



Informatica®  
10.4.0

# Developer マッピングガイド

Informatica Developer マッピングガイド  
10.4.0  
2019 年 12 月

© 著作権 Informatica LLC 2014, 2020

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

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

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

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

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

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

発行日: 2020-03-02

# 目次

<b>序文</b>	<b>12</b>
Informatica のリソース	12
Informatica Network	12
Informatica ナレッジベース	12
Informatica マニュアル	12
Informatica 製品可用性マトリックス	13
Informatica Velocity	13
Informatica Marketplace	13
Informatica グローバルカスタマサポート	13
<b>第 1 章: マッピング</b>	<b>14</b>
マッピングの概要	14
マッピングコンポーネント	15
データオブジェクト操作	16
トランスフォーメーション	20
マップレット	20
セグメント	21
セグメントのコピー	21
ビュー	21
マッピングの検証	22
接続の検証	23
式の検証	23
オブジェクトの検証	23
パラメータの検証	23
同期の検証	23
マッピングランタイムプロパティ	24
検証環境	24
実行環境	25
拒否ファイルディレクトリ	25
最大並行処理	26
ターゲットコミット間隔	26
エラー時の停止	27
偽装ユーザー名のマッピング	27
提案された並行処理	27
ターゲットロード順の制約	28
挿入行と削除行に関する制約	28
ターゲットのロード順に関するルールとガイドライン	29
ターゲットのロード順の例	29
マッピング設定	31
マッピング設定プロパティ	32

デフォルトマッピング設定の構成方法. . . . .	34
再利用可能なマッピング設定の作成方法. . . . .	34
カスタムマッピング設定の作成および使用方法. . . . .	34
詳細マッピングオプション. . . . .	35
詳細オプションを使用したマッピングの実行方法. . . . .	36
マッピングの開発方法. . . . .	38
マッピングの作成. . . . .	38
マッピングへのオブジェクトの追加. . . . .	38
マッピングオブジェクトの接続. . . . .	39
ターゲットロード順の制約の作成. . . . .	40
マッピングの検証. . . . .	41
マッピングの実行. . . . .	41
<b>第2章: マプレット. . . . .</b>	<b>42</b>
マプレットの概要. . . . .	42
マプレットのタイプ. . . . .	43
マプレットの入力と出力. . . . .	43
マプレットの入力. . . . .	43
マプレットの出力. . . . .	44
生成されたマプレット. . . . .	44
生成されたマプレットのルールおよびガイドライン. . . . .	44
マプレットの生成. . . . .	45
ルール仕様およびマプレット. . . . .	46
ルール仕様のプロパティ. . . . .	47
マプレットの作成. . . . .	48
マプレット検証. . . . .	48
マプレットの検証. . . . .	48
ルール検証としてのマプレット. . . . .	48
<b>第3章: マッピングパラメータ. . . . .</b>	<b>49</b>
マッピングパラメータの概要. . . . .	49
システムパラメータ. . . . .	50
ユーザー定義のパラメータ. . . . .	51
日付/時刻パラメータ. . . . .	52
ユーザー定義のパラメータを作成する場所. . . . .	53
マッピングのパラメータ. . . . .	53
パラメータインスタンス値. . . . .	54
マプレットのパラメータ. . . . .	55
マプレットのパラメータインスタンス値. . . . .	55
マッピングのマプレットパラメータ. . . . .	56
マプレットのパラメータの例. . . . .	56
論理データオブジェクトのパラメータ. . . . .	57
仮想テーブルマッピングのパラメータ. . . . .	58

パラメータセット	59
パラメータファイル	60
パラメータファイル構造	60
project 要素	61
application 要素	62
パラメータファイルに関するルールとガイドライン	62
パラメータファイルの例	63
パラメータファイルのエクスポート	64
infacmd ms ListMappingParams からのパラメータファイルの作成	64
パラメータ階層	65
パラメータ階層を使用したパラメータのオーバーライド	65
実行時にパラメータをバインドしてパラメータをオーバーライドする	66
マッピング内のパラメータのオーバーライドのユースケース	66
パラメータの設定方法	68
トランスフォーメーションプロパティのパラメータの作成	69
式内のパラメータの作成	71
マッピングパラメータとしてトランスフォーメーションパラメータを公開する	74
パラメータインスタンス値の設定	75
パラメータセットの作成	76
パラメータを使用してマッピングを実行する方法	78
Developer tool からパラメータを使用してマッピングを実行する方法	78
コマンドラインからパラメータを使用してマッピングを実行する方法	81
<b>第 4 章: パラメータを割り当てる場所</b>	<b>82</b>
パラメータを割り当てる場所概要	82
圧縮形式のパラメータ	86
Hive ソース用のカスタムクエリのパラメータ	87
リレーショナルソース用のカスタムクエリのパラメータ	89
式のパラメータ	89
式パラメータ	91
式パラメータ例	92
例. フィルタ条件内の式パラメータ	92
例. 式トランスフォーメーション内の式パラメータ	93
例. ポート式内のネストしたパラメータ	94
フィールドおよびプロパティ値のパラメータ	95
リレーショナルテーブルリソースのパラメータ	96
SQL 文のパラメータ	97
フィルタおよび結合条件内のパラメータ	98
SQL 文の文字列パラメータ	98
SQL 文でのパラメータの使用のヒント	99
ポートリストのパラメータ	99

<b>第 5 章 : マッピング出力</b>	<b>101</b>
マッピング出力の概要	101
ユーザー定義マッピング出力	102
[出力] ビュー	102
マッピング出力式	103
システム定義のマッピング出力	104
デプロイ済みマッピングでのマッピング出力の保持とバインド	105
マッピング出力の設定およびバインド方法	105
保持されている出力のメンテナンス	109
保持されているマッピング出力およびデプロイメントのルールとガイドライン	110
マップレットのマッピング出力	110
マップレット出力のマッピング出力へのバインド	112
論理データオブジェクトのマッピング出力	114
マップレット出力をマッピング出力にバインドする方法	114
マップレット出力の定義	115
マップレットでのマッピング出力式の設定	116
マップレットからの出力のマッピング出力へのバインド	116
 <b>第 6 章 : SQL クエリからのマッピングの生成</b>	 <b>118</b>
SQL クエリからのマッピングの生成の概要	118
SQL クエリから生成されたマッピングの例	118
マッピングを生成するための SQL 構文	119
相関サブクエリ	119
マッピングを生成するクエリでの関数のサポート	120
サポートされない関数を使用した SQL クエリからのマッピングの生成	120
INSERT、UPDATE、および DELETE の構文	121
INSERT 文、UPDATE 文、および DELETE 文のルールおよびガイドライン	121
SQL クエリからのマッピングまたは論理データオブジェクトの生成	122
SQL 文からのマッピングの生成	122
SQL 文の作成	123
SQL 文の Developer tool への貼り付けまたはインポート	123
マッピングの開発の完了	124
 <b>第 7 章 : 動的マッピング</b>	 <b>125</b>
動的マッピングの概要	125
動的マッピングの設定	126
動的データソース	126
動的マッピングのポートおよびリンク	127
動的マッピングのルール	128
動的マッピングのパラメータ	128
動的ソース	129
データソースからカラムを取得する	130

フラットファイル名にパラメータを割り当てる.....	131
リレーショナルソースのプロパティにパラメータを割り当てる.....	132
ソースデータオブジェクトにパラメータを割り当てる.....	132
動的ターゲット.....	133
データソースからカラムを取得する.....	135
マッピングフローに基づくターゲットの定義.....	135
データオブジェクトに基づくターゲットの定義.....	136
ターゲットスキーマ戦略の定義.....	136
リレーショナルターゲットのプロパティにパラメータを割り当てる.....	138
ターゲットデータオブジェクトにパラメータを割り当てる.....	138
動的ターゲットのルールおよびガイドライン.....	139
動的ポートおよび生成されたポート.....	139
動的ポートおよび生成されたポートの設定.....	140
動的ポートおよび生成されたポートのルールおよびガイドライン.....	141
動的式.....	141
動的式のパラメータ化.....	142
入力ルール.....	143
入力ルールの設定.....	143
ポートを含めるまたは除外する.....	144
残りのすべてのポートを含める.....	145
生成されたポートの名前の変更.....	146
ソースポート名のリストア.....	148
生成されたポートの順序変更.....	149
選択ルールとポートセレクタ.....	152
ポートセレクタの設定.....	152
選択ルール.....	153
選択ルールとポートセレクタの例.....	154
設計時リンク.....	154
リンクの解決.....	156
ランタイムリンク.....	156
ランタイムリンクの設定.....	157
ランタイムリンクの例.....	159
動的マッピングのトラブルシューティング.....	159
<b>第 8 章 : 動的マッピングを開発して実行する方法.....</b>	<b>162</b>
動的マッピングの開発と実行.....	162
動的ソースの設定.....	164
動的マッピングのソースとしてのパラメータの使用.....	164
実行時にメタデータの変更を取得するためのソースの設定.....	164
動的ポートの作成.....	165
入力ルールを使用した動的ポートの設定.....	166
手順 1. [入力ルール] ダイアログボックスを開く.....	167
手順 2. 入力ルールの定義.....	167

手順 2a. 演算子と選択条件の選択. . . . .	168
手順 2b. 名前選択条件の詳細の設定. . . . .	168
手順 2c. タイプ選択条件の詳細の設定. . . . .	169
手順 2d. パターン選択条件の詳細の設定. . . . .	169
手順 3. 生成されたポートの名前の変更. . . . .	170
手順 4. 生成されたポートの順序変更. . . . .	170
手順 5. 動的ポート設定の検証. . . . .	170
ポートセレクタの作成. . . . .	170
動的式の作成. . . . .	171
動的ターゲットの設定. . . . .	173
動的マッピングのターゲットとしてのパラメータの使用. . . . .	174
実行時のデータソースからのターゲットオブジェクトカラムの取得. . . . .	175
実行時にターゲットを作成または置換するための DDL クエリの定義. . . . .	175
書き込みトランスフォーメーションポートの定義. . . . .	177
ランタイムリンクの作成と設定. . . . .	178
動的マッピングの検証. . . . .	181
動的ソースおよびターゲットの検証. . . . .	181
動的マッピングの実行. . . . .	182
<b>第 9 章 : 動的マッピングの使用例. . . . .</b>	<b>183</b>
使用例: リレーショナルソースのメタデータ変更のための動的マッピング. . . . .	183
ソーステーブル. . . . .	183
ターゲットテーブル. . . . .	184
動的マッピング. . . . .	184
手順 1. 読み取りトランスフォーメーションの設定. . . . .	185
手順 2. ジョイナトランスフォーメーションの設定. . . . .	186
手順 3. アグリゲータトランスフォーメーションの設定. . . . .	187
手順 4. 書き込みトランスフォーメーションの設定. . . . .	190
手順 5. ランタイムリンクの作成と設定. . . . .	191
手順 6. マッピングの検証と実行. . . . .	191
手順 7. ソーススキーマ変更後のマッピングの実行. . . . .	192
使用例: さまざまなソースおよびターゲットの動的マッピングの再利用. . . . .	194
ソースファイル. . . . .	194
ターゲットファイル. . . . .	196
動的マッピング. . . . .	196
手順 1. Read_Customer_FF 読み取りトランスフォーメーションの設定. . . . .	197
手順 2. Exp_TRIM 式トランスフォーメーションの設定. . . . .	197
手順 3. Exp_Output 式トランスフォーメーションの設定. . . . .	201
手順 4. Write_customerTrim_FF 書き込みトランスフォーメーションの設定. . . . .	203
手順 5. マッピングの検証と保存. . . . .	205
手順 6. 異なるソースおよびターゲットに対する動的マッピングの実行. . . . .	205



<b>第 10 章 : マッピング管理</b>	<b>208</b>
マッピング管理の概要	208
マッピングジョブのプロパティの表示	209
マッピングジョブのサマリ統計の表示	209
マッピングジョブの詳細統計の表示	209
マッピングジョブのログの表示	210
デプロイ済みのマッピングジョブの再発行	210
マッピングジョブのキャンセル	211
拒否ファイル	211
拒否ファイルの場所	211
拒否ファイルの内容	212
<b>第 11 章 : PowerCenter へのエクスポート</b>	<b>214</b>
PowerCenter へのエクスポートの概要	214
PowerCenter のリリースの互換性	215
互換性レベルの設定	215
マプレットのエクスポート	215
パラメータを含むマッピングのエクスポート	216
PowerCenter へのエクスポートのオプション	217
PowerCenter へのオブジェクトのエクスポート	218
エクスポートの制限	219
PowerCenter へのエクスポートに関するルールおよびガイドライン	221
PowerCenter へのエクスポートのトラブルシューティング	222
<b>第 12 章 : PowerCenter からのインポート</b>	<b>223</b>
PowerCenter からのインポートの概要	223
オーバーライドプロパティ	224
競合の解決	226
インポートサマリ	227
データ型の変換	227
トランスフォーメーションの変換	228
トランスフォーメーションのプロパティの制限	229
パラメータの変換	235
パラメータを使用したマッピングのインポートのルールとガイドライン	235
変数を使用したマッピングのインポートのルールとガイドライン	236
オーバーライドのインポート時のルールとガイドライン	236
システムパラメータ変換	236
PowerCenter リポジトリの接続プロパティ	237
接続の割り当て	238
PowerCenter からのオブジェクトのインポート	238
インポートの制限	240
ソースおよびターゲットの制限	240

トランスフォーメーション制限. . . . .	241
マッピングの制限. . . . .	241
関数の制限. . . . .	242
マッピング変数の制限. . . . .	242
セッションのプロパティ制約. . . . .	242
ワークフローの制限. . . . .	243
タスクの制限. . . . .	243
インポートパフォーマンス. . . . .	243

## 第 13 章 : パフォーマンスのチューニング..... 244

Performance Tuning Overview. . . . .	244
最適化方式. . . . .	245
初期プロジェクション最適化方法. . . . .	245
Early Selection Optimization Method. . . . .	245
ブランチ刈り込み最適化方式. . . . .	246
述部最適化方式. . . . .	246
コストベースの最適化方式. . . . .	247
データシップ結合最適化方式. . . . .	247
準結合最適化方式. . . . .	248
最適化されたマッピングの表示. . . . .	249
Optimizer Levels. . . . .	249
Setting the Optimizer Level for a Developer Tool Mapping. . . . .	250
デプロイ済みのマッピングに対する最適化レベルの設定. . . . .	251

## 第 14 章 : プッシュダウンの最適化..... 252

プッシュダウンの最適化の概要. . . . .	252
プッシュダウンタイプ. . . . .	253
完全なプッシュダウンの最適化. . . . .	253
ソースプッシュダウン. . . . .	254
プッシュダウンの設定. . . . .	254
トランスフォーメーションプッシュロジック. . . . .	255
ソースへのプッシュダウンの最適化. . . . .	256
リレーショナルソースへのプッシュダウンの最適化. . . . .	256
ネイティブソースへのプッシュダウンの最適化. . . . .	258
PowerExchange 非リレーショナルソースへのプッシュダウンの最適化. . . . .	258
ODBC ソースへのプッシュダウンの最適化. . . . .	258
SAP ソースへのプッシュダウンの最適化. . . . .	259
プッシュダウンの最適化の式. . . . .	260
関数. . . . .	260
演算子. . . . .	276
データ統合サービスとソースの出力の比較. . . . .	277

<b>第 15 章 : パーティション化されたマッピング</b>	<b>278</b>
パーティション化されたマッピングの概要	278
各パイプラインステージごとに 1 つのスレッド	279
各パイプラインステージごとに複数のスレッド	280
パーティション化されたフラットファイルソース	282
同時読み込みのパーティション化	282
パーティション化されたリレーショナルソース	283
パーティション化のリレーショナル接続タイプ	284
パーティション化されたリレーショナルソースの SQL クエリ	284
リレーショナルソースパーティションに関するルールおよびガイドライン	285
パーティション化されたフラットファイルターゲット	285
パーティション化されたファイルターゲットに対応するための出力ファイルディレクトリの最適化	286
パーティション化されたファイルターゲットのマージオプション	286
パーティション化されたファイルターゲットのコマンド	288
パーティション化されたリレーショナルターゲット	289
パーティション化のリレーショナル接続タイプ	290
リレーショナルターゲットパーティションに関するルールおよびガイドライン	290
パーティション化されたトランスフォーメーション	291
パーティション化されたトランスフォーメーションに関する制限	291
トランスフォーメーションのキャッシュのパーティション化	292
トランスフォーメーションのためのパーティション化の無効化	293
パーティション化されたマッピングにおける順序の維持	294
安定ソートの維持	295
マッピングの最大並行処理のオーバーライド	295
トランスフォーメーションのための提案された並行処理	296
アドレスバリデータおよび一致トランスフォーメーションの実行インスタンス数	297
並行処理の最大値のオーバーライド	298
パーティション化されたマッピングのトラブルシューティング	299
 <b>第 16 章 : Developer tool の命名規則</b>	 <b>300</b>
トランスフォーメーションの命名規則	300
オブジェクトタイプの命名規則	302
ワークフローオブジェクトの命名規則	302
 <b>索引</b>	 <b>304</b>

# 序文

『*Informatica Developer マッピングガイド*』を使用して、マッピングの開発、実行、管理方法を学びます。マッピングの概念、マッピングパラメータと動的マッピングによって柔軟性を持たせる方法、およびチューニングとパーティション化によってマッピングを最適化する方法について理解します。

## 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 Network

Informatica Network は、Informatica ナレッジベースや Informatica グローバルカスタマサポートなど、多くのリソースへの入口です。Informatica Network を利用するには、<https://network.informatica.com> にアクセスしてください。

Informatica Network メンバーは、次のオプションを利用できます。

- ナレッジベースで製品リソースを検索できます。
- 製品の提供情報を表示できます。
- サポートケースを作成して確認できます。
- 最寄りの Informatica ユーザーグループネットワークを検索して、他のユーザーと共同作業を行えます。

### 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](mailto:KB_Feedback@informatica.com)) です。

### 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](mailto:infa_documentation@informatica.com)) までご連絡ください。

## Informatica 製品可用性マトリックス

製品可用性マトリックス (PAM) には、製品リリースでサポートされるオペレーティングシステム、データベースなどのデータソースおよびターゲットが示されています。Informatica PAM は、<https://network.informatica.com/community/informatica-network/product-availability-matrices> で参照できます。

## Informatica Velocity

Informatica Velocity は、Informatica プロフェッショナルサービスが開発したヒントとベストプラクティスのコレクションで、多数のデータ管理プロジェクトから得た実体験に基づいています。Informatica Velocity には、世界中の組織と連携してデータ管理ソリューションを計画、開発、デプロイ、管理する Informatica コンサルタントによる集合知を表しています。

Informatica Velocity リソースには、<http://velocity.informatica.com> からアクセスしてください。Informatica Velocity についての質問、コメント、またはアイデアがある場合は、[ips@informatica.com](mailto:ips@informatica.com) から Informatica プロフェッショナルサービスにお問い合わせください。

## Informatica Marketplace

Informatica Marketplace は、お使いの Informatica 製品を拡張したり強化したりするソリューションを検索できるフォーラムです。Marketplace で、Informatica デベロッパーやパートナーからの多数のソリューションを活用すれば、生産性を向上したり、プロジェクトでの実装時間を短縮したりできます。Informatica Marketplace は、<https://marketplace.informatica.com> からアクセスしてください。

## Informatica グローバルカスタマサポート

電話または Informatica Network からグローバルサポートセンターに連絡できます。

各地域の Informatica グローバルカスタマサポートの電話番号は、Informatica Web サイト (<https://www.informatica.com/services-and-training/customer-success-services/contact-us.html>) を参照してください。

Informatica Network でオンラインサポートリソースを見つけるには、<https://network.informatica.com> にアクセスし、eSupport オプションを選択します。

# 第 1 章

## マッピング

この章では、以下の項目について説明します。

- [マッピングの概要, 14 ページ](#)
- [マッピングコンポーネント, 15 ページ](#)
- [ビュー, 21 ページ](#)
- [マッピングの検証, 22 ページ](#)
- [マッピングランタイムプロパティ, 24 ページ](#)
- [ターゲットロード順の制約, 28 ページ](#)
- [マッピング設定, 31 ページ](#)
- [詳細マッピングオプション, 35 ページ](#)
- [マッピングの開発方法, 38 ページ](#)

## マッピングの概要

マッピングは、ソースとターゲットの間のデータフローを表す入力オブジェクトと出力オブジェクトのセットです。データトランスフォーメーションのルールを定義するトランスフォーメーションオブジェクトでリンクされます。データ統合サービスは、マッピングに設定された命令を使用して、データの読み込み、変換、および書き込みを行います。

複数のマッピングを順次実行できるように、ワークフローからマッピングを実行できます。または、マッピング実行の前後にコマンドを実行して手順を実行するワークフローを開発できます。物理データオブジェクトが設定されたマッピングを、ワークフローのマッピングタスクにおよび出力として含めることができます。

マッピングに含めるオブジェクトと出力オブジェクトのタイプによって、マッピングのタイプが決まります。Developer tool では、以下のタイプのマッピングを作成できます。

### 論理データオブジェクトマッピング

論理データオブジェクトを 1 つ以上の物理データオブジェクトにリンクします。論理データオブジェクトマッピングを使用すると、複数のソースおよび形式のデータを標準化されたビューに統合できます。

### 操作マッピング

操作が、マッピング、マッピング出力、またはその両方として含まれています。操作マッピングでは、Web サービスクライアントに対して Web サービス操作が実行されます。

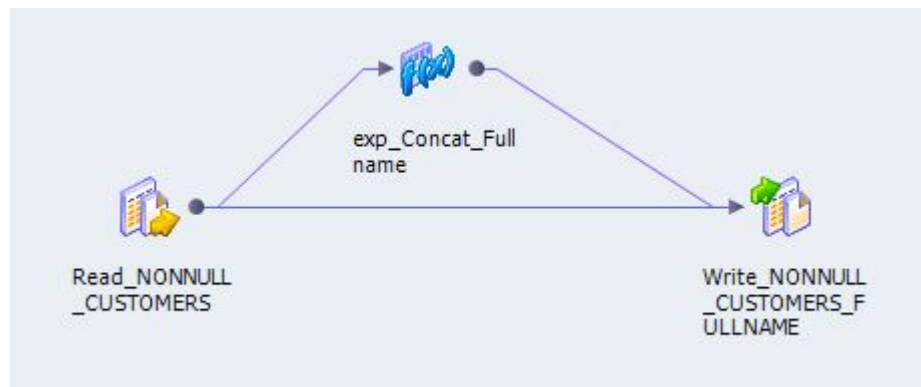
## 仮想テーブルマッピング

仮想テーブルが、マッピング出力として含まれています。仮想テーブルマッピングは、SQL データサービスのソースと仮想テーブル間の仮想データフローを定義します。データを変換するには仮想テーブルマッピングを使用します。

## 動的マッピング

定義したパラメータとルールに基づいて、ソース、ターゲット、トランスフォーメーションロジックを実行時に変更できるマッピング。動的マッピングを使用して、スキーマまたはメタデータへの頻繁な変更を管理します。

次の図は、マッピングの例を示しています。



# マッピングコンポーネント

マッピングコンポーネントは、ソースとターゲットの間のデータフローを決定します。

すべてのマッピングに、マッピングコンポーネントまたはファイルからデータを読み取る入力オブジェクトが含まれている必要があります。さらにすべてのマッピングに、マッピングコンポーネントまたはファイルにデータを書き込む出力オブジェクトも含まれている必要があります。

各マッピングには以下のコンポーネントを含めることもできます。

## データオブジェクト操作

ソースまたはターゲットに対して特定のランタイム操作を実行するために必要なプロパティを含むリポジトリオブジェクト。一部の PowerExchange アダプタデータソースに必須です。

## トランスフォーメーション

データをターゲットに書き込む前に変更します。さまざまなトランスフォーメーションオブジェクトを使用して、種々の関数を実行します。

## マップレット

複数のマッピングで使用できるトランスフォーメーションのセットを含む再利用可能なオブジェクト。

## セグメント

マッピング、マップレット、ルール、または仮想ストアドプロシージャ内の 1 つ以上のオブジェクトで構成されます。

## データオブジェクト操作

データオブジェクト操作は、ソースまたはターゲットに対して特定のランタイム操作を実行するために必要なプロパティを格納するリポジトリオブジェクトです。PowerExchange アダプタデータソースの中には、構造が複雑であるために、Developer tool でマッピングランタイムに必要なすべてのプロパティをインポートできないものがあります。

例えば、PowerExchange Microsoft Dynamics CRM ソースをインポートしても、そのソースの精度とスケールはインポートされません。さらに、Microsoft Dynamics CRM で特定のランタイム操作を実行したり、データ統合サービスが 1 回のバッチで読み取ることができるレコードの最大数を制御したりする必要がある場合もあります。これらのプロパティは、データオブジェクト読み取り操作で設定できます。

PowerExchange アダプタデータソースからデータをインポートした場合、ソースの複雑な構造に格納されたデータは、Developer tool のトランスフォーメーションと互換性がない可能性もあります。データオブジェクト読み取り操作を使用すると、ソースにネイティブのデータ型を、Developer tool がマッピングワークフローで使用するトランスフォーメーションデータ型に変換できます。

構造が複雑な PowerExchange アダプタデータソースのターゲットにデータを書き込む場合にも、同様にデータオブジェクト書き込み操作を使用して、データをトランスフォーメーションデータ型からデータソースにネイティブのデータ型に戻すことが必要になることがあります。

リソースから物理データオブジェクトをインポートすると、その同じリソースに基づいて物理データオブジェクトのデータオブジェクト操作を作成できます。リソースは、データを読み取る対象のデータオブジェクトの一部です。例えば、データベースのリソースはテーブルやビューになります。

物理データオブジェクトおよび必要なデータオブジェクト操作を作成した後、オブジェクトエディタで物理データオブジェクトを表示できます。オブジェクトエディタビューには、データオブジェクト操作のプロパティを設定できるタブが含まれています。1 つのデータオブジェクトに複数の読み取り操作および書き込み操作を設定できます。

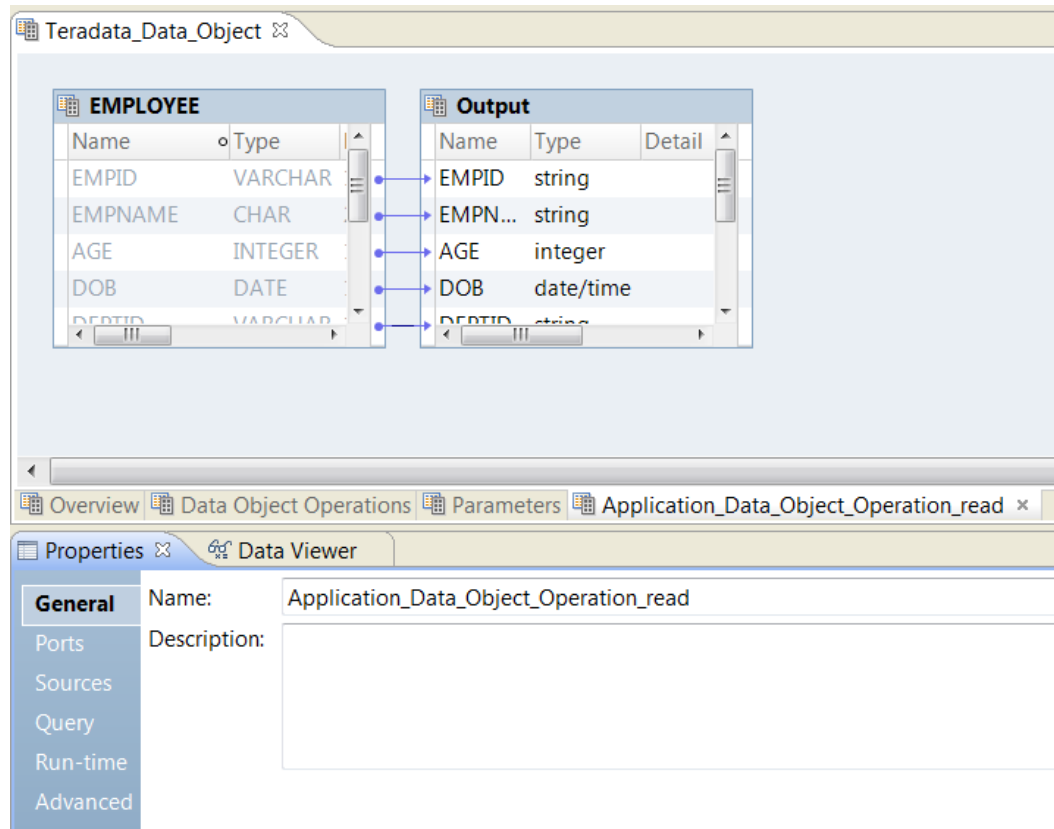
### データオブジェクト読み取り操作

データオブジェクト読み取り操作は、ソースデータオブジェクトに関連付けられます。オブジェクトエディタで、データオブジェクト読み取り操作を作成し、そのプロパティを設定できます。

オブジェクトエディタでデータオブジェクト読み取り操作を表示すると、ソースオブジェクトと出力オブジェクトが表示されます。このソースオブジェクトと出力オブジェクトがデータオブジェクト読み取り操作を構成しています。出力オブジェクトでは、詳細プロパティとランタイムプロパティを編集します。

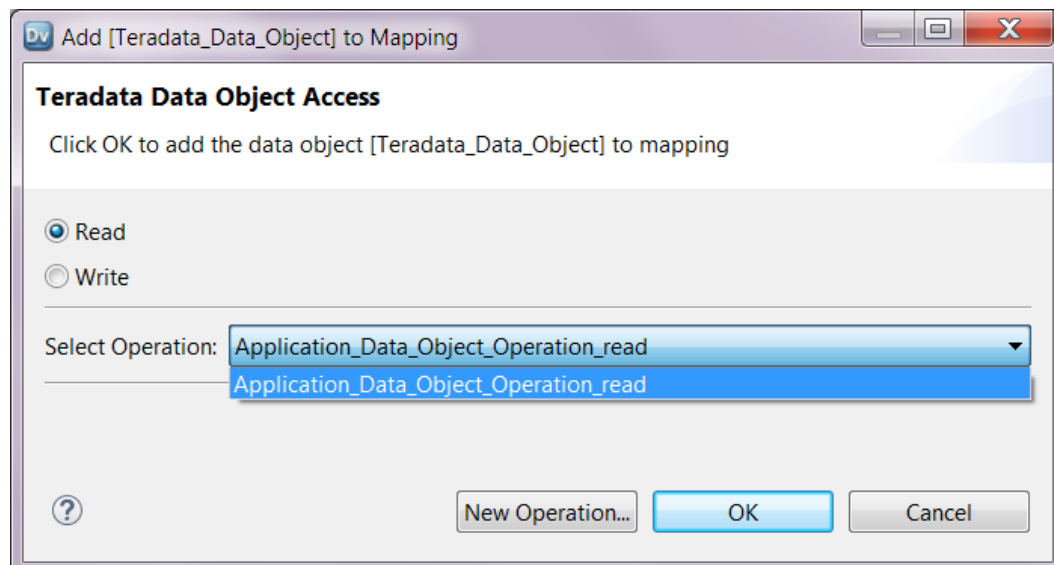


次の図は、ソースオブジェクトが EMPLOYEE で、出力オブジェクトが Output である Teradata データオブジェクトのデータオブジェクト読み取り操作の例を示しています。



物理データオブジェクトのデータオブジェクト読み取り操作を作成した後、読み取りトランスフォーメーションを作成して、マッピングワークフローのソースとして物理データオブジェクトを追加できます。物理データオブジェクトをマッピングに追加するときには、読み取りトランスフォーメーションと、使用するデータオブジェクト読み取り操作を指定できます。

次の図は、マッピングに Teradata データオブジェクトを追加したときに表示されるウィザードを示しています。



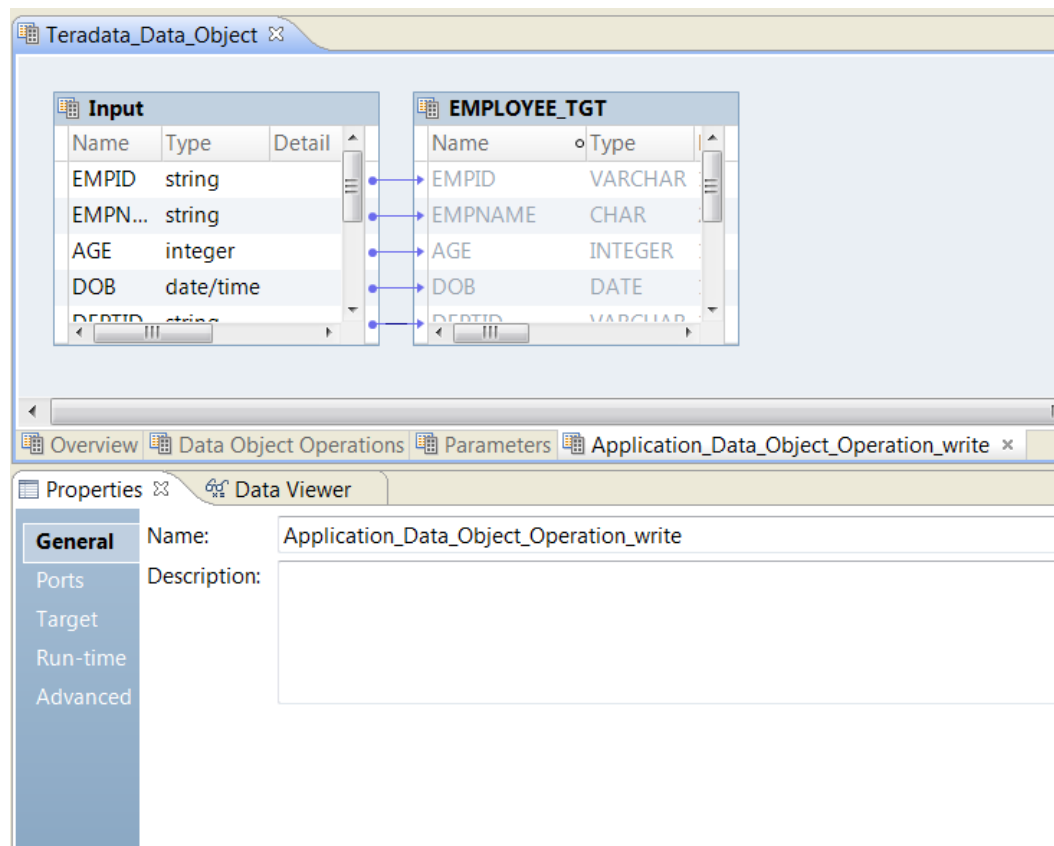
読み取りトランスフォーメーションでは、データオブジェクト読み取り操作の出力オブジェクトで設定したプロパティが使用されます。

## データオブジェクト書き込み操作

データオブジェクト書き込み操作は、ターゲットデータオブジェクトに関連付けられます。オブジェクトエディタでは、データオブジェクト書き込み操作を作成し、そのデータオブジェクト書き込み操作のプロパティを設定できます。

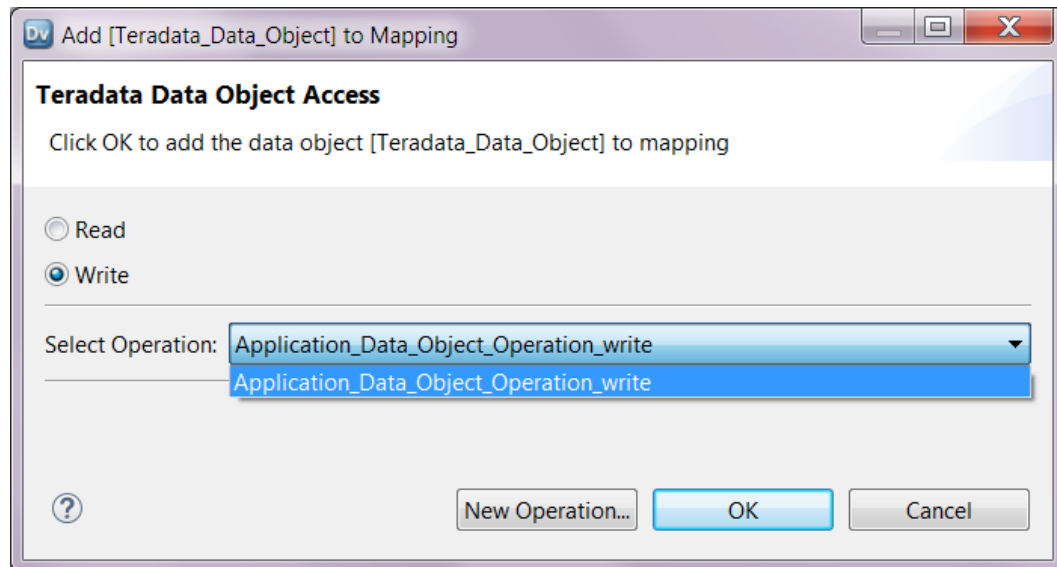
オブジェクトエディタでデータオブジェクト書き込み操作を表示すると、オブジェクトエディタに入力オブジェクトとターゲットオブジェクトが表示されます。入力オブジェクトおよびターゲットオブジェクトには、データオブジェクト書き込み操作があります。入力オブジェクトでは、詳細プロパティとランタイムプロパティを編集します。

次の図は、入力オブジェクトが Input で、ターゲットオブジェクトが EMPLOYEE\_TGT である Teradata データオブジェクトのデータオブジェクト書き込み操作の例を示しています。



物理データオブジェクトのデータオブジェクト書き込み操作を作成した後、書き込みトランスフォーメーションを作成して、マッピングワークフローのターゲットとして物理データオブジェクトを追加できます。物理データオブジェクトをマッピングに追加すると、書き込みトランスフォーメーションと、使用するデータオブジェクト書き込み操作を指定できます。

次の図は、マッピングに Teradata データオブジェクトを追加したときに表示されるウィザードを示しています。



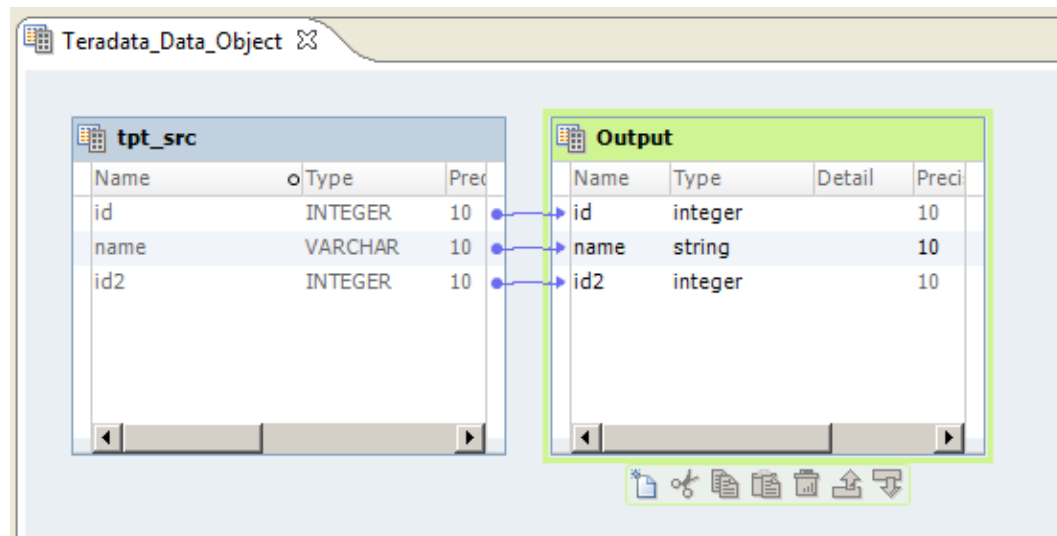
書き込みトランスフォーメーションでは、データオブジェクト書き込み操作の入力オブジェクトで設定したプロパティが使用されます。

## Teradata データオブジェクトの例

Teradata データオブジェクトは、Teradata リソースに基づいてメタデータを表します。

Teradata データオブジェクトのデータオブジェクト読み取り操作を設定して、オブジェクトエディタで表示すると、ソースオブジェクトと出力オブジェクトが表示されます。出力オブジェクトには、Teradata リソースからのメタデータが表示されます。

次の図は、Teradata データオブジェクトのデータオブジェクト読み取り操作を示しています。



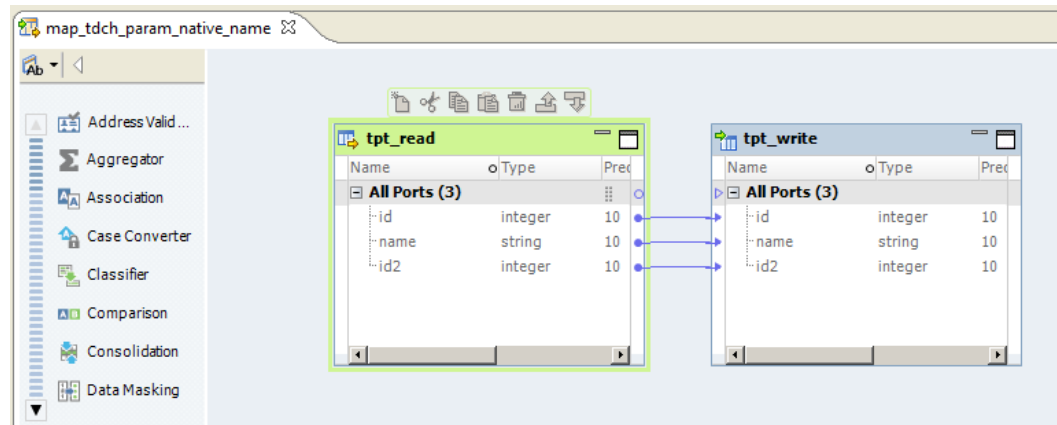
ソースオブジェクトは tpt\_src で、出力オブジェクトは Output です。このソースオブジェクトと出力オブジェクトがデータオブジェクト読み取り操作を構成しています。

出力オブジェクトのメタデータを見ると、tpt\_src のネイティブデータ型が出力オブジェクトのトランスフォーメーションデータ型に変換されていることがわかります。例えば、**name** ポートのネイティブデータ型 VARCHAR は、トランスフォーメーションデータ型 string に変換されます。

マッピングで Teradata データオブジェクトを使用するには、読み取りトランスフォーメーションを作成します。マッピングに読み取りトランスフォーメーションを追加するときには、以前に設定したデータオブジェクト読み取り操作と、Teradata データオブジェクトを選択します。

読み取りトランスフォーメーションでは、データオブジェクト読み取り操作の出力オブジェクトに格納されているトランスフォーメーションデータ型にアクセスする必要があるため、Teradata データオブジェクトではなく、オブジェクト操作に基づいて読み取りトランスフォーメーションが作成されます。読み取りトランスフォーメーションでは、データオブジェクト読み取り操作に対して設定したのと同じプロパティが使用されます。

次の図は、マッピングエディタで、読み取りトランスフォーメーション tpt\_read をハイライトしています。



読み取りトランスフォーメーションのメタデータと、データオブジェクト読み取り操作の出力オブジェクトが同じであることに注意してください。例えば、**name** ポートのメタデータは、ネイティブデータ型 VARCHAR ではなく、トランスフォーメーションデータ型 string を使用します。

## トランスフォーメーション

トランスフォーメーションは、データの生成、変更、または受け渡しを行うオブジェクトです。

Informatica Developer には、特定の関数を実行する一連のトランスフォーメーションが用意されています。例えば、アグリゲータトランスフォーメーションはデータのグループに対して計算を実行します。マッピング内のトランスフォーメーションは、データ統合サービスがデータに対して実行する処理を表します。データは、マッピングまたはマップレット内のリンクされたトランスフォーメーションポートを通過します。

トランスフォーメーションはアクティブであるかパッシブであるかのいずれかです。トランスフォーメーションはデータフローに接続されているか、接続されていないかのいずれかです。トランスフォーメーションの詳細については、『Developer トランスフォーメーションガイド』を参照してください。

## マップレット

マップレットは、複数のマッピングで使用できるトランスフォーメーションのセットを含む再利用可能オブジェクトです。

マッピング内でマップレットを使用する際には、そのインスタンスを用います。マップレットに加えた変更はそのインスタンスすべてに引き継がれます。マップレットには、他のマップレットを含めることができます。また、1つのマッピングまたはマップレットで、マップレットを複数回使用することもできます。マップレットは手動で作成できます。マッピングまたはマップレット内のセグメントからマップレットを生成することもできます。

マップレットの詳細については、[第2章、「マップレット」 \(ページ 42\)](#)を参照してください。

## セグメント

セグメントは、マッピング、マップレット、ルール、または仮想ストアドプロシージャ内の 1 つ以上のオブジェクトで構成されます。セグメントには、ソース、ターゲット、トランスフォーメーション、またはマップレットを含めることができます。

フォルダーまたはプロジェクト間でセグメントをコピーできます。セグメントをコピーするときは、次のルールおよびガイドラインに従います。

- Developer ツールは、可能な場合には依存関係を再利用します。依存関係を再利用できない場合は、依存関係をコピーします。
- マッピング、マップレット、ルール、または仮想ストアドプロシージャにパラメータが含まれ、そのパラメータを参照するトランスフォーメーションをコピーした場合、コピー先のオブジェクトのトランスフォーメーションではパラメータのデフォルト値が使用されます。
- 入力トランスフォーメーションおよび出力トランスフォーメーションはコピーできません。
- セグメントをコピーした後、前のアクションを取り消すことはできません。

マッピングまたはマップレットのセグメントからマップレットを生成することもできます。マッピングまたはマップレットに、再使用する予定の接続されたトランスフォーメーションフローが含まれている場合に、マップレットを生成する場合があります。マップレットの生成方法の詳細については、[「マップレットの生成」\(ページ 45\)](#)を参照してください。

## セグメントのコピー

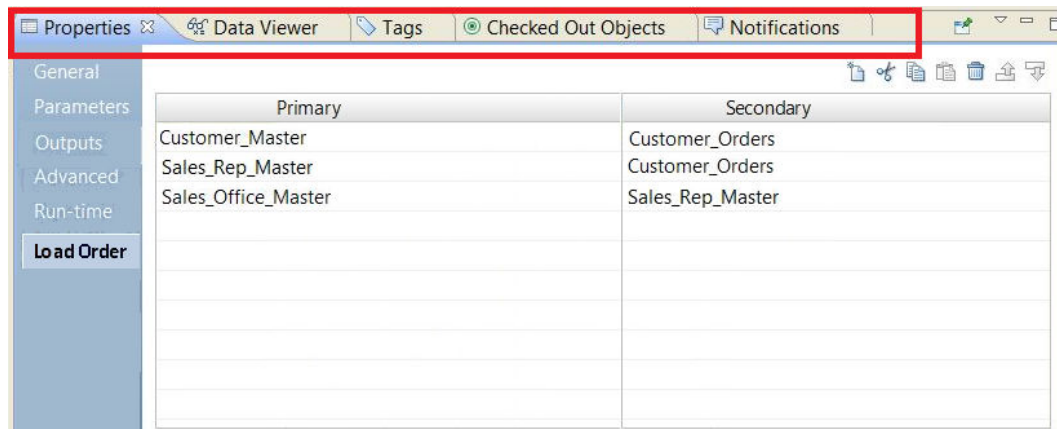
別のマッピング、マップレット、ルール、または仮想ストアドプロシージャのマッピングロジックの一部を再利用する場合、セグメントをコピーできます。

1. コピーするセグメントを含むオブジェクトを開きます。
2. コピーする各オブジェクトをハイライト表示して、セグメントを選択します。  
複数のオブジェクトを選択するには Ctrl キーを押します。また、エディタでマウスをドラッグしてオブジェクトを囲んでもセグメントを選択できます。
3. **【編集】 > 【コピー】** をクリックして、セグメントをクリップボードにコピーします。
4. コピー先のマッピング、マップレット、ルール、または仮想ストアドプロシージャを開きます。
5. **【編集】 > 【貼り付け】** をクリックします。

## ビュー

エディタ内部をクリックすると、各種のビューを表示できるようになります。ビューは、情報階層を移動したり、オブジェクトプロパティを表示したりできるワークベンチパーツです。これらのビューを切り替えると、エディタでプロパティや他の詳細情報を変更できます。またこれらのビューを使用して、エディタに表示させるオブジェクトを選択することもできます。

次の図は、Informatica Developer の各種のビューを示しています。



次に示す各ビューを切り替えて、さまざまなタスクを実行します。

### プロパティ

マッピング名、ランタイムプロパティ、ロード順序の制約など、一般的なマッピングプロパティを設定します。

### データビューア

各トランスフォーメーションのデータをプレビューし、マッピング出力を表示します。[データビューア]ビューでは、データをエクスポートすることもできます。

### タグ

メタデータを追加するタグを作成し、タグをオブジェクトに割り当て、オブジェクトに割り当てられたすべてのタグを表示します。

### チェックアウトされたオブジェクト

チェックアウトしたオブジェクトを表示します。

### 通知

スコアカード通知のグローバル設定を設定します。ワークフロー中に通知を受け取る Informatica ドメイン内の受信者を選択することもできます。

## マッピングの検証

マッピングを開発する場合、Data Integration Service がマッピング全体を読み込み、そして処理できるようにマッピングを設定する必要があります。Data Integration Service でのマッピングの実行を妨げるエラーが Developer ツールで検出された場合、そのマッピングは無効とマークされます。

Developer tool は次のタイプの検証を実行します。

- 接続検証
- 式検証
- オブジェクト検証
- パラメータの検証
- 同期の検証

## 接続の検証

マッピング内のポートの接続、マッピングの検証を行うたびに、Developer ツールによって接続の検証が行われます。

ポートを接続すると、Developer ツールは有効な接続かどうかを検証します。マッピングを検証すると、Developer tool は接続が有効かどうか、必要なポートがすべて接続されているかどうかを検証します。

Developer tool は以下の接続の検証を実行します。

- 少なくとも 1 つの入力オブジェクトおよび出力オブジェクトが接続されている。
- 少なくとも 1 つのマプレット入力ポートおよび出力ポートがマッピングに接続されている。
- データ型はポート間で互換性がある。ポートのデータ型を接続先のポートと互換性のないものに変更すると、Developer tool によってエラーが生成され、マッピングが無効になります。ただし、Char や Varchar のように接続したポートとの互換性が維持される場合は、データ型を変更することができます。

## 式の検証

マッピングを作成するときに、トランスフォーメーション内の式を検証することができます。エラーを修正しない場合、マッピングを検証すると **【検証ログ】** ビューにエラーメッセージが表示されます。

式で使用した入力ポートを削除すると、Developer ツールはマッピングを無効とマークします。

## オブジェクトの検証

マッピングを検証すると、Developer ツールは入力トランスフォーメーションやマプレットなど独立したオブジェクトの定義がマッピング内のインスタンスと一致するかどうかを検証します。

マッピングを設定する際にオブジェクトを 1 つでも変更すると、マッピングにエラーが含まれる場合があります。マッピングの設定中ではないときにオブジェクトのいずれかが変更された場合、Developer ツールは、その変更がマッピングに与える影響について追跡します。

## パラメータの検証

Developer tool でマッピングパラメータを検証するには、解決済みのパラメータを使用してマッピングを表示し、マッピングを検証します。

マッピングパラメータを解決すると、Developer tool では、実行時に解決されたパラメータを示すマッピングのランタイムインスタンスを生成します。マッピングのランタイムインスタンスを検証して、マッピングパラメータを検証します。マッピング、パラメータセット、またはパラメータファイルでデフォルトのパラメータ値を使用するマッピングパラメータを検証できます。

## 同期の検証

動的ソースおよびターゲットの実行時同期を検証できます。

実行時に同期するように設定されたデータソースを使用する動的マッピングを設定する場合、マッピングのランタイムインスタンスをプレビューして、同期を検証できます。マッピングのランタイムインスタンスをプレビューするには、解決済みパラメータを使用してマッピングを表示します。解決済みパラメータを使用してマッピングを表示すると、Developer tool は、動的ソースおよびターゲットでの実行時の変更を反映したマッピングのランタイムインスタンスを生成します。ランタイムマッピングを検証し、データソースの同期を検証します。

# マッピングランタイムプロパティ

マッピングランタイムプロパティはマッピングに選択した実行環境により異なります。

以下のマッピングランタイムプロパティを設定します。

- 検証環境
- 実行環境
- 拒否ファイルディレクトリ
- 最大並行処理
- ターゲットコミット間隔
- エラー時の停止
- 偽装ユーザー名のマッピング
- 提案された並行処理

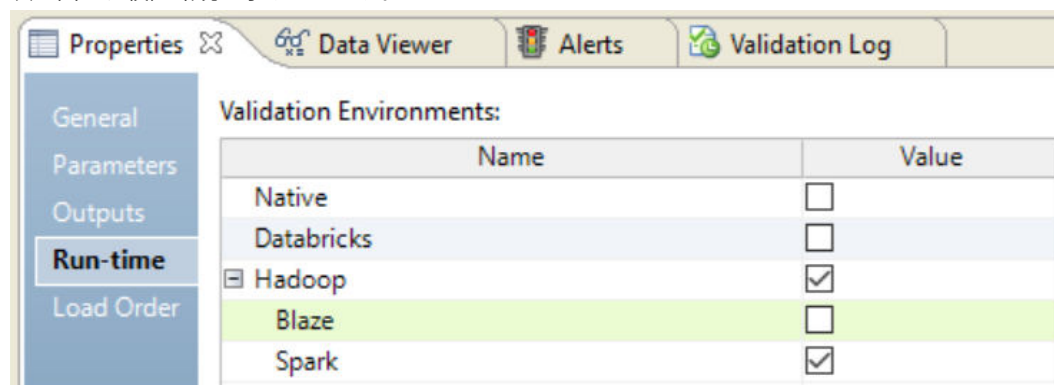
## 検証環境

検証環境は、Developer tool が、ネイティブまたは非ネイティブ実行環境のマッピング定義を検証したかどうかを示します。ネイティブ環境でマッピングを実行する場合は、データ統合サービスがマッピングを処理します。

ライセンスに基づいて、非ネイティブ環境でマッピングを実行できます。非ネイティブ環境でマッピングを実行する場合、データ統合サービスはクラスタ接続を使用してマッピング実行を計算クラスタにプッシュします。計算クラスタがマッピングを処理します。

Hadoop 実行環境を選択した場合、マッピングを処理するために、Blaze、Spark のいずれかのエンジンを選択できます。

次の図は、検証環境を示しています。



次の状況では、ネイティブ、Hadoop、および Databricks 環境を選択します。

- 非ネイティブ環境でマッピングを実行する前に、ネイティブ環境でマッピングをテストします。
- マッピングを実行するときは、パラメータ内で実行環境の値を定義します。

両方の環境を選択する場合は、[ランタイム] プロパティでマッピングの実行環境を選択する必要があります。

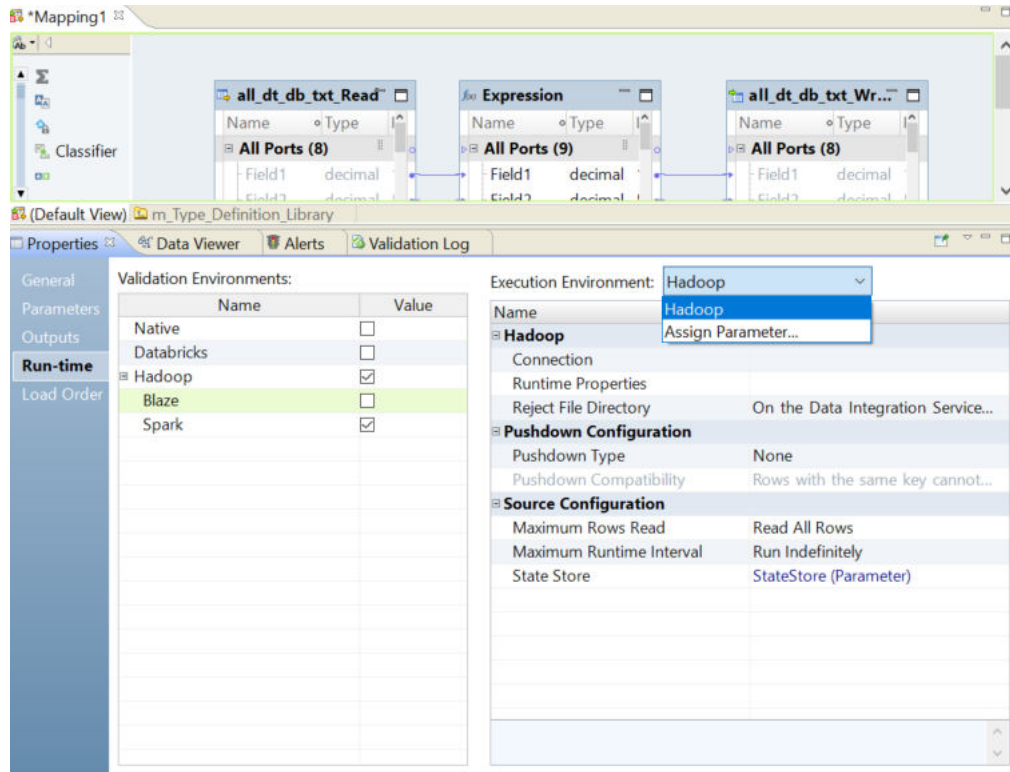


## 実行環境

マッピング実行時に使用する実行環境を選択します。ネイティブ環境でマッピングを実行する場合は、データ統合サービスがマッピングを処理します。非ネイティブ環境でマッピングを実行する場合、データ統合サービスはクラスタ接続を使用してマッピング実行を計算クラスタにプッシュします。計算クラスタがマッピングを処理します。

マッピングパラメータを使用して実行環境を示すことができます。文字列パラメータを設定します。実行環境を選択する場合、[パラメータの割り当て] をクリックして、設定したパラメータを選択します。

次の図に、マッピング実行環境を選択する場所を示します。



実行環境を選択する場合、Developer tool はマッピングの実行に関連付けられた検証環境のいずれかを保存します。

## 拒否ファイルディレクトリ

Hadoop 接続に拒否ファイルディレクトリが設定されている場合は、Hadoop 環境でマッピングを実行するときに、拒否ファイルを保存する場所を選択できます。Blaze エンジン、フラットファイルターゲット、HDFS ターゲット、および Hive ターゲットの拒否ファイルを Hadoop 環境に書き込むことができます。Spark エンジン、フラットファイル、および HDFS ターゲットの Hadoop 環境に拒否ファイルを書き込むことができます。

拒否ファイルは、データ統合サービスマシンまたは Hadoop クラスタに書き込むことができます。または、Hadoop 接続設定に従うこともできます。

次のいずれかのオプションを選択します。

- データ統合サービスマシン上。データ統合サービスは、RejectDir システムパラメータに基づいて拒否ファイルを格納します。

- Hadoop クラスタ上。拒否ファイルは、Hadoop 接続で設定された拒否ディレクトリに移動します。ディレクトリが設定されていない場合、マッピングは失敗します。
- Hadoop 接続に従います。拒否ファイルは、Hadoop 接続プロパティで拒否ディレクトリが有効になっているかどうかに基づいて移動します。拒否ディレクトリが有効になっている場合、拒否ファイルは Hadoop 接続で設定されている拒否ディレクトリに移動します。そうでない場合は、データ統合サービスは、RejectDir システムパラメータに基づいて拒否ファイルを格納します。

Hadoop 接続に従うようにマッピングランタイムプロパティを設定すると、アクティブな Hadoop 接続の拒否ファイルを書き込むように選択しているかどうかに基づいて、この設定が施されたすべてのマッピングの拒否ファイルが移動されます。拒否ファイルディレクトリを変更するためにマッピングランタイムプロパティを手動で変更する必要はありません。

例えば、拒否ファイルが現在データ統合サービスマシンに移動されていて、それらのファイルを Hadoop 接続に設定されたディレクトリに移動する場合は、拒否ファイルを書き込むように Hadoop 接続プロパティを編集します。Hadoop 接続に従うように設定されているすべてのマッピングの拒否ファイルが、設定したディレクトリに移動されます。

また、複数の Hadoop 接続を交互に使用するように接続がパラメータ化されている場合にも、Hadoop 接続に従うように選択できます。例えば、拒否ファイルを書き込むように設定されている Hadoop 接続と、拒否ファイルを書き込むように設定されている別の Hadoop 接続を交互に使用するようにパラメータを指定できます。Hadoop 接続に従うように選択すると、接続パラメータのアクティブな Hadoop 接続に応じて、拒否ファイルが移動されます。

実行時に拒否ファイルディレクトリを作成することはできません。拒否ファイルを保存する前に、選択するオプションに応じて、構成済みの拒否ファイルディレクトリがデータ統合サービスマシンまたは Hadoop 環境内に存在する必要があります。

## 最大並行処理

最大並行処理は、ネイティブ実行環境で有効です。最大並行処理は、単一のマッピングパイプラインステージを処理する並行スレッドの最大数を参照します。管理者は、マッピングのパーティション化を有効にするため、データ統合サービスの最大並行処理を 2 以上の値に設定します。管理者は、Administrator ツールで最大並行処理を設定します。

マッピングのデフォルトの最大並行処理値は、[自動] です。各マッピングは、データ統合サービスに定義されている最大並行処理値を使用します。デフォルトの最大並行処理値を変更し、特定のマッピングの最大値を定義できます。データ統合サービスとマッピングで設定されている最大並行処理の整数値が異なる場合、データ統合サービスでは最小値が使用されます。

デフォルトは [自動] です。最大値は 64 です。

パーティション化の詳細については、[第 15 章、「パーティション化されたマッピング」 \(ページ 278\)](#)を参照してください。

## ターゲットコミット間隔

ターゲットコミット間隔は、コミットの基準として使用する行の数を参照します。データ統合サービスは、処理するターゲット行の数およびターゲットテーブルの制約に基づいてデータをコミットします。データ統合サービスはコミット間隔をチューニングします。デフォルトのコミット間隔は 10,000 行です。

コミット間隔はデータ統合サービスがコミットを発行するおおよその間隔です。データ統合サービスはコミット間隔より前、間隔通り、またはコミット間隔より後にコミットを発行することがあります。通常、データ統合サービスはライタバッファブロック全体を書き込んだ後にターゲットコミット間隔をチェックします。

## エラー時の停止

読み取り、書き込み、またはトランスフォーメーションのスレッドで致命的でないエラーが発生した場合にマッピングを停止します。デフォルトでは無効になっています。

エラー時の停止を有効にすると、次のタイプのエラーによりマッピングが停止します。

### 読み取りエラー

ソースデータベースまたはソースファイルの読み取り中にデータ統合サービスで発生するエラー。読み取りエラーには Unicode モードでセッションを実行中のアライメントエラーが含まれます。

### 書き込みエラー

ターゲットデータベースまたはターゲットファイルへの書き込み中データ統合サービスに発生するエラー。書き込みエラーにはキー制約違反、非 NULL フィールドへの NULL のロード、およびデータベーストリガ応答が含まれます。

### トランスフォーメーションエラー

データを変換中にデータ統合サービスに発生するエラー。トランスフォーメーションエラーには、変換エラーおよび NULL 入力などの ERROR として設定されたすべての条件が含まれます。

## 偽装ユーザー名のマッピング

偽装ユーザー名のマッピングは、ネイティブ実行環境および Hadoop 実行環境で有効です。マッピングの偽装を使用して、データ統合サービスのユーザーを偽装し、Kerberos 認証を使用する Hive、HBase、または HDFS のソースとターゲットに接続します。

ユーザー名を <Hadoop service name>/<hostname>@<YOUR-REALM> の形式で入力します。

説明:

- Hadoop service name は、Hive、HBase、または HDFS のソースとターゲットが存在する Hadoop サービスの名前です。
- ホスト名は、Hadoop サービスの名前または IP アドレスです。
- YOUR-REALM は、Kerberos レalm です。

特殊文字「/」および「@」は、区切り文字としてのみ使用できます。

## 提案された並行処理

提案された並行処理は、最大並行処理プロパティが 1 より大きい値に割り当てられているか、パラメータに割り当てられている場合にネイティブ実行環境に対して有効です。トランスフォーメーションパイプラインステージを処理する並行スレッドの推奨数。

提案された並行処理値をトランスフォーメーションに定義すると、データ統合サービスによりそのトランスフォーメーションパイプラインステージの最適なスレッド数を判断するときにこの値が考慮されます。多数のポートが含まれるトランスフォーメーションや複雑な計算を実行するトランスフォーメーションでは、パフォーマンスを最適化するために提案された並行処理値を定義することをお勧めします。

デフォルトは [自動] で、トランスフォーメーションはマッピングに定義された最大並行処理数を使用します。最大値は 64 です。

# ターゲットロード順の制約

ターゲットのロード順序の制約は同じマッピングで互いに関係している 2 つのターゲットインスタンスに対して、データ統合サービスが行をロードおよびコミットする方法を制限します。

Developer tool で、制約を設定しデータ統合サービスが行をターゲットテーブルにロードする順序を制限できます。

データ統合サービスにプライマリターゲットインスタンスのデータを完全にロードした後に、セカンダリターゲットインスタンスにデータをロードさせるよう制約を設定することができます。プライマリターゲットとセカンダリターゲットとして定義するテーブルは入力行のトランザクションによって異なります。

ターゲットロード順の制約について次のシナリオを考慮します。

## マスタおよび詳細ターゲットへの行の挿入。

プライマリキーと外部キーのリレーションがあるターゲットに行を挿入する場合、ターゲットのロード順序の制約を設定することがあります。プライマリキーを持つターゲットをプライマリターゲットインスタンスとして設定します。外部キーを持つターゲットをセカンダリターゲットインスタンスとして設定します。データ統合サービスは、プライマリターゲットのロードが完了するまで、セカンダリターゲットのデータをステージングできます。

## マスタおよび詳細ターゲットからの行の削除。

プライマリキーと外部キーのリレーションがあるターゲットから行を削除する必要がある場合は、異なる制約を設定します。外部キーを持つターゲットをプライマリターゲットインスタンスとして設定し、詳細ターゲットから先に行を削除します。プライマリキーを持つターゲットをセカンダリターゲットインスタンスとして設定します。

## 同じリレーショナルテーブルへの行の挿入と行の更新。

2 つの別のトランスフォーメーションからリレーショナルテーブルに行の挿入と行の更新をロードするマッピングに対して、ターゲットのロード順序の制約を設定できます。データ統合サービスで行の挿入のロードが完了するまで行の更新のロードが制限されるよう制約を設定します。

