードされます。
- 無効な XML タグや特殊文字（&、>、<）を、アプリケーションの本文や入力フィールドまたは xml メディアタイプで使用すると、タスクは XML の検証中に例外をスローします。例えば、入力データが `<BODID>infor-nid:infor:15101113565:?Employee&verb=Sync</BODID>`の場合、次のエラーメッセージが入力データに基づいて表示されます。
  `[ERROR][致命的なエラー] :1:516: エンティティ「verb」への参照の delimiter 末尾には区切り文字「;」が必要です`

  このエラーメッセージは無視してかまいません。
- ページパラメータ、開始ページ、終了ページ、URL の上書きなどの詳細プロパティに対して、入出力パラメータを作成することはできません。
- REST V2 コネクタは、Content-Type ヘッダーと Accept ヘッダーを上書きします。
- エンドポイント API パスにスペースが含まれる場合は、そのパスをパスパラメータとして定義する必要があります。

- 指定した Swagger URL にアクセスできない、またはプロキシサーバーにアクセスできない場合、マッピングは失敗し、セッションログに次のいずれかのエラーが記録されます。ソース、ミッドストリーム、ターゲットのマッピングで同じ振る舞いになります。

  - [ERROR] com.informatica.sdk.helper.common.ApiException: java.net.UnknownHostException: inv28pers108Communication exception, Proxy settings might be incorrect.

  - [ERROR] com.informatica.sdk.helper.common.ApiException: java.net.ConnectException: Connection refused: connectCommunication exception, Proxy settings might be incorrect.

- ペイロードが JSON 形式である場合、要求された入力フィールドでページパラメータを定義できます。ページパラメータは、JSON 要求ペイロードが、名前のないオブジェクトである場合のみ、解決可能です。例:

  ```
  "body" : {
  "a" : "ty",
  "b" : 0
  }
  ```

  次のシナリオではページパラメータは解決されません。

  - JSON 要求ペイロードが名前付きオブジェクトである場合。
  - 定義されたパラメータが、ネストされた入力要素である場合。
  - 定義されたパラメータが配列要素である場合。
  - ペイロードが XML 形式である場合。

- body パラメータおよび securityDefinitions パラメータ以外の要求入力パラメータのデフォルト値を定義できます。以下に例を示します。

  ```
  "parameters": {
  "keyReference": {
  "name": "keyReference",
  "in": "header",
  "required":false,
  "type": "integer",
  "default": 123
  }
  }
  ```

  デフォルト値は、integer、string、boolean、array of primitive 型フィールドに適用できます。

- REST V2 コネクタをソースとして使用しているときに、要求メッセージエディタで入力パラメータに空値を渡した場合、Swagger ファイルで定義されているデフォルト値は適用されません。要求メッセージフィールドに値を指定しない状態で、そのフィールドに Swagger 定義で定義されたデフォルト値を適用する場合には、要求メッセージエディタからそのフィールドを削除する必要があります。

- 入力パラメータのデフォルト値が空白として定義されている場合、マッピングで明示的に値が指定されない限り、空白値は NULL として扱われます。

- Swagger ファイルのインライン要素を使用して XML 属性を定義することは出来ません。Swagger ファイルの別の要素として XML 属性を定義し、データを抽出するときにその XML 属性を参照してください。

- REST V2 接続を使用してソースまたはターゲット操作を設定し、**[操作]** フィールドに有効な名前を手動で入力してオブジェクトを変更しようとした場合、タスクは次の検証エラーにより失敗します。
  オブジェクトの検証中にエラーが発生しました: 選択した接続はこの操作をサポートしていません。

  回避策: **[選択...]** ダイアログボックスから同じ操作を選択して、タスクを再度実行します。

- 応答の配列がある JSON データを読み取ると、セッションログには配列の最初のインスタンスのみが含まれます。

- 要求本文でパラメータを定義する場合、パラメータ下と定義下の名前が同じであることを確認してください。次の例では、パラメータ名が **getAccounts** の場合、$ref の定義名も **getAccounts** である必要があります。

```
"parameters": {
"name": "getAccounts",
"in": "body",
"required": true,
"schema":{
"$ref":"#/definitions/getAccounts"
 }
```

- 単一 XML 要素で属性または値を定義できます。次の例では、単一 XML 要素に属性と値の両方が含まれており、これは適用できません。

```
<wd:ID wd:type="Organization_Role">Manager</wd:ID>
```

第 4 章

REST V2 コネクタを使用したマッピングとマッピングタスク

マッピングを作成するには、データ統合の Mapping Designer を使用します。マッピングを作成するときは、Rest V2 オブジェクトを表すソースまたはターゲットを設定します。詳細モードでは、Mapping Designer で、高度な機能を有効にするためのトランスフォーメーションと関数を含むようにマッピングキャンバスが更新されます。

マッピングにおける REST ソーストランスフォーメーション

ソーストランスフォーメーションを設定する際、REST V2 接続を選択し、Web サービスソースを表す操作を指定します。

接続を通してソース操作を選択できます。要求メッセージテンプレートを使用して要求メッセージを作成します。要求 XML で入力値をパラメータ化できます。ページング属性を設定します。

応答構造は、フィールドマッピングに表示できます。応答構造の要素を出力フィールドにマッピングすると、Secure Agent は出力グループを作成し、加えて、フィールド名のプライマリキーと外部キーを作成します。マッピングタスクでマッピングをデプロイし、そのタスクを実行すると、Secure Agent が Web サービスからデータを読み取ります。

