Informatica®

Informatica® Cloud Data Integration

Snowflake Data Cloud Connector

Informatica Cloud Data Integration Snowflake Data Cloud Connector
2023 年 10 月

発行日: 2023-11-30

目次

序文 7

Informatica のリソース 7

- Informatica マニュアル 7
- Informatica Intelligent Cloud Services Web サイト 7
- Informatica Intelligent Cloud Services コミュニティ 7
- Informatica Intelligent Cloud Services マーケットプレース 8
- データ統合のコネクタのドキュメント 8
- Informatica ナレッジベース 8
- Informatica Intelligent Cloud Services Trust Center 8
- Informatica グローバルカスタマサポート 8

第 1 章 : Snowflake Data Cloud Connector の概要 9

Snowflake Data Cloud Connector アセット 9

第 2 章 : Snowflake Data Cloud への接続 11

認証の準備 11

- 標準 11
- 承認コード 12
- キーペア 12
- クライアント資格情報。 14

Snowflake への接続 14

- 始める前に 14
- 接続の詳細 14
- 認証タイプ 15

JDBC URL パラメータ 21

プロキシサーバーの設定 22

Snowflake にアクセスするためのプライベートリンク 23

第 3 章 : Snowflake Data Cloud のマッピング 24

始める前に 24

- 権限の確認 24
- ポートの確認 25

Snowflake Data Cloud のトランスフォーメーション 25

SQL トランスフォーメーション 25

動的スキーマの処理 27

- 詳細モードのマッピングの動的スキーマの処理 27
- 動的スキーマ処理のルールとガイドライン 28

マッピングの例 28

詳細モードのマッピングの例 30

詳細モードのマッピングのパラメータ化の制限 33

第 4 章 : Snowflake Data Cloud のソース 34
Snowflake Data Cloud のソースプロパティ 34
ソースオブジェクトと操作 36
キー範囲パーティション化 37
SQL のオーバーライド 38

第 5 章 : Snowflake Data Cloud のターゲット 40
Snowflake Data Cloud のターゲットプロパティ 40
ターゲットオブジェクトと操作 43
パススルーのパーティション化 45
ターゲットの指定 45
更新操作のオーバーライド 46
.csv ファイルサイズの最適化 46
バッチサイズとローカルステージングファイルの数の設定 47
マッピングでのプロパティのロードの設定 48
詳細モードのマッピングを実行するための追加のランタイムパラメータの設定 49
Capturing changed data from CDC sources 50
CDC ソースから読み取るためのマッピングタスクの設定 51
リカバリメカニズムの無効化 53
ジョブの統計の表示 53
Snowflake Data Cloud ターゲットトランスフォーメーションのルールとガイドライン 55

第 6 章 : Snowflake Data Cloud のルックアップ 56
Snowflake Data Cloud のルックアッププロパティ 57
パラメータ化 58
複数一致の制限 58
ルックアップキャッシングの有効化 59
キャッシュを使用しないルックアップクエリのロギング 59

第 7 章 : プッシュダウンの最適化 61
プッシュダウンの最適化のタイプ 61
プッシュダウンの最適化のシナリオ 62
Snowflake Data Cloud 接続を使用したプッシュダウンの最適化の準備 62
Google Cloud Storage バケット内のデータファイルへのアクセス 63
Microsoft Azure Data Lake Storage Gen2 コンテナ内のデータファイルへのアクセス 63
プッシュダウンの最適化のプレビュー 65
プッシュダウンの最適化の設定 65
複数のターゲットに対するコンテキストベースの最適化 66
SCD タイプ 2 マージマッピングに関する説明 66
異なるスキーマを持つデータベースへのロジックのプッシュ 67
クロスデータベースプッシュダウンの最適化の設定 67
別のデータベースへのロジックのプッシュ 67

クロスデータベースプッシュダウン最適化の設定. 68
プッシュダウンの最適化ジョブのクリーンな停止. 68

第 8 章 : Snowflake Data Cloud 接続を使用したプッシュダウンの最適化（SQL ELT） 69
Snowflake Data Cloud 接続のプッシュダウンの互換性. 70
Snowflake Data Cloud での関数. 70
Snowflake Data Cloud での演算子. 71
Snowflake Data Cloud での変数. 72
Snowflake Data Cloud でのトランスフォーメーション. 72
アグリゲータトランスフォーメーション. 73
式トランスフォーメーション. 73
階層プロセッサトランスフォーメーション. 74
ルックアップトランスフォーメーション. 74
ノーマライザトランスフォーメーション. 75
ルータートランスフォーメーション. 76
シーケンスジェネレータトランスフォーメーション. 76
SQL トランスフォーメーション. 77
共有体トランスフォーメーション. 78
アップデートストラテジトランスフォーメーション. 78
機能. 78
Snowflake Data Cloud のソース、ターゲット、およびルックアップ. 78
Amazon S3 V2 ソース. 82
Google Cloud Storage V2 ソース. 83
Microsoft Azure Data Lake Storage Gen2 ソース. 83
ソースオーバーライドの一時ビューの作成. 83
ターゲットコピーコマンドオプションの設定. 84
セッションログのプッシュダウンクエリの検証. 84
トラブルシューティング 85

第 9 章 : マッピングの移行 86
移行の計画. 86
同じパス内でのマッピングの移行. 86
別のパスへのマッピングの移行. 87
移行オプション. 87
移行の制限. 88
複数のソースオブジェクトを含むマッピングの移行. 88
詳細フィルタとテーブル名のオーバーライドを含むマッピングの移行. 88
SQL オーバーライドとカスタムクエリを含むマッピングの移行. 88

付録 A : データ型リファレンス 90
Snowflake Data Cloud およびトランスフォーメーションデータ型. 90
半構造化データ型とトランスフォーメーションデータ型. 91

データ型のルールとガイドライン. 92

付録 B : 追加のランタイム設定 93

JVM メモリ要件の設定. 93
一括処理の設定. 93
ロギングプロパティの設定. 94
マッピングのための一時ディレクトリの設定. 94
マッピングのステージングパフォーマンスの最適化. 95

付録 C : Snowflake Data Cloud Connector へのアップグレード 97

接続切り替えの例. 98
切り替え後に保持されるプロパティ. 100
ルールおよびガイドライン. 101

索引 102

序文

Snowflake Data Cloud Connector のガイドを使用して、Snowflake との間で読み書きを行う方法を学びます。データ統合で接続を作成する方法や、マッピング、マッピングタスク、動的マッピングタスク、およびデータ転送タスクを開発および実行する方法についても確認します。処理用のトランスフォーメーションロジックを Snowflake データベースにプッシュダウンする方法についても説明します。

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/

データ統合のコネクタのドキュメント

データ統合の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。

サブスクライブするには、Informatica Intelligent Cloud Services Status ページで **[サブスクライブして更新]** をクリックします。電子メール、SMS テキストメッセージ、Webhook、RSS フィード、またはこれらの 4 つの任意の組み合わせとして送信された通知を受信するという選択ができます。

Informatica グローバルカスタマサポート

グローバルサポートセンターには、Informatica Network または電話でお問い合わせください。

Informatica Network でオンラインサポートリソースを検索するには、Informatica Intelligent Cloud Services のヘルプメニューで **[サポートにお問い合わせください]** をクリックして、**Cloud Support** ページに移動します。**Cloud Support** ページには、システムステータス情報とコミュニティディスカッションが記載されています。追加のリソースを検索する場合や電子メールで Informatica グローバルカスタマサポートに問い合わせる場合は、Informatica Network にログインし、**[サポートが必要な場合]** をクリックしてください。

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

Snowflake Data Cloud Connector の概要

Snowflake 外部テーブル、ハイブリッドテーブル、マテリアライズドビューなどの Snowflake テーブルとビューから読み取りを行うことができます。また、Snowflake のテーブルやビューに書き込むこともできます。さらに、Apache Iceberg テーブルに対して読み取りと書き込みを行うこともできます。

さらに、他のアプリケーション、データベース、およびフラットファイルからデータを読み取り、Snowflake にデータを書き込むことができます。また、Snowflake Data Cloud Connector を使用して、Azure、Amazon、Google Cloud Platform、または Snowflake GovCloud のステージングデータで有効になっている Snowflake に対して、データの読み取りおよび書き込みを行うことも可能です。

Snowflake Data Cloud Connector を使用すると、Snowflake Data Cloud 接続を作成し、その接続をデータ統合マッピングおよびタスクで使用できます。

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。

Snowflake Data Cloud マッピングまたはタスクを実行すると、ワークフローと Snowflake Data Cloud 接続設定に基づいて、Secure Agent によって Snowflake にデータが書き込まれます。

詳細クラスタは、Amazon Web Services、Google Cloud Platform、Microsoft Azure 環境、またはセルフサービスクラスタでホストされます。

プレビュー通知: Apache Iceberg テーブルに対する読み取りおよび書き込みのサポートはプレビューで利用できます。プレビュー機能は評価を目的としてサポートされていますが、保証対象外で、本番環境または本番環境にプッシュする予定の環境には対応していません。Informatica は、本番環境用の今後のリリースにプレビュー機能を含める予定ですが、市場や技術的な状況の変化に応じて導入を行わない場合もあります。プレビュー POD で作業している場合は、すべてのデータが SOC 2 コンプライアンスカバレッジから除外されることに注意してください。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。

Snowflake Data Cloud Connector アセット

データ統合でアセットを作成し、Snowflake Data Cloud Connector を使用してデータを統合します。

Snowflake Data Cloud Connector を使用する場合は、次のようなデータ統合アセットを含めることができます。

- データ転送タスク
- 動的マッピングタスク
- マッピング

- マッピングタスク

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。

第 2 章

Snowflake Data Cloud への接続

Snowflake に対して安全なデータの読み取りまたは書き込みを行うには、Snowflake Data Cloud 接続を作成します。Snowflake Data Cloud 接続を使用して、マッピングおよびマッピングタスクでソース、ターゲット、およびルックアップを指定できます。Snowflake 接続を SQL トランスフォーメーションで使用することもできます。

認証の準備

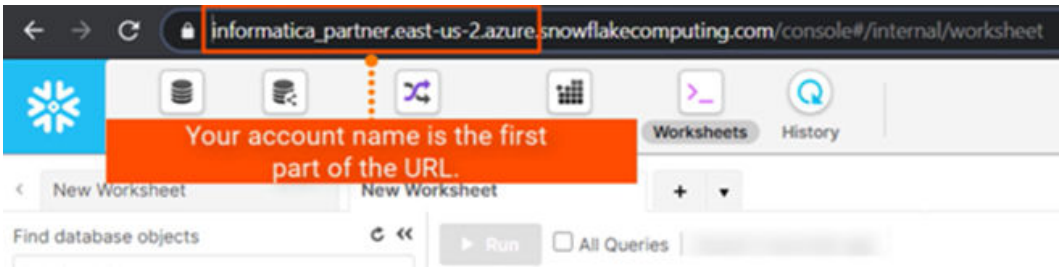
Snowflake にアクセスするための認証タイプ（標準、認証コード、キーペア、およびクライアント資格情報）を設定できます。

接続プロパティを設定する前に、使用する認証のタイプに基づいて認証の詳細を用意しておく必要があります。

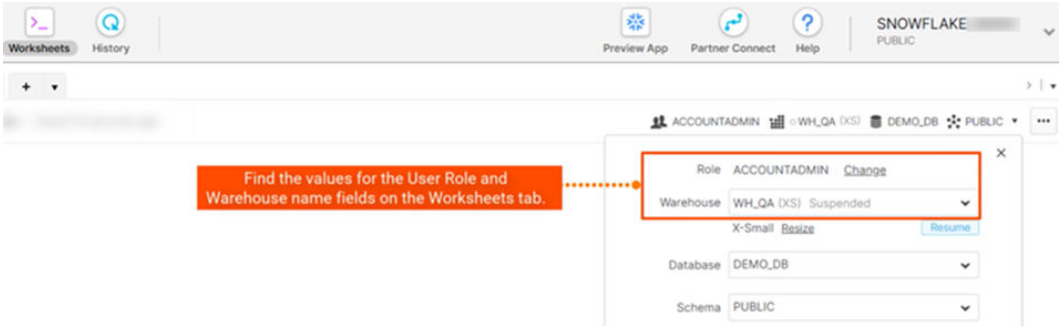
標準

標準認証を使用して Snowflake に接続するには、Snowflake アカウントのユーザー名とパスワードが必要です。

Snowflake アカウント名、ウェアハウス、ロールの詳細などの必要な詳細を Snowflake アカウントから取得してみましょう。



右側に、ウェアハウスとロールの詳細が表示されます。



承認コード

OAuth 2.0 認証コードを使用して Snowflake に接続するには、Snowflake クライアント ID、認証 URL、アクセストークン URL、およびアクセストークンが必要です。

最初に Snowflake で認証統合を作成してから、認証の詳細を取得する必要があります。

認証コードを使用するには、最初に Informatica リダイレクト URL をセキュリティ統合に登録する必要があります。セキュリティ統合とは、OAuth をサポートするクライアントがユーザーを認証ページにリダイレクトし、Snowflake にアクセスするためのアクセストークン、および必要に応じて更新トークンを生成できるようにする統合の一種です。

Informatica リダイレクト URL をセキュリティ統合に登録します。

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

アクセストークンが期限切れになると、顧客のファイアウォール外にある Informatica リダイレクト URL は、エンドポイントへの接続と新しいアクセストークンの取得を試みます。

認証の詳細を取得する方法に関する詳細については、Snowflake のドキュメントを参照してください。

注: データ統合の詳細モードで設定されたマッピングに対して認証コードの認証を設定することはできません。

キーペア

キーペア認証を使用して Snowflake に接続するには、Snowflake アカウントのユーザー名とともに、プライベートキーファイルとプライベートキーファイルのパスワードが必要です。

OpenSSL を使用してパブリックキーとプライベートキーのペアを生成します。キーペア認証方法には 2048 ビットの RSA キーペアが必要です。Snowflake にアクセスするには、接続プロパティでプライベートキーファイルへのパスおよびパスワードを指定します。

パブリックキーとプライベートキーの生成

キーペア認証用のパブリックキーとプライベートキーを生成するには、Snowflake で SecurityAdmin 以上のロールが必要です。

1. Open SSL コマンドラインから、プライベートキーを生成します。
   - 暗号化されていないプライベートキーを生成するには、次のコマンドを実行し、表示されたプロンプトに従ってパスフレーズを入力します。

     ```
     $ openssl genrsa 2048 | openssl pkcs8 -topk8 -inform PEM -out rsa_key.p8 -nocrypt
     ```

- 暗号化されているプライベートキーを生成するには、次のコマンドを実行し、表示されたプロンプトに従ってパスフレーズを入力します。

  ```
  $ openssl genrsa 2048 | openssl pkcs8 -topk8 -inform PEM -out rsa_key.p8
  ```

  パスフレーズは、Snowflake への接続中に、プライベートキーファイルを暗号化するために使用されます。

2. パブリックキーを生成します。次のコマンドを実行し、ファイルにある rsa_key.p8 などの暗号化されたプライベートキーを指定します:

   ```
   openssl rsa -in rsa_key.p8 -pubout -out rsa_key.pub
   ```

3. Secure Agent がアクセスできるディレクトリにパブリックキーファイルとプライベートキーファイルをコピーします。例: C:\Program Files\Informatica Cloud Secure Agent\apps\Data_Integration_Server\data\snowflake\rsa_key.p8
   Snowflake 接続を設定するときにパスの詳細が必要になります。

4. Snowflake で、ALTER USER コマンドを使用してパブリックキーを Snowflake ユーザーに割り当てます。

   ```
   alter user <user> set rsa_public_key='<content of the public key
   after removing the header and footer lines>';
   ```

   例えば、`alter user jsmith set rsa_public_key='MIIXBIjABCdef...';`

Snowflake 用のキーペア認証の設定の詳細については、Snowflake のマニュアルを参照してください。

詳細クラスタでのプライベートキーの設定

OpenSSL を使用してパブリックキーとプライベートキーのペアを生成した後に、詳細モードのマッピングで接続が機能するように特定のタスクを追加で実行する必要があります。

詳細クラスタで設定した接続を使用してマッピングを実行する前に、マッピングタスクでクラスタアプリケーションのプロパティを設定します。

次のリストに、マッピングタスクの詳細セッションプロパティで設定する必要があるプロパティを示します。
Spark.NeedUserCredentialFileForAdapter=true

Spark.UserCredentialDirOnDIS に指定した場所のプライベートキーの内容を、Agent マシンから Spark ドライバおよびエグゼキュータにコピーします。クラスタアプリケーションにコピーするシークレットキーコンテンツの資格情報ファイルは、1 MB 以下である必要があります。資格情報ファイルを含むフォルダには、1MB の制限はありません。この値は、true または false に設定できます。デフォルトは false です。

このフラグを設定していない場合、またはこのフラグを false に設定した場合、プライベートキーファイルはクラスタアプリケーションにコピーされず、マッピングは失敗します。

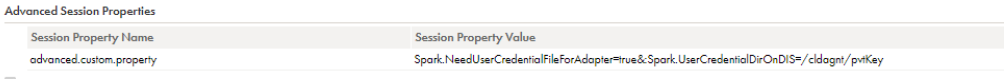
Spark.UserCredentialDirOnDIS=<**プライベートキーファイルディレクトリ**>

プライベートキーを含むデフォルトの Agent ディレクトリを、プライベートキーの内容をクラスタアプリケーションにコピーするために指定したディレクトリでオーバーライドします。デフォルトのディレクトリは/infa/user/credentials です。ディレクトリにプライベートキーファイルの名前を含めることはできません。

このフラグを設定しない場合は、デフォルトの場所が使用されます。デフォルトの場所を使用するには、/infa/user/credentials ディレクトリを Agent マシンに作成し、そこにプライベートキーファイルをコピーします。

マッピングタスクの詳細セッションプロパティの場所をオーバーライドするフラグを設定する場合、*Spark.UserCredentialDirOnDIS* に指定したオーバーライドの場所にプライベートキーファイルが含まれていることを確認します。オーバーライドの場所とプライベートキーファイルに書き込み権限があることを確認します。

次の図に、マッピングタスクの詳細セッションプロパティに設定されるプロパティを示します。



クライアント資格情報。

OAuth 2.0 クライアント資格情報を使用して Snowflake に接続するには、Snowflake クライアント ID、アクセストークン URL、クライアントシークレット、スコープ、およびアクセストークンが必要です。

最初にクライアント資格情報付与タイプを使用して OAuth エンドポイントを設定し、次にセキュリティ統合を作成して認証の詳細を取得する必要があります。

クライアント資格情報認証を使用して Snowflake に接続する前に、組織の管理者は前提条件のタスクを実行する必要があります。

1. Snowflake で使用する OAuth と互換性のあるクライアントアプリケーションを作成します。
2. クライアント資格情報付与タイプを使用して認証サーバーを設定します。
3. Snowflake で OAuth タイプのセキュリティ統合を作成します。
   セキュリティ統合の作成の詳細については、Snowflake のドキュメントの「Create security integration for external OAuth」を参照してください。

注: データ統合の詳細モードで設定されたマッピングでは、クライアント資格情報認証を設定できません。

Snowflake への接続

Snowflake に接続するための Snowflake Data Cloud 接続プロパティを設定してみましょう。

始める前に

開始する前に、設定する認証タイプに基づいて Snowflake アカウントから情報を取得する必要があります。

認証の前提条件に関する詳細については、「「認証の準備」 (ページ 11)」を参照してください。

接続の詳細

次の表に、基本的な接続プロパティを示します。

プロパティ	説明
接続名	接続の名前。 各接続名は組織内で一意である必要があります。接続名には、英数字、スペース、および次の特殊文字を含めることができます。_ .+ -, 最大長は 255 文字です。
説明	接続の説明。最大長は 4000 文字です。

プロパティ	説明
タイプ	Snowflake Data Cloud
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できます。 サーバーレス環境の設定および使用方法の詳細については、Administrator のヘルプにある「*ランタイム環境*」の「サーバーレスランタイム環境のセットアップ」を参照してください。 詳細モードのマッピングで接続を使用する場合は、ホステッドエージェントを使用しないでください。

認証タイプ

Snowflake にアクセスするための認証タイプ（標準、認証コード、キーペア、およびクライアント資格情報）を設定できます。

必要な認証方法を選択し、認証固有のパラメータを設定します。

標準認証

標準認証はデフォルトのタイプで、少なくとも Snowflake アカウント名とパスワードが必要です。

次の表に、標準認証の基本的な接続プロパティに関する説明を示します。

プロパティ	説明
ユーザー名	Snowflake アカウントに接続するためのユーザー名。
パスワード	Snowflake アカウントに接続するためのパスワード。
アカウント	Snowflake アカウントの名前。 例えば、Snowflake の URL が https://<123abc>.us-east-2.aws.snowflakecomputing.com/console/login#/の場合、アカウント名は URL の最初のセグメント（snowflakecomputing.com より前）です。ここでは、123abc.us-east-2.aws がアカウント名です。 Snowsight の URL を使用する場合、例えば https://app.snowflake.com/us-east-2.aws/<123abc>/dashboard では、アカウント名は 123abc.us-east-2.aws です。 **注**: アカウント名にアンダースコアが含まれていないことを確認します。エイリアス名を使用するには、Snowflake カスタマーサポートにお問い合わせください。
ウェアハウス	Snowflake ウェアハウス名。

詳細設定

次の表に、標準認証の詳細な接続プロパティに関する説明を示します。

プロパティ	説明
ロール	ユーザーに割り当てられた Snowflake ロール。

追加の JDBC URL パラメータ	追加の JDBC 接続パラメータ。
	複数の JDBC 接続パラメータをアンパサンド（&）で区切って、次の形式で指定できます。
	`<param1>=<value>&<param2>=<value>&<param3>=<value>....`
	例えば、Snowflake に接続するときに次のデータベースとスキーマの値を渡すことができます。
	`db=mydb&schema=public`
	パラメータを追加するときは、等号（=）の前後にスペースを加えないようにしてください。
	設定可能な追加の JDBC パラメータのリストについては、「JDBC URL パラメータ」 (ページ 21)」を参照してください。

認証コードの認証

OAuth 2.0 認証では、Snowflake に接続するために、認証コード付与タイプを備えた OAuth 2.0 プロトコルが必要です。認証コードを使用すると、ログイン資格情報を共有または保存することなく、Snowflake への認証されたアクセスが可能になります。

次の表に、OAuth 2.0 認証コードの認証の基本的な接続プロパティに関する説明を示します。

プロパティ	説明
アカウント	Snowflake アカウントの名前。 例えば、Snowflake の URL が https://<123abc>.us-east-2.aws.snowflakecomputing.com/console/login#/の場合、アカウント名は URL の最初のセグメント（snowflakecomputing.com より前）です。ここでは、123abc.us-east-2.aws がアカウント名です。 Snowsight の URL を使用する場合、例えば https://app.snowflake.com/us-east-2.aws/<123abc>/dashboard では、アカウント名は 123abc.us-east-2.aws です。 **注**: アカウント名にアンダースコアが含まれていないことを確認します。エイリアス名を使用するには、Snowflake カスタマーサポートにお問い合わせください。
ウェアハウス	Snowflake ウェアハウス名。

プロパティ	説明
認証 URL	ユーザー要求を承認するために使用する Snowflake サーバーのエンドポイント。 認証 URL は、https://<アカウント名>.snowflakecomputing.com/oauth/authorize です。この<アカウント名>には、Snowflake が提供するアカウントの完全な名前を指定します。 例: https://<abc>.snowflakecomputing.com/oauth/authorize **注**: アカウント名にアンダースコアが含まれている場合は、エイリアス名を使用します。 また、仮想プライベートクラウドネットワークで認証サーバーをサポートする認証コード付与タイプを使用することもできます。
アクセストークン URL	アクセストークンの認証コードを交換するために使用する Snowflake アクセストークンのエンドポイント。 アクセストークンの URL は、https://<アカウント名>.snowflakecomputing.com/oauth/token-request です。この<アカウント名>には、Snowflake が提供するアカウントの完全な名前を指定します。 例: https://<abc>.snowflakecomputing.com/oauth/token-request **注**: アカウント名にアンダースコアが含まれている場合は、エイリアス名を使用します。
クライアント ID	Snowflake で OAuth タイプのセキュリティ統合を作成するときに生成されるアプリケーションのクライアント ID。 詳細については、Snowflake のドキュメントを参照してください。
クライアントシークレット	クライアント ID に対して生成されたクライアントシークレット。
アクセストークン	アクセストークンの値。 取り込まれたアクセストークンの値を入力するか、**[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取り込みます。

詳細設定

次の表に、OAuth 2.0 認証コードの認証の詳細接続プロパティに関する説明を示します。

プロパティ	説明

追加の JDBC URL パラメータ	追加の JDBC 接続パラメータ。
	複数の JDBC 接続パラメータをアンパサンド（&）で区切って、次の形式で指定できます。
	`<param1>=<value>&<param2>=<value>&<param3>=<value>....`
	例えば、Snowflake に接続するときに次のデータベースとスキーマの値を渡すことができます。
	`db=mydb&schema=public`
	パラメータを追加するときは、等号（=）の前後にスペースを加えないようにしてください。
	設定可能な追加の JDBC パラメータのリストについては、「JDBC URL パラメータ」 (ページ 21)」を参照してください。

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、アクセス制御を決定します。 例えば、デフォルトのユーザーロールの値を上書きするスコープとして、`session:role:CQA_GCP` を指定します。この値は、Security Integration で割り当てたロールの 1 つである必要があります。 複数のスコープ属性をスペースで区切って入力します。
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 形式で定義します。 例えば、次のようなパラメータを定義します。 `[{"Name":"code_verifier","Value":"5PMddu6Zcg6Tc4sbg"}]`
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 形式で定義します。 例えば、次のようなパラメータを定義します。 `[{"Name":"code_challenge","Value":"Ikr-vv52th0UeVRi4"},{"Name":"code_challenge_method","Value":"S256"}]`
更新トークン	更新トークンの値。 取り込まれた更新トークンの値を入力するか、**[トークンの生成]** をクリックして、更新トークンの値を取り込みます。アクセストークンが有効でないか、有効期限切れの場合、エージェントは、更新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。 **注**: 更新トークンが期限切れの場合は、有効な更新トークンを指定するか、**[トークンの生成]** をクリックして新しい更新トークンを再生成します。

キーペア認証

キーペア認証には、プライベートキーファイルとプライベートキーファイルのパスワード、および Snowflake に接続するための Snowflake アカウントのユーザー名が必要です。

次の表に、キーペア認証の基本的な接続プロパティに関する説明を示します。

プロパティ	説明
ユーザー名	Snowflake アカウントに接続するためのユーザー名。
アカウント	Snowflake アカウントの名前。 例えば、Snowflake の URL が https://<123abc>.us-east-2.aws.snowflakecomputing.com/console/login#/ の場合、アカウント名は URL の最初のセグメント（snowflakecomputing.com より前）です。ここでは、123abc.us-east-2.aws がアカウント名です。 Snowsight の URL を使用する場合、例えば https://app.snowflake.com/us-east-2.aws/<123abc>/dashboard では、アカウント名は 123abc.us-east-2.aws です。 **注**: アカウント名にアンダースコアが含まれていないことを確認します。エイリアス名を使用するには、Snowflake カスタマーサポートにお問い合わせください。
ウェアハウス	Snowflake ウェアハウス名。
プライベートキーファイル	プライベートキーファイル名を含む、Secure Agent が Snowflake にアクセスするために使用するプライベートキーファイルへのパス。 例えば、次のパスとキーファイル名を指定します。 - Windows の場合: `C:\Users\path_to_key_file\rsa_key.p8` - Linux の場合: `/export/home/user/path_to_key_file/rsa_key.p8` **注**: キーストアが FIPS 認証されていることを確認します。

詳細設定

次の表に、キーペア認証の詳細な接続プロパティに関する説明を示します。

プロパティ	説明

追加の JDBC URL パラメータ	追加の JDBC 接続パラメータ。
	複数の JDBC 接続パラメータをアンパサンド（&）で区切って、次の形式で指定できます。
	`<param1>=<value>&<param2>=<value>&<param3>=<value>....`
	例えば、Snowflake に接続するときに次のデータベースとスキーマの値を渡すことができます。
	`db=mydb&schema=public`
	パラメータを追加するときは、等号（=）の前後にスペースを加えないようにしてください。
	設定可能な追加の JDBC パラメータのリストについては、「JDBC URL パラメータ」 (ページ 21)」を参照してください。

プライベートキーファイルのパスワード	プライベートキーファイルのパスワード。

クライアント資格情報認証

OAuth 2.0 クライアント資格情報認証には、少なくともクライアント ID、アクセストークン URL、クライアントシークレット、スコープ、およびアクセストークンが必要です。