## フラットファイルのターゲットロード順。

複数のフラットファイルターゲットに行をロードするマッピングに対して、ターゲットロード順の制約を設定できます。プライマリフラットファイルの後にセカンダリフラットファイルをロードするように、ターゲットロード順を設定します。

1 つのマッピングに複数の制約を設定できます。データ統合サービスは制約を違反せずにターゲットをロードする最も効率的な実行計画を決定します。

# 挿入行と削除行に関する制約

ターゲットロード順序の制約には、同じファイルの挿入行、更新行、および削除行の特殊な処理は含まれていません。

挿入行、更新行、および削除行を処理する必要がある場合、削除行とは異なるターゲットインスタンスに挿入行および更新行を返すようにルータートランスフォーメーションを設定できます。ターゲットロード順序の制約を設定して、ターゲットをロードする順序を指定します。

例えば、Order\_Header および Order\_Detail というターゲットがあるとします。Order\_Detail テーブルには、Order\_Header テーブルへの OrderID 外部キーがあります。両方のテーブルの挿入、更新、および削除を処理する必要があります。

ルータートランスフォーメーションを使用して、挿入行と更新行を削除行から切り離すことができます。ルータートランスフォーメーションからの次の出力グループを設定します。

1. Order\_Header の挿入行および更新行

2. Order\_Header の削除行
3. Order\_Detail の挿入行および更新行
4. Order\_Detail の削除行

これらの行をターゲットにロードする場合、次の制約を作成できます。

```
Group #4 before group #2
Group #2 before group #1
Group #1 before group #3
```

これらの制約により、データ統合サービスは、Order\_Header の削除より前に Order\_Detail の削除を処理します。データ統合サービスは、挿入行および更新行の前にすべての削除を処理します。また、Order\_Detail の挿入および更新の前に Order\_Header の挿入および更新を処理します。

## ターゲットのロード順に関するルールとガイドライン

ターゲットロード順の制約を定義する場合は、以下のルールおよびガイドラインに従います。

- Developer tool で、一部のターゲットカラムをプライマリキーまたは外部キーとして設定できます。ロード順序の制約では、これらのキーは無視されます。ターゲットにプライマリキーと外部キーの制約がある場合は、ロード順序の制約を定義する必要があります。
- Developer tool は定義されたロード順序の制約を検証しません。Developer tool は、マッピングを検証するときにロード順序の制約を検証します。
- データ統合サービスは、ターゲットロード順の制約のセカンダリターゲットインスタンスのデータをローカルディスクにステージングできます。マッピングに複数のセカンダリターゲットインスタンスがある場合、データ統合サービスは制約に違反することなく、ステージングされたデータをターゲットにロードします。
- データ統合サービスは、行が挿入、削除、または更新のいずれであるかを判別することなく、1つのターゲットインスタンスに続いて別のターゲットインスタンスをロードします。プライマリキーと外部キーの制約があるターゲットテーブルの場合、孤立行はプライマリキーターゲットに一致する行がない外部キーターゲットの行です。データ統合サービスは孤立行をチェックしません。データ統合サービスはロード順序の制約に指定された順序ですべての行をロードします。

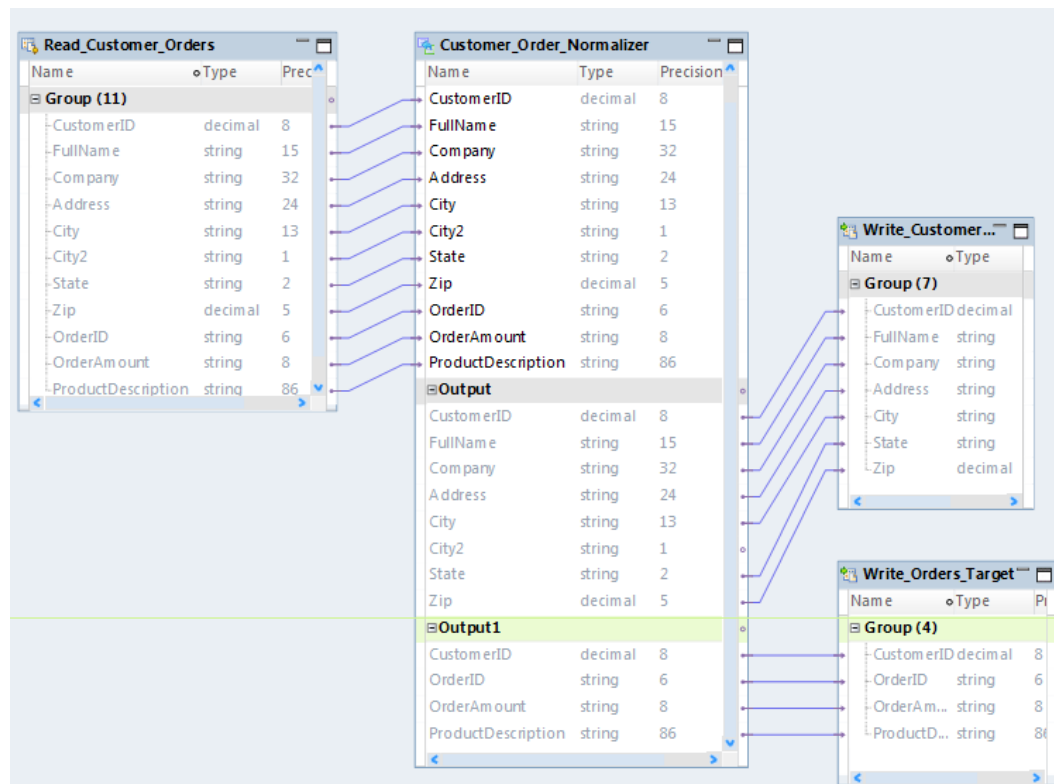
**警告:** 一時ファイルまたはファイル内のデータの使用を避けることを強くお勧めします。ステージングファイルやテーブル内のデータをユーザーが変更したことによりデータが破損しても、Informatica 社では責任を負いかねます。ステージングファイルの構造は、Informatica のバージョンによって異なる場合があります。

## ターゲットのロード順の例

組織は顧客注文を 1 日に 2 回処理します。同じトランザクションファイルで顧客情報と注文情報を受信します。組織は注文ファイルを処理するマッピングに、注文をロードする前に顧客情報をロードすることを確実に実行させる必要があります。

開発者は顧客情報を Customer\_Target テーブルに返すマッピングを作成します。マッピングは注文を Orders\_Target テーブルに返します。Customer\_Master のプライマリキーは CustomerID です。注文テーブルの各注文には Customer\_Master の CustomerID への外部キーがあります。開発者はターゲットロード順の制約を作成します。制約は、顧客情報のターゲットへのロードが完了するまで、データ統合サービスが注文をロードするのを制限します。

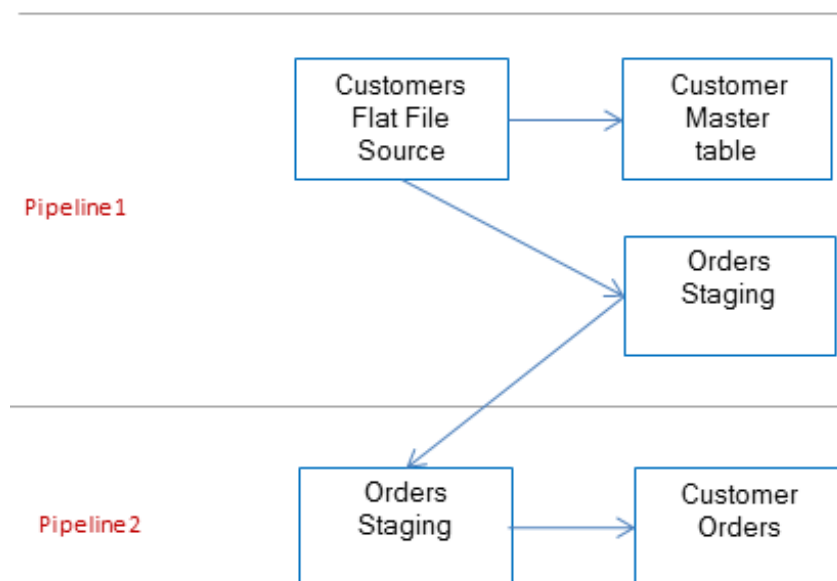
次の画像は、マッピングを示しています。



ノーマライズトランスフォーメーションは、顧客および注文データの個別の出力グループを作成します。開発者は注文データより前に顧客データのターゲットへのロードが確実に行われるようにする必要があります。

データ統合サービスでは、別のストラテジを使用してターゲットロード順の制約を実現できます。次の例では、データ統合サービスは2つのパイプラインを作成して、顧客データと注文データをターゲットテーブルにロードします。

次の図に、顧客データと注文データをターゲットテーブルにロードするパイプラインを示します。





最初のパイプラインで、データ統合サービスは顧客を Customer\_Master にロードして、注文をローカルディスクファイルにステージングします。2 番目のパイプラインでは、ステージングされた注文を注文テーブルにロードします。

## マッピング設定

マッピング設定は、マッピングを実行したときに Developer tool が使用するマッピングデプロイメントプロパティを制御します。

デフォルトのマッピング設定、再利用可能なマッピング設定、またはカスタムマッピング設定を構成できます。デフォルトのマッピング設定は、デフォルトのマッピングプロパティを指定します。再利用可能なマッピング設定またはカスタムマッピング設定が指定されていない限り、データ統合サービスはデフォルトのマッピング設定を使用してマッピングを実行します。

次の表に、マッピング設定タイプを示します。

マッピング設定	説明
デフォルトのマッピング設定	デフォルトのマッピング設定は、デフォルトのマッピングデプロイメントプロパティを定義します。再利用可能なマッピング設定またはカスタムマッピング設定が指定されていない限り、データ統合サービスはデフォルトのマッピング設定を使用してマッピングを実行します。 Developer tool の設定で、デフォルトのマッピング設定を構成します。
再利用可能なマッピング設定	再利用可能なマッピング設定を使用して、マッピングの実行時にデフォルトのマッピング設定をオーバーライドします。マッピング設定は、現在のマッピングを実行している間保存されます。 <b>【実行設定】</b> ダイアログボックスで、再利用可能なマッピング設定を構成します。 <b>【実行設定】</b> ダイアログボックスでマッピング設定を選択してマッピングを実行するか、詳細オプションを使用してマッピングを実行するときにマッピング設定を選択します。マッピングを実行するたびに、マッピング設定を選択する必要があります。
カスタムマッピング設定	カスタムマッピング設定を使用して、マッピングの実行時にデフォルトのマッピング設定をオーバーライドします。カスタムマッピング設定プロパティには、データ統合サービス、データ統合サービスのオペレーティングシステムプロファイル、オーバーライドトレースレベル、最適化レベルのみが含まれます。カスタムマッピング設定は、現在のマッピングを実行している間保存されます。 <b>【詳細オプションを使用したマッピングの実行】</b> ダイアログボックスでカスタムマッピング設定を構成します。マッピングを実行するたびに、カスタムマッピング設定を再構成する必要があります。

## マッピング設定プロパティ

[実行設定] ダイアログボックスでマッピング設定を構成するか、Developer tool の設定でデフォルトのマッピング設定を構成します。マッピングに設定できるプロパティには、データ統合サービスのプロパティ、ソースプロパティ、詳細プロパティが含まれています。

### データ統合サービスのプロパティ

以下の表に、Data Integration Service に対して構成するプロパティを示します。

プロパティ	説明
デフォルトのデータ統合サービスの使用	マッピングの実行にデフォルトのデータ統合サービスを使用します。 デフォルトでは有効になっています。
データ統合サービス	デフォルトのデータ統合サービスを使用しない場合に、マッピングを実行するデータ統合サービスを指定します。
使用可能なオペレーティングシステムプロファイル	データ統合サービスでオペレーティングシステムのプロファイルを使用できる場合、マッピングを実行するためのオペレーティングシステムのプロファイルを指定します。  Developer tool でこのプロパティが表示されるのは、管理者がオペレーティングシステムのプロファイルを少なくとも 1 つユーザーに割り当てた場合のみです。データ統合サービスはユーザーに割り当てられたデフォルトのオペレーティングシステムのプロファイルを使用してマッピングを実行します。オペレーティングシステムのプロファイルは、使用可能なオペレーティングシステムのプロファイルのリストから変更できます。

### ソースプロパティ

以下の表に、ソースに設定するプロパティを示します。

プロパティ	説明
すべての行の読み取り	ソースからすべての行を読み取ります。 デフォルトでは有効になっています。
次の行数まで読み取る	すべての行を読み取らない場合にソースから読み取る最大行数を指定します。 注: カスタマイズデータオブジェクトに書き込むマッピングにこのオプションを指定すると、Data Integration Service はターゲットに書き込む前にターゲットテーブルを切り詰めません。 デフォルトは 1000 です。
すべての文字を読み取る	カラム内のすべての文字を読み取ります。 デフォルトでは無効になっています。
次の文字数まで読み取る	すべての文字を読み取らない場合に各カラムで読み取る最大文字数を指定します。SAP ソースの場合、このプロパティは無視されます。 デフォルトは 4000 です。



## 詳細プロパティ

以下の表に詳細プロパティを示します。

プロパティ	説明
デフォルトの日時形式	マッピングで文字列が日付に変換される場合に、データ統合サービスが使用する日付/時刻形式。 デフォルトは MM/DD/YYYY HH24:MI:SS です。
トレースレベルのオーバーライド	マッピングの各トランスフォーメーションのトレースレベルをオーバーライドします。トレースレベルは、データ統合サービスがマッピングログファイルに送信する情報量を決定します。 以下のいずれかのトレースレベルを選択します。 <ul style="list-style-type: none"> <li>- なし。データ統合サービスは、マッピングに設定されたトレースレベルを使用します。</li> <li>- Terse。データ統合サービスは、初期化情報、エラーメッセージ、および却下されたデータの通知をログに記録します。</li> <li>- ノーマル。データ統合サービスは、初期化情報、ステータス情報、検出されたエラー、およびトランスフォーメーション行エラーが原因でスキップされた行をログに記録します。マッピングの結果を要約しますが、個別行のレベルでは行いません。</li> <li>- Verbose initialization。ノーマルトレースに加え、データ統合サービスにより、追加の初期化の詳細、インデックスおよび使用されたデータファイルの名前、および詳細なトランスフォーメーション統計もログに記録されます。</li> <li>- 冗長データ。データ統合サービスにより、Verbose 初期化トレースに加え、マッピングに渡された各行がログに記録されます。また、カラムの精度に合わせて文字列データを切り詰めた場所と、詳細なトランスフォーメーション統計も記録します。</li> </ul> デフォルトは [なし] です。
ソート順	データ統合サービスがマッピングで文字データをソートする順番。 デフォルトは [バイナリ] です。
最適化レベル	データ統合サービスがマッピングに適用する最適化の方法を制御します。 <b>自動</b> データ統合サービスは、実行モードとマッピングコンテンツに基づいて最適化を適用します。 <b>なし</b> データ統合サービスは最適化は適用されません。 <b>最小</b> データ統合サービスは初期プロジェクション最適化方式を適用します。 <b>ノーマル</b> データ統合サービスは、初期プロジェクション、初期選択、ブランチ刈り込み、プッシュイン、グローバル述部、述部の最適化方式を適用します。 <b>完全</b> データ統合サービスは、コストベース、初期プロジェクション、初期選択、ブランチ刈り込み、述部、プッシュイン、準結合、データシップ結合の最適化方式を適用します。 デフォルトは [通常] です。

プロパティ	説明
高精度	マッピングを高精度で実行します。 [高精度] を選択すると、データ値の精度が高くなります。15 桁以上の精度など、マッピングが大きな数値を生成する場合や、正確な値を必要とする場合に [高精度] を有効にします。[高精度] を有効にすると、桁数の多い数値で精度の損失を防ぐことができます。 デフォルトでは有効になっています。
クライアントにログを送信	Developer ツールでログファイルを表示できるようにします。このオプションを無効にした場合、ログファイルは Administrator ツールで表示する必要があります。 デフォルトでは有効になっています。

## デフォルトマッピング設定の構成方法

Developer tool の設定で、デフォルトのマッピング設定プロパティを更新します。マッピング設定が指定されていない場合、Developer tool はデフォルトのマッピング設定プロパティを使用してマッピングを実行します。

- Developer tool で、**【ウィンドウ】 > 【設定】** をクリックします。  
[設定] ダイアログボックスが表示されます。
- 【Informatica】 > 【実行設定】 > 【マッピング】** に移動します。
- デフォルトのマッピング設定プロパティを構成します。
- 【OK】** をクリックします。  
Developer tool がデフォルトのマッピング設定プロパティを更新します。

## 再利用可能なマッピング設定の作成方法

[実行設定] ダイアログボックスで、再利用可能なマッピング設定を作成します。マッピング設定のプロパティを指定します。Developer tool でマッピングを実行するときは、マッピング設定を選択します。

- Developer tool で、**【実行】 > 【実行ダイアログを開く】** をクリックします。  
[実行設定] ダイアログボックスが表示されます。
- 【マッピング設定】** を右クリックして、**【新規】** を選択します。
- 新しいマッピング設定を選択します。
- マッピング設定名を入力します。
- マッピング設定のプロパティを構成します。

## カスタムマッピング設定の作成および使用方法

詳細オプションを使用してマッピングを実行する前に、カスタムマッピング設定を構成します。マッピングを実行するたびに、カスタムマッピング設定を再構成する必要があります。

- Developer tool で、エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックします。**【詳細オプションを使用したマッピングの実行】** を選択します。  
[詳細オプションを使用したマッピングの実行] ダイアログボックスが表示されます。

2. **【カスタムマッピング設定の指定】**を選択します。
3. カスタムマッピング設定のプロパティを構成します。データ統合サービス、データ統合サービスのオペレーティングシステムプロファイル、オーバーライドトレースレベル、最適化レベルを指定します。
4. **【実行】**をクリックします。  
カスタムマッピング設定を使用してマッピングが実行されます。

## 詳細マッピングオプション

Developer tool で詳細オプションを使用してマッピングを実行できます。詳細マッピングオプションには、マッピング設定オプションとマッピングパラメータオプションが含まれます。

マッピング設定とマッピングパラメータは、マッピングを実行する都度指定します。マッピング実行時に再利用可能なマッピング設定を選ぶか、使用するマッピング設定をカスタマイズします。マッピング内のデフォルトのパラメータ値を使用するか、パラメータセットまたはパラメータファイル内のパラメータ値を使用します。指定したマッピング設定およびマッピングパラメータは、現在のマッピングの実行に対して保持されます。

詳細オプションを使用して、パラメータが解決されたマッピングインスタンスを実行する場合、マッピングパラメータを指定してマッピングを実行することはできません。マッピング内で解決されたパラメータを使用してマッピングが実行されます。

次の表に、マッピング設定の指定に使用するオプションを記載します。

オプション	説明
マッピング設定の選択	ドロップダウンメニューからマッピング設定を選択します。新しいマッピング設定を作成するには、 <b>【新規設定】</b> を選択します。
カスタムマッピング設定の指定	現在のマッピングを実行している間保存されるカスタムマッピング設定を作成します。

次の表に、カスタムマッピング設定の指定時に設定するプロパティを記載します。

プロパティ	説明
デフォルトのデータ統合サービスの使用	マッピングの実行にデフォルトのデータ統合サービスを使用します。デフォルトでは有効になっています。
データ統合サービス	デフォルトのデータ統合サービスを使用しない場合に、マッピングを実行するデータ統合サービスを指定します。
使用可能なオペレーティングシステムプロファイル	データ統合サービスでオペレーティングシステムのプロファイルを使用できる場合、マッピングを実行するためのオペレーティングシステムのプロファイルを指定します。
トレースレベルのオーバーライド	マッピングの各トランスフォーメーションのトレースレベルをオーバーライドします。トレースレベルは、データ統合サービスがマッピングログファイルに送信する情報量を決定します。
最適化レベル	データ統合サービスがマッピングに適用する最適化方式を制御します。

次の表に、マッピングパラメータの指定に使用するオプションを記載します。

マッピングパラメータ	説明
マッピングでのデフォルト値の適用	マッピングパラメータに設定されたデフォルト値に基づいて、マッピングパラメータを解決します。マッピングのパラメータが設定されていないと、マッピングではパラメータは解決されません。
パラメータセットの適用	指定されたパラメータセットで定義されたパラメータ値に基づいて、マッピングパラメータを解決します。
パラメータファイルの適用	指定されたパラメータファイルで定義されたパラメータ値に基づいて、マッピングパラメータを解決します。

**注:** マッピングにパラメータが含まれない場合、マッピングパラメータを指定するオプションは無効になります。

## 詳細オプションを使用したマッピングの実行方法

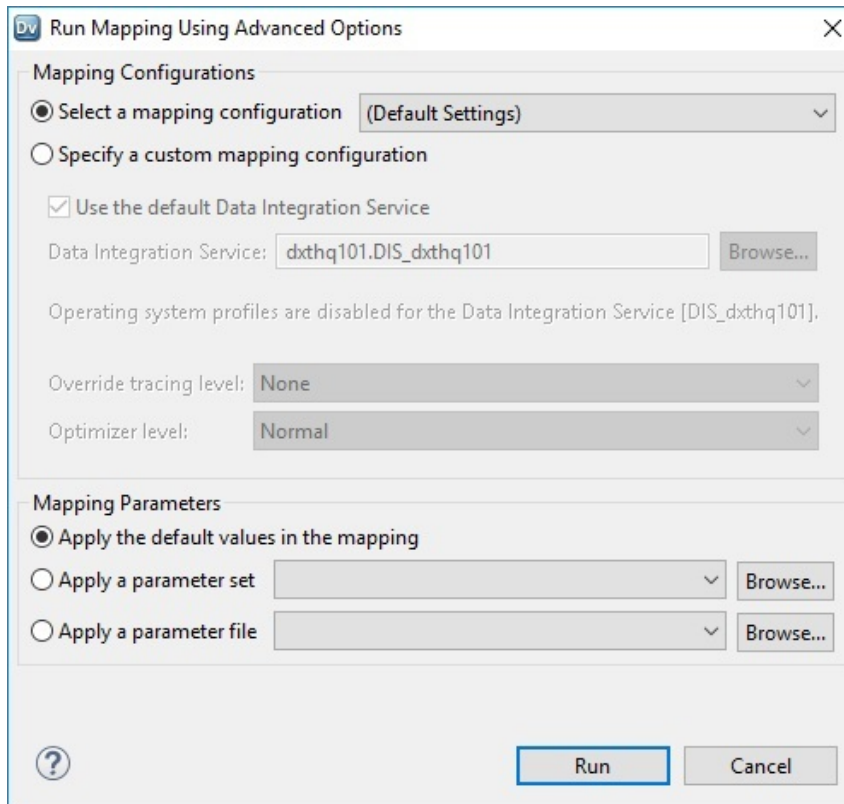
詳細オプションを使用してマッピングを実行するには、**【詳細オプションを使用したマッピングの実行】** ダイアログボックスを使用してマッピングを実行します。マッピング設定を選択するか、カスタムマッピング設定を

作成し、マッピングパラメータを指定します。指定したマッピング設定およびマッピングパラメータは、現在のマッピングの実行に対して保持されます。

詳細オプションを使用して、パラメータが解決されたマッピングインスタンスを実行する場合、マッピングパラメータを指定してマッピングを実行することはできません。マッピング内で解決されたパラメータを使用してマッピングが実行されます。

1. Developer tool で、エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックします。[詳細オプションを使用したマッピングの実行] を選択します。

[詳細オプションを使用したマッピングの実行] ダイアログボックスが表示されます。



The image shows a dialog box titled "Run Mapping Using Advanced Options". It is divided into two main sections: "Mapping Configurations" and "Mapping Parameters".

**Mapping Configurations:**

- ☒ Select a mapping configuration (Default Settings) [Dropdown arrow]
- ☐ Specify a custom mapping configuration
- ☒ Use the default Data Integration Service
- Data Integration Service: dxthq101.DIS\_dxthq101 [Browse...]
- Operating system profiles are disabled for the Data Integration Service [DIS\_dxthq101].
- Override tracing level: None [Dropdown arrow]
- Optimizer level: Normal [Dropdown arrow]

**Mapping Parameters:**

- ☒ Apply the default values in the mapping
- ☐ Apply a parameter set [Dropdown arrow] [Browse...]
- ☐ Apply a parameter file [Dropdown arrow] [Browse...]

At the bottom, there is a question mark icon, a "Run" button, and a "Cancel" button.

2. 次のいずれかのマッピング設定オプションを選択します。
  - マッピング設定の選択再利用可能なマッピング設定を選択し、マッピングを実行します。
  - カスタムマッピング設定の指定データ統合サービス、オペレーティングシステムプロファイル、オーバーライドトレースレベル、最適化レベルを選択します。
3. オプション。マッピングにパラメータが含まれる場合、次のいずれかのパラメータオプションを選択できます。
  - マッピングでのデフォルト値の適用データ統合サービスは、マッピング内で設定されたデフォルトのパラメータ値を適用します。
  - パラメータセットの適用データ統合サービスは、パラメータセットで設定されたパラメータ値を適用します。
  - パラメータファイルの適用データ統合サービスは、パラメータファイルで設定されたパラメータ値を適用します。

# マッピングの開発方法

ビジネスニーズに従ってデータの読み取り、変換、および書き込みを行うためのマッピングを開発します。

マッピングを開発して実行するには、次のタスクを実行します。

1. 作成するマッピングのタイプを決定します。
2. マッピングで使用する必要がある、入力、出力、および再利用可能なトランスフォーメーションやマップレットなどの再利用可能なオブジェクトを作成します。  
物理データオブジェクト、論理データオブジェクト、または仮想テーブルをマッピングの入力および出力に使用できます。
3. マッピングを作成します。
4. オブジェクトをマッピングに追加します。入力および出力オブジェクトをマッピングに追加する必要があります。必要に応じて、トランスフォーメーションおよびマップレットを追加します。
5. マッピングオブジェクトをリンクしてソースからターゲットへのデータフローを作成します。マップレットやトランスフォーメーションを通じて、このフローに沿ってデータの追加、削除、または変更が行われます。
6. マッピングを検証してエラーを特定します。
7. モデルリポジトリにマッピングを保存します。
8. マッピングを実行してマッピング出力を確認します。

## マッピングの作成

マッピングを作成して、ソースとターゲット間でデータを移動し、データを変換します。

1. **【オブジェクトエクスプローラ】** ビューで、プロジェクトまたはフォルダを選択します。
2. **【ファイル】** > **【新規】** > **【マッピング】** をクリックします。
3. マッピング名を入力します。
4. **【完了】** をクリックします。  
空のマッピングがエディタで表示されます。

## マッピングへのオブジェクトの追加

ソースとターゲットの間のデータフローを決定するにはマッピングにオブジェクトを追加します。

- データオブジェクトをエディタにドラッグして、**【読み取り】** を選択し、データオブジェクトをソースとして追加します。
- データオブジェクトをエディタにドラッグして、**【書き込み】** を選択し、データオブジェクトをターゲットとして追加します。
- ルックアップトランスフォーメーションを追加するには、フラットファイルデータオブジェクト、論理データオブジェクト、参照テーブル、リレーショナルデータオブジェクトのいずれかをエディタにドラッグして、**【ルックアップ】** を選択します。
- 再利用可能なトランスフォーメーションを追加するには、トランスフォーメーションを **【オブジェクトエクスプローラ】** ビューの **【トランスフォーメーション】** フォルダからエディタにドラッグします。  
この手順を、追加する再利用可能なトランスフォーメーションごとに繰り返します。
- 再利用不可能なトランスフォーメーションを追加するには **【トランスフォーメーション】** パレットでトランスフォーメーションを選択して、エディタにドラッグします。  
この手順を、追加する再利用不可能なトランスフォーメーションごとに繰り返します。

- 再利用不可能なトランスフォーメーションごとに、ポートとプロパティを設定します。
- 必要に応じて、マプレットをエディタにドラッグします。

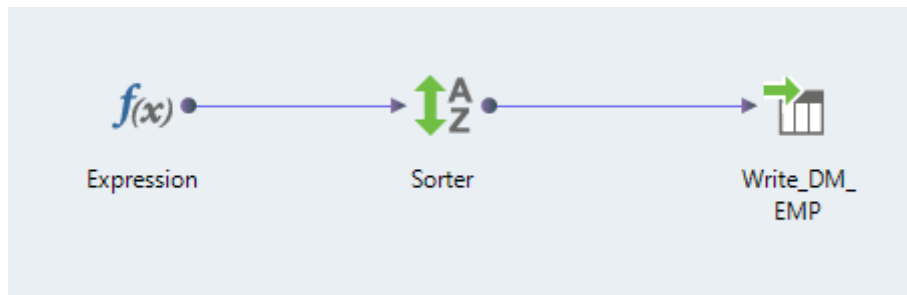
## マッピングオブジェクトの接続

マッピングオブジェクトを接続して、マッピングを完了します。ポートを介してマッピングオブジェクトを接続できます。データは、入力ポート、出力ポート、および入出力ポート経由でトランスフォーメーションとやり取りされます。

オブジェクトをマッピングに追加する際に、データ統合サービスのデータ変換方法に応じて、プロパティを接続します。エディタには、マッピングオブジェクトが以下の方法で表示されます。

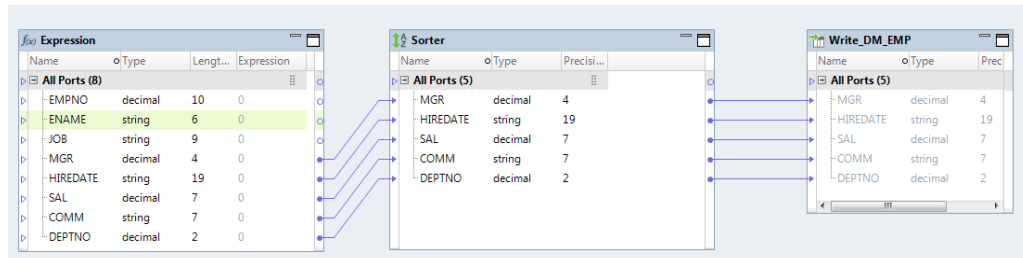
- アイコン化。オブジェクトのアイコンを、オブジェクト名と共に表示します。

次の図は、アイコン化されたオブジェクトのあるマッピングを示しています。



- ノーマル。コラムと、入力ポートおよび出力ポートのインジケータを表示します。ノーマルビュー内のオブジェクトは接続することができます。

次の図は、上記のアイコン化されたマッピングをノーマルで表示した場合を示しています。



入力オブジェクト、トランスフォーメーション、マプレット、および出力オブジェクト間でポートをリンクする場合、次のタイプのリンクを作成できます。

- 1対1のリンク。入力オブジェクトの1つのポートを、出力オブジェクトの1つのポートにリンクします。
- 1対多のリンク。1つのポートを複数の出力オブジェクトにリンクします。1つのオブジェクトの複数のポートを複数の出力オブジェクトにリンクすることもできます。

ポートは手動でリンクすることも自動的にリンクすることもできます。

- ポートの手動リンク。1つのポートまたは複数のポートを手動でリンクできます。入力オブジェクトのポートから出力オブジェクトのポートにドラッグします。
- ポートの自動リンク。ポートを自動的にリンクする場合は、位置または名前によってリンクを作成できます。

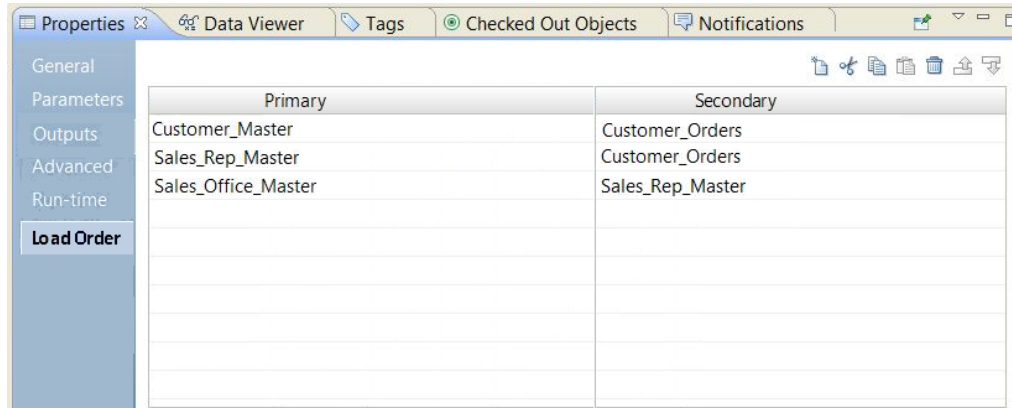
ポートのリンクの詳細については、『Developer トランスフォーメーションガイド』を参照してください。

## ターゲットロード順の制約の作成

ターゲットロード順の制約では、データ統合サービスが行を同じマッピング内のターゲットテーブルにロードする順序を制限します。マッピングの【ロード順序】タブで、ターゲットロード順序の制約を作成します。

1. エディタの内部をクリックします。  
マッピングの【プロパティ】タブが下のウィンドウに表示されます。

2. 【ロード順序】タブをクリックします。  
次の図は、【ロード順序】タブを示しています。



この画像には3つの制約が示されています。各制約には、プライマリターゲットおよびセカンダリターゲットが含まれています。これらの制約は、データ統合サービスが Customer\_Orders ターゲットの前に Customer\_Master ターゲットをロードする必要があることを指定しています。Sales\_Rep\_Master は Customer\_Orders ターゲットの前にロードする必要があります。Sales\_Office\_Master は Sales\_Rep\_Master の前にロードする必要があります。

3. 制約を入力するには、【新規】ボタンをクリックします。  
Developer tool によって、制約の行が作成されます。
4. 【プライマリ】フィールドをクリックします。  
マッピングのターゲットインスタンスのリストが表示されます。
5. 最初にロードするターゲットインスタンスを選択します。
6. 【セカンダリ】フィールドで、2 番目にロードするターゲットインスタンスを選択します。

制約は任意の順序で入力できます。次に示す制約のペアを入力しても、前の画像と同じ制約を指定することができます。

プライマリ	セカンダリ
Sales_Office_Master	Sales_Rep_Master
Sales_Rep_Master	Customer_Master
Customer_Master	Customer_Orders

7. 必要な数だけ制約を入力します。



## マッピングの検証

マッピングを検証して、Data Integration Service がマッピング全体を読み取りおよび処理できることを確認します。

1. マッピングを右クリックして、**【検証】** を選択します。  
エラーは**【検証ログ】** ビューに表示されます。
2. エラーを修正し、マッピングを再度検証します。

## マッピングの実行

マッピングを実行し、出力をソースからターゲットに移動してデータを変換します。

ドメインに複数のデータ統合サービスが含まれていて、デフォルトのサービスをまだ選択していない場合、Developer tool では、データのプレビュー時またはマッピングの実行時にいずれかのサービスを選択するように求められます。

- ▶ エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックして、**【マッピングの実行】** をクリックします。

データ統合サービスによって、マッピングが実行され、出力がターゲットに書き込まれます。  
オペレーティングシステムのプロファイルを使用するようにデータ統合サービスが設定されている場合、そのオペレーティングシステムのプロファイルを使用してマッピングを実行します。

## 第 2 章

# マプレット

この章では、以下の項目について説明します。

- [マプレットの概要, 42 ページ](#)
- [マプレットのタイプ, 43 ページ](#)
- [マプレットの入力と出力, 43 ページ](#)
- [生成されたマプレット, 44 ページ](#)
- [ルール仕様およびマプレット, 46 ページ](#)
- [マプレットの作成, 48 ページ](#)
- [マプレット検証, 48 ページ](#)

## マプレットの概要

マプレットは、複数のマッピングで使える一連のトランスフォーメーションを含む再利用可能オブジェクトです。マプレットはマッピングで使用するか、またはルールとして検証します。

マプレットのトランスフォーメーションには、再利用できるものと再利用できないものがあります。シーケンスジェネレータトランスフォーメーションをマプレットに追加する場合、それは再利用可能である必要があります。

マッピング内でマプレットを使用する際には、そのインスタンスを用います。マプレットに施した変更はそのインスタンスすべてに引き継がれます。

マプレットには、他のマプレットを含めることができます。また、1つのマッピングまたはマプレットで、マプレットを複数回使用することもできます。マプレットの循環ネストを行うことはできません。例えば、マプレット A にマプレット B が含まれている場合、マプレット B にマプレット A を含めることはできません。

マプレットは手動で作成できます。マッピングまたはマプレット内のセグメントからマプレットを生成することもできます。

# マプレットのタイプ

マプレットのタイプは、マプレットの入力と出力によって決定されます。

以下のタイプのマプレットを作成または生成できます。

- ソース。マプレットには、入力としてデータソースが含まれ、出力として出力トランスフォーメーションが含まれます。
- ターゲット。マプレットには、入力として入力トランスフォーメーションが含まれ、出力としてデータソースが含まれます。
- 中間ストリーム。マプレットには、入力トランスフォーメーションと出力トランスフォーメーションが含まれます。入力または出力用のデータソースは含まれません。

## マプレットの入力と出力

マッピング内でマプレットを使用するためには、マプレットの入力と出力を設定する必要があります。

マプレットは、以下の入力と出力の要素で構成されます。

- マプレットの入力。データソースや入力トランスフォーメーション、またはその両方を使用して、マプレットにデータを渡すことができます。マプレットをルールとして検証する場合は、入力トランスフォーメーションを使用してマプレットにデータを渡す必要があります。入力トランスフォーメーションを使用する場合は、マッピング内のソースまたはアップストリームトランスフォーメーションに接続します。
- マプレットの出力。データソースや出力トランスフォーメーション、またはその両方を使用して、マプレットからデータを渡すことができます。マプレットをルールとして検証する場合は、出力トランスフォーメーションを使用してマプレットからデータを渡す必要があります。出力トランスフォーメーションを使用する場合は、マッピング内のターゲットまたはダウンストリームトランスフォーメーションに接続します。
- マプレットポート。マッピングエディタでマプレットポートを表示できます。マプレット入力ポートおよび出力ポートは、入力トランスフォーメーションと出力トランスフォーメーションから生成されます。データソースからは生成されません。

## マプレットの入力

マプレットは、データソースまたは入力トランスフォーメーションを入力元とすることができます。

1つのマプレット内に複数のパイプラインを作成することができます。複数のデータソースまたは入力トランスフォーメーションを使用します。また、データソースと入力トランスフォーメーションを組み合わせで使用することもできます。

1つ以上のデータソースを使用して、マプレットにソースデータを提供します。マッピング内でマプレットを使用する場合、マプレットはマッピングパイプラインにおける最初のオブジェクトとなり、入力ポートを含みません。

入力トランスフォーメーションを使用して、マッピングから入力を受け取ります。入力トランスフォーメーションは入力ポートを提供し、これによりマプレット内でデータを受け渡すことが可能になります。マプレット内の別のトランスフォーメーションに接続した入力トランスフォーメーションの各ポートがマプレットの入力ポートとなります。

マプレットへの入力を提供するために入力トランスフォーメーションを使用するときに、入力トランスフォーメーションの出力ポートにデフォルト値を設定できます。入力トランスフォーメーションの出力ポートのデフォルト値が、マプレットの入力ポートのデフォルト値になります。マッピングから入力トランスフォーメーション

ョンに NULL データが渡された場合、データ統合サービスは NULL 値をデフォルト値に置き換えます。マプレットの入力では、デフォルト値を使用します。

入力トランスフォーメーションはマプレット内の複数のトランスフォーメーションに接続することができます。入力トランスフォーメーションの 1 つのポートをマプレット内の複数のトランスフォーメーションに接続することもできます。

入力トランスフォーメーションは、1 つのアクティブソースからのデータを受け取ることができます。接続されていないポートはマッピングエディタに表示されません。

## マプレットの出力

ターゲットマプレットを作成するときは、データソースを出力として使用します。マプレット内の出力トランスフォーメーションは、マプレットを介してマッピングにデータを渡すために使用します。

1 つ以上のデータソースを使用して、マプレットにターゲットデータを提供します。マッピング内でマプレットを使用する場合、マプレットはマッピングパイプラインにおける最後のオブジェクトとなり、出力ポートを含みません。

出力トランスフォーメーションを使用して、マッピング内のダウストリームトランスフォーメーションまたはターゲットに出力を渡します。出力トランスフォーメーション内で接続されている各ポートは、マッピング内でマプレットの出力ポートとして表示されます。マプレット内の各出力トランスフォーメーションは、出力グループとして表示されます。出力グループは、データをマッピング内の複数のパイプラインに渡すことができます。

## 生成されたマプレット

マッピングまたはマプレットのセグメントからマプレットを生成できます。マッピングまたはマプレットに、再使用する予定の接続されたトランスフォーメーションフローが含まれている場合に、マプレットを生成する場合があります。

Developer tool は、生成プロセスの一部としてセグメントをマプレットとして検証します。検証エラーを回避するには、生成されたマプレットのルールおよびガイドラインを確認してください。

## 生成されたマプレットのルールおよびガイドライン

次の条件が当てはまる場合、マプレット生成が失敗します。

- 選択したトランスフォーメーションが連続していない。
- セグメントに読み取りトランスフォーメーションと書き込みトランスフォーメーションの両方が含まれている。ただし、セグメントに複数の読み取りトランスフォーメーションまたは複数の書き込みトランスフォーメーションが含まれていても問題ありません。
- セグメントに再利用不可能なシーケンスジェネレータートランスフォーメーション、入力トランスフォーメーション、出力トランスフォーメーション、または操作の設定を含むトランスフォーメーションが含まれている。
- 選択したセグメントにパイプラインブランチのすべてのトランスフォーメーションが含まれていない。
- セグメントの最初と最後のトランスフォーメーションに動的フィールドが含まれている。
- セグメントに最初のトランスフォーメーションへの入力ランタイムリンクまたは最後のトランスフォーメーションからの出力ランタイムリンクが含まれている。

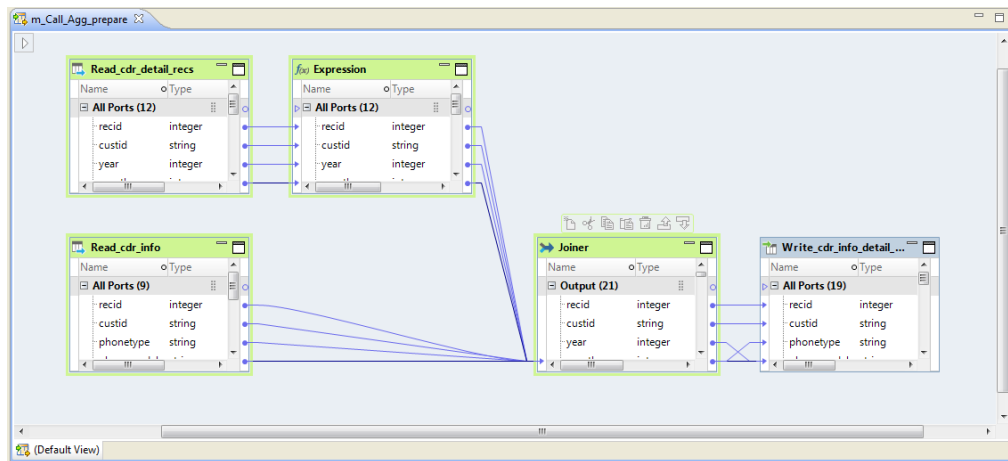
- セグメントが1つのパラメータ化された読み取りトランスフォーメーション、書き込みトランスフォーメーション、またはルックアップトランスフォーメーションで構成されている。

## マップレットの生成

接続されたトランスフォーメーションが含まれるセグメントからマップレットを生成します。セグメントに読み取りトランスフォーメーション、書き込みトランスフォーメーション、またはミッドストリームトランスフォーメーションが含まれる場合があります。

- マップレットに生成するセグメントが含まれるマッピングまたはマップレットを開きます。
- マップレットに組み込むトランスフォーメーションを選択します。
- 選択したトランスフォーメーションのいずれかを右クリックし、**【マップレットの抽出】**を選択します。

次の図に、4つのトランスフォーメーションが選択されたマッピングを示します。



生成プロセスでセグメントが検証され、検証エラーがあれば報告されます。

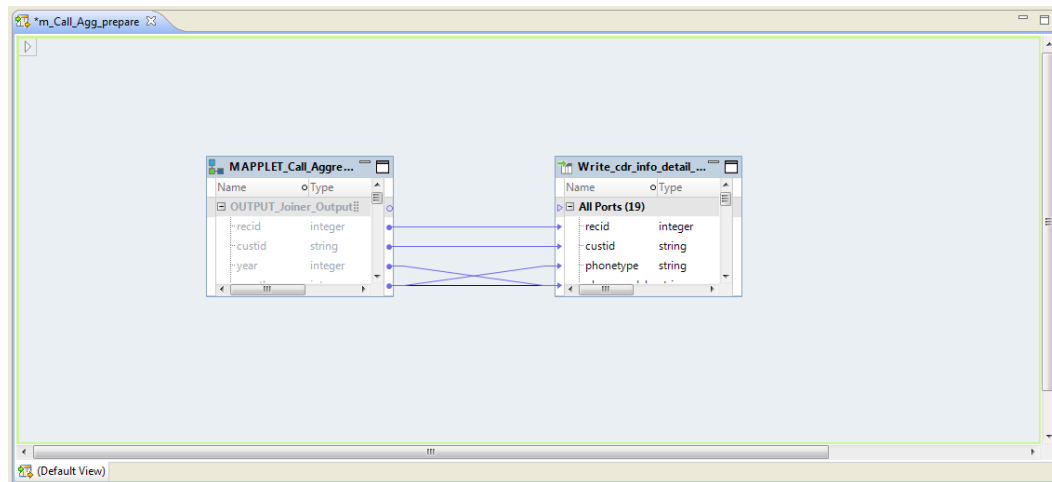
- マップレットを生成する **【マップレット】** ノードを参照します。

Developer tool は、デフォルトで現在のプロジェクト内の **【マップレット】** ノードにマップレットを生成します。

- 【完了】** をクリックします。

このマップレットは、元のマッピングまたはマップレットで選択されたトランスフォーメーションを置換します。このツールは、セグメントに読み取りトランスフォーメーション、書き込みトランスフォーメーション、またはミッドストリームトランスフォーメーションが含まれているかどうかに基づいて入力トランスフォーメーションまたは出力トランスフォーメーションをマップレットに追加します。

次の図に、選択したトランスフォーメーションがマップレットによって置換される様子を示します。



選択したトランスフォーメーションをマップレットで置換するには、変更されたマッピングまたはマップレットを明示的に保存する必要があります。マッピングまたはマップレットを元の状態に戻すには、【ファイル】>【元に戻す】を3回選択します。

## ルール仕様およびマップレット

ルール仕様は、ビジネスロジックを使用してトランスフォーメーション操作を記述するモデルリポジトリオブジェクトです。ユーザーは、Informatica Analyst でルール仕様を作成します。マッピングにマップレットを追加するのと同じ方法で、ルール仕様をマップに追加できます。

また、マップレットにルール仕様を追加し、Developer tool からルール仕様を Web サービスとしてデプロイすることもできます。

Analyst ツールユーザーは、ルール仕様から1つ以上のマップレットを生成できます。各マップレットには、ルール仕様ロジックを表すトランスフォーメーションが含まれています。ルール仕様または対応するマップレットのいずれかを含むマッピングを実行すると、同じ結果が得られます。

Developer tool でマップレットを作成するのと同じ方法で、ルール仕様からユーザーが生成するマップレットを編集できます。Developer tool でルール仕様を編集することはできません。マッピングでルール仕様を表すロジックを適用する場合は、マッピングにルール仕様を追加します。ルール仕様とは無関係にマップレットロジックを使用または更新する場合は、対応するマップレットをマッピングに追加します。

### ルール仕様のルールとガイドライン

- ルール仕様には、プライマリルールセットが含まれ、必要に応じてルールセットを追加できます。プライマリルールセットは、ルール仕様の完全なロジックを表します。追加のルールセットでは、個別のデータ分析操作を定義し、他のルールセットが読み取ることができる出力を提供します。

プライマリルールセットを表すマップレットは、ルール仕様と同じ名前になります。

- Developer tool でルール仕様の名前を変更すると、ユーザーがルール仕様を開いたときに、Analyst ツールに変更後の名前が表示されます。プライマリルールセットのマップレットの名前を変更しても、ルール仕様名は変わりません。

- Analyst ツールユーザーがルール仕様で入力を追加、削除、または編集すると、マッピング内の他のオブジェクトへのすべての入力リンクが壊れます。Analyst ツールユーザーがルール仕様で出力を追加、削除、または編集すると、マッピング内の他のオブジェクトへのすべての入力リンクが壊れます。リンクが壊れる編集としては、入力または出力の名前や精度やデータ型の変更などがあります。そのルール仕様を使用しているすべてのマッピングでリンクを更新してください。

Analyst ツールユーザーがルール仕様でビジネスロジックを更新しても、入力や出力を変更していなければ、入力リンクも出力リンクも壊れません。ユーザーが加えた変更は、ユーザーがルール仕様を保存するとマッピングで有効になります。

## ルール仕様のプロパティ

ルール仕様には、Developer tool で表示および編集できるプロパティがあります。それらのプロパティを使用すると、ルール仕様名の説明用メタデータを確認できます。また、それらのプロパティでは、ルール仕様がダウストリームマッピングオブジェクト用に生成する出力ポートの数を指定することもできます。

プロパティを表示するには、ルール仕様を含むマッピングを開き、[ルール仕様] アイコンを選択します。次に、そのマッピングで **【プロパティ】** タブを選択します。

このタブには、次のビューが表示されます。

### 全般

全般プロパティには、ルール仕様インスタンスの名前と説明が含まれています。

マッピングでルール仕様の名前または説明を更新すると、その変更は現在のマッピングにのみ適用されます。

### プロパティ

プロパティには、[全般] ビューに表示されるルール仕様名などがあります。また、Analyst ツールユーザーがルール仕様を設定した日付範囲も指定します。日付範囲は、ルール仕様が使用可能な期間を示します。

**注:** ルール仕様の子ルールごとに出力を許可するオプションを選択またはクリアできます。子ルールは、ルール仕様を設定されたルールです。このオプションを選択すると、Developer tool によって、マッピングのルール仕様を設定されたルールごとに出力ポートが追加されます。このオプションは、デフォルトでクリアされています。マッピング内のダウストリームオブジェクトに対してルールセット出力を使用できるようにするオプションを選択します。

### ポート

ポートプロパティには、ルール仕様インスタンスの入出力ポートが一覧表示されます。ポートプロパティには、各ポートの名前、データ型、精度、およびスケールが表示されます。必要に応じて、ポートに説明を追加できます。この説明は、現在のルール仕様インスタンスのポートに適用されます。

### ランタイムリンク

ランタイムリンクプロパティによって、ルール仕様ポートが動的マッピング内の他のオブジェクトに現在リンクされているかが決まります。

### 詳細

詳細プロパティには、トレースレベルの設定などがあります。トレースレベルによって、ルール仕様のログに記載される詳細の量が定義されます。[簡易]、[通常]、[詳細 - 初期化]、[詳細 - データ] から選択できます。デフォルト値は [通常] です。

# マプレットの作成

マプレットを作成して、複数のマッピングで使用できる一連のトランスフォーメーションを含む再利用可能オブジェクトを定義します。

1. **【Object Explorer】** ビューで、プロジェクトまたはフォルダを選択します。
2. **【ファイル】 > 【新規】 > 【マプレット】** をクリックします。
3. マプレット名を入力します。
4. **【完了】** をクリックします。  
空のマプレットがエディタに表示されます。
5. マプレットの入力、出力、およびトランスフォーメーションを追加します。

## マプレット検証

マッピングに追加する前にマプレットを検証できます。プロファイルでルールと使用するマプレットを検証することもできます。

### マプレットの検証

マプレットをマッピングに追加する前に検証します。マプレットをルールとして検証し、プロファイルに含めることもできます。

1. マプレットエディタを右クリックします。
2. **【検証】 > 【マプレット】** または **【検証】 > 【ルール】** を選択します。  
検証ログに発生したエラーが表示されます。

### ルール検証としてのマプレット

ルールは、プロファイルの実行時にソースデータに適用される条件を定義するビジネスロジックです。これは、プロファイルで使用する中間ストリームマプレットです。プロファイルでルールとして使用するマプレットを検証できます。

ルールは、以下の要件を満たす必要があります。

- ルールには入力トランスフォーメーションと出力トランスフォーメーションを含める必要があります。ルールでデータソースを使用することはできません。
- ルールには式トランスフォーメーション、ルックアップトランスフォーメーション、およびパッシブなデータ品質トランスフォーメーションを含めることができます。ルールには他のタイプのトランスフォーメーションを含めることはできません。例えば、一致トランスフォーメーションはアクティブなトランスフォーメーションなので、ルールに含めることはできません。
- ルールでは、入力グループ間のカーディナリティを指定しません。

**注:** ルールの機能はプロファイリングに限定されません。Analyst ツールでルールとして検証したマプレットをプロファイルに追加できます。例えば、郵便アドレスを検証するように設定されているルールを選択してプロファイルに追加することで、郵便アドレスのデータ品質を評価できます。



## 第 3 章

# マッピングパラメータ

この章では、以下の項目について説明します。

- [マッピングパラメータの概要, 49 ページ](#)
- [システムパラメータ, 50 ページ](#)
- [ユーザー定義のパラメータ, 51 ページ](#)
- [ユーザー定義のパラメータを作成する場所, 53 ページ](#)
- [マッピングのパラメータ, 53 ページ](#)
- [マップレットのパラメータ, 55 ページ](#)
- [論理データオブジェクトのパラメータ, 57 ページ](#)
- [仮想テーブルマッピングのパラメータ, 58 ページ](#)
- [パラメータセット, 59 ページ](#)
- [パラメータファイル, 60 ページ](#)
- [パラメータ階層, 65 ページ](#)
- [パラメータの設定方法, 68 ページ](#)
- [パラメータを使用してマッピングを実行する方法, 78 ページ](#)

## マッピングパラメータの概要

マッピングパラメータとは、マッピングを実行するたびに変更できる定数値です。異なる値でマッピングを再実行できるように、パラメータを作成します。パラメータを使用して、接続、ファイルディレクトリ、式のコンポーネント、ポートリスト、ポートリンク、およびタスクプロパティの値を変更します。

システムパラメータまたはユーザー定義のパラメータを設定できます。

### システムパラメータ。

データ統合サービスの組み込みパラメータ。システムパラメータは、データ統合サービスがログファイル、キャッシュファイル、拒否ファイル、ソースファイル、ターゲットファイル、一時ファイルを格納するディレクトリを定義します。管理者は、データ統合サービスのシステムパラメータのデフォルト値を Administrator ツールで定義します。

### ユーザー定義のパラメータ。

トランスフォーメーション、論理データオブジェクト、マッピング、およびワークフロー内で定義するパラメータ。異なる接続、フラットファイル、キャッシュファイル、一時ファイル、式、ポート、または参照テーブルの値でマッピングを再実行できるように、ユーザー定義のパラメータを作成します。

パラメータを使用して、生成されたどのポートを、実行時に動的マッピングで使用するかを決定することができます。実行時にリンクするポートを示すように、パラメータを設定できます。読み取り、書き込み、またはルックアップトランスフォーメーションのデータオブジェクトを変更するように、パラメータを割り当てることができます。

パラメータセットまたはパラメータファイルをマッピングに割り当てることで、パラメータ値を上書きできます。パラメータセットは、マッピングパラメータの値が含まれるリポジトリオブジェクトです。パラメータファイルは、パラメータ値が含まれる XML ファイルです。パラメータセットまたはパラメータファイルを使用してマッピングを実行する場合、データ統合サービスはパラメータセットまたはパラメータファイルで定義されたパラメータ値を使用します。この値は、トランスフォーメーション、マッピング、マップレット、またはワークフローで設定されたデフォルトのパラメータ値を上書きします。

ワークフローパラメータの詳細については、『*Informatica Developer ワークフローガイド*』を参照してください。

## 関連項目：

- [「動的マッピングのパラメータ」 \(ページ 128\)](#)

# システムパラメータ

システムパラメータは、データ統合サービスがキャッシュファイル、拒否ファイル、ソースファイル、ターゲットファイル、ログファイル、および一時ファイルを格納するディレクトリを定義する定数値です。

データ統合サービスの実行オプション内のいくつかのシステムパラメータの値を定義します。管理者は、Administrator ツールで値を更新できます。データ統合サービスは、他のシステムパラメータの値を実行時に決定します。パラメータファイルまたはパラメータセットのシステムパラメータ値を上書きすることはできません。

システムパラメータは作成できません。Developer tool には定義済みのシステムパラメータのリストが用意されており、マッピングのデータオブジェクトまたはトランスフォーメーションに割り当てることができます。例えば、アグリゲータトランスフォーメーションを作成すると、キャッシュディレクトリのシステムパラメータがデフォルト値として、Informatica Administrator のキャッシュディレクトリフィールドに割り当てられます。異なるキャッシュディレクトリの場所を使用する場合は、ユーザー定義のパラメータを作成し、デフォルトのパラメータ値を設定します。

Analyst ツールには、システムパラメータのファイルパスが`$$[Parameter Name]/[Path]`という形式で表示されます。例えば、「`$$$SourceDir/ff_dept.txt`」のようになります。

以下の表に、システムパラメータを示します。

システムパラメータ	タイプ	説明
CacheDir	String	インデックスファイルとデータキャッシュファイルのデフォルトディレクトリ。
LogDir	String	マッピングタスクのログファイルのデフォルトディレクトリ。
RejectDir	String	拒否ファイルのデフォルトディレクトリ。
SourceDir	String	ソースファイルのデフォルトディレクトリ。
TargetDir	String	ターゲットファイルのデフォルトディレクトリ。

システムパラメータ	タイプ	説明
TempDir	String	一時ファイルのデフォルトディレクトリ。
ApplicationName	String	アプリケーションの名前
ExecutionEnvironment	String	Hadoop またはネイティブ環境。
MappingName	String	実行中のマッピングの名前。
MappingRunStartTime	Date/Time	実行中のマッピングの開始時刻。
ServiceName	String	データ統合サービス名。
UserName	String	マッピングを実行しているユーザーの名前。

## ユーザー定義のパラメータ

ユーザー定義のパラメータは、マッピングの実行間で変更できる定数値を表します。

例えば、顧客の注文を処理するマッピングを作成するとします。このマッピングは、1つの国の顧客データを含むリレーショナルテーブルから顧客情報を読み取ります。米国、カナダ、メキシコの顧客に対してマッピングを使用するとします。この場合、顧客テーブルへの接続を表すユーザー定義のパラメータを作成します。米国の顧客テーブル、カナダの顧客テーブル、およびメキシコの顧客テーブルへの接続名を設定する3つのパラメータセットを作成します。マッピングのそれぞれの実行に異なるパラメータセットを使用して実行します。

以下の型パラメータを作成できます。

### 接続パラメータ

Informatica の接続名。

### 日付/時刻パラメータ

日付。

### 式

結合条件、フィルタ式、またはルックアップ条件を定義する式。

### 入力リンクセット

【ランタイムリンク】ダイアログボックス内の、リンクするポートのセット。

### 数値パラメータ

Integer、Bigint、Decimal、Double 型のパラメータ。

### ポート

1つのポートの名前。ランクトランスフォーメーションのランクポートで、ポートパラメータを使用できます。

### ポートリスト

グループを含めるポートのリスト。例えば、ポートリストパラメータをアグリゲータトランスフォーメーションやランクトランスフォーメーション内で使用できます。

## リソース

リレーショナルデータオブジェクトのテーブル、ビュー、またはシノニムの名前。リソース名をパラメータ化すると、データ統合サービスはそのパラメータ値をランタイムクエリ内で使用してオブジェクトを取得します。

## ソートリスト

ソートトランスフォーメーションを使用してソートするポートのリスト。リストには、ポート名と、昇順または降順のソートシーケンスを示すインジケータが含まれます。

## ソートキーリスト

ウィンドウイング用に設定された式トランスフォーメーションの順序キーを使用してソートするポートのリスト。このリストには、ポート名と、昇順または降順のソートシーケンスを示すインジケータが含まれます。

## String

文字列パラメータは、フラットファイル名、ディレクトリ、テーブル名、またはランタイムプロパティを表します。文字列パラメータは、32768 文字以下の精度で定義します。

パラメータを作成する場合、パラメータ名の先頭にドル記号 (\$) を使用することはできません。

パラメータを使用してプロパティ値を設定する場合、プロパティに適切なパラメータのタイプを使用する必要があります。例えば、ターゲットファイル名に接続タイプパラメータは使用できません。数値式でパラメータを使用する場合は、数値タイプのパラメータを使用する必要があります。

リレーショナルデータオブジェクトでは、SQL オーバーライド、フィルタ条件、または結合条件でドル記号 (\$) をエスケープする必要はありません。データ統合サービスは、SQL 文におけるドル記号で始まるフィールドをパラメータとして扱います。

パラメータには一連の値を含めることはできません。パラメータで一連の値を指定すると、データ統合サービスはパラメータ値を単一の文字列値として処理します。

例えば、パラメータ \$IndexParameter1 (value 2) と \$IndexParameter2 (value1, value2, value3) があるとします。これらのパラメータを、次のように式 INDEXOF に指定します。

```
INDEXOF($IndexParameter1,'value1','value2','value3')
```

データ統合サービスは、value 2 ではなく value 0 を返します。

## 日付/時刻パラメータ

日付パラメータを作成して式内で使用できます。

日付パラメータは以下のいずれかの形式で定義する必要があります。

```
MM/DD/RR
MM/DD/YYYY
MM/DD/YYYY HH24:MI
MM/DD/RR HH24:MI
MM/DD/RR HH24:MI:SS
YYYY/MM/DD HH24:MI:SS
MM/DD/RR HH24:MI:SS.NS
MM/DD/YYYY HH24:MI:SS.NS
```

# ユーザー定義のパラメータを作成する場所

ユーザー定義のパラメータは、フラットファイルデータオブジェクト、トランスフォーメーション、カスタムデータオブジェクト、マップレット、マッピング、およびワークフロー内に作成できます。パラメータを作成したら、条件、式、接続、ディレクトリ、ファイル名など、各種フィールドにパラメータを割り当てることができます。

トランスフォーメーション、論理データオブジェクト、マップレット、マッピング、またはワークフローに対してパラメータを作成した場合、パラメータはそのオブジェクトに適用されます。例えば、読み取りトランスフォーメーション内にパラメータを作成し、トランスフォーメーションでデータソースから読み取るカラムをパラメータ化することができます。または、ランタイム環境を定義するマッピングパラメータを作成できます。

次のオブジェクトのパラメータを作成できます。

- トランスフォーメーションまたはデータオブジェクト
- 論理データオブジェクト
- マップレット
- マッピング
- ワークフロー

パラメータセットまたはパラメータファイルでパラメータ値を設定することにより、ワークフローパラメータの値とマッピングパラメータの値を実行時に設定できます。

パラメータを作成すると同時に、フィールドとプロパティにパラメータを割り当てることができます。パラメータをフィールドに割り当てる際に、使用するパラメータを作成できます。以前作成したパラメータを参照することもできます。

**注:** **[パラメータ]** タブでパラメータを作成する場合、パラメータ名の先頭にドル記号 (\$) を含めないでください。

ユーザー定義のパラメータは、トランスフォーメーションまたはデータオブジェクトの **[パラメータ]** タブで維持します。マッピング、マップレット、ワークフロー、論理データオブジェクトにも **[パラメータ]** タブがあります。**[パラメータ]** タブでは、パラメータの追加、変更、削除ができます。

また、**[アウトライン]** ビューから直接パラメータにアクセスすることもでき、ここにはパラメータを使用、定義、およびバインドする場所が示されます。**[アウトライン]** ビューでパラメータをクリックすると、パラメータプロパティが **[パラメータ]** タブに表示されます。

## マッピングのパラメータ

マッピングプロパティにパラメータを割り当てたり、マッピングオブジェクトのパラメータを作成したりできます。

マッピングパラメータを定義する場合、マッピングパラメータをトランスフォーメーションパラメータにバインドできます。マッピングパラメータの値が、トランスフォーメーション内のデフォルトのパラメータ値を上書きします。

マッピングパラメータをトランスフォーメーションパラメータにバインドする場合、それらのパラメータのタイプが同じである必要があります。マッピングパラメータ名は、トランスフォーメーションパラメータ名と同じである必要はありません。

パラメータセットまたはパラメータファイルを使用して、マッピングパラメータの値を実行時に設定することができます。パラメータセットまたはパラメータファイルを使用して、再利用可能なトランスフォーメーション

ン、再利用可能なマップレット、または再利用可能なデータソースパラメータ値を設定することはできません。実行時に再利用可能なオブジェクトでパラメータ値を変更するには、マッピングパラメータとしてパラメータ値を公開します。パラメータセットまたはパラメータファイルで公開された値のマッピングパラメータを設定します。

以下のいずれかの方法を使用して、マッピングパラメータを定義します。

#### マッピングの【プロパティ】ビューの【パラメータ】タブで、マッピングパラメータを定義する

マッピングの【パラメータ】タブで、各パラメータ名、パラメータ属性、およびデフォルト値を手動で入力できます。マッピングにトランスフォーメーションを追加するときには常に、これらのパラメータをトランスフォーメーションパラメータにバインドできます。マッピングの【パラメータ】タブで、マッピングパラメータを更新できます。また、【アウトライン】ビューではパラメータを表示してアクセスすることもできます。

#### 再利用可能なトランスフォーメーションパラメータからマッピングパラメータを追加する

マッピングにトランスフォーメーションを追加したら、マッピングパラメータをトランスフォーメーションの【パラメータ】タブから直接作成できます。トランスフォーメーションパラメータからマッピングパラメータを作成するには、トランスフォーメーションパラメータをマッピングパラメータとして公開します。Developer tool は、トランスフォーメーションパラメータと同じ名前とタイプを持つマッピングパラメータを作成します。

#### 再利用不可能なトランスフォーメーションにパラメータを追加する

マッピング内でトランスフォーメーションを作成した場合、そのトランスフォーメーションは再利用不可能なトランスフォーメーションです。トランスフォーメーションのいずれかのプロパティをパラメータ化する場合、トランスフォーメーションパラメータではなくマッピングパラメータを作成します。マッピングパラメータは、トランスフォーメーションパラメータにバインドされます。