詳細ソースプロパティ

マッピングで、Web サービスアプリケーションソースを表すソースを設定できます。マッピングで使用される REST ソース接続に対して、Mapping Designer の［**ソース**］タブで詳細プロパティを設定できます。

以下の表に、ソースに設定できる詳細プロパティを示します。

プロパティ	説明
ページングタイプ	次のいずれかの値を指定します。 **ページ**。ページングを有効にして、[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] の各プロパティ値を考慮します。 **なし**。[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] のプロパティ値を無視します。
ページパラメータ	ページング操作で使用するパラメータの名前。クエリパラメータまたはパスパラメータを使用できます。 パラメータは整数型で、要求メッセージから抽出されます。
開始ページ	ページ操作の実行範囲の最初のページを示すページ番号。
終了ページ	ページ操作の実行範囲の最後のページを示すページ番号。デフォルトは 10000 です。 ページングは、終了ページに達するか、応答終了式が満たされると停止します。
ページ増分ファクタ	ページパラメータを増分する整数。ページ増分ファクタは、要求ごとにフェッチされるレコード数と同じにする必要があります。ページ増分ファクタオプションが、要求ごとにフェッチされるレコード数と同じでない場合、2 つのコール間でレコードがなかったり重複したりすることが発生します。
応答終了式	ページングを制御する式または文字列を指定します。次のいずれかの動作になります。 - 文字列または式を指定した場合、値がページ応答と一致すると、ページングが停止します。応答終了と一致するページは解析されません。 - 文字列も式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。 - 終了ページと応答終了式の両方を指定すると、先に満たされた条件に応じてページングが停止します。 - 終了ページも応答終了式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。
URL の上書き	Swagger 仕様で指定されている URL を上書きします。URL の上書きではクエリパラメータを使用できません。パスパラメータを [URL の上書き] に含める場合は、パスパラメータを中かっこ{}で囲むようにしてください。例: Swagger 仕様で指定されている URL: `http://invr28pers102:13080/sample/day/20170505?a:b` 20170505 をパス変数として変数名を `path1` と定義すると、URL の上書きにより、`http://invr28pers102:13080/sample/day/{path1}`のようになります。
トレースレベル	ソースのログに表示される情報の量。 以下のトレースレベルを使用します。 - 簡易 - ノーマル - 詳細 - 初期化 - 詳細 デフォルトは [ノーマル] です。
Web サービス応答のキャッシュサイズ (KB)	Web サービス応答で利用可能なメモリ。 Web サービス応答に多数の行またはカラムが含まれている場合は、キャッシュサイズを大きくします。デフォルトは 1024 KB です。最大は 99999KB です。

応答終了式のサンプル

ページングに対して応答終了式を使用します。次のスニペットはページのサンプル応答です。

```
{
"code": "SUCCESS",
"validationResult": [],
"systemErrors": [],
"patientResponseData": [],
"count": 0,
"message": "Unable to retrieve the implant details"
}
```

応答終了式に対して、文字列「Unable to retrieve the implant details」を使用できます。応答ページで複数の条件を一致させる必要がある場合は、次の式を使用できます。

```
(.*)"patientResponseData": []](.*)Unable to retrieve the implant details
```

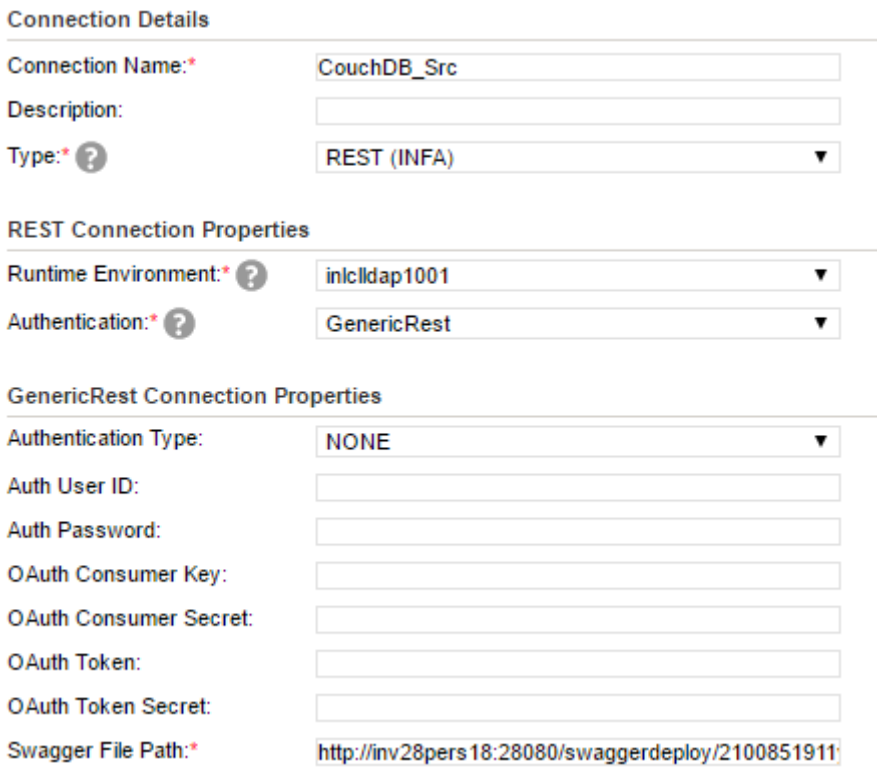
上記の式では、ページ応答で「"patientResponseData": []」と「Unable to retrieve the implant details」の両方の条件が満たされたところでページングが停止するようになっています。