次の表に、OAuth 2.0 クライアント資格情報認証の基本的な接続プロパティに関する説明を示します。

プロパティ	説明
アカウント	Snowflake アカウントの名前。 例えば、Snowflake の URL が https://<123abc>.us-east-2.aws.snowflakecomputing.com/console/login#/である場合、URL の最初のセグメント（snowflakecomputing.com より前）がアカウント名です。ここでは、123abc.us-east-2.aws がアカウント名です。 Snowsight の URL を使用する場合、例えば https://app.snowflake.com/us-east-2.aws/<123abc>/dashboard では、アカウント名は 123abc.us-east-2.aws です。 **注**: アカウント名にアンダースコアが含まれていないことを確認します。エイリアス名を使用するには、Snowflake カスタマーサポートにお問い合わせください。
ウェアハウス	Snowflake ウェアハウス名。
アクセストークン URL	アクセストークンの認証コードを交換するために使用する Snowflake アクセストークンのエンドポイント。 OAuth エンドポイントから取得したアクセストークン URL を指定します。
クライアント ID	アプリケーションを OAuth 用に設定したときに生成されるアプリケーションのクライアント ID。 詳細については、Snowflake のドキュメントを参照してください。
クライアントシークレット	クライアント ID に対して生成された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、アクセス制御を決定します。 例えば、デフォルトのユーザーロールの値を上書きするスコープとして、`session:role:CQA_GCP` を指定します。この値は、Security Integration で割り当てたロールの 1 つである必要があります。 複数のスコープ属性をスペースで区切って入力します。
アクセストークン	アクセストークンの値。 取り込まれた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取り込みます。
アクセストークンの生成	指定した OAuth 属性に基づいてアクセストークンと更新トークンを生成します。

詳細設定

次の表に、OAuth 2.0 クライアント資格情報認証の詳細接続プロパティに関する説明を示します。

プロパティ	説明
追加の JDBC URL パラメータ	追加の JDBC 接続パラメータ。 複数の JDBC 接続パラメータをアンパサンド（&）で区切って、次の形式で指定できます。 <param1>=<value>&<param2>=<value>&<param3>=<value>.... 例えば、Snowflake に接続するときに次のデータベースとスキーマの値を渡すことができます。 db=mydb&schema=public パラメータを追加するときは、等号（=）の前後にスペースを加えないようにしてください。 設定可能な追加の JDBC パラメータのリストについては、「「JDBC URL パラメータ」 (ページ 21)」を参照してください。
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 形式で定義します。 例えば、次のようなコード検証パラメータを定義できます。 [{"Name":"code_verifier","Value":"5PMddu6Zcg6Tc4sbg"}]

JDBC URL パラメータ

Snowflake Data Cloud 接続の追加の JDBC URL パラメータフィールドを利用して、Snowflake への接続に必要な追加パラメータをカスタマイズおよび設定できます。

入力された設定は認証済みのものですが、このフィールドを柔軟に利用することで、特定の要件に応じてパラメータをカスタマイズできます。

Snowflake Data Cloud 接続の追加の JDBC URL パラメータでは、次のようなプロパティを設定できます。

- Snowflake で一時テーブルを作成するために使用されるデータベースとスキーマ名をオーバーライドするには、次の形式でデータベース名とスキーマ名を入力します。

  ProcessConnDB=<DB name>&ProcessConnSchema=<schema_name>
- Snowflake テーブルのインポート中に指定したデータベースおよびスキーマのみを表示するには、次の形式でデータベースおよびスキーマ名を入力します。

  db=<database_name>&schema=<schema_name>
- Snowflake から UDF 文字列と数値データを読み取るには、Snowflake で UDF が作成されたデータベースとスキーマを次の形式で入力します。

  db=<database_name>&schema=<schema_name>
- Okta SSO 認証を介して Snowflake にアクセスするには、次の形式で SAML 2.0 プロトコルを実装する Web ベースの IdP を入力します。

  authenticator=https://<Your_Okta_Account_Name>.okta.com

  注: Microsoft ADFS は使用できません。

Okta 認証の設定の詳細については、「Configuring Snowflake to use federated authentication」を参照してください。

- プッシュダウンの最適化のために Google Cloud Storage または Microsoft Azure Data Lake Storage Gen2 から Snowflake にデータをロードするには、Snowflake でこれらに対して作成したクラウドストレージ統合の名前を次の形式で入力します。

  ```
  storage_integration=<Storage Integration name>
  ```

  ストレージ統合の名前は大文字と小文字が区別されます。例えば、Snowflake で Google Cloud Storage または Microsoft Azure Data Lake Storage Gen2 用に作成したストレージ統合の名前が *STORAGE_INT* である場合は、同じ統合名を指定する必要があります:

  ```
  storage_integration=STORAGE_INT
  ```

- プロキシサーバーを使用して Snowflake に接続するには、次のパラメータを入力します。

  ```
  useProxy=true&
  proxyHost=<proxy host IP address>&
  proxyPort=<proxy server port number>&
  proxyUser=<proxy server user name>&
  proxyPassword=<proxy server password>
  ```

- テーブル内の二重引用符を無視し、大文字と小文字を区別しないものとしてすべてのテーブルを処理するには、次のパラメータを入力します。

  ```
  QUOTED_IDENTIFIERS_IGNORE_CASE=true
  ```

  接続のこのプロパティを true に設定すると、Snowflake はテーブル内の二重引用符を無視し、大文字と小文字を区別しないものとしてすべてのテーブルを処理します。このプロパティを true に設定すると、Snowflake によりクエリ内の二重引用符が無視されるため、既存の大文字と小文字を区別するテーブルを同じ接続を使用してクエリすることができなくなります。大文字と小文字を区別する既存のテーブルを取得するには、新しい接続を作成する必要があります。

プロキシサーバーの設定

組織で送信プロキシサーバーを使用してインターネットに接続している場合、Secure Agent は、そのプロキシサーバー経由で Informatica Intelligent Cloud Services に接続します。

Windows および Linux 上でプロキシサーバーを使用するように、Secure Agent とサーバーレスランタイム環境を設定できます。認証されていないプロキシサーバーまたは認証されたプロキシサーバーを使用できます。プロキシ設定は、マッピングおよび詳細モードのマッピングで使用される接続に適用されます。

Secure Agent のプロキシの設定を行うには、次のいずれかの方法を使用します。

- Windows の場合は Secure Agent Manager を使用し、Linux の場合はシェルコマンドを使用して Secure Agent を設定します。

  手順については、データ統合のヘルプの『*基本操作*』にある、トピック「Windows でのプロキシ設定」または「Linux でのプロキシ設定」を参照してください。

- Secure Agent のプロパティで、DTM の JVM オプションを設定します。手順については、ナレッジベースの記事「Proxy server settings」を参照してください。

- Snowflake 接続の追加の JDBC URL パラメータで、プロキシサーバーのプロパティを設定します。詳細については、「Set JDBC URL Parameters」を参照してください。

サーバーレスランタイム環境のプロキシの設定を行うには、Administrator のヘルプの「*ランタイム環境*」にある「プロキシサーバーの使用」を参照してください。

Snowflake にアクセスするためのプライベートリンク

AWS または Azure プライベートリンクエンドポイントを使用して Snowflake にアクセスできます。Snowflake Data Cloud 接続の作成時に、Snowflake プライベートリンクアカウント名を指定します。

AWS または Azure プライベートリンクの設定によって、Snowflake への接続が AWS または Azure 内部ネットワークを使用して確立され、パブリックインターネットを介した接続が行われないようになります。

プライベート AWS ネットワーク経由で Snowflake アカウントに接続するには、「AWS Private Link and Snowflake」を参照してください。

プライベート Azure ネットワーク経由で Snowflake アカウントに接続するには、「Azure Private Link and Snowflake」を参照してください。

第 3 章

Snowflake Data Cloud のマッピング

マッピングを設定するときに、ソースからターゲットへのデータフローを記述します。

マッピングによって、マッピングタスクで使用できる再利用可能なデータフローロジックを定義します。

マッピングを作成するときに、Snowflake Data Cloud オブジェクトを表すために、ソース、ターゲット、およびルックアップトランスフォーメーションを定義します。データ統合の Mapping Designer を使用して、マッピングキャンバスにソース、ターゲット、またはルックアップトランスフォーメーションを追加し、Snowflake Data Cloud のソース、ターゲット、およびルックアップのプロパティを設定します。

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。

Monitor を使用してジョブを監視できます。

始める前に

マッピングの設定と実行を開始する前に、必要な前提条件を完了している必要があります。

権限の確認

権限により、Snowflake で実行できる操作のアクセスレベルが定義されます。

次の表に、Snowflake アカウントに必要な権限を示します。

オブジェクトタイプ	特権
データベース	使用方法
スキーマ	使用方法、テーブルの作成、ビューの作成、プロシージャの作成、シーケンスの作成
テーブル	すべて
シーケンス	すべて
Stored Procedure	すべて

詳細については、Snowflake のドキュメントにある Access Control Privileges を参照してください。

ポートの確認

Snowflake のポート 443 と 80 が開いていて、アクセス可能であることを確認します。

Snowflake Data Cloud のトランスフォーメーション

マッピングにトランスフォーメーションを追加して、データに対して実行する操作を表します。

次のようなトランスフォーメーションを追加して、マッピングでトランスフォーメーションの詳細を定義できます。

ソーストランスフォーメーション

Snowflake または他のデータソースからデータを読み取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。単一または複数の Snowflake テーブルから読み取ることができます。クエリまたはパラメータをソースタイプとして使用し、Snowflake から読み取ることもできます。

1 つのマッピングで 1 つ以上のソーストランスフォーメーションを使用できます。単一のトランスフォーメーションを使用して複数のテーブルから読み取る場合は、結合を使用できます。マッピングで 2 つのソーストランスフォーメーションを使用する場合、ジョイナトランスフォーメーションを使用してデータを結合します。

ターゲットトランスフォーメーション

Snowflake にデータを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。マッピングでは、1 つ以上のターゲットトランスフォーメーションを使用できます。ターゲットトランスフォーメーションを設定するときは、単一の Snowflake ターゲットオブジェクトを使用できます。新規または既存の Snowflake ターゲットオブジェクトを使用できます。マッピングタスクを実行するときは、パラメータを使用してターゲット接続やターゲットオブジェクトを定義することもできます。

ルックアップトランスフォーメーション

指定されたルックアップ条件に基づいてデータを取得するには、ルックアップトランスフォーメーションを追加します。ルックアップオブジェクトは、単一のオブジェクトまたはクエリです。ルックアップオブジェクトと接続をパラメータ化することもできます。

マッピングまたは詳細モードのマッピングで、これらのトランスフォーメーションにそれぞれ設定できるオブジェクトとプロパティの詳細については、第 4 章, 「Snowflake Data Cloud のソース」 (ページ 34)、第 5 章, 「Snowflake Data Cloud のターゲット」 (ページ 40)、第 6 章, 「Snowflake Data Cloud のルックアップ」 (ページ 56)の章を参照してください。

マッピングに他のトランスフォーメーションを追加して、データを変換することもできます。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にある「*トランスフォーメーション*」を参照してください。

SQL トランスフォーメーション

Snowflake で SQL クエリとストアドプロシージャを処理するため、Snowflake Data Cloud マッピングで SQL トランスフォーメーションを設定できます。

SQL トランスフォーメーションをマッピングに追加する場合は、**[SQL]** タブで、データベース接続およびトランスフォーメーションが処理する SQL のタイプを定義します。

SQL トランスフォーメーションでパラメータ化された接続を使用することを選択できます。実行時にパラメータファイルで定義された値を上書きすることもできます。SQL トランスフォーメーションでパラメータ化された接続を使用するには、最初に、有効な接続を使用するマッピングで SQL トランスフォーメーションを作成します。次に、SQL トランスフォーメーションで接続をパラメータ化します。SQL トランスフォーメーションを使用して、Snowflake で Java または SQL ユーザー定義関数（UDF）から読み取りを行うこともできます。

SQL トランスフォーメーションを使用して、次のタイプの SQL 文を処理できます。

ストアドプロシージャ

Snowflake のストアドプロシージャを呼び出すように SQL トランスフォーメーションを設定できます。ストアドプロシージャ名では大文字と小文字が区別されます。データベースからストアドプロシージャを選択するか、SQL トランスフォーメーションで呼び出すストアドプロシージャの正確な名前を入力します。SQL トランスフォーメーションを作成するには、Snowflake データベースにストアドプロシージャが存在している必要があります。

詳細 SQL プロパティで Snowflake データベース、スキーマ、およびプロシージャ名を指定した場合、エージェントでは最初にストアドプロシージャで指定したプロパティが考慮され、次に詳細ソースプロパティ、次に接続内の追加の JDBC URL パラメータ、最後にソースオブジェクトのメタデータが考慮されます。

マッピングが開いている状態で新しいストアドプロシージャをデータベースに追加すると、新しいストアドプロシージャは使用可能なストアドプロシージャのリストに表示されません。リストを更新するには、マッピングを閉じてから再度開きます。

SQL **クエリ**

SQL エディタで定義した入力済みクエリを処理するように SQL トランスフォーメーションを設定できます。SQL トランスフォーメーションで複数の SQL クエリを使用しないでください。

SQL トランスフォーメーションはクエリを処理し、行を返します。SQL トランスフォーメーションは、基になるデータベースからエラーが発生した場合、またはユーザー構文にエラーがある場合でもエラーを返します。

SQL クエリとストアドプロシージャの詳細については、データ統合のドキュメントにある「*トランスフォーメーション*」を参照してください。

ルールおよびガイドライン

次のような制限付きで、SQL トランスフォーメーションを使用できます。

- 保存されたクエリタイプを SQL トランスフォーメーションで使用することはできません。
- スキーマオーバーライドを指定して、同じ名前と引数の数を持つ複数のストアドプロシージャが生成された場合、マッピングは失敗します。
- ストアドプロシージャに Unicode 文字が含まれている場合、マッピングは失敗します。
- ストアドプロシージャから NULL が返されると、ユーザーインターフェイスとセッションログに警告が表示されます。ただし、マッピングは引き続き行を処理します。
- ランタイム処理では、SQL トランスフォーメーションの次のプロパティが無視されます。

  - 入出力および入力パラメータ

  - 詳細プロパティ

  - 自動コミット

  - トランスフォーメーション範囲

  - エラー時に停止
- マッピングに SQL トランスフォーメーションが含まれている場合は、キー範囲パーティション化を使用できません。

動的スキーマの処理

マッピングをマッピングタスクに追加するときに、データ統合がデータオブジェクトスキーマの変更を処理する方法を選択できます。タスクを実行するたびにスキーマが更新されるようにするには、タスクで動的スキーマ処理を有効にします。

スキーマの変更には、データオブジェクトに対する次の 1 つ以上の変更が含まれます。

- フィールドの追加、削除、またはフィールド名の変更。
- フィールドのデータ型、精度、またはスケールの更新。

タスクの設定時に、**［スケジュール］**タブの**［詳細オプション］**セクションで、スキーマ変更処理を設定します。非同期的または動的なスキーマ変更処理を設定できます。

動的なスキーマ変更処理を設定する場合は、次のオプションから選択してスキーマを更新できます。

変更して変更を適用

データ統合はソーススキーマからターゲットスキーマに対する次のような変更を適用します。

- 新しいフィールド。ターゲットスキーマを変更し、ソースから新しいフィールドを追加します。
- 名前が変更されたフィールド。名前が変更されたフィールドをターゲットの新しいカラムとして追加します。
- データ型と精度の更新。これらの変更をターゲットに適用します。スケールの更新には適用されません。
- 削除されたフィールド。削除されたフィールドを無視します。

注: 詳細モードのマッピングの名前が変更されたフィールドまたは新しいフィールドのスキーマ変更を含めるには、「詳細モードのマッピングの動的スキーマの処理」 (ページ 27)を参照してください。

DDL **の変更を適用しない**

データ統合はスキーマの変更をターゲットに適用しません。

現在のものを削除してから再作成

既存のターゲットテーブルを削除し、ソースからのすべての受信メタデータフィールドを使用して、実行時にターゲットテーブルを再作成します。

詳細モードのマッピングの動的スキーマの処理

詳細モードのマッピングの実行時にカラムの名前が変更された場合やソースにフィールドが追加された場合は、ソーススキーマとターゲットスキーマのカラムに発生する差異によって、ターゲットスキーマに不整合が生じます。

これを回避するには、ターゲットトランスフォーメーションの**［追加の書き込みランタイムパラメータ］**フィールドで、次のプロパティをアンパサンドで区切って設定します。

- column_mapping=name
- column_mismatch_behavior=ignore

これらのプロパティの設定を行うと、データ統合には名前が含まれるようになり、カラム名に大文字、数字、またはアンダースコアが含まれている場合、ソーススキーマとターゲットスキーマ間の不一致も無視されます。

カラム名に大文字が含まれていない場合は、さらにソーストランスフォーメーションとターゲットトランスフォーメーションで keep_column_case=on パラメータを設定する必要があります。

- ソースについては、Snowflake Data Cloud 接続の**［JDBC URL の追加パラメータ］**でパラメータを設定します。

- ターゲットについては、ターゲットトランスフォーメーションの［**追加の書き込みランタイムパラメータ**］プロパティでパラメータを指定します。

動的スキーマ処理のルールとガイドライン

動的スキーマ変更処理を有効にする場合は、次のルールとガイドラインを考慮してください。

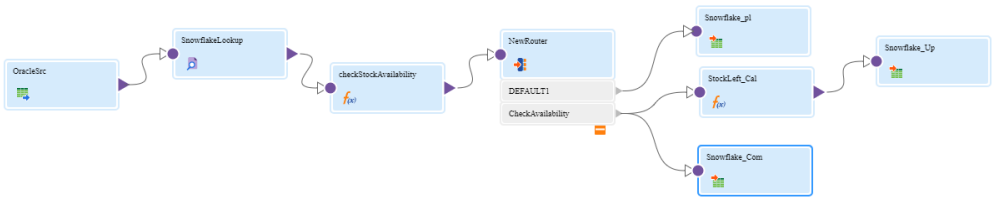
- 詳細プロパティからのターゲットまたはソースに対するオーバーライドを含めないでください。マッピングで動的スキーマ処理が有効で、ソースまたはターゲットのオーバーライドが含まれている場合、マッピングは次のエラーで失敗します: `Exception: class java.lang.NullPointerException occurred in writeMetadata`
ただし、動的スキーマ処理が有効なマッピングでは SQL オーバーライドを指定できます。
- 接続プロパティで keep_column_case=on パラメータを設定して、大文字以外のカラム名の動的スキーマ変更を修正する場合は、この接続を詳細モードのマッピングにのみ使用するようにしてください。データ統合は、この接続を使用して、詳細クラスタでマッピングを実行します。詳細モードではないマッピングで同じ接続を使用すると、エラーが発生します。
- 詳細モードでは、ターゲットトランスフォーメーションプロパティで［**ターゲットテーブルの切り詰め**］オプションを指定しないでください。カラムの名前が変更されるか、カラムがソーススキーマに追加された場合、［**ターゲットテーブルの切り詰め**］を設定してターゲットトランスフォーメーションに［**追加の書き込みランタイムパラメータ**］も設定すると、マッピングは次のエラーで失敗します:
`java.lang.UnsupportedOperationException`

マッピングの例

エンタープライズアプリケーションは、Oracle データベースを使用して製品トランザクション詳細を格納します。ソースデータからの完了したトランザクションおよび保留中のトランザクションに基づいて、ストックの可用性を確認する必要があります。さらなる分析を行うためには、利用可能なストックとトランザクションの詳細を Snowflake に統合する必要があります。

マッピング

以下の図は、このユースケースの Snowflake Data Cloud マッピングを示しています。



Oracle ソースから製品トランザクション詳細を読み取るためのマッピングを作成し、製品の詳細とその可用性を格納している Snowflake の PRODUCTDET テーブルにルックアップ条件を適用します。可用性と要件に基づいて、トランザクションを Snowflake の PENDINGTRANSACTION テーブルおよび COMPLETEDTRANSACTION テーブルに書き込み、完了したトランザクションに対して PRODUCTDET テーブルの INSTOCK フィールドを更新します。

Snowflake データクラウドマッピングでは次のトランスフォーメーションを使用します。

ソーストランスフォーメーション

Oracle ソースから製品トランザクションの詳細を読み取るには、Oracle に接続するためのソーストランスフォーメーションに Oracle 接続を含めます。マッピングタスクは Oracle からの OracleSrc テーブルです。

次の図は、読み取りを行う OracleSrc テーブルに格納されているトランザクションの詳細を示しています。

transactionID	CustomerID	productID	quantity	OrderPlacedOn
Tran511	CUST21	P45	100	2016-04-05
Tran512	CUST22	P46	200	2016-07-05
Tran513	CUST23	P47	20	2016-07-25
Tran514	CUST24	P47	100	2016-10-25
Tran515	CUST25	P45	1000	2016-12-02
Tran517	CUST27	P46	5000	2017-01-02
Tran516	CUST26	P48	60	2017-01-02
Tran518	CUST28	P49	60	2017-01-03
Tran519	CUST29	P50	700	2017-03-13
Tran520	CUST30	P47	750	2017-03-14

ルックアップトランスフォーメーション

マッピングタスクのルックアップオブジェクトは、製品とその可用性の詳細を格納している Snowflake の PRODUCTDET テーブルです。製品の詳細および製品 ID に基づく製品の可用性が格納された Snowflake の PRODUCTDET テーブルにルックアップ条件を適用します。

以下の図は、PRODUCTDET テーブルに格納されているデータを示しています。

Data Preview

Connection: snowflake_CQA Object: PRODUCTDET

PRODUCTID	IN STOCK	PRODUCTDET	PRICE
p45	900	2.5" 80GB IDE Laptop Har...	1968
p46	10000	Laptop Internal CD/DVD R...	1229
p47	5000	New HP ProBook 430 G3 ...	5289
p48	50	New HP ProBook 430 G3 ...	9594
p49	20	Dell Inspiron 15R N5110 B...	1699
p50	800	HP 15-be016TU 15.6-inch...	27490

式トランスフォーメーション

式トランスフォーメーションでは、式を使用してストックの状況を計算します。

ルータートランスフォーメーション

ルータートランスフォーメーションは、ストックの可用性に基づいてデータをフィルタリングし、完了したトランザクション、保留中のトランザクション、および製品の詳細を適切なターゲットテーブルにリダイレクトします。

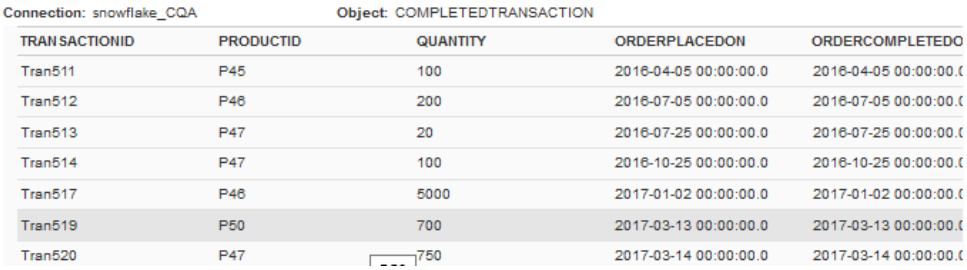
ターゲットトランスフォーメーション

マッピングタスクには、完了したトランザクション、保留中のトランザクション、および製品の詳細を書き込むための次のようなターゲットオブジェクトがあります。

COMPLETEDTRANSACTION

COMPLETEDTRANSACTION テーブルには、TRANSACTIONID、PRODUCTID、QUANTITY、ORDERPLACEDON、ORDERCOMPLETEDON の各フィールドが含まれています。

以下の図は、COMPLETEDTRANSACTION テーブルに格納されているデータを示しています。



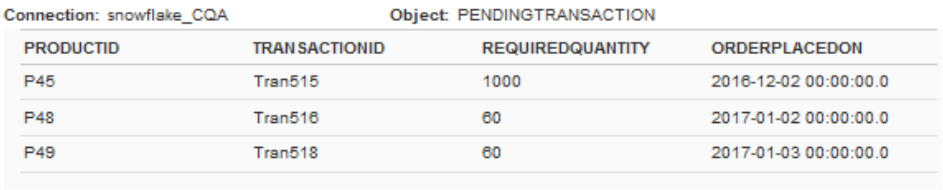
Connection: snowflake_CQA　　Object: COMPLETEDTRANSACTION

TRANSACTIONID	PRODUCTID	QUANTITY	ORDERPLACEDON	ORDERCOMPLETEDO
Tran511	P45	100	2016-04-05 00:00:00.0	2016-04-05 00:00:00.(
Tran512	P46	200	2016-07-05 00:00:00.0	2016-07-05 00:00:00.(
Tran513	P47	20	2016-07-25 00:00:00.0	2016-07-25 00:00:00.(
Tran514	P47	100	2016-10-25 00:00:00.0	2016-10-25 00:00:00.(
Tran517	P46	5000	2017-01-02 00:00:00.0	2017-01-02 00:00:00.(
Tran519	P50	700	2017-03-13 00:00:00.0	2017-03-13 00:00:00.(
Tran520	P47	750	2017-03-14 00:00:00.0	2017-03-14 00:00:00.(

PENDINGTRANSACTION

PENDINGTRANSACTION テーブルには、PRODUCTID、TRANSACTIONID、REQUIREDQUANTITY、ORDERPLACEDON の各フィールドが含まれています。

以下の図は、PENDINGTRANSACTION テーブルに格納されているデータを示しています。



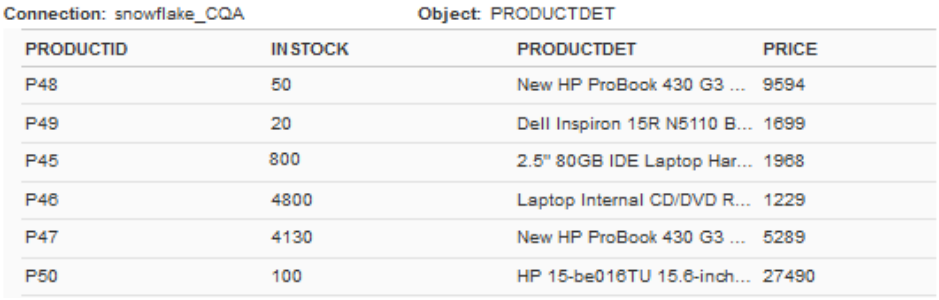
Connection: snowflake_CQA　　Object: PENDINGTRANSACTION

PRODUCTID	TRANSACTIONID	REQUIREDQUANTITY	ORDERPLACEDON
P45	Tran515	1000	2016-12-02 00:00:00.0
P48	Tran516	60	2017-01-02 00:00:00.0
P49	Tran518	60	2017-01-03 00:00:00.0

PRODUCTDET

PRODUCTDET テーブルには、PRODUCTID、INSTOCK、PRODUCTDET、PRICE の各フィールドが含まれています。完了したトランザクションに基づいて、INSTOCK フィールドが更新されます。

以下の図は、PRODUCTDET テーブルに格納されているデータを示しています。



Connection: snowflake_CQA　　Object: PRODUCTDET

PRODUCTID	INSTOCK	PRODUCTDET	PRICE
P48	50	New HP ProBook 430 G3 ...	9594
P49	20	Dell Inspiron 15R N5110 B...	1699
P45	800	2.5" 80GB IDE Laptop Har...	1968
P46	4800	Laptop Internal CD/DVD R...	1229
P47	4130	New HP ProBook 430 G3 ...	5289
P50	100	HP 15-be016TU 15.6-inch...	27490

マッピングを実行すると、エージェントはソースからトランザクション詳細を読み取り、ルックアップからフィールドを取得し、適用された条件に基づいて、使用可能な数量とトランザクション詳細をターゲットテーブルに書き込みます。

詳細モードのマッピングの例

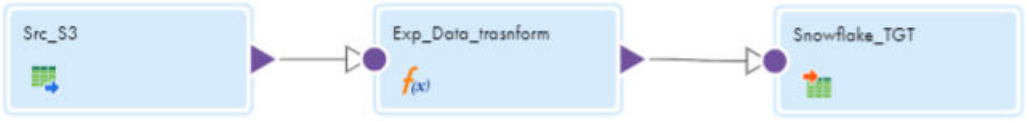
5 万以上の製品を提供し、全世界に店舗が分散している小売企業で働いているとします。会社は大量の顧客エンゲージメント詳細をトランザクション CRM システムから Amazon S3 へ取り込みます。

営業チームはあらゆる接点での顧客エンゲージメントと満足度を高めたいと考えています。シームレスなカスタマエクスペリエンスとパーソナライズされたサービスをさまざまな販路で提供するために、この小売企業は

parquet ファイル形式で保存されたデータを Amazon S3 バケットから Snowflake にロードすることを計画しています。

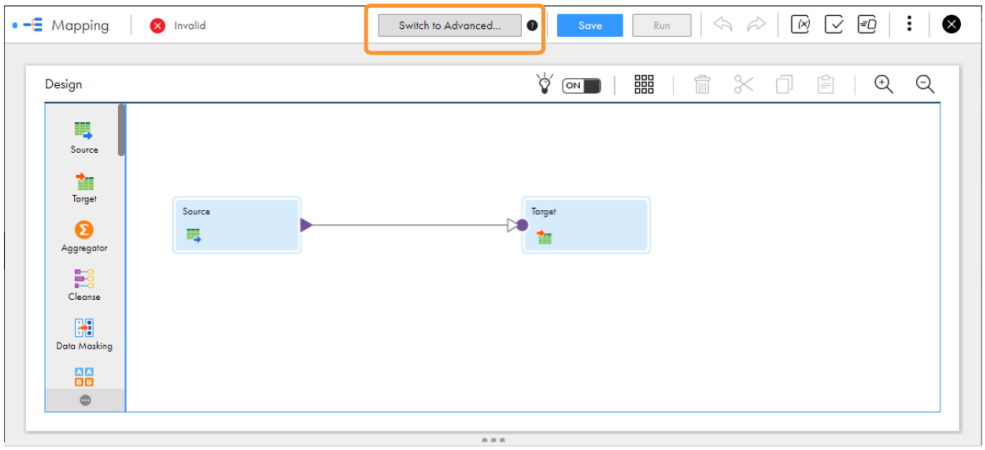
詳細モードでマッピングを作成して、Amazon S3 バケットからデータを読み取り、Snowflake ターゲットにデータを書き込むことができます。マッピングにトランスフォーメーションを追加して、Amazon S3 バケットから読み取った RAW データを処理し、キュレートされたデータを Snowflake に書き込むという選択ができます。