## パラメータインスタンス値

パラメータ付きの再利用可能な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、トランスフォーメーションの各パラメータのインスタンス値を設定できます。

このインスタンス値は、特定のマッピングのパラメータ値です。デフォルト値、特定の値、またはマッピングパラメータの値にインスタンス値を設定できます。

マッピングパラメータまたはマップレットパラメータは、トランスフォーメーションパラメータのデフォルト値を上書きできます。マッピングパラメータまたはマップレットパラメータを選択し、そのパラメータをトランスフォーメーションパラメータにバインドします。

【プロパティ】ビューの、トランスフォーメーションの【パラメータ】タブで、インスタンス値を設定します。

【インスタンス値】に対して、次のいずれかのオプションを選択します。

#### マッピングパラメータとして公開

トランスフォーメーションパラメータと同じ属性を持つマッピングパラメータを作成し、同じ手順で、マッピングパラメータをトランスフォーメーションパラメータにバインドします。【マッピングパラメータとして公開】ボタンを2度目にクリックしたときに、トランスフォーメーションパラメータがすでにマッピングパラメータにバインドされている場合、Developer tool ではマッピングパラメータを変更しません。

#### パラメータ

トランスフォーメーションパラメータにバインドするマッピングパラメータを参照し、選択します。マッピングパラメータを作成してトランスフォーメーションパラメータにバインドすることもできます。マッピングパラメータを作成してバインドする場合は、【マッピングパラメータとして公開】オプションと同じタスクを実行します。ただし、マッピングパラメータを手動で作成する場合、トランスフォーメーションパラメータとは別の名前を設定できます。

### デフォルトを使用

トランスフォーメーションパラメータのデフォルト値を使用します。トランスフォーメーションパラメータへのマッピングパラメータのバインドをスキップします。

### 値

マッピングで使用するデフォルトのパラメータ値を入力します。トランスフォーメーションパラメータへのマッピングパラメータのバインドをスキップします。

## マップレットのパラメータ

マップレットパラメータは、データオブジェクトのパラメータにバインドしたり、マップレット内にあるトランスフォーメーションのパラメータにバインドしたりできます。

マップレットパラメータを定義する場合、マップレットパラメータを特定のトランスフォーメーションパラメータにバインドできます。マップレットパラメータの値が、トランスフォーメーション内のデフォルトのパラメータ値を上書きします。マップレットパラメータをトランスフォーメーションパラメータにバインドする場合、それらのパラメータのタイプが同じである必要があります。マップレットパラメータ名は、トランスフォーメーションパラメータ名と同じである必要はありません。マップレットパラメータを複数のトランスフォーメーションパラメータにバインドすることができます。

以下のいずれかの方法を使用して、マップレットパラメータを定義します。

### マップレットの【プロパティ】ビューの【パラメータ】タブで、マップレットパラメータを定義する

マップレットの【パラメータ】タブで、各パラメータ名、パラメータ属性、およびデフォルト値を手動で入力できます。パラメータを定義した後、[アウトライン] ビューでそのパラメータを表示してアクセスできます。

### トランスフォーメーションパラメータからマップレットパラメータを追加する

マップレットにトランスフォーメーションを追加したら、マップレットパラメータをトランスフォーメーションの【パラメータ】タブから直接作成できます。

## マップレットのパラメータインスタンス値

トランスフォーメーションパラメータ付きの再利用可能なトランスフォーメーションをマップレットに追加する場合、各パラメータのインスタンス値を設定できます。パラメータのインスタンス値は、特定のマップレットのパラメータ値です。

マップレットにトランスフォーメーションを追加したら、トランスフォーメーションの【パラメータ】タブでインスタンス値を設定します。

【インスタンス値】に対して、次のいずれかのオプションを選択します。

### マップレットパラメータとして公開

トランスフォーメーションパラメータと同じ属性のマップレットパラメータを作成します。同じ手順で、マッピングパラメータをトランスフォーメーションパラメータにバインドします。

### パラメータ

トランスフォーメーションパラメータにマップレットパラメータをバインドします。トランスフォーメーションパラメータにバインドするマップレットパラメータを参照し、選択することができます。マップレットパラメータを作成し、そのパラメータをトランスフォーメーションパラメータにバインドすることもできます。マップレットパラメータを作成してバインドする場合は、【マップレットパラメータとして公



**開]** オプションと同じタスクを実行します。ただし、マップレットパラメータを手動で作成する場合、トランスフォーメーションパラメータとは別の名前と別のデフォルト値を設定できます。

#### デフォルトを使用

トランスフォーメーションパラメータのデフォルト値を使用します。トランスフォーメーションパラメータへのマップレットパラメータのバインドをスキップします。

#### 値

マップレットで使用する別のデフォルトのパラメータ値を入力します。トランスフォーメーションパラメータへのマップレットパラメータのバインドをスキップします。

## マッピングのマップレットパラメータ

マップレットパラメータ付きのマップレットをマッピングに追加する場合、マップレットパラメータのインスタンス値を設定できます。マップレットパラメータのインスタンス値は、特定のマッピングのパラメータ値です。

**[プロパティ]** ビューの、マップレットの **[パラメータ]** タブで、インスタンス値を設定します。

**[インスタンス値]** に対して、次のいずれかのオプションを選択します。

#### マッピングパラメータとして公開

マップレットパラメータと同じ属性のマッピングパラメータを作成します。同じ手順で、マッピングパラメータをマップレットパラメータにバインドします。

#### パラメータ

マップレットパラメータにマッピングパラメータをバインドします。マップレットパラメータにバインドするマッピングパラメータを参照し、選択することができます。マッピングパラメータを作成してマップレットパラメータにバインドすることもできます。マッピングパラメータを作成してバインドする場合は、**[マッピングパラメータとして公開]** オプションと同じタスクを実行します。ただし、マッピングパラメータを手動で作成する場合、マップレットパラメータとは別の名前とデフォルト値でマッピングパラメータを設定できます。

#### デフォルトを使用

マップレットパラメータのデフォルト値を使用します。マップレットパラメータへのマッピングパラメータのバインドをスキップします。

#### 値

マッピングで使用するデフォルトのパラメータ値を入力します。マップレットパラメータへのマッピングパラメータのバインドをスキップします。

## マップレットのパラメータの例

マップレットパラメータを定義し、マッピングパラメータでマップレットパラメータを上書きできます。

顧客テーブルから顧客データを返す SQL トランスフォーメーションを定義することができます。SQL トランスフォーメーションをマップレットに追加し、ランタイム接続をパラメータ化します。

次に、異なるデータベースから顧客データを取得するマッピングにマップレットを追加します。マップレットパラメータのデフォルト接続をオーバーライドするように、各マッピングのマッピングパラメータを定義します。



次の表に、マップレットとマッピングに対して作成できる接続パラメータのリストを示します。

オブジェクト名	オブジェクトタイプ	パラメータ名	パラメータのデフォルト値
mp_Get_Customer	マップレット	mp_cust_connection	Oracle_Default
m_billing_mapping	マッピング	m_acctg_connection	Oracle_AcctDB
m_order_fill_mapping	マッピング	m_shipping_connection	Oracle_Warehouse
m_cust_main_mapping	マッピング	m_master_connection	Oracle_Cust_Mast

mp\_Get\_Customer のマップレットには、mp\_cust\_connection という接続パラメータがあります。このパラメータのデフォルトの接続名は Oracle\_Default です。この接続では、例えばテストデータベースを参照します。

各マッピングに、mp\_cust\_connection パラメータをオーバーライドする接続パラメータがあります。各マッピングは、会計データベース、倉庫データベース、または顧客マスタデータベースに接続されます。

各マッピングパラメータをマップレットパラメータにバインドし、デフォルト値をオーバーライドする必要があります。マッピングパラメータの値を実行時に変更するには、パラメータセットまたはパラメータファイルを設定できます。

## 論理データオブジェクトのパラメータ

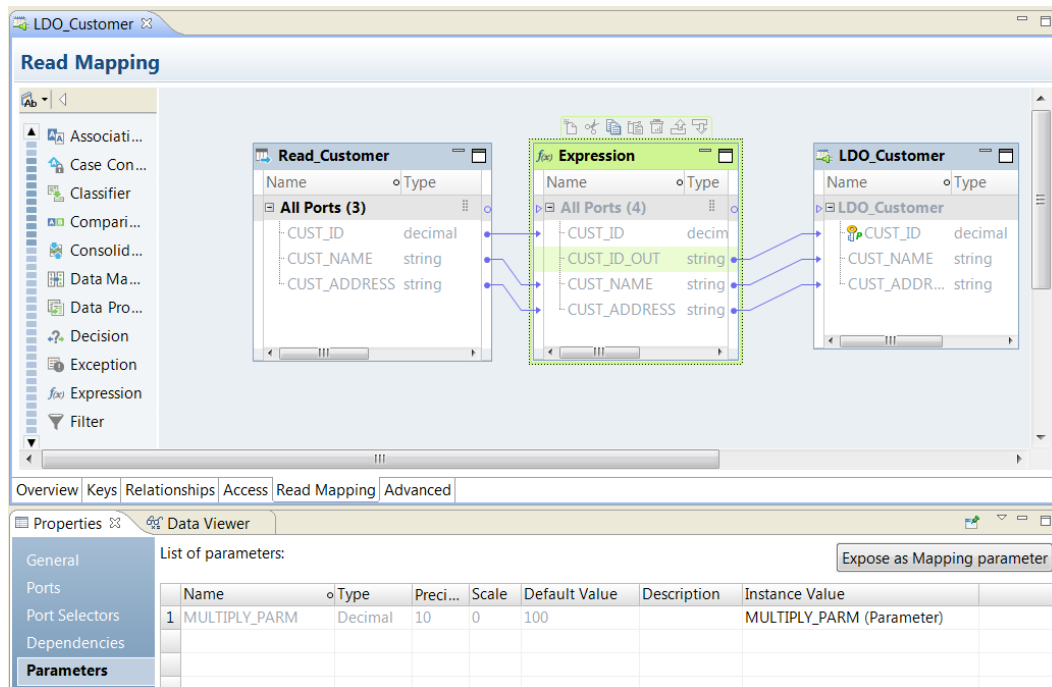
論理データオブジェクトにパラメータを含めることができます。パラメータは、トランスフォーメーションや読み取りおよび書き込みマッピングで使用できます。

論理データオブジェクトは、読み取りマッピングと書き込みマッピングを持つことができます。読み取りまたは書き込みマッピングには、パラメータを使用するトランスフォーメーションを含めることができます。再利用可能なトランスフォーメーションパラメータを読み取りまたは書き込みマッピングのパラメータにバインドできます。

例えば、式トランスフォーメーションが含まれる読み取りマッピングを持っている論理データオブジェクトがあります。式トランスフォーメーションには、式内の decimal 値を定義するパラメータがあります。デフォルト値は 100 です。

式トランスフォーメーションを読み取りマッピングに追加する場合、異なるパラメータ値を使用することが必要になる場合があります。読み取りマッピングレベルでパラメータを作成し、トランスフォーメーションパラメータをオーバーライドすることができます。[マッピングパラメータとして公開] をクリックし、重複するパラメータを読み取りマッピングで作成します。Developer tool は、重複するパラメータをトランスフォーメーションパラメータにバインドします。

次の図は、読み取りマッピング内の式トランスフォーメーションの [パラメータ] タブを示しています。



重複するパラメータを表示するには、[アウトライン] ビューで論理データオブジェクトを選択します。読み取りマッピングレベルでパラメータのデフォルト値を変更できます。

論理データオブジェクトをマップレットまたはマッピングに追加する場合、読み取りマッピングパラメータをオーバーライドできます。マップレットまたはマッピングで、重複するパラメータを作成します。重複するパラメータのデフォルト値を変更します。

## 仮想テーブルマッピングのパラメータ

仮想テーブルマッピングは、SQL データサービスのソースと仮想テーブル間のデータフローを定義します。仮想テーブルマッピングにはパラメータを含めることができますが、パラメータファイルまたはパラメータセットを使用してパラメータのデフォルト値をオーバーライドすることはできません。

仮想テーブルマッピングには、パラメータが含まれる再利用可能なトランスフォーメーションまたはマップレットを含めることができます。マッピングパラメータを、仮想テーブルマッピングのトランスフォーメーションパラメータまたはマップレットパラメータにバインドできます。

ただし、仮想テーブルマッピングにパラメータが含まれる場合、データ統合サービスは、マッピングレベルのデフォルトのパラメータ値を適用します。データ統合サービスは、パラメータファイルまたはパラメータセットから仮想テーブルマッピングのパラメータに値をバインドできません。

仮想テーブルマッピングに接続された、パラメータ化されたソースを使用できます。マッピングはデフォルトのパラメータ値を使用します。

# パラメータセット

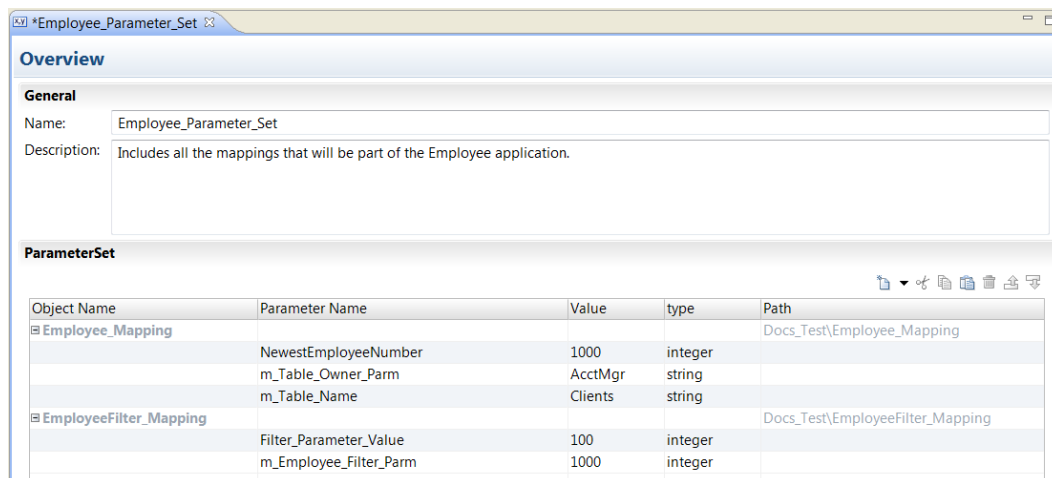
パラメータセットはモデルリポジトリ内のオブジェクトであり、マッピングおよびワークフローの実行に使用する一連のパラメータとその値を含みます。

パラメータセットを作成する場合、パラメータを使用するマッピングまたはワークフローを選択します。マッピングまたはワークフローを選択後、パラメータセットのパラメータを手動で入力するか、マッピングまたはワークフローのリポジトリにすでにあるパラメータを選択できます。

さまざまな状況でパラメータセットを使用できます。例えば、テスト環境でワークフローを実行する場合に特定のパラメータセットを使用することがあります。

マッピング、マッピングタスク、またはワークフローでパラメータセットを使用します。アプリケーションをデプロイする場合、アプリケーションに 1 つ以上のパラメータセットを追加できます。複数のアプリケーションに 1 つのパラメータセットを追加し、デプロイすることができます。ワークフローまたはマッピングでパラメータセットを使用するには、ワークフローまたはマッピングのデプロイ時にアプリケーションにパラメータセットを追加する必要があります。

次の画像は、2 つのマッピングのパラメータを含むパラメータセットを示します。



Object Name	Parameter Name	Value	type	Path
Employee_Mapping	NewestEmployeeNumber	1000	integer	Docs_Test\Employee_Mapping
	m_Table_Owner_Parm	AcctMgr	string	
	m_Table_Name	Clients	string	
EmployeeFilter_Mapping	Filter_Parameter_Value	100	integer	Docs_Test\EmployeeFilter_Mapping
	m_Employee_Filter_Parm	1000	integer	

パラメータセットには次の情報が含まれます。

## オブジェクト名

パラメータの定義を含むマッピング、マップレット、またはワークフローの名前

## パラメータ名

マッピング、マップレット、またはワークフローのパラメータ名

## 値

ランタイムに使用するパラメータ値パラメータセットのパラメータ値は、マッピングまたはワークフローのパラメータ値をオーバーライドします。

## タイプ

パラメータのタイプ。パラメータタイプの例には、文字列、数値タイプ、接続、ポートリスト、ソートリスト、および日付/時刻パラメータが含まれます。

**注:** パラメータセットで指定したパラメータタイプが、マッピング、マッピングタスク、またはワークフロー内のパラメータタイプと一致している必要があります。パラメータタイプが一致しない場合、マッピング、マッピングタスク、またはワークフローはパラメータのデフォルト値を使用します。

パラメータセットを使用してマッピングパラメータ値を設定する場合、パラメータセットとマッピング間のリンクはマッピングが定義されたプロジェクトによって異なります。プロジェクト名が変更された場合は、リンクを再確立する必要があります。

リンクを再確立するには、パラメータセットを編集し、パラメータセットを使用するマッピングを再選択します。

例えば、パラメータセット ps\_1 を使用するマッピング m\_1 を含む、P1 という名前のプロジェクトがあるとします。プロジェクト名を P2 に変更しました。その後、パラメータセットを編集し、マッピング m\_1 を再選択する必要があります。

## パラメータファイル

パラメータファイルは、ユーザー定義パラメータおよびその割り当て値を列挙した XML ファイルです。パラメータファイルを使用すると、mapping を実行するたびにパラメータ値を変更することができます。

パラメータ値を使用して、マッピングのプロパティを定義します。データ統合サービスでこれらの値を適用するには、マッピングを実行するときにパラメータファイルを指定します。

パラメータファイルでマッピングパラメータおよびワークフローパラメータを定義できます。再利用可能オブジェクトパラメータを指定する場合は、再利用可能オブジェクトパラメータをマッピングパラメータとして公開します。パラメータファイルで、マッピングパラメータ値を指定します。

パラメータファイルでシステムパラメータの値を定義することはできません。

1 つのパラメータファイルで複数の mapping のパラメータを定義できます。複数のパラメータファイルを作成し、mapping を実行するたびに異なるファイルを使用することもできます。データ統合サービスは、mapping 実行の開始時にパラメータファイルを読み込み、パラメータを解決します。

Developer tool からパラメータファイルをエクスポートできます。マッピングまたはワークフローの【**パラメータ**】タブからファイルをエクスポートします。Developer tool によって、マッピングまたはワークフローのパラメータとデフォルトのパラメータ値を含むパラメータファイルが生成されます。パラメータファイル名を指定し、ファイルの保存場所を選択できます。

**注:** パラメータファイルの構造は、マッピングでもワークフローでも同じです。1 つのパラメータファイルでデプロイ済みのマッピングとデプロイ済みのワークフローのパラメータを定義できます。

コマンドラインから mapping で使用されるパラメータとデフォルト値を一覧表示することもできます。コマンドライン出力は、パラメータファイルのテンプレートとして使用できます。

Developer tool からパラメータファイルを使用してマッピングを実行するか、コマンドラインからマッピングを実行します。

## パラメータファイル構造

パラメータファイルは、少なくとも 1 つのパラメータとその割り当て値を含む XML ファイルです。

データ統合サービスはパラメータファイルで定義された階層を使用して、パラメータおよびその定義値を識別します。この階層によって、パラメータを使用したマッピングまたはワークフローが識別されます。

パラメータ値は、最上位の要素である project 要素または application 要素で定義します。project 要素では、プロジェクト内の特定の mapping を実行する場合に使用するパラメータ値を定義します。project 要素はまた、プロジェクト内のオブジェクトを使用する任意の mapping を実行する場合に使用するパラメータ値も定義します。

application 要素では、特定のデブロイ済みのアプリケーションで mapping を実行する場合に使用するパラメータ値を定義します。同じパラメータファイルで最上位の project 要素と application 要素の両方に同じパラメータが定義されている場合、application 要素で定義されているパラメータ値が優先されます。

データ統合サービスは、以下の順でパラメータ値を探します。

1. application 要素内に指定された値。
2. project 要素内に指定された値。
3. パラメータのデフォルト値。

パラメータファイルは、パラメータファイルの XML スキーマ定義（XSD）の構造に従う必要があります。パラメータファイルがスキーマ定義に従わない場合、マッピングの実行は失敗します。mapping

Developer tool をホストするマシンでは、パラメータファイル XML スキーマ定義は次のディレクトリにあります。

```
<Informatica Installation Directory>\clients\DeveloperClient\infacmd\plugins\ms\parameter_file_schema_1_0.xsd
```

Informatica サービスをホストするマシンでは、パラメータファイル XML スキーマ定義は次のディレクトリにあります。

```
<Informatica Installation Directory>\isp\bin\plugins\ms\parameter_file_schema_1_0.xsd
```

## project 要素

project 要素では、プロジェクト内の特定の mapping を実行する場合に使用するパラメータ値を定義します。project 要素はまた、プロジェクト内のオブジェクトを使用する任意の mapping を実行する場合に使用するパラメータ値も定義します。

project 要素では、パラメータを使用するオブジェクトを含むモデルリポジトリ内のプロジェクトを定義します。project 要素には、ワークフローまたはマッピングを含めることができます。project 要素にトランスフォーメーション、マップレット、またはデータオブジェクト要素を含めることはできません。

次の表に、project 要素に含めることができる要素を示します。

エレメント名	説明
folder	プロジェクト内のフォルダを定義します。folder 要素は、オブジェクトをプロジェクト内の複数のフォルダにまとめる場合に使用します。folder 要素には、mapping または workflow 要素、または別の folder 要素を含めることができます。
mapping	パラメータを使用するプロジェクト内のマッピングを定義します。mapping 要素には、マッピング、パラメータを受け付けるマッピング内の再利用不可能なトランスフォーメーション、マップレット、またはデータオブジェクトのパラメータ値を定義する 1 つ以上のパラメータ要素が含まれています。

最上位の project 要素でパラメータ値が定義されているパラメータファイルを使用してマッピングを実行すると、プロジェクト内で指定されたマッピングにそのパラメータ値が適用されます。

例えば、「MyMapping」というマッピングを実行するときに「MyMapping\_Param」によって定義されたパラメータ値が適用されるようにします。次の要素を使用して、パラメータファイル内のパラメータを定義できます。

```
<project name="MyProject">

  <!-- Apply this parameter value to mapping "MyMapping" in project "MyProject". -->
  <mapping name="MyMapping">
    <parameter name="MyMapping_Param">Param_value</parameter>
  </mapping>
```

</project>

## application 要素

application 要素は、project 要素の実行時の範囲を指定する要素です。application 要素では、デプロイ済みのアプリケーションと、データ統合サービスがデプロイ済みのアプリケーションでマッピングまたはワークフローに適用するパラメータ値を定義します。

パラメータファイルで application 要素を使用して、実行するアプリケーションに応じてデータ統合サービスがマッピングまたはワークフローに適用するさまざまなパラメータセットを定義できます。

application 要素には、project、folder、mapping、または workflow 要素を含めることができます。次の表に、application 要素に含めることができる要素を示します。

エレメント名	説明
プロジェクト	モデルリポジトリのプロジェクトを定義します。project 要素を使用して、プロジェクト内に存在するフォルダ、マッピングまたはワークフローを配置します。
folder	プロジェクト内のフォルダを定義します。folder 要素は、オブジェクトをプロジェクト内の複数のフォルダにまとめる場合に使用します。folder 要素には、mapping または workflow 要素、または別の folder 要素を含めることができます。
mapping	パラメータを使用するアプリケーション内のマッピングを定義します。mapping 要素には、マッピング、パラメータを受け付けるマッピング内の再利用不可能なトランスフォーメーション、マップレット、またはデータオブジェクトのパラメータ値を定義する 1 つ以上のパラメータ要素が含まれています。  異なる application 要素内で同じマッピングを指定して、実行するアプリケーションに応じて異なるパラメータを使用できます。

例えば、「MyApp」というデプロイ済みのアプリケーションで「MyMapping」というマッピングを実行するときにパラメータ値が適用されるようにするとします。他のアプリケーションでマッピングを実行する場合や、プロジェクト「MyProject」内の別のマッピングを実行する場合は、パラメータ値を使用しません。以下の要素内にパラメータを定義します。

```
<application name="MyApp">
  <mapping name="MyMapping">
    <project name="MyProject">
      <mapping name="MyMapping">
        <parameter name="MyMapping_Param">Param_value</parameter>
      </mapping>
    </project>
  </mapping>
</application>
```

## パラメータファイルに関するルールとガイドライン

パラメータファイルを作成するときは、特定のルールとガイドラインが適用されます。

パラメータファイルを作成する際のルールは次のとおりです。

- パラメータファイルでマッピングおよびワークフローパラメータを参照できます。再利用可能なトランスフォーメーション、マップレット、またはデータオブジェクトパラメータは参照できません。再利用可能オブジェクトパラメータを参照するには、再利用可能オブジェクトパラメータをマッピングパラメータとして公開します。パラメータファイルで、マッピングパラメータ値を指定します。

- アプリケーション要素には、アプリケーション実行時に指定したアプリケーションのみに適用されるマッピングまたはワークフローパラメータが含まれます。Developer tool からパラメータファイルを使用してマッピングを実行する場合、アプリケーションとしてマッピングをデプロイしないでください。また、アプリケーション要素を指定しないでください。プロジェクト要素内のマッピングを指定します。
- パラメータ値は空にできません。例えば、パラメータファイルに以下のエントリがある場合、データ統合サービスは mapping しません。  

```
<parameter name="Param1"> </parameter>
```
- 要素内のアーティファクト名は大文字と小文字が区別されません。したがって、データ統合サービスは  

```
<parameter name="SrcDir">
```

と  

```
<parameter name="Srcdir">
```

を同じアプリケーションとして解釈します。
- 参照テーブルを特定するパラメータは、リポジトリフォルダパスでフォルダ名を区切るのにフォワードスラッシュ (/) を使用する必要があります。

## パラメータファイルの例

以下に、mapping の実行に使用されるパラメータファイルの例を示します。

```
<?xml version="1.0"?>
<root description="Sample Parameter File"
  xmlns="http://www.informatica.com/Parameterization/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <!--
    The Data Integration Service uses this section only when you run mapping "Map1" or "Map2"
    in project "Project1" in deployed application "App1."

    This section assigns values to parameters created in mappings "Map1" and "Map2."
  -->
  <application name="App1">
    <mapping name="Map1">
      <project name="Project1">
        <mapping name="Map1">
          <parameter name="MAP1_PARAM1">MAP1_PARAM1_VAL</parameter>
          <parameter name="MAP1_PARAM2">MAP1_PARAM2_VAL</parameter>
        </mapping>
      </project>
    </mapping>
    <mapping name="Map2">
      <project name="Project1">
        <mapping name="Map2">
          <parameter name="MAP2_PARAM1">MAP2_PARAM1_VAL</parameter>
          <parameter name="MAP2_PARAM2">MAP2_PARAM2_VAL</parameter>
        </mapping>
      </project>
    </mapping>
  </application>

  <!--
    The Data Integration Service uses this section only when you run mapping "Map1" in
    project "Project1" in deployed application "App2."

    This section assigns values to parameters created in the following objects:

    * Mapping "Map1"
  -->
  <application name="App2">
    <mapping name="Map1">
      <project name="Project1">
        <mapping name="Map1">
          <parameter name="MAP1_PARAM2">MAP1_PARAM2_VAL</parameter>
        </mapping>
      </project>
    </mapping>
  </application>
</root>
```



## パラメータファイルのエクスポート

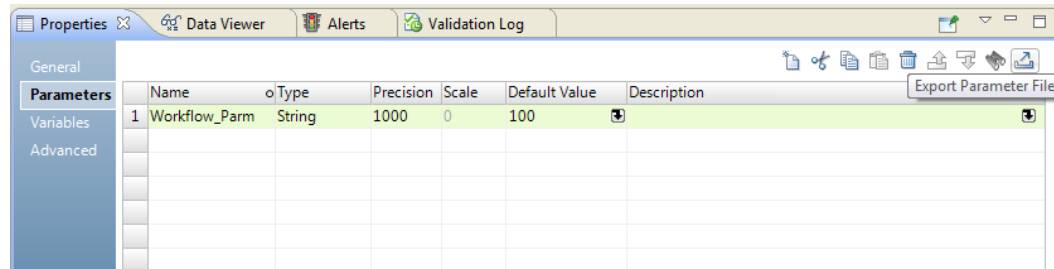
Developer tool から、マッピングパラメータファイルまたはワークフローパラメータファイルのエクスポートできます。Developer tool でパラメータを定義してから、そのパラメータをファイルにエクスポートします。Developer tool では、.XML 形式でパラメータファイルを作成します。

マッピングパラメータまたはワークフローパラメータを含むパラメータファイルのエクスポートできます。マッピングの **【パラメータ】** タブまたはワークフローの **【パラメータ】** タブから、パラメータをエクスポートできます。Developer tool によって、**【パラメータ】** タブからすべてのパラメータがエクスポートされます。

パラメータファイルのエクスポートするには、次の手順を実行します。

1. マッピングまたはワークフローのパラメータおよびパラメータのデフォルトを定義します。
2. マッピングまたはワークフローの **【プロパティ】** の **【パラメータ】** タブで、**【パラメータファイルのエクスポート】** オプションをクリックします。
3. パラメータファイルの名前を入力して、ファイルを配置する場所を参照します。
4. **【保存】** をクリックします。

次の図は、ワークフローの **【パラメータ】** タブの **【パラメータファイルのエクスポート】** オプションを示しています。



パラメータファイルのエクスポートすると、Developer tool によって、マッピングパラメータまたはワークフローパラメータのいずれかを含むパラメータファイルが作成されます。Developer tool では、マッピングパラメータとワークフローパラメータを同じファイルにエクスポートしません。

例えば、ワークフローパラメータ Workflow\_Parm をエクスポートすると、Developer tool では次のパラメータファイルを作成します。

```
<?xml version="1.0" encoding="UTF-16LE"?>
<root version="2.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema" xmlns="http://www.informatica.com/Parameterization/1.0">
  <project name="Orders">
    <workflow name="Customer_Workflow">
      <parameter name="Workflow_Parm">100</parameter>
    </workflow>
  </project>
</root>
```

## infacmd ms ListMappingParams からのパラメータファイルの作成

infacmd ms ListMappingParams コマンドを実行して、デプロイ済みアプリケーションの mapping で使用されるパラメータと各パラメータのデフォルト値をリスト表示します。このコマンドの出力を使用して、パラメータファイルを作成します。

1. infacmd ms ListMappingParams コマンドを実行して、mapping で使用されるパラメータと各パラメータのデフォルト値をリスト表示します。  
-o 引数は、コマンド出力を XML ファイルに出力します。



例えば、次のコマンドでは、マッピング「MyMapping」のパラメータがファイル「MyOutputFile.xml」にリストされます。

```
infacmd ms ListMappingParams -dn MyDomain -sn MyDataIntSvs -un MyUser -pd MyPassword -a MyApplication -m MyMapping -o MyOutputFile.xml
```

データ統合サービスは、すべてのマッピングパラメータをデフォルト値とともにリスト表示します。

2. -o 引数を指定しない場合、コマンド出力を XML ファイルにコピーしてファイルを保存できます。
3. XML ファイルを編集し、パラメータのデフォルト値を、mapping の実行時に使用する値に置き換えます。
4. XML ファイルを保存します。

## パラメータ階層

パラメータ階層は、ユーザー定義されたパラメータを作成できるレベルを定義します。

階層は次の順序で構成されています。

```
Workflow parameters
  Mapping parameters
    Mapplet parameters
      Logical data objects
        Transformation/data object parameters
```

階層内の上位のオブジェクトに割り当てられたパラメータは、階層の下位に割り当てられたパラメータを置き換えることができます。階層を使用して、次のようにパラメータをオーバーライドできます。

- 階層内の上位のオブジェクトにパラメータを割り当てることで、階層内の下位のオブジェクトに割り当てられたパラメータをオーバーライドします。
- 下位のオブジェクトパラメータを上位のオブジェクトパラメータにバインドして、ランタイムマッピングまたはワークフロー内の上位のオブジェクトパラメータのデフォルト値を使用します。

例えば、再利用可能なトランスフォーメーションをマッピングに追加できます。再利用可能なトランスフォーメーション内のパラメータを置き換えるには、マッピングパラメータを作成してトランスフォーメーションパラメータをオーバーライドします。または、トランスフォーメーションパラメータをマッピングパラメータとして公開し、トランスフォーメーションパラメータをマッピングパラメータにバインドできます。トランスフォーメーションは実行時にデフォルトのマッピングパラメータ値を使用します。

## パラメータ階層を使用したパラメータのオーバーライド

トランスフォーメーション、論理データオブジェクト、マップレット、またはマッピングのパラメータを作成する場合、パラメータ階層の上位でトランスフォーメーション、論理データオブジェクト、マップレット、またはマッピングの新しいパラメータを指定することでパラメータをオーバーライドできます。

例えば、トランスフォーメーションでパラメータを作成します。次に、トランスフォーメーションをマップレットに追加します。これでトランスフォーメーションのデフォルトのパラメータ値を使用できます。また、マップレットパラメータを作成してトランスフォーメーションのパラメータ値を上書きすることもできます。

パラメータセットまたは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することもできます。マッピング内のパラメータは、パラメータセットまたはパラメータファイル内で指定されたパラメータを使用します。パラメータセットまたはパラメータファイルでは、ワークフローまたはマッピングパラメータのみを指定できます。

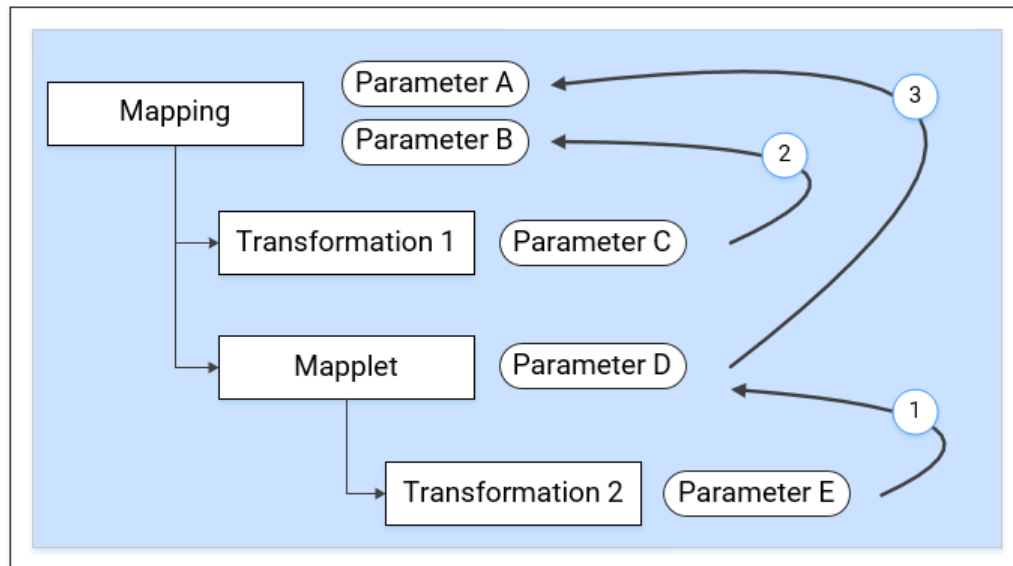
## 実行時にパラメータをバインドしてパラメータをオーバーライドする

再利用可能オブジェクトをマッピングまたはワークフローに追加するときは、パラメータ階層内の上位パラメータにパラメータをバインドし、再利用可能オブジェクトパラメータのデフォルト値をオーバーライドします。実行時に、データ統合サービスは上位パラメータを再利用可能オブジェクトに適用します。

たとえば、再利用可能トランスフォーメーションおよびトランスフォーメーション内のパラメータを作成します。次に、再利用可能トランスフォーメーションをマププレットに追加します。これでトランスフォーメーションのデフォルトのパラメータ値を使用できます。あるいは、トランスフォーメーションパラメータファイルをマププレットパラメータにバインドできます。実行時にマププレットパラメータのデフォルト値を変更して、トランスフォーメーションパラメータのデフォルト値をオーバーライドできます。

ユーザー定義パラメータをバインドするには、パラメータを公開します。たとえば、再利用可能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、トランスフォーメーションパラメータをマッピングパラメータとして公開し、トランスフォーメーションパラメータをマッピングパラメータにバインドします。

次の図は、データ統合サービスが Developer tool のマッピングでパラメータをバインドする方法を示しています。



1. トランスフォーメーション内のパラメータは、マププレット内のパラメータにバインドします。実行時に、トランスフォーメーションパラメータはマププレットのデフォルト値を使用します。
2. マププレット内のパラメータは、マッピング内のパラメータにバインドします。実行時に、マププレットパラメータはマッピングパラメータのデフォルト値を使用します。
3. トランスフォーメーション内のパラメータは、マッピング内のパラメータにバインドします。実行時に、トランスフォーメーションパラメータはマッピングパラメータのデフォルト値を使用します。

## マッピング内のパラメータのオーバーライドのユースケース

再利用不可能および再利用可能なマッピングオブジェクトのパラメータをオーバーライドするにあたって、さまざまなガイドラインがあります。マッピングパラメータを作成または再設定することで、再利用不可能なオブジェクトおよび再利用可能なオブジェクトのパラメータの両方をオーバーライドできます。再利用可能なオブジェクトパラメータについては、まずオブジェクトパラメータをマッピングパラメータとして公開します。次に、マッピングパラメータを再設定します。

マッピングパラメータをオーバーライドするには、マッピングを実行するときにパラメータセットまたはパラメータファイルを指定します。

## マッピング内の再利用不可能なトランスフォーメーションパラメータのオーバーライド

マッピング内で再利用不可能なトランスフォーメーションを作成する場合、トランスフォーメーションパラメータはマッピングパラメータです。

トランスフォーメーション内のデフォルトのパラメータ値をオーバーライドするには、次のいずれかのタスクを完了します。

- マッピングパラメータを再設定する。
- パラメータセットまたはパラメータファイルを設定してマッピングパラメータをオーバーライドする。

例えば、マッピング内で再利用不可能なフィルタトランスフォーメーションを作成し、フィルタトランスフォーメーションのパラメータを設定します。Developer tool がトランスフォーメーションパラメータを複製し、マッピングパラメータを作成します。トランスフォーメーションパラメータをオーバーライドするには、マッピングパラメータのデフォルト値を再設定するか、パラメータセットまたはパラメータファイルを設定してマッピングパラメータをオーバーライドします。

## マッピング内の再利用可能なトランスフォーメーションパラメータのオーバーライド

再利用可能なトランスフォーメーションをマッピングに追加すると、トランスフォーメーションパラメータは、そのパラメータがマッピングパラメータとして公開されるまではトランスフォーメーションパラメータです。トランスフォーメーションパラメータをマッピングパラメータとして公開する場合は、トランスフォーメーションパラメータをマッピングパラメータにバインドします。マッピングパラメータは、トランスフォーメーションパラメータと同じ名前とタイプを持ちます。マッピングパラメータのデフォルト値は、再利用可能なトランスフォーメーションパラメータのインスタンス値を使用します。

再利用可能なトランスフォーメーション内のパラメータ値をオーバーライドするには、次のいずれかのタスクを完了します。

- トランスフォーメーションパラメータをオーバーライドするための新しいマッピングパラメータを作成する。
- マッピングパラメータとしてトランスフォーメーションパラメータを公開する。マッピングパラメータのデフォルト値を再設定する。
- マッピングパラメータとしてトランスフォーメーションパラメータを公開する。パラメータセットまたはパラメータファイルを設定してマッピングパラメータのデフォルト値をオーバーライドする。

例えば、再利用可能なフィルタトランスフォーメーションのパラメータを設定できます。再利用可能なフィルタトランスフォーメーションをマッピングに追加すると、マッピングパラメータを設定してフィルタトランスフォーメーションパラメータをオーバーライドしたり、マッピングパラメータとしてフィルタトランスフォーメーションパラメータを公開したりできます。フィルタトランスフォーメーションパラメータをマッピングパラメータとして公開すると、フィルタトランスフォーメーションパラメータはマッピングパラメータにバインドされます。公開されたトランスフォーメーションパラメータをオーバーライドするには、トランスフォーメーションにバインドするマッピングパラメータのデフォルト値を再設定します。実行時に、トランスフォーメーションはマッピングパラメータのデフォルト値を使用します。

## 再利用可能なマップレット内の再利用不可能なトランスフォーメーションパラメータのオーバーライド

再利用不可能なトランスフォーメーションを含む再利用可能なマップレットを作成すると、トランスフォーメーションは再利用不可能なままになります。マップレット内で再利用不可能なトランスフォーメーションを作成する場合、トランスフォーメーションパラメータはマップレットパラメータです。再利用不可能なトランスフォーメーションを含む再利用可能なマップレットをマッピングに追加すると、マップレット内の再利用不可

能なトランスフォーメーションは、マッピング内で再利用不可能なトランスフォーメーションのままになります。

マップレットパラメータをマッピングパラメータとして公開できます。マップレットパラメータを公開すると、マップレットパラメータはマッピングパラメータにバインドされ、再利用不可能なトランスフォーメーションパラメータはマッピングパラメータになります。

再利用可能なマップレットパラメータをオーバーライドするには、次のいずれかのタスクを完了します。

- マッピングパラメータを設定する。
- マップレットパラメータをマッピングパラメータとして公開する。マップレットパラメータを公開すると、実行時に、マップレットパラメータはマッピングパラメータのデフォルト値を使用します。マッピングパラメータを再設定する。
- マップレットパラメータをマッピングパラメータとして公開する。マップレットパラメータを公開すると、実行時に、マップレットパラメータはマッピングパラメータのデフォルト値を使用します。パラメータセットまたはパラメータファイルを設定してマッピングパラメータをオーバーライドする。

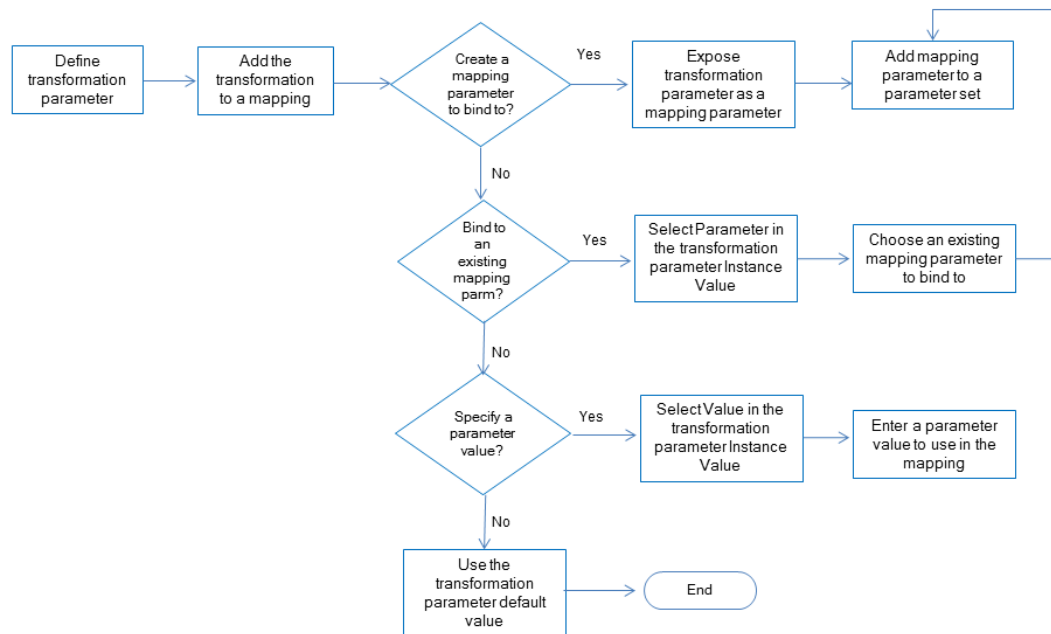
再利用不可能なトランスフォーメーションパラメータをオーバーライドするには、次のいずれかのタスクを完了します。

- マップレットパラメータを再設定する。
- マップレットパラメータをマッピングパラメータとして公開する。マップレットパラメータを公開すると、再利用不可能なトランスフォーメーションパラメータはマッピングパラメータになります。マッピングパラメータを再設定する。
- マップレットパラメータをマッピングパラメータとして公開する。マップレットパラメータを公開すると、再利用不可能なトランスフォーメーションパラメータはマッピングパラメータになります。パラメータセットまたはパラメータファイルを設定してマッピングパラメータをオーバーライドする。

## パラメータの設定方法

トランスフォーメーション、マッピング、マップレット、またはワークフローでパラメータを定義します。

次の図は、マッピング内の再利用可能なトランスフォーメーションでパラメータを使用するためのプロセスを示しています。



1. 再利用可能なトランスフォーメーションで、トランスフォーメーション内のプロパティのパラメータ、または式エディタ内の変数のパラメータを作成します。
2. トランスフォーメーションをマッピングまたはマップレットに追加します。
3. トランスフォーメーションの **【パラメータ】** タブで、マッピングまたはマップレットでパラメータ値を設定する方法を選択します。
  - トランスフォーメーションパラメータをマッピングパラメータとして公開します。マッピングレベルで、トランスフォーメーションパラメータの複製を作成します。
  - トランスフォーメーションパラメータをマッピングパラメータにバインドします。マッピングパラメータを参照するか手動で作成して、トランスフォーメーションパラメータにバインドします。
  - 特定のパラメータ値を入力します。マッピング実行で使用するデフォルト値を入力します。
  - トランスフォーメーションパラメータのデフォルト値を使用します。マッピング内で元のパラメータ値を使用します。

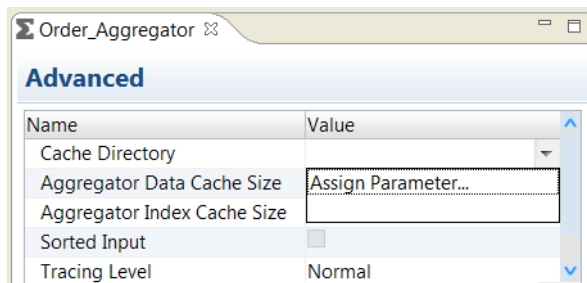
マッピングパラメータをトランスフォーメーションパラメータにバインドしたら、パラメータセットまたはパラメータファイルを作成してマッピングパラメータの値を実行時にオーバーライドできます。Developer tool またはコマンドラインからマッピングを実行します。パラメータセットまたはパラメータファイルを指定して、マッピングの実行に使用します。

## トランスフォーメーションプロパティのパラメータの作成

フィールドまたはトランスフォーメーションプロパティにパラメータを割り当てる場合、使用するパラメータを参照するか、該当のフィールド専用のパラメータを作成することができます。

1. 更新するフィールドまたはプロパティに移動します。
2. **【値】** カラムで選択矢印をクリックします。

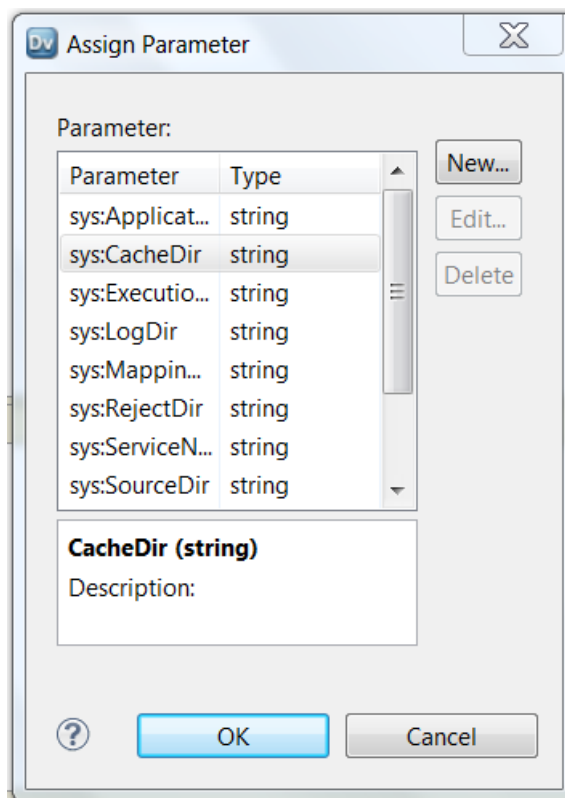
プロパティをパラメータ化できる場合、**【パラメータの割り当て】** オプションが表示されます。次の図は、キャッシュディレクトリの **【パラメータの割り当て】** オプションを示しています。



3. **【パラメータの割り当て】** をクリックします。

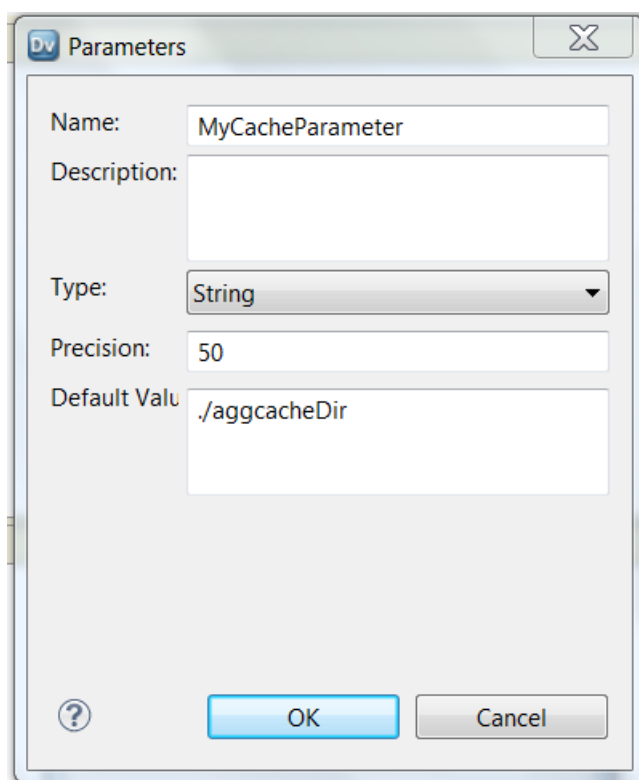
**【パラメータの割り当て】** ダイアログボックスが表示されます。ダイアログボックスには、システムパラメータと、トランスフォーメーション内で作成したユーザー定義のパラメータが表示されます。

次の図は、**【パラメータの割り当て】** ダイアログボックスを示しています。



4. パラメータを作成するには、**【新規】** をクリックします。
5. パラメータ名、タイプ、精度、およびデフォルト値を入力します。

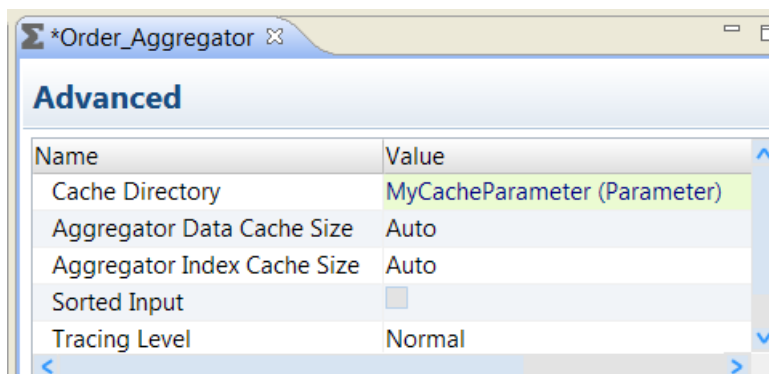
次の図は、**【パラメータ】** ダイアログボックス内の MyCacheParameter というパラメータを示しています。



6. **[OK]** をクリックします。

トランスフォーメーションプロパティにパラメータ名が表示されます。

次の図は、アグリゲータトランスフォーメーションのキャッシュディレクトリ内の MyCacheParameter を示しています。



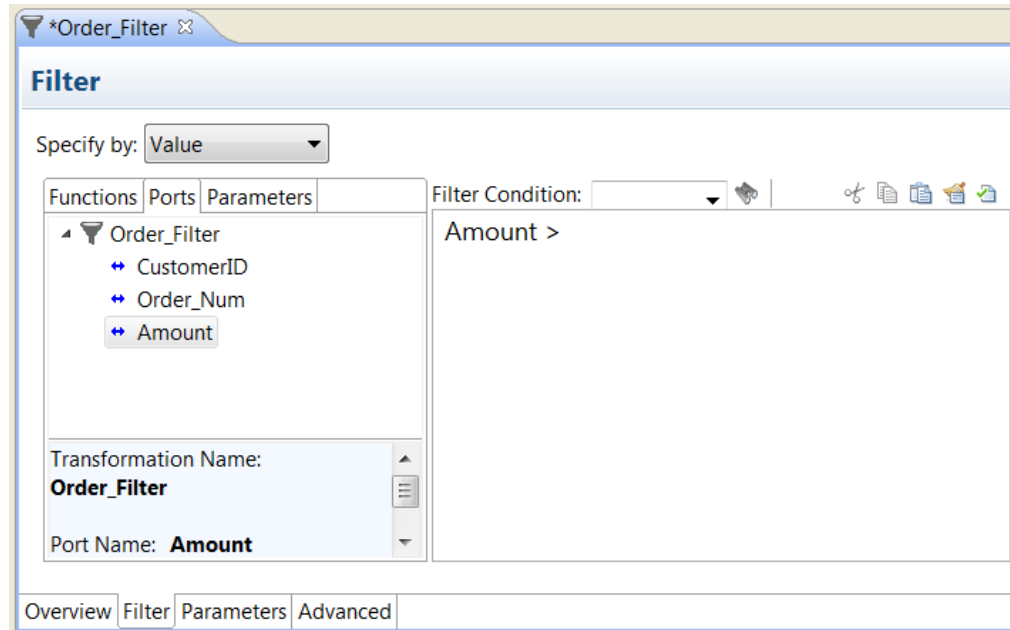
トランスフォーメーションの **[パラメータ]** タブでは、パラメータの追加、変更、削除ができます。

## 式内のパラメータの作成

パラメータを定義したら、式内でパラメータを参照できます。次の例は、フィルタ式のコンポーネントにパラメータを割り当てる方法を示しています。

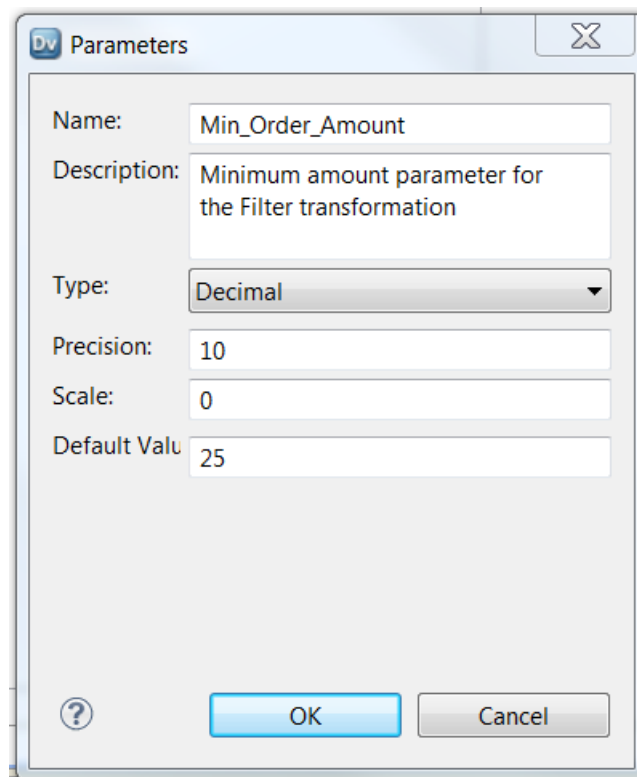
1. フィルタトランスフォーメーションで、**[フィルタ]** タブをクリックします。  
式エディタが表示されます。関数、ポート、およびパラメータを選択して式を作成できます。
2. 式パラメータを使用する代わりに式を定義するには、**[指定元:]** で **[値]** を選択します。

3. [フィルタ] タブで、[ポート] タブをクリックします。
4. Amount ポートを選択します。 [関数] タブで、「より大きい (>)」関数を選択します。  
次の図は、Amount ポートと「>」演算子を含む式を示しています。



5. 式エディタの [パラメータ] タブをクリックします。  
式エディタには、システムパラメータとユーザー定義のパラメータのリストが表示されます。
6. [パラメータの管理] をクリックしてパラメータを追加します。  
[パラメータ] ダイアログボックスが表示されます。
7. [新規] をクリックします。  
デフォルトのパラメータ値でダイアログボックスが表示されます。
8. パラメータ名、パラメータタイプ、精度、およびデフォルト値を入力します。  
次の図は、[パラメータ] ダイアログボックスを示しています。



A screenshot of a 'Parameters' dialog box. The dialog has a title bar with a 'Dv' icon and a close button. It contains several input fields: 'Name' with the value 'Min\_Order\_Amount', 'Description' with the text 'Minimum amount parameter for the Filter transformation', 'Type' with a dropdown menu set to 'Decimal', 'Precision' with the value '10', 'Scale' with the value '0', and 'Default Value' with the value '25'. At the bottom, there is a help icon (question mark), an 'OK' button, and a 'Cancel' button.

**Parameters**

Name: Min\_Order\_Amount

Description: Minimum amount parameter for the Filter transformation

Type: Decimal

Precision: 10

Scale: 0

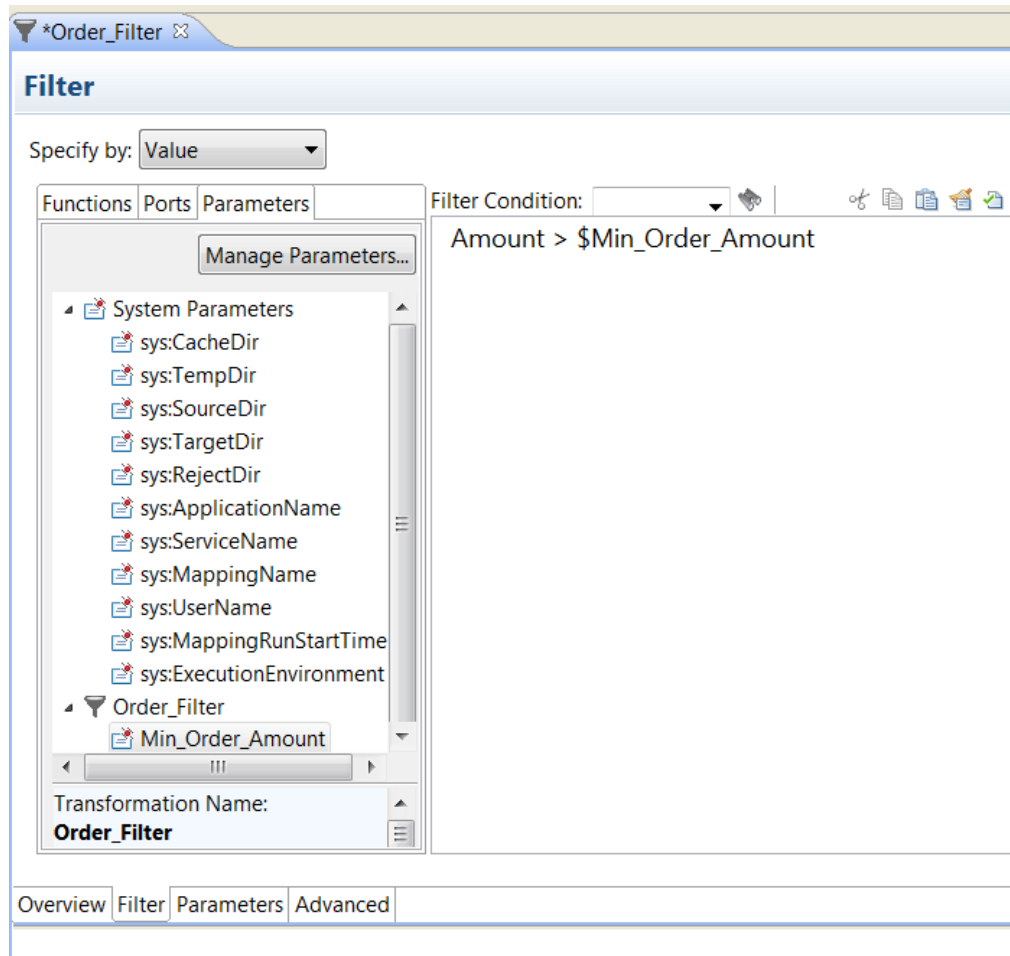
Default Value: 25

OK Cancel

9. 式エディタで、**[OK]** をクリックします。  
作成したパラメータがパラメータリストに表示されます。

10. Min\_Order\_Amount パラメータを選択して式に追加します。

Min\_Order\_Amount パラメータが式に表示されます。



パラメータは、式内ではドル記号（\$）識別子付きで表示されます。Min\_Order\_Amount のデフォルト値は 50 です。Min\_Order\_Parameter をオーバーライドせずにトランスフォーメーションをマッピングに追加した場合、フィルタトランスフォーメーションは、50 より大きい金額を持つ行を返します。

## マッピングパラメータとしてトランスフォーメーションパラメータを公開する

マッピングに再利用可能なトランスフォーメーションを追加したら、トランスフォーメーションパラメータをマッピングパラメータとして公開できます。再利用可能なトランスフォーメーションパラメータをマッピングパラメータとして公開する場合は、トランスフォーメーションパラメータからマッピングパラメータを作成します。マッピングパラメータは、トランスフォーメーションパラメータと同じ名前とタイプを持ちます。マッピングパラメータは、再利用可能なトランスフォーメーションパラメータのインスタンス値を使用します。

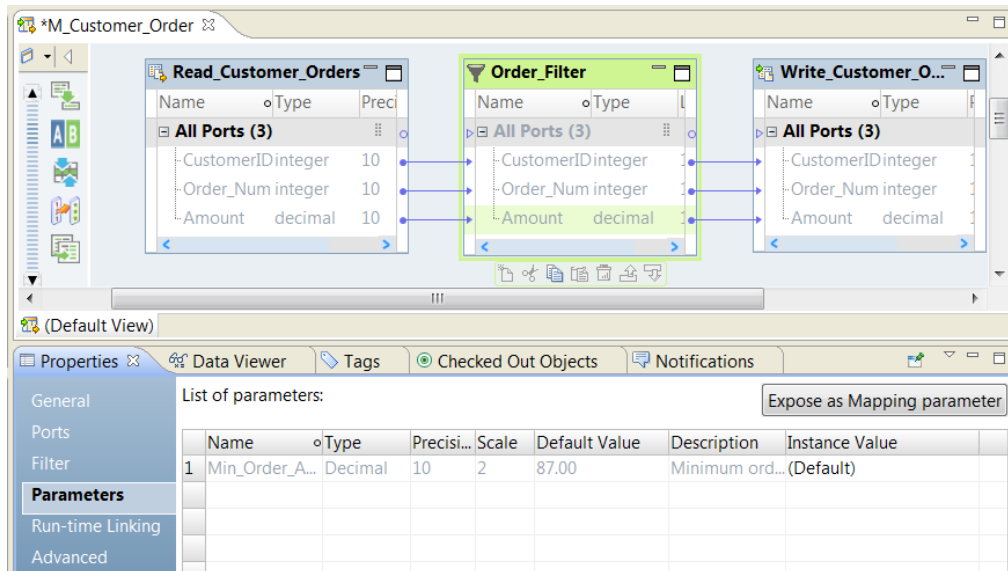
トランスフォーメーションパラメータに対して、**[マッピングパラメータとして公開]** を 1 回クリックできます。**[マッピングパラメータとして公開]** をクリックし、トランスフォーメーションパラメータがすでにマッピングパラメータにバインドされている場合、Developer tool はマッピングパラメータを変更しません。Developer tool は別のマッピングパラメータを作成することも、マッピングパラメータのデフォルト値を更新することはありません。マッピングパラメータを作成すると、複数のオブジェクトでそのマッピングパラメータ

タが使用される場合があります。マッピングパラメータのデフォルト値を変更する必要がある場合は、マッピングで値を変更するか、実行時に値を変更してください。

1. マッピングを開きます。[アウトライン] ビューでマッピングを選択します。

[パラメータ] タブが [プロパティ] ビューに表示されます。

次の図は、フィルタ変換の [パラメータ] タブを示しています。



2. パラメータのマッピングパラメータを作成するには、パラメータを選択し、[マッピングパラメータとして公開] をクリックします。

Developer tool は、同じ名前で作成して変換パラメータにバインドします。

3. マッピングパラメータを更新するには、[アウトライン] ビューからそのパラメータを選択します。

デフォルトのマッピングパラメータ値を変更できます。マッピングの [パラメータ] タブでは、マッピングパラメータを追加することもできます。

## パラメータインスタンス値の設定

変換の [パラメータ] タブの [インスタンス値] カラムで、パラメータのインスタンス値を設定できます。重複したマッピングパラメータを作成しない場合は、このカラムでインスタンス値を設定します。

変換パラメータをデフォルト値に設定したり、既存のマッピングパラメータを変換パラメータにバインドしたりできます。

1. 変換をマッピングに追加したら、変換の [プロパティ] ビューで [パラメータ] タブをクリックします。
2. マッピングパラメータを変換パラメータにバインドするには、以下の手順を実行します。
  - a. 変換パラメータの [インスタンス値] カラムをクリックします。  
[指定元] ダイアログボックスが表示されます。
  - b. [指定元:] で [パラメータ] をクリックします。

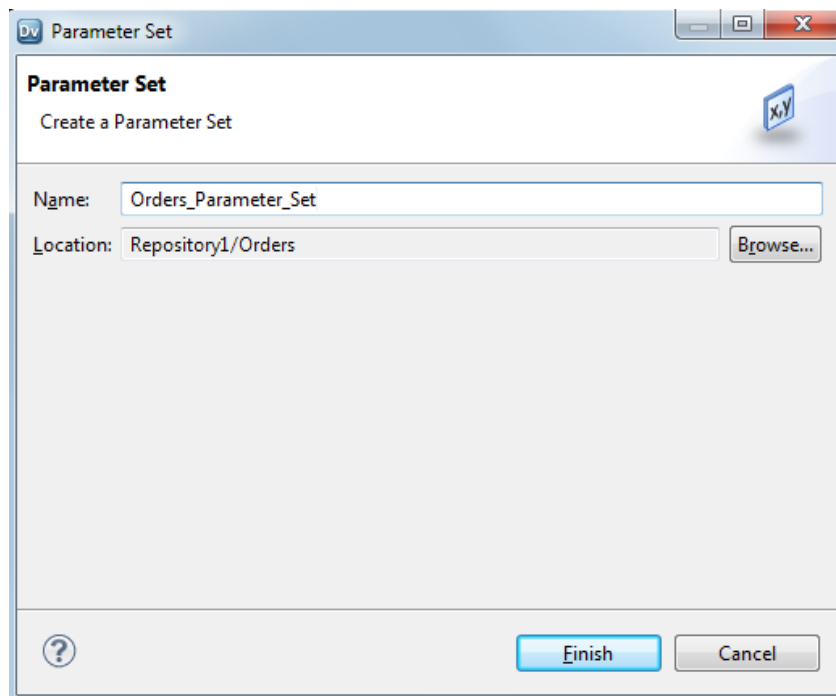
- c. **【パラメータの割り当て】** ダイアログボックスで、マッピングパラメータまたはシステム定義のパラメータを参照して選択し、トランスフォーメーションパラメータにバインドします。
  - d. **【OK】** をクリックします。  
**【指定元】** ダイアログボックスに、マッピングパラメータ名がパラメータ値として表示されます。
  - e. **【指定元】** ダイアログボックスで **【OK】** をクリックします。  
**【インスタンス値】** カラムにマッピングパラメータ名が表示されます。
3. トランスフォーメーションパラメータインスタンスのデフォルト値を設定するには、以下の手順を実行します。
    - a. トランスフォーメーションパラメータの **【インスタンス値】** カラムをクリックします。  
**【指定元】** ダイアログボックスが表示されます。
    - b. デフォルト値を入力するには、**【指定元:】** で **【値】** をクリックし、インスタンスのデフォルト値を入力します。
    - c. トランスフォーメーションパラメータのデフォルト値を使用するには、**【デフォルトを使用】** をクリックします。

## パラメータセットの作成

マッピングおよびワークフローのランタイムコンテキスト変更に使用できるパラメータセットを作成します。

パラメータセットを作成する場合、パラメータを含めるためのマッピングまたはワークフローを選択します。マッピングまたはワークフローを選択後、パラメータセットに手動でパラメータを入力するか、パラメータを選択できます。

1. **【オブジェクトエクスプローラ】** ビューで、プロジェクトを右クリックし、**【新規】** > **【パラメータセット】** をクリックします。
2. パラメータセット名を入力して、**【完了】** をクリックします。



3. **【プロパティ】** パネルをドラッグしてグリッドを表示し、パラメータをパラメータセットに追加します。

4. **【新規】 > 【マッピング/ワークフロー】** をクリックします。

**Overview**

**General**

Name: Orders\_Parameter\_Set

Description:

**Parameter Set**

Object Name	Parameter Name	Value	Type	Path

Mapping\Workflow  
Parameter

5. **パラメータの追加**ダイアログボックスで、**【参照】** をクリックし、セットに含める必要があるパラメータを含むマッピングまたはワークフローを見つけます。  
マッピングおよびワークフローのリストが表示されます。
6. マッピングまたはワークフローを選択し、**【OK】** をクリックします。  
マッピングまたはワークフローのパラメータのリストが表示されます。

**Add Parameters**

Select the objects to associate:

m\_Customer\_Order

Browse...