REST ソースマッピングの例

人事部長が、従業員の姓、名、電子メール、電話番号といった連絡先情報を Apache CouchDB からフラットファイルに抽出しようとしています。

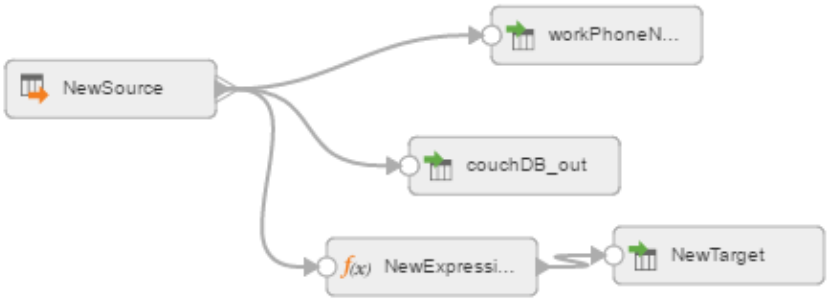
1. REST 接続を作成します。接続プロパティで、Swagger 仕様ファイルの絶対パスまたはホストされた URL、および認証方法を指定していることを確認します。

   次の図は、設定された REST 接続を示しています。



2. フラットファイルにデータを書き込むためのフラットファイル接続を作成します。
3. REST マッピングを作成します。

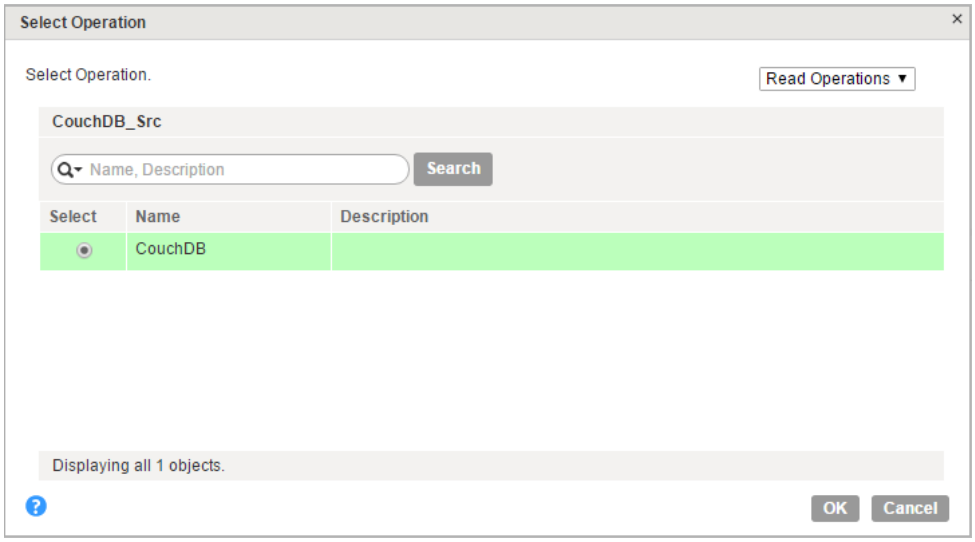
   次の図に、REST マッピングを示します。



4. ソーストランスフォーメーションを追加します。全般プロパティで名前と説明を指定します。
5. **［ソース］**タブで、次の手順を実行します。
   a. **［接続］**フィールドで、Couch データベースに接続するための設定済み REST 接続を選択します。

b. ［操作］フィールドで、CouchDB を操作として選択します。

次の図に、CouchDB 操作を示します。



c. **［要求オプション］**セクションで、要求メッセージを次の XML 形式で設定し、メッセージ内の属性を指定して、メッセージを検証します。

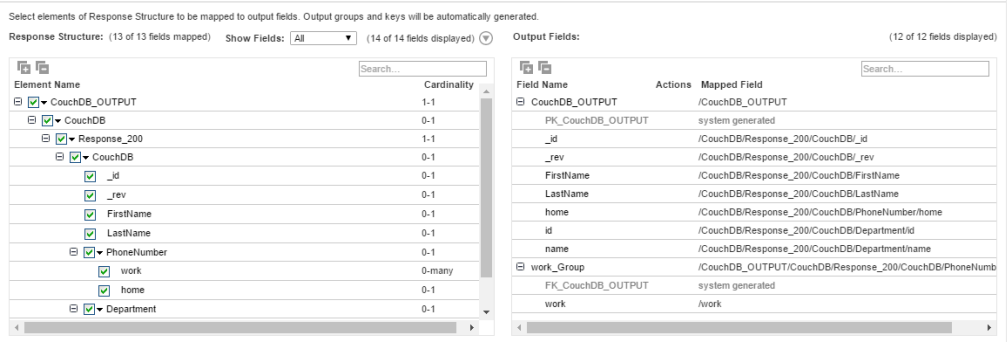
```
<!--1 or more repetitions:-->
<proc:CouchDB_INPUT xmlns:proc="http://xml.schemas/infa/procedure/">
  <!--Optional:-->
  <CouchDB>
    <!--1 or more repetitions:-->
    <emp>
      $$empID
    </emp>
  </CouchDB>
</proc:CouchDB_INPUT>
```

要求メッセージが、指定の従業員 ID に基づいて従業員の詳細をフェッチします。

d. **［詳細プロパティ］**セクションで、トレースレベルを［ノーマル］に設定し、デフォルトのキャッシュサイズ 1024 KB を使用します。

6. ［フィールドマッピング］タブで、応答構造から、出力フィールドにマッピングする_id、FirstName、LastName などの要素を選択します。

次の図は、左側のペインの階層形式の応答構造と、右側のペインのリレーショナル形式の出力グループを示しています。



Secure Agent は、2 つの出力グループ workPhoneNumber と couchDB_out を作成します。これが 2 つのリレーショナル出力ファイルになります。プライマリキーと外部キーが自動的に生成されます。