次の例に、Amazon S3 ソースから読み取り、Snowflake へ書き込むマッピングを作成する方法を示します。



1. データ統合で、**［新規］** > **［マッピング］** > **［マッピング］** をクリックします。
2. Mapping Designer で、**［詳細に切り替え］** をクリックします。

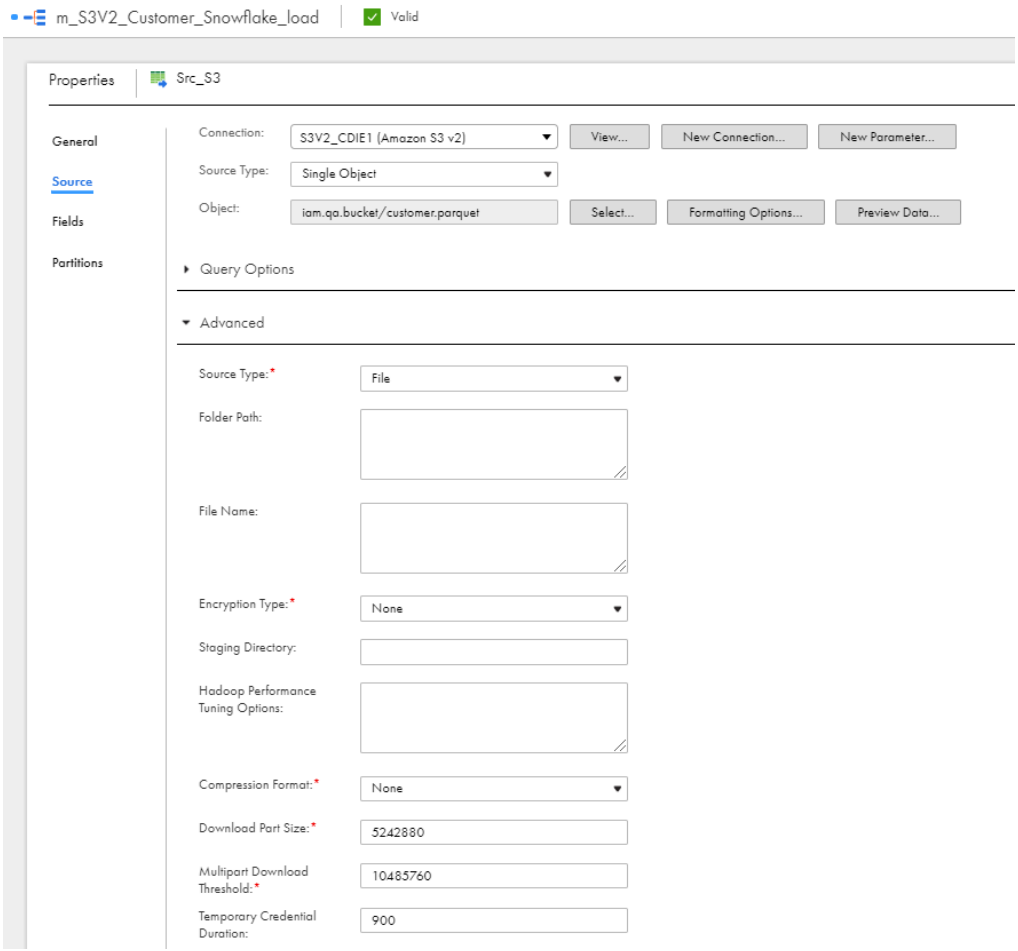
   次の画像は、Mapping Designer の **［詳細に切り替え］** ボタンを示しています。



3. **［詳細に切り替え］** ダイアログボックスで、**［詳細に切り替え］** をクリックします。

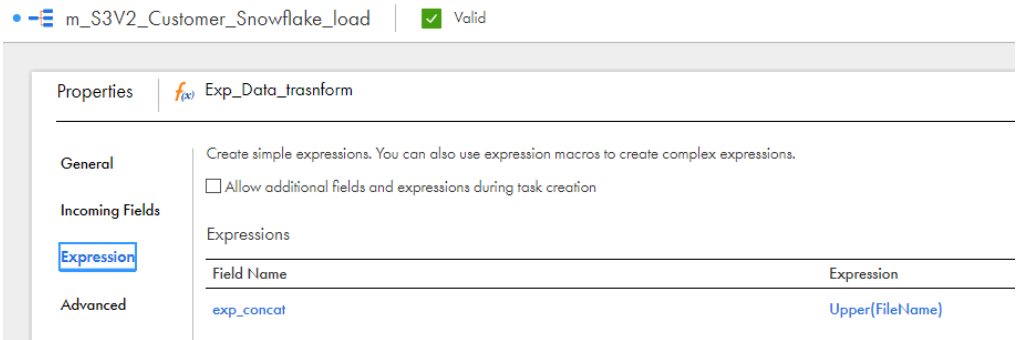
   Mapping Designer は、詳細モードで使用できるトランスフォーメーションや関数が表示されるようにマッピングキャンバスを更新します。
4. マッピングの名前、場所、説明を入力します。
5. ソーストランスフォーメーションを追加して、全般プロパティで名前と説明を指定します。
6. Amazon S3 ソースからデータを読み取るには、**［ソース］** タブで次の手順を実行します。
   a. **［接続］** フィールドで、Amazon S3 V2 接続を選択します。
   b. **［ソースタイプ］** フィールドで、ソースタイプとして単一オブジェクトを選択します。
   c. **［オブジェクト］** フィールドで、顧客の詳細を含む parquet ファイルオブジェクトを選択します。
   d. **［詳細プロパティ］** セクションで、必要なパラメータを指定します。

次の図に、顧客エンゲージメントの詳細を Amazon S3 オブジェクトから読み取る、設定済みのソーストランスフォーメーションプロパティを示します。



7. **[式]** タブで、データを Snowflake ターゲットに書き込む前に、ビジネス要件に基づいて、顧客 parquet ファイルのファイル名ポートを大文字に変更する式を定義します。

   次の図に、設定済みの式トランスフォーメーションプロパティを示します。



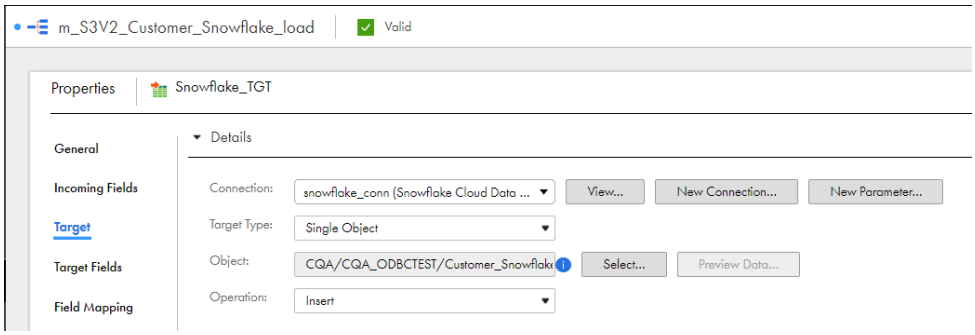
8. ターゲットトランスフォーメーションを追加して、全般プロパティで名前と説明を指定します。
9. **[ターゲット]** タブで、データを Snowflake に書き込む詳細を指定します。
   a. **[接続]** フィールドで、Snowflake Data Cloud ターゲット接続を選択します。
   b. **[ターゲットタイプ]** フィールドで、単一オブジェクトを選択します。

c. **［オブジェクト］**フィールドで、キュレーションされた顧客エンゲージメントデータを書き込む先の Snowflake オブジェクトを選択します。

d. **［操作］**フィールドで、挿入操作を選択します。

e. **［詳細プロパティ］**セクションで、必要な詳細ターゲットプロパティを指定します。

次の図に、構成された Snowflake ターゲットトランスフォーメーションプロパティを示します。



10. **［保存］** > **［実行］**をクリックして、マッピングを検証します。

［監視］で、タスクの実行後にログのステータスを監視できます。

詳細モードのマッピングのパラメータ化の制限

詳細モードのマッピングについては、次の制限を考慮してください。

- マッピングで接続のみをパラメータ化することはできません。マッピングを正常に実行するには、接続とオブジェクトの両方をパラメータ化する必要があります。
- ［パラメータ化］または［自動］としてフィールドマッピングを選択します。ソースおよびターゲットの接続とオブジェクトをパラメータ化する場合、手動フィールドマッピングは機能しません。
- オブジェクトをパラメータ化し、オーバーライドを指定した場合、オーバーライドされたオブジェクトのメタデータは、タスクプロパティで指定されたオブジェクトと一致している必要があります。
- 実行時に新しいターゲットを作成した場合、オブジェクトをパラメータ化することはできません。
- パラメータファイルを使用してパラメータをオーバーライドした場合、ソースフィルタ、カスタムクエリ、複数のオブジェクト、およびソース、ルックアップ、およびターゲットの詳細ソースプロパティに入力パラメータを使用することはできません。
- マッピングでパラメータを定義した場合にのみ、パラメータファイルを使用してオーバーライドできます。

第 4 章

Snowflake Data Cloud のソース

ソースからデータを抽出するには、ソーストランスフォーメーションを追加します。マッピングにソーストランスフォーメーションを追加するときに、Snowflake Data Cloud 接続タイプに関連するソース接続、ソースオブジェクト、およびソースプロパティを定義します。

Snowflake Data Cloud のソースプロパティ

マッピングでは、ソーストランスフォーメーションを設定して Snowflake Data Cloud ソースを表すことができます。

次の表に、ソーストランスフォーメーションで設定できる Snowflake Data Cloud のソースプロパティを示します。

プロパティ	説明
接続	ソース接続の名前。 既存の接続の選択、新しい接続の作成、またはソース接続プロパティのパラメータ値の定義ができます。 **注**: パラメータ化されていない Snowflake Data Cloud 接続とパラメータ化された Snowflake Data Cloud 接続を切り替えることができます。詳細プロパティ値は、切り替え中も保持されます。 実行時にソース接続プロパティを上書きする場合は、**［実行時にパラメータのオーバーライドを許可する］**オプションを選択します。 詳細セッションプロパティのパラメータファイルディレクトリと名前を指定します。
ソースタイプ	ソースオブジェクトのタイプ。 ［単一オブジェクト］、［複数のオブジェクト］、［クエリ］、または［パラメータ］を選択します。
パラメータ	タスクを編集せずに更新する値を定義するパラメータファイル。 ソースオブジェクトの既存のパラメータを選択するか、**［新しいパラメータ］**をクリックしてソースオブジェクトの新しいパラメータを定義します。 パラメータをソースタイプとして選択する場合にのみ、**［パラメータ］**プロパティが表示されます。 実行時にパラメータを上書きする場合は、パラメータの作成時に**［実行時にパラメータのオーバーライドを許可する］**オプションを選択します。タスクを実行すると、そのタスクは詳細セッションプロパティで指定したファイルからパラメータを使用します。

プロパティ	説明
オブジェクト	タスクのソースオブジェクト。 単一ソースのソースオブジェクトを選択します。複数のソースのオプションを選択した場合は、複数のソースオブジェクトを追加し、それらの間の関係を設定できます。
フィルタ	フィルタ条件に基づいてレコードをフィルタリングします。 簡易フィルタまたは詳細フィルタを指定できます。
ソート	指定した条件に基づいてレコードをソートします。 以下のソート条件を指定できます。 - パラメータを使用しません。使用するフィールドとソートのタイプを選択します。 - パラメータを使用します。ソートオプションを指定するには、パラメータを使用します。
個別の行のみ選択 [1]	ソーステーブルから個別の行を抽出します。 **注:** ソースタイプをクエリとして選択するか、SQL クエリプロパティを指定すると、[個別選択] オプションは無視されます。
[1] 詳細モードのマッピングには適用されません。	

以下の表に、ソース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。

詳細プロパティ	説明
データベース	接続で指定されたデータベースをオーバーライドします。
スキーマ	接続で指定されたスキーマをオーバーライドします。
ウェアハウス	接続で指定された Snowflake ウェアハウス名をオーバーライドします。
ロール	接続で指定された、ユーザーに割り当て済みの Snowflake ロールをオーバーライドします。 マッピングのウェアハウス名によって、接続で指定したウェアハウス名をオーバーライドします。接続プロパティで不適切なウェアハウス名を指定しても、接続は成功します。ただし、マッピングを実行する前に、マッピングプロパティで正しいウェアハウス名が指定されていることを確認してください。
Pre SQL	エージェントがデータを読み取る前に Snowflake ソーステーブルで実行される Pre-SQL コマンド。 例えば、テーブルからレコードを読み取る前にデータベースのレコードを更新する場合には、Pre-SQL 文を指定します。 クエリには、完全修飾テーブル名が含まれている必要があります。複数の Pre-SQL コマンドは、それぞれをセミコロンで区切って指定できます。
Post SQL	エージェントが読み取り操作を完了した後に Snowflake テーブルで実行される Post-SQL コマンド。 例えば、最新のレコードの読み込み後に一部のレコードを削除する場合には、Post-SQL 文を指定します。 クエリには、完全修飾テーブル名が含まれている必要があります。複数の Post-SQL コマンドは、それぞれをセミコロンで区切って指定できます。
テーブル名	インポートされた Snowflake ソーステーブルのテーブル名をオーバーライドします。

詳細プロパティ	説明
SQL オーバーライド	Snowflake ソースからデータを読み取るために使用されるデフォルトのクエリをオーバーライドする SQL 文。
トレースレベル	ログファイルに表示される詳細情報の量を決定します。[簡易]、[ノーマル]、[詳細 - 初期化]、または [詳細 - データ] を選択できます。デフォルト値は [ノーマル] です。

ソースオブジェクトと操作

ソーストランスフォーメーションでは、単一のオブジェクト、複数のオブジェクト、クエリ、またはパラメータをソースタイプとして使用して、Snowflake からデータを読み取ることができます。

特定のソースオブジェクトには、いくつかの制限が適用されます。

パラメータソースタイプ

ソースオブジェクトと接続をパラメータ化して、トランスフォーメーションで **[実行時にパラメータのオーバーライドを許可する]** オプションを有効にした場合は、db.schema.tablename などの完全修飾名を使用してオブジェクト名をオーバーライドすることはできません。

Snowflake Data Cloud 接続の **[JDBC URL の追加パラメータ]** フィールドにある db=<dbname>&schema<schemaname>値を渡すことができます。

複数のソースタイプ

単一のソーストランスフォーメーションを使用して、同じデータベース内の複数の Snowflake テーブルから読み取りを行うことができます。複数の Snowflake ソースから読み取るには、複数の Source トランスフォーメーションを作成してから、ジョイナトランスフォーメーションを使用してソースを結合します。

単一のソーストランスフォーメーションを使用して複数のテーブルから読み取るには、ソースタイプとして複数のオブジェクトを選択してから、テーブルを結合するように設定します。定義済みの PK-FK リレーションを含む関連オブジェクトを追加するか、またはテーブルを結合するためのリレーション条件を定義できます。テーブル間のリレーションを定義するための独自の条件を設定するには、**[関連オブジェクトのアクション]** メニューから **[詳細リレーション]** を選択し、リレーションを定義します。結合式を設定する場合、フィールドを選択し、結合クエリ構文を定義します。条件をのみを指定する必要があり、クエリ内の結合のタイプは必要ありません。式用にテキストボックスに指定した条件は、結合条件に追加されます。

詳細リレーションに結合条件を指定してテーブルを結合する場合、接続からのデータベースおよびスキーマ名をオーバーライドすることはできません。詳細リレーション条件でデータベースおよびスキーマ名を手動で変更する必要があります。条件に db.schema.tablename などの完全修飾名のカラムが含まれている場合は、オーバーライドを設定しないでください。詳細リレーション条件から完全修飾データベースおよびスキーマ名を削除してからマッピングを実行します。

複数のソースタイプの制限

複数のソースタイプには、次の制限があります。

- 関連オブジェクトを使用して結合された複数のテーブルからデータを読み取る場合、テーブルおよびカラム名に大文字と小文字を区別するカラムが含まれていると、マッピングが失敗します。
- 同じカラム名を持つ 1 つ以上のテーブルの結合で設定されたマッピングは失敗します。

- 同じデータベースとスキーマに属していない複数の Snowflake オブジェクトから読み取りを行うマッピングは失敗します。
- Snowflake ソースの詳細プロパティでテーブル、スキーマ、またはデータベースへのオーバーライドを指定すると、複数のテーブルから読み取る結合で設定されたマッピングが失敗します。

クエリソースタイプ

カスタム SQL クエリを使用して Snowflake テーブルをインポートする場合は、カスタム SQL クエリで Snowflake データベースとスキーマ名を指定します。データベースとスキーマ名を指定しない場合、エージェントでは接続プロパティで指定したデータベースとスキーマ名が考慮されます。Snowflake から読み取るクエリのテーブル名は完全修飾されたものである必要があります。カスタムクエリを使用してストアドプロシージャを呼び出す場合は、ロールがデータベースとスキーマにアクセスできることを確認してください。

次のような制限付きで、クエリソースタイプを使用できます。

- 接続内のデータベース名とスキーマ名のオーバーライドを設定しないでください。
- 次の操作は、クエリソースタイプには適用されません。
  - フィルタとソートのオプション。
  - ソースパーティション。
  - Pre-SQL 文と Post-SQL 文を除く詳細プロパティ。
  - 詳細モードでは、ソーストランスフォーメーションのカスタムクエリからストアドプロシージャを呼び出すことはできません。

ソースに関する一般的な制限

ソーストランスフォーメーションには、次の一般的な制限があります。

- 二重引用符を含む Snowflake のテーブル名とカラム名の読み取りまたは書き込みを行うことはできません。マッピングは次のエラーで失敗します: `SQL compilation error`
- 詳細モードでは、500 を超えるカラムから読み取りを行うマッピングは、次のエラーで失敗します: `HTTP POST request failed due to IO error`。
- フィルタでシステム変数を使用することはできません。

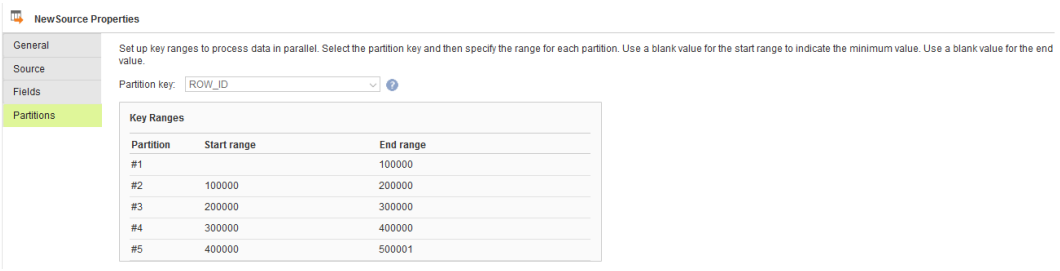
キー範囲パーティション化

マッピングを使用して、Snowflake ソースからデータを読み取る場合、キー範囲パーティション化を設定できます。パーティションタイプによって、エージェントがパーティションポイントのパーティション間でデータを分散する方法を制御します。パーティション化は、詳細モードのマッピングには適用されません。

パーティション化は、実行時のマッピングパフォーマンスを最適化します。キー範囲パーティション化により設定されたマッピングを実行するには、このエージェントはパーティションキーとして定義したフィールドに基づいてソースデータ行を分散します。エージェントは、フィールド値を各パーティションの範囲値と比較して、適切なパーティションに行を送信します。

[新しいキー範囲の追加] をクリックし、エージェントがデータをパーティション化する必要があるときの基準となるパーティションの数とキー範囲を定義します。

次の図は、**[パーティション]** タブにある、設定されたパーティション範囲の例を示しています。



最小値を示すには、開始範囲に空白値を使用します。最大値を示すには、終了範囲に空白値を使用します。

データ値が均等に分散されているカラムには、キー範囲パーティション化を使用します。そうしないと、パーティションのサイズが等しくなくなる可能性があります。例えば、あるカラムのキー値 1 から 1000 の間には 10 行あり、キー値 1001 から 2000 の間には 999 行があるというようになります。マッピングに複数のソースが含まれる場合は、ソースごとに同じ数のキー範囲を使用します。

カラムのキー範囲パーティション化を定義すると、エージェントはその指定されたパーティション範囲内にある行を読み取ります。例えば、カラムの 2 つのパーティションを範囲 10 から 20 と 30 から 40 に対して設定した場合、行 20 から 30 は指定したパーティション範囲内にないため、エージェントはこれらの行を読み取りません。

以下のデータ型のフィールドのパーティションキーを設定できます。

- Integer
- String
- すべての数値データ型。ただし、キー範囲の値には少数は使用できません。
- 日時。日付と時刻を指定するには、YYYY-MM-DD HH24:MI:SS という形式を使用します。例: 1971-01-01 12:30:3
  その他の形式で日付と時刻を指定すると、タスクは失敗します。

キー範囲パーティション化が有効になっているマッピングを実行すると、ソーストランスフォーメーションでフィルタを設定している場合でも、ソースデータをフィルタリングできません。

SQL のオーバーライド

オブジェクト、データベース、スキーマ、またはロールをオーバーライドする場合は、いくつかのガイドラインに従う必要があります。

Snowflake テーブルからのメタデータのインポートに使用されるカスタムクエリをオーバーライドする SQL オーバーライドクエリを指定する場合は、完全修飾テーブル名を指定します。

SQL オーバーライド、および詳細ソースプロパティからデータベース、スキーマ、ウェアハウス、またはロールへのオーバーライドを指定する場合は、オーバーライドの組み合わせについて次のガイドラインを考慮してください。

SQL とロールのオーバーライド

SQL と Snowflake オブジェクトのロールをオーバーライドする場合は、オーバーライドのロールがマッピングで選択されたテーブルにアクセスできることを確認してください。

SQL とロールをオーバーライドするには、次のいずれかのタスクを実行します。

- マッピングで選択した Snowflake オブジェクトに使用されているロールと同じアクセス権を持つオーバーライドロールを付与する。
- 詳細プロパティからテーブルへのオーバーライドを指定する。指定したオーバーライドテーブルは設計時に使用され、実行時には SQL オーバーライドが優先されます。

SQL、データベース、スキーマ、およびロールのオーバーライド

マッピングで Snowflake オブジェクトを選択し、SQL オーバーライド、および詳細プロパティからのデータベース、スキーマ、またはロールをオーバーライドして、オーバーライドするデータベースにそのテーブルが存在しない場合は、`table1 not found` というエラーが発生してマッピングが失敗します。

オーバーライドするデータベースとスキーマ名に対応する有効なテーブル名を指定してください。

SQL オーバーライドポート

SQL オーバーライドポートは、フィールドマッピングで接続されているポートと同じである必要があります。

第 5 章

Snowflake Data Cloud のターゲット

ターゲットトランスフォーメーションを追加して、データをターゲットに書き込みます。ターゲットトランスフォーメーションをマッピングに追加するときに、Snowflake Data Cloud 接続タイプに関連するターゲット接続、ターゲットオブジェクト、およびターゲットプロパティを定義します。

Snowflake Data Cloud のターゲットプロパティ

Snowflake Data Cloud ターゲットを表すようにターゲットトランスフォーメーションを設定できます。

次の表に、ターゲットトランスフォーメーションで設定できる Snowflake Data Cloud のターゲットプロパティを示します。

プロパティ	説明
接続	ターゲット接続の名前。 既存の接続の選択、新しい接続の作成、またはターゲット接続プロパティのパラメータ値の定義ができます。 **注**: パラメータ化されていない Snowflake Data Cloud 接続とパラメータ化された Snowflake Data Cloud 接続を切り替えることができます。接続を切り替えても、詳細プロパティ値は保持されます。 実行時にターゲット接続プロパティを上書きするには、**[実行時にパラメータのオーバーライドを許可する]** オプションを選択します。
ターゲットタイプ	ターゲットオブジェクトのタイプ。**[単一オブジェクト]** または **[パラメータ]** を選択します。
パラメータ	タスクを編集せずに更新する値を定義するパラメータファイル。 ターゲットオブジェクトの既存のパラメータを選択するか、**[新しいパラメータ]** をクリックしてターゲットオブジェクトの新しいパラメータを定義します。 パラメータをターゲットタイプとして選択する場合にのみ**パラメータ**プロパティが表示されます。 実行時にターゲットオブジェクトを上書きする場合は、**[実行時にパラメータのオーバーライドを許可する]** オプションを選択します。タスクを実行すると、そのタスクは詳細セッションプロパティで指定したファイルからパラメータを使用します。

プロパティ	説明
オブジェクト	タスクのターゲットオブジェクト。ターゲットオブジェクトを選択します。 既存のテーブルを選択するか、新規のテーブルを作成できます。既存のテーブルを選択するか、**［実行時に新規作成］**オプションを使用してターゲットに新しいテーブルを作成することにより、データを書き込むことができます。
実行時に新規作成	Snowflake ターゲットテーブルを、指定したテーブルタイプとパスに基づいて実行時に作成します。 実行時にターゲットテーブルを作成するには、次のパラメータを指定します。 - オプション。テーブルタイプを table として指定します。 - **［パス］**フィールドで、Snowflake データベース名とスキーマを<database name>/<schema>の形式で指定します。 エージェントは、指定したオブジェクト名とパスに基づいてターゲットテーブルを作成します。 **注**: ターゲットを作成する前にソースフィールドのメタデータを編集できます。
ソースフィールド名をターゲットでそのまま使用	**［実行時に新規作成］**オプションに適用されます。 実行時に、正確なソースフィールド名でターゲットオブジェクトを作成できるかどうかを決定します。 すべてのソースフィールド名（特殊文字を含む）をソースとまったく同じようにターゲットに保持する場合に選択します。このオプションを無効にすると、ソースの特殊文字は、ターゲットでアンダースコア文字に置換されます。 デフォルトでは無効になっています。
操作	ターゲット操作。Insert、Update、Upsert、Delete、Data Driven のいずれかのコマンドを選択します。
カラムの更新	Snowflake ターゲットからデータを更新または削除するための一時キーカラム。
データ依存条件	行に挿入、更新、更新/挿入、または削除操作のフラグを設定する式を定義できます。

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。

詳細プロパティ	説明
UpdateMode	指定したモードに基づいてターゲットにデータをロードします。 更新操作またはデータ依存操作を選択した場合に適用されます。 次のいずれかのモードから選択します。 - 更新時に更新。エントリが存在する場合、更新のフラグが立てられたすべての行を更新します。 - Update Else Insert。エントリがターゲットに存在する場合、エージェントは最初に更新のフラグが立てられたすべての行を更新します。エントリが存在しない場合、エージェントはエントリを挿入します。
データベース	オブジェクトのインポートに使用したデータベースをオーバーライドします。
スキーマ	オブジェクトのインポートに使用したスキーマをオーバーライドします。

詳細プロパティ	説明
ウェアハウス	接続で指定した Snowflake 名をオーバーライドします。 マッピングのウェアハウス名によって、接続で指定したウェアハウス名をオーバーライドします。 接続プロパティで不適切なウェアハウス名を指定しても、接続は成功します。ただし、マッピングを実行する前に、マッピングプロパティで正しいウェアハウス名が指定されていることを確認してください。
ロール	接続で指定された、ユーザーに割り当てられた Snowflake ロールをオーバーライドします。
Pre SQL	エージェントが Snowflake に書き込む前に実行する Pre-SQL コマンド。 例えば、テーブルにデータを書き込む前に、ターゲットテーブルのプライマリキーフィールドにシーケンスオブジェクトを割り当てる場合には、Pre-SQL 文を指定します。 複数の Pre-SQL コマンドは、それぞれをセミコロンで区切って指定できます。
Post SQL	エージェントが書き込み操作を完了した後に実行される Post-SQL コマンド。 例えば、[ターゲットの作成] オプションを使用して作成したテーブルを変更し、テーブルにデータを書き込む前に制約をテーブルに割り当てる場合には、Post-SQL 文を指定します。 複数の Post-SQL コマンドは、それぞれをセミコロンで区切って指定できます。
バッチ行サイズ [1]	エージェントの場所の単一のファイルに書き込まれる行の数。ファイルに書き込まれる行数が指定した値に達すると、エージェントはデータキューをフラッシュし、書き込みコマンドの処理を開始します。 バッチサイズ値の設定の詳細については、「「バッチサイズとローカルステージングファイルの数の設定」 (ページ 47)」を参照してください。
ローカルステージングファイルの数 [1]	データの単一バッチを表すファイルの数。考慮されるデフォルトのファイル数は 64 です。 エージェントが指定した数のローカルステージングファイルを Snowflake ユーザーステージにアップロードしたら、Snowflake はデータをターゲットテーブルにアンロードします。 ローカルステージングファイルの数の設定に関する詳細については、「「バッチサイズとローカルステージングファイルの数の設定」 (ページ 47)」を参照してください。
ターゲットテーブルの切り詰め	新しい行を挿入する前にデータベースターゲットテーブルをトランケートします。次のいずれかのオプションを選択します。 - True。ターゲットテーブルをトランケートしてから、すべての行を挿入します。 - False。ターゲットテーブルをトランケートせずに、新しい行を挿入します。 デフォルトは false です。