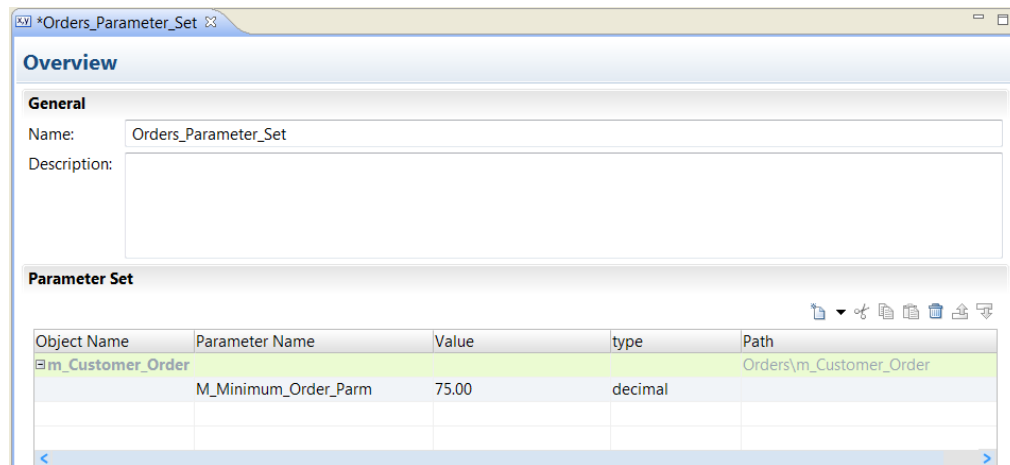
Key Values:

Name	Value	Type
<input checked="" type="checkbox"/> M_Minimum_Order_Parm	75.00	decimal

Select All  
Deselect All

OK Cancel

7. パラメータセットに含めるパラメータを選択し、**【OK】** をクリックします。  
パラメータセットにマッピングまたはワークフロー名とパスが表示されます。選択した各パラメータがオブジェクトの下に表示されます。



8. ワークフローまたはマッピングにないパラメータを追加するには、マッピングまたはオブジェクト名を右クリックして【パラメータ】挿入を選択します。

Developer tool によりワークフローまたはマッピングの下にパラメータが作成されます。パラメータ名、値、およびタイプを変更します。

**注:** パラメータセットを使用する前にマッピングまたはワークフローにパラメータを追加する必要があります。

## パラメータを使用してマッピングを実行する方法

Developer tool またはコマンドラインから、パラメータを使用してマッピングを実行できます。

Developer tool からパラメータを使用してマッピングを実行するには、詳細オプションを使用してマッピングを実行します。詳細オプションで、パラメータを指定します。コマンドラインからマッピングを実行するには、マッピングとパラメータセットまたはパラメータファイルを指定します。

### Developer tool からパラメータを使用してマッピングを実行する方法

Developer tool からパラメータを使用してマッピングを実行する前に、パラメータを検証します。パラメータを検証するには、マッピングパラメータを解決してマッピングを検証します。次に、詳細オプションを使用してマッピングを実行します。詳細オプションで、デフォルトのパラメータ値、パラメータセット、またはパラメータファイルを使用するかどうかを指定します。

#### パラメータの解決と検証

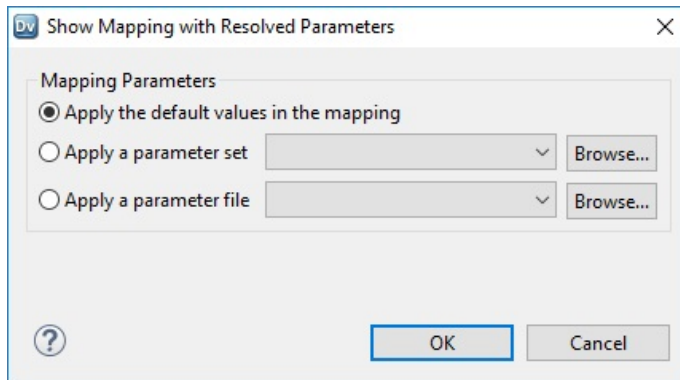
データ統合サービスが実行時にパラメータを読み取って処理できるように、マッピングパラメータを解決および検証できます。パラメータを解決する前に、マッピングが有効になっている必要があります。マッピングパ

ラメータを解決すると、Developer tool では、データ統合サービスが実行時にパラメータをどのように解決するかを示すマッピングのインスタンスを生成します。

さまざまなパラメータセットまたはパラメータファイルを使用して複数回マッピングを実行する場合、毎回解決済みのパラメータを使用してマッピングを検証し、パラメータセットまたはパラメータファイルにマッピングとの互換性があることを確認してください。

1. Developer tool で、エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックします。**[解決済みパラメータを使用したマッピングの表示]** を選択します。

**[解決済みパラメータを使用したマッピングの表示]** ダイアログボックスが表示されます。



2. 次のいずれかのマッピングパラメータオプションを選択します。
  - マッピングでのデフォルト値の適用データ統合サービスは、マッピング内で設定されたデフォルトのパラメータ値を適用します。
  - パラメータセットの適用データ統合サービスは、パラメータセットで設定されたパラメータ値を適用します。
  - パラメータファイルの適用データ統合サービスは、パラメータファイルで設定されたパラメータ値を適用します。
3. **[OK]** をクリックします。

Developer tool が、解決されたパラメータを示すマッピングのランタイムインスタンスを生成します。マッピングのランタイムインスタンスが新しいタブに表示されます。
4. マッピングのランタイムインスタンスを右クリックして、**[検証]** を選択します。

**[検証ログ]** ビューにエラーが表示される場合は、エラーを修正し、再度マッピングを検証します。
5. 選択したパラメータを使用してマッピングを実行するには、マッピングのランタイムインスタンスを右クリックして、**[実行]** をクリックします。

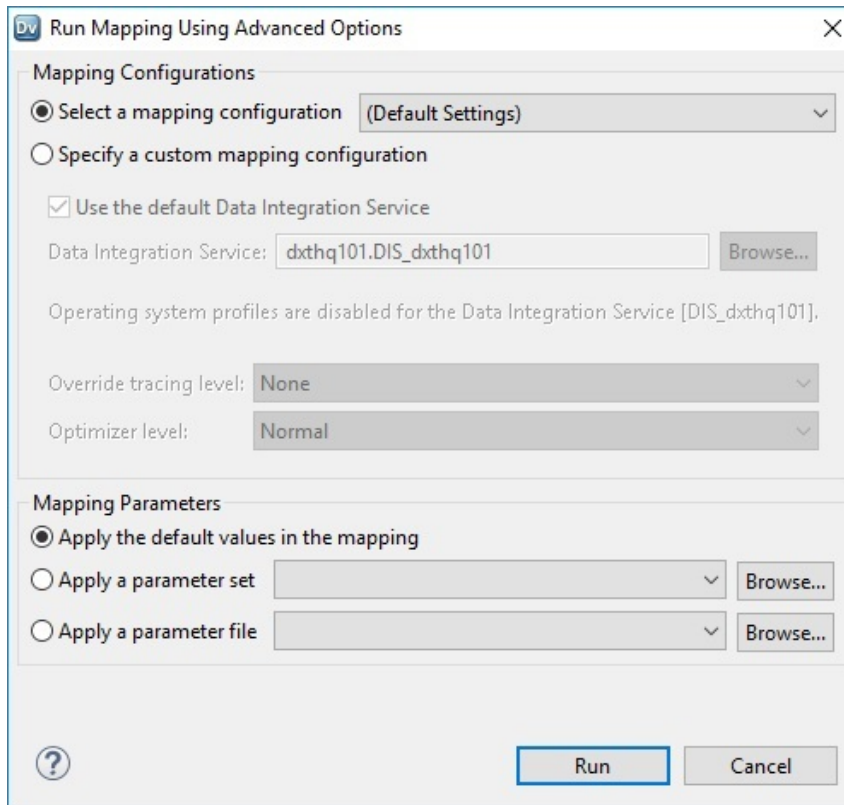
## マッピングの実行パラメータを使用

Developer tool からパラメータを使用してマッピングを実行するには、詳細オプションを使用してマッピングを実行します。Developer tool で、マッピング、パラメータセットまたはパラメータファイル内のデフォルトのパラメータ値を使用してマッピングを実行できます。

パラメータが解決されたマッピングインスタンスを実行する場合、異なるパラメータを指定してマッピングを実行することはできません。マッピング内で解決されたパラメータを使用してマッピングが実行されます。

1. Developer tool で、エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックします。[詳細オプションを使用したマッピングの実行] を選択します。

[詳細オプションを使用したマッピングの実行] ダイアログボックスが表示されます。



The dialog box titled "Run Mapping Using Advanced Options" contains two main sections: "Mapping Configurations" and "Mapping Parameters".

**Mapping Configurations:**

- ☒ Select a mapping configuration (Default Settings) [v]
- ☐ Specify a custom mapping configuration
- ☒ Use the default Data Integration Service
- Data Integration Service: dxtmq101.DIS\_dxtmq101 [Browse...]
- Operating system profiles are disabled for the Data Integration Service [DIS\_dxtmq101].
- Override tracing level: None [v]
- Optimizer level: Normal [v]

**Mapping Parameters:**

- ☒ Apply the default values in the mapping
- ☐ Apply a parameter set [v] [Browse...]
- ☐ Apply a parameter file [v] [Browse...]

At the bottom, there is a question mark icon, a "Run" button, and a "Cancel" button.

2. 次のいずれかのマッピング設定オプションを選択します。
  - マッピング設定の選択再利用可能なマッピング設定を選択し、マッピングを実行します。
  - カスタムマッピング設定の指定データ統合サービス、オペレーティングシステムプロファイル、オーバーライドトレースレベル、最適化レベルを選択します。
3. 次のいずれかのマッピングパラメータオプションを選択します。
  - マッピングでのデフォルト値の適用データ統合サービスは、マッピング内で設定されたデフォルトのパラメータ値を適用します。
  - パラメータセットの適用データ統合サービスは、パラメータセットで設定されたパラメータ値を適用します。
  - パラメータファイルの適用データ統合サービスは、パラメータファイルで設定されたパラメータ値を適用します。



## コマンドラインからパラメータを使用してマッピングを実行する方法

コマンドラインからパラメータを使用してマッピングを実行するには、マッピングをアプリケーションとしてデプロイする必要があります。アプリケーションをデプロイした後、マッピングを実行します。マッピングおよびパラメータセットまたはパラメータファイルを指定します。

### パラメータセットを使用したマッピングの実行

コマンドラインからパラメータセットを使用してマッピングを実行するには、マッピングをアプリケーションとしてデプロイして、アプリケーション内でパラメータセットを指定する必要があります。デプロイ済みのマッピングを実行し、パラメータセットを指定します。

異なるパラメータセットを使用する必要がある場合は、アプリケーションに複数のパラメータセットをデプロイできます。マッピングを実行すると、使用するパラメータセットを指定できます。

アプリケーションをデプロイした後、`infamcd addParameterSetEntries` コマンドを使用して、パラメータセットのエントリを追加します。`infacmd updateParameterSetEntries` コマンドを使用して、パラメータセットのエントリを更新します。

`infacmd` を使用してパラメータセットを使用する方法の詳細については、『*Informatica コマンドリファレンス*』を参照してください。

### パラメータファイルを使用した Mapping の実行

コマンドラインからパラメータファイルを使用してマッピングを実行するには、マッピングをアプリケーションとしてデプロイする必要があります。マッピングを実行してパラメータファイルを指定します。`infacmd ms RunMapping` コマンドを使用します。`-pf` 引数でパラメータファイル名を指定します。

例えば、次のコマンドでは、マッピング「MyMapping」がパラメータファイル「MyParamFile.xml」を使用して実行されます。

```
infacmd ms RunMapping -dn MyDomain -sn MyDataIntSvs -un MyUser -pd MyPassword -a MyApplication -m MyMapping -pf MyParamFile.xml
```

パラメータファイルを使用してマッピングを実行し、そのファイルが無効であった場合、データ統合サービスは、mapping を失敗にします。パラメータファイルが見つからないまたはアクセスできない場合、データ統合サービスは、mapping を失敗にします。

`infacmd` を使用してパラメータセットを使用する方法の詳細については、『*Informatica コマンドリファレンス*』を参照してください。

## 第 4 章

# パラメータを割り当てる場所

この章では、以下の項目について説明します。

- [パラメータを割り当てる場所概要, 82 ページ](#)
- [圧縮形式のパラメータ, 86 ページ](#)
- [Hive ソース用のカスタムクエリのパラメータ, 87 ページ](#)
- [リレーショナルソース用のカスタムクエリのパラメータ, 89 ページ](#)
- [式のパラメータ, 89 ページ](#)
- [式パラメータ, 91 ページ](#)
- [式パラメータ例, 92 ページ](#)
- [フィールドおよびプロパティ値のパラメータ, 95 ページ](#)
- [リレーショナルテーブルリソースのパラメータ, 96 ページ](#)
- [SQL 文のパラメータ, 97 ページ](#)
- [ポートリストのパラメータ, 99 ページ](#)

## パラメータを割り当てる場所概要

ユーザー定義のパラメータとシステムパラメータをフィールドに割り当てることができます。ユーザー定義のパラメータは、フィールドに割り当てる前に作成する必要があります。

オブジェクトおよびトランスフォーメーション内のいくつかのプロパティをパラメータ化できます。プロパティにパラメータを割り当てる場合、プロパティ値を設定するときにパラメータを割り当てるためのオプションが表示されます。

例えば、マッピング内で読み取りトランスフォーメーションを作成できます。読み取りトランスフォーメーションは、物理データオブジェクトに基づいて作成する再利用不可能なトランスフォーメーションです。トランスフォーメーションプロパティにパラメータを割り当てたり、物理データオブジェクトにパラメータを割り当てることができます。

パラメータ化ソース内にユーザー定義パラメータをネストすることはできません。ソースデータオブジェクトがパラメータ化されていると、ユーザー定義のパラメータをマッピングパラメータとして公開してパラメータ値を実行時にオーバーライドすることはできません。代わりに、マッピングではデフォルト値が使用されます。

次の表に、パラメータを割り当てることができるオブジェクトおよびフィールドを示します。

オブジェクト	フィールド
すべてのトランスフォーメーション	リンクの解決順序
関連付けトランスフォーメーション	キャッシュファイルディレクトリ キャッシュファイルサイズ
アドレスバリデータトランスフォーメーション	大文字小文字表記 デフォルトの国 Geocode データ型 グローバル最大フィールド長 行セパレータ 最大結果数 最適化レベル 無効なアドレスの標準化
アグリゲータトランスフォーメーション	キャッシュディレクトリ 式の要素 ポート式 グループ化
不良レコードの例外トランスフォーメーション	下限しきい値 上限しきい値
大文字小文字変換プログラムトランスフォーメーション	参照テーブル。
統合トランスフォーメーション	キャッシュファイルディレクトリ キャッシュファイルサイズ
カスタマイズデータオブジェクト	接続 データオブジェクト 所有者 SQL クエリの要素 テーブル名
カスタマイズされたデータオブジェクトの読み取り操作	カスタムクエリ フィルタ条件 結合条件 PreSQL PostSQL
カスタマイズされたデータオブジェクトの書き込み操作	PreSQL PostSQL 更新オーバーライド
ディシジョントランスフォーメーション	ディシジョンスクリプト。

オブジェクト	フィールド
重複レコードの例外トランスフォーメーション	キャッシュファイルディレクトリ 下限しきい値 上限しきい値
式トランスフォーメーション	式の要素 ポート式 ポートセクタ ソートキーリスト。ウィンドウイングのみ。
フィルタトランスフォーメーション	フィルタ条件の要素 フィルタ条件。完全な式。
フラットファイルデータオブジェクト	圧縮コーデック 圧縮形式 制御ファイルディレクトリ 制御ファイル名 接続名 デフォルトのスケール フラットファイルの区切り文字 マージファイルディレクトリ ソースファイルのディレクトリ ソースファイル名 出力ファイル名 出力ファイルディレクトリ 拒否ファイルディレクトリ ターゲットディレクトリ
ジョイナトランスフォーメーション	キャッシュディレクトリ 結合条件の要素 ポートセクタ
キージェネレータトランスフォーメーション	キャッシュファイルディレクトリ キャッシュファイルサイズ
ラベラトランスフォーメーション	参照テーブル
ルックアップトランスフォーメーション	カスタムクエリ。リレーショナルのみ。
ルックアップソースの物理データオブジェクトを除くルックアップトランスフォーメーション	データオブジェクト。再利用不可能なトランスフォーメーション。 動的ポートのルール。再利用不可能なトランスフォーメーション。 ルックアップ条件。完全な式、再利用不可能なトランスフォーメーション。 ポートセクタ。再利用不可能なトランスフォーメーション。
マッピング	Hive バージョン ランタイム環境 最大並行処理

オブジェクト	フィールド
一致トランスフォーメーション	[照合出力] タブのキャッシュディレクトリ [一致タイプ] タブのキャッシュディレクトリ [一致タイプ] タブのインデックスディレクトリ 永続方法 しきい値
非リレーショナルデータオブジェクト	接続
ランクトランスフォーメーション	キャッシュディレクトリ 式の要素 ポート式 グループ化ポート ランクポート
読み取りトランスフォーメーション	接続 カスタムクエリ。リレーショナルのみ。 データオブジェクト フィルタ条件。リレーショナルのみ。 結合条件。リレーショナルのみ。 所有者名。リレーショナルのみ。 PreSQL。リレーショナルのみ。 PostSQL。リレーショナルのみ。 リソース/テーブル名。リレーショナルのみ。
リレーショナルデータオブジェクト	フィルタ条件の要素 結合条件の要素 PreSQL クエリの要素 PostSQL クエリの要素 SQL オーバーライドの要素
ルータトランスフォーメーション	グループフィルタ条件の要素。 グループフィルタ条件。完全な式。
ソータトランスフォーメーション	ソートキー グループ化 作業ディレクトリ
SQL トランスフォーメーション	接続
標準化トランスフォーメーション	参照テーブル
トークンパーサトランスフォーメーション	参照テーブル

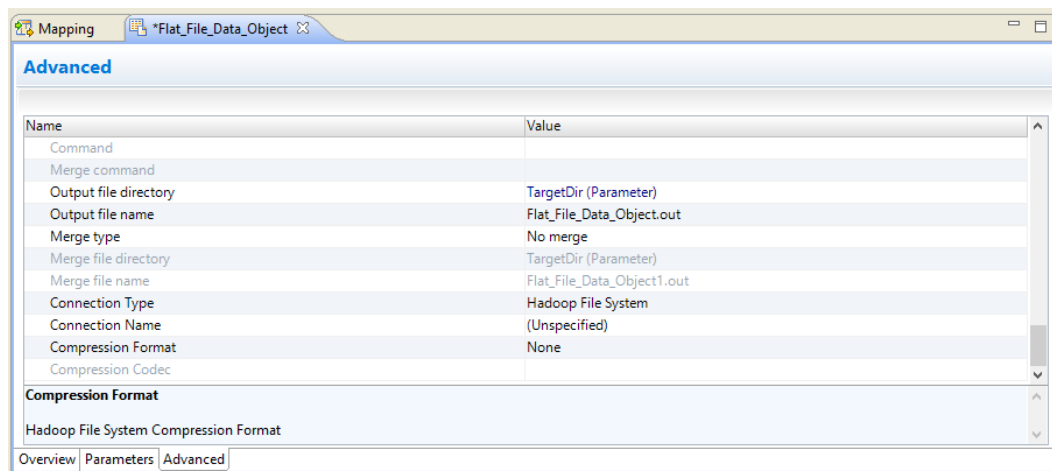
オブジェクト	フィールド
アップデートストラテジトランスフォーメーション	アップデートストラテジ式の要素。 アップデートストラテジ式。完全な式。
書き込みトランスフォーメーション	データオブジェクト リンクの解決順序 PreSQL。リレーショナルのみ。 PostSQL。リレーショナルのみ。 拒否ディレクトリ 拒否ファイル名 更新オーバーライド。リレーショナルのみ。

## 圧縮形式のパラメータ

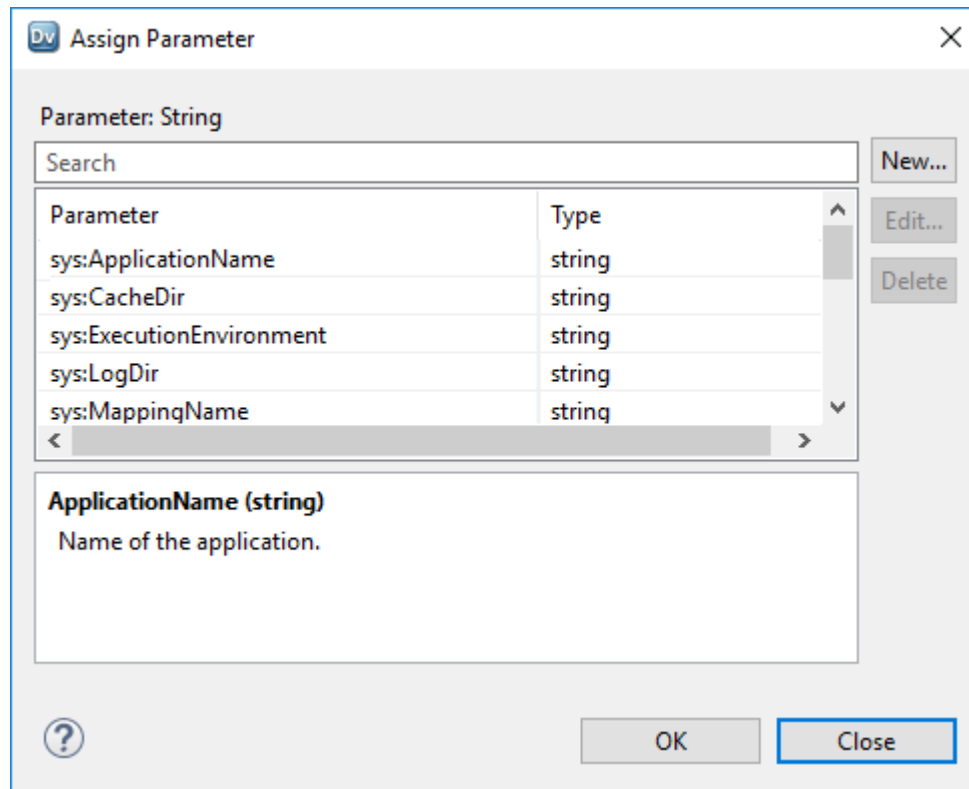
パラメータを圧縮形式および HDFS フラットファイルターゲットの圧縮コーデックに割り当てることができます。

パラメータを圧縮コーデックに割り当てる前に、圧縮形式にパラメータを割り当てる必要があります。パラメータを圧縮形式に割り当てる場合、圧縮コーデックにパラメータを割り当てる必要があります。

次の図に、圧縮形式と圧縮コーデックを設定できる HDFS フラットファイルの詳細プロパティを示します。



次の図は、パラメータを圧縮形式および圧縮コーデックに割り当てるために使用できるダイアログボックスを示しています。

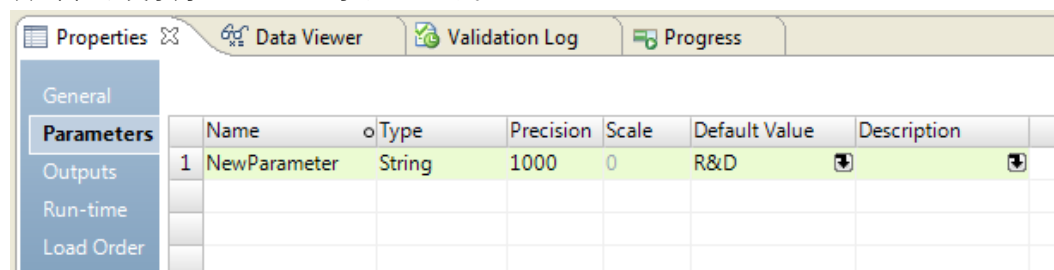


## Hive ソース用のカスタムクエリのパラメータ

Hive ソースの SQL オーバーライド、結合式、またはフィルタクエリで文字列パラメータを使用するとき、パラメータがリテラル値を表す場合は、パラメータ参照を引用符で囲む必要があります。一重引用符または二重引用符を使用できます。この要件は、ネイティブ実行環境または Hadoop 実行環境で実行されるマッピング内の Hive ソースに対応します。

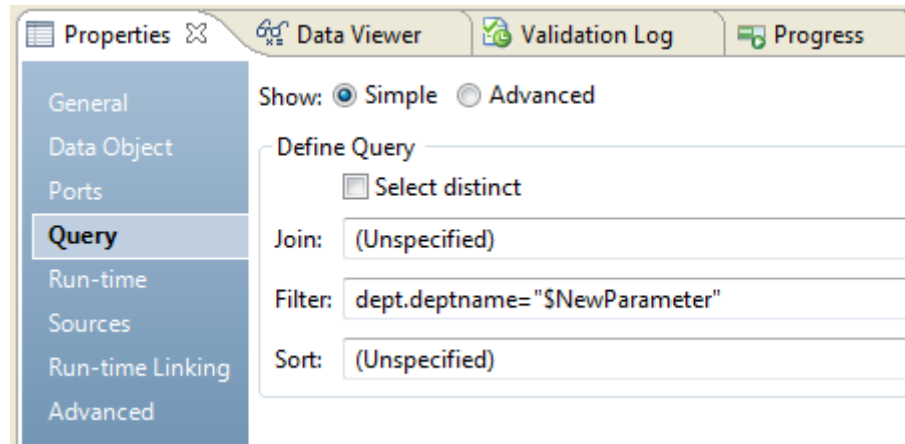
例えば、特定の部門名を含む Hive ソース行を選択するフィルタを作成する必要があるとします。部門名を表す文字列パラメータを作成します。部門名パラメータにデフォルト値の R&D を割り当てます。

次の図は、文字列パラメータを示しています。



Hive ソースのフィルタクエリでパラメータを使用する場合、パラメータ名を引用符で囲む必要があります。囲まなければ、実行時にマッピングが失敗し、SQL パーサーエラーが表示されます。

次の図は、【プロパティ】タブの【クエリ】ビューにある、Hive ソースのフィルタクエリを示しています。



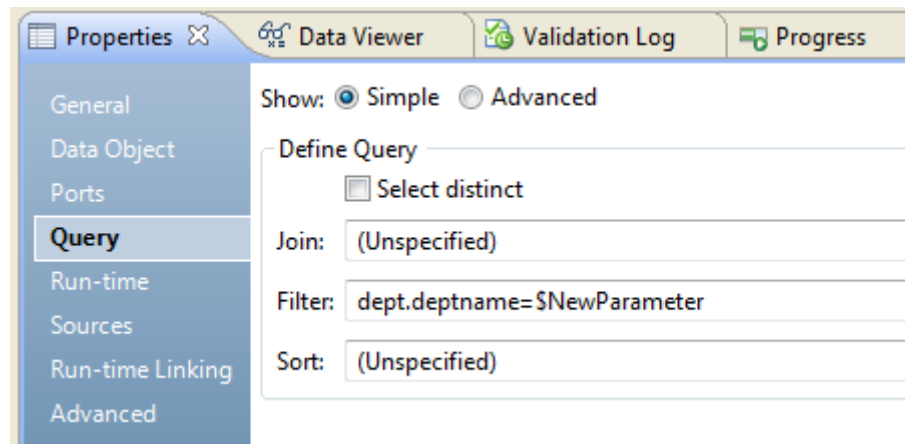
**注:** デフォルトでは、式エディタでパラメータが引用符で囲まれません。手動で囲む必要があります。

パラメータにカラム名またはサブクエリ名が含まれている場合は、パラメータ名を一重引用符または二重引用符で囲む必要はありません。

次の図は、デフォルト値がカラム名の文字列パラメータを示しています。

Properties					
Data Viewer					
Validation Log					
Progress					
General					
Parameters					
Outputs					
Run-time					
Load Order					
	Name	Type	Precision	Scale	Default Value
1	NewParameter	String	1000	0	dept.external_deptname

次の図は、パラメータを使用するフィルタクエリを示しています。



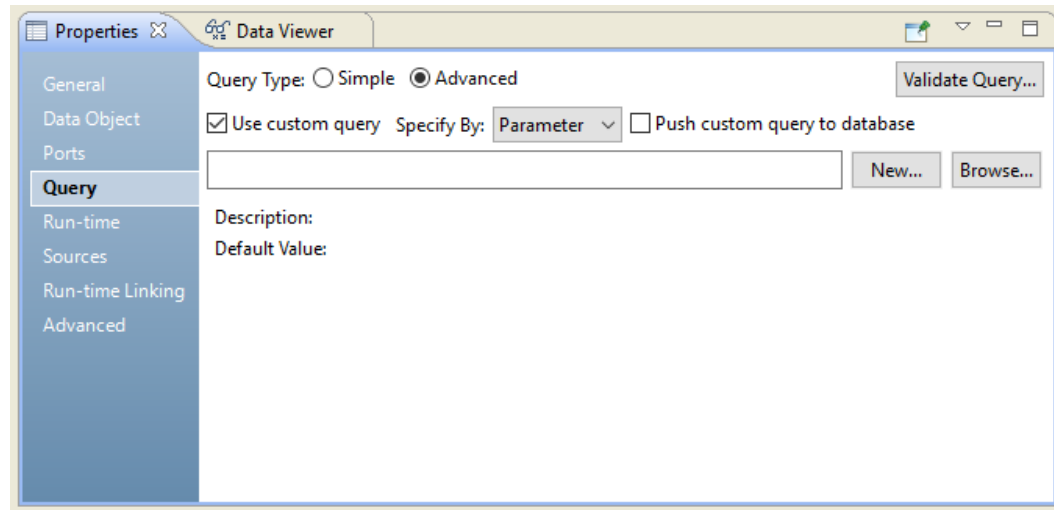


# リレーショナルソース用のカスタムクエリのパラメータ

カスタマイズされたデータオブジェクトの読み取り操作、または読み取りトランスフォーメーション内のリレーショナルデータオブジェクトのカスタムクエリにパラメータを割り当てることができます。

カスタムクエリを設定するときは、詳細クエリを作成して、パラメータを使用してクエリを指定する必要があります。

次の図は、カスタムクエリをパラメータ化できる場所を示しています。

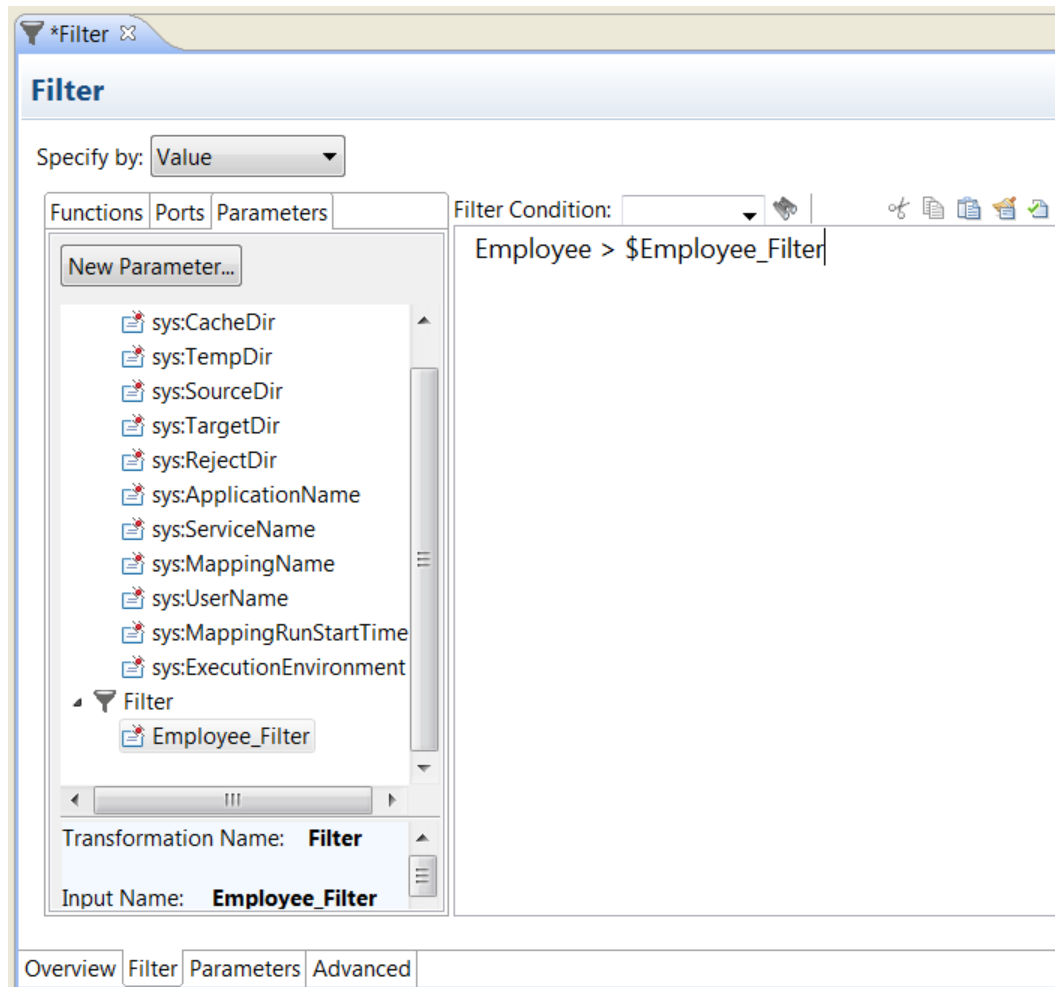


## 式のパラメータ

パラメータは、アグリゲータトランスフォーメーション、ルックアップ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、フィルタトランスフォーメーションなどのトランスフォーメーションの、式または条件内で設定できます。

例えば、フィルタトランスフォーメーションのフィルタ条件を設定します。条件に含めるポートとパラメータを選択します。フィルタ条件に含めるシステムパラメータまたはユーザー定義のパラメータを選択します。

次の図は、Employee ポートと Employee\_Filter パラメータを含むフィルタ条件を示しています。



ポート名を引数として受け付ける、同じ引数で式のパラメータを使用できます。パラメータを使用して式内の定数引数を置き換えることはできません。

例えば、string を decimal 値に変換する TO\_DECIMAL 式を考えてみます。

```
TO_DECIMAL( value [, scale] )
```

式内の scale 引数は定数値である必要があります。

次の有効な式には、scale の定数引数が含まれます。

```
TO_DECIMAL( Input_Port,10 )
```

次の式は有効ではありません。scale 引数に対してユーザー定義のパラメータが含まれているためです。

```
TO_DECIMAL( Input_Port,$Scale_Param )
```

パラメータに別のパラメータを含めることはできません。例えば、トランスフォーメーション内で Parameter1 と Parameter2 を設定した場合、Parameter1 のデフォルト値を \$Parameter2 に設定することはできません。パラメータをネストした場合、マッピングは実行時に検証エラーで失敗します。

# 式パラメータ

式のパラメータタイプを設定することができます。式パラメータとは、完全な式が含まれるパラメータです。次のトランスフォーメーションフィールドで式パラメータを使用できます。

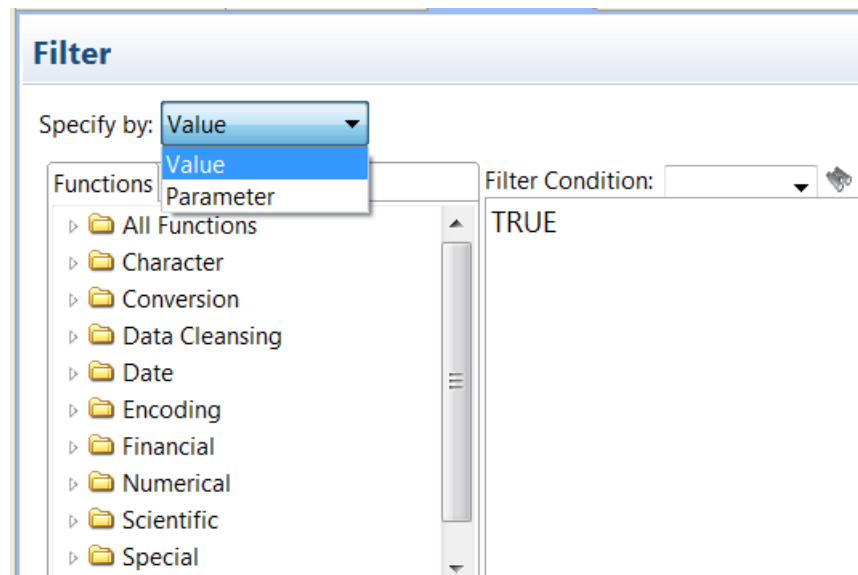
- アグリゲータ。ポート式。
- 式。ポート式。
- フィルタ。フィルタ条件。
- 結合。結合条件。
- ルックアップ。ルックアップ条件。
- ランク。ポート式。

フィルタ、結合、またはルックアップトランスフォーメーションで、フィルタ、結合、またはルックアップ条件の式パラメータを設定する場合、式パラメータが TRUE または FALSE として評価される必要があります。

アグリゲータ、式、またはランクトランスフォーメーションのポート式で式パラメータを設定する場合、ポートには式パラメータのみを含める必要があります。パラメータの結果値は、ポートのデータ型と一致している必要があります。たとえば、ポートのデータ型が文字列の場合、式は文字列として評価される必要があります。

式エディタで式パラメータを定義します。[指定元:] で **【パラメータ】** を選択し、完全な式をパラメータ化することを示します。

たとえば次の図は、[指定元:] の **【パラメータ】** オプションをフィルタ条件として示しています。



式パラメータを使用する場合、式パラメータを作成するか、既存の式パラメータを選択してトランスフォーメーションで使用することができます。式パラメータには、ポート、演算子、および定数を含めることができます。たとえば、動的マッピングで異なるポート名および演算子を指定して式パラメータを作成できます。式パラメータに他のパラメータを含めることはできません。

式パラメータがトランスフォーメーションパラメータである場合、トランスフォーメーションパラメータをマッピングパラメータにバインドできます。実行時にマッピングパラメータをオーバーライドできます。

## ポート式の制限

ポート式の式パラメータを使用する場合、次の制限を考慮してください。

- 式パラメータをマッピングパラメータとして作成する場合、パラメータがトランスフォーメーション内でポート式として使用されていない限り、パラメータのデフォルト値を検証できません。
- 式パラメータのデフォルト値が有効でない場合、マッピングは失敗します。
- 式パラメータの戻り値がポートのデータ型と一致しない場合、マッピングは失敗します。

## 式パラメータ例

このセクションには、トランスフォーメーションで式パラメータを使用する方法を示す例が含まれています。

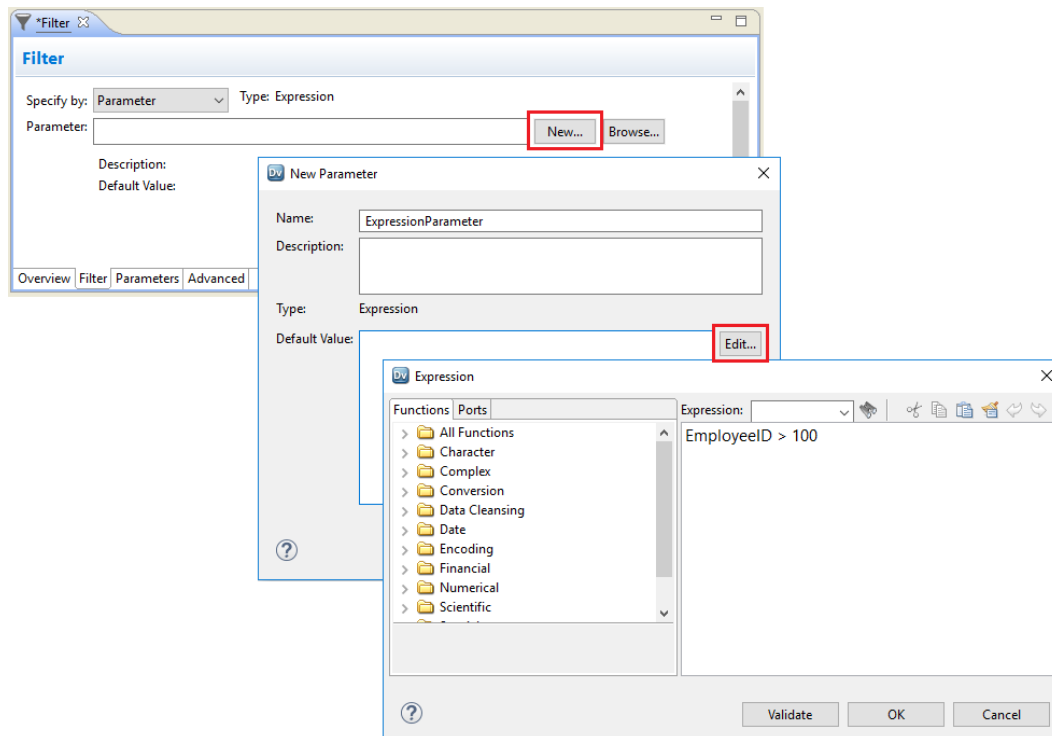
### 例.フィルタ条件内の式パラメータ

式パラメータを使用して、フィルタトランスフォーメーション内のフィルタ条件をパラメータ化します。

たとえば、次のデフォルト値を指定して、フィルタ条件の式パラメータを使用できます。

EmployeeID > 100

次の図は、式パラメータを設定できる場所を示しています。

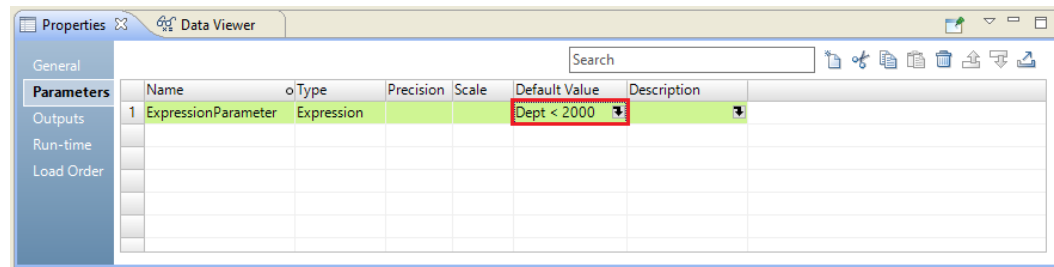


フィルタトランスフォーメーションが再利用可能である場合、トランスフォーメーションパラメータをマッピングパラメータにバインドできます。別のデフォルト値をマッピングパラメータに割り当てて、マッピング内の再利用不可能なトランスフォーメーションでマッピングパラメータを使用できます。

たとえば、マッピングパラメータを編集し、次のデフォルト値を設定できます。

Dept < 2000

次の図に、マッピングパラメータのデフォルト値を示します。



## 例.式トランスフォーメーション内の式パラメータ

式パラメータを使用して、式トランスフォーメーション内のポート式をパラメータ化します。

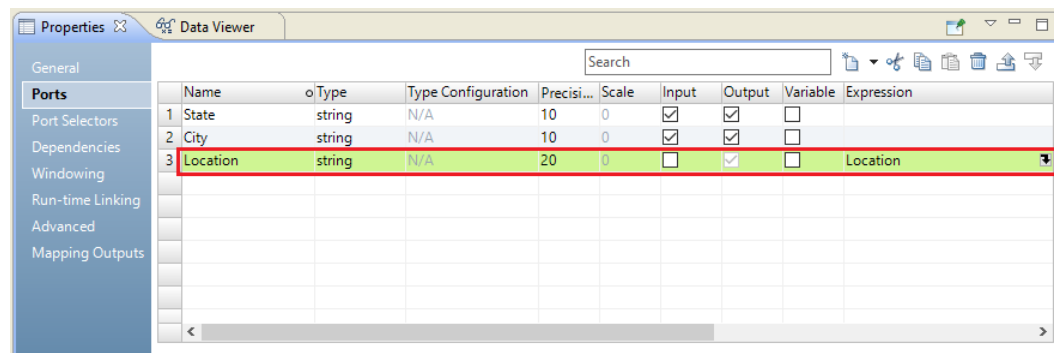
たとえば、式トランスフォーメーションを使用して都道府県と市区町村のデータを連結する場合があります。

次の表に、変換する可能性があるデータのリストを示します。

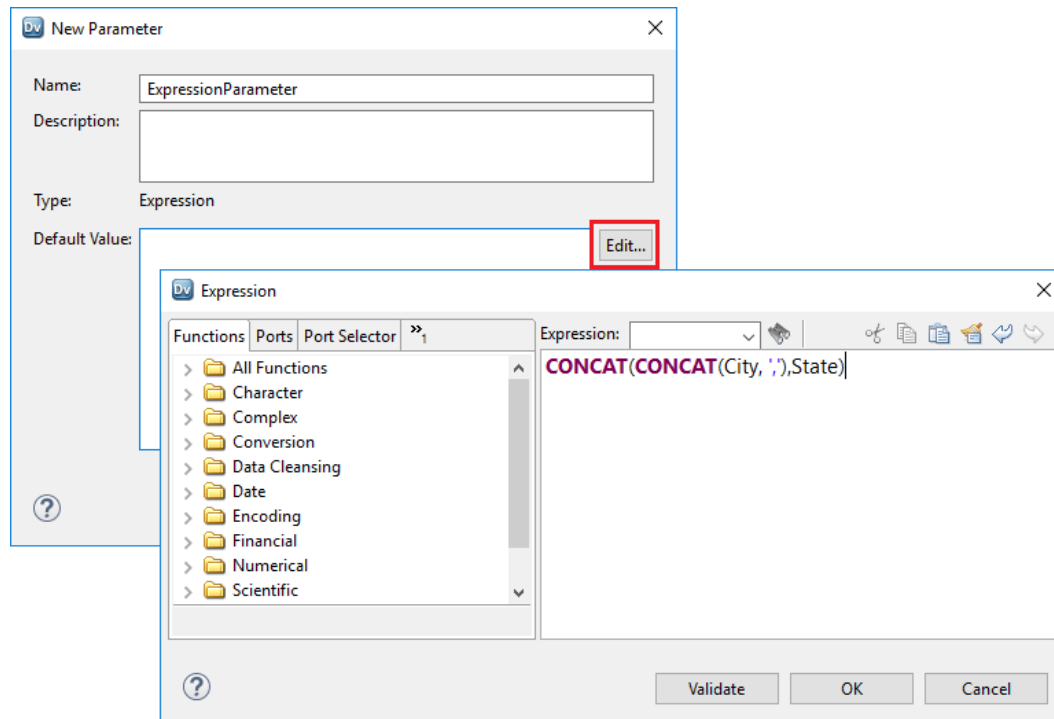
State	City
CA	Sacramento
TX	Austin
NY	Albany

データを連結するには、式 `CONCAT(CONCAT(City, ','),State)` を使用します。式トランスフォーメーションでは、出力ポートの Location を設定し、式パラメータをポート式に割り当てます。

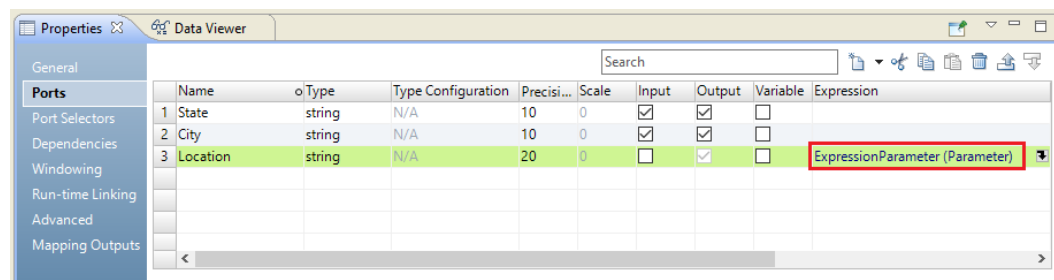
次の図は、式トランスフォーメーションでの出力ポートの Location を示しています。



次の図は、新しい式パラメータを設定できる場所を示しています。



次の図は、式パラメータが表示される場所を示しています。



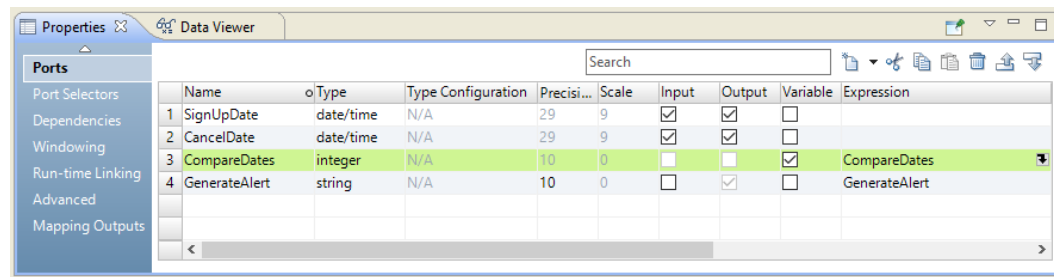
マッピングを実行すると、出力ポートは式パラメータを使用して、Sacramento, CA のように City, State の形式でデータを書き込みます。

## 例.ポート式内のネストしたパラメータ

式パラメータをネストさせるには、式パラメータを異なるポートに割り当てて、他の式パラメータ内でそのポートを使用します。

たとえば、顧客があなたの会社でアカウントを開設または解約した日付に関するデータがあるとします。データを調べて、システム内の日付が有効であることを確認する必要があります。日付が有効でない場合は警告を出したいと考えています。

データを調べるには、式トランスフォーメーションを作成し、次のポートを設定します。



入力ポート SignUpDate は、顧客がアカウントを開設した日付です。入力ポート CancelDate は、顧客がアカウントを解約した日付です。

変数ポート CompareDates は、ポート CancelDate の値がポート SignUpDate よりも早いかに遅いかに応じて、-1、0、または 1 を返します。ポート GenerateAlert は、ポート CompareDates の値が 1 である場合、「警告」を返します。

変数ポート CompareDates には、次のデフォルト値を使用して式パラメータを設定できます。

`DATE_COMPARE(SignUpDate, CancelDate)`

変数ポート CompareDates に使用する式パラメータを出力ポート GenerateAlert の式パラメータ内にネストさせるには、次のデフォルト値を使用して式パラメータを設定します。

`IIF(CompareDates=1, 'Warning')`

## フィールドおよびプロパティ値のパラメータ

トランスフォーメーションおよび物理データオブジェクトで、いくつかのフィールドおよびプロパティの値のパラメータを設定できます。

リレーショナルデータオブジェクト、カスタマイズデータオブジェクト、およびルックアップトランスフォーメーションの接続名を設定できます。フラットファイルデータオブジェクトでは、入力ファイルディレクトリ、出力ファイルディレクトリ、および拒否ファイルディレクトリのパラメータを設定できます。フラットファイル区切り文字のタイプを変更するように、パラメータを設定することもできます。

次の図は、物理データオブジェクトの【詳細】タブにあるフラットファイル区切り文字のパラメータを示しています。

**Advanced**

**Format**

Code page: MS Windows Latin 1 (ANSI), superset of Latin1

Format

☒ Delimited (fields separated by delimiters)

☐ Fixed-width (fields aligned in columns)

Delimiters

☐ Use Fixed Values

☐ Tab ☐ Semicolon ☐ Comma ☐ Space ☐ Other:  ...

☒ Assign Parameter

...

Text qualifier

☒ No quotes ☐ Single quotes ☐ Double quotes

## リレーショナルテーブルリソースのパラメータ

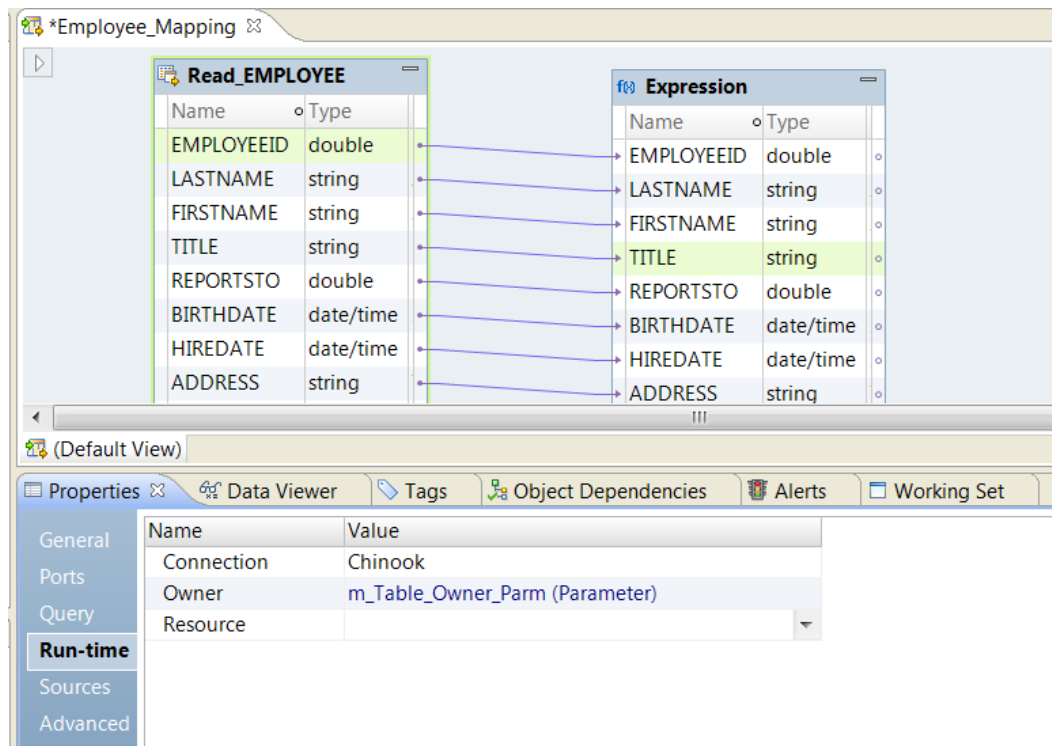
読み取りトランスフォーメーションのリソース名、テーブル所有者、および接続をパラメータ化できます。リソースは、リレーショナルデータオブジェクトのテーブル、ビュー、またはシノニムの名前です。

動的マッピングで同じデータベースの複数のテーブルを処理する必要がある場合、リソース名をパラメータ化できます。

マッピングで読み取りトランスフォーメーションを選択します。【プロパティ】ビューの【ランタイム】タブで、【値】カラムをクリックし、接続、テーブル所有者、またはリソースのパラメータを割り当てます。

次の図は、読み取りトランスフォーメーションの接続、リソース名、およびテーブル所有者のパラメータを割り当てる場所を示しています。

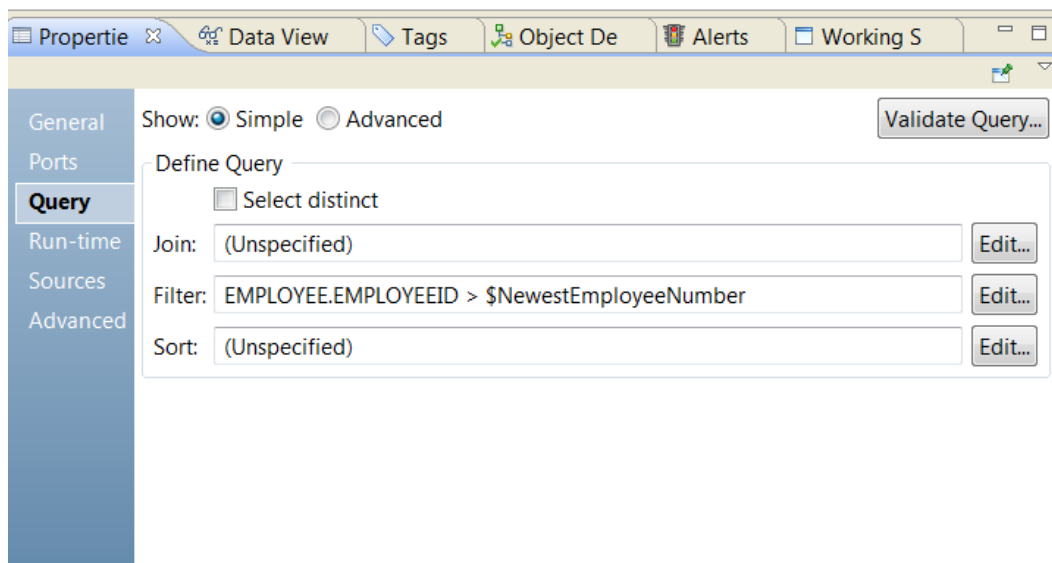




## SQL 文のパラメータ

リレーショナルデータオブジェクト、カスタマイズされたデータオブジェクトに、またはリレーショナルデータオブジェクトやカスタマイズされたデータオブジェクトを使用する読み取りおよびルックアップトランスフォーメーションに追加する SQL 文にパラメータを含めることができます。

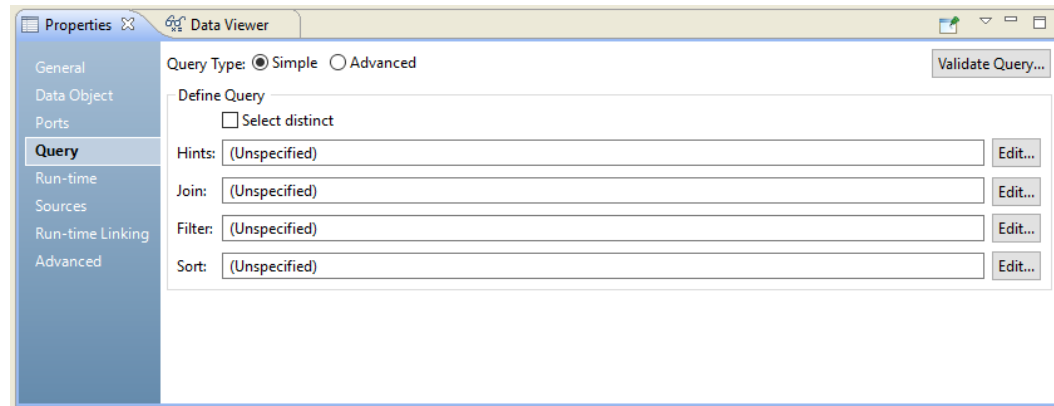
次の図に、リレーショナルソースを読み取る SQL クエリをパラメータ化する方法を示します。



## フィルタおよび結合条件内のパラメータ

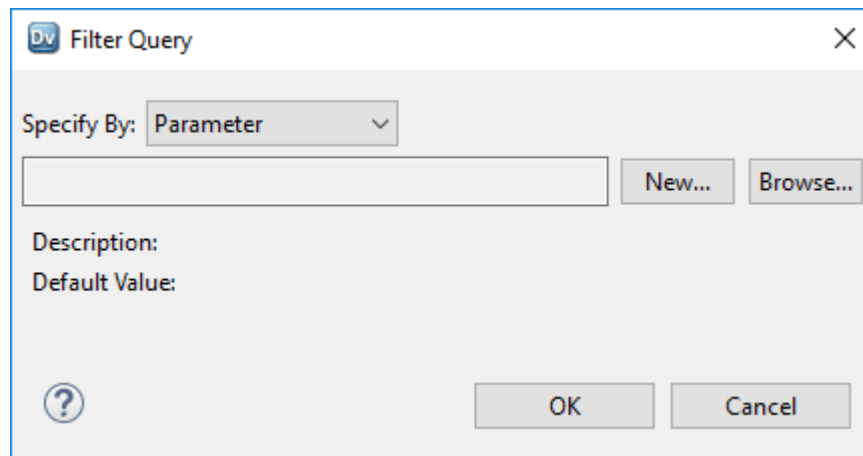
フィルタまたは結合条件を追加する SQL 文にパラメータを割り当てることができます。カスタマイズされたデータオブジェクトの読み取り操作、または読み取りトランスフォーメーション内のリレーショナルデータオブジェクトにパラメータを割り当てることができます。

次の図は、フィルタ条件または結合条件に SQL パラメータを割り当てることができる場所を示しています。



結合またはフィルタ条件を編集するときは、パラメータを使用して条件を指定できます。

例えば次の図は、フィルタ条件を編集するときにパラメータを SQL 文に割り当てのために使用できるダイアログボックスを示しています。



## SQL 文の文字列パラメータ

SQL 文に文字列パラメータを定義するときに、クエリ内のパラメータに単一引用符 ( ' ) を追加する必要があります。クエリではなくデフォルト値に単一引用符が含まれる場合、データ統合サービスでは、単一引用符で囲まれた各パラメータ内のデータをエスケープし、デフォルト値に含まれる単一引用符それぞれに対して、追加で単一引用符を付加します。

### 例 - クエリ内の単一引用符

例として、\$date\_parm という日付パラメータを持つ SQL 文を取り上げます。SQL 文は、次の式として表されます。

```
select * from <table_name> where <date_port> > $date_parm
```

パラメータ\$date\_parm は文字列のため、クエリのパラメータに単一引用符を追加します。次の式は、パラメータに単一引用符が追加されたクエリを示しています。

```
select * from <table_name> where <date_port> > '$date_parm'
```

次の式は、パラメータ\$date\_parm のデフォルト値が 01/31/2000 00:00:00 の場合に、データ統合サービスでクエリがどのように展開されるかを表しています。

```
select * from <table_name> where <date_port> > '01/31/2000 00:00:00'
```

### 例 - デフォルト値内の単一引用符

日付パラメータ\$date\_parm を使用した同じ SQL 文を使用します。SQL 文は、次の式として表されます。

```
select * from <table_name> where <date_port> > $date_parm
```

デフォルト値'01/31/2000 00:00:00'に単一引用符を追加します。次の式は、展開後のクエリを表します。

```
select * from <table_name> where <date_port> > ''01/31/2000 00:00:00''
```

デフォルト値に単一引用符が含まれているため、データ統合サービスは、追加の単一引用符を使用してデータをエスケープします。展開後のクエリの文字列パラメータに二重引用符が含まれるため、クエリは失敗するか、結果が生成されない可能性があります。

## SQL 文でのパラメータの使用のヒント

ヒントを活用して、SQL 文のパラメータをより効率的に使用できます。

- 文字列パラメータの場合、クエリ内でパラメータを定義するときに単一引用符を使用します。
- パラメータが文字列でない場合は、パラメータに単一引用符を使用しないでください。予期しない結果になる可能性もあります。
- 文字列パラメータの場合、SQL 文とパラメータのデフォルト値に単一引用符がない場合は、マッピングにアップストリームのフィルタトランスフォーメーションを追加できます。フィルタトランスフォーメーションで、パラメータを単一引用符で囲むようにフィルタ条件を編集します。
- パラメータ名にピリオド (.) を含めることはできません。ピリオドを含むパラメータを持つ SQL クエリは無効です。例えば、以下の SQL 文のパラメータ名には、ピリオドが含まれます。

```
SELECT $tname.ID,"MY_SOURCE"."NAME" FROM "MY_SOURCE" where FIELDX=1
```

このクエリを検証すると、データ統合サービスは、tname.ID パラメータを見つけることができないというエラーを返します。

## ポートリストのパラメータ

ポートのリストを含むパラメータを作成できます。このパラメータを、ソータートランスフォーメーション、ランクトランスフォーメーション、ジョイナー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などの各種トランスフォーメーションで参照できます。