7. 3 つのターゲットトランスフォーメーションを追加し、各トランスフォーメーションに対してターゲットオブジェクトを指定します。以下の手順を実行します。
   a. 各ターゲットトランスフォーメーションに対して、フラットファイルにデータを書き込むためのフラットファイル接続を選択します。
   b. ターゲットにデータを書き込むためのターゲットオブジェクトとして workPhoneNumber.csv ファイル、couchDB_out.csv ファイル、NewTarget.csv ファイルを作成します。
   c. ターゲットオブジェクトに関連付ける出力グループを CouchDB から選択します。
8. 式トランスフォーメーションを追加して、`SetVariable` 関数とパラメータ化値`$$empID` を式に含め、タスクが、各実行の終了時に 1 ずつ `empID` 値を更新するようにします。

   次の図は、入出力パラメータを利用する、設定済みの式を示しています。

9. **［新規］** > **［タスク］** > **［マッピングタスク］** の順にクリックして、タスクに対してマッピングを選択します。
10. マッピングタスクを保存して実行すると、Secure Agent は、CouchDB から従業員レコードを取得し、対応するフラットファイルにそのデータを書き込みます。

マッピングにおける REST ミッドストリームトランスフォーメーション

ミッドストリームトランスフォーメーションを設定する前に、**［新規］** > **［コンポーネント］** タブで、ビジネスサービスを作成する必要があります。ビジネスサービスを作成する際、REST V2 接続と操作を選択する必要があります。

ミッドストリームトランスフォーメーションを設定する際、**［Web サービス］** タブでビジネスサービスと操作を選択します。

［要求マッピング］ タブで、ソースファイルから入力フィールドを要求構造内の要素にマッピングできます。

応答構造は、フィールドマッピングに表示できます。応答構造の要素を出力フィールドにマッピングすると、Secure Agent は出力グループを作成し、加えて、フィールド名のプライマリキーと外部キーを作成します。

詳細ミッドストリームプロパティ

以下の表に、ミッドストリームトランスフォーメーションで設定できる詳細プロパティを示します。

プロパティ	説明
ページングタイプ	次のいずれかの値を指定します。 **ページ**。ページングを有効にして、[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] の各プロパティ値を考慮します。 **なし**。[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] のプロパティ値を無視します。
ページパラメータ	ページング操作で使用するパラメータの名前。クエリパラメータまたはパスパラメータを使用できます。 パラメータは整数型で、要求メッセージから抽出されます。
開始ページ	ページ操作の実行範囲の最初のページを示すページ番号。
終了ページ	ページ操作の実行範囲の最後のページを示すページ番号。デフォルトは 10000 です。 ページングは、終了ページに達するか、応答終了式が満たされると停止します。
ページ増分ファクタ	ページパラメータを増分する整数。ページ増分ファクタは、要求ごとにフェッチされるレコード数と同じにする必要があります。ページ増分ファクタオプションが、要求ごとにフェッチされるレコード数と同じでない場合、2 つのコール間でレコードがなかったり重複したりすることが発生します。
応答終了式	ページングを制御する式または文字列を指定します。次のいずれかの動作になります。 - 文字列または式を指定した場合、値がページ応答と一致すると、ページングが停止します。応答終了と一致するページは解析されません。 - 文字列も式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。 - 終了ページと応答終了式の両方を指定すると、先に満たされた条件に応じてページングが停止します。 - 終了ページも応答終了式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。
URL の上書き	Swagger 仕様で指定されている URL を上書きします。URL の上書きではクエリパラメータを使用できません。パスパラメータを [URL の上書き] に含める場合は、パスパラメータを中かっこ{}で囲むようにしてください。例: Swagger 仕様で指定されている URL: `http://invr28pers102:13080/sample/day/20170505?a:b` 20170505 をパス変数として変数名を `path1` と定義すると、URL の上書きにより、`http://invr28pers102:13080/sample/day/{path1}`のようになります。
Web サービス要求のキャッシュサイズ（KB）	デフォルトは 100 KB です。要求が 100 KB を超える場合は、キャッシュサイズを増やすことができます。
Web サービス応答のキャッシュサイズ（KB）	Web サービス応答で利用可能なメモリ。Web サービス応答に多数の行またはカラムが含まれている場合は、キャッシュサイズを大きくします。デフォルトは 100 KB です。