詳細プロパティ	説明

追加の書き込みランタイムパラメータ	追加のランタイムパラメータを指定します。
	例えば、Snowflake データベースでユーザー定義ステージを指定してローカルステージングファイルをアップロードする場合は、ステージの場所の名前を次の形式で指定します。
	`remoteStage=REMOTE_STAGE`
	書き込みパフォーマンスを最適化する場合は、Snowflake テーブルに書き込む前にファイルの圧縮を選択できます。例えば、次のように圧縮パラメータをオンまたはオフに設定できます。
	`Compression=On`
	デフォルトでは、圧縮はオンです。
	複数のランタイムパラメータは&で区切ります。

テーブル名	Snowflake ターゲットテーブルのテーブル名をオーバーライドします。
拒否されたファイルパス [1]	拒否されたレコードを書き込むエージェントマシン上のファイルのファイル名とパス。 例: `\rejectedfiles\reject7`
更新オーバーライド	エージェントが更新操作用に生成するデフォルトの更新クエリを、更新クエリでオーバーライドします。
成功ファイルディレクトリ	該当なし。
エラーファイルのディレクトリ	該当なし。
拒否された行を転送	トランスフォーメーションが、拒否された行を次のトランスフォーメーションに渡すか、拒否された行を削除するかを決定します。デフォルトでは、エージェントは拒否された行を次のトランスフォーメーションに転送します。

[1] 詳細モードのマッピングには適用されません。

ターゲットオブジェクトと操作

ターゲットトランスフォーメーションでは、単一のオブジェクトまたはパラメータをターゲットタイプとして使用できます。

ターゲットタイプを選択する際、Snowflake ターゲットでデータを挿入、更新、更新/挿入、または削除する操作を選択できます。データ依存操作を使用して、挿入、更新、削除、または拒否操作の行にフラグを立てる式を定義することもできます。

ターゲット操作の制限

ターゲット操作を設定する際は、いくつかのルールが適用されます。

パラメータ化されたターゲット

ソース、ルックアップ、またはターゲットオブジェクトおよび接続をパラメータ化して、トランスフォーメーションで［**実行時にパラメータのオーバーライドを許可する**］オプションを有効にした場合は、db.schema.tablename などの完全修飾名を使用してオブジェクト名をオーバーライドすることはできません。Snowflake Data Cloud 接続の［**JDBC URL の追加パラメータ**］フィールドにある db=<dbname>&schema<schemaname>値を渡す必要があります。

更新カラム

Snowflake ターゲットからデータを更新または削除するための一時キーカラムを指定する必要があります。Snowflake ターゲットにプライマリキーカラムが含まれていない場合は、［**追加**］をクリックして一時キーを追加します。複数のカラムを選択できます。

ソーステーブルのレコードに重複するプライマリキーが含まれている場合は、マッピングで次のいずれかのタスクを実行して、Snowflake のレコードを更新または削除します。

- ターゲットテーブルをインポートする前に、ターゲットテーブルで複数のプライマリキーを定義します。
- 詳細ターゲットプロパティの［カラムの更新］オプションを使用して、ターゲットオブジェクトに対して複数のカスタムキーを定義します。

詳細モードでマッピングを設定する場合は、さらに次のガイドラインを考慮する必要があります。

- データドリブン条件の式を指定して、［**更新カラム**］フィールドにカラムを指定していない場合、検証メッセージは表示されません。DD_INSERT および DD_REJECT の場合、更新カラムは必須ではありません。DD_UPDATE や DD_DELETE などの他の操作の場合、操作を機能させるには、更新カラムで少なくとも 1 つのフィールドを選択します。
- ターゲットトランスフォーメーションでパラメータ化された Snowflake 接続を使用した場合、［**更新カラム**］フィールドはターゲットプロパティに表示されません。ターゲットトランスフォーメーションで、リストに表示される有効な接続とオブジェクトを選択してから、操作を選択します。
- マッピングで Snowflake ターゲットオブジェクトがパラメータ化されており、選択した操作がデータドリブンまたは更新/挿入である場合、［更新カラム］フィールドは動的マッピングタスクのターゲットプロパティに表示されません。

データ依存操作

詳細モードのマッピングから Snowflake の行を更新するようにデータドリブン式を設定した場合、IIF 条件で 3 番目の引数を指定する必要があります。IIF 条件で 3 番目の引数を指定しない場合、エージェントでは一致しないすべての行に対する挿入として操作を扱います。

例えば、次の式 IIF(Update_column=2,DD_UPDATE,DD_DELETE)の DD_DELETE など 3 番目の引数を含めずに条件 IIF(Update_column=2,DD_UPDATE)を指定すると、update_column の値が 2 と等しくない他の行に対してエージェントはデフォルトで挿入を実行します。

例では、式 IIF(COL_INTEGER = 2, DD_UPDATE)から COL_INTEGER=2 は 1 に解決され、COL_INTEGER! =2 resolves to 0 となります。ここで、1 は更新操作の内部 ID、0 は挿入操作の内部 ID です。指定されていない場合の 3 番目の引数の値は、デフォルトで 0 になります。

詳細モードのマッピングには、次の制限付きでデータドリブン操作を使用できます。

- 挿入のマークが付いたソース行の 1 つがターゲットですでに使用可能である場合でも、エージェントはその行をターゲットに挿入します。
- データ依存操作タイプの式を定義すると、式エディタには最大 120 文字のカラム名のみが表示されます。
- データドリブン条件の式に特殊文字が含まれている場合、エージェントによるデータドリブン条件の検証が失敗しますが、マッピングは正常に実行されます。

一般的な制限

詳細モードでは、500 カラムを超えるデータの書き込みを行うマッピングは、次のエラーで失敗します: HTTP POST request failed due to IO error

パススルーのパーティション化

Snowflake ターゲットにデータを書き込むときのマッピングまたは実行時のパフォーマンスを最適化するようにパーティション化を設定できます。

パーティションタイプによって、エージェントがパーティションポイントのパーティション間でデータを分散する方法を制御します。パーティションタイプをパススルーパーティションとして定義できます。パーティション化が設定されていると、エージェントはパーティションとして定義したスレッドの数に基いてターゲットデータの行を分散します。

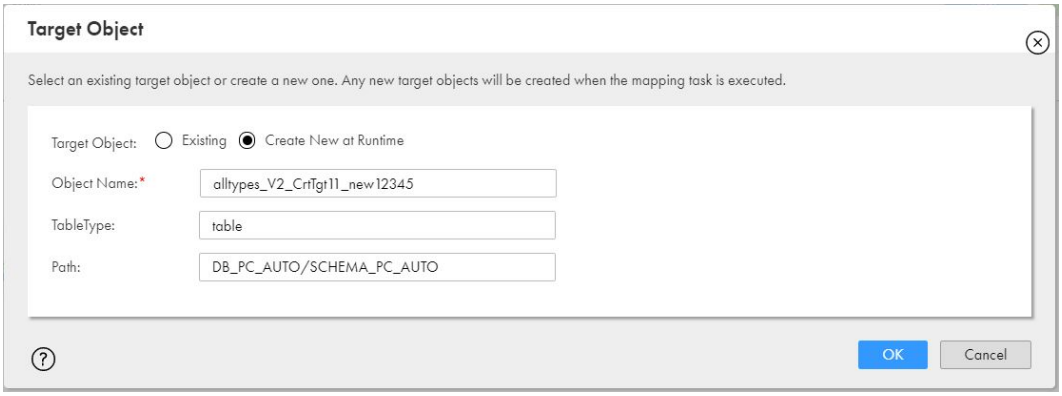
ターゲットの指定

既存のターゲットを使用するか、新しいターゲットを作成して Snowflake にデータを書き込むことができます。新しいターゲットを作成することを選択した場合は、タスクの実行時にエージェントによりターゲットが作成されます。

ターゲットオブジェクトに指定したスキーマとデータベースのパスが大文字であることを確認します。Snowflake のデータベース名とスキーマを、「<database name>/<schema name>」（ターゲットオブジェクトは大文字）という形式で指定します。パスを指定しない場合、Secure Agent は、Snowflake Data Cloud 接続プロパティから **[JDBC URL の追加パラメータ]** フィールドに指定したスキーマおよびデータベース名を考慮します。

TableType を table に指定します。TableType プロパティはオプションです。

次の図は、実行時に作成するように設定されたターゲットオブジェクトの例を示しています。



Snowflake ターゲットの制限

ターゲットを設定する際は、次のルールが適用されます。

- Windows に Secure Agent がインストールされている環境では、テーブル名に次の特殊文字が含まれている場合、Snowflake ターゲットテーブルにデータを書き込むことはできません: /\:*?"<>|
- 次のような制限付きで、実行時のターゲットを作成できます。
  - テーブル名とカラム名では大文字と小文字が区別されます。実行時にターゲットテーブルを作成する際に、Secure Agent によってテーブル名とカラム名に二重引用符が追加されます。Snowflake のテーブル名とカラム名をクエリする場合は、二重引用符を使用する必要があります。

- ソースオブジェクトに詳細フィルタ条件を追加する場合、またはソーストランスフォーメーションで複数のソースオブジェクトのリレーションを定義する場合、条件にデータベース名、スキーマ名、およびテーブル名を区切る「/」区切り文字が含まれていると、マッピングは SQL コンパイルエラーで失敗します。SQL クエリのスラッシュ「/」セパレータをドット「.」に変更して、マッピングを実行します。
- 指定したターゲット名に特殊文字が含まれており、ターゲット内の正確なソースフィールド名を使用するようにターゲットを設定せず、マッピングに式トランスフォーメーションまたはアグリゲータトランスフォーメーションも含まれている場合、ターゲットテーブルは特殊文字を使用して作成されます。アップストリームトランスフォーメーションからの、特殊文字を含む残りのフィールドは、ターゲット内でアンダースコア文字に置換されます。
- マッピングでターゲットオブジェクトをパラメータ化し、実行時にターゲットを作成する場合、ターゲットオブジェクトはパラメータファイルにフルパスを持っている必要があります。ただし、既存のターゲットオブジェクトを使用する場合、ターゲットオブジェクトに含まれるのはテーブル名のみであることが必要です。

更新操作のオーバーライド

更新オーバーライドを指定して、Secure Agent が更新操作に対して生成する更新クエリをオーバーライドできます。

更新オーバーライドを設定すると、Secure Agent は指定したクエリを使用し、ファイル内のデータをステージングし、Snowflake のローダーコピーコマンドを使用してデータを一時テーブルにロードします。一時テーブルのデータは次に Snowflake ターゲットテーブルにロードされます。更新クエリに指定する構文は Snowflake によりサポートされる必要があります。

更新クエリを次の形式で指定します。

```
UPDATE <Target table name> SET <Target table name>.<Column1> = :TU.<Column1>, <Target table name>.<Column2>
= :TU.<Column2>, ... <Target table name>.<ColumnN> = :TU.<ColumnN> FROM :TU WHERE <Target table name>.<Update
Column1> = :TU.<Update Column1> AND <Target table name>.<Update Column2> = :TU.<Update Column2> AND ...
<Target table name>.<Update ColumnN> = :TU.<Update ColumnN>
```

ここで、*:TU.*はターゲットポートの受信データソースを表します。

Secure Agent はマッピングの実行中に、*:TU.*を一時テーブル名に置き換え、更新クエリを検証しません。

Snowflake に書き込むようにマッピングの更新オーバーライドを設定する場合は、次のルールを考慮してください。

- :TU のカラム名がターゲットテーブルカラム名と一致していることを確認します。
- カラム名が完全修飾名であることを確認します。
- Snowflake Data Cloud コネクタは:TU を一時テーブル名に置き換え、更新クエリを検証しないため、有効な SQL 構文で更新クエリを指定します。
- 更新オーバーライドオプションを設定するときは、マッピングのカラムの順序を変更しないでください。
- マッピングの更新クエリに、ターゲットへの未接続フィールドを含めることはできません。

.csv ファイルサイズの最適化

Snowflake への書き込みのマッピングを作成するときに、ローカルステージングの.csv ファイルのサイズをバイト単位で指定できます。ローカルステージングファイルサイズのプロパティ csvFileSize を、詳細 Snowflake ターゲットプロパティの **[追加の書き込みランタイムパラメータ]** フィールドで指定します。デフォルトのファイルサイズは 50MB です。

目的のファイルサイズが 50MB の場合は、csvFileSize の値を 50*1024*1024 などのようにバイト単位で計算し、次に csvFileSize を 52428800 と指定します。最適なパフォーマンスを得るには、CSV ファイルの数とサイズが合っている必要があります。

バッチサイズとローカルステージングファイルの数の設定

Snowflake にデータの書き込みを行うマッピングで最適なパフォーマンスを実現するには、マッピングが処理するデータ量に基づいて、バッチサイズおよびローカルステージングファイルの数にデフォルト値が考慮されるようにします。値を設定する必要がある場合以外は、デフォルト値を使用してマッピングを処理することをお勧めします。

どちらのしきい値に先に到達したかによって、マッピングでのこれらの 2 つの設定の優先度が決定されます。指定したバッチ行サイズに到達する前にマッピングがローカルステージングファイルの数の基準を満たした場合は、そのしきい値に先に到達するため、ローカルステージングファイルの数が優先されます。

例として、指定したバッチサイズに対するさまざまな設定に発行される挿入コマンドの数と、Snowflake への書き込み操作でのローカルステージングファイルの数を示した次の表を参照してください。

ソース行の数	行サイズ	デフォルトのファイルサイズ	バッチサイズ	ローカルステージングファイルの数	挿入の数
4	52019206 バイト	52428800 バイト	4	1	2
4	52019206 バイト	52428800 バイト	4	設定されていません。64 というデフォルト値が考慮されます。	1
4	52019206 バイト	52428800 バイト	1	設定されていません。64 というデフォルト値が考慮されます。	4
4	52019206 バイト	52428800 バイト	3	1	2
5	52019206 バイト	52428800 バイト	1	1	4

シナリオ 1: バッチ行サイズが 4 で、ローカルステージングファイルの数が 1 である場合。

バッチ行サイズが 4 で、ローカルステージングファイルが 1 つである場合は、データの処理時に Snowflake で 2 つの挿入ステートメントが実行されます。

行が 2 つあり、ファイルサイズがデフォルトの 50 MB を超えると、次の受信行に新しいファイルが作成されます。ローカルステージングファイルの数が 1 に設定されている場合、ステージには最大 1 つのファイルが保持されます。したがって、2 つのステージが作成され、それぞれに 2 行の 1 つのファイルが保持されます。各ファイルサイズは 104038412 バイト、つまり 1 行あたり 52019206 バイトです。各ステージごとに挿入コマンドが発行され、2 つの挿入コマンドが生成されます。

シナリオ 2: バッチ行サイズが 4 で、ローカルステージングファイルの数の値が設定されていない場合。

ローカルステージングファイルの 64 というデフォルト値を使用して、マッピングでは、ローカルステージングファイルのしきい値に到達しない場合のバッチサイズが考慮されます。

シナリオ 3: バッチ行サイズが 1 で、ローカルステージングファイルの数の値が設定されていない場合。

バッチ行サイズが 1 で、特定のローカルステージングファイル値が設定されていない場合、ステージングファイルのしきい値に到達しないため、マッピングは 4 つの行に対して 4 つの挿入ステートメントを発行します。

シナリオ 4: バッチ行サイズが 3 で、ローカルステージングファイルの数の値が 1 である場合。

バッチ行サイズ 3 と 1 つのローカルステージングファイルを使用して、マッピングは、第 1 段階で 2 行、第 2 段階で 1 行を処理した後に、2 つの挿入コマンドを発行します。

シナリオ 5: バッチの行サイズが 1 で、ローカルステージングファイルの数の値が 1 である場合。

バッチ行サイズが 1 でローカルステージングファイルが 1 つである場合、行サイズが小さく、ステージングファイルのしきい値に到達しないため、行ごとに 4 つの挿入コマンドが個別に発行されます。

マッピングでのプロパティのロードの設定

ターゲットトランスフォーメーションの Snowflake の詳細ターゲットプロパティにある**［追加の書き込みランタイムパラメータ］**フィールドで、Snowflake Data Cloud にデータをロードするための書き込みプロパティを設定できます。

次の表に、Snowflake にデータをロードする場合に指定できる追加のランタイムパラメータの一部を一覧で示します。

プロパティ	説明
oneBatch	すべてのデータを 1 回のバッチで処理します。 タイプはブール値です。 デフォルトは false です。
remoteStage	内部ステージを使用するように指定します。外部ステージは使用されません。 タイプは文字列です。 デフォルトは「~」（ユーザーステージ）です。
onError	ファイルからデータを読み込むときにエラーが発生した場合に実行するアクションを指定します。 例: `onError option ABORT_STATEMENT|CONTINUE|SKIP_FILE` タイプは文字列です。 デフォルトは CONTINUE です。
compressFileByPut	PUT でファイルを圧縮します。 タイプはブール値です。 デフォルトは false です。
compressDataBeforePut	PUT の前にデータを圧縮します。 ローダーは、アップロードする前にデータを gzip 形式に圧縮します。 タイプはブール値です。 デフォルトは true です。
copyEmptyFieldAsEmpty	空の受信フィールドを NULL に設定するための COPY コマンドオプション。 タイプはブール値です。
enablePGZIP	ファイル圧縮の並列処理を有効にします。 タイプはブール値です。 デフォルトは true です。
onError=ABORT_STATEMENT&oneBatch=true	すべてのデータを単一バッチにロードし、エラーが発生した場合はタスクを停止します。同時に、ユーザーが指定した拒否ファイルパスを検証し、エラーレコードをこのファイルとセッションログに書き込みます。 タイプは onError - 文字列または oneBatch - ブール値です。

追加のランタイムパラメータフィールドの値を設定するときに、設定済みの各パーティションは新しいローダーインスタンスを初期化し、設定した値はすべてのパーティションに渡って同様に適用されます。

例 1.ファイルを圧縮する

データを Snowflake にロードする前に、Put コマンドを使用してファイルを圧縮する例を考えます。

次の圧縮オプションを指定します: compressDataBeforePut=false&compressFileByPut=true。

両方のオプションに true を指定した場合、Snowflake は compressDataBeforePut オプションを考慮します。

例 2.空の値を NULL に置き換える

データを Snowflake にロードしている間に、空の値を持つ受信フィールドを NULL に置き換える例を考えます。

copyEmptyFieldAsEmpty Boolean オプションを指定して、要件に応じて値を true または false に設定します。

copyEmptyFieldAsEmpty Boolean パラメータを設定する前に、次のシナリオを考慮します。

- このパラメータを設定しない場合、NULL 値は NULL として受信され、空の値は Empty として受信されます。これはデフォルトの動作です。
- パラメータ copyEmptyFieldAsEmpty=false を設定した場合、NULL 値は NULL として受信され、空の値は NULL として受信されます。
- パラメータ copyEmptyFieldAsEmpty=true を設定した場合、NULL 値は空として受信され、空の値は空として受信されます。

例 3.単一バッチでデータを書き込む

単一バッチ内のソースデータを Snowflake に書き込む例を考えます。ロード中にエラーが発生した場合、そのエラーをキャプチャする必要もあります。

要件に応じて、onError=ABORT_STATEMENT&oneBatch=true プロパティを指定します。

単一バッチでロード中にエラーが発生した場合、Secure Agent は指定された拒否ファイル名があるかどうかをチェックし、COPY コマンドを実行し、拒否ファイルを検証してから、エラー（ある場合）をキャプチャするためのファイル名を渡します。

詳細モードのマッピングを実行するための追加のランタイムパラメータの設定

詳細モードでは、Snowflake にデータを書き込む追加のプロパティを設定できます。ターゲットトランスフォーメーションの**［追加の書き込みランタイムパラメータ］**フィールドにパラメータを指定します。

Snowflake にデータを書き込む場合、次のような追加のランタイムパラメータを設定できます。

autopushdown

オプション。自動クエリプッシュダウンが有効かどうかを決定します。

プッシュダウンを有効にし、クエリが詳細クラスタで実行されると、クラスタアプリケーションはクエリの一部をプッシュして Snowflake で処理するため、これらのクエリのパフォーマンスが最適化されます。

コネクタが Spark の互換バージョンを使用する場合、デフォルトは *on* です。コネクタが Spark の互換バージョンを使用しない場合、デフォルト値は *off* です。

continueOnError

オプション。有効ではないデータを入力すると COPY コマンドが操作を強制終了するかどうかを決定します。例えば、有効ではないバリアントデータ型カラムに JSON 形式を指定します。

値は *on* および *off* です。値を *on* に指定した場合、エラーが発生した場合でも COPY コマンドは続行されます。*off* を指定した場合、エラーが発生すると COPY コマンドは強制終了します。デフォルトは *off* です。

このオプションは *off* にしておくことが推奨されます。そうしないと、データの Snowflake へのコピー中にエラーが発生した場合、データの一部が失われる可能性があります。

並行処理

Secure Agent が Snowflake と詳細クラスタの間でデータをアップロードまたはダウンロードするときに使用するスレッドプールのサイズ。デフォルトは 4 です。

スループットを増加または減少させる必要がない限り、デフォルト値を変更しないでください。高いスループットが必要な場合、並列処理を任意の大きい数に設定しないでください。並列処理を大きい値にすると、望ましくない出力になる可能性があり、操作が遅くなります。

パージ

詳細クラスタから Snowflake へ外部データ転送を経由してデータを転送するときに、作成された一時ファイルを Secure Agent が削除するかどうかを指定します。使用できる値は *on* および *off* です。デフォルトは *off* です。

このパラメータを *off* に設定すると、Secure Agent は一時ファイルを自動的に削除します。パージは、詳細クラスタから Snowflake へのデータ転送に対してのみ機能し、Snowflake から詳細クラスタへのデータ転送では機能しません。このパラメータを *on* に設定すると、Secure Agent は一時ファイルを自動的に削除しません。

usestagingtable

オプション。データのロード操作でステージングテーブルを使用するかどうかを決定します。

Snowflake は、一時的な名前でステージングテーブルを作成します。データのロード操作が正常に実行されると、Snowflake は元のターゲットテーブルを削除し、ステージングテーブルの名前を元のターゲットテーブル名に変更します。データのロード操作が失敗すると、Snowflake はステージングテーブルを削除し、操作前にターゲットテーブルに含まれていたデータは保持されます。

Snowflake ではステージングテーブルを使用することを強く推奨します。ステージングテーブルを作成するには、テーブルを作成するための COPY コマンドを実行するのに十分な権限が必要です。テーブルを作成するための権限がない場合は、ステージングテーブルを使用せずに直接ロードすることができます。

値は *on* および *off* です。**usestagingtable** パラメータを *on* に指定した場合、Snowflake はステージングテーブルを使用します。この値を *off* に指定した場合、Snowflake はデータをターゲットテーブルに直接ロードします。デフォルトは *on* です。

Capturing changed data from CDC sources

You can use Snowflake Data Cloud Connector to write changed data from a CDC source such as Oracle CDC and Oracle CDC V2 and write the changed data to a Snowflake target.

When you configure a mapping, add the CDC sources and then run the associated mapping task to write the changed data to Snowflake. If you define a column as required in the Snowflake target table, map a column in the CDC source to the required column in the Snowflake target in the mapping before you run the task.

When the mapping task processes the changed data from a CDC source, Snowflake Data Cloud Connector creates a state table in Snowflake. When the changed data is received from the CDC source, Snowflake Data Cloud Connector uploads the changed data to the staging table. Then, it generates a `Job_Id` and writes the `Job_Id` to the state table along with the restart information.

The connector then merges the stage table with the actual target table in Snowflake. Each time you run the mapping task, Snowflake Data Cloud Connector creates the state table, if it does not exist, to store the state information.

Snowflake Data Cloud Connector uses the following naming convention for the tables:

- State table name: <MappingTaskID>_<Instance(s)>
- Staging table name: <MappingTaskID>_<TargetInstanceName>

Restrictions

Consider the following restrictions when you capture changed data from CDC sources:

Mapping

You can use a Snowflake target in a CDC mapping to write data from CDC sources. You cannot use a Snowflake source or lookup in a CDC mapping.

staging optimization property

The optimization property that you set for staging data in the DTM of the Secure Agent is not applicable. If you run a mapping with both CDC and staging property enabled, the mapping runs successfully. However, staging optimization is disabled and you can view a message logged in the session logs.

Recovery initialization error

When you run a CDC mapping to write data from a CDC source to a Snowflake target created at runtime, you might encounter the following error:

```
Error occured while initializing CCI State Handler
com.informatica.cci.runtime.internal.utils.impl.CExceptionImpl: Internal error: Recovery Init failed
```

To avoid this error, you must have the grant permissions to create a table in Snowflake.

DTM error

When you run a CDC mapping that uses a Snowflake target connection configured with the ProcessConnDB and ProcessConnSchema parameters in the **Additional JDBC URL Parameters** field, the mapping might fail with the following DTM error:

```
Internal error. The DTM process terminated unexpectedly. Contact Informatica Global Customer Support.
```

To avoid this error, remove the ProcessConnDB and ProcessConnSchema parameters from the **Additional JDBC URL Parameters** field in the Snowflake Data Cloud connection. However, if you want to create temporary stage tables and recovery state tables using the ProcessConnDB and ProcessConnSchema parameters, verify that you have the necessary permissions in the required database and schema.