複数のポート名が含まれる以下のタイプのパラメータを設定できます。

### ポートリスト

カンマ区切りのポート名のリスト。ポートリストパラメータの構文は次のとおりです。Port1,Port2,Port3

### ソートリスト

ポート名のリスト、および各ポートのソートタイプ。ソートリストパラメータの構文は次のとおりです。

Port1:A,Port2:A,Port3:D

### 入力リンクセット

実行時にリンクするポートのセット。リンクセットのパラメータには名前と値のペアが含まれます。このペアの構文は次のとおりです。Port1>:=Port2, Port3>:=Port4

## 第 5 章

# マッピング出力

この章では、以下の項目について説明します。

- [マッピング出力の概要, 101 ページ](#)
- [ユーザー定義マッピング出力, 102 ページ](#)
- [システム定義のマッピング出力, 104 ページ](#)
- [デプロイ済みマッピングでのマッピング出力の保持とバインド, 105 ページ](#)
- [マップレットのマッピング出力, 110 ページ](#)
- [論理データオブジェクトのマッピング出力, 114 ページ](#)
- [マップレット出力をマッピング出力にバインドする方法, 114 ページ](#)

## マッピング出力の概要

マッピングはマッピング出力を返すことができます。マッピング出力は、マッピングで処理する各行のフィールドまたは式を集計した結果である単一値です。

マッピング出力は、マッピングの実行に関する情報を提供する値を返します。例えば、マッピング出力ではマッピングで見つかったエラー行の数を返すことができます。マッピング出力では、マッピングで処理された最後の注文日、およびすべての注文の合計数を返すことができます。

トランスフォーメーションはマッピング出力値を受け取りません。マッピングは、マッピングが完了すると各マッピング値を返します。複数のマッピング出力を同一のマッピングに定義することができます。

マッピングでは、ユーザー定義マッピング出力またはシステム定義マッピング出力を返すことができます。

### ユーザー定義マッピング出力

ユーザー定義マッピング出力は、マッピングの各行のフィールドまたは式を集計することによってマッピングが返す数値または日付です。例えば、注文が特定のしきい値に達したときを認識する必要がある場合があります。マッピングが処理した注文の合計数を返すように、マッピングを設定できます。

TotalOrderAmt というマッピング出力を定義して、すべての行の Order\_Amount フィールドを集計するようにマッピングを設定します。式またはポート名を定義して、式トランスフォーメーションで集計します。

### システム定義のマッピング出力

システム定義のマッピング出力は、マッピングの完了時にマッピングが必ず返す組み込み値です。マッピングは、マッピングが処理したソース行数、ターゲット行数、およびエラー行数を返します。ワークフロー変数のこれらの値はワークフローの別のタスク（通知タスク、排他的なゲートウェイタスクなど）に渡すことができます。システム定義のマッピング出力を定義する必要はありません。

ワークフロー内のマッピングタスクとしてマッピングを実行する場合、マッピング出力をリポジトリ内に保持したり、マッピング出力をワークフローの変数にバインドしたりできます。

マッピングタスク内のマッピング出力については、『*Informatica Developer ワークフローガイド*』の「マッピングタスク」の章を参照してください。

## ユーザー定義マッピング出力

ユーザー定義マッピング出力は、マッピングの各行のフィールドまたは式を集計することによってマッピングが返す数値または日付です。集計する式、および結果値のデータ型を定義します。

[プロパティ] ビューの【出力】タブでマッピング出力を定義します。マッピング出力の名前および結果のデータ型を設定し、結果を返すために実行する集計のタイプを指定します。

マッピング出力名および出力タイプを定義したら、マッピングに式トランスフォーメーションを設定します。式トランスフォーメーションで、集計する出力式を定義します。式にはポート名を含めるか、ポート、関数、およびパラメータを持つ式を含めることができます。

### [出力] ビュー

マッピング出力は、マッピングの【プロパティ】の【出力】ビューで定義します。各マッピング出力を定義するときに、マッピング出力名、マッピング出力タイプ、および実行する集計のタイプを入力します。

次の図は、マッピングの【プロパティ】ビューの【出力】タブのマッピング出力を示しています。

Name	Type	Precisi...	Scale	Aggregation Type	Binding	Description
Total_OrderAmt	decimal	10	2	SUM		Add total order amounts
Last_Order_Date	date/time	29	9	MAX		Latest date processed

【出力】ビューには以下のフィールドがあります。

#### 名前

出力の名前。デフォルトは「Output」です。

#### タイプ

マッピング出力のタイプ。numeric タイプまたは date/time タイプを選択できます。デフォルトは Integer です。

### 精度

マッピング出力フィールドの長さ。デフォルトは 10 です。

### スケール

マッピング出力フィールドの小数点の右側の桁数。デフォルトはゼロです。

### 集計タイプ

以下のいずれかの集計タイプを選択できます。

#### SUM

データ統合サービスが処理した各入力行のフィールドまたは式の合計を返します。

#### MIN

データ統合サービスが各入力行の特定のフィールドまたは式から処理した最小の数値または日付を返します。

#### MAX

データ統合サービスが各入力行の特定のフィールドまたは式から処理した最大の数値または日付を返します。

### バインディング

マッピング出力にバインドするマップレットまたは論理データオブジェクトの出力の名前。マッピング出力がマッピングの式トランスフォーメーションからではなくマップレットから返される場合を除き、このフィールドは空白です。

### 説明

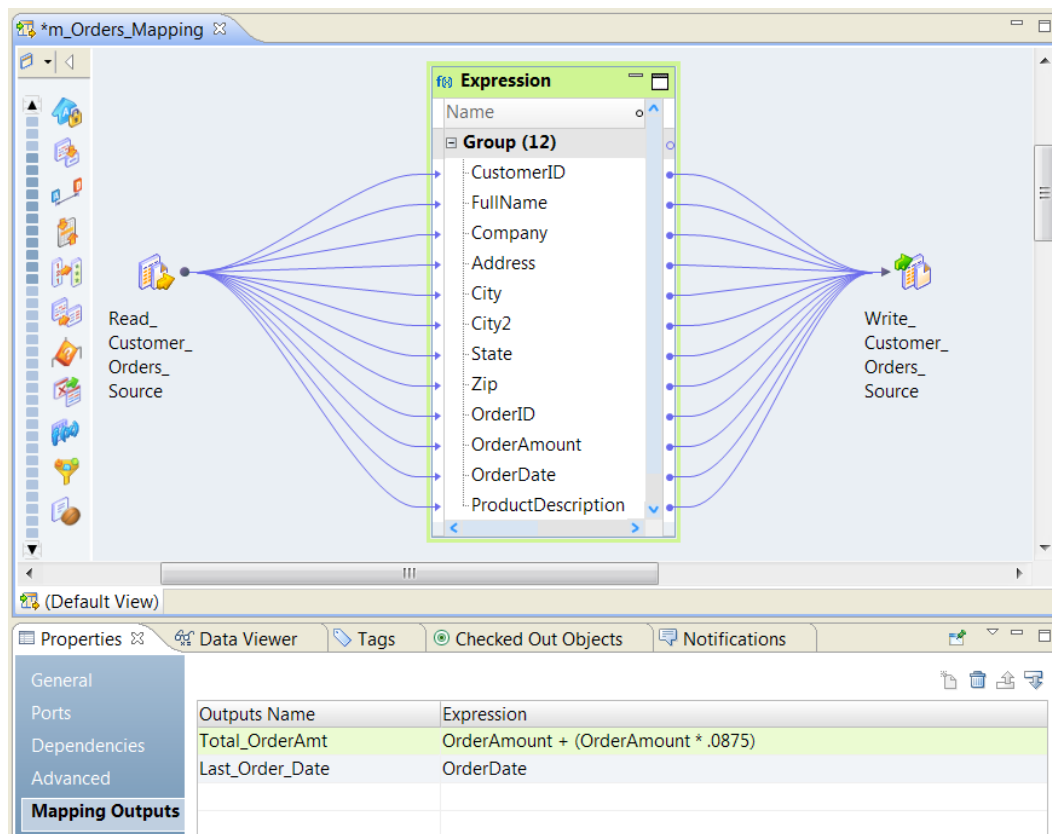
マッピング出力の説明。

## マッピング出力式

マッピング出力式は、式トランスフォーメーションの【マッピング出力】ビューで設定します。マッピング出力式は、式トランスフォーメーションが受け取る行から集計するフィールドまたは式です。

マッピングに式トランスフォーメーションを設定して、集計する出力式を含めます。マッピングにフィルタまたはアクティブなトランスフォーメーションが含まれているかどうかによって、パイプラインでの式トランスフォーメーションの位置がマッピング出力の結果に影響するかどうかが決まります。異なるパイプラインの行を集計する必要がある場合は、複数の式トランスフォーメーションをマッピングに追加できます。

次の図は、式トランスフォーメーションの【マッピング出力】ビューの式を示しています。



【マッピング出力】ビューには以下のフィールドがあります。

#### 出力名

マッピングレベルで作成したマッピング出力の名前。マッピング出力は最初にマッピングレベルで作成する必要があります。マッピング出力を式トランスフォーメーションに追加する場合は、すでに作成した出力のリストから出力名を選択します。

#### 式

マッピングの各行を集計するための式。式エディタにポート名または式を入力します。式の結果は数値または日付である必要があります。式にパラメータを使用できます。データ統合サービスは、式トランスフォーメーションが受け取る各行に式を適用します。各マッピング出力は、マッピングが完了すると1つの値を返します。

**注:** 式トランスフォーメーションには、実行する集計のタイプは指定しません。マッピングが各行を処理する際に集計するフィールドまたは式を示します。

## システム定義のマッピング出力

システム定義のマッピング出力は、各マッピングで生成されるマッピング出力です。システム定義のマッピング出力には集計を設定する必要はありません。

マッピングは、次のタイプのシステム定義マッピング出力を返します。

numberOfTargetRows

マッピングがターゲットに書き込んだ行の数。



numberOfSourceRows

マッピングがソースから読み取った行の数。

numberOfErrorRows

マッピングによって生成されたエラー行の数。

システム定義のマッピング出力をワークフロー変数にバインドできます。

ワークフロー内のマッピング出力については、『*Informatica Developer ワークフローガイド*』の「マッピングタスク」の章を参照してください。

## デプロイ済みマッピングでのマッピング出力の保持とバインド

マッピング出力を保持して出力値をリポジトリに保存します。ネイティブ環境で、または Spark エンジン上で実行するようにデプロイされたマッピングの場合は、出力をマッピングパラメータにバインドすると、同じマッピングの後続の実行で値を使用できます。

マッピング出力を保持してバインドする場合は、読み取りトランスフォーメーション用のフィルタクエリとマッピング出力式を、式トランスフォーメーションで定義します。フィルタクエリは、保持されている出力値と照らして新しいデータをチェックします。マッピング出力式は、マッピング用に出力値を集計します。

マッピングパラメータを定義するときは、パラメータにデフォルト値を割り当てます。同じパラメータの値をパラメータセットまたはパラメータファイルで設定する場合を除き、マッピング出力をマッピングパラメータにバインドすると、保持されている出力によってデフォルト値がオーバーライドされます。パラメータセットまたはパラメータファイルでパラメータ値を定義する場合は、その値がパラメータに使用されます。

マッピング出力値を保持するには、ランタイムインスタンス名を設定する必要があります。ランタイムインスタンス名を設定するには、`infacmd ms runMapping` コマンドの `-RuntimeInstanceName` オプションを使用します。詳細については、『*Informatica コマンドリファレンス*』を参照してください。

バインディングのユースケースの 1 つに、マッピングが処理する最新の ID 番号の保持があります。マッピングを次回実行するときに、マッピングパラメータは最後に処理された ID の値を使用します。マッピングでは、最後に処理された注文日より後の注文日を持つ行のみを含めるように、パラメータソース行をフィルタ処理できます。

## マッピング出力の設定およびバインド方法

マッピング出力を設定するときは、マッピング出力をマッピングレベルで定義し、集計する式をトランスフォーメーションレベルで設定します。保持されている出力をマッピングパラメータにバインドします。

デプロイ済みのマッピングでマッピング出力を設定してバインドするには、次の手順を実行します。

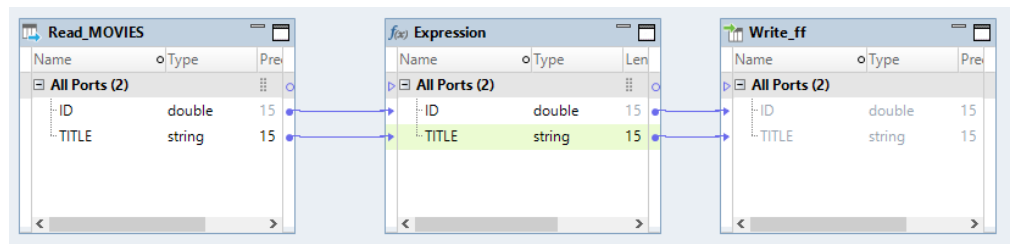
1. 式トランスフォーメーションを使用してマッピングを作成し、マッピングパラメータを設定します。
2. リポジトリに保持されるようにマッピング出力を定義して、出力の計算に使用する集計タイプを設定します。
3. マッピングが処理する各行の出力を集計するマッピング出力式を設定します。
4. パラメータ化されたフィールドおよびその値に基づいてデータソースからデータの差分を読み取るフィルタクエリを、読み取りトランスフォーメーションで定義します。
5. 保持されている値を後続のマッピング実行で使用するために、マッピング出力をパラメータにバインドします。

## 手順 1. マッピングの作成とマッピングパラメータの設定

ローカルのフラットファイルターゲットに対して書き込みを行う式トランスフォーメーションを使用してマッピングを作成します。マッピング用の入力パラメータを定義します。

1. MOVIES ファイルのデータを処理するマッピングを作成します。読み取りトランスフォーメーションを使用してデータソースから読み取ります。
2. ソースの直後に式トランスフォーメーションを含めます。式トランスフォーメーションには、出力値を集計する式が含まれています。
3. 書き込みトランスフォーメーションを使用して、ローカルのフラットファイルターゲットに結果を書き込みます。

次の図に、マッピングを示します。



4. エディタをクリックしてマッピングプロパティにアクセスします。
5. [パラメータ] ビューを開きます。
6. col\_id\_val という保持されたマッピング出力値を受け入れるパラメータを作成します。パラメータのデフォルト値を設定します。
7. col\_id\_name というソーステーブル内の ID カラムの名前用にパラメータを作成します。

次の図は、マッピングパラメータを示しています。

The screenshot shows the mapping tool interface with the 'Parameters' view selected. The 'Parameters' tab is active, displaying a table of parameters. The table has columns: Name, Type, Precision, Scale, Default Value, and Description. The parameters are listed as follows:

Name	Type	Precision	Scale	Default Value	Description
col_id_val	Double	15	0	0.0	
col_id_name	String	1000	0	id2	

8. col\_id\_name パラメータを含むパラメータセットを作成します。パラメータの値を MOVIES.ID に設定します。

## 手順 2. マッピング出力と集計タイプの定義

マッピング出力は、マッピングの [プロパティ] の [出力] タブで定義します。各マッピング出力の定義では、実行する集計のタイプ、および結果のデータ型を記述します。[出力] カラムには、ユーザー定義とシステム定義の両方のマッピング出力が含まれます。

1. マッピングを作成したら、エディタをクリックして、マッピングプロパティにアクセスします。
2. [出力] ビューを開きます。
3. 新しいマッピング出力を作成します。

Developer tool がマッピング出力を作成して、デフォルトのフィールドの値を設定します。

4. マッピング出力を識別する名前を Output\_ID に変更します。
5. 数値のマッピング出力タイプを選択します。精度およびスケールを入力します。
6. マッピング出力の集計タイプを MAX として選択し、処理済みの最大のオーダー ID を保存します。  
集計タイプを選択するときは、マッピングが処理する値の合計を計算するか、最小または最大の値を求めるかを選択できます。デフォルトは SUM です。  
**注:** マッピング出力を保持する場合は、以前に保持した値と新しく計算した値を比較する際にも集計タイプが使用されます。
7. **【ファイル】 > 【保存】** をクリックして、マッピング出力を保存します。  
式トランスフォーメーションにマッピング出力式を作成する前に、マッピング出力を保存する必要があります。

次の図は、マッピング出力 Output\_ID を示しています。

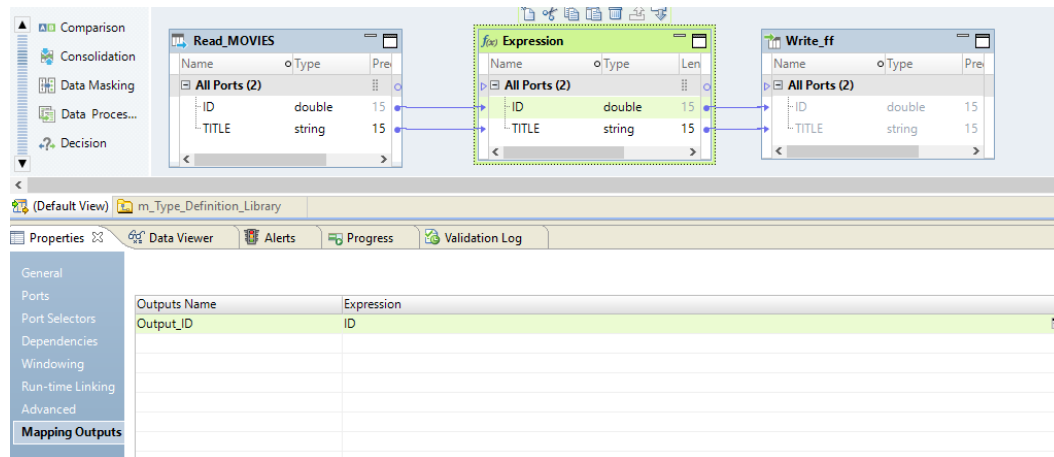
Properties Data Viewer Alerts Progress Validation Log							
General Parameters Outputs Run-time Load Order							
Name	Type	Precision	Scale	Aggregation Type	Binding	Description	
Output_ID	double	15	0	MAX			

### 手順 3.式トランスフォーメーションの設定

式トランスフォーメーションで、マッピングが処理する各行を集計する式を設定します。

1. マッピングエディタで式トランスフォーメーションを選択し、トランスフォーメーションのプロパティを表示します。
2. **【マッピング出力】** ビューを開きます。
3. マッピング出力式を作成します。  
Developer tool が、マッピングレベルで作成したいいずれかのマッピング出力と一致する出力名で出力式を作成します。マッピングのプロパティに複数のマッピング出力がある場合は、使用する適切なマッピング出力名を選択します。
4. **【式】** カラムをクリックして、**【式エディタ】** で式を入力します。  
式には、ポート名のみを含めるか、関数、ポート、およびパラメータを含めることができます。式にパラメータが含まれている場合、パラメータ値はパラメータファイルまたはパラメータセットで設定できます。
5. **【検証】** をクリックして式が有効であることを確認します。
6. **【OK】** をクリックして式を保存します。  
式がマッピング出力式の **【式】** カラムに表示されます。
7. **【ファイル】 > 【保存】** をクリックして、式トランスフォーメーションを保存します。

次の図は、保存されたマッピング出力式を示しています。式ではポートの「ID」のみを使用してマッピング出力を計算します。



## 手順 4.フィルタクエリの定義

読み取りトランスフォーメーションで、パラメータ化されたフィールドおよびその値に基づいてデータの差分を読み取るフィルタクエリを設定します。

1. マッピングエディタで読み取りトランスフォーメーションを選択し、トランスフォーメーションのプロパティを表示します。
2. **【クエリ】** ビューを開きます。
3. **【フィルタ】** プロパティの **【編集】** ボタンを選択します。
4. 保持されている値より大きい値を検索するフィルタを、次の形式を使用して設定します。\$<入力ポートのパラメータ>>\$<保持されている値のパラメータ>。

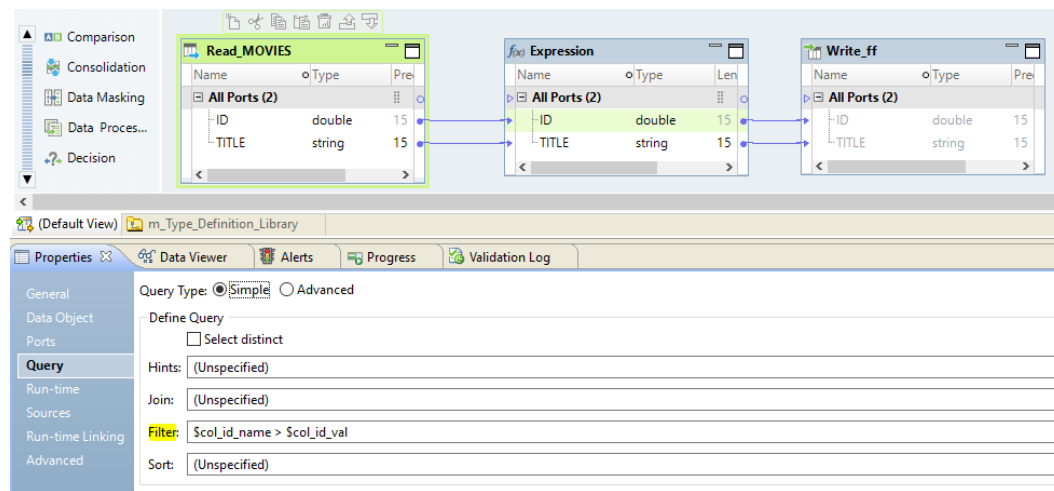
入力ポートのパラメータは、ソースリレーショナルテーブルの ID カラムのマッピングパラメータ col\_id\_name です。

保持されている値のパラメータは、保持されているマッピング出力値とバインドするマッピングパラメータ col\_id\_val です。

パラメータセットまたはパラメータファイルで col\_id\_val パラメータに値を割り当てると、クエリはその割り当て値をパラメータに使用します。それ以外の場合は、保持されている値がある場合はそれを使用し、ない場合はデフォルト値を使用します。

5. **【OK】** をクリックしてフィルタクエリを保存します。  
フィルタクエリが **【フィルタ】** プロパティに表示されます。
6. **【ファイル】** > **【保存】** をクリックして、読み取りトランスフォーメーションを保存します。

次の図は、設定されたフィルタクエリを示しています。

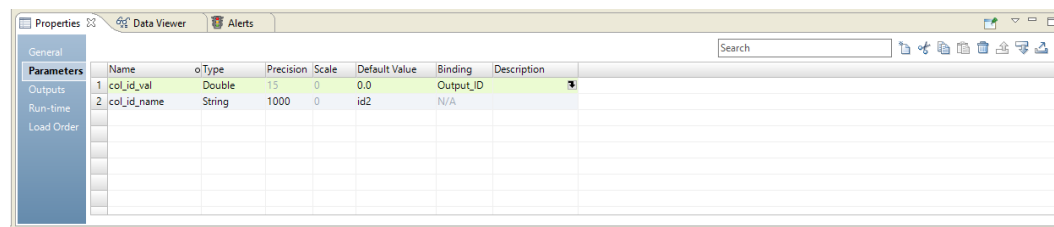


## 手順 5. マッピング出力のマッピングパラメータへのバインド

マッピング出力を設定済みのマッピングパラメータにバインドします。

1. エディタをクリックしてマッピングプロパティにアクセスします。
2. **【パラメータ】** ビューを開きます。
3. 保持されているマッピング出力を受け入れるように定義したパラメータ col\_id\_val に関して、**【バインディング】** カラムのドロップダウンメニューを使用してマッピング出力 Output\_ID を選択します。

その後マッピングを実行するときに、Output\_ID の保持されている値がマッピングパラメータ col\_id\_val に渡されます。次の図は、マッピングパラメータを示しています。



## 保持されている出力のメンテナンス

リポジトリ内の保持されているマッピング出力を一覧表示および削除できます。

保持されているマッピング出力値に対して、次の infacmd ms コマンドを実行できます。

listMappingPersistedOutputs

デプロイ済みのマッピングについて保持されているマッピング出力を一覧表示します。出力はアプリケーション名とマッピングのランタイムインスタンス名に基づいて一覧表示されます。

deleteMappingPersistedOutputs

デプロイ済みのマッピングについて保持されているマッピング出力をすべて削除します。アプリケーション名とマッピングのランタイムインスタンス名を使用して削除する出力を指定してください。特定の出力を削除するには、-OutputNamesToDelete オプションを使用してください。

infacmd の詳細については、『*Informatica®* コマンドリファレンス』の「infacmd ms コマンドリファレンス」の章を参照してください。

## 保持されているマッピング出力およびデプロイメントのルールとガイドライン

マッピングを再デプロイするか、マッピング出力を変更すると、保持されたマッピング出力の状態に影響することがあります。

保持されたマッピング出力のルールおよびガイドラインは、次のとおりです。

- 保持されたマッピング出力がマッピングに含まれている場合は、マッピングを初めてデプロイするときに、追加のタスクを実行する必要がありません。
- マッピングを再デプロイするときに、マッピング出力がリポジトリ内に保持されます。infacmd ms deleteMappingPersistedOutputs を実行すると、保持された出力をリポジトリから削除できます。
- マッピングをバックアップするときに、マッピング出力の状態はバックアップされません。
- マッピング出力を名前変更した場合は、以前にデプロイされたマッピングの実行で保持されたマッピング出力値を使用できません。マッピング出力を同じ名前で再作成した場合は、保持された出力値を使用できます。

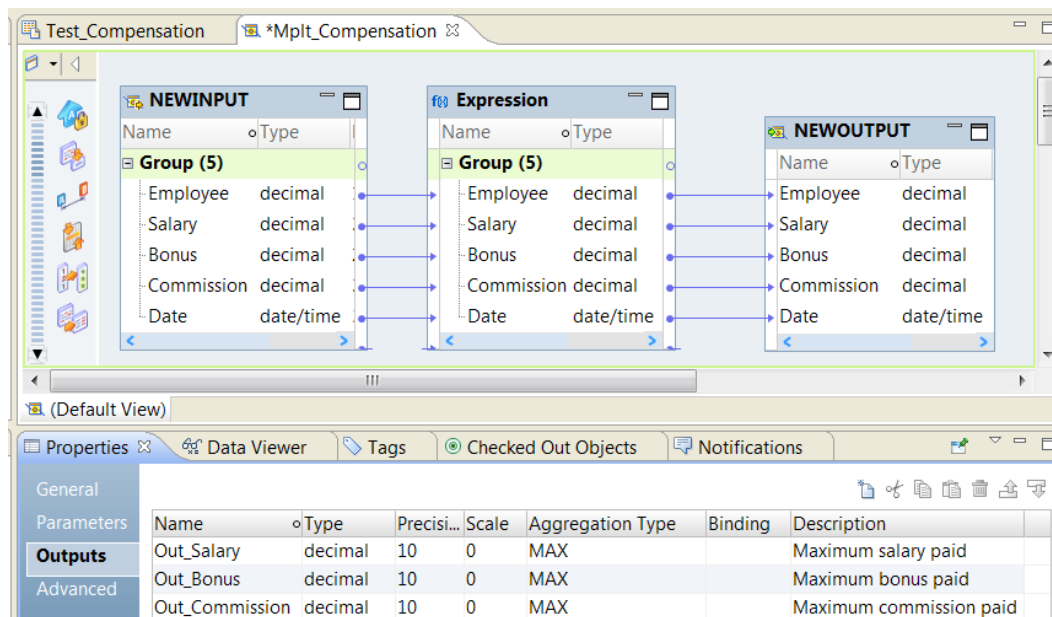
## マップレットのマッピング出力

マッピング出力を返すようにマップレットを設定できます。マップレットのマッピング出力を、マッピング出力にマッピングレベルでバインドできます。

マップレットをマッピングに含めると、マップレットは出力の値を計算して、出力値をマッピングに渡します。マップレットの複数の出力を、同一の出力にマッピングレベルでバインドできます。マップレットのシステム定義の出力をマッピング出力にバインドすることもできます。マップレット出力とマッピング出力は、同じタイプである必要があります。

例えば、マップレットは、Salary ポート、Bonus ポート、および Commission ポートの最大値を 3 つのマッピング出力として返すことができます。

次の図は、**[出力]** ビューの Out\_Salary、Out\_Bonus、および Out\_Commission マッピング出力を示しています。



【出力】ビューには以下のフィールドがあります。

#### 名前

出力の名前。デフォルトは「Output」です。

#### タイプ

マッピング出力のタイプ。numeric タイプまたは date/time タイプを選択できます。デフォルトは Integer です。

#### 精度

マッピング出力フィールドの長さ。

#### スケール

マッピング出力フィールドの小数点の右側の桁数。

#### 集計タイプ

出力式で実行される集計のタイプ。SUM、MIN、または MAX を選択します。デフォルトは SUM です。

#### バインディング

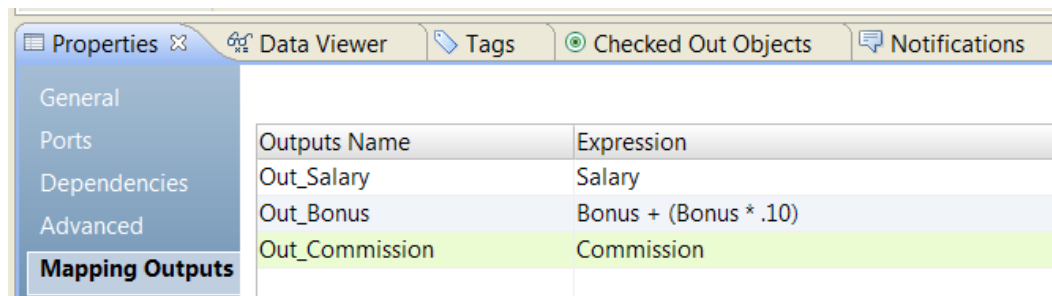
このマッピング出力にバインドする別のマップレットの出力の名前。マップレット内にマッピング出力を返す別のマップレットが含まれている場合を除き、【バインディング】フィールドは空です。

#### 説明

マッピング出力の説明。

マップレット内の各マッピング出力について、関連付けられる出力式を式トランスフォーメーションに作成します。各式では集計するフィールドを指定します。

次の図は、式トランスフォーメーションのマッピング出力式を示しています。



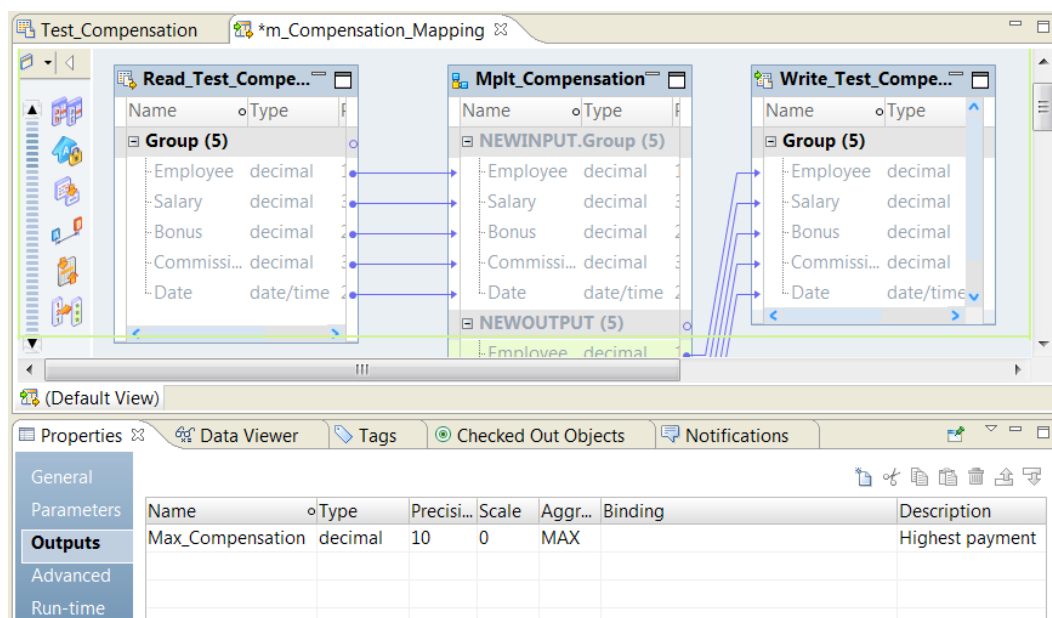
この例の場合、式トランスフォーメーションは Salary ポート値および Commission ポート値を集計します。Out\_Bonus マッピング出力は、Bonus ポート値に Bonus の 10%を加えている式です。

## マップレット出力のマッピング出力へのバインド

マップレットがマッピング出力を計算する場合は、マップレットの出力値をマッピングに渡す必要があります。

マッピングの【出力】ビューで、マップレット出力をマッピング出力にバインドします。

次の図は、マッピングレベルの Max\_Compensation マッピング出力を示しています。

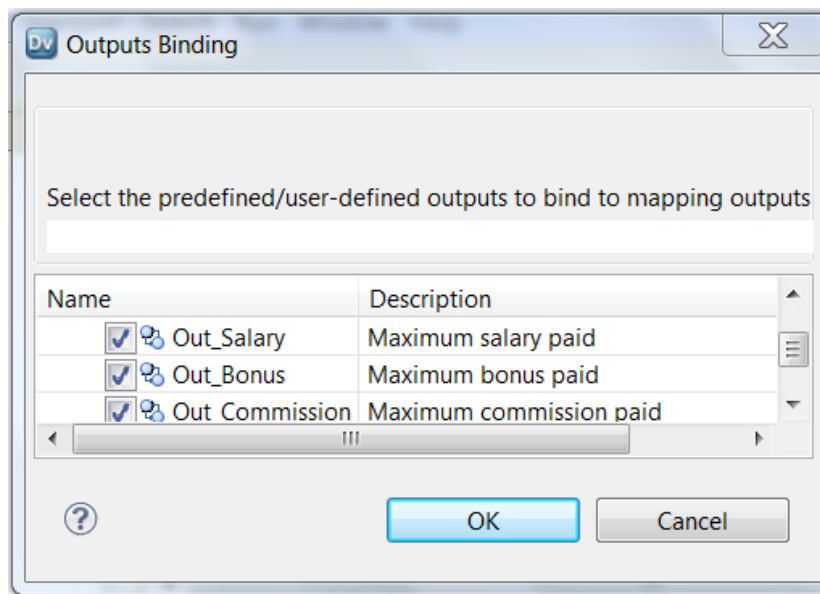


マッピングレベルで、Salary マップレット出力、Bonus マップレット出力、および Commission マップレット出力を、Max\_Compensation という同一のマッピング出力にバインドできます。

マップレット出力をマッピング出力にバインドするには、マッピング出力の【バインディング】カラムをクリックします。使用可能なマップレット出力のリストが表示されます。リストには、マッピング出力と同じタイプおよび集計のマッピング出力が表示されます。マッピング出力に割り当てるマップレット出力を選択します。

次の図は、【出力バインディング】ダイアログボックスを示しています。

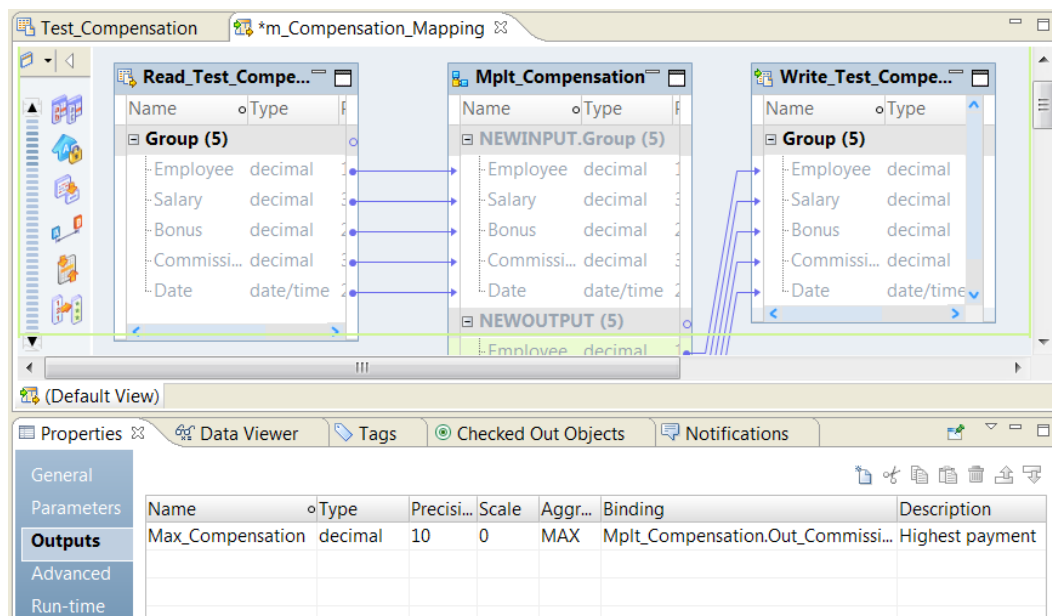




Max\_Compensation にバインドする出力を選択したら、【バインディング】フィールドに次のテキストが表示されます。

Mplt\_Compensation.Out\_Salary,Mplt\_Compensation.Out\_Bonus,Mplt\_Compensation.Out\_Commission

次の図は、【バインディング】フィールドのマッピング出力を示しています。



マッピング出力の集計タイプは MAX に設定します。データ統合サービスは、Salary、Bonus、または Commission ポートから見つかった最も高い報酬値を返します。

# 論理データオブジェクトのマッピング出力

論理データオブジェクトには、読み取りマッピングまたは書き込みマッピングを含めることができます。マッピング出力を返すようにそれらのマッピングを設定できます。論理データオブジェクトのマッピング出力を、マッピングのマッピング出力にバインドできます。

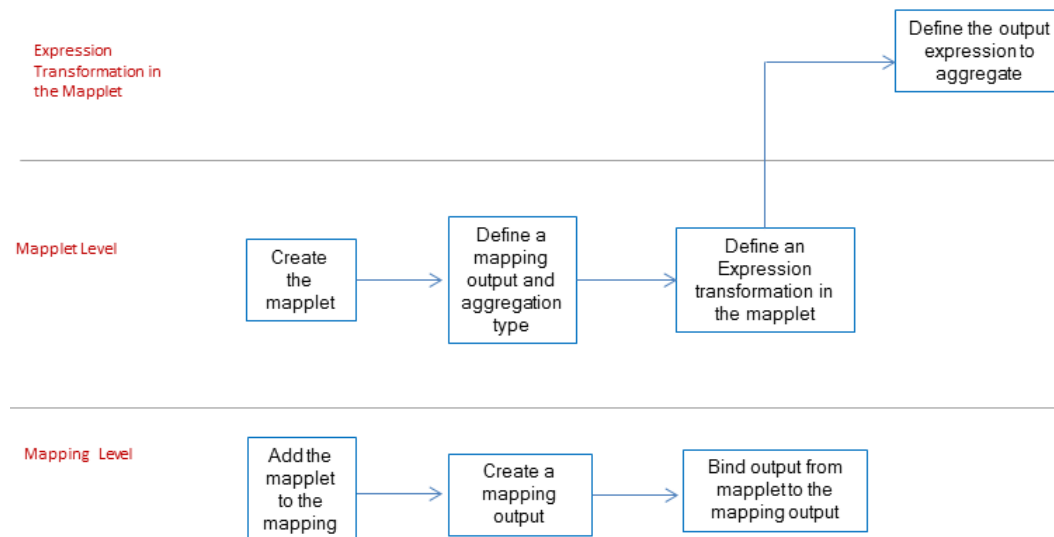
論理データオブジェクトをマッピングに含めると、読み取りマッピングまたは書き込みマッピングはマッピング出力の値を計算します。論理データオブジェクトは出力値をマッピングに渡します。論理データオブジェクトのマッピングの複数の出力を、同一の出力にマッピングレベルでバインドできます。論理データオブジェクトのシステム定義の出力をマッピング出力にバインドすることもできます。論理データオブジェクトのマッピング出力とマッピング出力は、同じタイプである必要があります。

## マップレット出力をマッピング出力にバインドする方法

マッピング出力を返すようにマップレットを設定できます。マップレットのマッピング出力を、マッピング出力にマッピングレベルでバインドできます。

マップレットをマッピングに含めると、マップレットは出力の値を計算して、出力値をマッピングに渡します。マップレットの複数の出力を、同一の出力にマッピングレベルでバインドできます。マップレットのシステム定義の出力をマッピング出力にバインドすることもできます。

次の図は、マップレット出力を設定してそれらをマッピング出力にバインドするプロセスを示しています。



マップレットの出力をマッピング出力にバインドするには、以下の手順を実行します。

1. マップレットを作成する。
2. マップレットの【出力】ビューで、マップレット出力名および集計のタイプを定義します。
3. マップレットに式トランスフォーメーションを追加して、式の【マッピング出力】ビューでマッピング出力式を設定します。
4. マップレットをマッピングに追加します。
5. マッピングにマッピング出力を作成します。

6. マップレットの出力をマッピング出力にバインドします。

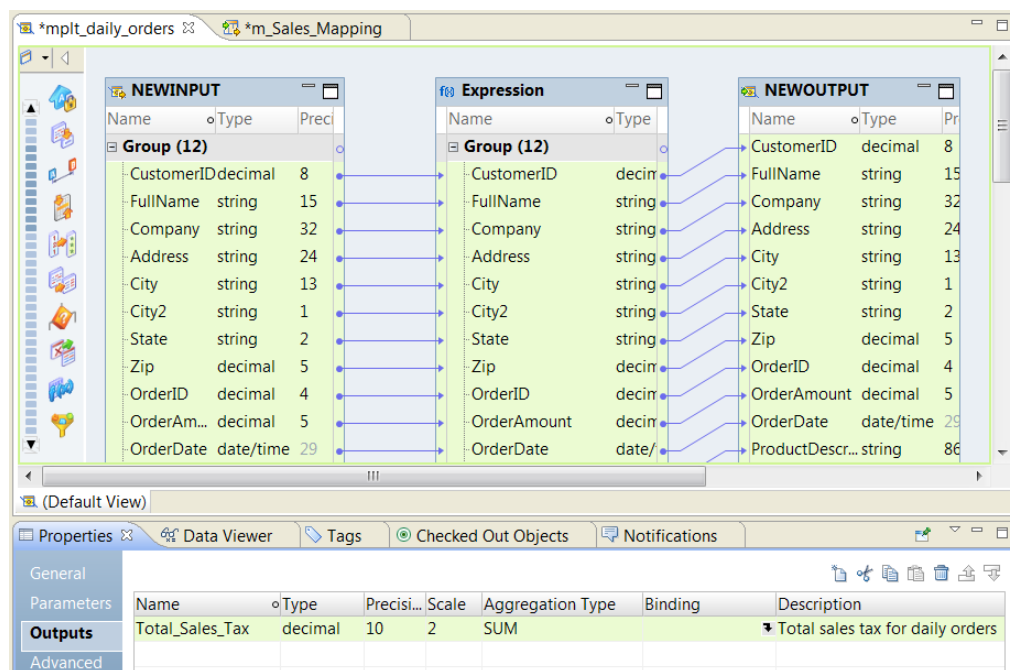
## マップレット出力の定義

マップレットを作成して、マップレットの【プロパティ】ビューの【出力】タブでマッピング出力を定義します。各マッピング出力の定義では、実行する集計のタイプ、および結果のデータ型を記述します。

1. マップレットを作成したら、マッピングキャンパスの内側をクリックして、マップレットのプロパティにアクセスします。
2. 【出力】ビューをクリックします。
3. 【新規】をクリックして、マッピング出力を作成します。  
Developer tool がマッピング出力を作成して、デフォルトのフィールドの値を設定します。
4. マッピング出力を識別する名前を変更します。
5. 数値または日付のマッピング出力タイプを選択します。数値タイプを作成する場合は、精度とスケールを入力します。
6. マッピング出力の集計タイプを選択します。

出力式を集計するか、マッピングが処理した最小式値または最大式値を見つけることができます。デフォルトは SUM です。

次の図は、集計タイプ SUM の Total\_Sales\_Tax というマップレット出力を示しています。



7. 【ファイル】 > 【保存】 をクリックして、マッピング出力を保存します。

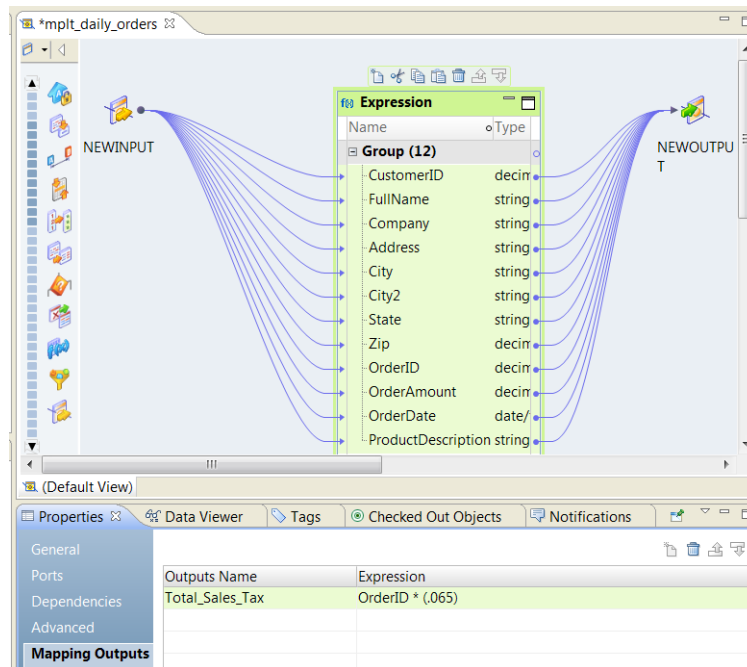
式トランスフォーメーションにマッピング出力式を作成する前に、マッピング出力を保存する必要があります。

## マップレットでのマッピング出力式の設定

マップレットが処理する各行を集計する式を設定します。

1. 式トランスフォーメーションをマップレットに追加します。  
トランスフォーメーションを配置する場所を決定する前に、マップレットロジックを検討します。
2. 式トランスフォーメーションで、**【マッピング出力】** ビューをクリックします。
3. **【新規】** をクリックして、出力式を追加します。  
Developer tool が、マップレットレベルで作成したいいずれかのマッピング出力と一致する出力名でマッピング出力を作成します。複数の出力を用意して、そこから選択することができます。
4. 式エディタで式を入力します。  
式には、ポート名を含めるか、関数、ポート、およびパラメータを含めることができます。
5. **【検証】** をクリックして式が有効であることを確認します。
6. **【OK】** をクリックして式を保存します。

次の図は、消費税を計算するマッピング出力式が表示された **【マッピング出力】** ビューを示しています。



7. **【ファイル】** > **【保存】** をクリックして、式トランスフォーメーションを保存します。

## マップレットからの出力のマッピング出力へのバインド

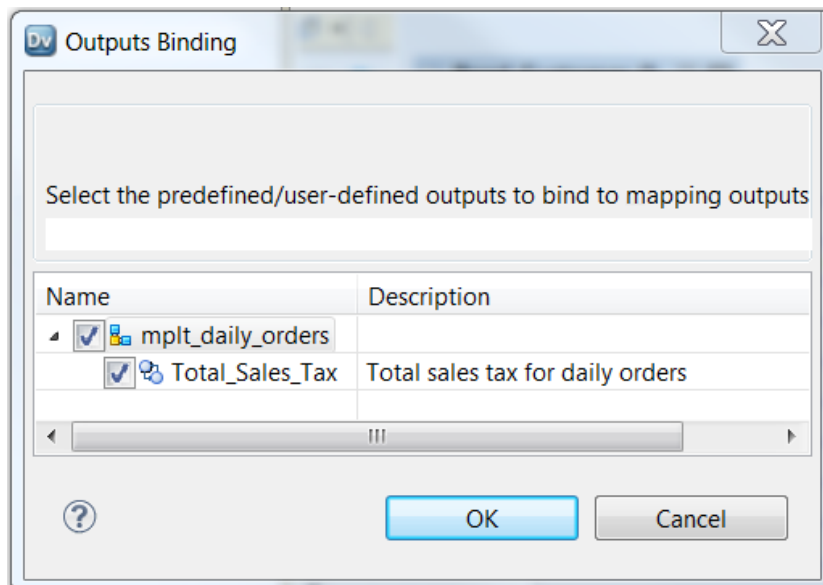
マップレットをマッピングに含めた場合、マッピングレベルで定義したマッピング出力にマップレットの出力をバインドできます。

1. マッピングを定義して、マップレットをマッピングに追加します。
2. マッピングキャンバスをクリックして、マッピングの **【プロパティ】** を表示します。
3. **【出力】** ビューをクリックします。
4. **【新規】** をクリックして、マッピング出力を作成します。

Developer tool がマッピング出力を作成して、デフォルトのフィールドの値を設定します。

5. バインド先となるマップレットのフィールドと一致するように、マッピング出力タイプ、集計タイプ、精度、およびスケールを変更します。
6. 必要に応じて、名前を変更して説明を入力します。
7. [バインディング] フィールドの選択矢印をクリックして、出力のリストを表示します。

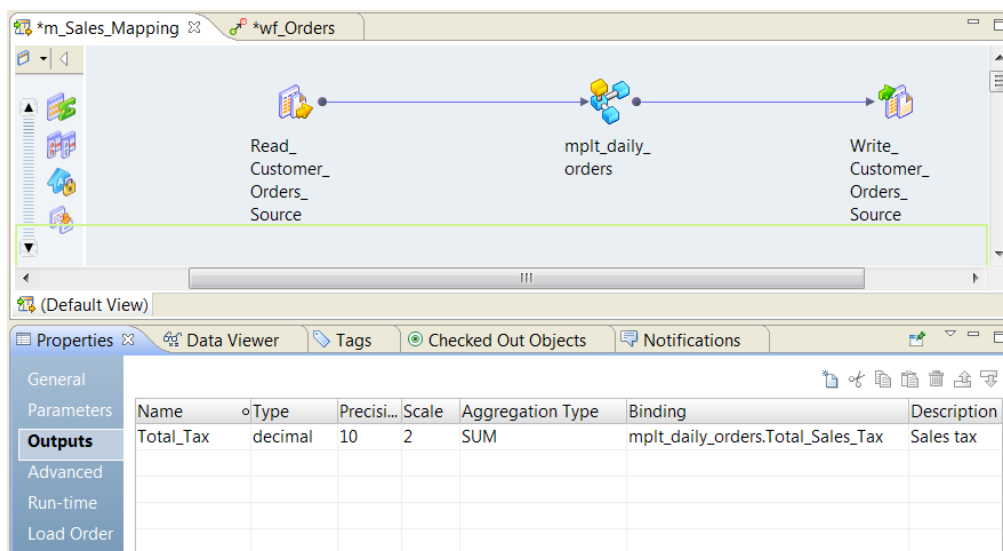
次の図は、[出力バインディング] ダイアログボックスを示しています。



8. マッピング出力にバインドするマップレット出力を選択します。  
複数のマップレット出力を選択して、同一のマッピング出力にバインドできます。
9. [OK] をクリックします。

選択したマップレット出力が [バインディング] フィールドに表示されます。

次の図は、マッピング出力の [バインディング] フィールドのマップレット出力名を示しています。



## 第 6 章

# SQL クエリからのマッピングの生成

この章では、以下の項目について説明します。

- [SQL クエリからのマッピングの生成の概要, 118 ページ](#)
- [SQL クエリから生成されたマッピングの例, 118 ページ](#)
- [マッピングを生成するための SQL 構文, 119 ページ](#)
- [マッピングを生成するクエリでの関数のサポート, 120 ページ](#)
- [SQL クエリからのマッピングまたは論理データオブジェクトの生成, 122 ページ](#)
- [SQL 文からのマッピングの生成, 122 ページ](#)

## SQL クエリからのマッピングの生成の概要

Developer tool で SQL クエリからマッピングを生成できます。マッピングを生成するために、SQL クエリを入力するか、クエリが含まれるテキストファイルをロードできます。必要に応じて、クエリテーブルのソースを定義できます。Developer tool が SQL クエリを検証し、マッピングを生成します。

SELECT 文のみが含まれる SQL クエリから論理データオブジェクトを生成することもできます。

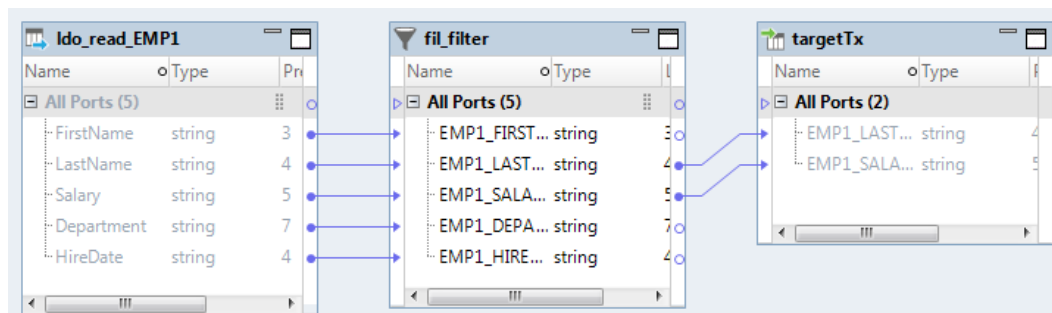
## SQL クエリから生成されたマッピングの例

従業員テーブルがあり、2001 年 1 月 1 日以降に雇用された従業員の給与のリストが必要だとします。

次の SQL 文を作成します。

```
SELECT LastName, Salary from emp1 where HireDate > 01/01/2001
```

次の図に、SQL 文から作成されたマッピングを示します。



## マッピングを生成するための SQL 構文

ANSI 準拠の SQL 文を使用して、Developer tool でマッピングを生成できます。

Developer tool は、標準的な SELECT クエリからマッピングを生成できます。以下に例を示します。

```
SELECT column_list FROM table-name
[WHERE clause]
[GROUP BY clause]
[HAVING clause]
[ORDER BY clause]
```

SELECT SQL 文に相関サブクエリが含まれている場合、クエリを単一の標準的なクエリとしてフラット化または書き換えることが可能であれば、クエリは有効です。

ANSI SQL では一部のデータ型がサポートされていません。例えば、カラムの 1 つのデータ型が timeStampTZ であるデータソースから結果を求めるクエリの場合、SQL は有効ではありません。

## 相関サブクエリ

相関サブクエリとは、WHERE 句に外部クエリからの値を使用するサブクエリです。データ統合サービスはクエリを実行する前に相関サブクエリをフラット化します。

以下の表に、データ統合サービスがフラット化した相関サブクエリの結果を示します。

タイプ	クエリ
フラット化なし	SELECT huge.* FROM huge WHERE c1 IN (SELECT c1 FROM tiny)
フラット化済み	SELECT huge.* FROM huge, tiny WHERE huge.c1 = tiny.c1

データ統合サービスが相関サブクエリをフラット化できるのは、相関サブクエリが次の要件を満たす場合です。

- タイプが IN または定量化された比較である。
- OR 演算子内にない、または SELECT リストの一部でない。
- LIMIT キーワードは含まれていない。
- GROUP BY 句を含まず、SELECT リスト、EXIST または NOT IN 論理演算子に集約する。
- 一意の結果を生成する。相関サブクエリの 1 つの列がプライマリキーである。例えば、r\_regionkey 列が vs.nation 仮想テーブルのプライマリキーである場合、次のクエリを発行できます: SELECT \* FROM vs.nation WHERE n\_regionkey IN (SELECT b.r\_regionkey FROM vs.region b WHERE b.r\_regionkey = n\_regionkey)。

- クエリに FROM リストが含まれる場合、FROM リスト内の各テーブルは SQL データサービス内の仮想テーブルです。

# マッピングを生成するクエリでの関数のサポート

Informatica では、ANSI SQL-92 規格を満たす関数をサポートしています。

さらに、一部の関数には特定の構文要件があります。

次の表に、関数とサポートされる構文を示します。

機能	構文
DATE()	<p>日付の形式を指定する:</p> <p>DATE(format '&lt;format&gt;')</p> <p>ここで、&lt;format&gt;は標準的な日付形式です。</p> <p>例:</p> <p>SELECT DATE(format 'dd-mm-yyyy') from table</p>
POSITION()	<p>リテラル文字列のサブstringの位置を判断する:</p> <p>POSITION('&lt;substring&gt;', '&lt;string&gt;')</p> <p>例:</p> <p>POSITION('MA', 'James Martin')</p> <p>テーブルカラムのサブstringの位置を判断する:</p> <p>POSITION('&lt;substring&gt;', &lt;column_name&gt;)</p> <p>例:</p> <p>POSITION('MA', FULL_NAME)</p>

## サポートされない関数を使用した SQL クエリからのマッピングの生成

Developer tool は SQL からマッピングを生成するときに、クエリの関数を検証します。ANSI 準拠の SQL を使用して、有効なマッピング生成であることを確認します。

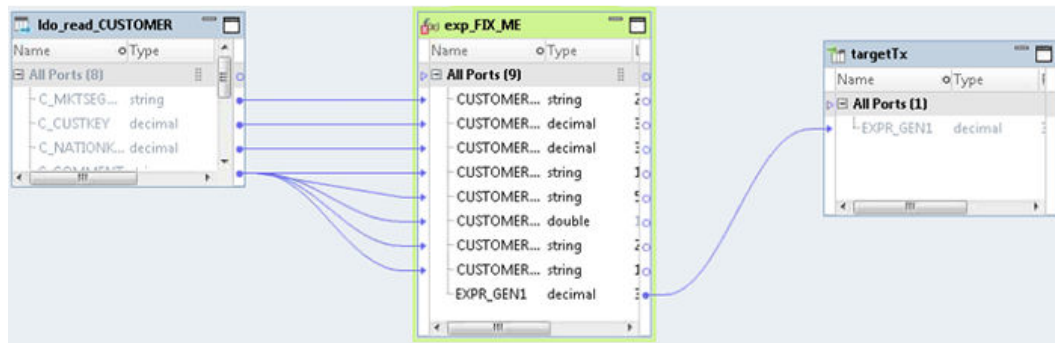
Developer tool が、有効な SQL 文に不明な関数を検出すると、FIX\_ME というラベルの付いたトランスフォーメーションまたは FIX\_EXPR というラベルの付いた式が含まれるマッピングを生成することがあります。マッピングを修正して有効な結果を得るには、これらのオブジェクトを編集します。不明な関数は、マッピングログファイルで警告メッセージとして表示されます。

例えば、次の SQL 文を使用してマッピングを生成します。

```
SELECT unknownFunctionABC(c_custkey,c_comment) from customer
```

次の図に、修正が必要な式トランスフォーメーションが含まれる SQL 文から生成されたマッピングを示します。





式トランスフォーメーションにエラーアイコンが付いている点に注意してください。[ポート] タブを使用して、正しくない式を編集します。エラーを修正するまでマッピングは有効になりません。

## INSERT、UPDATE、および DELETE の構文

次の構文を使用して、有効な INSERT 文、UPDATE 文、および DELETE 文を作成します。

- INSERT 文には次の構文を使用します。

```
INSERT INTO <TABLENAME> [<list>]
<select query>
```

- UPDATE 文には次の構文を使用します。

```
UPDATE [schema .] { table | view } [ alias ]
SET column = { expr | subquery }
[, column = { expr | subquery } ]... [WHERE condition]
```

- DELETE 文には次の構文を使用します。

```
DELETE FROM <Table> [[<AS>] <ALIAS>] [WHERE condition]
```

## INSERT 文、UPDATE 文、および DELETE 文のルールおよびガイドライン

INSERT 文、UPDATE 文、および DELETE 文では以下のルールおよびガイドラインについて検討してください。

- INSERT 文、UPDATE 文、または DELETE 文により、論理データオブジェクトであるマッピングでソースおよびターゲットオブジェクトが作成される。
- 1 つの INSERT 文、UPDATE 文、または DELETE 文のみが有効。例えば、INSERT とネストされた UPDATE 文が含まれる文は有効ではありません。
- INSERT、UPDATE、または DELETE の SQL 文に相關サブクエリが含まれる場合、Developer tool はマッピングを生成できない。
- UPDATE 文または DELETE 文により、マッピングでアップデートストラテジトランスフォーメーションが作成される。アップデートストラテジトランザクションにはプライマリキーが必要であるため、データターゲットにはプライマリキーを含める必要があります。マッピング生成後に、プライマリキーを確認します。
- リレーショナルデータベースは、データ挿入時に順序付けに従わないため、Developer tool は、ORDER BY 句の INSERT 文を無視します。

# SQL クエリからのマッピングまたは論理データオブジェクトの生成

SQL 文をマッピングまたは論理データオブジェクトに変換できます。論理データオブジェクトを生成して、他のマッピングで再利用できるオブジェクトを作成することもできます。

1. **【ファイル】 > 【新規】 > 【SQL クエリからのマッピング】** をクリックします。  
**【SQL クエリからのマッピングまたは論理データオブジェクトの生成】** ダイアログボックスが開きます。
2. SQL クエリを入力するか、SQL クエリが含まれるファイルを選択するかを選択します。
  - 編集可能な SQL クエリを入力するには、**【SQL クエリを入力してください】** を選択し、エディタで SQL クエリを入力するか貼り付けます。次に、**【クエリの検証】** をクリックします。
  - SQL クエリが含まれるファイルを選択するには、**【SQL ファイルを選択してください】** を選択し、SQL クエリが含まれるファイルを参照します。

Developer tool が SQL 構文を検証します。構文が有効でない場合、修正してからでないと続行することはできません。
3. マッピングではなく論理データオブジェクトを生成するには、**【マッピングを論理データオブジェクトとして生成します】** を選択します。
4. 必要に応じて、生成するマッピングまたは論理データオブジェクトの名前を変更します。
5. **【次へ】** をクリックします。  
ダイアログボックスに、データソースに対応するテーブルが表示されます。
6. **【データソース】** でテーブルの行をクリックし、マッピングのデータソースを選択します。  
テーブルにデータソースがある場合、クリックして必要に応じてデータソースを変更できます。  
**【データソースの選択】** ダイアログボックスが開き、アクセスできるモデルリポジトリのテーブルがリストされます。
7. モデルリポジトリで任意のデータソースを選択します。
8. **【完了】** をクリックします。

生成されたマッピングまたは論理データオブジェクトがエディタで開きます。

マッピングで任意のオブジェクトを選択し、表示または編集できます。その後、マッピングを実行したり、データ統合サービスにデプロイするアプリケーションまたはワークフローに含めたりすることができます。

論理データオブジェクトを作成した場合、他のマッピングで再利用できます。例えば、生成された論理データオブジェクトをマッピングでソースとして再利用できます。

## SQL 文からのマッピングの生成

SQL 文からマッピングを生成するには、次のタスクを実行します。

1. SQL 文を作成します。
2. SQL 文を Developer tool に貼り付けるかインポートし、SQL 文を検証して、マッピングを生成します。
3. マッピングの開発を完了します。次の手順に従ってください。
  - a. マッピングをテストして、要件を満たすまで作成を繰り返します。
  - b. マッピングをデータ統合サービスにデプロイします。

## SQL 文の作成

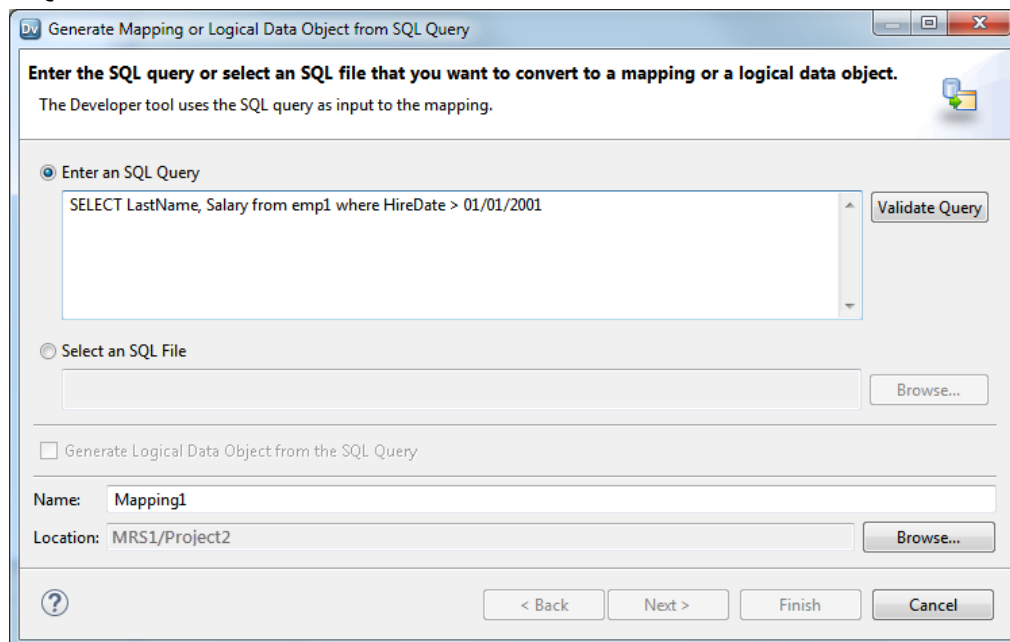
SQL 文を使用してマッピングを生成するために、SQL 文を作成します。

SQL クエリツールを使用するか、SQL 文を最初から記述して、SQL 文を作成できます。この記事では、構文のガイドラインに従います。

**注:** Informatica 以外のいくつかの関数がサポートされています。マッピングを生成する有効なクエリで使用できる関数もありますが、結果は有効ではありません。SQL 文での関数のサポートの詳細については、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。

## SQL 文の Developer tool への貼り付けまたはインポート

1. インポートする SQL 文を含む SQL ファイルを見つけるか、文全体をクリップボードにコピーします。
2. Developer tool で、[ファイル] > [新規] > [SQL クエリからのマッピング] をクリックします。  
[SQL クエリからのマッピングまたは論理データオブジェクトの生成] ダイアログボックスが開きます。



3. クエリをダイアログボックスにインポートします。次の方法のいずれかを選択してください。
  - [SQL クエリを入力してください] を選択して、クリップボードからエディタにクエリを貼り付ける。
  - [SQL ファイルを選択してください] を選択し、ファイルを参照して選択する。