プロパティ	説明
入力フラッシュを許可	該当なし。
トランザクションコミット制御	該当なし。

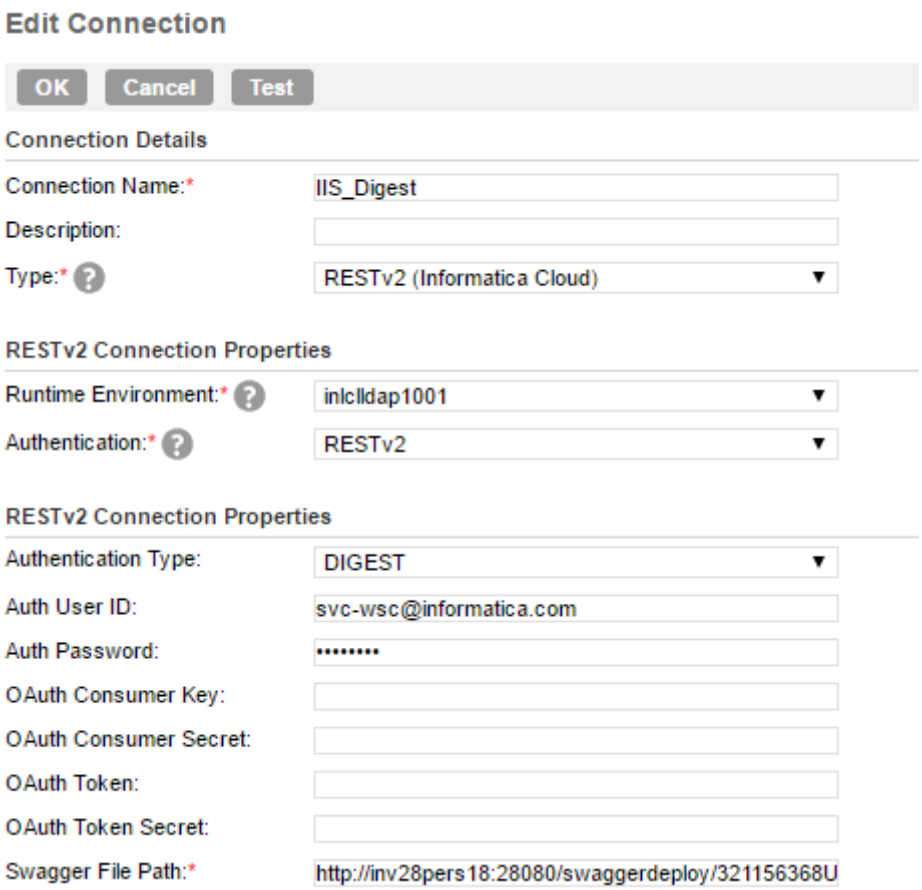
ミッドストリームトランスフォーメーションマッピングの例

人事部長が、従業員の詳細な情報を Apache CouchDB から抽出しようとしています。

CouchDB から従業員の詳細な情報を抽出し、フラットファイルに書き込むには、以下のタスクを実行します。

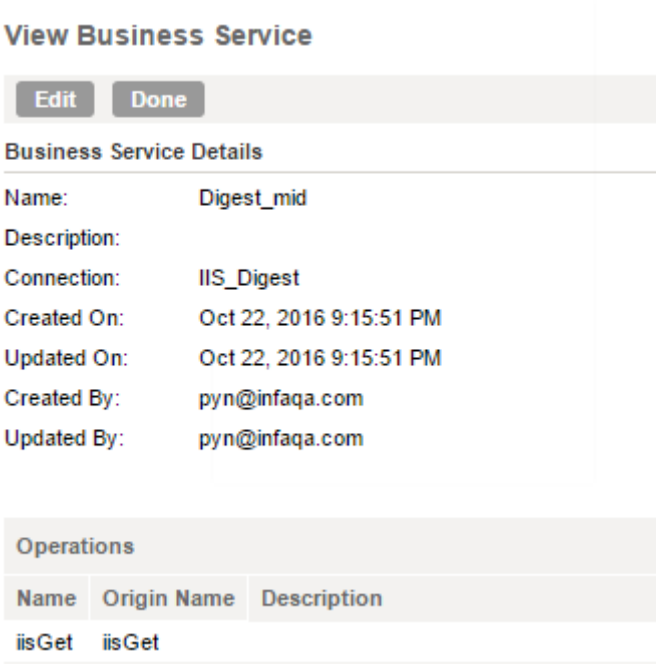
1. CouchDB からデータを読み取るための REST V2 接続を作成します。