CDC ソースから読み取るためのマッピングタスクの設定

Snowflake Data Cloud Connector を使用して、CDC ソースから変更データをキャプチャし、変更データを Snowflake ターゲットに書き込むことができます。

CDC ソースをマッピングに追加してから、関連するマッピングタスクを実行して、変更データを Snowflake ターゲットに書き込みます。単一のマッピングで複数のパイプラインを設定して、キャプチャされた変更データを Snowflake ターゲットに書き込むこともできます。

変更データを CDC ソースから Snowflake に書き込むようにマッピングを設定した場合は、Snowflake ターゲットトランスフォーメーションで次のような詳細プロパティを設定できます。

- データベース
- スキーマ
- ウェアハウス
- ロール

- Pre SQL
- Post SQL
- ターゲットテーブルの切り詰め
- テーブル名
- 更新オーバーライド

手順 1.ソースを設定します

Oracle CDC や Oracle CDC V2 などの CDC ソースから読み取るようにソーストランスフォーメーションを設定します。

1. ソーストランスフォーメーションの全般プロパティで、名前と説明を指定します。
2. **［ソース］**タブで、設定済みの任意の CDC 接続を選択し、必要なソースプロパティを指定します。

   ソースオブジェクトにプライマリキーを含めることをお勧めします。

手順 2.ターゲットを設定します

変更されたデータを CDC ソースから Snowflake に書き込むようにターゲットトランスフォーメーションを設定します。

CDC ソースから変更されたデータを書き込む場合は、マッピングで単一の Snowflake Data Cloud ターゲットトランスフォーメーションのみを設定できます。

1. **［ターゲット］**タブで、次の手順を実行してターゲットプロパティを設定します。
   a. **［接続］**フィールドで、Snowflake Data Cloud 接続を選択します。
   b. **［ターゲットの種類］**フィールドで、ターゲットオブジェクトのタイプを選択します。
   c. **［オブジェクト］**フィールドで、必要なターゲットオブジェクトを選択します。
   d. **［操作］**フィールドで、**［挿入］**または**［データドリブン］**を選択します。

      注: ターゲットの更新、更新/挿入、および削除操作は適用されません。ターゲットテーブルに Snowflake で定義されたプライマリキーがないことを確認します。
   e. **［データドリブン条件］**を選択した場合は、**DD_INSERT** 条件を指定します。

      注: ターゲットテーブルに Snowflake で定義されたプライマリキーがないことを確認します。
   f. CDC モードに適用可能な詳細ターゲットプロパティを設定します。
2. **［フィールドマッピング］**タブで、受信フィールドをターゲットフィールドにマップします。手動でマップすることも、フィールド名に基づいて自動的にマップすることもできます。

   Snowflake ターゲットテーブルで必要に応じてカラムを定義する場合は、CDC ソースのカラムをマッピングの Snowflake ターゲットの必要なカラムにマッピングします。

手順 3.マッピングタスクを設定します

マッピングを作成した後に、マッピングをマッピングタスクに追加して、詳細プロパティを設定します。関連するマッピングタスクを実行して、変更されたデータを Snowflake に書き込みます。

1. **［アクション］**メニューで、**［新しいマッピングタスク］**をクリックします。

   ［新しいマッピングタスク］ページが表示されます。
2. **［定義］**タブで、タスク名を入力し、設定済みのマッピングを選択します。
3. **［CDC ランタイム］**タブで、選択した CDC ソースに必要なプロパティを指定します。

［**CDC ランタイム**］プロパティの詳細については、選択した CDC ソースのソースプロパティを参照してください。

4. ［**スケジュール**］タブの［**詳細セッションプロパティ**］セクションで、次のプロパティを追加します。
   a. ［**ファイル終端（EOF）でコミット**］フィールドで、このプロパティの値として［**いいえ**］を選択します。
   b. ［**コミットタイプ**］フィールドで、このプロパティの値として［**ソース**］を選択します。
   c. ［**リカバリストラテジ**］フィールドで、このプロパティの値として［**最終チェックポイントから再開**］を選択します。
5. ［**保存**］ > ［**実行**］をクリックして、マッピングタスクを保存して実行します。

   スケジュールを作成して、手動で操作しなくてもマッピングタスクが定期的に実行されるようにすることもできます。マッピングタスクの実行間隔ができるだけ短くなるようにスケジュールを定義できます。**モニタ**で、タスクの実行後にログのステータスを監視できます。

パフォーマンスを向上させるには、マッピングタスクウィザードの［CDC ランタイム］ページで［コミットあたりの最大行数］プロパティのコミット間隔をより長く設定します。ただし、コミット間隔を長くすると、失敗した場合に回復に時間がかかります。

リカバリメカニズムの無効化

CDC ジョブでエラーが発生したときにリカバリ状態テーブルを使用して変更内容を取得しないようにする場合は、ターゲットおよびマッピングタスクでリカバリを無効にします。

リカバリを無効にするには、次のタスクを実行します。

1. Snowflake Data Cloud ターゲットトランスフォーメーションの［**追加の書き込みランタイムパラメータ**］フィールドで *disableSnowflakeRecoveryCDC=true* プロパティを設定します。
2. マッピングタスクの［**スケジュール**］タブの［**詳細セッションプロパティ**］で、設定したリカバリストラテジを削除します。

注: CDC マッピングのソースが Snowflake ではないことを確認してください。マッピングに Snowflake ソースとターゲットが含まれており、ターゲットに対して *disableSnowflakeRecoveryCDC* プロパティが設定されている場合、マッピングジョブでは正しい結果が表示されません。

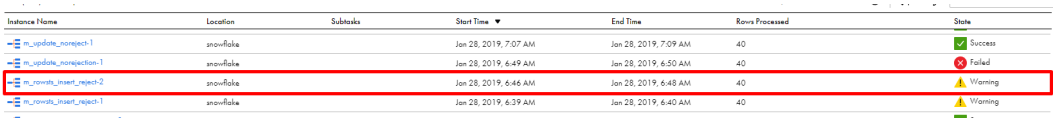
ジョブの統計の表示

各ジョブの統計を表示して、処理された行数とジョブの状態を確認できます。

ジョブは、次のいずれかの状態になります。

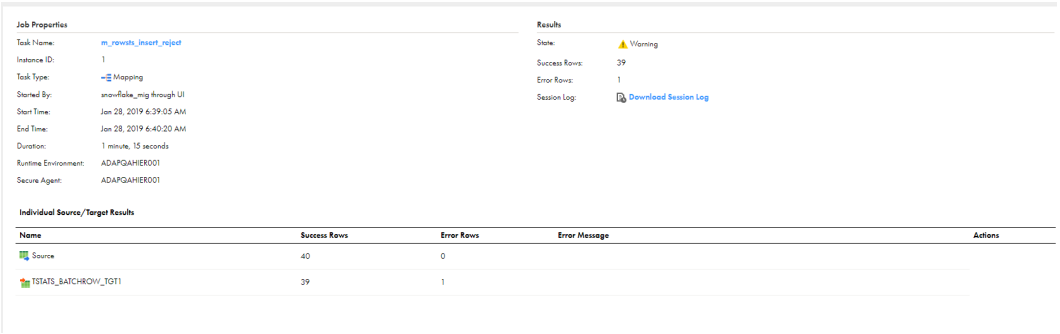
- 成功。Secure Agent が、挿入、更新、および削除操作のすべての行を適用しました。
- 警告。Secure Agent が 1 つ以上の行を拒否しました。［**マイジョブ**］ページの［**処理された行数**］フィールドには、Secure Agent が処理した行の合計数が反映されます。
- 失敗しました。エラーが発生したため、ジョブは完了しませんでした。

次の図は、Snowflake Data Cloud ジョブの状態と処理された行数の詳細を示す［**マイジョブ**］ページを示しています。



処理された行のうち成功した行の数とエラーが発生した行の数を表示するには、特定のインスタンス名を選択して［**結果**］セクションを表示します。成功した行の数とエラーが発生した行の数を表示できます。

次の図は、Snowflake Data Cloud タスクの詳細を示しています。



セッションログをダウンロードして、出力行数、影響を受けた行、適用された行、および拒否された行の詳細を取得できます。

また、Snowflake Data Cloud の書き込み操作に関するターゲット統計の次のようなシナリオが発生する可能性があります。

制約違反

挿入、更新、削除操作のシナリオでは、Secure Agent が制約違反のために行を拒否した場合、［**ジョブプロパティ**］ページに警告が表示されます。ターゲット統計を確認するためにセッションログをダウンロードできます。

一致するものが見つかりません

更新または削除操作を行って、Secure Agent で一部のレコードに一致するレコードが見からない場合、一致しないレコードの数は［**マイジョブ**］ページとセッションログには反映されません。例えば、5 つの入力行があり、Secure Agent が 4 つのターゲット行のみを更新した場合、処理された行の数のステータスは 5 と表示されます。この問題は、Snowflake が拒否された行に対してエラーメッセージを返さない場合に発生します。

非一意の一致

更新または削除操作を行って、Secure Agent が非一意の一致によってより多くの行を更新または削除した場合、更新または削除されたレコードの実際の数は、［**マイジョブ**］ページおよびセッションログには反映されません。例えば、5 つの入力レコードがあり、Secure Agent が 10 個のターゲット行を更新した場合、［**マイジョブ**］ページには処理された行数として 5 行のみが表示されます。

更新されない成功した行

［**ジョブプロパティ**］ページに表示されるターゲットオブジェクトの成功した行数は更新されず、取り込んだ行数が表示されます。例えば、ターゲットに 5 行のレコードを書き込み、2 行のレコードが拒否された場合、成功した行数は 5 と表示されます。

Snowflake Data Cloud ターゲットトランスフォーメーションのルールとガイドライン

Snowflake Data Cloud ターゲットトランスフォーメーションには、次のルールとガイドラインが適用されます。

- 大量のデータを Snowflake に書き込んだ際に却下されたレコードがある場合、却下されたレコードの統計がセッションログに正しく表示されていても、却下されたレコードの一部がファイルに表示されないことがあります。
- 実行時に新しいターゲットを作成するように設定されたマッピングを実行した場合、テーブル名にスラッシュ（/）またはバックスラッシュ（\）が含まれていると、Snowflake ターゲットにデータを書き込むことができません。
- 詳細モードでは、次の制限が適用されます。
  - Snowflake のテーブル名に二重引用符が含まれている場合、マッピングは失敗します。カラム名は二重引用符で囲むことができます。
  - 実行時に作成するターゲットのテーブル名を指定し、別のテーブル名で更新オーバーライドも指定した場合、マッピングでは設計時に指定したテーブル名が考慮されます。
  - ソーストランスフォーメーションのカスタムクエリからストアドプロシージャを呼び出すことはできません。
  - ターゲットトランスフォーメーションで選択された更新カラムに、カラム名に定義された最大長が含まれている場合、タスクは失敗します。カラム名の長さが 75 文字を超えないようにしてください。

第 6 章

Snowflake Data Cloud のルックアップ

指定されたルックアップ条件に基づいてデータを取得するには、ルックアップトランスフォーメーションを追加します。ルックアップトランスフォーメーションをマッピングに追加するときに、Snowflake に関連するルックアップ接続、ルックアップオブジェクト、およびルックアッププロパティを定義します。

選択した条件に基づいて Snowflake のデータ値を検索できます。ルックアップトランスフォーメーションで、ルックアップ接続とオブジェクトを選択します。次に、ルックアップ条件と複数一致の結果を定義します。

マッピングは、ルックアップフィールドと定義されたルックアップ条件に基づいてルックアップソースへのクエリを実行します。ルックアップ操作によって結果がルックアップトランスフォーメーションに返され、ルックアップトランスフォーメーションは結果をダウンストリームに渡します。

以下のルックアップを設定することができます。

- **接続済み。**マッピングには、キャッシュまたはキャッシュを使用しない接続済みのルックアップを使用できます。また、動的ルックアップキャッシュを使用して、ルックアップキャッシュをターゲットと同期し続けるようにすることができます。
- **未接続。**キャッシュされたルックアップを使用できます。式トランスフォーメーションを使用するトランスフォーメーションの:LKP 式から、未接続のルックアップトランスフォーメーションの入力値を指定する必要があります。

注: 詳細モードでは、キャッシュされたルックアップとキャッシュを使用しないルックアップは適用されません。

ルックアップトランスフォーメーションの詳細については、データ統合のドキュメントにある「*トランスフォーメーション*」を参照してください。

Snowflake Data Cloud のルックアッププロパティ

以下の表に、ルックアップトランスフォーメーションで設定できる Snowflake Data Cloud のルックアップオブジェクトプロパティについて説明します。

プロパティ	説明
接続	ルックアップ接続の名前。 既存の接続の選択、新しい接続の作成、またはルックアップ接続プロパティのパラメータ値の定義ができます。 **注**: パラメータ化されていない Snowflake Data Cloud 接続とパラメータ化された Snowflake Data Cloud 接続を切り替えることができます。詳細プロパティ値は、切り替え中も保持されます。 実行時にルックアップ接続プロパティを上書きする場合は、**[実行時にパラメータのオーバーライドを許可する]** オプションを選択します。
ソースタイプ	ソースオブジェクトのタイプ［単一オブジェクト］、［クエリ］[1]、または［パラメータ］を選択します。
パラメータ	タスクを編集せずに更新する値を定義するパラメータファイル。ルックアップオブジェクトに既存のパラメータを選択するか、**[新しいパラメータ]** をクリックしてルックアップオブジェクトに新しいパラメータを定義します。 パラメータをルックアップタイプとして選択する場合にのみ、**パラメータ**プロパティが表示されます。 実行時にパラメータを上書きする場合は、**[実行時にパラメータのオーバーライドを許可する]** オプションを選択します。 タスクを実行すると、そのタスクは詳細セッションプロパティで指定したファイルからパラメータを使用します。
ルックアップオブジェクト	マッピングのルックアップオブジェクトに名前を付けます。
複数一致	ルックアップ条件で複数の一致が返されるときの動作。すべての行、任意の行、最初の行、最後の行、またはエラーを返すようにすることができます。 ルックアップオブジェクトのプロパティで次のオプションから選択して、動作を決定します。 - 最初の行を返す [1] - 最後の行を返す [1] - 任意の行を返す - すべての行を返す - エラーを報告
フィルタ	該当なし。
ソート	該当なし。

[1] 詳細モードのマッピングには適用されません。

以下の表に、ルックアップトランスフォーメーションで設定できる Snowflake Data Cloud ルックアップオブジェクトの詳細プロパティについて説明します。

詳細プロパティ	説明
データベース	接続で指定されたデータベースをオーバーライドします。
スキーマ	接続で指定されたスキーマをオーバーライドします。
ウェアハウス	接続で指定された Snowflake ウェアハウス名をオーバーライドします。 マッピングのウェアハウス名によって、接続で指定したウェアハウス名をオーバーライドします。接続プロパティで不適切なウェアハウス名を指定しても、接続は成功します。ただし、マッピングを実行する前に、マッピングプロパティで正しいウェアハウス名が指定されていることを確認してください。
ロール	接続で指定された、ユーザーに割り当てられた Snowflake ロールをオーバーライドします。
Pre SQL	該当なし。
Post SQL	該当なし。
SQL オーバーライド [1]	Snowflake ソースからデータを読み取るために使用されるデフォルトの SQL クエリをオーバーライドします。

[1] 詳細モードのマッピングには適用されません。

パラメータ化

ルックアップオブジェクトと接続をパラメータ化する場合は、いくつかのルールに従ってオブジェクト名を上書きする必要があります。

トランスフォーメーションで **［実行時にパラメータのオーバーライドを許可する］** オプションを有効にした場合、`db.schema.tablename` などの完全修飾名を使用してオブジェクト名をオーバーライドすることはできません。

Snowflake Data Cloud 接続の **［JDBC URL の追加パラメータ］** フィールドにある `db=<dbname>&schema<schemaname>` 値を渡す必要があります。

複数一致の制限

ルックアップの設定時に、ルックアップ条件が複数一致を返したときの動作を定義します。すべての行、任意の行、最初の行、最後の行、またはエラーを返すようにすることができます。

次の設定には、複数一致のポリシーの制限があります。

- 多重度オプションが［エラーを報告］に設定されている場合、タスクは失敗しません。
- キャッシュを使用しないルックアップを有効にしてルックアップオーバーライドを設定した場合、［最初の行を返す］および［最後の行を返す］の複数一致オプションは適用されません。
- Snowflake ソースに大文字と小文字が区別されるテーブルとカラム名が含まれている場合、キャッシュを使用しないルックアップおよび複数一致用に設定されたマッピングは失敗します。

ルックアップキャッシングの有効化

マッピングでルックアップトランスフォーメーションを設定すると、ランタイムセッション中にルックアップデータをキャッシュできます。

[ルックアップキャッシュの有効化] を選択にすると、データ統合はルックアップソースを一度クエリし、セッション中に使用する値をキャッシュします。これにより、パフォーマンスを向上できます。キャッシュされたルックアップを保存するディレクトリを指定できます。動的ルックアップキャッシュおよび永続ルックアップキャッシュを設定できます。

ルックアップキャッシュの詳細については、データ統合ドキュメントの「*トランスフォーメーション*」にある、ルックアップトランスフォーメーションに関する章を参照してください。

動的ルックアップキャッシュには、次のような制限があります。

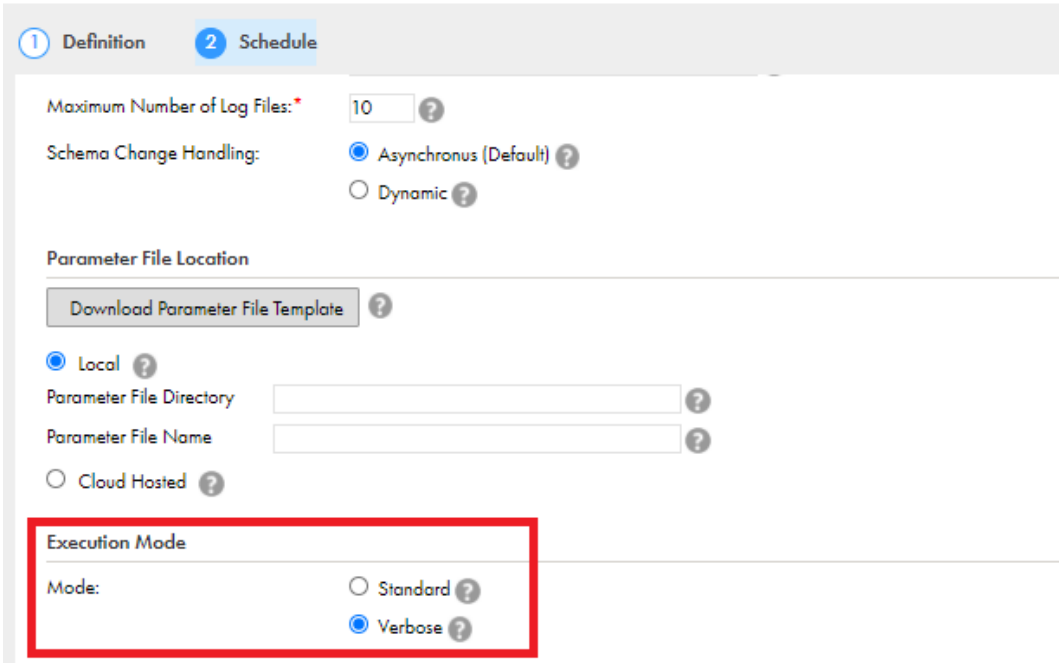
- 動的ルックアップキャッシュを設定する際は、**[複数一致]** プロパティを **[エラーを報告]** に設定します。プロパティをリセットするには、動的ルックアップを静的ルックアップに変更し、プロパティを変更してから静的ルックアップを動的ルックアップに変更します。すべてのルックアップフィールドがマッピングされていることを確認します。
- 動的ルックアップのルックアップ条件は、等しい演算子のみを使用します。
- プッシュダウンの最適化は適用されません。
- 動的ルックアップキャッシュを使用するように設定されているルックアップトランスフォーメーションをパラメータ化することはできません。

キャッシュを使用しないルックアップクエリのロギング

キャッシュを有効にすると、データ統合はルックアップソースへのクエリを 1 回実行し、その結果をキャッシュしてセッション中に使用します。キャッシュを無効にすると、行がトランスフォーメーションに渡されるたびに、SELECT 文がルックアップ値を受け取ります。

データ統合は、接続されたルックアップと未接続のルックアップのキャッシュを使用しないルックアップクエリをセッションログに記録します。マッピングで **[ルックアップキャッシュを有効にする]** プロパティを有効にしてから、Snowflake Data Cloud マッピングタスクで Verbose モードを有効にします。

次の図は、マッピングタスクで選択された Verbose モードを示しています。



マッピングを実行すると、データ統合はキャッシュを使用しないルックアップクエリをセッションログに記録します。

第 7 章

プッシュダウンの最適化

プッシュダウンの最適化を使用して、トランスフォーメーションロジックを Snowflake データベースにプッシュできます。

プッシュダウンの最適化用に設定されたタスクを実行すると、データ統合はトランスフォーメーションロジックを SQL クエリまたは Snowflake コマンドに変換します。データ統合はクエリをデータベースに送信し、データベースでクエリが実行されます。データ統合がデータベースにプッシュするトランスフォーメーションロジックの量は、データベース、トランスフォーメーションロジック、およびマッピング設定によって異なります。データ統合は、データベースにプッシュできないすべてのトランスフォーメーションロジックを処理します。

タスクのプロパティで、マッピングにプッシュダウンの最適化を設定できます。マッピングまたは詳細モードのマッピングを参照するタスクで、プッシュダウンの最適化を設定できます。

プッシュダウンの最適化のタイプ

プッシュダウンの最適化を適用すると、タスクのプロパティで指定した最適化のタイプに基づいて、データ統合は、トランスフォーメーションロジックをソースデータベースまたはターゲットデータベースにプッシュします。データ統合は、選択した接続に基づいて、トランスフォーメーションロジックを SQL クエリに変換するか、Snowflake コマンドを Snowflake データベースに変換します。

ソースデータベースまたはターゲットデータベースで SQL クエリまたは Snowflake コマンドが実行され、トランスフォーメーションが処理されます。データベースにプッシュできるトランスフォーメーションロジックの量は、データベース、トランスフォーメーションロジック、マッピング設定、およびセッション設定によって異なります。データ統合は、データベースにプッシュできないすべてのトランスフォーメーションロジックを処理します。

マッピングおよび詳細モードのマッピングでは、次のタイプのプッシュダウン最適化を設定できます。

完全

データ統合は、ターゲットデータベースで処理するために可能な限り多くのトランスフォーメーションロジックをプッシュダウンします。

Snowflake との間で読み取りおよび書き込みを行うマッピングの場合、データ統合はソースからターゲットへのすべてのトランスフォーメーションを分析します。ターゲットにおいてすべてのトランスフォーメーションに互換性がある場合、マッピングロジック全体がターゲットにプッシュされます。マッピングロジック全体をターゲットにプッシュできない場合、データ統合は最初にできるだけ多くのトランスフォーメーションロジックをソースデータベースにプッシュし、次にできるだけ多くのトランスフォーメーションロジックをターゲットデータベースにプッシュします。

注: ソースとターゲットの Snowflake アカウントが個別に異なるリージョンに存在していても、同じクラウドプラットフォームでホストされている場合は、完全なプッシュダウンの最適化を設定できます。ただ

し、Snowflake アカウントユーザーとターゲット Snowflake アカウントのロールが Snowflake ソースアカウントにアクセスできることを確認してください。

ソース

データ統合は、ソースデータベースで処理するために可能な限り多くのトランスフォーメーションロジックをプッシュダウンします。

プッシュダウンの最適化のシナリオ

Snowflake Data Cloud Connector をマッピングおよび詳細モードのマッピングで使用して、プッシュダウンの最適化を設定できます。

以下のシナリオに対するプッシュダウンの最適化を設定することができます。

ソースエンドポイントとターゲットエンドポイント	説明
Snowflake ソース Snowflake ターゲット	Snowflake Data Cloud 接続を使用した Snowflake テーブルに対する読み取りと書き込み。Snowflake の外部テーブル、ビュー、マテリアライズドビューからも読み取りができます。 プッシュダウンの最適化タイプはソースまたは完全です。
Amazon S3 ソース Snowflake ターゲット	Amazon S3 V2 接続を使用して Amazon S3 から読み取り、Snowflake Data Cloud 接続を使用して Snowflake に書き込みます。 プッシュダウンの最適化タイプは完全です。
Google Cloud Storage ソース Snowflake ターゲット	Google Cloud Storage V2 接続を使用して Google Cloud Storage から読み取り、Snowflake Data Cloud 接続を使用して Snowflake に書き込みます。 プッシュダウンの最適化タイプは完全です。 詳細モードのマッピングには適用されません。
Microsoft Azure Data Lake Storage Gen2 ソース Snowflake ターゲット	Microsoft Azure Data Lake Storage Gen2 V2 接続を使用して Microsoft Azure Data Lake Storage Gen2 から読み取り、Snowflake Data Cloud 接続を使用して Snowflake に書き込みます。 プッシュダウンの最適化タイプは完全です。
Secure Agent、Hosted Agent、またはサーバーレスランタイム環境を使用して、マッピングロジックをデータベースにプッシュできます。	

Snowflake Data Cloud 接続を使用したプッシュダウンの最適化の準備

Snowflake Data Cloud マッピングにプッシュダウンの最適化を設定して Google Cloud Storage または Microsoft Azure Data Lake Storage Gen2 から Snowflake にデータを読み込む前に、特定の要件を満たす必要があります。

Google Cloud Storage バケット内のデータファイルへのアクセス

Google Cloud Storage から Snowflake にデータをロードするようにプッシュダウンの最適化を設定する前に、Snowflake でクラウドストレージ統合オブジェクトを作成します。

ストレージ統合には、データの読み取り元となる有効な Google Cloud Storage バケットの詳細が含まれています。Snowflake Data Cloud Connector は、作成したクラウドストレージ統合を使用する一時的な外部ステージを作成します。

Snowflake でクラウドストレージ統合を作成した後に、Snowflake Data Cloud 接続プロパティでクラウドストレージ統合の名前を指定します。**[JDBC URL の追加パラメータ]** フィールドで名前を指定します。ストレージ統合の値は大文字と小文字が区別されます。

Google Cloud Storage のストレージ統合の設定

Snowflake が Google Cloud Storage バケットからデータを読み取れるようにするためのストレージ統合を作成します。

Snowflake ドキュメントを参照して、次の手順を実行できます。

1. Snowflake でクラウドストレージ統合を作成します。
2. Snowflake アカウント用の Cloud Storage Service アカウントを取得します。
3. バケットオブジェクトにアクセスするためのアクセス権をサービスアカウントに付与します。
   a. カスタム IAM ロールを作成します。
   b. カスタムロールを Cloud Storage Service アカウントに割り当てます。
4. 外部ステージを作成するための権限をロールに付与します。

   ロールには、スキーマに対する CREATE STAGE 特権と、ストレージ統合に対する USAGE 特権が必要です。

   例えば、次のようなコマンドを実行して、これらの特権を付与します。

   ```
   grant create stage on schema public to role myrole;
   grant usage on integration gcs_int to role myrole;
   ```

Microsoft Azure Data Lake Storage Gen2 コンテナ内のデータファイルへのアクセス

Microsoft Azure Data Lake Storage Gen2 から Snowflake にデータをロードするようにプッシュダウンの最適化を設定する前に、Snowflake でクラウドストレージ統合オブジェクトを作成します。

ストレージ統合には、データの読み取り元となる有効な Microsoft Azure Data Lake Storage Gen2 コンテナの詳細が含まれています。Snowflake Data Cloud Connector は、作成したクラウドストレージ統合を使用する一時的な外部ステージを作成します。

Snowflake でクラウドストレージ統合を作成した後に、Snowflake Data Cloud 接続プロパティでクラウドストレージ統合の名前を指定します。**[JDBC URL の追加パラメータ]** フィールドで名前を指定します。ストレージ統合の値は大文字と小文字が区別されます。

Microsoft Azure Data Lake Storage Gen2 のストレージ統合の設定

Snowflake が Microsoft Azure Data Lake Storage Gen2 コンテナからデータを読み取れるようにするためのストレージ統合を作成します。

Snowflake ドキュメントを参照して、次の手順を実行できます。

1. Snowflake でクラウドストレージ統合を作成します。

2. Snowflake アカウント用の Cloud Storage Service アカウントを取得します。「格納場所へのアクセス権の付与」 (ページ 64)を参照してください。
3. バケットオブジェクトにアクセスするためのアクセス権をサービスアカウントに付与します。
   a. カスタム IAM ロールを作成します。
   b. カスタムロールを Cloud Storage Service アカウントに割り当てます。
4. 外部ステージを作成するための権限をロールに付与します。

   ロールには、スキーマに対する CREATE STAGE 特権と、ストレージ統合に対する USAGE 特権が必要です。

   例えば、次のようなコマンドを実行して、これらの特権を付与します。

```
grant create stage on schema public to role myrole;
grant usage on integration adls_int to role myrole;
```

格納場所へのアクセス権の付与

Snowflake サービスのプリンシパルアクセスを Azure サービスのストレージアカウントに付与します。

1. DESCRIBE INTEGRATION コマンドを実行して、次の同意 URL を取得します: `desc storage integration <integration_name>;`

   ここで、integration_name は、作成した統合の名前です。
   AZURE_CONSENT_URL カラムの URL の形式は次のとおりです。

   `https://login.microsoftonline.com/<tenant_id>/oauth2/authorize?client_id=<snowflake_application_id`

   AZURE_MULTI_TENANT_APP_NAME カラムの値をコピーします。これは、アカウント用に作成された Snowflake クライアントアプリケーションの名前です。この情報は、格納場所のアクセストークンを取得するために必要なアクセス権をこのアプリケーションに付与するために必要です。
2. Web ブラウザで、AZURE_CONSENT_URLURL カラムの URL に移動します。

   ページに、Microsoft のアクセス権要求ページが表示されます。
3. **［承認］**をクリックします。

   これにより、Snowflake アカウント用に作成した Azure サービスプリンシパルが、テナント内の任意のリソースでアクセストークンを取得できるようになります。アクセストークンは、コンテナに対する適切な権限をサービスプリンシパルに付与した場合にのみ正常に生成されます。
4. Microsoft Azure ポータルにログインします。
5. **［Azure サービス］** > **［ストレージアカウント］**に移動し、Snowflake サービスのプリンシパルアクセスを許可するストレージアカウントの名前をクリックします。
6. **［アクセス制御（IAM）］** > **［ロールの割り当ての追加］**をクリックします。
7. Snowflake サービスプリンシパルに付与する必要のあるロールを選択します。
   - Storage Blob Data Reader: 読み取りアクセスのみを許可します。ストレージアカウントにステージングされたファイルからデータをロードできます。
   - Storage Blob Data Contributor: 読み取りアクセスおよび書き込みアクセスを許可します。ストレージアカウントにステージングされたファイルからのデータのロードやファイルへのデータのアンロードができます。
8. Snowflake サービスプリンシパルを検索します。

   これは、手順 1 の DESC STORAGE INTEGRATION 出力の AZURE_MULTI_TENANT_APP_NAME プロパティの ID です。Microsoft 要求ページから要求した Snowflake サービスプリンシパルが Azure で作成されるまでに 1 時間以上かかる場合があります。サービスプリンシパルがすぐに利用できない場合は、1〜2 時間待ってから、もう一度検索を実行することをお勧めします。サービスプリンシパルを削除すると、ストレージ統合は機能しなくなります。

9. **［保存］**をクリックします。

   ロールの割り当てが有効になるまでに 5 分ほどかかる場合があります。

プッシュダウンの最適化のプレビュー

プッシュダウンの最適化用に設定されたマッピングタスクを実行する前に、マッピングの作成時にプッシュダウン最適化が可能かどうかをプレビューできます。プッシュダウンの最適化は、Mapping Designer の**［プッシュダウンの最適化］**パネルでプレビューできます

プッシュダウンの最適化に関する必要なオプションを選択し、プレビューを実行すると、データ統合により、一時的なプッシュダウンのプレビューマッピングタスクが作成および実行されます。ジョブが完了すると、データ統合によって、実行する SQL クエリと警告が**［プッシュダウンの最適化］**パネルに表示されます。警告メッセージは、設定されたマッピングのどのトランスフォーメーションがプッシュダウンの最適化に該当しないかを判断するのに役立ちます。プッシュダウンの最適化が失敗すると、失敗の時点までに生成されたクエリが一覧表示されます。プッシュダウンの最適化のマッピングを実行する前に、マッピングを編集したり、必要なトランスフォーメーションを修正することが可能です。