4. [検証] をクリックします。

Developer tool で、SQL 文が検証されます。エラーがあれば修正します。

5. マッピングではなく論理データオブジェクトを生成する場合は、[SQL クエリからの論理データオブジェクトの生成] を選択します。

このオプションを選択した場合は、次の手順を実行します。

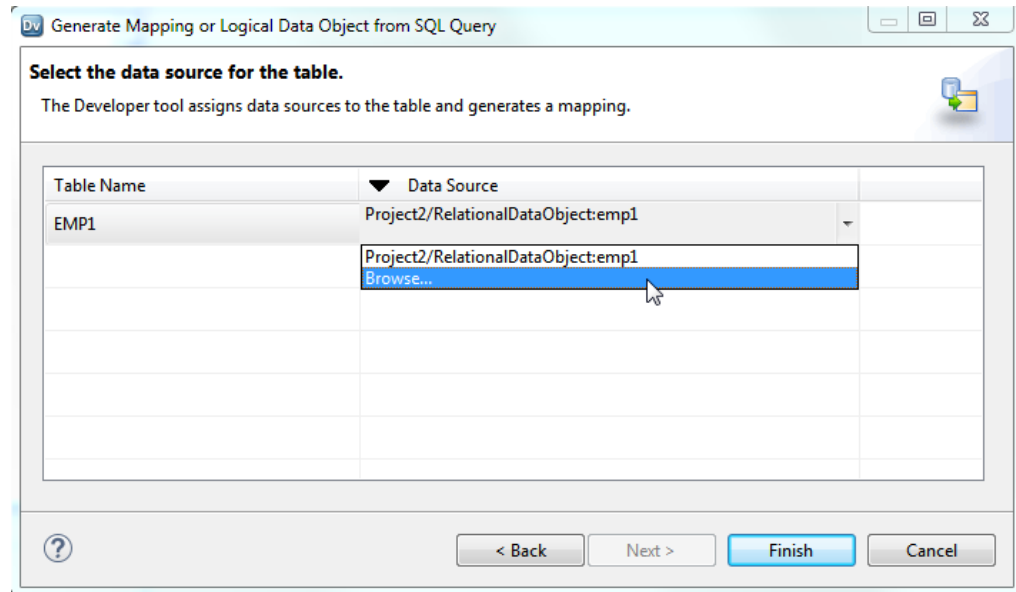
1. 必要に応じて、作成する論理データオブジェクトの名前を変更します。
2. 必要に応じて [参照] をクリックし、論理データオブジェクトの格納場所を選択するか、デフォルトの場所を受け入れます。
3. [検証] をクリックします。  
Developer tool で、SQL 文が検証されます。エラーがあれば修正します。

6. [次へ] をクリックします。

[テーブルのデータソースを選択してください] ダイアログボックスが表示されます。

7. テーブルのデータソースを選択するには、[データソース] ルカラムをクリックして、[参照] をクリックします。

次の図は、データソースを選択するためにどこで [参照] をクリックするかを示しています。



8. [完了] をクリックします。

Developer tool によって SQL クエリからマッピングが生成され、エディタにマッピングが表示されます。

## マッピングの開発の完了

マッピングを作成したら、次の手順を実行して、マッピングの開発を完了します。

1. マッピングを実行して結果を表示します。
2. 要件を満たすまで、マッピングの編集と再実行を繰り返します。
3. データ統合サービスにマッピングをデプロイして実行します。  
マッピングを単独でデプロイするか、デプロイするアプリケーションにマッピングを含めることができます。マッピングを単独でデプロイする場合は、データ統合サービスによって、そのマッピングを含むアプリケーションが作成されます。

マッピング、アプリケーション、およびデプロイメントの詳細については、『*Informatica Developer tool ガイド*』を参照してください。

## 第 7 章

# 動的マッピング

この章では、以下の項目について説明します。

- [動的マッピングの概要, 125 ページ](#)
- [動的マッピングの設定, 126 ページ](#)
- [動的ソース, 129 ページ](#)
- [動的ターゲット, 133 ページ](#)
- [動的ポートおよび生成されたポート, 139 ページ](#)
- [動的式, 141 ページ](#)
- [入力ルール, 143 ページ](#)
- [選択ルールとポートセレクト, 152 ページ](#)
- [設計時リンク, 154 ページ](#)
- [ランタイムリンク, 156 ページ](#)
- [動的マッピングのトラブルシューティング, 159 ページ](#)

## 動的マッピングの概要

動的マッピングとは、実行時にソース、ターゲット、およびトランスフォーメーションロジックに対する変更に対応できるマッピングです。動的マッピングは、スキーマまたはメタデータへの頻繁な変更を管理する、あるいは異なるスキーマを持つデータソースにマッピングロジックを再利用する場合に使用します。動的マッピングを作成するには、ルール、パラメータ、全般的なトランスフォーメーションパラメータを設定します。

ソース、ターゲット、またはルックアップでデータソースが変更される場合、マッピングを設定すれば実行時にメタデータの変更を動的に取得できます。マッピングのすべてのステージで変更を受信してプロパゲートするには、マッピング内でパラメータ、ルール、ポート、リンクを設定します。マッピングを再び実行する前に手動でデータオブジェクトを同期して各トランスフォーメーションを更新する必要はありません。データ統合サービスは、トランスフォーメーションポート、ポート内のトランスフォーメーションロジック、マッピング内のポートリンクを動的に決定できます。

### 動的マッピングの例

毎週、さまざまな部門から顧客データを受信し、結合して集計する必要があるとします。部門は、部門分析用の追加のカラムを含めるためにソーススキーマを定期的に変更する可能性があります。

データソースへのこのような変更に対応するために、動的マッピングを作成します。読み取り時にデータオブジェクトカラムを取得するように、読み取りトランスフォーメーションを設定します。必要なカラムを含めて、その他のすべてのカラムを除外するように入力ルールを作成します。

## 動的マッピングの設定

ソースが変更された場合、変更に対応するために読み取りトランスフォーメーションを設定できます。例えば、異なるデータソースを使用するか、データソースに基づいてデータオブジェクトを更新するためにトランスフォーメーションを設定できます。ターゲットが変更された場合、ターゲットの変更に対応するために書き込みトランスフォーメーションを設定できます。例えば、関連付けられているデータオブジェクトまたはマッピングフローに基づいてカラムを生成するために書き込みトランスフォーメーションを設定できます。ターゲットがリレーショナルの場合、実行時にテーブルを作成または置換できます。

マッピング全体を通して変更を受信してプロパゲートするために、マッピングまたはマップレットでトランスフォーメーションを設定します。データフローに基づいて新しいカラムまたは変更されたカラムを受信するために動的ポートを作成します。動的カラムは各入力カラムのポートを生成します。動的ポートが受信するカラムを決定し、生成されたポートの名前または順序を変更するように入力ルールを設定します。

式で動的ポートまたは選択ルールを使用して動的式を作成します。動的ポートを含めた場合、式は動的ポートが生成する各ポートに対して実行されます。選択ルールを含めた場合、式はルールの各ポートに対して実行されます。

式、ジョイナ、またはルックアップトランスフォーメーションに生成されたポートが含まれる場合、マッピングが実行されるときに生成されたポートに対する変更に対応するポート選択ルールを設定できます。例えば、販売データに基づいて計算を実行する必要があるが、販売カラム名が各ソースで異なっているとします。計算するには正しいカラムを選択するルールを作成します。

パラメータを使用して実行時に値を変更できます。パラメータを使用して、マッピング内のソース、ターゲット、接続、ルールなどの値を変更します。

トランスフォーメーションは、マッピングを設計するときに直接リンクを作成できないように変更される場合があります。設計時にリンクを作成できない場合、ランタイムリンクを設定できます。ランタイムリンクはポリシーまたはパラメータを使用して実行時にトランスフォーメーショングループ間をリンクするポートを決定します。

## 動的データソース

実行時にソースおよびターゲットに対する変更に対応するためにマッピングを設定できます。動的マッピングにはフラットファイルおよびリレーショナルデータソースを含めることができます。パラメータを使用して、予測される変更のタイプに基づいてトランスフォーメーションのプロパティを設定できます。

次のデータソースに対するランタイム変更に対応するためにマッピングを設定できます。

### ソース

動的ソースにはリレーショナルソースおよびフラットファイルソースを含めることができます。ランタイム変更に対応するために読み取りトランスフォーメーションおよび物理データオブジェクトを設定します。ファイルまたはソース接続の場所、入力ソースカラムに対する変更、またはデータオブジェクトに基づいてソースメタデータを変更できます。

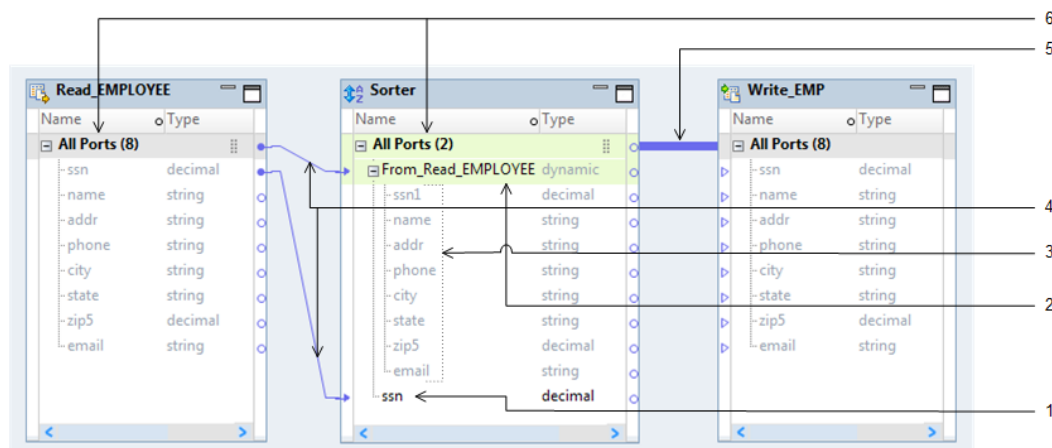
### ターゲット

動的ターゲットにはリレーショナルターゲットおよびフラットファイルターゲットを含めることができます。マッピングフローまたは関連付けられているデータオブジェクトに基づいてターゲットを定義できます。データソースからデータオブジェクトのカラムを取得することも選択できます。パラメータを使用して、ターゲットデータオブジェクトやターゲット接続など、ランタイムプロパティを変更することができます。

## 動的マッピングのポートおよびリンク

メタデータの変更を処理するために、通常のマッピングには含まれないポートおよびリンクのタイプを設定できます。

次の画像は、動的マッピングで表示されるポートおよびリンクを示しています。



1. 静的ポート (ポート)
2. 動的ポート
3. 生成されたポート
4. 設計時リンク (リンク)
5. ランタイムリンク
6. ポートグループ

### 静的ポート (ポート)

マッピング、動的、非動的のどのタイプでも作成できるポート。データはポートの内外に渡すことができ、データには動的設定が一切含まれません。

### 動的ポート

アップストリームトランスフォーメーションから1つ以上のカラムを受信できるトランスフォーメーションのポート。動的ポートは、マッピングを使用して渡すメタデータに基づき、新規または変更されたカラムを受信できます。

### 生成されたポート

動的ポート内の1つのカラムを表すポート。動的ポートは動的ポートのルールに基づいて各カラムの生成されたポートを作成します。

### 設計時リンク (リンク)

データを1つのトランスフォーメーションから別のトランスフォーメーションにプロパゲートするポートを接続するために作成するリンク。これらのリンクは通常のマッピングで作成することもできます。

### ランタイムリンク

ポリシー、パラメータ、またはその両方に基づいてデータ統合サービスが実行時に接続するポートを決定するために使用するトランスフォーメーショングループ間のリンク。

### ポートグループ

データの行を表すマッピング内のポートのセット。動的マッピングでは、グループをダウンストリームトランスフォーメーションにドラッグして動的ポートを作成できます。



## 動的マッピングのルール

動的トランスフォーメーション内でルールを作成し、動的ポートが受信するポートと生成するポートを制御します。

次のタイプの動的マッピングのルールを設定できます。

### 入力ルール

入力ルールは動的ポートが生成するポートを定義します。ポートを含めることまたは除外することを選択できます。生成されたポートの名前や順序を変更することも選択できます。

### 選択ルールとポートセレクト

選択ルールを作成して、データ統合サービスが実行時に処理する生成されたポートを定義します。ポートセレクト内で選択ルールを作成します。ポートセレクトには、式または結合条件またはルックアップ条件で参照できるポートが含まれます。実行時に予測されるメタデータの変更に基づいて、1つのトランスフォーメーションで複数のポートセレクトを設定できます。

## 動的マッピングのパラメータ

パラメータは、マッピングの実行間で変わる定数値です。フラットファイルまたはリレーショナルリソースのソースやターゲットを変更するには、動的マッピングのパラメータを使用します。パラメータを使用して、入力ルール、選択ルール、トランスフォーメーションプロパティ、ランタイムリンクを変更することもできます。

次の表に、動的マッピングコンポーネントについて作成できるパラメータの機能を示します。

動的マッピングコンポーネント	パラメータの機能
アグリゲータトランスフォーメーション	グループ化ポートを変更する。
ジョイナトランスフォーメーション	結合条件を変更する。
ルックアップトランスフォーメーション	ルックアップ条件を変更する。
ランクトランスフォーメーション	グループ化ポートを変更する。
読み取りトランスフォーメーション	次のタスクを実行するためにパラメータを作成します。 <ul style="list-style-type: none"><li>- フラットファイルソースの入力ファイル名またはディレクトリを変更する。</li><li>- リレーショナルソースの接続を変更する。</li><li>- フラットファイルデータオブジェクト、カスタマイズされたデータオブジェクト、またはリレーショナルデータオブジェクトを変更する。</li></ul>
ルール	次のタスクを実行するためにパラメータを作成します。 <ul style="list-style-type: none"><li>- 名前またはパターン別に入力ルール条件を変更する。</li><li>- 名前またはパターン別に選択ルール条件を変更する。</li></ul>
ランタイムリンク	トランスフォーメーショングループ間をリンクするためのポートのセットを変更する。



動的マッピングコンポーネント	パラメータの機能
ソートトランスフォーメーション	ソートキーを変更する。
書き込みトランスフォーメーション	次のタスクを実行するためにパラメータを作成します。 <ul style="list-style-type: none"> <li>- フラットファイルターゲットの出力ファイル名またはディレクトリを変更する。</li> <li>- リレーショナルターゲットの接続を変更する。</li> <li>- フラットファイルデータオブジェクト、カスタマイズされたデータオブジェクト、またはリレーショナルデータオブジェクトを変更する。</li> </ul>

## 関連項目：

- [「マッピングパラメータの概要」 \(ページ 49\)](#)

# 動的ソース

動的ソースとは実行時に変更できるソースです。フラットファイル、リレーショナル、Amazon S3、Amazon Redshift、HBase、HDFS、Microsoft Azure Blob ストレージ、Microsoft Azure Data Lake Store および複合ファイルの動的ソースを、1つのマッピングで設定できます。

ソースの動的ランタイム機能は、次の方法で設定できます。

## データソースからカラムを取得する。

実行時にソースがわずかに変更されることが予想される場合、実行時にフラットファイルまたはリレーショナルオブジェクトカラムを取得するように読み取りトランスフォーメーションを設定できます。リレーショナルデータソースまたはフラットファイルデータソースの構造に基づいて、実行時に読み取りトランスフォーメーションのポートを更新できます。

## パラメータを割り当ててソースのフラットファイル名およびディレクトリを決定する。

フラットファイルソースが類似している場合、パラメータをファイル名またはディレクトリに割り当てることができます。パラメータを使用する場合、各ソースのデータオブジェクトを作成する必要はありません。

## パラメータを割り当ててリレーショナルデータオブジェクトのリソース、テーブル所有者、またはディレクトリを決定する。

リレーショナルソースが類似している場合、パラメータを割り当ててリソース、接続、テーブル所有者プロパティを取得できます。

## パラメータを割り当ててファイルまたはリレーショナルソースに使用するデータオブジェクトを決定する。

ソースがわずかに変更されることが予想される場合、リレーショナルデータソースまたはフラットファイルデータソースの構造に基づいて、実行時に読み取りトランスフォーメーションのポートを更新できます。

次の表に、ソースの動的ランタイム機能を設定できるケースを示します。

動的ランタイムソース機能	設定
データソースからカラムを取得する。	次のソースタイプの読み取りトランスフォーメーションで【データオブジェクト】タブを設定する。 <ul style="list-style-type: none"><li>- フラットファイル</li><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- Microsoft Azure Blob ストレージ</li><li>- Microsoft Azure Data Lake Store</li><li>- HBase</li><li>- HDFS</li><li>- Snowflake</li></ul>
パラメータを割り当ててフラットファイル名およびディレクトリを決定する。	次のソースタイプの物理データオブジェクトで【詳細】タブを設定する。 <ul style="list-style-type: none"><li>- フラットファイル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- Microsoft Azure Blob ストレージ</li><li>- Microsoft Azure Data Lake Store</li><li>- HDFS</li><li>- HBase</li><li>- Snowflake</li></ul>
パラメータを割り当てて接続、所有者、またはリソースを決定する。	次のソースタイプの読み取りトランスフォーメーションで【ランタイム】タブを設定する。 <ul style="list-style-type: none"><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- Snowflake</li></ul>
パラメータを割り当ててデータオブジェクトを決定する。	次のソースタイプの読み取りトランスフォーメーションで【データオブジェクト】タブを設定する。 <ul style="list-style-type: none"><li>- フラットファイル</li><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- HDFS</li><li>- Snowflake</li></ul>

## データソースからカラムを取得する

リレーショナルまたはフラットファイルデータソースの構造に基づいて、実行時に読み取りトランスフォーメーションのポートを更新できます。この方法を使用して、読み取りトランスフォーメーションのランタイムインスタンスを更新することもできます。

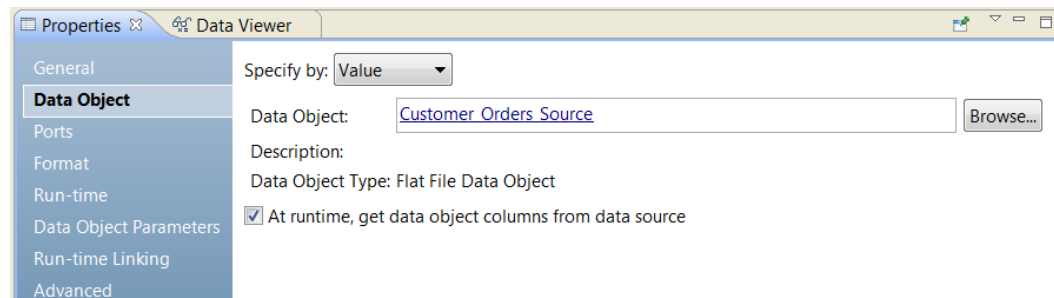
ソースがわずかに変更されることが予想される場合、実行時にカラムを更新します。例えば、別の組織からソースを処理する必要があるが、組織ではソースファイル内のカラムの順序については保証できないとします。実行時にデータオブジェクトカラムを更新するオプションを有効にした場合、データ統合サービスはソースデータの構造に基づいて読み取りトランスフォーメーションのポートを変更します。読み取りトランスフォーメーションは、処理を行う動的マッピングのダウンストリームトランスフォーメーションにデータを渡します。

実行時にデータオブジェクトカラムを更新した場合、データ統合サービスは読み取りトランスフォーメーションのランタイムインスタンスを更新します。モデルリポジトリ内のメタデータは更新しません。Developer tool 内の変更を確認するには、解決されたパラメータを使用してマッピングを表示します。モデルリポジトリ

の物理データオブジェクト定義を更新するには、Developer tool で同期オプションを使用します。Developer tool は物理データオブジェクトのメタデータを再度インポートし、メタデータを変更します。

**注:** カスタム SQL クエリを動的マッピングで使用することはできません。

次の画像は、**【データオブジェクト】** タブでオプションを有効にする箇所を示しています。



データ統合サービスはスキーマでリレーショナルソースの構造を決定します。**【ランタイム】** タブに表示されるリソースのスキーマを調査します。次に、スキーマに基づいてトランスフォーメーションデータオブジェクトのカラムを更新します。

データ統合サービスは、フラットファイルの物理データオブジェクトを設定する方法に基づいてフラットファイルソースの構造を決定します。実行時にカラム名を生成するために、データオブジェクトを設定します。

この機能は、フラットファイルまたはリレーショナルソースの読み取りトランスフォーメーションの**【データオブジェクト】** タブで設定します。

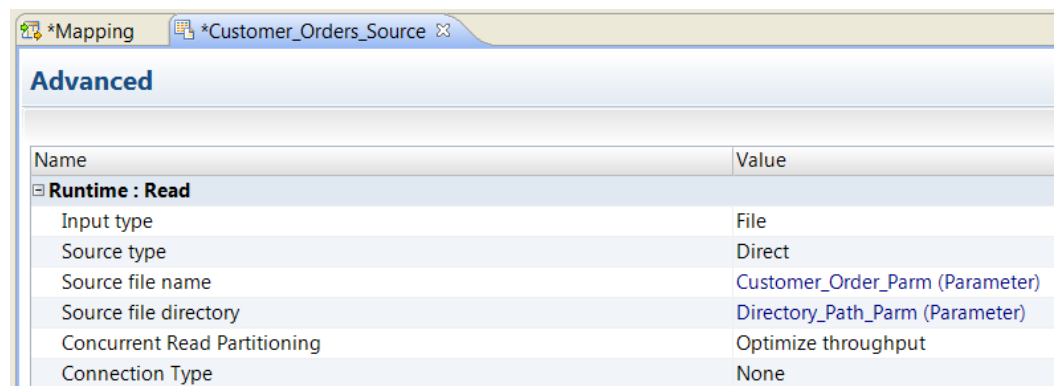
フラットファイル物理データオブジェクトのプロパティの詳細については、『*Informatica Developer Tool ガイド*』を参照してください。

## フラットファイル名にパラメータを割り当てる

類似したフラットファイルソースを使用して動的マッピングを実行するために、ファイル名またはディレクトリにパラメータを割り当てることができます。パラメータを使用する場合、各ソースのデータオブジェクトを作成する必要はありません。

フラットファイル物理データオブジェクトのファイル名とディレクトリをパラメータ化できます。データオブジェクトからトランスフォーメーションを作成する前にプロパティをパラメータ化できます。物理データオブジェクトプロパティの**【詳細】** タブでパラメータを設定します。物理データオブジェクトからトランスフォーメーションを作成するときに、マッピングパラメータを使用してパラメータのデフォルト値をオーバーライドできます。

次の画像は、物理データオブジェクトの**【詳細】** タブを示しています。



この機能は、フラットファイルソースの物理データオブジェクトの**【詳細】** タブで設定します。

## リレーショナルソースのプロパティにパラメータを割り当てる

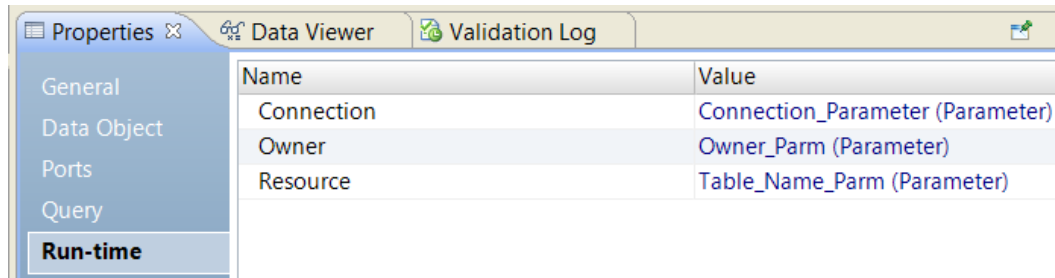
類似したリレーショナルソースを使用して動的マッピングを実行するために、読み取りトランスフォーメーションでリソース、接続、テーブル所有者プロパティにパラメータを割り当てることができます。

同じデータベースで異なるが類似しているテーブルを使用してマッピングを実行するには、リソースのパラメータを使用します。リソースのパラメータを使用する場合、各ソースのデータオブジェクトを作成する必要はありません。接続用のパラメータを使用して異なるデータベースにアクセスします。複数のリレーショナルソースに対して一意の SQL クエリを実行する必要がある場合があります。

トランスフォーメーションプロパティの **【ランタイム】** タブでリレーショナルテーブルのパラメータを設定します。リレーショナル物理データオブジェクトでこれらのプロパティはパラメータ化できません。読み取りトランスフォーメーションのプロパティのパラメータを作成する場合、マッピングパラメータを作成します。読み取りトランスフォーメーション用のランタイムパラメータを設定すると、読み取りトランスフォーメーションが使用するデータソースは、リレーショナルデータベースに保存されたデータに基づいて更新されます。

デフォルトでは、接続の接続タイプパラメータを作成します。テーブル所有者のテーブル名と文字列パラメータにリソースタイプパラメータを設定します。

次の画像は、読み取りトランスフォーメーションの **【ランタイム】** タブを示しています。



この機能は、リレーショナルソースの読み取りトランスフォーメーションの **【ランタイム】** タブで設定します。

## ソースデータオブジェクトにパラメータを割り当てる

データオブジェクトにパラメータを割り当てて、実行時に読み取りトランスフォーメーションのソースを変更することができます。

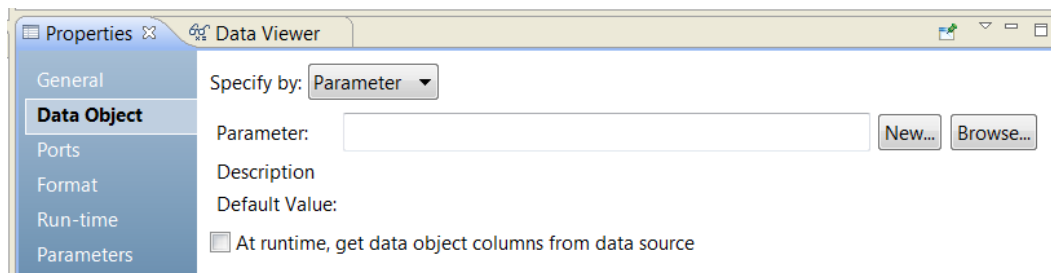
各データソースのモデルリポジトリに他の物理データオブジェクトがある場合、データオブジェクトをパラメータ化します。フラットファイルまたはデータベーステーブルの同じトランスフォーメーションを設定する必要がある場合にデータオブジェクトをパラメータ化することができます。データオブジェクトをパラメータ化すると、トランスフォーメーションで異なるプロパティまたは一意の SQL クエリがあるデータオブジェクトを使用できます。

物理データオブジェクトから読み取りトランスフォーメーションを作成すると、データオブジェクトの情報はトランスフォーメーションプロパティの **【データオブジェクト】** タブに表示されます。データオブジェクト名をクリックすると、モデルリポジトリにある物理データオブジェクトの定義を表示することができます。

データオブジェクトをパラメータ化するには、リソースタイプパラメータを作成するか、リソースパラメータを参照します。パラメータのデフォルト値は、モデルリポジトリ内の物理データオブジェクトの名前です。デフォルトのパラメータ値を作成する場合、リポジトリ内のデータオブジェクトのリストから物理データオブジェクト名を選択します。

データオブジェクトを変更すると、トランスフォーメーションポートが変更されます。トランスフォーメーションプロパティの **【ポート】** タブで、ポートを表示できます。

次の図は、パラメータでデータオブジェクトを指定した場合の **【データオブジェクト】** タブを示しています。



次の表に、【データオブジェクト】タブのパラメータオプションを示します。

パラメータオプション	説明
パラメータ	データオブジェクトとして設定したリソースパラメータの名前。読み取り専用。
説明	パラメータの説明。読み取り専用。
新規	リソースパラメータを作成します。パラメータのデフォルト値としてモデルリポジトリ内のデータオブジェクトを参照し、選択します。
参照	リソースパラメータを参照し、パラメータを選択します。
デフォルト値	データオブジェクトに設定したリソースパラメータのデフォルト値。デフォルト値は物理データオブジェクト名です。読み取り専用。

この機能は、フラットファイルまたはリレーショナルソースの読み取りトランスフォーメーションの【データオブジェクト】タブで設定します。

## 動的ターゲット

動的ターゲットでは、実行時にターゲットが変わります。

マッピングを実行すると、動的ターゲットは、リレーショナルテーブル、フラットファイル、Amazon Redshift、Amazon S3、Microsoft Azure Data Lake Store、Microsoft Azure Blob ストレージ、HDFS、HBase、カスタマイズされたデータオブジェクトを含む物理データターゲットからメタデータの変更を取得できます。また、アップストリームカラム定義に基づいてカラムを生成することもできます。

ターゲットの動的ランタイム機能は、次の方法で設定できます。

### データソースからカラムを取得する。

ターゲットがわずかに変更されることが予想される場合、書き込みトランスフォーメーションを設定して、実行時にリレーショナルオブジェクトカラムを取得できます。書き込みトランスフォーメーションを設定して、ターゲットからメタデータを取得する場合、動的に更新してターゲットオブジェクトとの同期を維持するように書き込みトランスフォーメーションを設定できます。

### マッピングフローに基づいてターゲットカラムを定義する。

マッピングフローに基づいてカラムを定義する場合、ターゲットカラムはアップストリームトランスフォーメーションによって決定されます。

### データオブジェクトに基づいてターゲットカラムを定義する。

データオブジェクトに基づいてカラムを定義する場合、ターゲットカラムは関連付けられているデータオブジェクトによって決定されます。

### ターゲットスキーマ戦略を定義する。

ターゲットスキーマ戦略を定義して、実行時にターゲットスキーマを維持するか、ターゲットテーブルを作成または置換できます。書き込みトランスフォーメーションでターゲットスキーマ戦略のパラメータ値を指定することもできます。

書き込みトランスフォーメーション内のターゲットスキーマ戦略を設定して実行時にターゲットを作成または置換する場合、データ統合サービスはデータオブジェクトまたはマッピングフローに基づいてターゲットを作成します。クエリに基づいてターゲットを作成するように DDL クエリを定義することもできます。

### パラメータを割り当ててリレーショナルデータオブジェクトのリソース、テーブル所有者、またはディレクトリを決定する。

リレーショナルターゲットが類似している場合、パラメータを割り当ててリソース、接続、テーブル所有者プロパティを取得できます。

### パラメータを割り当ててファイルまたはリレーショナルターゲットに使用するデータオブジェクトを決定する。

カスタマイズされたデータオブジェクトを書き込みトランスフォーメーションとして作成し、パラメータ値をトランスフォーメーションのターゲットとして指定できます。パラメータの値を変更した場合、そのパラメータを使用するすべてのオブジェクトでターゲットが変更されます。

次の表に、ターゲットの動的ランタイム機能を設定できるケースを示します。

動的ランタイムターゲット機能	設定
データソースからカラムを取得する。	次のターゲットタイプの書き込みトランスフォーメーションで <b>【データオブジェクト】</b> タブを設定する。 <ul style="list-style-type: none"><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- Microsoft Azure Blob ストレージ</li><li>- Microsoft Azure Data Lake Store</li><li>- HBase</li><li>- HDFS</li><li>- Snowflake</li></ul>
データオブジェクトまたはマッピングフローに基づいてターゲットカラムを定義する。	次のターゲットタイプの書き込みトランスフォーメーションで <b>【ポート】</b> タブを設定する。 <ul style="list-style-type: none"><li>- フラットファイル</li><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Amazon S3</li><li>- Microsoft Azure Blob ストレージ</li><li>- Microsoft Azure Data Lake Store</li><li>- HBase</li><li>- HDFS</li><li>- Snowflake</li></ul>
ターゲットスキーマ戦略を選択する。	次のターゲットタイプの物理データオブジェクトで <b>【詳細】</b> タブを設定する。 <ul style="list-style-type: none"><li>- リレーショナル</li><li>- Amazon Redshift</li><li>- Snowflake</li></ul>

動的ランタイムターゲット機能	設定
DDL クエリを定義して実行時にターゲットテーブルを作成します。	次のターゲットタイプの物理データオブジェクトで <b>【詳細】</b> タブを設定する。 - リレーショナル
パラメータを割り当てて接続、所有者、またはリソースを決定する。	次のターゲットタイプの書き込みトランスフォーメーションで <b>【ランタイム】</b> タブを設定する。 - リレーショナル - Amazon Redshift - Amazon S3 - Snowflake
パラメータを割り当ててデータオブジェクトを決定する。	次のターゲットタイプの書き込みトランスフォーメーションで <b>【データオブジェクト】</b> タブを設定する。 - フラットファイル - リレーショナル - Amazon Redshift - Amazon S3 - Snowflake

## データソースからカラムを取得する

リレーショナルデータソースの構造に基づいて、実行時に書き込みトランスフォーメーションのポートを更新できます。

ターゲットカラムがわずかに変更されることが予想される場合、実行時にカラムを更新します。実行時にデータソースからデータオブジェクトカラムを取得した場合、データ統合サービスはターゲットの構造に基づいてデータオブジェクトのランタイムインスタンスを作成します。モデルリポジトリ内のメタデータは更新しません。Developer tool 内のデータオブジェクトのランタイムインスタンスを表示するには、解決済みパラメータを使用してマッピングを表示します。

**注:** データソースからカラムを取得してマッピングフローに基づいてターゲットを定義するために書き込みトランスフォーメーションを設定した場合、マッピングは失敗します。

この機能は、リレーショナルターゲットの書き込みトランスフォーメーションの **【データオブジェクト】** タブで設定します。

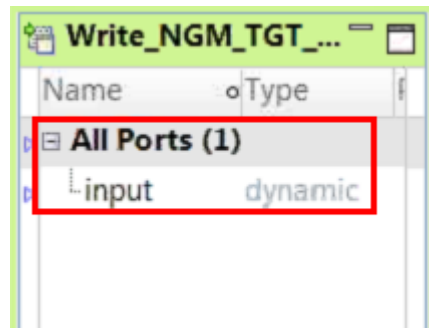
## マッピングフローに基づくターゲットの定義

マッピングフローによってカラムを定義する場合、ターゲットカラムはアップストリームトランスフォーメーションによって決定されます。アップストリームトランスフォーメーションによってポートの順序とメタデータが変更された場合、書き込みトランスフォーメーションはその変更を取得します。

アップストリームトランスフォーメーション内のキーカラムがターゲット内のキーカラム名に一致する場合、ターゲットを作成または置換するときにキーを維持できます。



次の画像は、マッピングフローに基づいてターゲットカラムを定義する場合に書き込みトランスフォーメーションがどのように表示されるかを示しています。



**注:** 予期しない結果にならないように、マッピングフローに基づいてターゲットを定義する書き込みトランスフォーメーションに対するランタイムリンクは設定しないでください。

この機能は、フラットファイルおよびリレーショナルターゲットの書き込みトランスフォーメーションの【ポート】タブで設定します。

## データオブジェクトに基づくターゲットの定義

書き込みトランスフォーメーションを設定して、関連付けられているデータオブジェクトに基づいてターゲットカラムを定義できます。

データオブジェクトに基づいてターゲットカラムを定義する場合、書き込みトランスフォーメーションには動的ポートと生成されたポートが含まれます。

実行時にターゲットを作成または置換することもできます。カラム名が一致する場合、ターゲットを作成または置換するときにターゲットキーを維持できます。カラム名が一致するようにルールを設定できます。

この機能は、フラットファイルおよびリレーショナルターゲットの書き込みトランスフォーメーションの【ポート】タブで設定します。

## ターゲットスキーマ戦略の定義

書き込みトランスフォーメーションでは、ターゲットテーブルのターゲットスキーマ戦略を定義できます。

次のターゲットスキーマ戦略オプションから選択できます。

### RETAIN - 既存のターゲットスキーマを維持

データ統合サービスは、既存のターゲットスキーマを維持します。

### CREATE - 実行時にテーブルを作成または置換

データ統合サービスは書き込みオブジェクトに関連付けられた既存のターゲットテーブルを削除し、データオブジェクトまたはマッピングフローに基づいてテーブルを作成します。

データ統合サービスがデータオブジェクトに基づいてテーブルを作成した場合、テーブルにはデータオブジェクトのポートに一致するカラムが含まれます。カスタマイズされたデータオブジェクトを使用して実行時にターゲットを作成または置換する場合、データ統合サービスはデータオブジェクト接続で参照される名前を使用してテーブルを作成します。

データ統合サービスがマッピングフローに基づいてテーブルを作成する場合、テーブルには書き込みトランスフォーメーションの生成されたポートに一致するカラムが含まれます。

### パラメータの割り当て

ターゲットスキーマ戦略の値を示すためのパラメータを割り当てて、実行時にパラメータを変更できます。

この機能は、データオブジェクトの【詳細】タブで設定します。



## DDL クエリの定義

実行時に、データ統合サービスで定義した DDL クエリに基づいてリレーショナルターゲットテーブルおよび Hive ターゲットテーブルを作成または置換できます。

書き込みトランスフォーメーション内のターゲットスキーマ戦略を設定してターゲットテーブルを作成または置換すると、デフォルトでデータ統合サービスにより、書き込みオブジェクトに関連付けられた既存のターゲットテーブルがすべて削除されます。次に、マッピングフローまたはデータオブジェクトに基づいてテーブルが作成されます。

テーブルをカスタマイズするか、パーティションなどの追加パラメータを指定する場合、DDL クエリを定義することができます。データ統合サービスは、この DDL クエリに基づいてターゲットテーブルを作成または置換します。テーブルには、DDL クエリで定義するカラムが含まれます。

パーティション化された Hive テーブルを作成するには、次の構文を使用します。

```
CREATE TABLE `hiveTable` (`Field` STRING) PARTITIONED BY ({INFA_PORT_SELECTOR: PortSelector})
```

次の図に、**[DDL クエリ]** フィールドを示します。

Properties		
General	Name	Value
	Tracing Level	Normal
Data Object	Target	
	Load type	Normal
Port Selectors	Target Schema Strategy	CREATE - Create or replace table at run time
Run-time	DDL query for create or replace	CREATE TABLE (INFA_TABLE_NAME) (INFA_COLUMN_LIST)
Run-time Linking	Truncate target table	<input type="checkbox"/>
	Truncate target partition	<input type="checkbox"/>
Advanced	PreSQL	<input type="checkbox"/>
	PostSQL	<input type="checkbox"/>
	Maintain row order	<input type="checkbox"/>

DDL クエリにはプレースホルダを入力できます。データ統合サービスにより、実行時にプレースホルダが実際の値に置き換えられます。例えば、テーブルに 50 個のカラムが含まれている場合、DDL クエリにすべてのカラム名を入力する代わりに、プレースホルダを入力できます。

DDL クエリには、次のプレースホルダを入力できます。

**INFA\_TABLE\_NAME**

実行時にターゲットテーブル名を取得します。

**INFA\_COLUMN\_LIST**

実行時にターゲットテーブルのカラムのリストを取得します。

**INFA\_PORT\_SELECTOR**

ポートセレクトを追加します。

プレースホルダは 2 つの中括弧で囲む必要があります。例えば、{INFA\_TABLE\_NAME} のようになります。

この機能は、データオブジェクトの **【詳細】** タブで設定します。

## 実行時にターゲットを作成または置換するためのルールおよびガイドライン

実行時にターゲットを作成または置換する場合、次のルールおよびガイドラインを検討してください。

- データベースのテーブルの中でもターゲットテーブルに循環型の依存関係がある場合は、データベースがテーブルを削除または作成するコマンドを実行できず、マッピングが失敗する。
- データ統合サービスがターゲットを置換した場合、ターゲットテーブルのインデックスと権限が保持されない。
- 動的ポートを含むように書き込みトランスフォーメーションを設定していない場合、データ統合サービスはデータオブジェクトに基づいてリンクされたポートとリンクされていないポートを含むターゲットを作成する。データはリンクされたポートに書き込まれます。

- データオブジェクト内のリソースがシノニムまたはビューであったとしても、データ統合サービスはテーブルを作成する。各接続は別々のデータベースインスタンスを指すことができますが、動的マッピングのすべての接続は、同じデータベースタイプである必要があります。

## リレーショナルターゲットのプロパティにパラメータを割り当てる

類似したリレーショナルターゲットを使用して動的マッピングを実行するために、書き込みトランスフォーメーションでリソース、接続、テーブル所有者プロパティにパラメータを割り当てることができます。

同じデータベースで異なるが類似しているテーブルを使用してマッピングを実行するには、リソースのパラメータを使用します。リソースのパラメータを使用する場合、各ターゲットのデータオブジェクトを作成する必要はありません。接続用のパラメータを使用して異なるデータベースにアクセスします。

物理データオブジェクトでこれらのプロパティはパラメータ化できません。書き込みトランスフォーメーションのプロパティのパラメータを作成する場合、マッピングパラメータを作成します。

デフォルトでは、接続の接続タイプパラメータを作成します。テーブル所有者のテーブル名と文字列パラメータにリソースタイプパラメータを設定します。

この機能は、リレーショナルターゲットの書き込みトランスフォーメーションの【ランタイム】タブで設定します。

## ターゲットデータオブジェクトにパラメータを割り当てる

カスタマイズされたデータオブジェクトにパラメータを割り当てて、実行時に書き込みトランスフォーメーションのソースを変更することができます。

複数のターゲットデータソースでモデルリポジトリにカスタマイズされたデータオブジェクトがある場合、データオブジェクトをパラメータ化します。パラメータの値を変更した場合、そのパラメータを使用するすべてのオブジェクトでターゲットが変更されます。

カスタマイズされたデータオブジェクトから書き込みトランスフォーメーションを作成した場合、トランスフォーメーションプロパティの【データオブジェクト】タブに、データオブジェクトに関する情報が表示されます。データオブジェクト名をクリックすると、モデルリポジトリから定義を表示することができます。データオブジェクトをパラメータ化するには、リソースタイプパラメータを作成するか、リソースパラメータを参照します。パラメータのデフォルト値は、モデルリポジトリ内のカスタマイズされたデータオブジェクトの名前です。デフォルトのパラメータ値を作成する場合、リポジトリ内のデータオブジェクトのリストからカスタマイズされたデータオブジェクト名を選択します。

データオブジェクトを変更すると、トランスフォーメーションポートが変更されます。トランスフォーメーションプロパティの【ポート】タブで、ポートを表示できます。

次の表に、【データオブジェクト】タブのパラメータオプションを示します。

パラメータオプション	説明
パラメータ	データオブジェクトとして設定したリソースパラメータの名前。読み取り専用。
説明	パラメータの説明。読み取り専用。
新規	リソースパラメータを作成します。パラメータのデフォルト値としてモデルリポジトリ内のデータオブジェクトを参照し、選択します。

パラメータオプション	説明
参照	リソースパラメータを参照し、パラメータを選択します。
デフォルト値	データオブジェクトに設定したリソースパラメータのデフォルト値。デフォルト値はカスタマイズされたデータオブジェクト名です。読み取り専用。

この機能は、リレーショナルターゲットの書き込みトランスフォーメーションの【データオブジェクト】タブで設定します。

## 動的ターゲットのルールおよびガイドライン

動的ターゲットに関する作業を行うときは、次のルールおよびガイドラインを検討してください。

- 動的ターゲットをプレビューする場合、Developer tool はスキーマ定義を更新しない。データソースからカラムを取得する設定や実行時にターゲットを置換する設定など、スキーマの変更に起因する不一致がある場合、データのプレビューは失敗します。読み取りまたは書き込みトランスフォーメーションを手動で同期します。エラーが解消されない場合、マッピングを実行して結果を確認します。
- 動的ターゲットが入力データに対して小さすぎる場合、マッピングは失敗し、値がカラムに対して大きすぎることを示すメッセージが表示される。
- ターゲットテーブルのデータ型は、書き込みトランスフォーメーションのデータ型と異なる場合がある。データ統合サービスがマッピングを実行すると、アップストリームトランスフォーメーションとターゲットテーブル間でデータ型が変更される場合があります。

## 動的ポートおよび生成されたポート

トランスフォーメーションで動的ポートを作成して、アップストリームトランスフォーメーションから新規または変更されたカラムを受信することができます。動的ポートは 1 つ以上のカラムを受信し、入力ルールに基づいてポートを生成します。入力ルールにより、動的ポートが受信して生成するカラムが決定されます。

動的ポートは、次のタスクの実行に使用します。

**新規または変更されたカラムを受信する。**

動的ソースまたはパラメータ化されたソースからデータを取得するために、ダウストリームトランスフォーメーションに動的ポートを作成して、新規または変更されたカラムを受信します。マッピングに動的ソースが含まれている場合、ダウストリームトランスフォーメーションの動的ポートは、自動的に新規または変更されたカラムを取得します。例えば、新規カラム「title」を動的ソースに追加すると、読み取りトランスフォーメーションはこの新規カラムを動的ポートに渡して、動的ポートは「title」カラムに対して生成されたポートを作成します。

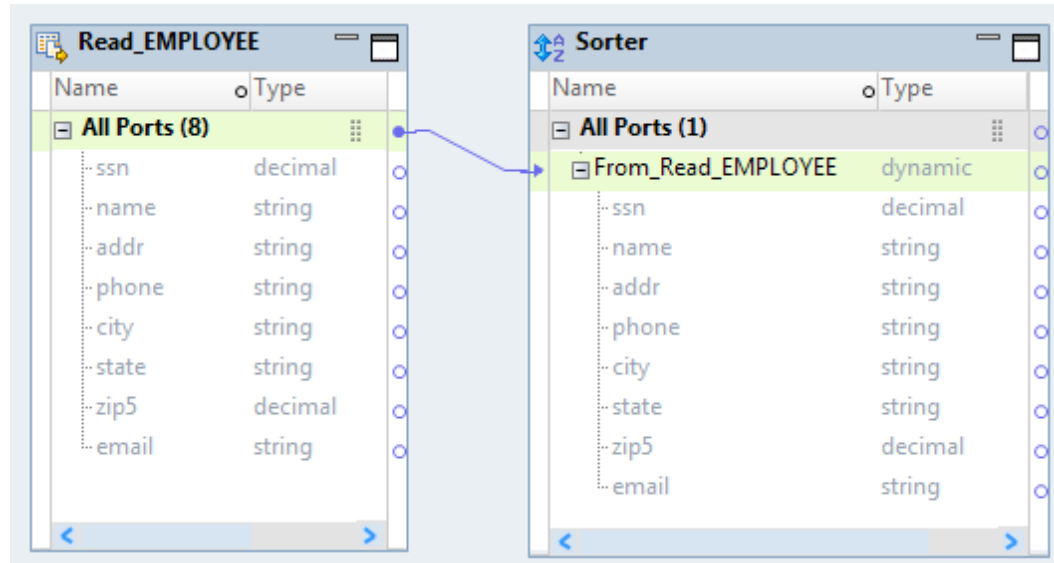
**入力ルールに基づいてカラムをフィルタリングする。**

トランスフォーメーションの特定タイプのカラムのみを処理するために、動的ポートを作成し、カラムをフィルタリングする入力ルールを定義します。例えば、マッピングソースに decimal、string、および date/time データ型のカラムがあるとしします。decimal データ型のカラムのみでデータを処理する必要があります。この場合に、動的ポートを作成して、decimal 型のカラムのみを含める入力ルールを定義します。

式トランスフォーメーションの複数ポートで同じ計算を繰り返す。

複数のポートに対して同じ計算を実行するために、動的式で動的ポートを使用します。動的ポートの各ポートに対して動的式は1回実行され、動的出力ポートに結果が返されます。

次の画像は、[From\_Read\_Employee] という動的ポートと生成されたポートを示しています。



## 動的ポートおよび生成されたポートの設定

読み取りトランスフォーメーションの [すべてのポート] グループ、アップストリームトランスフォーメーションのグループ、またはアップストリームトランスフォーメーションの動的ポートから動的ポートを作成できます。Developer tool はデータ型の値が動的である動的ポートを作成します。トランスフォーメーションでは、複数の動的ポートを作成できます。

**【新規】** ボタンを使用してポートを作成した場合は、Developer tool によってデフォルト名が割り当てられます。動的ポートの名前を変更して、各トランスフォーメーション内でポート名が一意になるようにします。同じ名前のポートをトランスフォーメーションに追加した場合、Developer tool はポート名の競合を解決するために動的ポートまたは生成されたポートに数字を追加します。

次の各トランスフォーメーションで動的ポートを作成できます。

- アグリゲータ
- 式
- フィルタ
- ジョイナ
- ルックアップ
- ランク
- 読み取り
- ルータ
- シーケンスジェネレータ
- ソータ
- アップデートストラテジ
- 書き込み

マッピングに動的ポートを含むことができないトランスフォーメーションが含まれる場合、ソースメタデータが変更されたときにマッピングを手動で更新する必要がある場合もあります。

**注:** ポート属性への変更はすべて、パイプラインの生成されたポートにプロパゲートされます。変更されたポート属性を手動でプロパゲートする必要はありません。

## 動的ポートおよび生成されたポートのルールおよびガイドライン

動的ポートおよび生成されたポートに関する作業を行うときは、次のルールおよびガイドラインを検討してください。

- 生成されたポートを仮想テーブルマッピングの出力トランスフォーメーションにリンクすることはできない。
- 生成されたポートを操作マッピングのフォールト、入力、または出力トランスフォーメーションにリンクすることはできない。

## 動的式

動的出力ポートの式を設定すると、式が動的式になります。動的式では複数の出力ポートを生成できます。

動的式では、ポートセクタまたは動的ポートを参照できます。ポートセクタまたは動的ポートに複数のポートが含まれる場合、動的式は各ポートに対して実行されます。

動的式を設定すると、Developer tool は、生成されたポートが式に対して有効なタイプかどうかを検証しません。例えば、string 型を必要とする式で decimal 型ポートを含むポートセクタを参照する場合、設計時に式は有効と表示されます。

### 例

式トランスフォーメーションには、生成された以下の入力ポートがあります。

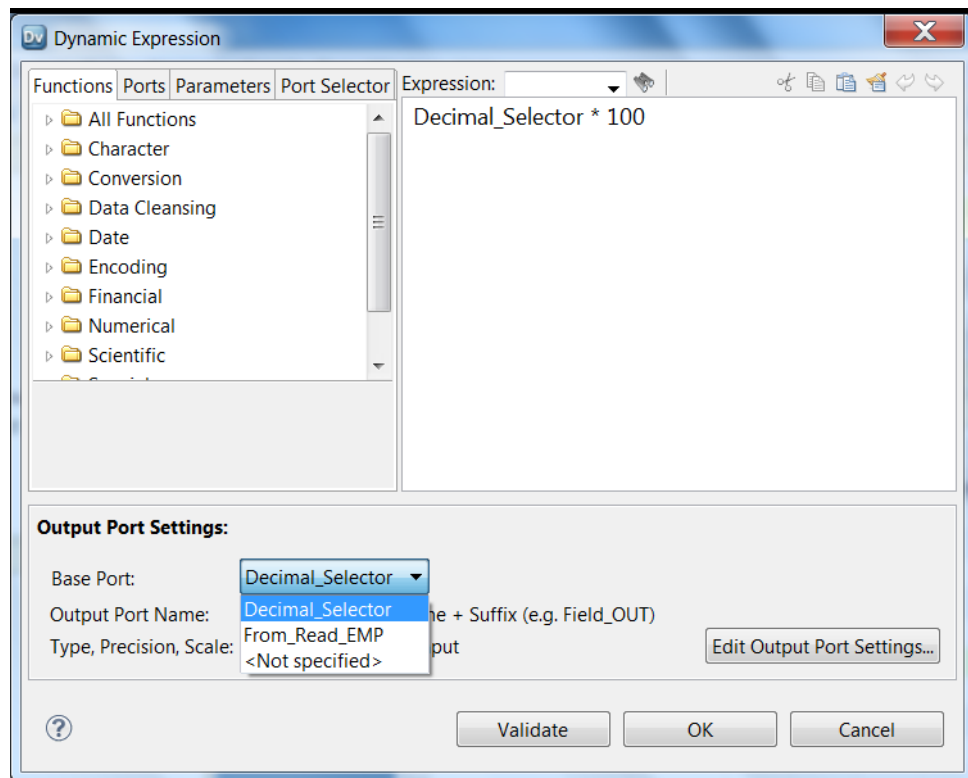
```
EMPNO    Decimal
NAME      String
SALARY    Decimal
DEPTNO    Decimal
```

トランスフォーメーションには MyDynamicPort という名前の動的出力ポートが含まれます。出力ポートは動的式の結果を返します。動的式は、ポートセクタの各ポートの値に 100 をかけます。式はポートセクタの各ポートに対して 1 回実行されます。各インスタンスは異なる結果を返すことがあります。式トランスフォーメーションは、各結果に対して別の出力ポートを生成します。

Decimal\_Selector ポートセクタには、decimal データ型のポートを含む選択ルールがあります。

```
EMPNO    Decimal
SALARY    Decimal
DEPTNO    Decimal
```

次の図は、Decimal\_Selector ポートセクタを参照する動的式を示しています。



出力ポート設定を編集して、出力ポート名と出力ポートプロパティを変更します。また、ベースポートも選択できます。

## 動的式のパラメータ化

アグリゲータ、式、リンクトランスフォーメーションで動的式を使用する場合、式パラメータを作成できます。式パラメータには完全な式が含まれています。

### 例

あなたはインフラストラクチャセクタのコンサルティング会社に勤務しています。この会社では、建物の構造のさまざまな場所に対して気象現象がもたらす影響に関するデータを収集しています。動的マッピングを使用して、収集したデータを変換し、与えられた影響の値を丸める必要があります。

動的マッピングでは、次の生成済み入力ポートを受信する式トランスフォーメーションを使用します。

```
Force_Location_A    Decimal
Force_Location_B    Decimal
Force_Location_C    Decimal
```

選択ルールを持つポートセレクタ Port\_Selector\_Force を作成し、プレフィックス Force\_を持つポートを含めます。動的出力ポート Dynamic\_Forces も作成し、動的式の式パラメータを使用します。式パラメータのデフォルト値は、次の式です。

```
CEIL(Port_Selector_Force)
```

複数のパラメータセットで式パラメータを使用することもできます。例えば、パラメータセット内の式パラメータのデフォルト値が CEIL(Port\_Selector\_Force) で、別のパラメータセットでは式パラメータのデフォルト値が FLOOR(Port\_Selector\_Force) の場合があります。

# 入カルール

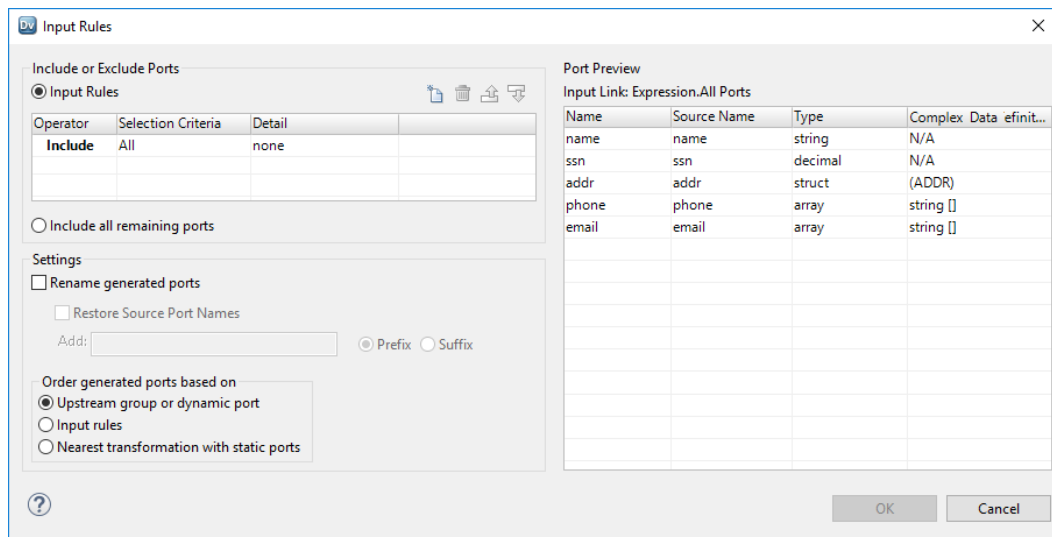
入カルールとは、動的ポート内の生成するポートを定義する条件のセットです。動的ポートが受信するカラムをフィルタリングして、フィルタリングされたカラムに生成されたポートを作成するよう、入カルールを設定できます。入カルールを使用してマッピングの特定カラムをダウンストリームにプロパゲートします。

動的ポートの特定のカラムまたは特定タイプのカラムを処理するには、名前、データ型、パターン、または複合データ型定義に基づいてカラムをフィルタリングするよう、入カルールを定義します。例えば、従業員データベーステーブルに decimal、string、および date/time データ型のカラムがあるとします。SAL で始まりデータ型が decimal のカラムのみでデータ処理を行う必要があります。動的ポートを作成して、この条件を満たすカラムのみを含めるよう入カルールを定義します。

## 入カルールの設定

**【入カルール】** ダイアログボックスを使用して、含めるポートの定義、生成されたポートの名前の変更、生成されたポートの順序変更、およびルールの結果の表示を行います。

次の画像は、アップストリームトランスフォーメーションからのすべてのポートを含めるデフォルトの入カルールを設定した **【入カルール】** ダイアログボックスを示しています。



入カルールを設定した場合、次のプロパティを設定します。

### ポートを含めるまたは除外する

ポート名またはデータ型に基づき、動的ポートに含めるまたは除外するポートを指定します。複数のルールを定義できます。データ統合サービスは、入カルールリストに表示される順序でルールを適用します。デフォルトの入カルールではすべてのポートが含まれます。動的ポートの「含む」入カルールを 1 つ以上作成します。

### 残りのすべてのポートを含める

トランスフォーメーションの他の動的ポートから除外されたポートを追加します。トランスフォーメーションに複数の動的ポートが含まれる場合、最後の動的ポートのアップストリームトランスフォーメーションから残りのすべてのポートを含めることができます。

### 生成されたポートの名前の変更

生成されたポート名にプレフィックスまたはサフィックスを追加します。ポートが生成されるトランスフォーメーションを示すか、各トランスフォーメーション内でポート名が一意になるようにするには、プレフィックスまたはサフィックスを使用します。



## ソースポート名のリストア

生成されたポートでソースポート名をリストアします。Developer tool は、静的ポートを使用する最も近いアップストリームトランスフォーメーションに表示されるポート名をリストアします。

## 生成されたポートの順序変更

入力したルールの順序に従って生成されたポートを表示します。デフォルトでは、Developer tool はアップストリームトランスフォーメーションに表示されるのと同じ順序でポートを表示します。読み取りトランスフォーメーションでのポートの順序に応じて、順序を変更することもできます。ただし 1 つ以上のミッドストリームトランスフォーメーションに動的ポートと静的ポートがある場合、Developer tool は、静的ポートを使用する最も近いアップストリームトランスフォーメーションに表示されるのと同じ順序でポートを表示します。

ルールを設定した後で、生成されたポートをプレビューしてルールの組み合わせを確認できます。データ統合サービスは、**【入力ルール】** ダイアログボックスに表示される順序でルールを評価します。正しい順序で実行されるように、ルールの順序を変更できます。

# ポートを含めるまたは除外する

ポート名またはデータ型に基づいてポートを含めるまたは除外することができます。各入力ルールでは演算子および選択条件を使用してポートをフィルタリングします。複数のルールを定義できます。データ統合サービスは、入力ルールリストに表示される順序でルールを適用します。デフォルトの入力ルールではすべてのポートが含まれます。

次の入力ルールを設定して、含めるまたは除外するポートを決定します。

## 演算子

ポートを含めるか除外するかを決定します。デフォルトの設定ではポートを含めます。

## 選択条件

ポート名またはデータ型に基づいてポートをフィルタリングするかどうかを決定します。選択条件を選択した場合、条件に基づいて入力ルールの詳細ダイアログボックスが表示されます。例えば、**【名前】** 選択条件の詳細は **【入力ルールの詳細: 名前リスト順】** ダイアログボックスで指定します。

## 詳細

入力したポート名またはデータ型の詳細に基づいてフィルタリングするポートを決定します。

次の表に、選択条件と条件の詳細を指定する方法を示します。

選択条件	説明	条件の詳細
すべて	すべてのポートを含めます。除外演算子とともにこの選択条件を使用しないでください。	詳細を指定する必要はありません。
名前	ポート名に基づいてポートをフィルタリングします。	値のリストからポート名を選択するか、[ポート] または [ポートリスト] のパラメータを使用します。ソース名によってポート名を選択することもできます。 注: 名前の値の大文字と小文字は区別されません。
タイプ	ポートのデータ型に基づいてポートをフィルタリングします。	リストからデータ型を選択します。



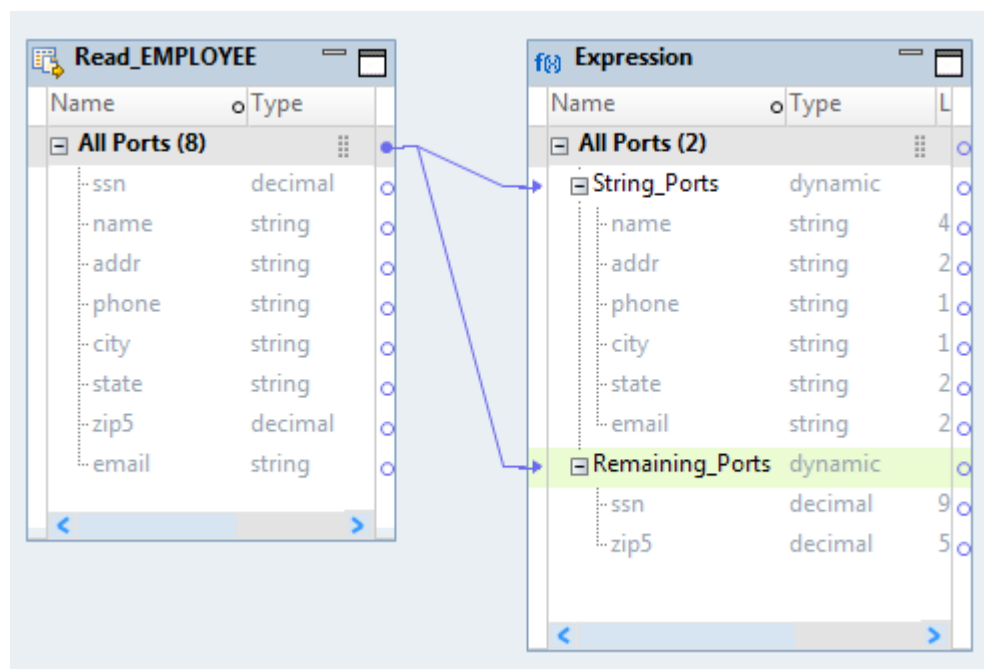
選択条件	説明	条件の詳細
パターン	ポート名のパターンに基づいてポートをフィルタリングします。	ポート名のパターンタイプとして、プレフィックス、サフィックス、または正規表現を選択します。次に、パターンの値を入力するか、文字列タイプのパラメータを使用します。ソース名によってポートを選択し、ソース名のパターンを検索することもできます。 注: パターンの値の大文字と小文字は区別されません。
複合データ型定義	複合データ型定義に基づいてポートをフィルタ処理します。	複合データ型定義のパターンタイプとして、プレフィックス、サフィックス、または正規表現を選択します。次に、パターンの値を入力するか、文字列タイプのパラメータを使用します。 注: パターンの値の大文字と小文字は区別されません。

## 残りのすべてのポートを含める

トランスフォーメーションに複数のポートが含まれる場合、他の動的ポートに含まなかったすべてのポートを含めるように最後の動的ポートを設定できます。

例えば、テーブルの string 型カラムから先頭の空白を削除し、他のすべてのカラムに対するデータとともに string 型データの出力をターゲットに書き込む場合を想定します。式トランスフォーメーションで、2 つの動的ポートを作成します。一方のポートにすべての string 型データを含めて、残りのすべてのデータをもう一方のポートに配置するように入力ルールを設定します。最後の動的ポートの **【残りのすべてのポートを含める】** オプションを選択します。

次の画像は、式トランスフォーメーションの 2 つの動的ポート String\_Ports および Remaining\_Ports を示しています。

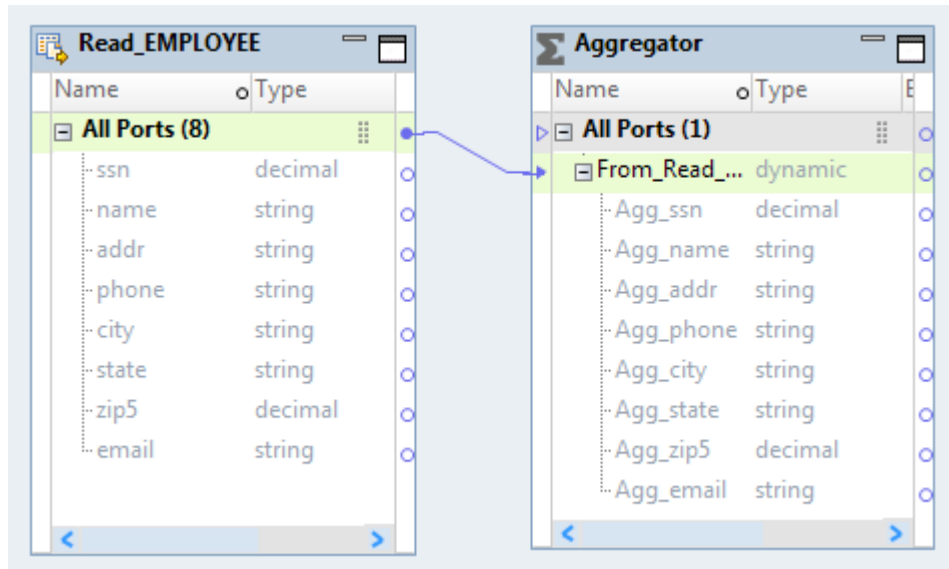


## 生成されたポートの名前の変更

生成されたポートにプレフィックスまたはサフィックスを付けて名前を変更する設定を使用して、一意のポート名を生成できます。

例えば、Agg\_プレフィックスを追加すれば、アグリゲータトランスフォーメーションで生成されたポートであることを示すことができます。

次の画像は、アグリゲータトランスフォーメーションの名前の変更により Agg\_プレフィックスがついた生成されたポートを示しています。



同じ名前のポートをトランスフォーメーションに追加する場合、Developer tool は生成されたポートに数字を追加してポート名の競合を解決します。データ統合サービスが実行時にポート名の競合を解決できない場合、生成されたポートの名前を変更する必要がある場合もあります。マッピングで動的ソースを使用していると、データ統合サービスで実行時にポート名の競合が発生する場合があります。データ統合サービスでポート名の競合が発生した場合、生成されたポートの名前を変更しようと試みます。データ統合サービスがポート名の競合を解決できないと、マッピングは失敗します。マッピングが失敗するのは次のような場合です。