   次の図は、設定された CouchDB 接続を示しています。

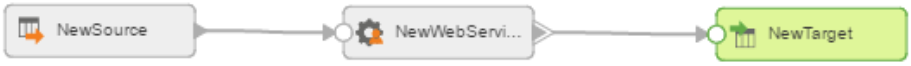


2. ビジネスサービスを作成して、REST V2 接続と操作を関連付けます。

次の図は、設定済みのビジネスサービスを示しています。

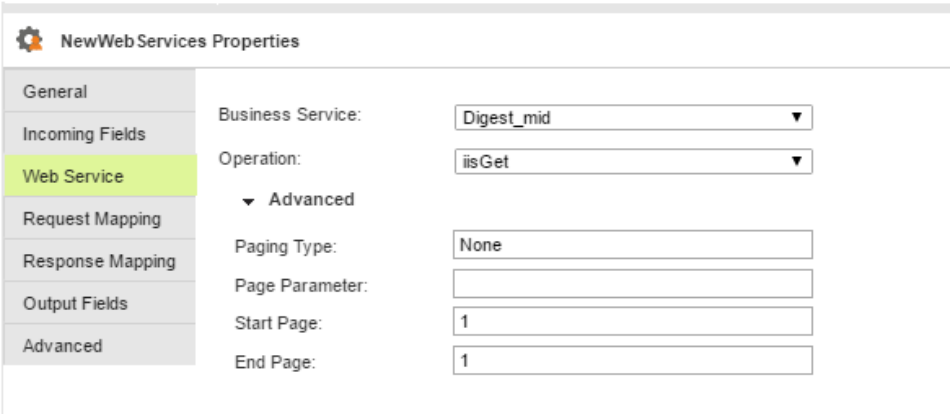


3. REST V2 マッピングを作成します。

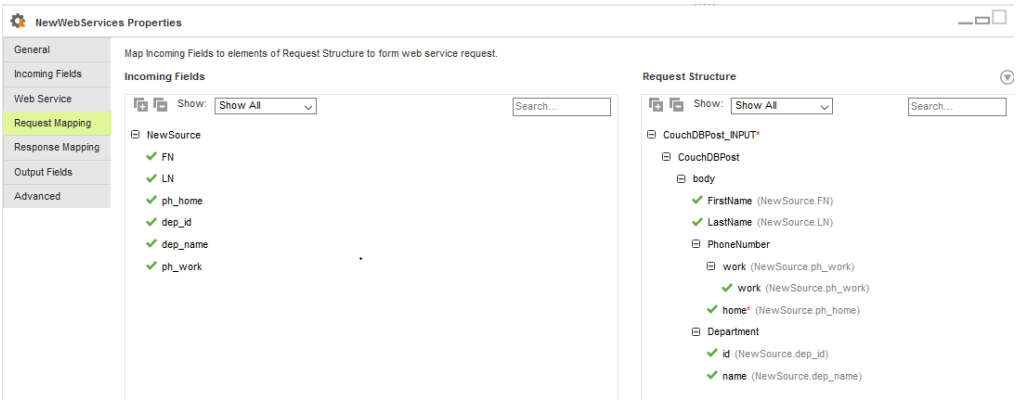


4. Web サービストランスフォーメーションを追加します。全般プロパティで名前と説明を指定します。
5. **［Web サービス］** タブで、設定済みのビジネスサービスと操作を選択します。ページングパラメータも指定します。

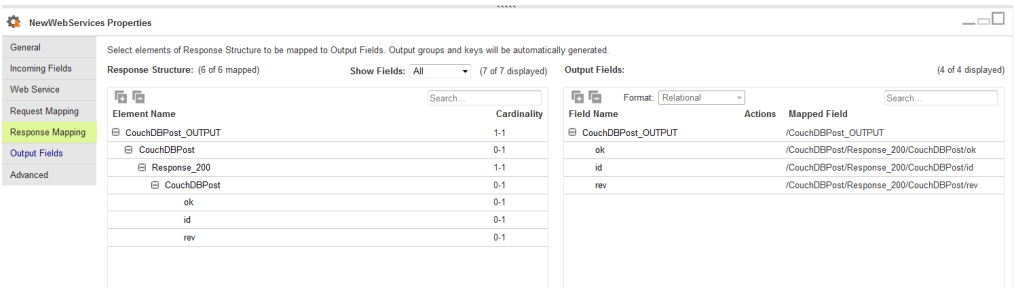
   次の図は、設定済みの Web サービスを示しています。



6. ［**要求マッピング**］タブで、ソースの入力フィールドを、CouchDB の対応するフィールドにマッピングします。



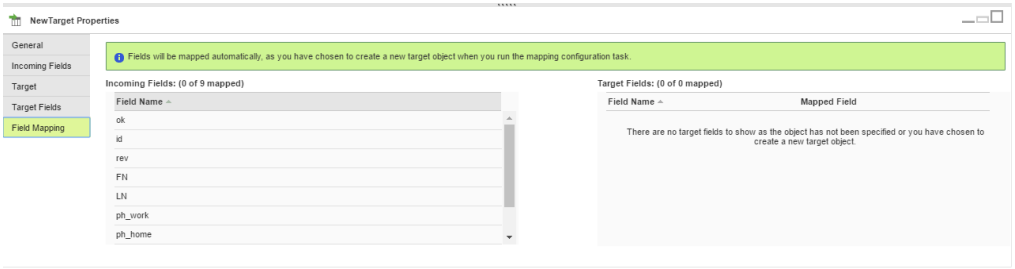
7. ［**応答マッピング**］タブで、［**応答構造**］上のターゲットファイルに書き込む従業員情報フィールドを選択します。



8. ［**詳細**］タブで、キャッシュサイズの詳細設定を行います。

   Web サービストランスフォーメーションの［**応答構造**］から選択したフィールドは、ターゲットオブジェクトの入力フィールドとして表示されます。

9. 必要に応じて、この入力ファイルをフラットファイルフィールドにマッピングします。



10. ［**新規**］ > ［**タスク**］ > ［**マッピングタスク**］の順にクリックして、タスクに対してマッピングを選択します。

11. マッピングタスクを保存して実行すると、Secure Agent は、［要求マッピング］タブでマッピングした従業員についての従業員レコードを CouchDB から取得し、対応するフラットファイルにそのデータを書き込みます。

マッピングにおける REST V2 ターゲット

ターゲットトランスフォーメーション用に REST V2 接続を選択する際、操作を選択できます。操作は、REST V2 接続の設定中に指定された Swagger 仕様ファイルに基づきます。

マッピングの前に、REST ターゲットに複数の入力グループを追加し、その複数の入力グループ間にプライマリキーと外部キーのリレーションを定義できます。

詳細ターゲットプロパティ

マッピングで、REST ターゲットを表すターゲットを設定できます。マッピングで使用する REST ターゲット接続については、マッピングタスクウィザードの **[ターゲット]** ページで詳細プロパティを設定できます。