［マイジョブ］で作成された一時ジョブを表示し、セッションログをダウンロードして、生成されたクエリを確認することもできます。

プッシュダウンの最適化をプレビューする方法については、データ統合のドキュメントにある、「*マッピング*」のトピック「プッシュダウンの最適化のプレビュー」を参照してください。

プッシュダウンの最適化の設定

マッピングを最適化するには、マッピングをタスクに追加してから、マッピングタスクでプッシュダウンの最適化を設定します。

1. マッピングタスクを作成します。
2. **［スケジュール］**タブの**［プッシュダウンの最適化］**セクションで、プッシュダウンの最適化の値を**［完全］**または**［ソースへ］**に設定します。
3. 完全なプッシュダウンの最適化が利用できない場合は、**［プッシュダウンの最適化のフォールバックオプション］**メニューで、データ統合がプッシュダウンの最適化を処理する方法を選択します。
   - 部分的な PDO。デフォルト。データ統合はトランスフォーメーションロジックを可能な限りソースデータベースとターゲットデータベースにプッシュします。データベースにプッシュできないトランスフォーメーションロジックすべてが処理されます。部分的な PDO は、Snowflake に対して読み取りと書き込みを行う場合にのみ使用できます。
   - PDO なし。このタスクはプッシュダウンの最適化を使用せずに実行されます。
   - タスクの失敗。データ統合はタスクに失敗します。

   注: フォールバックオプションは、詳細モードのマッピングには適用されません。

マッピングタスクを実行するときに、設定されたデータベースにトランスフォーメーションロジックがプッシュされます。マッピングが最適化されたことを確認するには、ジョブのセッションログをチェックします。モニタにジョブのログを表示します。

複数のターゲットに対するコンテキストベースの最適化

Snowflake Data Cloud 接続を使用して複数の Snowflake ターゲットに書き込むように完全なプッシュダウンの最適化を有効にするマッピングを設定した場合、複数挿入で緩やかに変化する次元タイプ 2 マージシナリオの最適化コンテキストを指定できます。

[最適化コンテキストタイプ] は、タスクの **[スケジュール]** タブで指定できます。指定した最適化コンテキストに基づいて、データ統合は、複数のターゲットから発行されたクエリを組み合わせてプッシュダウンの最適化のための単一のクエリを構築し、タスクを最適化します。

マッピングで指定したターゲット操作に基づいて、タスクプロパティで以下の最適化モードを有効にできます。

複数挿入

Snowflake ソースから複数の Snowflake ターゲットにデータを挿入する場合は、このモードを有効にします。データ統合は、ターゲットごとに生成されたクエリを組み合わせて、単一のクエリを発行します。

SCD **タイプ** 2 **マージ**

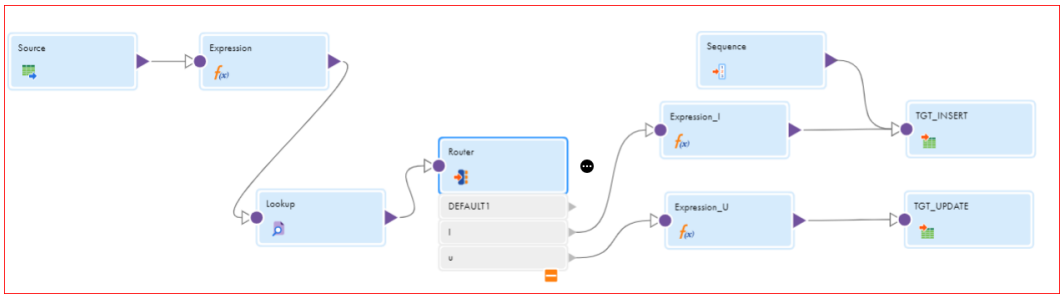
2 つの Snowflake ターゲットに書き込む場合は、このモードを有効にします。一方のターゲットを使用してデータを挿入し、もう一方のターゲットを使用してデータを更新します。データ統合は、両方のターゲットのクエリを組み合わせて、マージクエリを発行します。

デフォルトは［なし］です。

SCD タイプ 2 マージマッピングに関する説明

SCD タイプ 2 マージマッピングは、Snowflake ソースと同じ Snowflake テーブルに書き込む 2 つのターゲットトランスフォーメーションを使用します。一方のターゲットトランスフォーメーションはテーブルを更新し、もう一方のトランスフォーメーションはデータを Snowflake テーブルに挿入します

次の図は、緩やかに変化するディメンションデータを Snowflake ターゲットテーブルに書き込むマッピングを示しています。



ルックアップと式トランスフォーメーションを追加して、ソースデータを既存のターゲットデータと比較します。データ統合で既存のターゲットと比較するルックアップ条件とソースカラムを入力します。

ターゲット内に一致するプライマリキーがない各ソース行に対して、式トランスフォーメーションにより新しい行が示されます。ターゲット中に一致するプライマリキーのあるソース行では、Expression はユーザ定義のソースカラムとターゲットカラムを比較します。これらのカラムが一致しない場合には、Expression は行に変更があったというフラグを設定します。その後、このマッピングは 2 つのデータフローに分かれます。

最初のデータフローは、ルータートランスフォーメーションを使用して、新しい行のみを式トランスフォーメーションに渡します。式トランスフォーメーションは、ターゲットに新しい行を挿入します。シーケンスジェネレータは各行のプライマリキーを作成します。式トランスフォーメーションは、キー間の増分値を 1,000 ずつ増やし、それぞれの新規行に対して 0 というバージョン番号を作成します。

2 番目のデータフローでは、ルータートランスフォーメーションは変更された行のみを式トランスフォーメーションに渡します。式トランスフォーメーションは、変更された行をターゲットに挿入します。式トランスフォーメーションによって、キーおよびバージョン番号が 1 ずつ増加します。

制限

SCD タイプ 2 マージマッピングでは、フィルタ、ジョイナ、およびカスタム SQL クエリを使用できません。

異なるスキーマを持つデータベースへのロジックのプッシュ

マッピングタスクにクロススキーマプッシュダウンの最適化を使用して、同じ Snowflake データベース内にある異なるスキーマに関連付けられた Snowflake オブジェクトに対してデータを読み書きできます。

クロススキーマプッシュダウンの最適化を使用するには、2 つの接続を作成し、各接続にスキーマを指定します。ソース接続のスキーマとターゲット接続のスキーマがそれぞれ異なり、両方のスキーマが同じデータベースに属している必要があります。

クロスデータベースプッシュダウンの最適化の設定

Snowflake データベース内のさまざまなスキーマの処理ロジックをプッシュするには、マッピングタスクでクロススキーマプッシュダウンの最適化を設定します。

1. マッピングタスクを作成します。
   a. 設定されたマッピングを選択します。
   b. **［スケジュール］**タブまたは**［ランタイムオプション］**の**［プッシュダウンの最適化］**のセクションで、プッシュダウンの最適化の値を**［完全］**に設定します。
2. クロススキーマプッシュダウンの最適化を有効にするには、**［クロススキーマプッシュダウンの最適化の有効化］**を選択します。

   デフォルトでは有効になっています。
3. タスクを保存し、**［完了］**をクリックします。

別のデータベースへのロジックのプッシュ

クロスデータベースのプッシュダウンの最適化を有効にして、複数の Snowflake データベースに分散しているデータに対してクエリを実行できます。

マッピングタスクで、クロスデータベースプッシュダウン最適化を設定できます。マッピング内の Snowflake ソースおよびターゲットトランスフォーメーションが 2 つの異なる Snowflake Data Cloud 接続または Snowflake ODBC 接続を使用していることを確認してください。

クロスデータベースプッシュダウン最適化の設定

異なる Snowflake データベース間で処理ロジックをプッシュするには、マッピングタスクでクロスデータベースのプッシュダウンの最適化を設定します。

1. マッピングタスクを作成します。
   a. 設定されたマッピングを選択します。
   b. **［スケジュール］**タブまたは**［ランタイムオプション］**の**［プッシュダウンの最適化］**のセクションで、プッシュダウンの最適化の値を**［完全］**に設定します。
2. データベース間でのプッシュダウンを有効にするには、**［スケジュール］**タブの**［詳細オプション］**で、**［データベース全体でのプッシュダウンの許可］**に**［セッションプロパティ値］**を選択し、**［セッションプロパティ値］**を**［はい］**に設定します。

プッシュダウンの最適化ジョブのクリーンな停止

プッシュダウンの最適化が有効になっているタスクが実行されている場合は、ジョブにクリーンな停止を使用して、ジョブによって生成されたすべての発行済みの文とプロセスを終了できます。

データ統合の［マイジョブ］ページと［監視］の［すべてのジョブと実行中のジョブ］ページの**［クリーンな停止］**オプションを使用します。

プッシュダウンの最適化タスクにクリーンな停止を使用する場合は、事前に以下の例外を参照してください。

- Snowflake からの読み取りまたは Snowflake への書き込みを行うソースプッシュダウンの最適化が有効になっているタスクにクリーンな停止を使用し、マッピングのターゲットプロパティまたはソースプロパティに Pre-SQL 文または Post-SQL 文が含まれている場合、選択クエリが終了しても、ジョブはターゲットの Post-SQL クエリを実行し続けます。
- 実行時に新しいターゲットを作成し、ジョブにすぐにクリーンな停止使用するように設定されたマッピングを実行すると、ジョブが終了した場合でも、データ統合によってターゲットテーブルが作成されます。

第 8 章

Snowflake Data Cloud 接続を使用したプッシュダウンの最適化（SQL ELT）

Snowflake Data Cloud 接続を使用して、マッピングタスクにプッシュダウンの最適化（SQL ELT）を設定できます。プッシュダウンの最適化により、タスクのパフォーマンスが向上します。

プッシュダウンの最適化用に設定されたタスクを実行すると、トランスフォーメーションロジックが Snowflake クエリに変換されます。タスクはクエリを Snowflake に送信し、マッピングロジックが Snowflake データベースで処理されます。

次のシナリオでは、マッピングまたは詳細モードのマッピングにプッシュダウンの最適化を設定できます。

Snowflake **から** Snowflake

Snowflake Data Cloud 接続を使用した Snowflake に対する読み取りと書き込み。

Amazon S3 **から** Snowflake

ソーストランスフォーメーションの Amazon S3 V2 接続を使用した Amazon S3 からの読み取りと、ターゲットトランスフォーメーションの Snowflake Data Cloud 接続を使用した Snowflake への書き込み。

Microsoft Azure Data Lake Storage Gen2 **ソースから** Snowflake

ソーストランスフォーメーションで Microsoft Azure Data Lake Storage Gen2 接続を使用して Microsoft Azure Data Lake Storage Gen2 から読み取り、ターゲットトランスフォーメーションで Snowflake Data Cloud 接続を使用して Snowflake に書き込みます。

Google Cloud Storage **から** Snowflake

ソーストランスフォーメーションの Google Cloud Storage 接続を使用した Google Cloud Storage からの読み取りと、ターゲットトランスフォーメーションの Snowflake Data Cloud 接続を使用した Snowflake への書き込み。

注: 詳細モードのマッピングには適用されません。

例

あなたはヘルスケアソリューションで働いており、組織は薬局や薬局チェーンにヘルスケアテクノロジを提供しています。薬局が処方箋を処理し、医療記録へのアクセスを保存および提供して、患者の転帰を改善できるようにするとします。組織はデータを Google Cloud Storage に保存しています。

経営陣は、患者中心の薬局管理システムを作成したいと考えています。この組織は、Snowflake のウェアハウスインフラストラクチャを活用し、そのすべてのデータを Snowflake にロードして、運用、財務、および臨床上の意思決定を容易に行えるようにしようと計画しています。

データを Google Cloud Storage オブジェクトから Snowflake にロードするには、データウェアハウスモデルをサポートする必要なトランスフォーメーションで ETL および ELT を使用する必要があります。

Google Cloud Storage V2 接続を使用して Google Cloud Storage バケットからデータを読み取り、Snowflake Data Cloud 接続を使用して Snowflake ターゲットに書き込みます。パフォーマンスを最適化するためにマッピングで完全なプッシュダウンの最適化を設定します。Google Cloud Storage ソースデータは PUT コマンドを使用して Snowflake ステージへアップロードされます。Snowflake COPY コマンドは、データを Snowflake にロードするときに、トランスフォーメーションを対応する SQL 関数および式に変換するのに使用されます。プッシュダウンの最適化により、タスクのパフォーマンスが向上し、関係するコストが削減されます。

Snowflake Data Cloud 接続のプッシュダウンの互換性

トランスフォーメーション、変数、関数、および演算子をデータベースにプッシュするようにタスクを設定できます。

プッシュダウンの最適化を使用する場合、Secure Agent はデータベース内で同等の演算子、変数、関数を特定することで、トランスフォーメーションの式を変換します。対応する演算子、変数、および関数が存在しない場合、Secure Agent はトランスフォーメーションロジックを処理します。

Snowflake Data Cloud での関数

プッシュダウンの最適化を使用する場合、データ統合はデータベース内で同等の関数を特定することで、トランスフォーメーションの式を変換します。対応する関数が存在しない場合は、データ統合がトランスフォーメーション論理を処理します。

次の表に、完全なプッシュダウンの最適化を使用して Snowflake にプッシュできるプッシュダウン機能の可用性を示します。

機能	機能	機能
ABS()	IS_SPACES	SIN()
ASCII()	LAST_DAY()	SINH()
ADD_TO_DATE()	LENGTH()	SQRT()
AVG()	LN()	STDDEV()*
CEIL()	LOG()	SUBSTR()
CHR()	LOWER()	SUM()
CONCAT()	LPAD()	SYSDATE()
COS()	LTRIM()	SYSTIMESTAMP()
COSH()	MAX()	TAN()
COUNT()	MAKE_DATE_TIME	TANH()
DATE_COMPARE()	MEDIAN()	TO_BIGINT

機能	機能	機能
DATE_DIFF()	MIN()	TO_CHAR(DATE)
DECODE()	MOD()	TO_CHAR(NUMBER)
EXP()	POWER()	TO_DATE()
FLOOR()	REG_EXTRACT()	TO_DECIMAL()
GET_DATE_PART()	REG_MATCH()	TO_FLOAT()
IIF()	REG_REPLACE	TO_INTEGER()
IN()	REPLACECHR()	TRUNC(DATE)
INITCAP()	REPLACESTR()	TRUNC(NUMBER)
INSTR()	ROUND(NUMBER)	UPPER()
IS_DATE	RPAD()	MD5()
IS_NUMBER	RTRIM()	VARIANCE()
ISNULL()	SIGN()	-
*STDDEV()関数では、フィルタ条件の引数を渡すことはできません。		

注: Snowflake の完全なプッシュダウンの最適化でサポートされていない関数を指定すると、タスクは部分的なプッシュダウンの最適化を使用して、または完全なプッシュダウンの最適化を使用せずに実行されます。

Snowflake Data Cloud での演算子

プッシュダウンの最適化を使用する場合、データ統合はデータベース内で同等の演算子を特定することで、トランスフォーメーションの式を変換します。対応する演算子が存在しない場合は、データ統合がトランスフォーメーションロジックを処理します。

次の表に、Snowflake にプッシュできる演算子を示します。

演算子	演算子
+	>=
-	<=
*	! =
/	AND
%	OR
||	NOT
>	IS NULL

演算子	演算子
<	IS NOT NULL
=	

Snowflake Data Cloud での変数

完全なプッシュダウンを使用して、SESSSTARTTIME 変数を Snowflake データベースにプッシュできます。

Snowflake Data Cloud でのトランスフォーメーション

プッシュダウンの最適化を設定すると、データ統合は設定済みのトランスフォーメーションを Snowflake にプッシュするように試みます。

フルプッシュダウンを使用して、次のようなトランスフォーメーションを Snowflake にプッシュできます。

- アグリゲータ
- 式
- フィルタ
- 階層プロセッサ
- ジョイナ
- ルックアップ
- ノーマライザ
- ランク
- ルータ
- シーケンスジェネレータ
- SQL*
- ソータ
- 共有体
- アップデートストラテジ*

*詳細モードのマッピングには適用されません。

注: ルータートランスフォーメーションはソースのプッシュダウンの最適化にのみ適用でき、階層プロセッサトランスフォーメーションは詳細モードのマッピングにのみ適用できます。

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にある「*トランスフォーメーション*」を参照してください。

アグリゲータトランスフォーメーション

完全なプッシュダウンの最適化を設定して、Snowflake で処理するアグリゲータトランスフォーメーションをプッシュできます。

集計計算

次の集計計算を実行できます。

- AVG
- COUNT
- MAX
- MIN
- MEDIAN
- SUM
- VARIANCE

受信ポート

アグリゲータトランスフォーメーションを設定し、受信ポートが集計関数またはマッピングのグループ化フィールドで使用されていない場合、グループ化または集計関数の一部ではないカラムに対して ANY_VALUE()関数が使用されます。この場合、ANY_VALUE()関数はポートから任意の値を返すため、出力は確定的ではありません。ただし、詳細モードでは、受信ポートがグループ化フィールドの一部でない場合、MAX()関数が使用されます。また、アグリゲータトランスフォーメーションにより、値が 1 の追加のカラムが生成されます。ただし、このカラムはプッシュダウンクエリの挿入部分では削除され、使用されません。

式トランスフォーメーション

完全なプッシュダウンの最適化を設定し、式トランスフォーメーションをプッシュして Snowflake で処理することができます。

マッピング内の各ソースに式トランスフォーメーションを追加してから、マッピング内のダウンストリームに結合することができます。さらに、トランスフォーメーションから分岐してから、マッピングのダウンストリームのトランスフォーメーションに分岐する複数の式トランスフォーメーションを追加できます。

式トランスフォーメーションを設定するときは、次のルールを考慮して、変数を式に含めてください。

- 前の行の処理中に割り当てられた値を現在の行の計算に使用している場合、変数を使用することはできません。この場合、マッピングはプッシュダウンの最適化なしで実行されます。
- 変数はネストできますが、式で定義する前に変数を参照することはできません。
  変数がこの順序で定義されていない場合、マッピングはプッシュダウンの最適化なしで実行されます。

  例:

  ```
  var: AGEPLUS2 = AGEPLUS1 + 1
  var: AGEPLUS1 = AGE + 1
  out: NEXTAGE = AGEPLUS2 + 1
  ```

  ここで、AGE +1 は後で定義されます。最初の変数の AGEPLUS2 は AGEPLUS1 を参照しており、未解決のままです。
  これを解決するには、次の順序で変数を指定します。

  ```
  var: AGEPLUS1 = AGE + 1
  var: AGEPLUS2 = AGEPLUS1 + 1
  out: NEXTAGE = AGEPLUS2 + 1
  ```
- 変数は、循環型の式またはそれ自体を参照する式を持つことはできません。

例:

```
 var: AGEPLUS1 = AGEPLUS2 + 1
var: AGEPLUS2 = AGEPLUS1 + 1
out: NEXTAGE= AGEPLUS2
```

ここで、AGEPLUS1 は AGEPLUS2 を参照し、未解決のままです。

階層プロセッサトランスフォーメーション

詳細モードでは、Amazon S3 V2 または Microsoft Azure Data Lake Storage Gen2 ソースから階層入力またはリレーショナル入力を読み取り、リレーショナルまたは階層出力として Snowflake ターゲットに書き込むように階層プロセッサトランスフォーメーションを設定できます。

階層プロセッサトランスフォーメーションは、構造または配列を表す階層フィールドを処理します。

次のような制限付きで階層プロセッサトランスフォーメーションを設定できます。

- 階層出力フィールドの配列要素に固定小数点数値データ型が含まれている場合、マッピングはプッシュダウンの最適化なしで実行されます。
- Double データ型の 10 進数値は指数で表記されます。
  例えば、Double データ型の 2341.6789 を Snowflake ターゲットの出力フィールドに書き込むと、出力は 2.341678900000000e+03 と表示されます。
- **［入力グループまたは受信フィールドの使用］**をデータソースとして選択し、複数の行を含むソースから階層入力またはリレーショナル入力を読み取り、階層出力として Snowflake ターゲットに書き込む場合、データ統合は行ごとにレコードを複製します。
  重複した行がターゲットに書き込まれることを避けるには、**［親のデータソースの継承］**を選択するか、フィルタ条件を使用してデータソースから子フィールドをフィルタリングします。
- Integer データ型または Bigint データ型を構造フィールドから Snowflake ターゲットに書き込むには、**［セッションプロパティ名］**リストから **［advanced.custom.property］**を選択し、マッピングタスクに次の値を入力します。
  `DisableAdvancedMappingRuntimeValidation=true`

ルックアップトランスフォーメーション

完全なプッシュダウンの最適化を設定して、ルックアップトランスフォーメーションをプッシュし、Snowflake で処理できます。接続済みのルックアップおよび未接続のルックアップをプッシュできます。

マッピングに未接続のルックアップが含まれている場合は、未接続のルックアップ関数を他の式関数とネストすることもできます。例えば、`:LKP.U_LOOKUP(Upper(argument1), argument)`のようにすることができます。

ルックアップオブジェクト

ルックアップを設定するときは、次のルールを考慮してください。

- ソーストランスフォーメーションが次のソースを使用する場合、Snowflake のルックアップを設定できます。

  - Amazon S3

  - Google Cloud Storage*

  - Microsoft Azure Data Lake Storage Gen2

  - Snowflake ソース

  *詳細モードのマッピングには適用されません。

- 対応する Amazon S3、Google Cloud Storage、Microsoft Azure Data Lake Storage Gen2、または Snowflake ソースをソーストランスフォーメーションが使用している場合にのみ、Amazon S3、Google Cloud Storage、Microsoft Azure Data Lake Storage Gen2、または Snowflake オブジェクトにルックアップを設定できます。

接続済みのルックアップ

Snowflake Data Cloud のソースとターゲットを使用したマッピングで、ルックアップオブジェクトの **[複数一致]** オプションを **[すべての行を返す]** に設定します。

未接続のルックアップ

接続されていないルックアップトランスフォーメーションを設定する場合は、次のルールを考慮してください。

- 未接続のルックアップから受信した出力には式を設定しないでください。
- 接続されていないルックアップトランスフォーメーションからの入力フィールドの一部が Snowflake ターゲットオブジェクトにマッピングされていない Snowflake Data Cloud ソースとターゲットを使用したマッピングでは、選択クエリにマッピングされていないすべてのフィールドが含まれます。

接続されたルックアップと未接続のルックアップでの複数一致の動作

接続済みのルックアップまたは未接続のルックアップを含むマッピングに対してプッシュダウンの最適化を有効にする場合は、次のガイドラインに従う必要があります。

- 未接続のルックアップでは、**[複数一致]** オプションを必ず **[エラーを報告]** に選択してください。データを検索し、検索条件で複数の一致が見つかった場合、一致するすべての行が選択され、プッシュダウンの最適化でタスクが実行されます。**[エラーを報告]** 以外のオプションに対して **[複数一致]** を有効にした場合、マッピングはプッシュダウンの最適化なしで実行されます。
- 接続済みのルックアップでは、**[複数一致]** オプションを **[すべての行を返す]** または **[エラーを報告]** に設定します。**[複数一致]** オプションを **[エラーを報告]** に設定すると、タスクの詳細セッションプロパティで *Lkp_apdo_allow_report_error* カスタムフラグを設定して、データ統合が複数の一致を処理する方法を決定できます。
  - このプロパティを *[はい]* に設定し、データ内に複数の一致がある場合、複数一致ポリシーは無視され、ジョブはプッシュダウンの最適化で正常に実行されます。
  - このプロパティを設定せず、データ内に複数の一致がある場合、データ統合はポリシーを考慮して警告メッセージを表示します。プッシュダウンの最適化は無視され、タスクは失敗します。

FileName ポート

Amazon S3 ソースと Snowflake ターゲットを含むマッピングで Amazon S3 ソースのルックアップを設定した場合は、Amazon S3 ソースおよびルックアップオブジェクトから FileName ポートを削除します。FileName ポートは使用できません。

ルックアップクエリオブジェクト

Snowflake のデータをルックアップするためのマッピングでルックアップトランスフォーメーションのクエリとしてルックアップオブジェクトを使用する場合は、詳細ルックアッププロパティまたは Snowflake Data Cloud 接続の追加の JDBC URL パラメータでデータベースとスキーマを指定する必要があります。

ノーマライザトランスフォーメーション

複数回発生するデータのインスタンスごとに行を返すように、マッピングでノーマライザトランスフォーメーションを設定できます。

例えば、リレーショナルソースに、四半期の売上データが格納された 4 つのフィールドがあるとします。この場合、四半期ごとに別個の出力行を生成するように、ノーマライザトランスフォーメーションを設定することができます。

詳細モードでは、次のような制限付きでノーマライザトランスフォーメーションを設定できます。

- 異なるグループから複数回発生するフィールドを正規化するには、各グループを一意のターゲットにマッピングし、グループがターゲットにマッピングされていない場合でも、異なるグループからのすべてのフィールドを受信フィールドのリストから除外します。
- 正規化されたデータを複数のターゲットに書き込むには、正規化されたフィールドの生成されたすべてのカラム ID を対応するターゲットにマッピングします。

ルータートランスフォーメーション

ソースプッシュダウンの最適化を設定して、ルータートランスフォーメーションをデータベースにプッシュして処理することができます。

ルータートランスフォーメーションを設定する場合は、1 つの出力グループのみをターゲットトランスフォーメーションに接続またはマッピングします。

シーケンスジェネレータトランスフォーメーション

シーケンスジェネレータトランスフォーメーションを設定すると、シーケンスジェネレータトランスフォーメーションに続くトランスフォーメーションで NEXTVAL ポートを単一または複数のポートに接続できます。

次のような制限付きで、シーケンスジェネレータトランスフォーメーションをプッシュできます。

- シーケンスジェネレータトランスフォーメーションで共有シーケンスを使用することはできません。
- マッピング内のシーケンスジェネレータトランスフォーメーションの後に追加できるのは、式またはターゲットトランスフォーメーションのみです。

詳細モードのマッピングでシーケンスジェネレータトランスフォーメーションを 使用するためのガイドライン

詳細モードでのシーケンスジェネレータトランスフォーメーションは、マッピングと同じルールに従います。ただし、詳細モードのマッピングでシーケンスジェネレータトランスフォーメーションを使用するには、Snowflake のシーケンスの名前を使用して、マッピングタスクプロパティにプロパティを追加で設定する必要があります。

1. タスクプロパティの **[スケジュール]** タブで、**[詳細セッションプロパティ]** セクションに移動します。
2. カスタムプロパティを追加し、次のセッションプロパティ値を次の形式で入力します。

   ```
   Pushdown.<Sequence transformation name in mapping>.SequenceName=<sequence name to create in Snowflake>
   ```

   例:

   ```
   Pushdown.Seq_SF.SequenceName=snowflake_first_sequence&:Pushdown.sequence_second.SequenceName=SECOND_SEQUENCE&:Pushdown.seq_third.SequenceName=THIRD_SEQ
   ```

複数のシーケンスジェネレータトランスフォーメーションを使用する場合、シーケンスごとに Snowflake データベースに作成されるシーケンスオブジェクトの名前は一意である必要があります。名前が一意でない場合、トランスフォーメーションは CREATE SEQUENCE IF NOT EXISTS クエリで同じ名前を使用して 2 回実行されます。最初のシーケンスは想定どおりに実行されますが、最初のシーケンスと同じ名前を持つ 2 番目のシーケンスによって最初のシーケンスがオーバーライドされることはありません。したがって、2 番目のシーケンスに生成される nextValue は誤った値となります。ただし、シーケンスオブジェクトに対して一意の名前を使用してシーケンスを実行した場合、マッピングを実行する回数に関係なく、そのシーケンスが再度更新されることはありません。

注: プッシュダウンの最適化を無効にしてマッピングを実行すると、シーケンス値は最初の開始値から再開されます。マッピングを削除した場合は、Snowflake データベースからもシーケンスを削除する必要があります。

SQL トランスフォーメーション

SQL トランスフォーメーションは、特定の関数と共有シーケンスをプッシュするためにのみ使用できます。

関数を使用したクエリの実行

入力したクエリに関数を SQL トランスフォーメーションに含めて、Snowflake ターゲットエンドポイントでクエリを実行できます。

関数をプッシュするには、SELECT 句の SQL 文のみを使用する必要があります。選択クエリまたは関数でカラム名を指定します。「SELECT * FROM TABLE」などの文を使用して関数をプッシュしないでください。

入力したクエリでは、次のような関数を使用できます。

- UUID_STRING
- RANDOM
- RANDSTR
- SIGN
- CURRENT_REGION
- CURRENT_ACCOUNT
- CURRENT_ROLE
- CURRENT_USER
- CURRENT_DATABASE
- CURRENT_SCHEMA
- DAYNAME
- SPLIT
- SPLIT_PART

SQL トランスフォーメーションで CURRENT_ROLE、CURRENT_DATABASE、および CURRENT_SCHEMA 関数を使用するには、Snowflake Data Cloud 接続の追加の JDBC パラメータフィールドにデータベース、ロール、およびスキーマ名を指定します。接続に値を指定しない場合、データ統合によってターゲットに NULL が挿入されます。

共有シーケンスの再利用

SQL トランスフォーメーションで定義した共有シーケンスを使用して、マッピングを Snowflake エンドポイントにプッシュできます。データ統合は、Snowflake ソースと同じ順序でデータをターゲットに書き込みます。