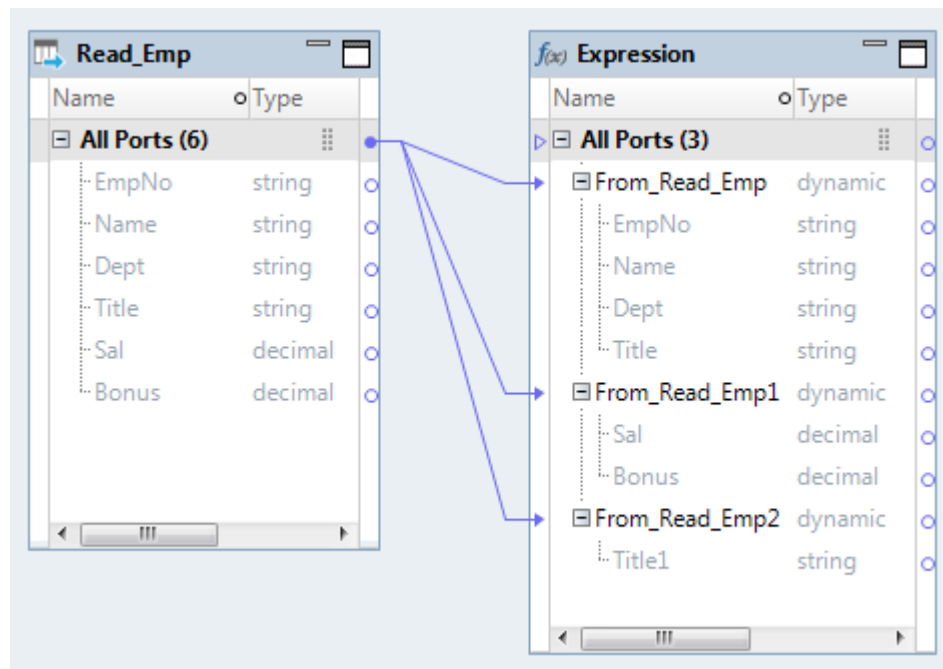
- 名前が変更された生成されたポートから静的ポートへの未解決のリンクが存在する。
- グループ化ポートや結合条件などのトランスフォーメーションプロパティで名前が変更された生成されたポートを使用している。

マッピングエラーを回避するには、各トランスフォーメーション内で一意になるように生成されたポートの名前を変更します。

## 生成されたポートの名前の変更の例

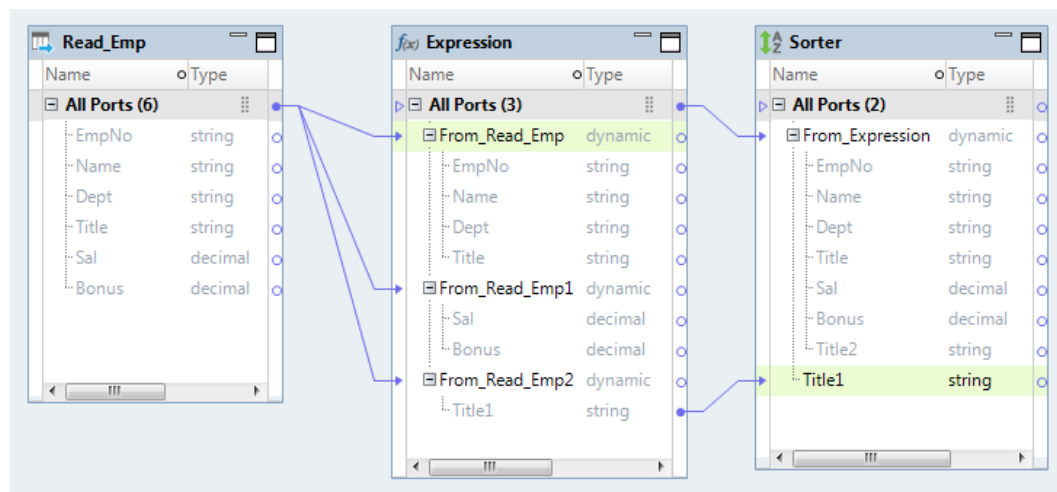
式トランスフォーメーションには3つの動的ポートがあります。動的ポート [From\_Read\_Emp] および [From\_Read\_Emp2] には、生成されたポート「Title」が含まれます。名前の競合を回避するため、Developer tool は [From\_Read\_Emp2] の生成されたポートの名前を「Title1」に変更します。

次の画像は、式トランスフォーメーションの名前が変更された生成されたポート Title1 を示しています。



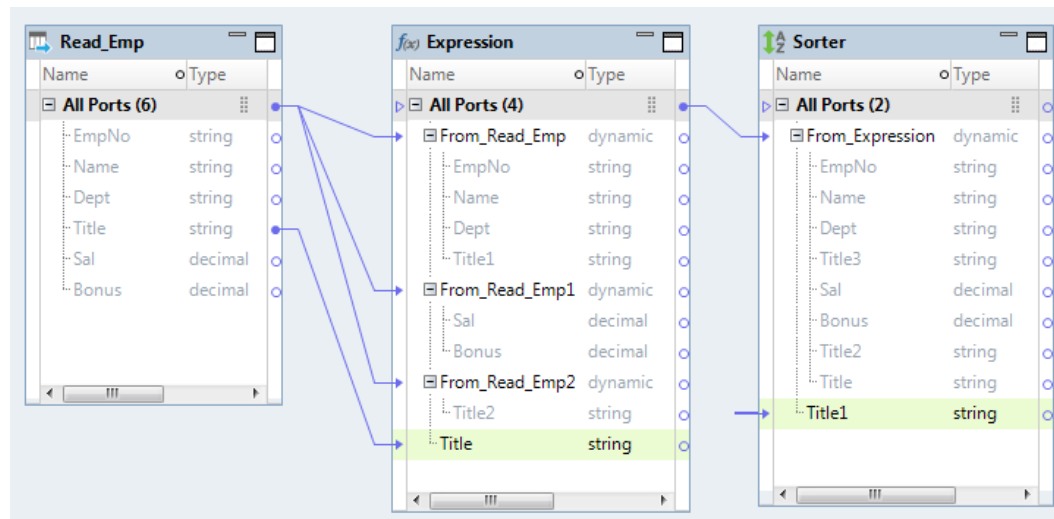
式トランスフォーメーションの生成されたポート「Title1」をソータートランスフォーメーションのポート「Title1」にリンクします。また、ソートキーとしても「Title1」を使用します。

次の画像は、式トランスフォーメーションの生成されたポートからソータートランスフォーメーションのポートへのリンクを示しています。



読み取りトランスフォーメーションのポート「Title」から式トランスフォーメーションのポート「Title」へのリンクを追加する場合、Developer tool は生成されたポートの名前を変更します。動的ポート [From\_Read\_Emp] の生成されたポートを「Title1」に名前変更します。動的ポート [From\_Read\_Emp2] の生成されたポートを「Title2」に名前変更します。ソータートランスフォーメーションの「Title1」へのリンクは未解決と表示されます。

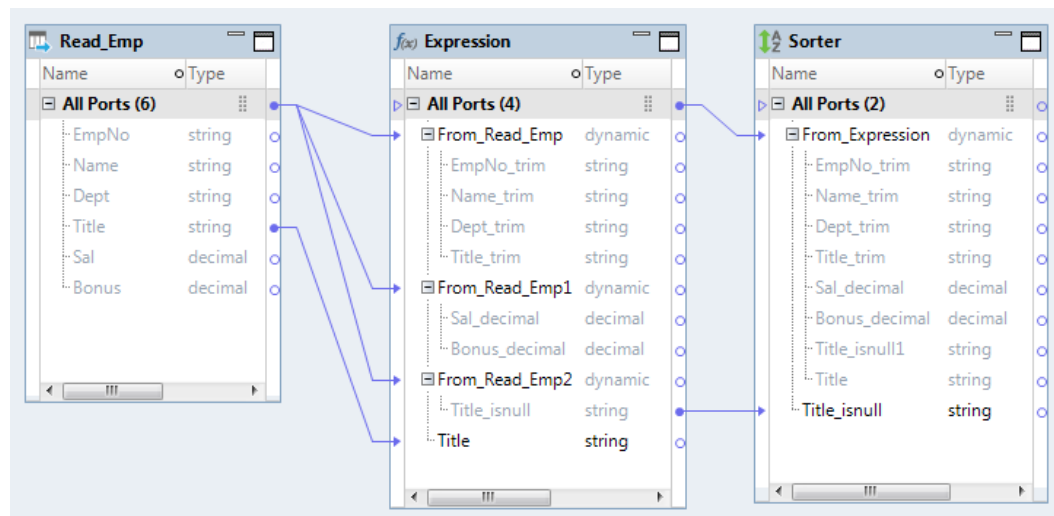
次の画像は、読み取りトランスフォーメーションと式トランスフォーメーション間の新しいリンク、Developer tool が式トランスフォーメーションで名前変更した生成されたポート、およびソータートランスフォーメーションへの未解決のリンクを示しています。



ソートキーに使用されている生成されたポートが使用を意図したポートではない可能性があるため、マッピングは実行時に失敗します。

マッピングエラーを回避するには、各トランスフォーメーション内で一意になるように生成されたポートの名前を変更します。例えば、動的ポート [From\_Read\_Emp] の文字列ポートの先頭の空白を切り捨てるとします。サフィックス\_trim を生成されたポートに追加します。動的ポート [From\_Read\_Emp2] のポートに NULL 値があるかどうか判定する必要があります。サフィックス\_isnull を生成されたポートに追加します。

次の画像は、式トランスフォーメーションの生成されたポートが名前変更された様子を示しています。



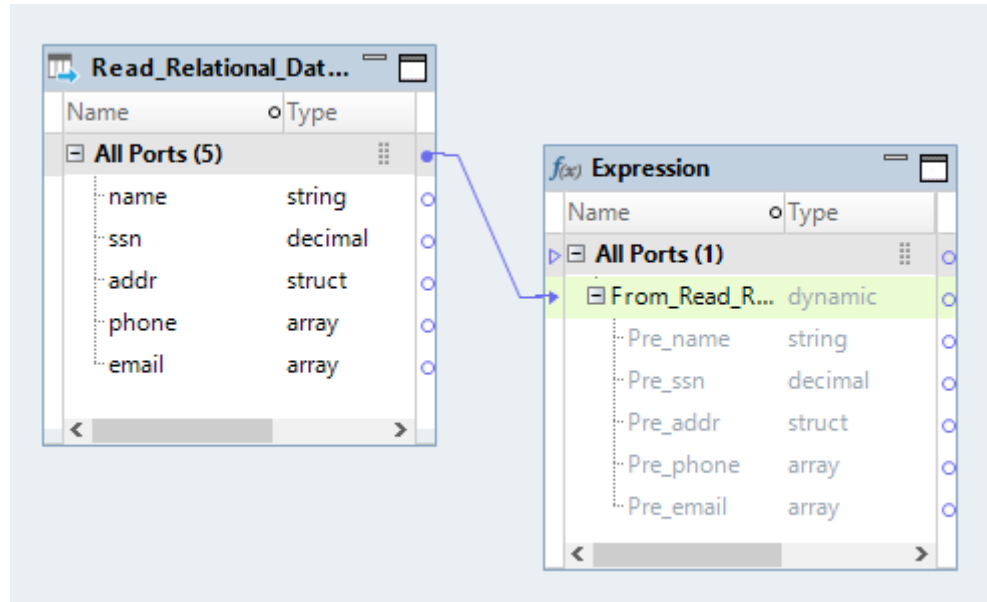
## ソースポート名のリストア

生成されたポートの名前を変更するときにソースポート名をリストアできます。

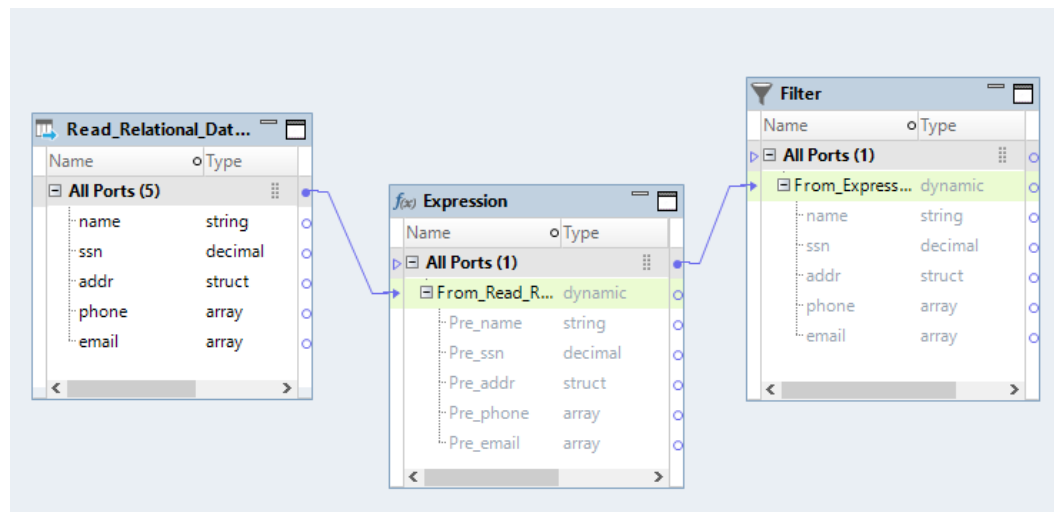
例えば、生成されたポートの名前を変更して、マッピングフロー内の特定のトランスフォーメーションのすべてのポートにプレフィックス Pre\_を追加できます。ダウンストリームトランスフォーメーションでは、ソースポート名をリストアして、プレフィックスを追加する前に名前をリストアできます。ソースポート名をリスト

アすると、Developer tool は、静的ポートを使用する最も近いアップストリームトランスフォーメーションに表示されるポート名をリストアップします。

次の図は、生成されたポートの名前がプレフィックス Pre\_ を使用して変更された式トランスフォーメーションを示しています。



次の図は、ソースポート名がリストされたフィルタトランスフォーメーションを示しています。



ポート名は、静的ポートを含むアップストリーム読み取りトランスフォーメーションに従ってリストアップされます。

## 生成されたポートの順序変更

入力ルール の順序または読み取りトランスフォーメーションのポートの順序に基づいてポートを順序変更する設定を使用して、生成されたポートを順序変更できます。デフォルトでは、Developer tool はアップストリームトランスフォーメーションに表示されるのと同じ順序で生成されたポートを表示します。

次のオプションの 1 つを選択して、生成されたポートを順序変更できます。

## アップストリームグループまたは動的ポート

アップストリームトランスフォーメーションのグループまたは動的ポートに表示されるのと同じ順序でポートを表示します。これがデフォルトのオプションです。

## 入力ルール

動的ポートの入力ルールの順序に基づいて、生成されたポートを表示します。

データ統合サービスは、**【入力ルール】** ダイアログボックスに表示された順序でルールを読み取ります。ポートの順序を確認して、入力ルールの順序に基づいて順序を変更してください。データ統合サービスがポートおよびルールを要求に応じた順序で処理することができます。ポートを順序変更すると、結果の表示および分析もしやすくなります。

## 静的ポートを使用する最も近いトランスフォーメーション

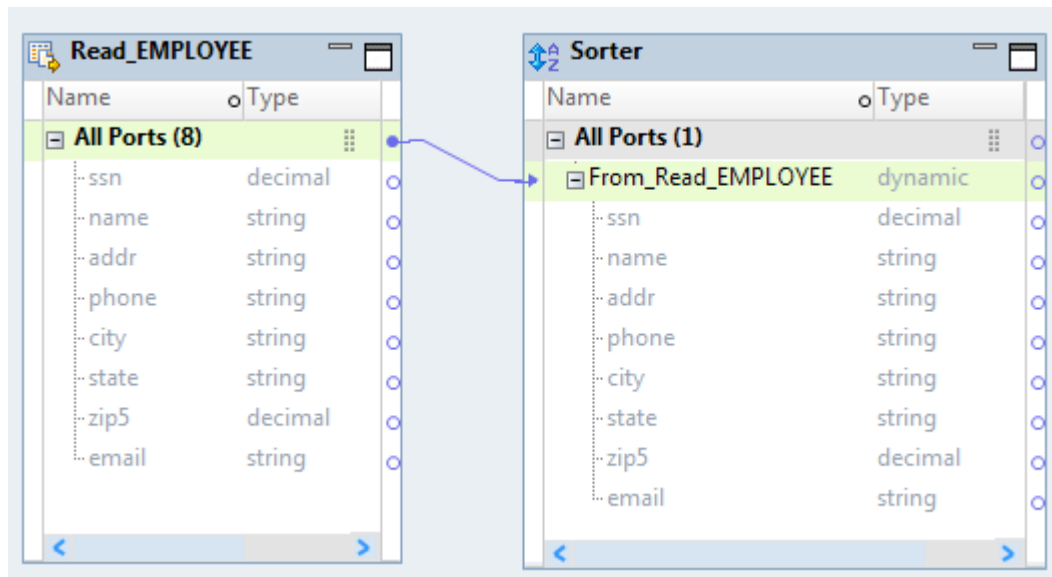
読み取りトランスフォーメーションのポートの順序に基づいて、生成されたポートを表示します。

このオプションに基づいてポートを順序変更すると、ソースでポートの元の順序を維持しやすくなります。ただし 1 つ以上のミッドストリームトランスフォーメーションに動的ポートと静的ポートがある場合、Developer tool は、静的ポートを使用する最も近いアップストリームトランスフォーメーションに表示されるのと同じ順序でポートを表示します。このオプションは、マッピングに単一のパイプラインがある場合に限り有効です。

## 生成されたポートの順序変更の例

従業員フラットファイルソースには頻繁に変更されるカラムが多数あります。ユーザーは、従業員を名前でソートし、先頭のカラムに従業員名、次のカラムに従業員が勤務する市区町村が出現するように従業員データを表示する必要があります。また、decimal 型タイプのカラムは、データ分析の対象外であるため、末尾に移動する必要があります。

次の画像は、動的ポート [From\_Read\_EMPLOYEE] を示しています。生成されたポートは、元の順序で表示されています。



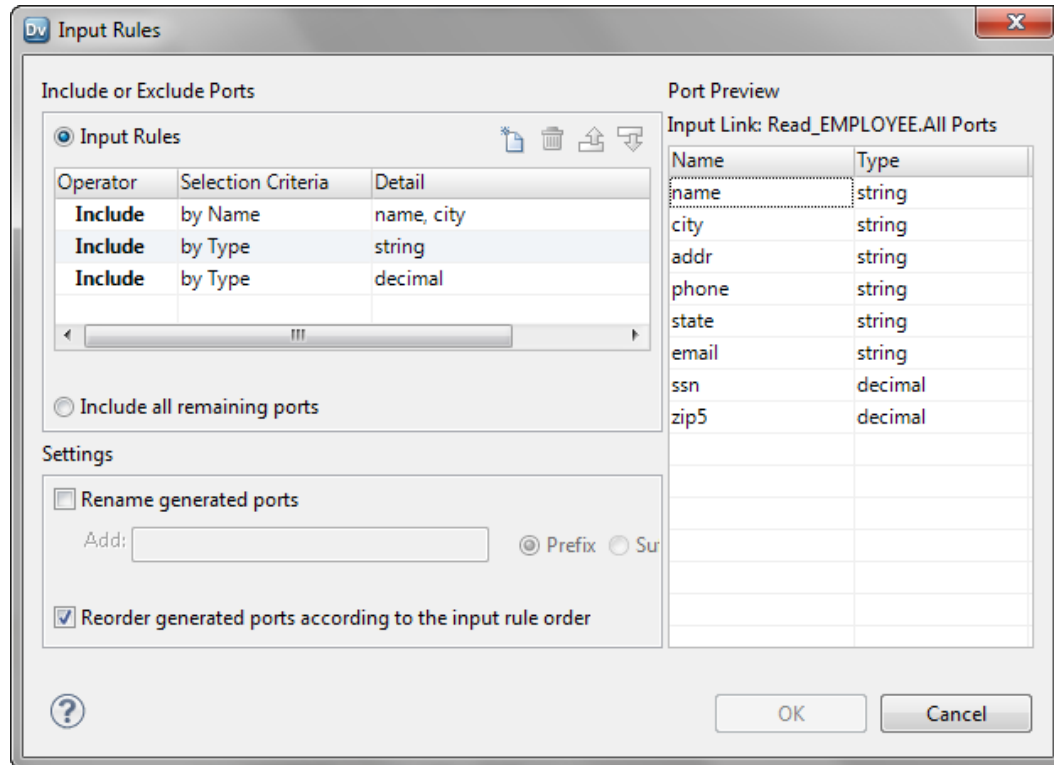
次の入力ルールを設定します。

- 「名前」と「市区町村」によってポートを含める。
- string 型によってすべてのポートを含める。

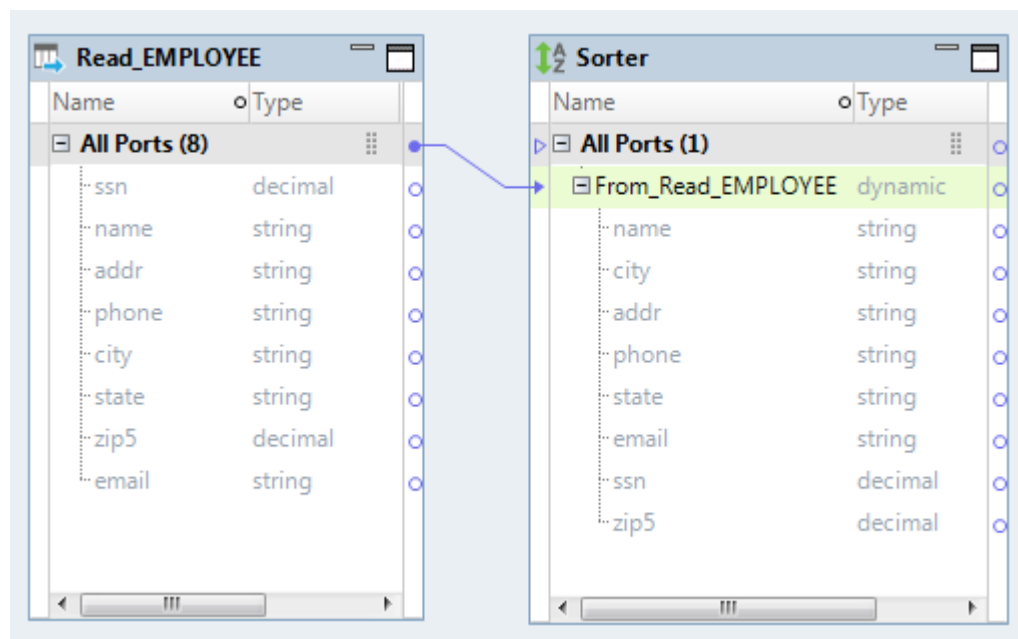
- decimal 型によってすべてのポートを含める。

次に、入力ルールに基づいてポートの順序を変更します。ポートの順序は、プレビューで確認できます。

次の画像は、入力ルールの順序を示しています。



次の画像は、入力ルール設定に基づいて順序変更された生成されたポートを示しています。



# 選択ルールとポートセレクト

トランスフォーメーションでポートを生成した場合、生成したポートが変更されてもトランスフォーメーションが正しく実行されるように設定する必要があります。ポートセレクトを使用して、動的式、ルックアップ条件、またはジョイナ条件でどのポートを使用するかを決定できます。

ポートセレクトとは、ポートを定義する選択ルールのセットです。ポートセレクトは式内で参照します。生成したポートが動的マッピング内で変更された場合、ポートセレクトに異なるポートが含まれている可能性があります。ポートセレクトは、式トランスフォーメーション、ルックアップトランスフォーメーション、またはジョイナトランスフォーメーション内で作成できます。これらのトランスフォーメーションには、ポートセレクト内のすべてのポートを参照できる式が含まれます。

以下のマッピングオブジェクトでポートセレクトを設定できます。

## 式トランスフォーメーション

動的式内でポートセレクトを参照できます。式内でポートセレクトを参照すると、その式がポートセレクト内の各ポートに対して実行されます。動的式は、ポートセレクト内のポートごとに別個の出力ポートに結果を返します。トランスフォーメーションにポートセレクトを参照する複数の式が含まれている場合、そのトランスフォーメーションは、式ごとに追加の出力ポートを返します。

## ジョイナトランスフォーメーション

1つの結合条件内で2つのポートセレクトを参照できます。マスタグループ用のポートセレクトと詳細グループ用のポートセレクトを定義します。データ統合サービスは、ポートセレクト内のポート順に従って、マスタグループ内の各ポートを詳細グループ内のポートと比較します。比較するポートのペアごとに1つのタイプの演算子を選択できます。各ポートセレクトには、同じ数のポートが含まれている必要があります。

例えば、ポート A、B、C を含むポートセレクト Master-SelectorX、およびポート D、E、F を含むポートセレクト Detail-SelectorY を設定するとします。このとき、Master-SelectorX = Detail-SelectorY という結合条件を指定すると、Developer tool によって  $A = D \text{ AND } B = E \text{ AND } C = F$  という結合条件が作成されます。

## ルックアップトランスフォーメーション

ポートのポートセレクトはルックアップ条件で設定できます。データ統合サービスは、各ポートセレクト内のポート順に従って、入力ポートセレクト内の各ポートをルックアップポートセレクト内のポートと比較します。各ポートセレクトには、同じ数のポートが含まれている必要があります。

## 書き込みトランスフォーメーション

書き込みトランスフォーメーションでポートのポートセレクトを設定できます。

データをリレーショナルターゲットまたは Hive ターゲットに書き込む場合、実行時にターゲットテーブルを作成または置換できます。DDL クエリを定義できます。データ統合サービスは実行時にこの DDL クエリに基づいてターゲットテーブルを作成または置換する必要があります。DDL クエリでポートセレクトを設定することもできます。

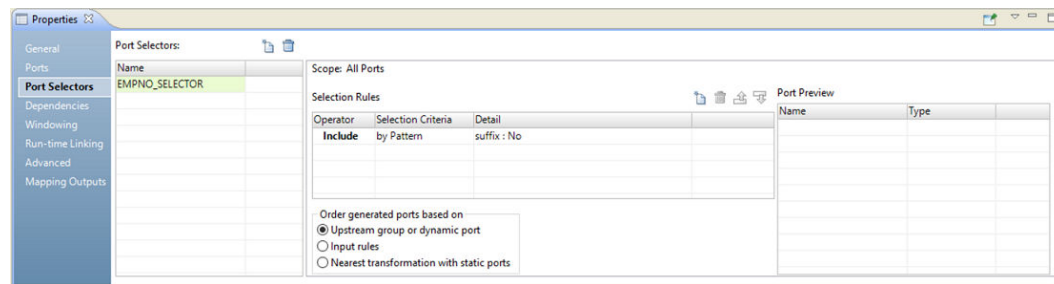
## ポートセレクトの設定

ポートセレクトを設定するには、選択ルールを定義して、含める生成済みポートを決定します。選択ルールは、動的ポートに設定できる入力ルールと同じです。

ポートセレクトには、静的または生成済みポートを含めることができます。ポートセレクトの設定は、**ポートセレクト**タブで行います。



次の図は、[ポートセレクト] タブを示しています。



ポートセレクトに次のプロパティを設定します。

### 名前

ポートセレクトを識別します。1つのトランスフォーメーションに複数のポートセレクトを作成し、式で参照できます。

### スコープ

ポートセレクトが適用されるポートグループを識別します。ジョイナまたはルックアップトランスフォーメーションのポートセレクトを作成する場合、スコープを選択する必要があります。これらのトランスフォーメーションには複数の入力グループがあります。ジョイナトランスフォーメーションにはマスタまたは詳細スコープがあります。ルックアップトランスフォーメーションにはインポートまたはルックアップスコープがあります。式トランスフォーメーションには入力グループが1つあります。スコープは常に[すべてのポート]です。

### 選択ルール

ポートセレクトに含めるポートを決定します。選択ルールを作成すると、[ポートのプレビュー] パネルに現在の入力ポートのうち適格なポートが表示されます。これらのポートは変わる可能性があります。さまざまなソースからのポートに対応できるように選択ルールを設定します。

## 選択ルール

ポートセレクトに関連付けられた選択ルールでポートセレクトに含めるポートを決定します。

選択ルールを作成すると、[ポートのプレビュー] パネルに現在の入力ポートのうち適格なポートが表示されます。これらのポートは変わる可能性があります。さまざまなソースからのポートに対応できるように選択ルールを設定します。

次の条件に基づいて選択ルールを作成します。

### 演算子

選択ルールが返すポートを含めるか、除外します。デフォルトは「含める」です。ポートを除外する前にポートを含める必要があります。

### 選択条件

作成する選択ルールのタイプ。カラム名、ポートタイプ、パターン、または複合データ型定義に基づいてルールを作成できます。カラム名に基づいてポートを含めるには、特定の名前を検索するか、名前に含まれる文字列パターンを検索します。

### 詳細

選択条件に適用する値。選択条件がカラム名基準になっている場合は、検索する文字列または名前を設定します。選択条件がポートタイプ基準になっている場合は、含めるポートタイプを選択します。

次の表に、選択条件と条件の詳細を指定する方法を示します。

選択条件	説明	詳細
すべて	すべてのポートを含めます。	詳細は不要です。
名前	ポート名に基づいてポートをフィルタリングします。	値のリストからポート名を選択するか、[ポート] または [ポートリスト] のパラメータを使用します。
タイプ	各ポートのデータ型に基づいてポートをフィルタリングします。	リストからデータ型を選択します。
パターン	名前に含まれる文字列または正規表現を使用してポートをフィルタリングします。	ポート名のパターンタイプとして、プレフィックス、サフィックス、または正規表現を選択します。次に、パターンの値を入力するか、文字列タイプのパラメータを使用します。
複合データ型定義	複合データ型定義を使用してポートをフィルタリングします。	複合データ型定義のパターンタイプとして、プレフィックス、サフィックス、または正規表現を選択します。次に、パターンの値を入力するか、文字列タイプのパラメータを使用します。

## 選択ルールとポートセレクトアの例

動的ソースを使用するようにマッピングを設定できますが、各ソースファイルの給与情報が含まれるカラムは名前が異なります。それぞれのソースのカラム名は Salary、Monthly\_Salary、または Base\_Salary です。

給与を示すあらゆるポート名を持つ式を実行するために、次のタスクを実行します。

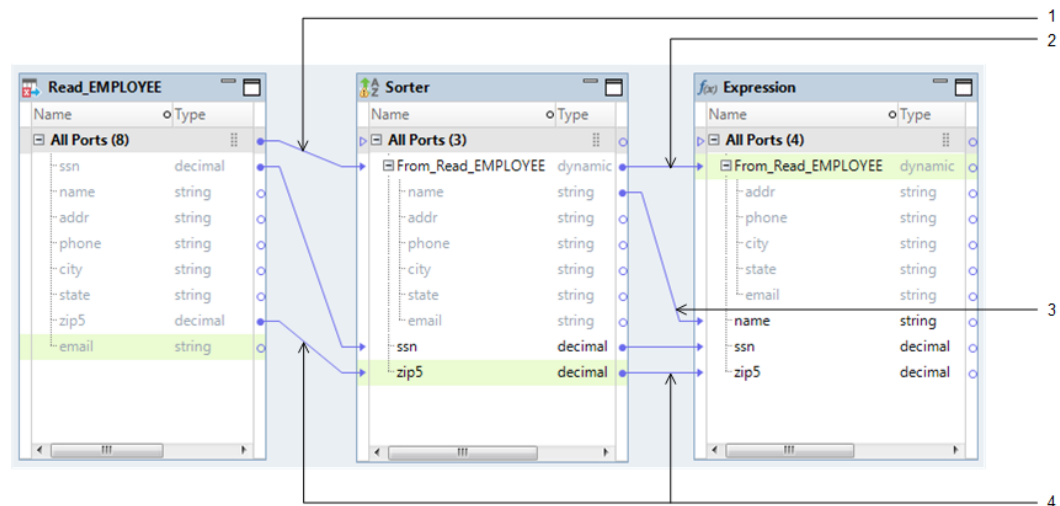
1. 「Salary\_PortSelector」というポートセレクトアを作成します。
2. サフィックスが「Salary」のあらゆるポート名を受け入れる選択ルールを作成します。
3. 特定のカラム名ではなくポートセレクトア名を含めるように、式を設定します。式の構文は次のとおりです。  
Salary\_PortSelector \* 12

## 設計時リンク

動的マッピングを設計するときにさまざまなタイプのリンクを作成できます。2つのポート間のリンク、グループと動的ポート間のリンク、2つの動的ポート間のリンク、および生成されたポートから静的ポートへのリンクを作成できます。

設計時リンク（リンク）とは、マッピング内で作成する直接リンクです。トランスフォーメーションは、マッピングを設計するときに直接リンクを作成できないように変更される場合があります。設計時にリンクを作成できない場合、実行時にリンクするポートを決定するランタイムリンクを設定できます。

次の画像は、動的マッピングのリンクを示しています。



1. 特定のグループから動的ポートへのリンク
2. 2つの動的ポート間のリンク
3. 生成されたポートから静的ポートへのリンク
4. 2つのポート間のリンク

マッピングを設計するときに次のタイプのリンクを作成できます。

#### グループから動的ポートにリンクする。

特定のグループから動的ポートへのリンクを作成した場合、1つ以上のコラムのデータがプロパゲートされます。1つのグループには、1つ以上のポートおよび動的ポートを含めることができます。動的ポートの下に表示される生成されたポートは、動的ポートの入力ルールによって決定されます。デフォルトルールでは、グループ内のすべてのコラムが、ダウンストリームトランスフォーメーションの動的ポート内の生成されたポートとして含められます。

例えば、前の画像は、読み取りトランスフォーメーションの「すべてのポート」グループからソータートランスフォーメーションの動的ポート「From\_Read\_EMPLOYEE」へのリンクを示しています。ソータートランスフォーメーションの動的ポート「From\_Read\_EMPLOYEE」の入力ルールには文字列ポートが含まれます。

#### 2つの動的ポートをリンクする。

2つの動的ポート間にリンクを作成した場合、1つ以上のコラムのデータがプロパゲートされます。動的ポートの下に表示される生成されたポートは、動的ポートの入力ルールによって決定されます。デフォルトルールでは、アップストリームの動的ポートからのすべてのコラムが、ダウンストリームトランスフォーメーションの動的ポート内の生成されたポートとして含められます。

例えば、前の画像は、ソータートランスフォーメーションの動的ポート「From\_Read\_EMPLOYEE」から式トランスフォーメーションの別の動的ポート「From\_Read\_EMPLOYEE」へのリンクを示しています。式トランスフォーメーションの動的ポートの入力ルールでは、文字列ポートは含められ、「名前」ポートは除外されます。

#### 生成されたポートを静的ポートにリンクする。

生成されたポートから特定のポートへのリンクを作成すると、単一のコラムのデータがプロパゲートされます。

例えば、前の画像は、ソータートランスフォーメーションの動的ポート「From\_Read\_EMPLOYEE」の下にある生成されたポート「名前」から式トランスフォーメーションのポート「名前」へのリンクを示しています。

## 2つの静的ポートをリンクする。

他のマッピングの場合と同じ方法でトランスフォーメーション間のポートをリンクします。

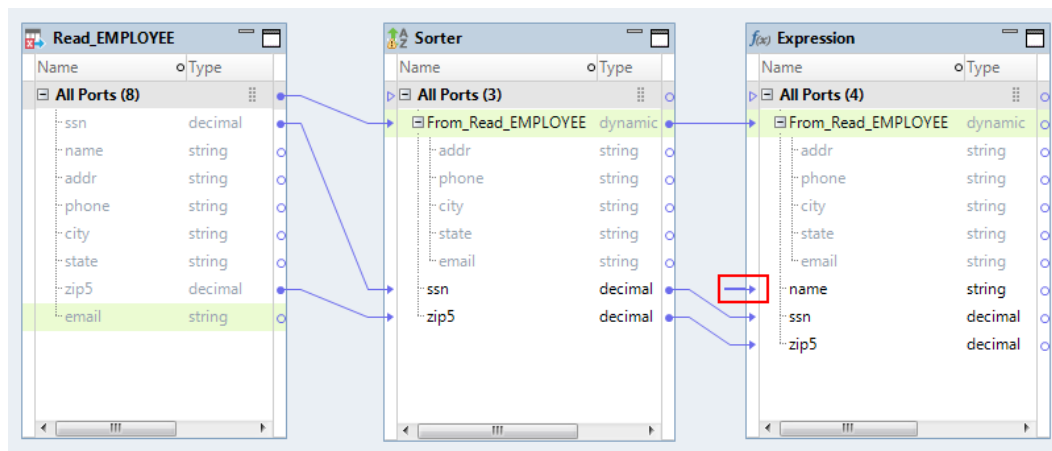
## リンクの解決

動的ポート内の生成されたポートは、動的ソースまたは入力ルールに基づいて変更される可能性があります。

生成されたポートからのリンクを作成した場合、生成されたポートが使用不可になると、Developer tool は、そのポートへのリンクを未解決リンクとして表示します。

例えば、ユーザーがソータートランスフォーメーションの動的ポート [From\_Read\_EMPLOYEE] の入力ルールを更新し、[名前] ポートを除外したとします。その場合、Developer tool は、そのリンクを未解決リンクに変更します。

次の画像は、式トランスフォーメーションのポート [名前] への未解決リンクを示しています。



マッピングを検証すると、Developer tool は未解決リンクに関する警告メッセージを表示します。マッピングの実行時、生成されたポートが使用可能であれば、データ統合サービスは、そのリンクを解決してマッピングを処理します。ただし、データ統合サービスがリンクを解決できない場合は、マッピングが失敗します。マッピングを正常に実行するには、未解決リンクを削除する必要があります。トランスフォーメーションを右クリックし、**未解決リンクのクリア**を選択して、トランスフォーメーションの未解決リンクをすべてクリアします。

## ランタイムリンク

ランタイムリンクとは、実行時にポートが変更される可能性があるグループ間のリンクです。データ統合サービスは、ポリシーおよびパラメータに基づいて実行時にリンクするポートを決定します。

実行時にアップストリームトランスフォーメーションのポートが変更される可能性がある場合は、マッピングオブジェクトのグループ間でランタイムリンクを作成します。ポートが実行時に変更される可能性があるとしても、マッピングの設計時にポートをリンクすることはできません。パラメータおよびリンクポリシーを使用して実行時にリンクするポートを決定できるランタイムリンクを作成します。

次の場合に、ランタイムリンクを作成します。

データソースからカラムを取得するため、またはパラメータによって定義されるソースを使用するために読み取りトランスフォーメーションを設定する場合。

例えば、読み取りトランスフォーメーションがパラメータを使用してソースを変更するか、実行時にソースからメタデータの変更を取得するとします。ダウンストリームトランスフォーメーションは、マッピング実行から次のマッピング実行の間に変更される可能性がある生成されたポートから、ポート経由でデータを受信します。ダウンストリームトランスフォーメーションへのランタイムリンクを作成して設定します。実行時に、データ統合サービスはリンクポリシーまたはパラメータ値に基づいてポートを接続します。

データソースまたはデータオブジェクトからカラムを取得するため、またはパラメータによって定義されるターゲットを使用するために書き込みトランスフォーメーションを設定する場合。

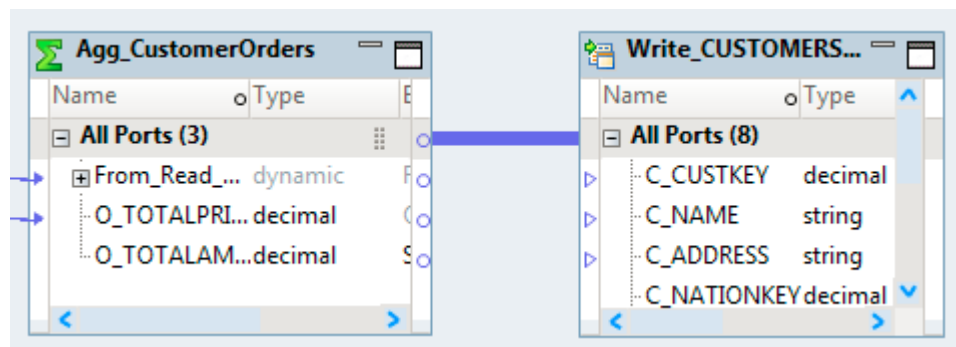
例えば、書き込みトランスフォーメーションが関連付けられているデータオブジェクトに基づいてカラムを定義するとします。書き込みトランスフォーメーションは、パラメータを使用してターゲットを変更するか、実行時にターゲットからメタデータの変更を取得します。書き込みトランスフォーメーションへのランタイムリンクを作成して設定します。

**注:** マッピングフローに基づいてターゲットカラムを定義する場合は、書き込みトランスフォーメーションへのランタイムリンクを作成しないでください。

実行時に、データ統合サービスはリンクポリシーまたはパラメータ値に基づいてポートを接続し、ダウンストリームポートにデータを渡します。

実行時にアップストリームトランスフォーメーションのポートが変更される可能性がある場合は、トランスフォーメーションのグループ間でランタイムリンクを作成します。データ統合サービスは、定義されたパラメータ、リンクポリシー、またはその両方に基づいて、実行時にリンクするポートを決定します。ランタイムリンクは、マッピングエディタで太線で表示されます。

次の画像は、アグリゲータトランスフォーメーションと書き込みトランスフォーメーション間のランタイムリンクを示しています。

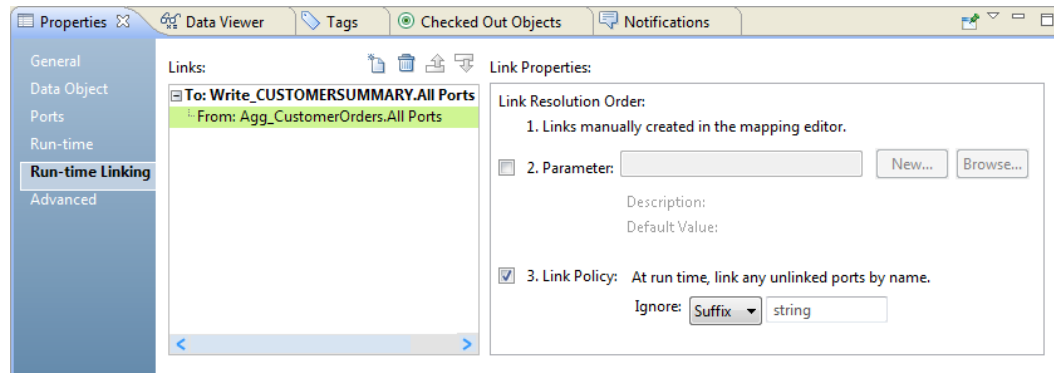


## ランタイムリンクの設定

実行時にマッピングオブジェクト間でリンクするポートを決定するようにランタイムリンクプロパティを設定します。パラメータを定義するか、リンクポリシーを選択するか、またはその両方を使用して、リンクするポートを決定できます。

**[ランタイムリンク]** ダイアログボックスまたは **[ランタイムリンク]** タブを使用してランタイムリンクプロパティを設定します。

次の画像は、書き込みトランスフォーメーションの【ランタイムリンク】タブを示しています。



データ統合サービスは、実行時に、次の順番でポート間のリンクを確立し、解決します。

1. マッピングエディタで手動で作成するリンク。
2. ランタイムリンク用に設定したパラメータに基づくリンク。
3. ランタイムリンク用に設定したリンクポリシーに基づくリンク。

ランタイムリンクの次のプロパティを設定します。

#### トランスフォーメーションの選択

【リンク】領域で、【新規】ボタンをクリックし、【新しいリンク】ダイアログボックスで、実行時にポートをリンクするトランスフォーメーションを選択します。トランスフォーメーションに入力ランタイムリンクがある場合、【リンク】領域には、リンクの起点となるグループポートが表示されます。

#### パラメータの設定

マッピング実行から次のマッピング実行の間にポート名が変更される可能性があり、ポート名の値がわかっているときは、パラメータを使用します。入力リンクセットタイプのパラメータを使用して、マッピング実行ごとに名前値でポートを接続します。

例えば、新しい入力リンクセットタイプの Cust\_InputLinkSet というパラメータを作成して、C\_Name -> Cust\_name のようにデフォルト値を指定します。実行時に、データ統合サービスは C\_Name ポートと Cust\_name ポート間のリンクを作成します。次のマッピング実行でパラメータの値を CustFirstName -> Cust\_name のように変更できます。

#### リンクポリシーの設定

リンクポリシーはリンクされていないポートを名前でリンクします。マッピングフローによってターゲットカラムを定義する場合、マッピングによって、ソースまたはアップストリームオブジェクトからのすべてのポートがプロパゲートされます。ランタイムリンクポリシーを使用して、特定のタイプのポートまたは特定の名前を持つポートをプロパゲートします。ポート名が同じ場合、リンクポリシーを選択します。同じ名前のポートを自動的にリンクするときは、リンクポリシーを使用します。

ポート名のサフィックスまたはプレフィックスを無視するようにリンクポリシーを設定することが可能です。例えば、サフィックス「\_OUT」を無視するようにリンクポリシーを設定した場合、データ統合サービスは、[SALARY\_OUT] を [SALARY] にリンクします。

グループをリンクするために行ったアクションに基づいて、【ランタイムリンク】ダイアログボックスまたは【プロパティ】の【ランタイムリンク】タブにリンクプロパティが表示されます。

- Ctrl を押しながらアップストリームトランスフォーメーションのグループをダウンストリームトランスフォーメーションのグループにドラッグすると、【ランタイムリンク】ダイアログボックスが開きます。
- 動的ポートを作成できない場合、アップストリームトランスフォーメーションからグループをドラッグすれば作成できます。次に、【リンクの作成】ダイアログボックスで【ランタイムリンクの作成】を選択すると、【ランタイムリンク】ダイアログボックスが開きます。

- ダウンストリームトランスフォーメーションの【プロパティ】ビューで【ランタイムリンク】タブを選択します。

## ランタイムリンクの例

部門別に合計給与を計算する再利用可能なアグリゲータトランスフォーメーションがマッピングに含まれているとします。アグリゲータトランスフォーメーションは、生成されたポートを持つ式トランスフォーメーションから従業員データを受信します。

式トランスフォーメーションは、動的式から次のような出力ポートを生成します。

Read\_EMP ソース 1:

EMPNO\_OUT  
NAME\_OUT  
SALARY\_OUT  
DEPTNO\_OUT

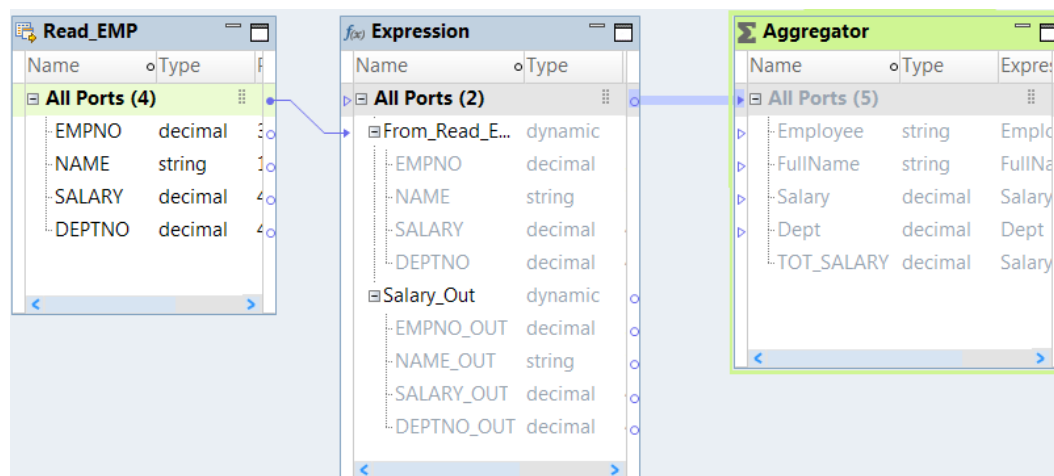
Read\_EMP ソース 2:

EMPNUM\_OUT  
FULLNAME\_OUT  
SALARY\_OUT  
DEPT\_OUT

アグリゲータトランスフォーメーションインスタンスには、動的ポートはありません。

EMPNO\_OUT または EMPNUM\_OUT 従業員番号、NAME\_OUT または FULLNAME\_OUT 文字列、SALARY 番号、および DEPTNO\_OUT または DEPT\_OUT 部門番号を受信するように、アグリゲータトランスフォーメーションインスタンスのランタイムリンクプロパティを設定します。

次の画像は、式トランスフォーメーションとアグリゲータトランスフォーメーション間のリンクを示しています。



## 動的マッピングのトラブルシューティング

動的マッピングを設計およびテストする際には、次のトラブルシューティングのヒントを考慮に入れてください。

マッピングの動的ポートに XML データ型のカラムが含まれており、マッピングが失敗した。



以下の条件のいずれかに当てはまる場合、マッピングを使用して XML データをプロパゲートすることはできません。

- データソースからカラムを取得するように読み取りまたは書き込みトランスフォーメーションを設定している。
- データフローからカラムを取得するように書き込みトランスフォーメーションを設定している。
- 実行時にターゲットを作成または置換するようにターゲットのデータオブジェクトを設定している。

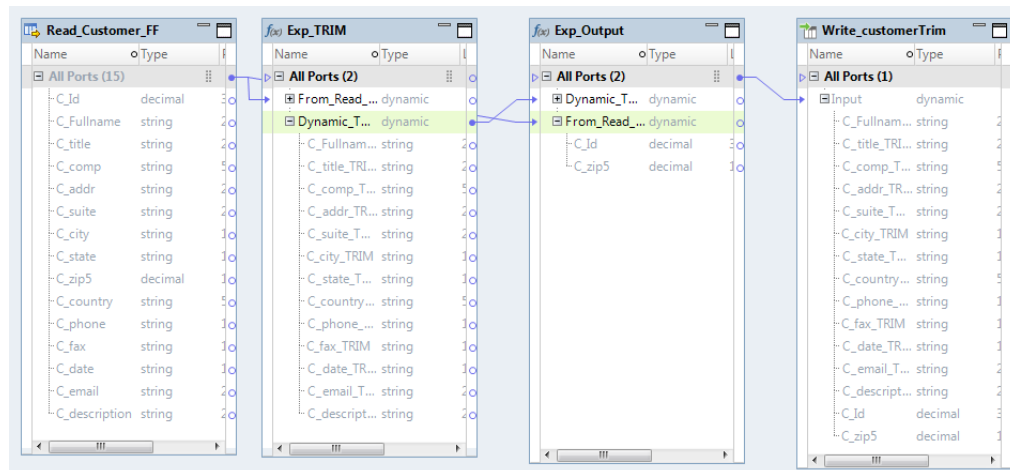
**パラメータおよびリンクの実行時エラーで動的マッピングが失敗します。マッピングを実行する前にパラメータとリンクが正常に解決されることを確認する方法を教えてください。**

動的マッピングを実行すると、データ統合サービスによってマッピングがコンパイルされ、以下のタスクが実行されます。

- パラメータ値を解決する。
- 動的ポートを展開して、生成されたポートを静的ポートに変換する。
- 静的ポートをリンクする。
- ランタイムリンクを解決してポートを接続する。

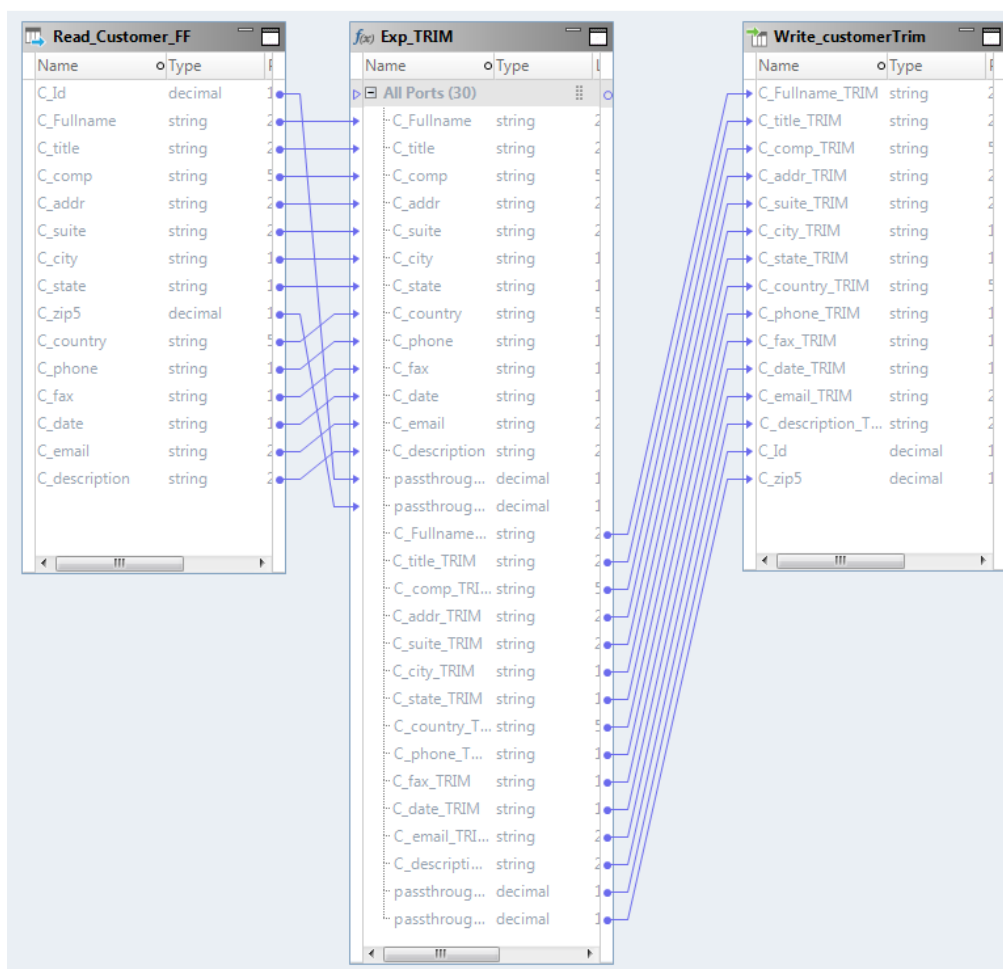
最適化されたマッピングを表示して、コンパイルバージョンのマッピングを確認できます。マッピングエディタの空の領域を右クリックし、**[最適化されたマッピングの表示]** をクリックします。データ統合サービスにより最適化されたマッピングが生成されます。最適化されたマッピングを確認し、問題があれば修正して、マッピングを実行できます。

次の画像は、動的ポートのあるトランスフォーメーションを含む動的マッピングを示しています。





次の画像は、コンパイルバージョンの動的マッピングを示しています。このマッピングでは、生成されたポートを静的ポートに変換し、リンクしています。



## 第 8 章

# 動的マッピングを開発して実行する方法

この章では、以下の項目について説明します。

- [動的マッピングの開発と実行, 162 ページ](#)
- [動的ソースの設定, 164 ページ](#)
- [動的ポートの作成, 165 ページ](#)
- [入力ルールを使用した動的ポートの設定, 166 ページ](#)
- [ポートセクタの作成, 170 ページ](#)
- [動的式の作成, 171 ページ](#)
- [動的ターゲットの設定, 173 ページ](#)
- [ランタイムリンクの作成と設定, 178 ページ](#)
- [動的マッピングの検証, 181 ページ](#)
- [動的ソースおよびターゲットの検証, 181 ページ](#)
- [動的マッピングの実行, 182 ページ](#)

## 動的マッピングの開発と実行

動的マッピングを作成すると、ソースメタデータの変更の管理や、さまざまなソースおよびターゲットのデータ統合ロジックの再利用ができます。メタデータが変更される可能性のある同じまたは異なるソースとターゲットに対して動的マッピングを実行します。

以下の表に、動的マッピングを作成して実行するタスクの概要を示します。タスクとその実行順序は、マッピングシナリオ、およびマッピングで使用するトランスフォーメーションによって決まります。

タスク	参照
マッピングを作成し、マッピングオブジェクトを追加する。	<a href="#">「マッピングの作成」 (ページ 38)</a> <a href="#">「マッピングへのオブジェクトの追加」 (ページ 38)</a>
実行時にフラットファイルまたはリレーショナルソースからメタデータの変更を取得するように、読み取りまたはルックアップの各トランスフォーメーションの動的ソースを設定する。	<a href="#">「動的ソースの設定」 (ページ 164)</a>

タスク	参照
トランスフォーメーションの動的ポートを作成し、ポートをリンクする。	<a href="#">「動的ポートの作成」 (ページ 165)</a>
<p>動的ポートの入力ルールを定義し、どの生成されたポートを作成するかを決定する。</p> <ul style="list-style-type: none"> <li>- ポートを含めるまたは除外する入力ルールを定義する。</li> <li>- 生成されたポートを名前変更する。</li> <li>- 必要に応じて、生成されたポートを並べ替える。</li> </ul>	<a href="#">「入力ルールを使用した動的ポートの設定」 (ページ 166)</a>
トランスフォーメーションを設定する。	マッピングのトランスフォーメーションを設定するには、『 <i>Informatica Developer トランスフォーメーションガイド</i> 』で詳細を確認のこと。
必要に応じて、ジョイナ、ルックアップ、または式の各トランスフォーメーションのトランスフォーメーションロジックで使用するポートセクタを作成する。	<a href="#">「ポートセクタの作成」 (ページ 170)</a>
必要に応じて、式トランスフォーメーションで使用する動的式を作成する。	<a href="#">「動的式の作成」 (ページ 171)</a>
<p>次の方法により、動的ターゲットに書き込むように書き込みトランスフォーメーションを設定する。</p> <ul style="list-style-type: none"> <li>- 関連付けられたデータオブジェクトからカラム定義を定義し、ターゲットファイルからメタデータの変更を取得するか、アップストリームトランスフォーメーションのマッピングフローからカラム定義を定義する。</li> <li>- リレーショナルおよび Hive ターゲットを表す書き込みトランスフォーメーションのターゲットスキーマ戦略を選択します。</li> </ul>	<a href="#">「動的ターゲットの設定」 (ページ 173)</a>
ランタイムリンクを作成および設定し、実行時にリンクするポートを決定する。	<a href="#">「ランタイムリンクの作成と設定」 (ページ 178)</a>
<p>マッピングでパラメータを使用する場所を決定した後、パラメータを作成して割り当てる。</p> <ul style="list-style-type: none"> <li>- ソースをパラメータとして設定する</li> <li>- ターゲットをパラメータとして設定する</li> <li>- トランスフォーメーションプロパティをパラメータとして設定する</li> </ul>	<a href="#">「パラメータの設定方法」 (ページ 68)</a>
マッピングを検査する。	<a href="#">「動的マッピングの検証」 (ページ 181)</a>
動的マッピングで同期されたデータソースとターゲットを検証する。	<a href="#">「動的ソースおよびターゲットの検証」 (ページ 181)</a>
動的マッピングをコンパイルおよび実行する。	<a href="#">「動的マッピングの実行」 (ページ 182)</a>

## 動的ソースの設定

トランスフォーメーションソースの変更時にカラム名を含むメタデータが動的に更新されるように、マッピングの読み取りおよびルックアップの各トランスフォーメーションを設定できます。

動的マッピング用に読み取りおよびルックアップの各トランスフォーメーションを設定するときは、次の方法を1つ以上使用できます。

### パラメータをソースとして使用する

読み取りまたはルックアップの各トランスフォーメーションのソースとしてパラメータ値を使用するときは、リポジトリ内の別の場所にある定義済みのソースデータオブジェクトを参照するパラメータ名を選択します。

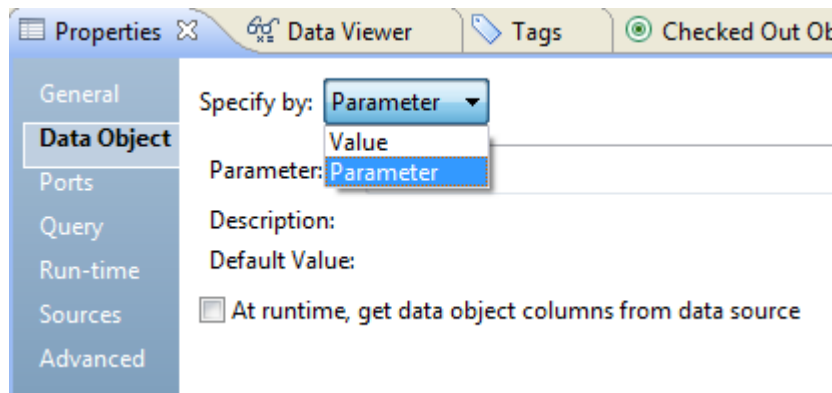
### 実行時にメタデータを取得するように読み取りおよびルックアップの各トランスフォーメーションを設定する

実行時にソースからデータオブジェクトカラムを取得するように読み取りまたはルックアップの各トランスフォーメーションを設定すると、マッピングの実行時にポート定義が更新されます。

## 動的マッピングのソースとしてのパラメータの使用

パラメータを動的マッピングソースオブジェクトのソースとして使用できます。

1. マッピングエディタでソースオブジェクトを選択します。
2. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。
3. マッピング実行間で異なる値をソースオブジェクトとして使用するには、**【指定元】** リストで **【パラメータ】** を選択します。



4. パラメータを新規作成する場合は **【新規】** をクリックし、既存のパラメータを選択する場合は **【参照】** をクリックします。

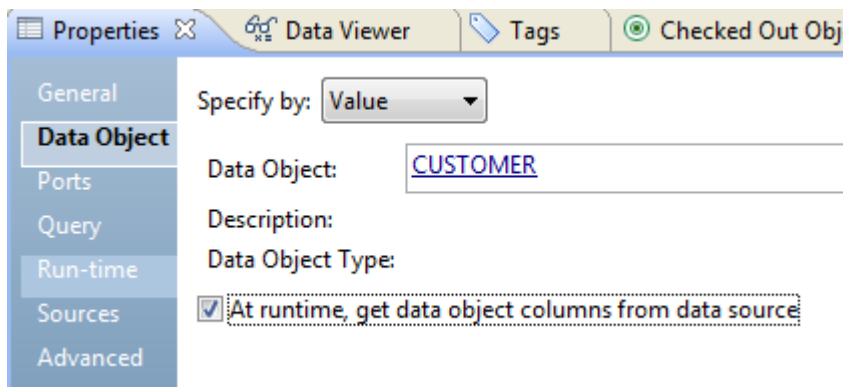
## 実行時にメタデータの変更を取得するためのソースの設定

実行時にメタデータの変更を取得するように、マッピングのソースオブジェクトのデータソースを設定できます。

マッピングの作成後にデータソースカラムのメタデータが変更された場合は、そのマッピングが期限切れになることがあります。データソースを設定する際、マッピングの実行時にこのデータを取得するオプションを指定できます。

1. マッピングエディタでソースオブジェクトを選択します。
2. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。

3. 実行時にデータソースファイルから動的にカラムを取得するために、**【実行時に、データソースからデータオブジェクトのカラムを取得します】**を選択します。



## 動的ポートの作成

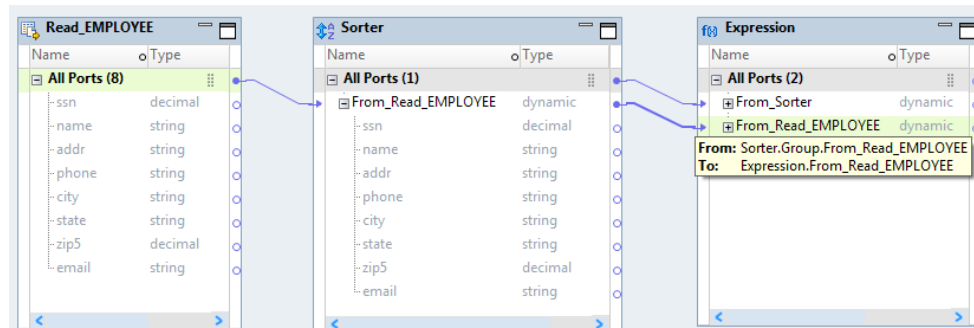
1つの動的ポートを作成して、アップストリームトランスフォーメーションから複数のカラムを受信することができます。これらのカラムは、実行時に変更される場合があります。トランスフォーメーションでは、複数の動的ポートを作成できます。

1. 動的ポートは、次の方法で作成します。

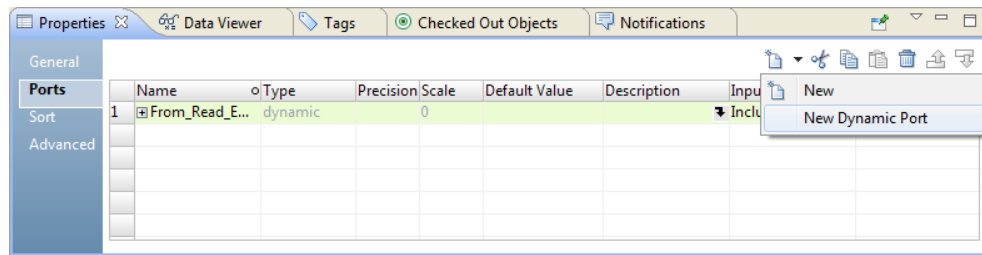
- [すべてのポート] グループまたは動的ポートを別のトランスフォーメーションからドラッグする。

Developer tool は、アップストリームトランスフォーメーションのすべてのカラムについて動的ポートおよび生成されたポートを作成し、これらのポートをリンクします。生成されたポートをフィルタリングするように入力ルールを変更できます。