以下の表に、ターゲットトランスフォーメーションで設定できる詳細プロパティを示します。

プロパティ	説明
ページングタイプ	次のいずれかの値を指定します。 **ページ**。ページングを有効にして、[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] の各プロパティ値を考慮します。 **なし**。[ページパラメータ]、[開始ページ]、[終了ページ]、[応答終了式] のプロパティ値を無視します。
ページパラメータ	ページング操作で使用するパラメータの名前。クエリパラメータまたはパスパラメータを使用できます。 パラメータは整数型で、要求メッセージから抽出されます。
開始ページ	ページ操作の実行範囲の最初のページを示すページ番号。
終了ページ	ページ操作の実行範囲の最後のページを示すページ番号。デフォルトは 10000 です。 ページングは、終了ページに達するか、応答終了式が満たされると停止します。
ページ増分ファクタ	ページパラメータを増分する整数。ページ増分ファクタは、要求ごとにフェッチされるレコード数と同じにする必要があります。ページ増分ファクタオプションが、要求ごとにフェッチされるレコード数と同じでない場合、2 つのコール間でレコードがなかったり重複したりすることが発生します。
応答終了式	ページングを制御する式または文字列を指定します。次のいずれかの動作になります。 - 文字列または式を指定した場合、値がページ応答と一致すると、ページングが停止します。応答終了と一致するページは解析されません。 - 文字列も式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。 - 終了ページと応答終了式の両方を指定すると、先に満たされた条件に応じてページングが停止します。 - 終了ページも応答終了式も指定しない場合、空のページか Null 応答に到達したところでページングが停止します。空のページまたは Null 応答が発生しない場合、ページングはデフォルトの終了ページで停止します。
URL の上書き	Swagger 仕様で指定されている URL を上書きします。URL の上書きではクエリパラメータを使用できません。パスパラメータを [URL の上書き] に含める場合は、パスパラメータを中かっこ{}で囲むようにしてください。例: Swagger 仕様で指定されている URL: `http://invr28pers102:13080/sample/day/20170505?a:b` 20170505 をパス変数として変数名を `path1` と定義すると、URL の上書きにより、`http://invr28pers102:13080/sample/day/{path1}`のようになります。

プロパティ	説明
Web サービス要求のキャッシュサイズ (KB)	Web サービス要求で利用可能なメモリ。Web サービス要求に多数の行またはカラムが含まれている場合は、キャッシュサイズを大きくします。デフォルトは 1024 KB です。
トランザクションコミット制御	トランスフォーメーションを通過する一連のカラムに基づいて、トランザクションのコミットまたはロールバックを制御します。データの量が多く、その処理方法を制御する場合は、トランザクションコミットコントロールを使用します。 リアルタイム処理の **［トランスフォーメーション範囲］** に［トランザクション］を選択する場合は適用されません。
トランスフォーメーション範囲	Secure Agent が受信データに対してトランスフォーメーションロジックを適用する方法です。デフォルトは［すべての入力］です。 リアルタイムソースからのデータを処理するには、トランスフォーメーション範囲に［トランザクション］を選択します。 リアルタイムではないソースの場合、トランスフォーメーション範囲に［すべての入力］を選択します。 1 つの有効なリアルタイムソースのみをリアルタイムターゲットに接続できます。

入力設定プロパティ

［入力設定］ で **［並べ替え済み入力］** を有効にできます。並べ替え済み入力は、入力データが事前に並べ替えられていることを示します。デフォルトでは無効になっています。

パフォーマンスを向上させるために、並べ替え済み入力を有効にします。

注: ［並べ替え済み入力］ が有効になっていて入力が並べ替えられていない場合、Secure Agent は入力を処理せず、マッピングは失敗します。

REST ターゲットマッピングの例

人事部長が、Apache CouchDB 上の従業員情報を更新しようとしています。

フラットファイルから CouchDB 上の従業員情報を更新するには、次の手順を実行します。

1. フラットファイルからデータを読み取るためのフラットファイル接続を作成します。
2. CouchDB にデータを書き込むための REST 接続を作成します。

次の図は、設定された CouchDB 接続を示しています。

View Connection

Edit Done Test

Connection Details

Connection Name:	CouchDB_POST
Description:	
Type:	REST (INFA)
Created On:	Oct 17, 2016 4:50:54 PM
Updated On:	Oct 17, 2016 4:50:54 PM
Created By:	pyn@infa.com
Updated By:	pyn@infa.com

REST Connection Properties

Runtime Environment:	INKW28QA67
Authentication:	GenericRest

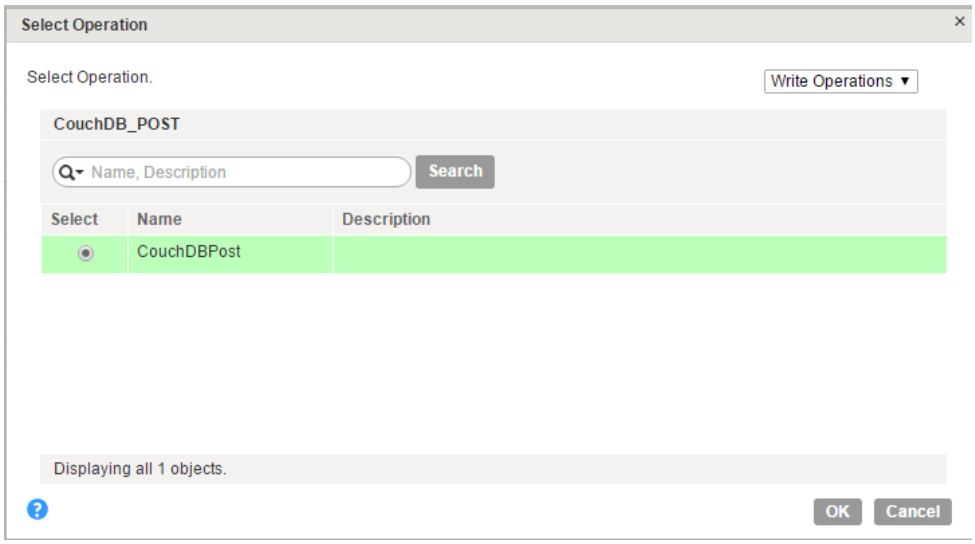
GenericRest Connection Properties

Authentication Type:	NONE
Auth User ID:	
Auth Password:	
OAuth Consumer Key:	
OAuth Consumer Secret:	
OAuth Token:	
OAuth Token Secret:	
Swagger File Path:	http://inv28pers18:28080/swaggerdeploy/2100851911y/CouchDBPost.json