Snowflake から共有シーケンスを取得し、SQL トランスフォーメーションで入力したクエリでシーケンスを定義します。

入力したクエリの共有シーケンスを次の構文で指定します: Select
`<Snowflake_schema_name>.<Snowflake_database_name>.<sequence_name>.NEXTVAL`

ユーザー定義関数

Snowflake の Java または SQL ユーザー定義関数（UDF）から読み取るように、SQL トランスフォーメーションでカスタムクエリを設定できます。

UDF には次のガイドラインが適用されます。

- UDF 名に改行文字が含まれた UDF を読み取ることはできません。
- UDF に配列パラメータが含まれている場合、マッピングはプッシュダウンの最適化なしで実行されます。

共有体トランスフォーメーション

次のような制限付きで、共有体トランスフォーメーションをプッシュできます。

- マッピングのソーストランスフォーメーションには、Snowflake ソースオブジェクトのみを含める必要があります。
- ソースが Amazon S3、Google Cloud Storage、または Microsoft Azure Data Lake Storage Gen2 である場合、マッピングはプッシュダウンの最適化を使用せずに実行されます。

アップデートストラテジトランスフォーメーション

アップデートストラテジトランスフォーメーションを使用することはできません。

代わりにターゲットトランスフォーメーションの更新および更新/挿入操作を使用して、Snowflake に書き込みを行うことができます。

機能

次のソースから読み取り、Snowflake ターゲットに書き込むマッピングまたは詳細モードのマッピングにプッシュダウンの最適化を設定できます。

- Snowflake ソース
- Amazon S3 ソース
- Google Cloud Storage ソース [1]
- Microsoft Azure Data Lake Storage Gen2 ソース

注: [1] 詳細モードのマッピングには適用されません。

マッピングを設定する場合、プッシュダウンの最適化が有効になっているマッピングについては、一部のパラメータがサポートされません。各ソースがサポートするパラメータのリストを参照してください。

Snowflake Data Cloud のソース、ターゲット、およびルックアップ

プッシュダウンの最適化を設定する場合、ソース、ターゲット、およびルックアップトランスフォーメーションでサポートされている Snowflake Data Cloud プロパティのこのリストを参照してください。

ソースのプロパティ

Snowflake ソーストランスフォーメーションでは次のプロパティを設定できます。

- ソース接続 - パラメータ、実行時にパラメータのオーバーライドを許可する
- ソースタイプ - 単一オブジェクト、複数オブジェクト、クエリ、およびパラメータ。パラメータファイルを使用して、マッピングタスクから、マッピング内の Snowflake のソース接続とオブジェクトをオーバーライドすることもできます。

  注: クエリソースタイプを使用して Snowflake から読み取った場合は、フィールドメタデータを保持してマッピングを保存するという選択ができます。クエリを編集してマッピングを実行した場合でも、設計時に指定したフィールドメタデータは保持されます。
- 実行時にパラメータのオーバーライドを許可する。

- クエリオプション - フィルタと結合。簡易フィルタ条件と詳細フィルタ条件を使用できます。結合を使用して、既存のリレーションに基づいて関連するオブジェクトを結合したり、詳細リレーションを作成したりすることができます。
- データベースオーバーライド
- スキーマオーバーライド
- ウェアハウスオーバーライド
- Pre-SQL および Post-SQL
- ロールオーバーライド
- テーブル名のオーバーライド
- SQL オーバーライド
- トレースレベル

次のソースプロパティは、完全なプッシュダウン最適化が有効になっているマッピングには適用されません。

- ウェアハウスオーバーライド
- Pre-SQL および Post-SQL
- ロールオーバーライド

次のソースプロパティは、Snowflake ソーストランスフォーメーションには適用されません。

- クエリオプション - ソート
- パーティション

ターゲットプロパティ

マッピングには複数の Snowflake ターゲットを追加できます。ターゲットには、複数回追加された同じ Snowflake ターゲットテーブル、または異なる Snowflake ターゲットテーブルを設定することができます。

Snowflake ターゲットトランスフォーメーションでは次のプロパティを設定できます。

- ターゲット接続 - パラメータ、実行時にパラメータのオーバーライドを許可する。
- ターゲットタイプ - 単一オブジェクト、パラメータ。パラメータファイルを使用して、マッピングタスクから、マッピング内の Snowflake のターゲット接続とオブジェクトをオーバーライドすることもできます。
- 実行時にパラメータのオーバーライドを許可する
- ターゲットオブジェクト - 既存のターゲット、実行時に新規作成。
- 操作 - 挿入、更新、更新/挿入、削除、またはデータドリブン
- データベースオーバーライド
- スキーマオーバーライド
- ウェアハウスオーバーライド
- ロールオーバーライド
- Pre-SQL および Post-SQL
- テーブル名のオーバーライド
- ターゲットテーブルの切り詰め
- 追加の書き込みランタイムパラメータ

次のターゲットプロパティは、Snowflake ターゲットトランスフォーメーションには適用されません。

- 更新モード
- バッチ行のサイズ
- ローカルステージングファイルの数
- 拒否ファイルパス
- 更新オーバーライド
- 拒否された行の転送

注: 詳細モードのマッピングで複数のターゲットに書き込む場合は、挿入操作のみを使用できます。

ルックアップのプロパティ

プッシュダウンの最適化を有効にすると、Snowflake の接続済みおよび未接続のルックアップに次のプロパティを設定できます。

- ルックアップ接続 - パラメータ、実行時にパラメータのオーバーライドを許可する
- ソースタイプ - 単一オブジェクト、クエリ、パラメータ。パラメータファイルを使用して、マッピングタスクから、マッピング内の Snowflake のルックアップ接続とオブジェクトをオーバーライドすることもできます。
- 複数一致 - エラーを報告
- データベースオーバーライド
- スキーマオーバーライド
- ウェアハウスオーバーライド
- ロールオーバーライド
- テーブル名のオーバーライド
- SQL オーバーライド
- トレースレベル

次のルックアッププロパティは、Snowflake の接続済みのルックアップと未接続のルックアップには適用されません。

- Pre SQL
- Post SQL

マッピングのガイドライン

マッピングを設定する場合は、次のガイドラインを考慮してください。

- 実行時に作成されたターゲットの場合は、Snowflake ソースに Time データ型のレコードが含まれていないことを確認してください。
- フィルタを設定する場合は、次のガイドラインを考慮してください。
  - マッピングにフィルタトランスフォーメーションがあり、ソーストランスフォーメーションにフィルタがある場合、マッピングは、これらのトランスフォーメーションのフィルタ条件を統合してレコードをフィルタリングします。ただし、マッピングで一度に使用するフィルタは 1 つのみにすることをお勧めします。
  - フィルタでシステム変数を使用することはできません。
  - クエリや、複数のソースオブジェクトにフィルタを適用することはできません。
  - 式トランスフォーメーションで IS_date 関数を設定する場合は、この関数の形式を指定します。関数の形式を指定しない場合、マッピングによって誤ったデータが取り込まれます。

- 2 つの Snowflake ターゲットに書き込むように 2 つのシーケンスジェネレータトランスフォーメーションを設定し、カスタムプロパティでシーケンスオブジェクトのシーケンス名が同じである場合、データは正しく取り込まれません。

- Snowflake に対して読み取りと書き込みを行うマッピングについては、次のガイドラインを考慮してください。
  - クエリを使用してストアドプロシージャから読み取ることはできません。
  - ソーストランスフォーメーションで Snowflake String データ型の精度を下げて Snowflake テーブルに書き込んだ場合でも、マッピングは切り詰めを行わずにデータを渡します。
  - ソースまたは部分的なプッシュダウンの最適化のためのマッピングを設定する際は、ソーストランスフォーメーションをマッピング内の複数のトランスフォーメーションに接続しないでください。ただし、完全なプッシュダウンの最適化が有効になっているマッピングでは、ソーストランスフォーメーションを、マッピングパイプライン内の複数のトランスフォーメーションに分岐できます。
  - ソーストランスフォーメーションでカスタムクエリを設定して、Snowflake の Java または SQL ユーザー定義関数（UDF）から読み取りを行うことができます。
  - 完全またはソースプッシュダウンの最適化を使用してマッピングを実行すると、セッションログ内のクエリの一部が正しくエイリアス化されません。単純クエリのエイリアスは正しく反映されます。
  - テーブル名とカラム名で大文字と小文字が区別され、特殊文字と Unicode 文字が含まれている場合、マッピングまたは詳細モードのマッピングは、関連オブジェクトを使用して結合された複数のテーブルからデータを読み取ることはできません。
  - 同じデータベースとスキーマに属していない複数の Snowflake オブジェクトから読み取りを行うマッピングは失敗します。
  - is_number 関数を使用すると、Snowflake の inf、inf、NaN などの一部の値に入力されるデータは、プッシュダウンの最適化が適用されている場合と適用されていない場合で異なります。
  - トランスフォーメーションで IS_NUMBER 関数を使用し、入力データに（例えば、+3.45d+32 または +3.45D-32 などの形式で）d または D が含まれている場合、関数は False または 0 を返します。
  - トランスフォーメーションで IS_DATE 関数を使用する場合は、J、MM/DD/YYYY SSSSS、MM/DD/Y、および MM/DD/RR 形式を使用しないでください。
  - テーブル名またはカラム名にマルチバイト文字を使用して Snowflake からの読み取りまたは Snowflake への書き込みを行うマッピングは、失敗する可能性があります。マルチバイト文字を含むデータの読み取りまたは書き込みを行うようにマッピングを設定する前に、Secure Agent プロパティの JVM オプションで *-DdisablePDOAdvancedAliasing* プロパティを設定します。
  - ノーマライザトランスフォーメーションで NULL 値を持つカラムを渡すと、NULL 値はターゲットに書き込まれません。
  - **[一時的なビューの作成]** プロパティで有効になっているマッピングタスクから date1 および date2 引数を指定して DATE_DIFF()関数をプッシュダウンすると、この関数はプッシュダウンの最適化なしで実行されるマッピングとは異なる次の値を返します。
    - date1 の値が date2 の値よりも後である場合、この関数は負の数を返します。
    - date1 の値が date2 の値よりも前である場合、この関数は正の数値を返します。

    正しい戻り値を取得するには、Administrator で Secure Agent の JVM オプションを *-DFixSnowflakeDateDiffForPDO=true* に設定します。
  - 受信フィールドの精度がターゲットフィールドの精度を超えると、マッピングまたは詳細モードでのマッピングでターゲットへのデータの書き込みが失敗します。

Amazon S3 V2 ソース

マッピングは、Amazon S3 V2 接続について、次のプロパティをサポートします。

- アクセスキー
- 秘密鍵

マッピングは、Amazon S3 V2 ソースについて、次のプロパティをサポートします。

- ソース接続パラメータ
- ソースタイプ - 単一、パラメータ
- 形式 - 区切り、Avro、ORC、Parquet、および JSON
- ソースタイプ - ファイルとディレクトリ
- フォルダパス
- ファイル名
- 圧縮形式。- Gzip

Amazon S3 V2 ソースから読み取り、Snowflake ターゲットに書き込むプッシュダウンの最適化が有効になっているマッピングには、いくつかの制限があります。

認証

IAM 認証が有効になっている Amazon S3 接続を使用して複数の Avro ファイルを読み取る場合は、Amazon S3 接続で正しいアクセスキーとシークレットキーを指定します。詳細については、Amazon S3 V2 コネクタのヘルプを参照してください。

実行時の新しいターゲットの作成

実行時に新しいターゲットを作成するマッピングには、次のような制限があります。

- Amazon S3 からの Avro、ORC、Parquet などのファイルデータタイプから Snowflake にデータを書き込む場合は、**［ファイル名］**フィールドを削除する必要があります。
- テーブル名に Unicode 文字が含まれている場合、マッピングはキャストエラーで失敗します。

データ型

マッピングには、特定のデータ型に対する次のような制限があります。

- 特殊文字を含む Avro ファイルを書き込むことはできません。
- バイナリデータ型を含むデータを書き込むことはできません。
- 特殊文字を含む JSON 形式のデータを読み取ることはできません。識別子の使用の詳細については、Snowflake のドキュメントの Identifiers Syntax を参照してください。
- S3 ファイル形式にエスケープ文字を指定すると、エスケープ文字はデフォルトで円記号になります。
- 1523 年の ORC ファイルは 1524 年として誤ってロードされます。
- Parquet ファイルからの Time データ型のデータを Amazon S3 から Snowflake に書き込む場合、時間の値はターゲットと異なります。
- JSON データの精度は、Snowflake ターゲットテーブルの精度を超えないようにする必要があります。
- Amazon S3 ソースタイプがディレクトリであり、ディレクトリに対してワイルドカード文字を有効にすると、マッピングは失敗します。読み取ったワイルドカード文字がプッシュダウンの最適化ではサポートされていないことを示す警告メッセージが表示されます。

サポートされているプロパティを設定する方法については、Amazon S3 V2 コネクタのドキュメントを参照してください。

Google Cloud Storage V2 ソース

マッピングは、Google Cloud Storage V2 ソース接続について、次のプロパティをサポートします。

- ソース接続、接続パラメータ
- ソースタイプ - 単一、パラメータ
- 形式 - 区切り、Avro、Parquet、および JSON
- Google Cloud Storage パス
- ソースファイル名
- ディレクトリ

注: 詳細モードでは、ソースとしての Google Cloud Storage V2 はプッシュダウンの最適化には適用されません。

サポートされているプロパティを設定する方法については、Google Cloud Storage V2 コネクタのドキュメントを参照してください。

Microsoft Azure Data Lake Storage Gen2 ソース

マッピングは、Microsoft Azure Data Lake Storage Gen2 ソース接続について、次のプロパティをサポートします。

- アカウント名
- ファイルシステム名

マッピングは、Microsoft Azure Data Lake Storage Gen2 ソースについて、次のプロパティをサポートします。

- ソース接続、接続パラメータ
- ソースタイプ - 単一、パラメータ
- 形式 - 区切り、Avro、Parquet、JSON、および ORC
- 形式オプション
- ファイルシステム名のオーバーライド
- ソースタイプ - ファイル、ディレクトリ
- ディレクトリのオーバーライド - 絶対パス、相対パス
- ファイル名のオーバーライド - ソースオブジェクト
- 圧縮形式 - フラットファイルを読み取るための GZip
- トレースレベル

サポートされているプロパティを設定する方法については、Microsoft Azure Data Lake Storage Gen2 コネクタのドキュメントを参照してください。

ソースオーバーライドの一時ビューの作成

カスタムクエリまたは SQL オーバーライドを Snowflake にプッシュするには、タスクによって、提供されたカスタムクエリに基づいて一時ビューを作成し、一時ビューを使用してプッシュダウンクエリを生成する必要があります。

カスタムクエリをソースオブジェクトとして設定する前、または SQL オーバーライドを設定する前に、次のタスクを実行します。

［スケジュール］タブの**［プッシュダウンの最適化］**セクションで、**［一時的なビューの作成］**を選択します。

注: ［一時的なビューの作成］プロパティを設定しない場合、マッピングはプッシュダウンの最適化なしで実行されます。

ターゲットコピーコマンドオプションの設定

COPY コマンドオプションを指定して、Amazon S3 から Snowflake にデータをロードできます。

ターゲットトランスフォーメーションの Snowflake Data Cloud 詳細ターゲットプロパティの**［追加の書き込みランタイムパラメータ］**フィールドにオプションを指定します。

複数のコピーコマンドオプションを指定する場合、各オプションをアンパサンド（*&*）で区切ります。

注: コピーオプション *MATCH_BY_COLUMN_NAME* は使用できません。

サポートされるコピーコマンドオプションの詳細については、次の Web サイトで Snowflake のマニュアルを参照してください: https://docs.snowflake.com/en/sql-reference/sql/copy-into-table.html

セッションログのプッシュダウンクエリの検証

プッシュダウンの最適化がマッピングの実行中に適用されたことを確認するには、ジョブのセッションログをチェックします。モニタにジョブのログを表示します。

セッションログのクエリをチェックして、マッピングがプッシュダウンの最適化を適用したかどうかを確認します。

例えば、完全なプッシュダウンの最適化が有効になっているマッピングのセッションログに次のクエリが生成されています。

```
39   OPT_63306 [2020-08-06 08:57:23.875] [Full Pushdown optimization is supported between the source and the target system.].
40   OPT_63306 [2020-08-06 08:57:23.875] [Validating mapping for Pushdown Optimization.].
41   OPT_63309 [2020-08-06 08:57:23.875] Pushdown Optimization was successfully enabled.


86   MAPPING> SNOWFLAKECLOUDDATAWAREHOUSE_10000 [2020-08-06 08:57:40.523] [INFO] Inserting data into Snowflake target table : INSERT INTO
     "DB_PC_AUTO"."SCHEMA_PC_AUTO"."TEST1_TGT"("F1_INT") SELECT (SINH((t0."F1_INT"::NUMBER(38,0))::DOUBLE)::NUMBER(18,0))::DOUBLE FROM "CQA"."CQA_SCHEMA"."TEST1_SRC"
     AS t0
```

この例では、生成された SQL に、単一の文としてデータベースにプッシュダウンされた、Insert Into および Select クエリの両方が含まれています。

セッションログにはプッシュダウンのステータスが表示されます。この詳細をチェックして、エラーをトラブルシューティングできます。

例えば、セッションログには、クエリの次のエラーの詳細が表示されます。

```
OPT_63306 [2020-09-10 14:09:05.716] [Source level filter is not supported for pushdown optimization.].
OPT_63307 [2020-09-10 14:09:05.716] Pushdown optimization failed at Analyze phase.
OPT_63310 [2020-09-10 14:09:05.716] Failed to enable Pushdown Optimization.
```

マッピングのプッシュダウンの最適化を有効にしていない場合、読み取りおよび書き込み操作に対して個別の select および insert 文が生成されます。

```
READER_1_1_1>SNOWFLAKECLOUDDATAWAREHOUSE_1000 [2020-09-10 14:09:29.4781] [INFO]
The Snowflake Connector uses the following SQL query to read data: SELECT "DEPTID", "DEPTNAME" FROM "DEPT"
```

```
WHERE
( 'DEPT'."DEPTID" >=103) ORDER BY "DEPT"."DEPTOD" desc
```

詳細モードのマッピングでプッシュダウンの最適化を有効にする場合は、次のサンプルクエリを使用して Snowflake との間で読み取りおよび書き込みを行います。

```
INSERT INTO "DB_PC_AUTO"."SCHEMA_PC_AUTO"."ALL_DATA_TYPES_TARGET"("ID")
     SELECT
            ALL_0."ID" as c0
     FROM "DB_PC_AUTO"."SCHEMA_PC_AUTO"."ALL_DATA_TYPES"
      AS ALL_0
```

トラブルシューティング

日付と時刻の情報を含む文字列カラムからデータを書き込むマッピングを実行すると、マッピングによって日付と時刻の情報が Snowflake **ターゲットに誤って書き込まれます。**

この問題を解決するには、Secure Agent の JVM オプションを-DHonorInfaDateFormat=true に設定します。

Administrator で JVM オプションを設定するには、次の手順を実行します。

1. **［Administrator］** > **［ランタイム環境］**を選択します。
2. **［ランタイム環境］**ページで、マッピングを実行する Secure Agent マシンを選択します。
3. **［編集］**をクリックします。
4. **［システム構成の詳細］**セクションで、**［サービス］**として **［データ統合サーバー］**を選択し、**［タイプ］**として **［DTM］**を選択します。
5. **［JVMOption］**システムプロパティを編集し、値を-DHonorInfaDateFormat=true に設定します。
6. **［保存］**をクリックします。

第 9 章

マッピングの移行

ある環境で接続とマッピングを設定した後で、別の環境に移行してマッピングを実行することができます。

また、詳細モードで設定したマッピングを移行することもできます。移行後は、Administrator サービスから接続プロパティを変更できますが、マッピングを変更する必要はありません。データ統合は、以前の環境で設定されたランタイム属性を使用して、新しい環境でマッピングを正常に実行します。

移行の計画

移行されたマッピングを新しい組織で実行する場合は、同じまたは異なる接続エンドポイントまたはオブジェクトパスを使用することをお勧めします。要件に応じて、マッピングの移行を実行する前に、この節のガイドラインを考慮してください。

次の表に、マッピングを移行するときに保持できる、オブジェクトタイプの設定済みの詳細プロパティを示します。

オブジェクトタイプ	詳細プロパティ
単一オブジェクト	データベース、スキーマ、ロール、テーブル
複数のオブジェクト	データベース、スキーマ、ロール
クエリ	SQL オーバーライド、ロール **注**: SQL オーバーライドとカスタムクエリは完全修飾されたものである必要があります。

動的スキーマ処理用に設定されたマッピングを移行することもできます。

同じパス内でのマッピングの移行

移行されたマッピングで以前の環境と同じオブジェクトパスを使用する場合は、以前の環境と新しい環境の両方の Snowflake アカウントで、同じデータベース、スキーマ、およびテーブルを維持する必要があります。

例えば、組織 1 に使用されるアカウント 1 と組織 2 に使用されるアカウント 2 という 2 つの異なるアカウントがある場合、データベース、スキーマ、およびテーブル名のオブジェクトパスは、両方のアカウントで同じである必要があります。

アカウント 1: DB1/SCHEMA1/TABLE1

アカウント 2: DB1/SCHEMA1/TABLE1

詳細プロパティのデータベース、スキーマ、ロール、およびテーブルをオーバーライドする必要はありません。接続で使用されるロールがテーブル 1 にアクセスできる場合、マッピングは正常に実行されます。

別のパスへのマッピングの移行

別のオブジェクトパスを使用して、新しい環境からマッピングを実行できます。

マッピングを移行する前に、オブジェクトメタデータ、ランタイム属性、または接続属性を変更して、移行された環境にオブジェクトパスを反映させることができます。新しい環境でマッピングを編集または更新する必要はありません。

原則として、詳細プロパティ、接続、またはオブジェクトのプロパティでデータベース、スキーマ、またはテーブルを指定すると、データ統合は次の優先順位で属性を使用します。

1. **ランタイムの詳細属性**。データベース、スキーマ、テーブル、マッピングのソース、ターゲット、またはルックアップトランスフォーメーションのロールなどの詳細プロパティ。
2. **接続属性**。接続プロパティの［**JDBC URL の追加パラメータ**］で設定されたデータベースやスキーマなどの属性。
3. **オブジェクトメタデータ**。マッピングのソース、ターゲット、またはルックアップトランスフォーメーションで選択されたオブジェクト。

移行オプション

別のパスへの移行を行う場合は、次のオプションのいずれかを選択して、オブジェクトパスを更新できます。

オプション 1: 接続プロパティを更新して、新しいオブジェクトを参照する

マッピングをインポートする場合、例えば組織 1 から組織 2 などの新しい組織にインポートする場合は、組織 2 の指定したデータベース、スキーマ、およびテーブルへのアクセスが可能な接続にマップするように、［**接続の確認**］セクションにある既存の接続を変更できます。

オプション 2: 詳細プロパティからプロパティをオーバーライドする

移行の前に、組織 1 のマッピングの詳細プロパティで、新しい組織のオブジェクトに必要なデータベース、スキーマ、テーブル名、およびロールを指定します。

移行後にマッピングを実行すると、Secure Agent は設定された詳細パラメータを使用して、指定されたオブジェクトを組織 1 からインポートされたマッピングでオーバーライドします。

オプション 3: マッピングのプロパティをパラメータ化する

移行前に、データベース、スキーマ、テーブル名、およびロールなどの詳細属性をパラメータ化するという選択ができます。マッピングでは、入力パラメータ、入出力パラメータ、およびパラメータファイルを設定できます。パラメータファイルを使用する場合は、パラメータファイルをローカルマシンまたはクラウドでホストされているディレクトリに保存できます。マッピングを移行した後は、マッピングを編集または更新しないでください。データベース、スキーマ、テーブル名、ロールなどの詳細属性に入出力パラメータを使用した場合は、パラメータファイルからこれらのパラメータを更新できます。

詳細プロパティのみをパラメータ化し、オブジェクトはパラメータ化しない

詳細プロパティのみをパラメータ化し、実行時にそれらのプロパティを使用する場合は、マッピングのオブジェクトプロパティでプレースホルダオブジェクトを選択してから、詳細プロパティからこのプレースホルダオブジェクトへのオーバーライドを指定します。プレースホルダオブジェクトに、オーバーライドとして指定した対応するテーブルと同じメタデータが含まれていることを確認してください。マッピングを実行すると、詳細プロパティで指定した値でプレースホルダオブジェクトがオーバーライドされます。

オブジェクトと詳細プロパティをパラメータ化する

Snowflake オブジェクトタイプと詳細フィールドをパラメータ化したままにする場合は、パラメータの追加時の入力パラメータウィンドウで［**実行時にパラメータのオーバーライドを許可**］オプションを選択さ

れていない状態のままにし、必要なオブジェクトをタスクレベルで選択する必要があります。タスクを実行すると、詳細プロパティで指定した値が優先されます。

パラメータ化は、**[ターゲットの作成]** オプションを使用しているマッピングには適用されません。詳細モードでは、入力パラメータのみが移行に適用されます。

移行の制限

アセットを移行する前に、一部の設定についていくつかのルールを検討する必要があります。

複数のソースオブジェクトを含むマッピングの移行

ソースオブジェクトの詳細プロパティでデータベースとスキーマをオーバーライドする場合は、選択されたデータベースとスキーマがすべてのソースオブジェクトで同じである必要があります。

例えば、マッピングに次のオブジェクトが含まれているとします。

object1: <db1>.<schema1.<object1>

object2: <db1>.<schema1.object2>

例えば、object1 のデータベースとスキーマが<db1>.<schema1.object1>で object2 が<db2>.<schema2.object2>のようにデータベースとスキーマが異なる場合、移行は実行されません。

マッピングに複数のオブジェクトが含まれている場合は、詳細プロパティからテーブルをオーバーライドしないでください。

詳細フィルタとテーブル名のオーバーライドを含むマッピングの移行

マッピングのクエリオプションセクションにオブジェクトの詳細フィルタ（<Table2>など）が含まれ、詳細プロパティにテーブル名へのオーバーライドが含まれている場合は、次のルールを考慮してテーブルをオーバーライドします。

マッピングでは、次のいずれかのオプションを検討してください。

- マッピングの詳細フィルタに、*$$Filtercondition* という値が含まれている必要があります。マッピングタスクでは、タスクプロパティのパラメータファイルの値をオーバーライドできます。パラメータファイルのパラメータ条件に、次の条件が含まれている必要があります: $$Filtercondition =<Table2>.id >= 1
- 詳細プロパティでオーバーライドとして指定したテーブル名を、詳細フィルタ条件で直接使用します。例: <Table2>.id >= 1

詳細モードで、オブジェクトの詳細プロパティでオーバーライドとして指定したテーブル名を詳細フィルタ条件で直接使用します。例: <Table2>.id >= 1

SQL オーバーライドとカスタムクエリを含むマッピングの移行

マッピングで SQL オーバーライドを指定し、使用するオブジェクトタイプがカスタムクエリである場合、カスタムクエリで指定したテーブルと SQL オーバーライドに同じメタデータが含まれていることを確認してください。使用するカスタムクエリと SQL オーバーライドは完全修飾されたものである必要があります。例: Select * from DB1.Schema.Table1

マッピングに SQL オーバーライドとカスタムクエリの組み合わせが含まれている場合は、マッピングの詳細ソースプロパティと Snowflake 接続の［追加の JDBC URL パラメータ］フィールドでデータベースまたはスキーマを指定しないでください。指定した場合、マッピングは失敗します。

部分的にパラメータ化されたカスタムクエリを設定するときは、パラメータのデフォルト値を指定します。例えば、`Select * from $$DB1.$$Schema.$$Table1` と指定する場合は、DB1、Schema1、および Table1 の入出力パラメータを作成するときにデフォルト値としてプレースホルダ値を追加します。

マッピングでソースタイプとしてパラメータを選択した場合は、**［実行時にパラメータのオーバーライドを許可する］**オプションを選択しないでください。その後、マッピングタスクでカスタムクエリを選択できます。

詳細プロパティからの SQL オーバーライドとの組み合わせがサポートされるオーバーライドの詳細については、「SQL のオーバーライド」 (ページ 38)を参照してください。

付録 A

データ型リファレンス

Cloud データ統合は、Snowflake Data Cloud マッピングと、プッシュダウンの最適化が有効または有効でない詳細モードのマッピングで次のデータ型を使用します。

- Salesforce のネイティブデータ型は、フィールドのメタデータを編集するときに、ソーストランスフォーメーションとターゲットトランスフォーメーションに表示されます。
- トランスフォーメーションデータ型。トランスフォーメーションで表示されるデータ型のセットです。ANSI SQL-92 汎用データ型に基づく内部データ型で、プラットフォーム間でデータを移動するときにエージェントによって使用されます。これらは、マッピングまたは詳細モードのマッピングのすべてのトランスフォーメーションに表示されます。

エージェントは、ソースデータを読み取るときに、ネイティブデータ型を対応するトランスフォーメーションデータ型に変換してから、データのトランスフォームを実行します。エージェントは、ターゲットに書き込むときに、トランスフォーメーションデータ型を対応するネイティブデータ型に変換します。