3. CouchDB マッピングを作成します。

4. 従業員情報を含むフラットファイルオブジェクトを含めるためのソーストランスフォーメーションを追加します。フラットファイル接続を追加します。
5. 従業員情報を CouchDB に書き込むためのターゲットトランスフォーメーションを追加します。**[ターゲット]** タブで次のタスクを実行します。
   a. **[接続]** フィールドで、CouchDB に接続するための REST 接続を選択します。
   b. **[操作]** フィールドで、CouchDBPost を操作として選択します。

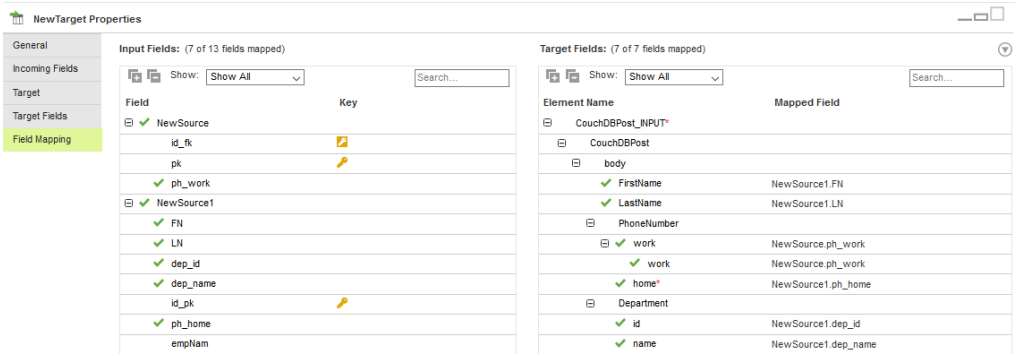
次の図は、書き込み操作一覧に含まれる CouchDBPost を示しています。



c. ［**詳細プロパティ**］セクションで、キャッシュサイズとトランザクションコミット間隔を設定します。

6. ［**フィールドマッピング**］タブで、ターゲットフィールドにマッピングする入力要素を選択します。

   次の図は、入力ファイルと CouchDB ターゲットとの間でマッピングされたすべてのフィールドを示しています。



7. マッピングタスクでマッピングを保存し、実行すると、Secure Agent が CouchDB 上の従業員情報を更新します。

付録 A

Swagger オブジェクト

次の表に、REST V2 コネクタが認める Swagger オブジェクトとフィールド名を示します。

オブジェクトカテゴリ	設定可能なオブジェクトまたはフィールド
Swagger	swagger info host basepath schemes consumes produces paths definitions parameters responses
パス	/{path}
パス項目	$ref get put post delete parameters
操作	summary operationId consumes produces parameters responses schemes 操作 ID 値が定義されていない場合、summary フィールドの値が操作 ID として処理されます。

オブジェクトカテゴリ	設定可能なオブジェクトまたはフィールド
パラメータのタイプ	path query header formData cookie body
パラメータフィールド	name 入力。`in` が `body` である場合、サポートされるフィールドは schema になります。`in` が `body` 以外の値である場合、サポートされるフィールドは type、format、および items になります。 required default。`body` パラメータおよび `securityDefinitions` パラメータには適用されません。
Items	type format items
応答	schema
スキーマ	$ref format タイプ アイテム プロパティ xml
定義	{name}
securityDefinitions	{name} セキュリティ定義タイプ Basic および apiKey を設定できます。
xml	name。wrapped オブジェクトと併用される場合のみ有効です。 attribute prefix namespace wrapped

重要: Rest V2 コネクタは、Swagger 定義で上記以外のオブジェクトとフィールドを処理しません。

索引

C

Cloud Application Integration コミュニティ
　URL 5
Cloud 開発者コミュニティ
　URL 5

I

Informatica Intelligent Cloud Services
　Web サイト 5
Informatica グローバルカスタマサポート
　連絡先情報 6

R

Request Message Editor 25
REST V2
　Swagger オブジェクト 52
　接続プロパティ 14, 21
　　認証
　　標準 14, 21
REST V2 コネクタ
　管理 10
REST V2 コネクタの概要 7

S

swagger 仕様 14

W

Web サイト 5

あ

アップグレード通知 6
アップロード
　ファイル 32

く

クエリパラメータ 14

し

システムステータス 6

す

ステータス
　Informatica Intelligent Cloud Services 6

そ

ソーストランスフォーメーション 27

た

ターゲットトランスフォーメーション、 48
ターゲットプロパティ 48
ターゲットマッピング 49

は

パスパラメータ 14
パック済みフィールド
　フィールドマッピング 27
パラメータ化 26

ふ

フィールドマッピング
　パック済みフィールド 27

へ

ヘッダフィールド 14

ま

マッピングのソーストランスフォーメーション 38

み

ミッドストリームトランスフォーメーションマッピング 45
ミッドストリーム操作 28

め

メンテナンスの停止 6