Snowflake Data Cloud およびトランスフォーメーションデータ型

以下の表に、データ統合は、対応するトランスフォーメーションデータ型をサポートしています。

Snowflake Data Cloud データ型	トランスフォーメーションデータ型	範囲と内容
Binary（Varbinary）	binary	最大値: 8,388,60 デフォルト値は 8,388,60 です。
Boolean	string	ブール属性。
日付	datetime	日付と時刻の値。
Float（Double、Double precision、Real、Float、Float4、Float8）	double	倍精度（64 ビット）の浮動小数点数。 最大値: 1.7976931348623158e+307 最小値: -1.79769313486231E+307
Number（Decimal、Numeric）	decimal	マッピングに対して 28 ビットの精度とスケールを持つ数値。 詳細モードのマッピングに対して 38 ビットの精度とスケールを持つ数値。

Snowflake Data Cloud データ型	トランスフォーメーションデータ型	範囲と内容
NUMBER（Int、Integer、Bigint、Smallint、Tinyint、Byteint）	decimal	28 ビットの精度と位取りが 0 の数値。 最大値: 9.99999999999999E+27 最小値: -9.99999999999999E+26
Time	datetime	日付と時刻の値。 詳細モードのマッピングには適用されません。
Timestamp_LTZ	datetime	日付と時刻の値。
Timestamp_NTZ（Timestamp_NTZ、datetime）	datetime	日付と時刻の値。
Timestamp_TZ	datetime	日付と時刻の値。
Varchar（Text、Char、Character、String）	string	最大値: 16,777,216 デフォルト値は 16,777,216 です。

半構造化データ型とトランスフォーメーションデータ型

Snowflake は、Variant データ型を使用して、半構造化データを格納および表現します。Variant データ型には、オブジェクトや配列など、圧縮されていない最大サイズ 16 MB までの他の型の値を格納できます。

Snowflake ソースの配列、オブジェクト、およびバリアントは、Cloud データ統合の String データ型にマッピングされます。ターゲットへの書き込み中に、これらの文字列は、Snowflake ターゲットへの Array、Object、または Variant 列として書き込むことができます。Snowflake に書き込む文字列は、読み取り操作の後に表示されるものと同じように、シリアル化形式である必要があります。

［実行時に新規作成］オプションを使用して Variant データ型をソースから Snowflake ターゲットに書き込む場合、データ統合は Variant を Varchar としてターゲットに書き込みます。ターゲットに書き込む前に、フィールドマッピングを編集して、Varchar を Variant にマッピングできます。完全にパラメータ化されたマッピングでは、ターゲットメタデータをデフォルトの Varchar データ型から Variant に編集することはできません。

次の表に、Snowflake から読み取ることができる半構造化データ型と、Cloud データ統合でこれらがマッピングされる、対応するトランスフォーメーションデータ型を示します。

Snowflake データ型	トランスフォーメーションデータ型	説明と範囲
Array	String	16,777,216
Object	String	16,777,216
Variant	String	16,777,216

注: 半構造化データ型の読み取りまたは書き込みのデフォルトサイズは 65536 バイトに設定されています。サイズの上限を増やすには、次のパラメータを追加し、Snowflake Data Cloud 接続プロパティの［**追加の JDBC URL パラメータ**］フィールドに必要な値を設定します: *semiStructuredDTPrecision=<size>*

データ型のルールとガイドライン

データの読み取りまたは書き込みを行う場合、特定のデータ型には処理と設定において、いくつかの違いが適用されます。

マッピング

マッピングについてのルールとガイドラインは次のとおりです。

- 16 進数形式の Binary データ型のデータを読み書きできます。
- エージェントは、最大浮動小数点値 *1.7976931348623158e+308* を無限として読み書きします。
- 以下の形式を使用して、Datetime データ型のフィルタ値を指定できます。

  - YYYY-MM-DD HH24:MI:SS

  - YYYY/MM/DD HH24:MI:SS

  - MM/DD/YYYY HH24:MI:SS
- Snowflake Cloud Data ルックアップオブジェクトに最大またはデフォルトの精度で String データ型のフィールドが含まれており、行のサイズが行の最大サイズを超えていると、タスクは失敗します。
- データに日付フィールドが含まれていると、書き込み操作のパフォーマンスが低下します。
- Snowflake ターゲットに Record データ型の繰り返しカラムが含まれている場合、CDC ソースから変更されたデータをキャプチャするタスクは失敗します。
- マッピングで動的スキーマを処理する場合、次の更新は適用されません。

  - Timestamp データ型と Date データ型に対するスキーマの更新。

  - Varchar データ型の精度の低下を伴うスキーマ更新。

詳細モードのマッピング

詳細モードのマッピングについては、次のルールとガイドラインを考慮してください。

- Time データ型を読み取る場合は、Time 型のカラムの SQL オーバーライドで Time を Timestamp にマッピングする必要があります。例として、次のサンプルの SQL オーバーライドクエリを参照してください。
  ```
  SELECT "C1_DATE", to_timestamp (to_char("C2_TIME_3" , 'HH24:MI:SS.FF'), 'HH24:MI:SS.FF') AS "C2_TIME_3", to_timestamp (to_char("C3_TIME_5" , 'HH24:MI:SS.FF'), 'HH24:MI:SS.FF') AS "C3_TIME_5", to_timestamp (to_char("C4_TIME" , 'HH24:MI:SS.FF'), 'HH24:MI:SS.FF') AS "C4_TIME" FROM "SALES"."SF_DEV"."SRC_DATE_TIME"
  ```
- 動的スキーマを処理する場合、次の更新は適用されません。

  - Varchar データ型の精度の低下を伴うスキーマ更新。

  - Timestamp データ型と Date データ型に対するスキーマの更新。

  -［変更して変更を適用する］オプションを使用して Snowflake ターゲットに書き込むと、データ統合は、ソーススキーマから Timestampntz、Datetime、Timestamptz、Timestampltz、および Boolean データ型に加えられた変更を無視します。

付録 B

追加のランタイム設定

Secure Agent のプロパティでは、一部の Snowflake Data Cloud タスクについて追加の属性を設定できます。

レコードの一括処理の有効化、サイズの大きいレコード用のログの設定、ローカルステージングディレクトリの設定、データステージングの最適化のほか、読み取り操作のメモリ要件を改善できます。

JVM メモリ要件の設定

Snowflake からデータを読み取るマッピングおよびマッピングタスクで JDBC ドライバの内部エラーを回避するには、Secure Agent プロパティで Java ヒープスペースメモリを *-Xmx256m* として指定します。

この更新は、Snowflake ODBC 接続を使用したプッシュダウンの最適化で設定されたマッピングには適用されません。

次の手順を実行して、JVM メモリを構成します。

1. Administrator で、**[ランタイム環境]** タブにリストされている Secure Agent を選択します。
2. **[編集]** をクリックします。
3. **[システム構成の詳細]** セクションで、サービスとして **[データ統合サービス]** を選択し、タイプとして **[DTM]** を選択します。
4. **[JVMOption1]** プロパティを編集し、「*-Xmx256m*」と入力します。
5. **[保存]** をクリックします。

一括処理の設定

一括処理を有効にして、Snowflake に大量のデータを書き込むことができます。一括処理では最小限の API 呼び出しを使用して、書き込み操作のパフォーマンスを最適化します。

一括処理を有効にするには、Secure Agent のプロパティでプロパティ - *DENABLE_WRITER_BULK_PROCESSING=true* を指定します。

マッピングを実行する前に、以下の手順を実行して一括処理を設定します。

1. Administrator で、**[ランタイム環境]** タブにリストされている Secure Agent を選択します。
2. **[編集]** をクリックします。
3. **[システム構成の詳細]** セクションで、サービスとして **[データ統合サーバー]** を選択し、タイプとして **[DTM]** を選択します。

4. JVM オプションを編集して、*-DENABLE_WRITER_BULK_PROCESSING=true* と入力します。
5. **［保存］**をクリックします。

注: この更新は、Snowflake ODBC 接続または Snowflake Data Cloud 接続を使用したプッシュダウンの最適化で設定されたマッピングタスクには適用されません。

ロギングプロパティの設定

場合によっては、Snowflake からのセッションログにエラーメッセージが表示される場合があります。

このエラーを回避するには、次のタスクを実行します。

1. 次の場所に移動します。
   <Secure Agent のインストールディレクトリ>\apps\jdk\1.8.0_Zulu<バージョン>\jre\lib
2. java.util.logging.ConsoleHandler.level = WARNING を含めて logging.properties フィールドを設定します。

このログプロパティを設定しないと、セッションログに次のエラーが表示される場合があります。

net.snowflake.client.jdbc.internal.apache.http.impl.execchain.RetryExec execute

INFO: I/O exception (net.snowflake.client.jdbc.internal.apache.http.NoHttpResponseException) caught when processing request to {s}...> The target server failed to respond.

マッピングのための一時ディレクトリの設定

Secure Agent は、ローカルステージングファイルをデフォルトの一時ディレクトリに作成します。これとは別のディレクトリにローカルステージングファイルを格納するように設定することもできます。

ローカルステージングファイル用に別のディレクトリを設定するには、以下の手順を実行します。

1. Administrator で、**［ランタイム環境］**をクリックします。
   ［ランタイム環境］ページが表示されます。
2. カスタム設定プロパティを設定する Secure Agent を選択します。
3. **［アクション］**で編集する Secure Agent に対応する **［Secure Agent の編集］**アイコンをクリックします。
   ［Secure Agent の編集］ページが表示されます。
4. **［システム構成の詳細］**セクションの **［サービス］**で **［データ統合サーバー］**を選択します。
5. **［システム構成の詳細］**セクションの **［タイプ］**で **［DTM］**を選択します。
6. JVM オプションを-Djava.io.tmpdir=E:\Snowflake\temp に設定します。
7. **［保存］**をクリックします。
8. Secure Agent を再起動します。

マッピングのステージングパフォーマンスの最適化

データ統合は、デフォルトで、Snowflake からの読み取りまたは Snowflake への書き込みを行う前にデータをステージングするために、一時フォルダにフラットファイルをローカルに作成します。データ統合を設定して、読み取りおよび書き込み操作のステージングパフォーマンスを最適化できます。

ステージングプロパティを設定しない場合、データ統合は最適化された設定なしでステージングを実行しますが、これがタスクのパフォーマンスに影響する場合があります。

ステージングプロパティの設定

読み取り操作および書き込み操作のマッピングパフォーマンスを最適化できます。

最適化する操作に基づいて、エージェントのプロパティで次のステージングプロパティを設定する必要があります。

- 読み取り操作を最適化するには、次のプロパティを設定します: INFA_DTM_RDR_STAGING_ENABLED_CONNECTORS
- 書き込み操作を最適化するには、次のプロパティを設定します: INFA_DTM_STAGING_ENABLED_CONNECTORS

次のタスクを実行して、Secure Agent の Tomcat プロパティにステージングプロパティを設定します。

1. Administrator で、**[ランタイム環境]** をクリックします。
2. このプロパティを設定する Secure Agent を編集します。
3. **[システム構成の詳細]** セクションで、**[データ統合サーバー]** に **[サービス]** を選択し、タイプに **[Tomcat]** を選択します。
4. Tomcat プロパティの値を Snowflake Data Cloud Connector のプラグイン ID に設定します。
   プラグイン ID は、次のディレクトリのマニフェストファイルにあります: <Secure Agent のインストールディレクトリ>/downloads/<Snowflake パッケージ>/CCIManifest

   次の図に、読み取りおよび書き込み操作用の Secure Agent のプロパティセットを示します。

マッピングを実行すると、プロパティを設定した操作に基づいて、マシンの次のディレクトリにフラットファイルが作成されます。

- 読み取り操作: C:\Windows\Temp\StagingReader\<Source_Name><data_.csv>
- 書き込み操作: C:\Windows\Temp\snowflake\stage\<Snowflake_Target.txt>

セッションログを確認します。

- 読み取り操作: フラットファイルによるステージングが正常に完了すると、データ統合では次のメッセージがセッションログに記録されます: Staging mode is enabled to read data.
- 書き込み操作: フラットファイルによるステージングが正常に完了すると、データ統合では次のメッセージがセッションログに記録されます: The INFA_DTM_STAGING is successfully enabled to use the flat file to create local staging files.

ルールおよびガイドライン

読み取り操作に対してステージングプロパティを有効にすると、Snowflake からストアドプロシージャを呼び出すカスタムクエリを設定できなくなります。クエリを実行してストアドプロシージャを呼び出すと、タスクは失敗します。

書き込み操作に対してステージングプロパティを有効にする場合は、次のガイドラインを考慮してください。

- プッシュダウンの最適化が有効になっているマッピングを実行すると、プッシュダウンの最適化なしでマッピングが実行されます。

- データにタイムゾーン付きの Timestamp データ型が含まれている場合、ローカルフラットファイルでデータをステージングせずにジョブが実行されます。
- マッピングにソースとして Oracle CDC が含まれ、ターゲットとして Snowflake が含まれている場合、ローカルフラットファイルでデータをステージングせずにジョブが実行されます。
- ステージングプロパティを有効にし、ターゲットトランスフォーメーションプロパティの **[追加の書き込みランタイムパラメータ]** フィールドで copyEmptyFieldAsEmpty を設定するときに、空の値と NULL 値を含むデータをターゲットに書き込む動作を指定できます。
  次の表に、これらのプロパティを設定した場合の空値と NULL 値の動作を示します。

追加の書き込みランタイムパラメータ	ステージングの最適化を有効にする	ステージングの最適化を無効にする
copyEmptyFieldAsEmpty=FALSE	空 -> NULL NULL -> NULL	空 -> NULL NULL -> NULL
copyEmptyFieldAsEmpty=TRUE	空 -> 空 NULL -> 空	空 -> 空 NULL -> 空
プロパティが設定されていません	空 -> 空 NULL -> NULL	空 -> 空 NULL -> NULL

ステージングプロパティを設定する前後の実行時間のパフォーマンスの比較については、次の How-To ライブラリの記事を参照してください:
Performance Tuning and Best Practices for Snowflake Data Cloud Connector

付録 C

Snowflake Data Cloud Connector へのアップグレード

Snowflake へのアクセスおよび Snowflake での操作の実行に、Snowflake V1 接続または Snowflake ODBC 接続を使用している場合は、Snowflake Data Cloud Connector へのアップグレードが可能です。

オブジェクトタイプには互換性があるため、Snowflake 接続または Snowflake Data Cloud 接続で Snowflake ODBC 接続を使用する既存のマッピングでソースまたはターゲット接続タイプを切り替えることができます。

マッピングで接続を置き換えると、以前に選択したオブジェクトは保持されません。Snowflake オブジェクトを再インポートする必要があります。2 つのコネクタ間で共通のフィールドに設定されたソース、ターゲット、およびルックアップの詳細プロパティは、新しいコネクタに保持されます。また、古いコネクタからの設定済みのフィールドマッピングを保持するためのオプションもあります。

古いコネクタの設定済みの値を使用して、マッピングを正常に実行できます。さらに、拡張された Snowflake Data Cloud Connector が提供する機能を設定することができます。

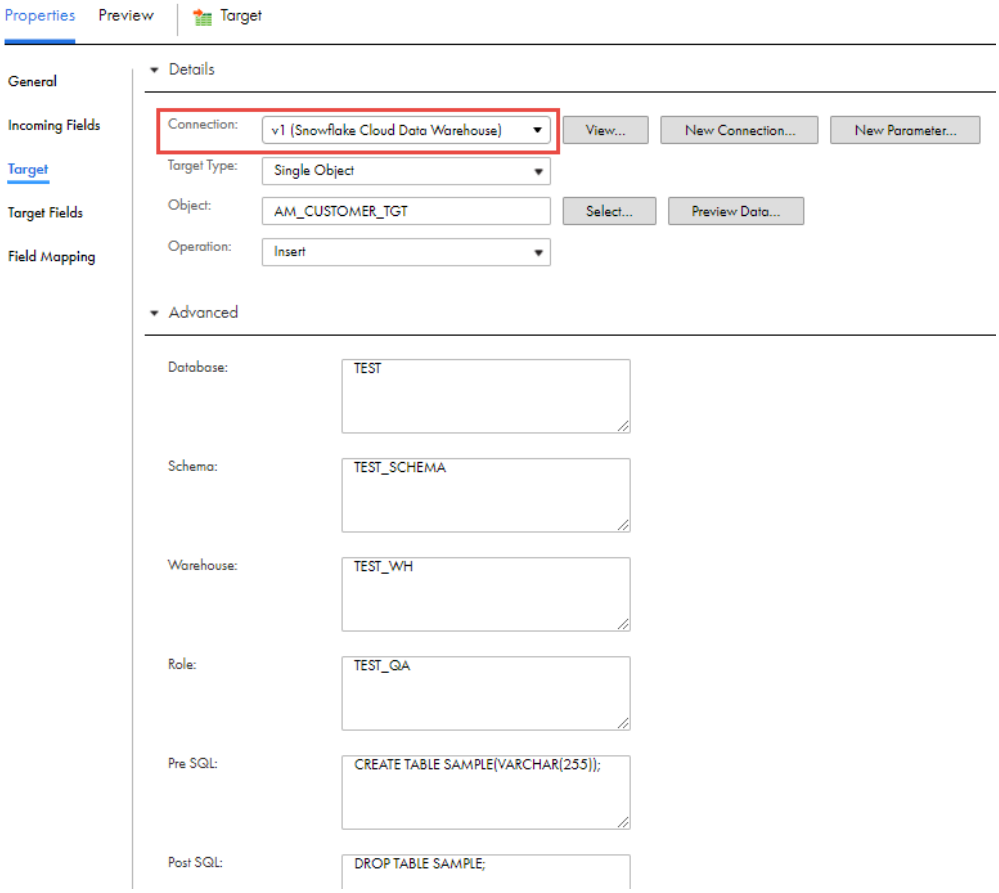
重要: また、データ統合 REST API を使用して、Snowflake V1 接続を既存のアセットの Snowflake Data Cloud 接続に移行することもできます。詳細については、「*REST API リファレンス*」の「connectionMigration」のトピックを参照してください。移行プロセスとルールおよびガイドラインの詳細については、次の How-To ライブラリの記事を参照してください:
Migrating a connector from previous versions using the Data Integration REST API

接続切り替えの例

Snowflake V1 接続を使用する既存の Snowflake マッピングを Snowflake Data Cloud 接続に切り替える必要があるとします。

1. Snowflake Data Cloud に切り替える既存の Snowflake V1 マッピングを開きます。

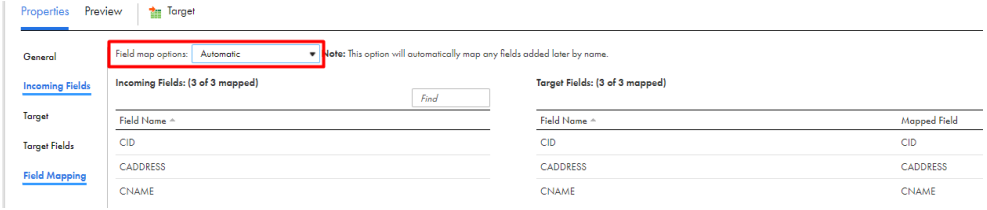
   次の図に、Snowflake V1 接続を使用する、ターゲットトランスフォーメーションで設定された詳細プロパティを含んだ既存のマッピングを示します。



   この例で設定されているターゲットオブジェクトパスは次のとおりです: CQA/CQA_SCHEMA/AM_CUSTOMER_SRC

2. 接続を切り替えたときにフィールドマッピングからのマッピングされたフィールドを保持するには、**［フィールドマッピング］**タブで、Snowflake V1 マッピングの次の **［フィールドマップオプション］**メニューから選択を行います。

   - 切り替え後に、自動的にマッピングされたフィールドを保持するには、**［自動］**を選択します。

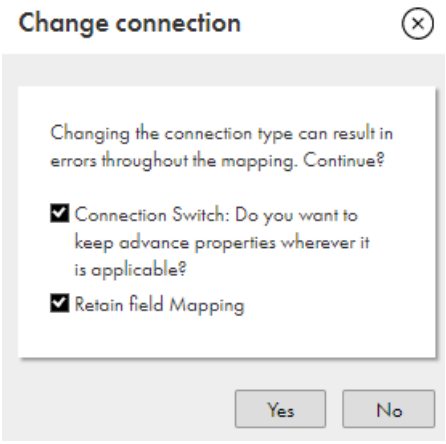


   - 切り替え後に、保持されたフィールドを手動でマッピングするには、**［手動］**を選択します。

     注: 手動を選択した場合は、接続を切り替えた後に、保持されたフィールドを以前のマッピングを使用して自動的にマッピングするオプションがあります。

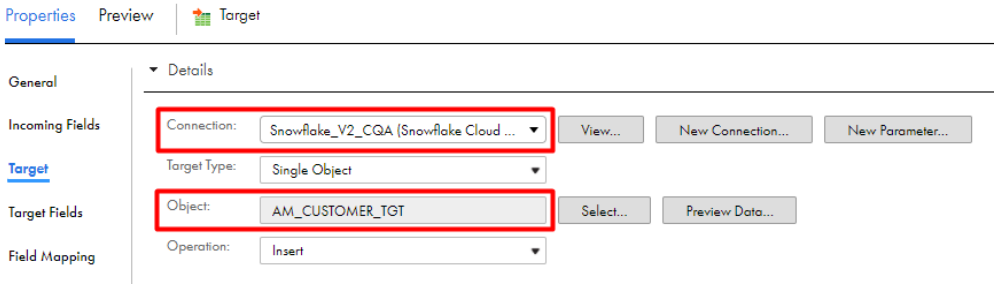
3. 接続を切り替えるには、**［接続］**フィールドで接続を Snowflake V1 から Snowflake Data Cloud に変更します。
4. **［接続の変更］**ダイアログボックスで次のプロパティを選択し、**［はい］**をクリックします。
   - **接続の切り替え**。選択した接続に切り替えます。
   - **フィールドマッピングの保持**。Snowflake V1 からの設定済みのフィールドマッピングを保持します。

   次の図に、選択する必要のあるオプションを示します。

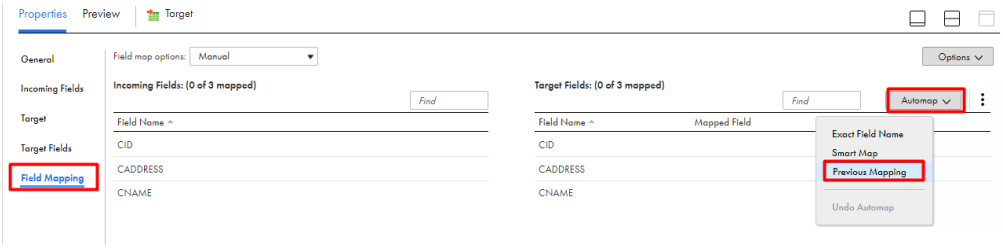


5. マッピングでは、Snowflake V1 と同じオブジェクトパスを使用します。

   次の図に、CQA/CQA_SCHEMA/AM_CUSTOMER_SRC という同じオブジェクトパスに切り替えた接続を示します。

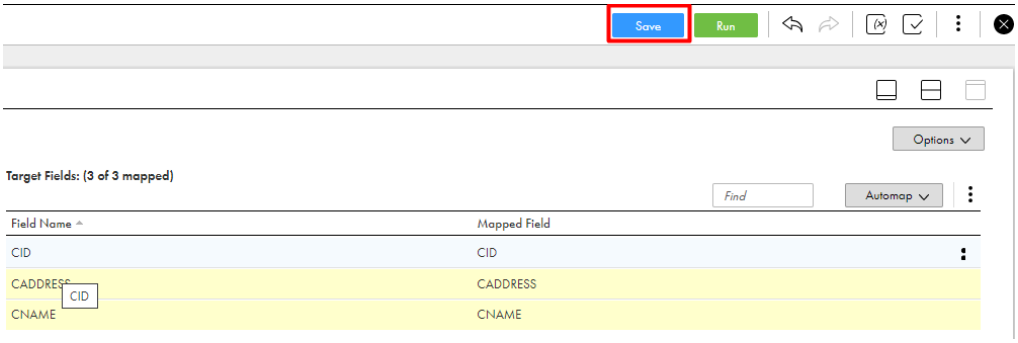


   Snowflake V1 マッピングからの設定済みのターゲットの詳細プロパティが、ターゲットトランスフォーメーションに反映されます。
6. Snowflake V1 マッピングの **［フィールドマッピング］**タブで **［手動］**を選択し、Snowflake Data Cloud のフィールドマッピングを反映するには、**［フィールドマッピング］**タブで **［自動マップ］**を選択し、**［以前のマッピング］**を選択します。



注: Snowflake V1 で **［自動］**を選択した場合、このタスクを実行する必要はありません。

次の図では、Snowflake V1 マッピングからの設定済みのマッピングされたフィールドが、Snowflake Data Cloud マッピングに反映されています。



7. **[保存]** をクリックします。

切り替え後に保持されるプロパティ

Snowflake V1 接続または Snowflake ODBC 接続を使用する既存のマッピングまたはマッピングタスクのソースまたはターゲット接続タイプを Snowflake Data Cloud 接続に置き換える場合に、設定済みの詳細プロパティとフィールドマッピングを保持するためのオプションがあります。

次の表に、Snowflake Data Cloud マッピングで保持される Snowflake ODBC 接続からの設定済みのソース、ルックアップ、およびターゲットの詳細プロパティを示します。

プロパティ	Snowflake Data Cloud **に保持される** Snowflake ODBC **プロパティ**
ソース	- Pre SQL - Post SQL
ターゲット	- Pre SQL - Post SQL - ターゲットのトランケート - 拒否された行の転送 - 更新オーバーライド

次の表に、Snowflake Data Cloud マッピングに保持される Snowflake V1 の設定済みのソース、ルックアップ、およびターゲットの詳細プロパティを示します。

プロパティ	Snowflake Data Cloud に保持される Snowflake V1 のプロパティ
ソース	- データベース - スキーマ - ウェアハウス - ロール - Pre SQL - Post SQL - テーブル名 - SQL オーバーライド - トレースレベル
Lookup	- データベース - スキーマ - ウェアハウス - ロール
ターゲット	- データベース - スキーマ - ウェアハウス - ロール - Pre SQL - Post SQL - バッチ行のサイズ - ローカルステージングファイルの数 - ターゲットテーブルの切り詰め - 追加のランタイムプロパティ - テーブル名 - 拒否ファイルパス - 成功ファイルディレクトリ - エラーファイルディレクトリ - 拒否された行の転送

ルールおよびガイドライン

既存のマッピングの Snowflake ODBC 接続または Snowflake V1 接続を Snowflake Data Cloud 接続に置き換える場合は、次のルールを考慮してください。

- フィルタとソートの値は保持されません。新しい接続に切り替えた後に、ソート値とフィルタ値を手動で追加する必要があります。
- 既存のマッピングに複数のオブジェクトが含まれている場合は、新しい接続への切り替え後、これらのオブジェクト間の関係は保持されません。

索引

C

Cloud Application Integration コミュニティ
  URL 7
Cloud 開発者コミュニティ
  URL 7

I

Informatica Intelligent Cloud Services
  Web サイト 7
Informatica グローバルカスタマサポート
  連絡先情報 8

S

Snowflake Cloud Data Warehouse V2 接続
  プッシュダウンの最適化の概要 61
  設定 68
Snowflake Data Cloud
  アセット 9
  コネクタ 9
  ソース 9
  ターゲット 9
  トランスフォーメーション 9
  プッシュダウン 69
  ルックアップ 9
Snowflake Data Cloud Connector
  ルールおよびガイドライン 92
Snowflake Data Cloud 接続
  設定 65, 67
Snowflake ODBC 接続
  プッシュダウンの最適化の概要 61
  設定 67, 68
Snowflake データクラウド
  マッピングの例 28
SQL トランスフォーメーション
  設定 25

W

Web サイト 7

あ

アップグレード通知 8

き

キャッシュ
  ルックアップキャッシングの有効化 59

こ

コネクタの概要 9

し

システムステータス 8

す

ステータス
  Informatica Intelligent Cloud Services 8

そ

ソート 34

て

データのステージング
  フラットファイル 95
データ型
  トランスフォーメーションデータ型 90, 91
  ネイティブデータ型 90, 91
  概要 90

と

トラブルシューティング
  Snowflake Data Cloud Connector 85
トランスフォーメーション
  プッシュダウンの最適化 72

は

パーティション化
  キー範囲パーティション化 37
  パススルーのパーティション化 45
パラメータ化されたソート 34
パラメータ化されていないソート 34

ふ

フィルタ 34
プッシュダウンの最適化
  トランスフォーメーション 70–72
  プレビュー 65
  関数 70, 71
プッシュダウンの最適化プレビュー 65

フラットファイル
データのステージング 95

ま

マッピング
Post-SQL 34, 40
Pre-SQL 34, 40
ウェアハウス 34, 40
スキーマ 34, 40
ソースプロパティ 34
ターゲットプロパティ 40
データベース 34, 40
ルックアップの概要 57
ルックアッププロパティ 57
ロール 34, 40
例 28

め

メンテナンスの停止 8

る

ルックアップ
ウェアハウス 57
キャッシュを使用しない 59
スキーマ 57
データベース 57
ロール 57
複数の一致 57
ルックアップキャッシュ
動的 59
ルックアップトランスフォーメーション
ルックアップキャッシュ 59

ろ

ローカルステージングファイル
JVM オプション 94
ディレクトリ設定 94