次の画像は、ソーターおよび式の各トランスフォーメーションの動的ポートを示しています。

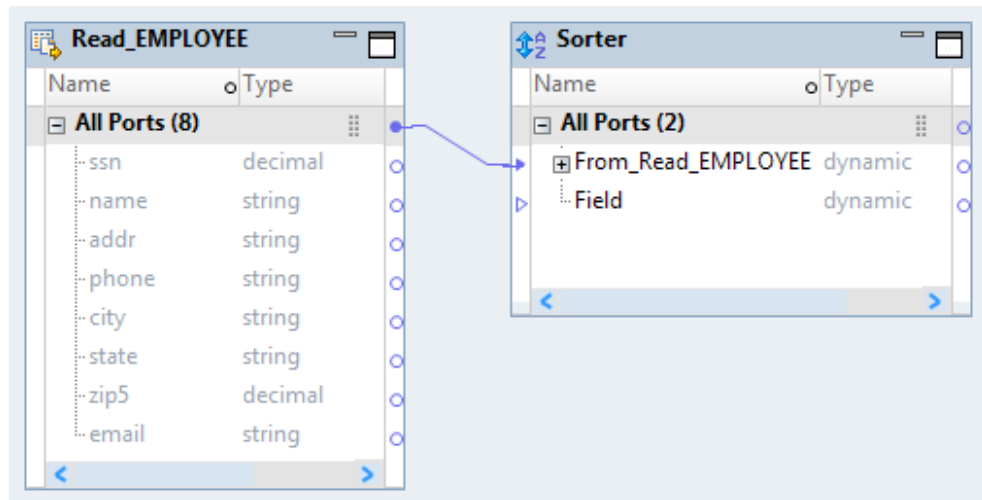


- トランスフォーメーションの【プロパティ】ビューにある【ポート】タブで、【新しい動的ポート】を選択する。



Developer tool は、ユーザーが設定可能な空の動的ポートを作成します。生成されたポートを作成するには、これらのポートを手動でリンクする必要があります。

次の画像は、ソータートランスフォーメーションの新しい動的ポートを示しています。このポートには、生成されたポートはありません。



2. 必要に応じて、この動的ポートの名前を変更し、説明を追加できます。

生成されたポートは、アップストリームトランスフォーメーションからポートのプロパティを継承するため、編集することはできません。

## 入力ルールを使用した動的ポートの設定

生成するポートを指定し、動的ポートを通じてパイプラインでプロパゲートするには、入力ルールを定義します。

**【入力ルール】** ダイアログボックスを使用すると、動的ポートの入力ルールを定義したり、マッピングでのポートの出現場所がわかるように、生成されたポートを名前変更したりすることができるほか、生成されたポートの順序の変更や、ルールの結果の表示も可能になります。ポートを含めたり、除外したりするルールを複数追加することができます。データ統合サービスは、リストの表示順にルールを適用します。

1. 【入力ルール】ダイアログボックスを開きます。

2. トランスフォーメーションの動的ポートごとに 1 つ以上の入力ルールを定義します。入力ルールごとに次の手順を実行します。
  - a. 入力ルールの演算子と選択条件を選択します。
  - b. 選択条件として [名前] を選択した場合は、名前またはパラメータ別に条件の詳細を指定します。
  - c. 選択条件として [タイプ] を選択した場合は、リストからポートのデータ型を選択します。
  - d. 選択条件として [パターン] を選択した場合は、パターンタイプを選択し、値またはパラメータとしてパターン文字列を指定します。必要に応じて、アップストリームトランスフォーメーションからの残りのすべてのポートを含めるように、トランスフォーメーションの最後の動的ポートの入力ルールを定義します。
3. 生成されたポートを名前変更する。
4. 必要に応じて、生成されたポートを並べ替えます。
5. 入力ルールの結果と入力ルールを設定を検証します。

## 手順 1. [入力ルール] ダイアログボックスを開く

入力ルールを定義または編集するために [入力ルール] ダイアログボックスを開きます。

- ▶ **[入力ルール]** ダイアログボックスは、次の方法で開きます。
    - トランスフォーメーションの動的ポートを右クリックし、**[入力ルールの編集]** を選択する。
    - トランスフォーメーションの **[ポート]** タブで、動的ポートの **[入力ルール]** をクリックする。
- [入力ルール]** ダイアログボックスが表示されたとき、デフォルトでは、[すべてを含める] 入力ルールが選択されています。

## 手順 2. 入力ルールの定義

アップストリームトランスフォーメーションから動的ポートが受信するポートを含めるまたは除外する入力ルールを定義します。

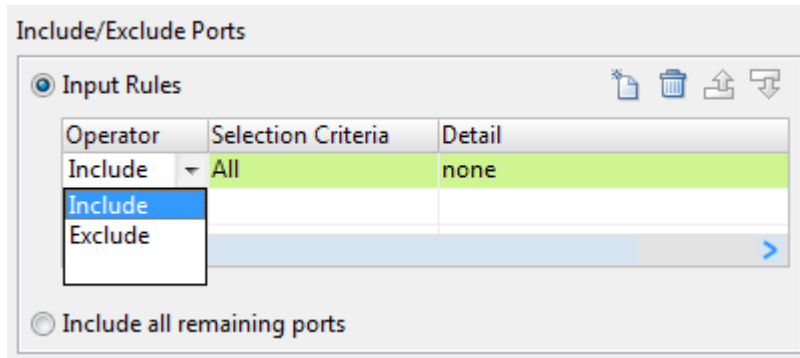
1. **[ポートを含めるまたは除外する]** 領域で **[入力ルール]** を選択します。
2. デフォルトの入力ルールを編集するには、次の手順を実行します。
  - a. 入力ルールの演算子と選択条件を選択します。
  - b. 選択条件の詳細を設定します。
3. 必要に応じて、Developer tool で実行する順序で入力ルールをさらに追加します。
  - a. **[新規]** をクリックし、入力ルールの新しい行を追加します。
  - b. 入力ルールごとに演算子と選択条件を選択し、条件の詳細を指定します。
4. アップストリームトランスフォーメーションからの残りのポートのみを含めるには、次の手順を実行します。
  - a. 別の動的ポートを作成するか、トランスフォーメーションの最後の動的ポートを選択します。
  - b. **[残りのすべてのポートを含める]** を選択します。

このルールでは、アップストリームトランスフォーメーションからのポートのうち、他の動的ポートの一部ではないものが含まれます。

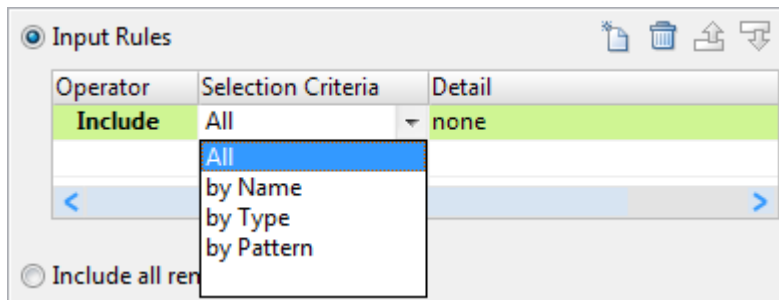
## 手順 2a. 演算子と選択条件の選択

ポートを含めるまたは除外する演算子、およびポート名またはデータ型に基づいてポートをフィルタリングする選択条件を選択します。

1. **【演算子】** カラムから**【含む】** 演算子または**【除外】** 演算子を選択します。  
ルールでポートを含めるのか、除外するのかは、選択した演算子によって決まります。



2. **【選択条件】** カラムでは、次のいずれかのオプションを選択します。
  - **【すべて】**。すべてのポートを含めます。このオプションを選択するときは、**【除外】** 演算子を選択しないでください。
  - **【名前】**。ポート名に基づいてポートを含めるまたは除外します。
  - **【タイプ】**。ポートのデータ型に基づいてポートを含めるまたは除外します。
  - **【パターン】**。ポート名のパターンに基づいてポートを含めるまたは除外します。



3. **【詳細】** カラムで**【詳細】** 矢印をクリックし、選択条件の詳細を指定します。  
選択条件に対する**【入力ルールの詳細】** ダイアログボックスが表示されます。

## 手順 2b. 名前選択条件の詳細の設定

入力ルールの選択条件として**【名前】** を選択した場合は、値のリストからポート名を選択します。あるいは、ポートタイプまたはポートリストタイプのパラメータを使用して、実行時に変更可能なポート名を指定します。

1. **【入力ルールの詳細: 名前別】** ダイアログボックスの**【指定元】** リストから次のいずれかを選択します。
  - **【値】**。ポート名を入力するか、リストからポート名を選択します。
  - **【パラメータ】**。パラメータを新規作成するか、**【ポートリスト】** タイプの既存のパラメータを選択します。
2. 次のいずれかの方法でポート名の値を指定します。
  - **【名前】** ボックスにポート名を入力し、**【追加】** をクリックする。



- **【選択】** をクリックして、**【ポート】** ダイアログボックスでポート名を選択し、**【OK】** をクリックする。
3. ポート名のパラメータを新規作成するには、次の手順を実行します。
    - a. **【新規】** をクリックします。
    - b. **【パラメータ】** ダイアログボックスでパラメータ名を入力します。
    - c. 必要に応じて、パラメータの説明を入力します。
    - d. **【ポート名】** パラメータのデフォルト値を入力します。 **【ポートのリストからポート名を選択する】** をクリックする方法もあります。
    - e. **【OK】** をクリックします。
  4. ポート名の既存のパラメータを選択するには、次の手順を実行します。
    - a. **【参照】** をクリックします。
    - b. **【パラメータの割り当て】** ダイアログボックスでパラメータを選択します。
    - c. 必要に応じて、このダイアログボックスでパラメータを新規作成するか、またはパラメータを編集します。
    - d. **【OK】** をクリックします。

## 手順 2c. タイプ選択条件の詳細の設定

入力ルールの選択条件として **【タイプ】** を選択した場合は、データ型のリストからタイプを選択します。

1. **【入力ルールの詳細: タイプ別】** ダイアログボックスのリストからデータ型を選択します。
2. **【OK】** をクリックします。

## 手順 2d. パターン選択条件の詳細の設定

入力ルールの選択条件として **【パターン】** を選択した場合は、ポート名のパターンタイプを選択します。パターンの値を入力するか、または文字列タイプのパラメータを使用して、実行時に変更できる値を指定します。

1. **【入力ルールの詳細: パターン別】** ダイアログボックスの **【パターンタイプ】** リストから次のいずれかを選択します。
  - **プレフィックス**。プレフィックスの文字列で始まるポート名を含めるまたは除外します。  
 例えば、プレフィックス値として「E」と入力した場合、入力ルールは、E または e で始まるポート名（EmpNo、empName、EmpTitle など）をフィルタリングします。
  - **サフィックス**。サフィックスの文字列で終わるポート名を含めるまたは除外します。  
 例えば、サフィックス値として「E」と入力した場合、入力ルールは、E または e で終わるポート名（empname、EMPTITLE など）をフィルタリングします。
  - **正規表現**。特定のパターンに従うポート名を含めるまたは除外します。  
 例えば、値として「E.\*No」と入力した場合、入力ルールは、E で始まり No で終わるポート名（ENo、EmpNo、EmployeeNo など）をフィルタリングします。
2. **【指定元】** リストから次のいずれかを選択します。
  - **【値】**。パターンの string 値を入力します。
  - **【パラメータ】**。パラメータを新規作成するか、string 型の既存のパラメータを選択します。
3. **【String】** ボックスにパターン値を指定して、**【OK】** をクリックします。
4. パターンのパラメータを新規作成するには、次の手順を実行します。
  - a. **【新規】** をクリックします。

- b. [パラメータ] ダイアログボックスでパラメータ名を入力します。
  - c. 必要に応じて、パラメータの説明を入力します。
  - d. パターンのデフォルト値を入力し、精度値を入力します。
  - e. **[OK]** をクリックします。
5. ポート名の既存のパラメータを選択するには、次の手順を実行します。
  - a. **[参照]** をクリックします。
  - b. [パラメータの割り当て] ダイアログボックスでパラメータを選択します。
  - c. 必要に応じて、このダイアログボックスでパラメータを新規作成するか、またはパラメータを編集します。
  - d. **[OK]** をクリックします。

## 手順 3.生成されたポートの名前の変更

生成されたポート名を変更し、トランスフォーメーション内でポート名が一意であることを確認します。

1. **[設定]** 領域で、**[ポートの名前の変更]** を選択します。
2. 生成されたポート名を変更するにあたって、プレフィックスまたはサフィックスを追加するかどうかを選択します。
3. 生成されたポートのプレフィックスまたはサフィックスとしてテキストを追加します。  
名前変更されたポートが **[ポートのプレビュー]** 領域に表示されます。

## 手順 4.生成されたポートの順序変更

結果を効果的に表示して分析できるように、生成されたポートの順序を変更します。

- ▶ **[設定]** 領域で、**[生成されたポートの順序を入力ルールの順序に従って変更]** を選択します。  
順序変更されたポートが **[ポートのプレビュー]** 領域に表示されます。 アップストリームトランスフォーメーションに表示される順序ではなく、入力ルールの順序に従って、生成されたポートが表示されます。

## 手順 5.動的ポート設定の検証

動的ポートに対して定義したルールと設定に基づいて、生成されたポートを表示します。

- ▶ **[入力ルール]** ダイアログボックスの **[ポートのプレビュー]** 領域で、動的ポートの入力ルール設定の結果を確認します。

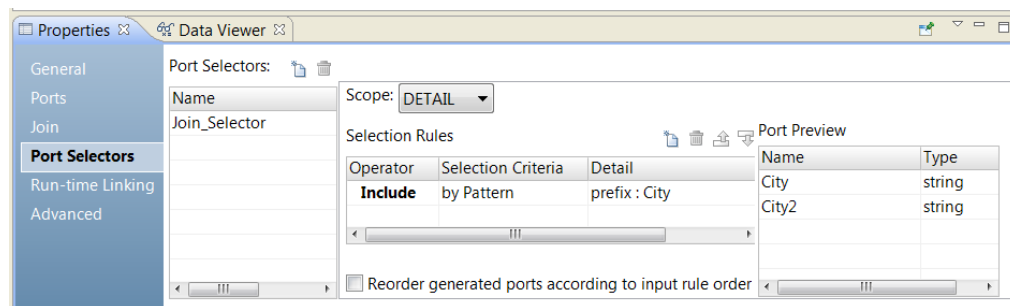
# ポートセレクトタの作成

動的式、ルックアップ条件、または結合条件で使用するポートを決定するために、ポートセレクトタを作成します。

1. **[ポートセレクトタ]** タブをクリックします。
2. **[ポートセレクトタ]** 領域で、**[新規]** をクリックします。  
Developer tool が、すべてのポートが含まれるデフォルトの選択ルールを使用してポートセレクトタを作成します。

3. **【ポートセレクトア】** 領域で、ポートセレクトア名を一意的な名前に変更します。
4. ジョイナトランスフォーメーションまたはルックアップトランスフォーメーションに対して作業している場合は、スコープを選択します。  
使用可能なポートは、選択したポートのグループに基づいて変更されます。
5. **【選択ルール】** 領域で、**【演算子】** を選択します。
  - 含む。ポートセレクトアのポートを含めるルールを作成します。ポートを除外する前にポートを含める必要があります。
  - 除外。ポートセレクトアから特定のポートを除外するルールを作成します。
6. **【選択条件】** を選択します。
  - 名前。特定のポートを名前を選択します。スコープ内のポートのリストからポート名を選択できます。
  - タイプ。ポートをタイプで選択します。1 つまたは複数のデータ型を選択できます。
  - パターン。ポート名の文字のパターンでポートを選択します。特定の文字で検索するか、正規表現を作成できます。

次の図は、**【ポートセレクトア】** タブを示しています。



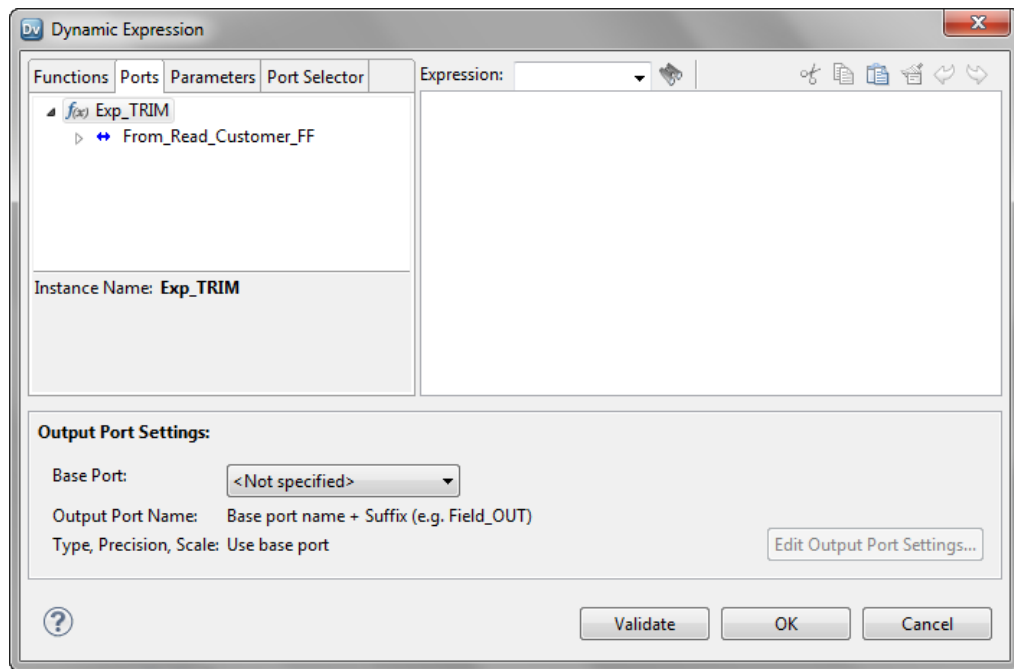
7. **【詳細】** カラムをクリックします。  
**【入力ルールの詳細】** ダイアログボックスが表示されます。
8. ポートをフィルタ処理する基準となる値を選択します。
  - 名前。値またはパラメータを基準としてポートリストを作成する場合に選択します。**【選択】** をクリックして、リスト内のポートを選択します。
  - タイプ。リストから 1 つ以上のデータ型を選択します。**【ポートのプレビュー】** 領域に、選択したタイプのポートが表示されます。
  - パターン。ポート名のプレフィックスまたはサフィックスから、特定の文字パターンを検索する場合に選択します。または、検索に使用する正規表現を作成することを選択します。パラメータを設定するか、検索に使用するパターンを設定します。
- 【ポートのプレビュー】** 領域に、設定したルールどおりにポートセレクトアのポートが表示されます。
9. ポートセレクトアのポートの順序を変更するには、**【生成されたポートの順序を入力ルールの順序に従って変更】** を選択します。

## 動的式の作成

式トランスフォーメーションで動的式を作成して、動的ポートまたはポートセレクトアの各ポートで式を 1 回実行します。動的式は、インスタンスごとに個別に生成されたポートに結果を返します。

1. 式トランスフォーメーションで **【プロパティ】** ビューに移動して、**【ポート】** タブをクリックします。

2. **【新しい動的ポート】** をクリックします。  
Developer tool で、デフォルトのプロパティを使用して動的ポートが作成されます。
3. 動的ポートの名前を変更し、入力オプションを無効にします。  
動的ポートは出力ポートである必要があります。
4. 動的出力ポートの **【式】** カラムで、**【開く】** ボタン (🔗) をクリックします。  
**【動的式】** ダイアログボックスが表示されます。



5. 式エディタに式を入力します。式にはポートセクタまたは動的ポートを含めることができます。  
例えば、LTRIM(RTRIM(Dynamic\_Customer))です。ここで、Dynamic\_Customer は動的ポートです。
6. **【検証】** をクリックして式を検証します。
7. **【OK】** をクリックして、**【式の検証】** ダイアログボックスを終了します。
8. **【出力ポート設定】** 領域で、**【ベースポート】** リストから動的出力ポートを選択するか、式で参照したポートセクタを選択します。  
Developer tool で、選択に基づいて出力ポートが生成されます。

9. 次の手順を実行して出力ポートの名前を変更します。
  - a. **【出力ポート設定の編集】** をクリックします。  
**【出力ポート設定】** ダイアログボックスが表示されます。

The screenshot shows the 'Output Port Settings' dialog box. The 'Base Port' field contains 'From\_Read\_Customer\_FF'. In the 'Output Ports:' section, the 'Name' dropdown is set to 'Base port name + Suffix', the 'Suffix' text box contains '\_OUT', and the 'Preview' field shows 'Field\_OUT'. In the 'Other Settings:' section, the 'Use base port' radio button is selected. The 'Type' dropdown is set to 'any', and the 'Precision' and 'Scale' text boxes are empty. At the bottom, there are 'OK' and 'Cancel' buttons, along with a help icon (?) on the left.

- b. **【名前】** リストで、オプションのいずれかを選択してプレフィックスまたはサフィックスの値を入力します。**【固定文字列 + 自動番号割り当て】** を選択した場合、出力ポート名のテキストを入力します。例えば、出力ポート名に TRIM と入力すると、出力ポート名は TRIM1、TRIM2、TRIM3 と表示されます。
  - c. 必要に応じて、**【その他の設定】** 領域の **【設定の指定】** を選択し、出力ポートのタイプ、精度、およびスケールを変更します。デフォルトでは、出力ポートでベースポートの設定が使用されます。
  - d. **【OK】** をクリックします。
10. **【OK】** をクリックして、**【動的式】** エディタを終了します。

## 動的ターゲットの設定

ターゲットメタデータが変更された場合、実行時にターゲットからカラムを受信するように書き込みトランスフォーメーションを設定します。必要に応じて、パラメータをターゲットデータオブジェクトとして指定し、

異なる値の割り当てができるようにします。また、関連付けられたオブジェクトまたはマッピングフローのどちらを使用して書き込みトランスフォーメーションがポートを定義するかを指定することもできます。

動的マッピングの書き込みトランスフォーメーションを設定する場合、次の1つ以上の方法を使用できます。

#### パラメータをターゲットとして使用する

パラメータをターゲットの基になるデータオブジェクトとして指定すると、パラメータを通して書き込みトランスフォーメーションのスキーマを変更できます。

#### 実行時にターゲットからデータオブジェクトのカラムを取得する

実行時にターゲットからデータオブジェクトのカラムを取得するオプションを有効にすると、書き込みトランスフォーメーションのポートがターゲットスキーマの変更内容で動的に更新されます。

#### ターゲットスキーマ戦略を選択する

ターゲットスキーマ戦略を選択し、実行時に既存のターゲットスキーマを保持するか、ターゲットテーブルを作成または置換するかを選択します。

書き込みトランスフォーメーションを設定してターゲットを作成または置換すると、デフォルトでデータ統合サービスにより、書き込みオブジェクトに関連付けられた既存のターゲットテーブルがすべて削除されます。次に、マッピングフローまたはデータオブジェクトに基づいてテーブルが作成されます。

テーブルをカスタマイズするか、パーティションなどの追加パラメータを指定する場合、DDL クエリを定義することができます。データ統合サービスは、この DDL クエリに基づいてターゲットテーブルを作成または置換します。テーブルには、DDL クエリで定義するカラムが含まれます。

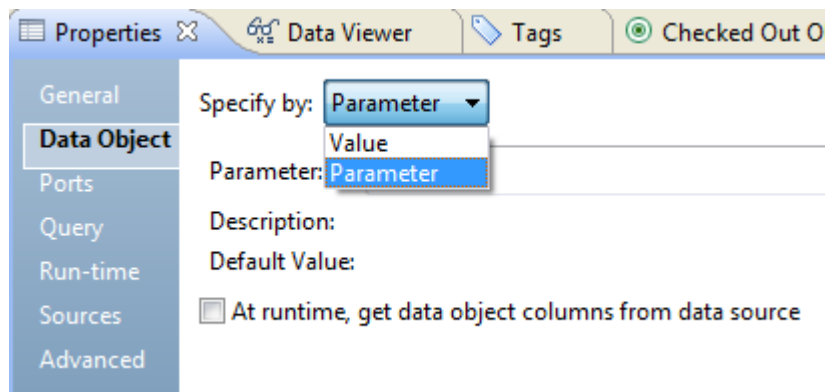
#### 書き込みトランスフォーメーションのポートをマッピングフローから定義する

マッピングフローからポートを定義するように選択した場合、データ統合サービスは、アップストリームのカラムの定義に基づいて書き込みトランスフォーメーションのポートを定義します。ターゲットのカラムが実行時に動的に更新されます。

## 動的マッピングのターゲットとしてのパラメータの使用

パラメータをトランスフォーメーションのデータオブジェクトとして使用し、パラメータを実行時に変更することができます。

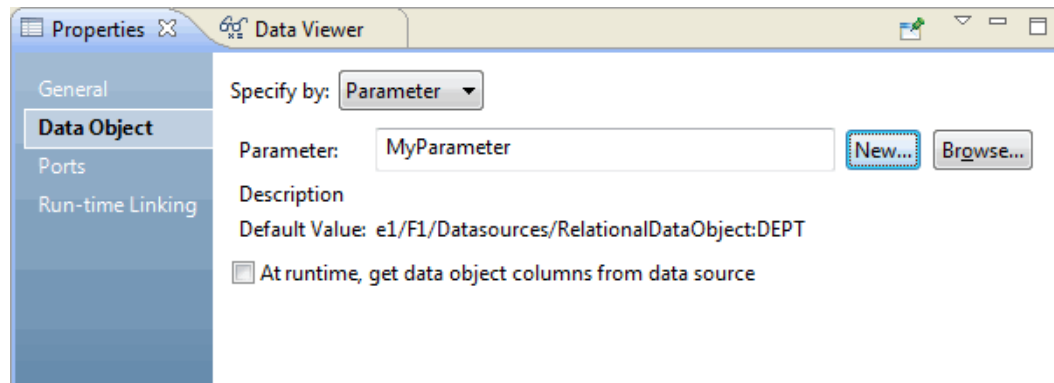
1. マッピングエディタで書き込みトランスフォーメーションを選択します。
2. 【プロパティ】ビューで、【データオブジェクト】タブをクリックします。
3. 【指定元:】ドロップダウンリストで、【パラメータ】を選択します。



4. 次のいずれかのオプションを選択します。
  - 【新規】をクリックしてパラメータを作成します。パラメータに名前を付けて、パラメータのデフォルト値を参照して選択します。

- **【参照】** をクリックして、既存のパラメータを選択します。

次の画像は、データソースとしてのパラメータを持つトランスフォーメーションを示しています。

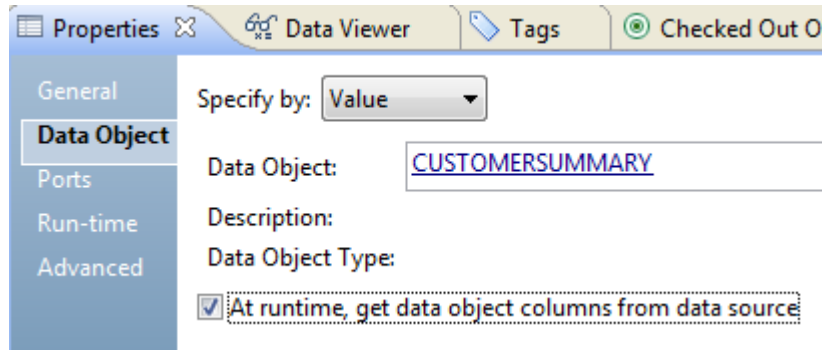


## 実行時のデータソースからのターゲットオブジェクトカラムの取得

実行時にデータソースからデータオブジェクトのカラムを取得するオプションを有効にすることができます。

実行時にデータソースからデータオブジェクトのカラムを取得するオプションを選択した場合、マッピングの実行時にマッピングはデータオブジェクトのカラムをトランスフォーメーションに取り込みます。データソースのカラムおよびメタデータが変更されると、マッピングは変更された情報を取得します。

1. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。
2. **【実行時に、データソースからデータオブジェクトのカラムを取得します】** を選択します。



## 実行時にターゲットを作成または置換するための DDL クエリの定義

ターゲットスキーマ戦略オプションを選択して実行時にターゲットを作成または置換する場合、DDL クエリを定義することができます。データ統合サービスは、この DDL クエリに基づいて実行時にターゲットテーブルを作成または置換します。DDL クエリは、リレーショナルターゲットおよび Hive ターゲットに対して定義できます。DDL クエリにはプレースホルダおよびパラメータを入力できます。

1. **【プロパティ】** ビューで **【詳細】** タブをクリックします。
2. **【ターゲットスキーマ戦略】** リストから、**【実行時にテーブルを作成または置換】** を選択します。  
**【DDL クエリ】** フィールドが入力可能になります。
3. **【編集】** をクリックします。  
**【DDL クエリ】** ダイアログボックスが表示されます。

4. エディタで DDL クエリを入力します。

DDL クエリにはプレースホルダを入力できます。データ統合サービスにより、実行時にプレースホルダが実際の値に置き換えられます。例えば、テーブルに 50 個のカラムが含まれている場合、DDL クエリにすべてのカラム名を入力する代わりに、プレースホルダを入力できます。

DDL クエリには、次のプレースホルダを入力できます。

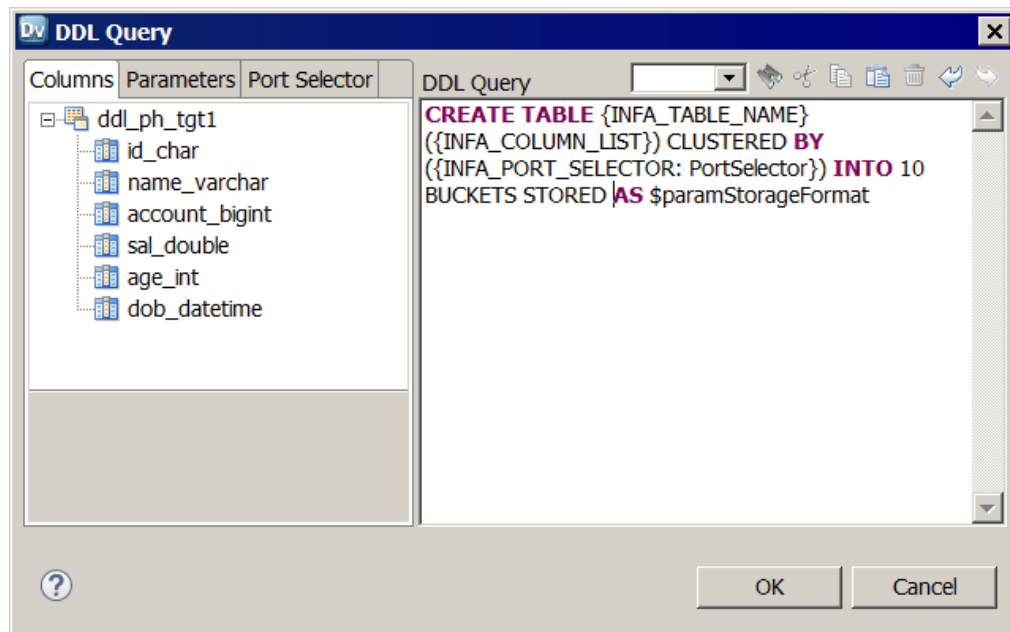
- INFA\_TABLE\_NAME.実行時にターゲットテーブル名を取得します。
- INFA\_COLUMN\_LIST.実行時にターゲットテーブルのカラムのリストを取得します。
- INFA\_PORT\_SELECTOR.ポートセレクトを追加します。

**注:** プレースホルダ名では、大文字と小文字が区別されます。プレースホルダは 2 つの中括弧で囲む必要があります。例えば、{INFA\_TABLE\_NAME}のようになります。

次の手順を実行して DDL クエリを定義することもできます。

- カラム名を追加するには、**[カラム]** タブでカラムをダブルクリックします。
- パラメータを定義するには、**[パラメータ]** タブをクリックし、パラメータ名をダブルクリックします。**[パラメータの管理]** をクリックしてパラメータを追加、編集、または削除することもできます。
- ポートセレクトを設定するには、**[ポートセレクト]** タブをクリックし、ポートセレクトをダブルクリックします。**[新規]** をクリックして新しいポートセレクトを設定することもできます。

次の図に、Hive ターゲットテーブルを作成するための DDL クエリを示します。



この図の DDL クエリには、INFA\_TABLE\_NAME、INFA\_COLUMN\_LIST、および INFA\_PORT\_SELECTOR というプレースホルダがあります。保存形式を定義するためのパラメータも含まれています。

DDL クエリを入力しない場合、データ統合サービスはマッピングフローまたはデータオブジェクトに基づいてターゲットを作成します。

5. **[OK]** をクリックして DDL クエリを保存します。

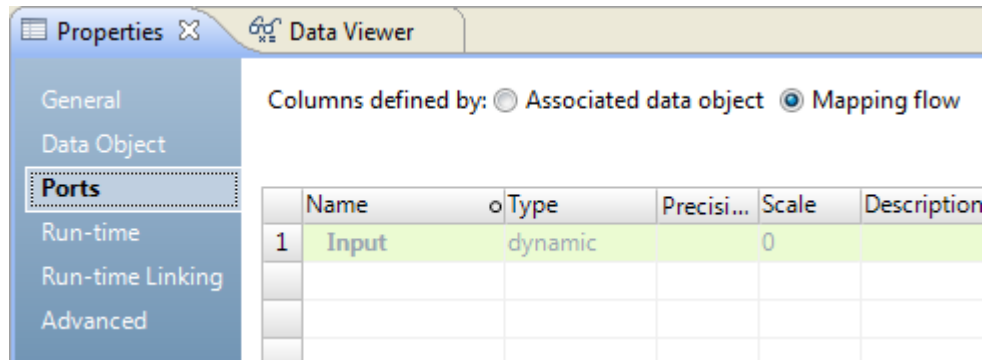


## 書き込みトランスフォーメーションポートの定義

マッピングフローでターゲットオブジェクトのカラムを定義することで、アップストリームのマッピングオブジェクトによって書き込みトランスフォーメーションの入力ポートを更新できます。

1. **【プロパティ】** ビューの **【ポート】** タブをクリックします。
2. **【次によって定義されたカラム:]** で **【マッピングフロー】** を選択します。

次の画像は、**【ポート】** タブを示しており、関連付けられたデータオブジェクトで定義されたポートが表示されています。

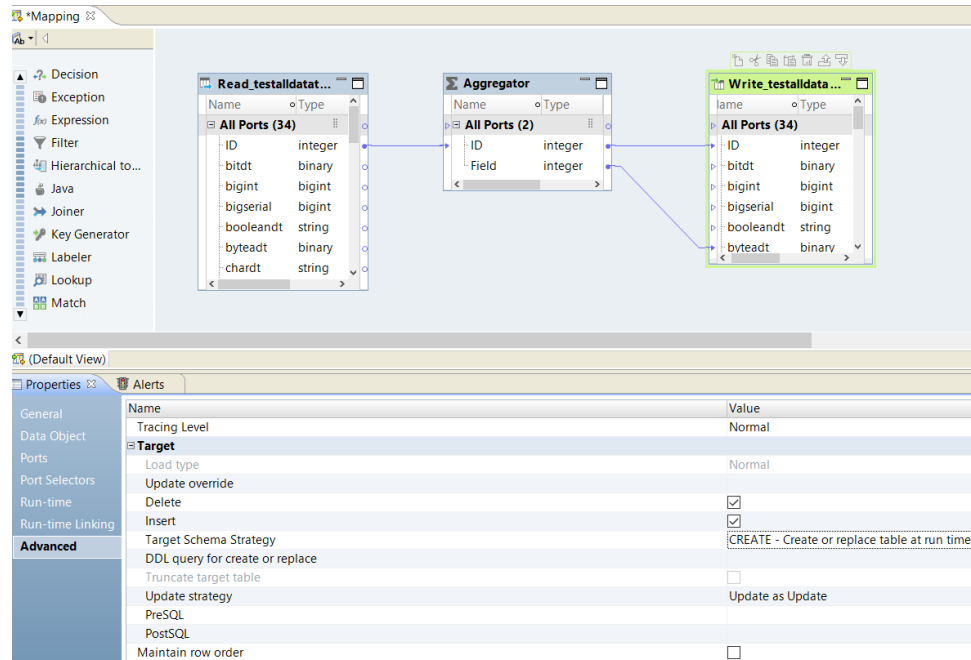


3. 動的ポートおよびターゲットを有効にします。
  - a. 書き込みトランスフォーメーションの **【入力】** ペインにアップストリームのポートをドラッグします。

ターゲットはカラムの定義をアップストリームのマッピングオブジェクトから取得します。
  - b. **【プロパティ】** ビューで **【詳細】** タブをクリックします。

- c. 【ターゲットスキーマ戦略】 リストから、【実行時にテーブルを作成または置換】を選択します。

次の図は、ターゲットオブジェクトの【詳細】タブの【ターゲットスキーマ戦略】 オプションを示しています。



実行時にデータ統合サービスはターゲットテーブルを維持するか、ターゲットテーブルを削除して置換します。

**注:** マッピングに複数のターゲットが含まれており、それらのカラムが同じ物理データオブジェクトによって定義されている場合、【作成 - 実行時にテーブルを作成または置換】 ターゲットスキーマ戦略オプションを1つのターゲットに対してのみ選択します。複数のターゲットに対してこのオプションを選択した場合、マッピングで作成されたテーブルのメタデータと一致するターゲットが1つのみとなるため、マッピングが失敗します。

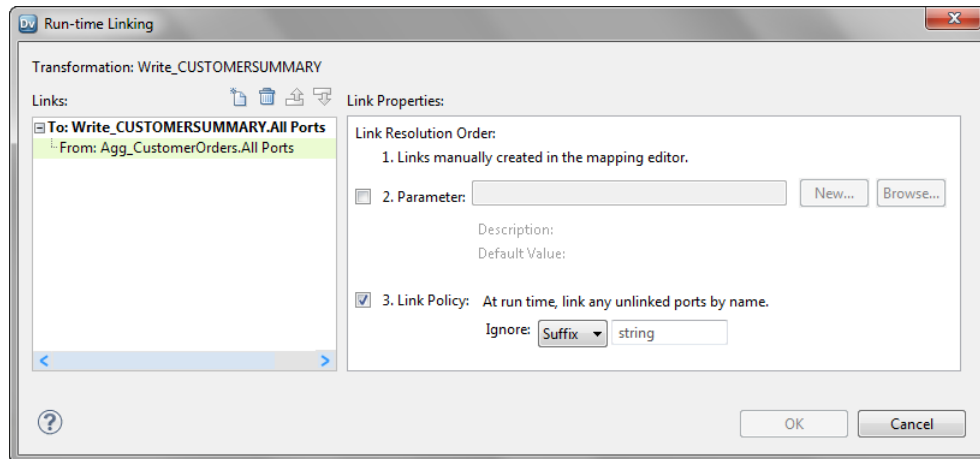
## ランタイムリンクの作成と設定

トランスフォーメーショングループ間のランタイムリンクを作成して、パラメータまたはリンクポリシーあるいはその両方に基づいてポートを実行時にリンクします。

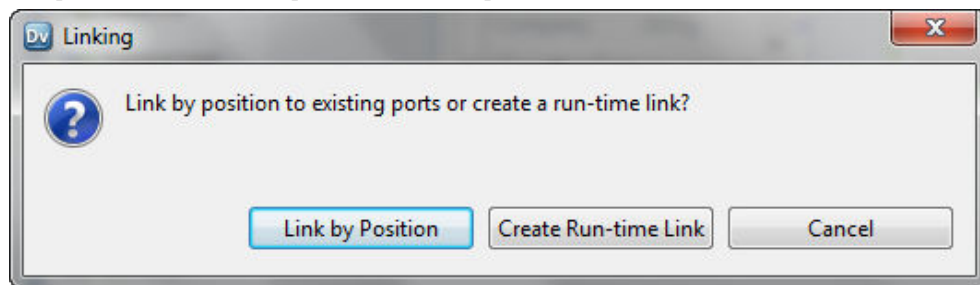
1. 以下の方法でランタイムリンクを作成します。

- Ctrl キーを押しながら、グループを動的マッピングのダウンストリームトランスフォーメーションにドラッグします。

[ランタイムリンク] ダイアログボックスが表示されます。




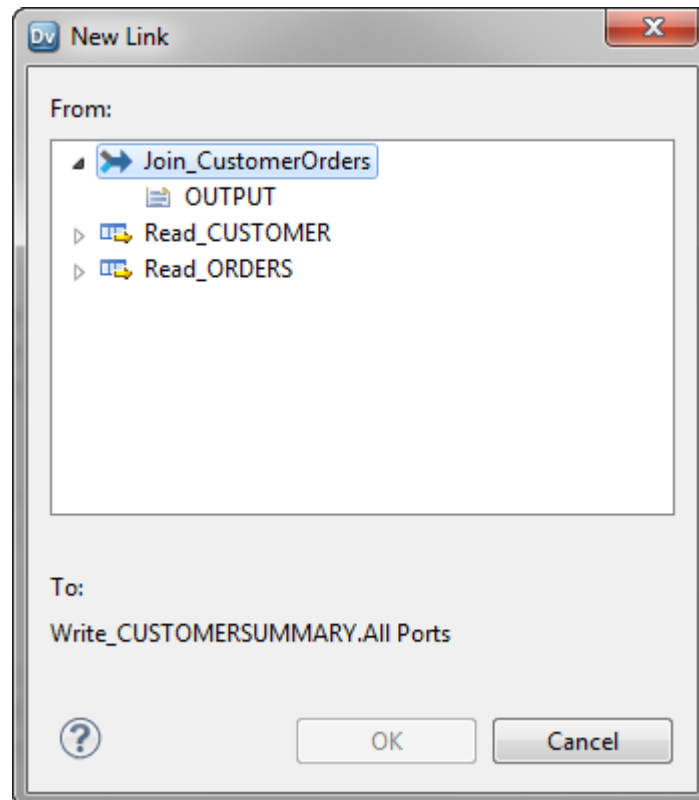
- 書き込みトランスフォーメーションまたは再利用可能なトランスフォーメーションへのランタイムリンクを作成するには、グループをアップストリームトランスフォーメーションから、再利用可能なトランスフォーメーションまたは書き込みトランスフォーメーション内のグループにドラッグします。次に、[リンクの作成] ダイアログボックスで [ランタイムリンクの作成] を選択すると、[ランタイムリンク] ダイアログボックスが開きます。
- 作成するランタイムリンクのリンク先とするダウンストリームトランスフォーメーションで、[プロパティ] ビューに移動して、[ランタイムリンク] タブをクリックします。



2. [リンクのプロパティ] 領域で、次のオプションの 1 つまたは両方を選択して、実行時にどのポートをリンクするかを特定します。
  - [パラメータ]。マッピングの実行間でポート名が変わる可能性があり、そのポート名を認識している場合、パラメータを使用します。新しいパラメータを作成するか、入力リンクセットタイプの既存のパラメータを選択します。
  - リンクポリシー。名前によってポートを自動的にリンクするには、リンクポリシーを使用します。デフォルトでは、このオプションはオンになっています。ポート名にプレフィックスまたはサフィックスが含まれる場合、無視する文字列を入力します。
3. [入力リンクセット] タイプの新しいパラメータを作成するには、次の手順を実行します。
  - a. [新規] をクリックします。
  - b. [パラメータ] ダイアログボックスでパラメータ名を入力します。  
例: Cust\_InputLinkSet
  - c. 必要に応じて、パラメータの説明を入力します。
  - d. パラメータのデフォルト値を、カンマ区切りのポートのペアとして入力します。  
例えば、次のようにデフォルト値を入力します。

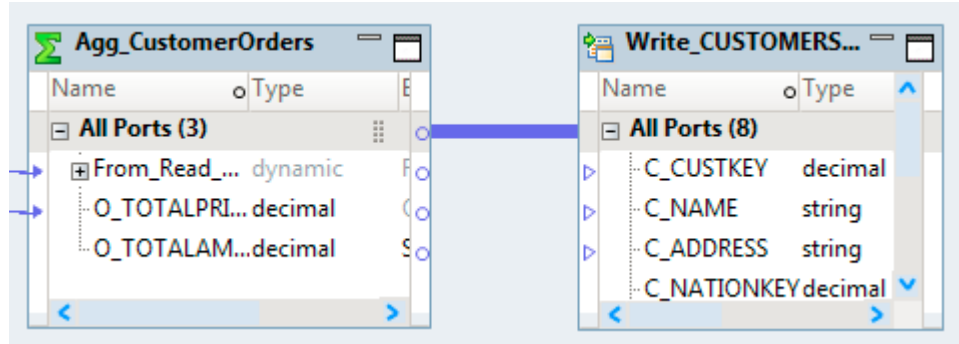
C\_NAME->Cust\_name, C\_ACCTBAL->Cust\_acctbal

- e. **【OK】** をクリックします。
- 4. **【入力リンクセット】** タイプの既存のパラメータを選択するには、次の手順を実行します。
  - a. **【参照】** をクリックします。
  - b. **【パラメータの割り当て】** ダイアログボックスでパラメータを選択します。
  - c. 必要に応じて、このダイアログボックスでパラメータを新規作成するか、またはパラメータを編集します。
  - d. **【OK】** をクリックします。
- 5. 必要に応じて、**【ランタイムリンク】** ダイアログボックスから別のランタイムリンクを追加するには、次の手順を実行します。
  - a. **【新規】** ボタン (  ) (**【リンク】** 領域内) をクリックします。  
**【新しいリンク】** ダイアログボックスが表示されます。



- b. 動的マッピング内の別のトランスフォーメーションからグループを選択します。
- 6. **【OK】** をクリックして、ランタイムリンクを作成します。

Developer tool によって、グループ間にランタイムリンクが作成されます。



7. 既存のランタイムリンクを編集するには、そのリンクを右クリックして、**【ランタイムリンク】** を選択します。

**【ランタイムリンク】** ダイアログボックスが表示され、そこでリンク先のポートを決定するオプションを変更できます。

## 動的マッピングの検証

マッピングを検証して、データ統合サービスがマッピング全体を読み取りおよび処理できることを確認します。

1. マッピングを開いて、**【編集】** > **【検証】** をクリックします。  
**【検証ログ】** ビューにエラーが表示される場合は、エラーを修正し、再度マッピングを検証します。
2. マッピングが有効であれば、**【ファイル】** > **【保存】** をクリックしてマッピングを保存します。
3. ソーススキーマが変更された後、またはパラメータ値を変更した後に、マッピングを再検証します。

## 動的ソースおよびターゲットの検証

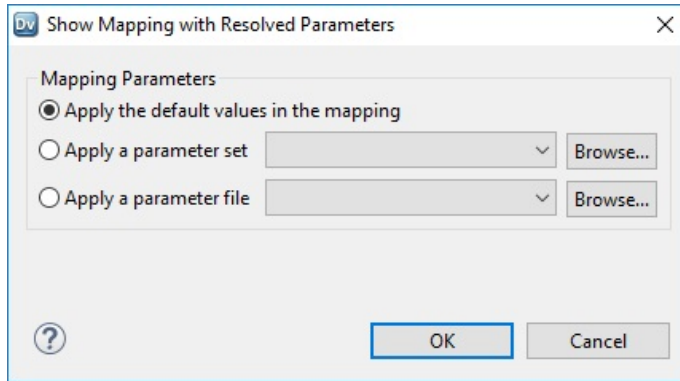
動的マッピングのソースおよびターゲットを検証し、実行時にソースまたはターゲットがリレーショナルデータベースと同期したときにデータ統合サービスがマッピング全体を読み取って処理できることを確認します。

ソースおよびターゲットを検証するには、マッピングパラメータを解決し、解決済みのパラメータを含むマッピングの生成されたインスタンスを検証します。マッピングパラメータを解決する前に、マッピングが有効になっている必要があります。

**注:** マッピングパラメータを解決するには、マッピング内でパラメータが設定されている必要があります。パラメータが設定されていない場合は、動的ソースおよびターゲットを検証できません。

1. Developer tool で、エディタまたはオブジェクトエクスプローラビューでマッピングを右クリックします。**[解決済みパラメータを使用したマッピングの表示]** を選択します。

**[解決済みパラメータを使用したマッピングの表示]** ダイアログボックスが表示されます。



2. 次のいずれかのマッピングパラメータオプションを選択します。
  - マッピングでのデフォルト値の適用データ統合サービスは、マッピング内で設定されたデフォルトのパラメータ値を適用します。
  - パラメータセットの適用データ統合サービスは、パラメータセットで設定されたパラメータ値を適用します。
  - パラメータファイルの適用データ統合サービスは、パラメータファイルで設定されたパラメータ値を適用します。
3. **[OK]** をクリックします。

Developer tool がマッピングのランタイムインスタンスを生成します。ランタイムマッピングが新しいタブに表示されます。
4. マッピングのランタイムインスタンスを右クリックして、**[検証]** を選択します。

**[検証ログ]** ビューにエラーが表示される場合は、エラーを修正し、再度マッピングを検証します。
5. マッピングが有効であれば、**[ファイル]** > **[保存]** をクリックしてマッピングを保存します。
6. ソーススキーマが変更された後、またはパラメータ値を変更した後に、マッピングを再検証します。

## 動的マッピングの実行

動的マッピングを実行し、変換されたデータをターゲットに書き込みます。

1. **[実行]** > **[マッピングの実行]** をクリックします。

**[マッピングの実行]** ウィンドウに、マッピング実行の進行状況が表示されます。マッピングが実行され、出力がターゲットファイルに書き込まれます。
2. **[ウィンドウ]** > **[ビューの表示]** > **[進行状況]** をクリックして、マッピング実行の進行状況を表示します。

**[進行状況]** ビューが開きます。
3. マッピング実行と次のマッピング実行の間でパラメータ値を変更します。
4. ソーススキーマが変更された後、またはパラメータ値を変更した後に、マッピングを再実行します。

## 第 9 章

# 動的マッピングの使用例

この章では、以下の項目について説明します。

- [使用例: リレーショナルソースのメタデータ変更のための動的マッピング, 183 ページ](#)
- [使用例: さまざまなソースおよびターゲットの動的マッピングの再利用, 194 ページ](#)

## 使用例: リレーショナルソースのメタデータ変更のための動的マッピング

顧客注文の合計額を集計する必要がある組織の開発者という立場を想定してみましょう。その組織では、顧客データと顧客注文データを 2 つのテーブルとして異なる部門から毎週受信しています。各部門では頻繁に、カラムの順序を変更したり、テーブルに新規のカラムを追加したりします。開発者は、ソーススキーマの変更に対応して顧客注文の合計額を集計できる動的マッピングを開発する必要があります。

### ソーステーブル

CUSTOMER と ORDERS は、マッピング内の読み取りトランスフォーメーションのソーステーブルです。

次の表に、C\_CUSTKEY カラムがプライマリキーとして設定された CUSTOMER テーブルの各カラムとメタデータを示します。

名前	ネイティブタイプ	精度	スケール
C_CUSTKEY	number(p,s)	38	0
C_NAME	varchar2	25	0
C_ADDRESS	varchar2	40	0
C_NATIONKEY	number(p,s)	38	0
C_PHONE	varchar2	15	0
C_ACCTBAL	number(p,s)	10	2
C_MKTSEGMENT	varchar2	10	0

次の表に、ORDERS テーブルの各カラムとメタデータを示します。

名前	ネイティブタイプ	精度	スケール
O_ORDERKEY	number(p,s)	38	0
O_CUSTKEY	number(p,s)	38	0
O_ORDERSTATUS	varchar2	1	0
O_TOTALPRICE	number(p,s)	10	2
O_ORDERDATE	date	19	0
O_ORDERPRIORITY	varchar2	15	0
O_CLERK	varchar2	15	0
O_SHIPPRIOIRITY	number(p,s)	30	0

## ターゲットテーブル

CUSTOMERSUMMARY は、マッピング内の書き込みトランスフォーメーションのターゲットテーブルです。

次の表に、CUSTOMERSUMMARY テーブルの各カラムとメタデータを示します。

名前	ネイティブタイプ	精度	スケール
C_CUSTKEY	number(p,s)	38	0
C_NAME	varchar2	25	0
C_ADDRESS	varchar2	40	0
C_NATIONKEY	number(p,s)	38	0
C_PHONE	varchar2	15	0
C_ACCTBAL	number(p,s)	10	2
C_MKTSEGMENT	varchar2	10	0
C_TOTALAMOUNT	number(p,s)	10	2

## 動的マッピング

マッピング m\_CustomerLoad を作成して、次の動的マッピング機能を設定します。

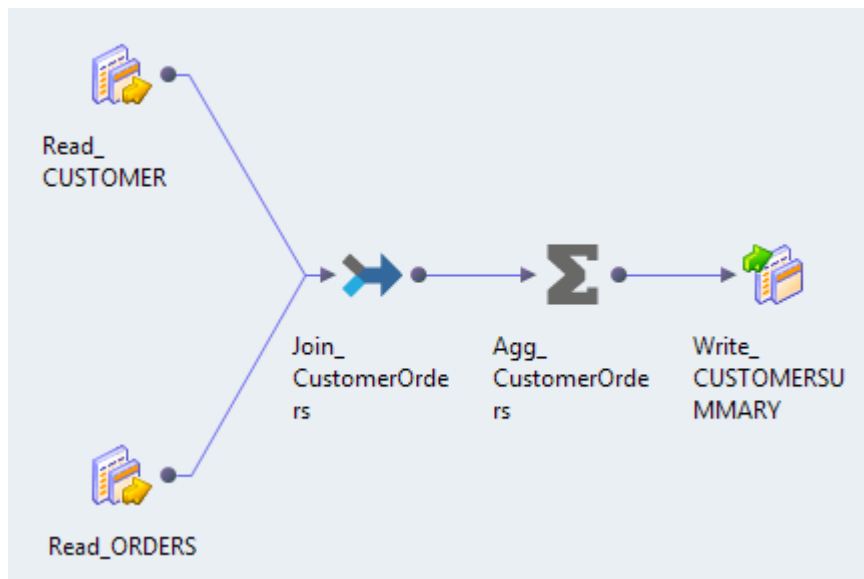
- 動的ソースから読み取ることができる読み取りトランスフォーメーション
- 新しいカラムおよび変更されたカラムを渡すことができる、ダウンストリームトランスフォーメーションの動的ポート
- 動的ターゲットに書き込むことができる書き込みトランスフォーメーション
- ポートを実行時に書き込みトランスフォーメーションに接続できるランタイムリンク



マッピングを実行すると、データ統合サービスによって以下のタスクが実行されます。

1. データオブジェクトの構造とソースファイルにおけるメタデータの変更を取得します。
2. 新しいカラムおよび変更されたカラムを動的ポート経由で各トランスフォーメーションに渡します。
3. 新しいポートおよび変更されたポートを書き込みトランスフォーメーションに接続します。
4. 変換されたデータをターゲットに書き込みます。

次の図は、マッピング内のオブジェクトを示しています。



マッピングには次のオブジェクトが含まれます。

Read\_CUSTOMER

リレーショナルソース CUSTOMER を表す読み取りトランスフォーメーション。このリレーショナルテーブルには、顧客ごとに 1 行が存在しています。

Read\_ORDERS

リレーショナルソース ORDERS を表す読み取りトランスフォーメーション。このリレーショナルテーブルには、顧客注文ごとにそれぞれ 1 行が存在しています。

Join\_CustomerOrders

CUSTOMER ソースと ORDERS ソースを結合するジョイナトランスフォーメーション。

Agg\_CustomerOrders

顧客注文合計を集計するアグリゲータトランスフォーメーション。

Write\_CUSTOMERSUMMARY

リレーショナルターゲット CUSTOMERSUMMARY を表す書き込みトランスフォーメーション。このリレーショナルテーブルには、顧客別の注文合計の集計値を書き込むマッピング用のカラムが含まれています。

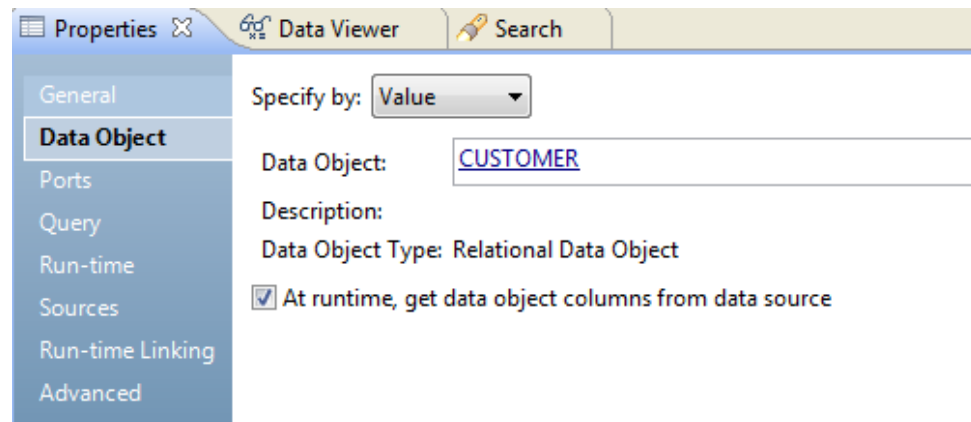
## 手順 1. 読み取りトランスフォーメーションの設定

実行時に動的ソースから直接、カラムおよびメタデータの変更を取得するように、読み取りトランスフォーメーションを設定します。

1. CUSTOMER および ORDERS リレーショナルデータオブジェクトを表す 2 つの読み取りトランスフォーメーションを追加します。

2. 実行時にソースから直接、カラムおよびメタデータの変更を取得するように、Read\_CUSTOMER トランスフォーメーションを設定します。
  - a. Read\_CUSTOMER トランスフォーメーションを選択します。
  - b. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。
  - c. **【実行時に、データソースからデータオブジェクトのカラムを取得します】** を選択します。

次の画像は、Read\_CUSTOMER トランスフォーメーションの **【データオブジェクト】** タブの設定を示しています。



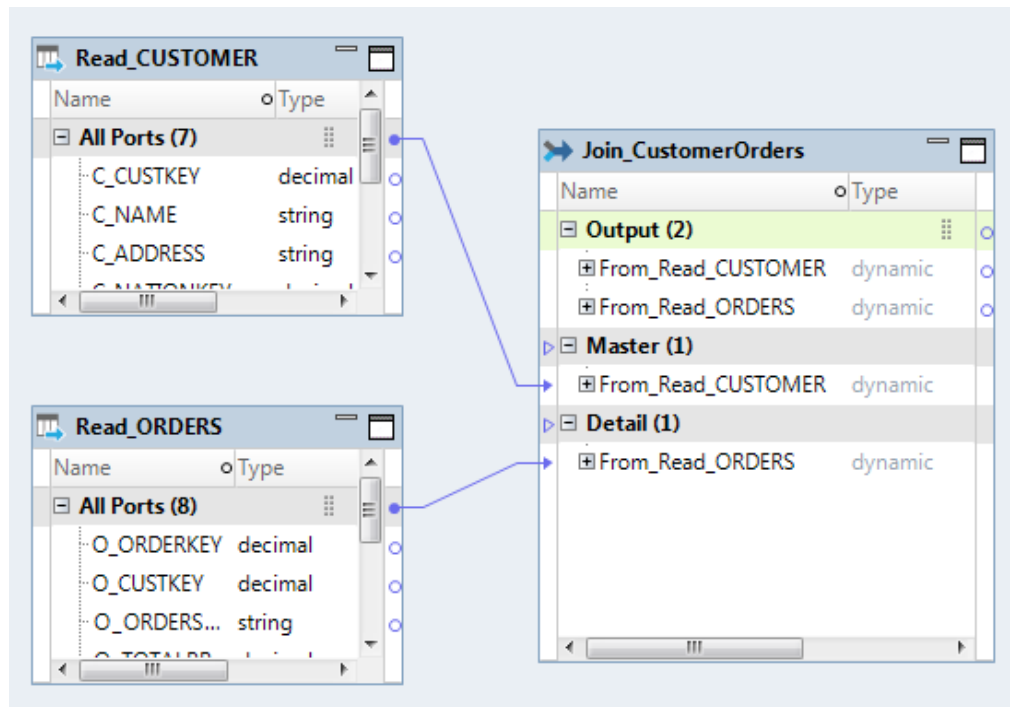
3. 実行時にソースから直接、カラムおよびメタデータの変更を取得するように、Read\_ORDERS トランスフォーメーションを設定します。
  - a. Read\_ORDERS トランスフォーメーションを選択します。
  - b. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。
  - c. **【実行時に、データソースからデータオブジェクトのカラムを取得します】** を選択します。

## 手順 2. ジョイナトランスフォーメーションの設定

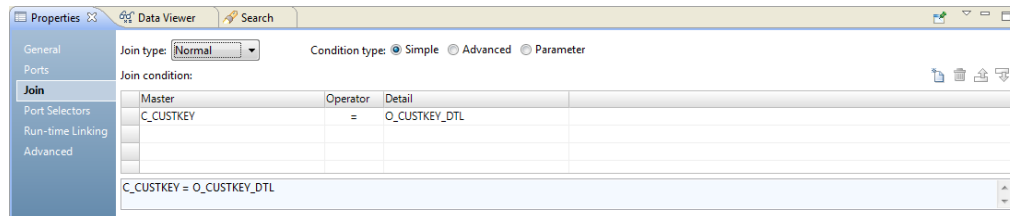
マッピングにジョイナトランスフォーメーションを追加して、読み取りトランスフォーメーションから新しいカラムおよび変更されたカラムを受信するように、動的ポートを設定します。2つのソーステーブル CUSTOMER と ORDERS を結合するための結合条件を定義します。

1. ジョイナトランスフォーメーション Join\_CustomerOrders をマッピングに追加します。
  2. 以下の手順で、ジョイナトランスフォーメーションに動的ポートを作成します。
    - a. Read\_Customer トランスフォーメーションの **【すべてのポート】** グループを、ジョイナトランスフォーメーションの **【マスタ】** グループにドラッグします。  
Developer tool は、マスタグループと出力グループに動的ポート From\_Read\_CUSTOMER を作成します。
    - b. Read\_Orders トランスフォーメーションの **【すべてのポート】** グループを、ジョイナトランスフォーメーションの **【詳細】** グループにドラッグします。  
Developer tool は、詳細グループと出力グループに動的ポート From\_Read\_ORDERS を作成します。
- 動的ポートには、対応する読み取りトランスフォーメーションのすべてのポートが生成されたポートとして含まれます。

次の図では、読み取りトランスフォーメーションの「すべてのポート」グループが、ジョイナトランスフォーメーション内の2つの動的ポートにリンクされています。



3. **【プロパティ】** ビューで、**【結合】** タブをクリックします。
  4. **【新規】** ボタンをクリックし、結合条件として  $C\_CUSTKEY = O\_CUSTKEY\_DTL$  を定義します。
- 次の図は、結合条件が定義された **【結合】** タブを示しています。

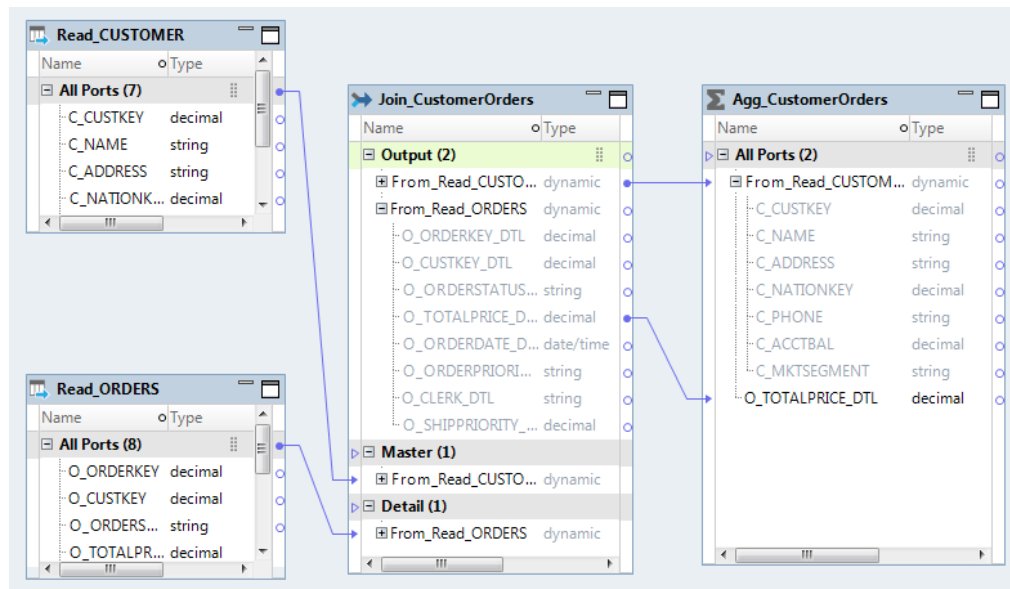


### 手順 3. アグリゲータトランスフォーメーションの設定

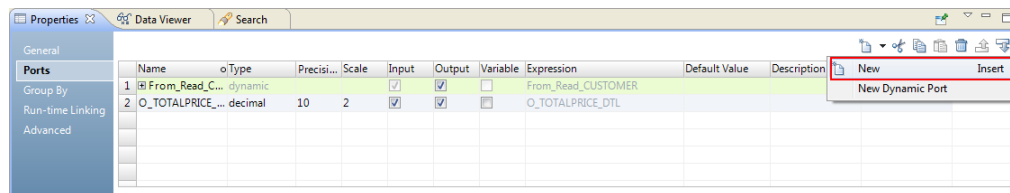
マッピングにアグリゲータトランスフォーメーションを追加して、ジョイナトランスフォーメーションから新しいカラムおよび変更されたカラムを受信するように、動的ポートを設定します。集計式を作成して、顧客注文の合計価格を計算し、さらに顧客別に集計をグループ化します。

1. アグリゲータトランスフォーメーション Agg\_CustomerOrders をマッピングに追加します。
2. 以下の手順で、アグリゲータトランスフォーメーションに動的ポートを作成します。
  - a. ジョイナトランスフォーメーションの出力グループから、動的ポート From\_Read\_CUSTOMER をアグリゲータトランスフォーメーションにドラッグします。  
アグリゲータトランスフォーメーションに、動的ポート From\_Read\_CUSTOMER が表示されます。
  - b. ジョイナトランスフォーメーションの出力グループ内の動的ポート From\_Read\_ORDERS から、生成されたポート O\_TOTALPRICE\_DTL をアグリゲータトランスフォーメーションにドラッグします。

次の図では、ジョイント変換のポートがアグリゲータ変換にリンクされています。



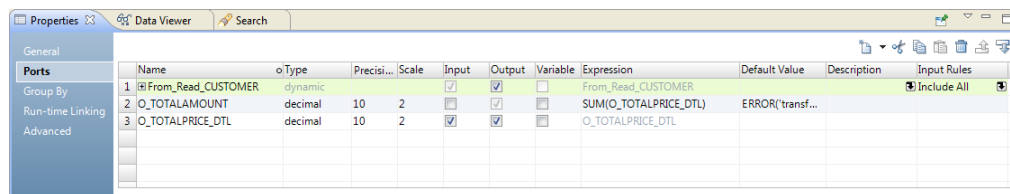
3. [プロパティ] ビューの [ポート] タブをクリックします。
4. [新規] ボタンをクリックして、注文価格を集計するポートを作成します。



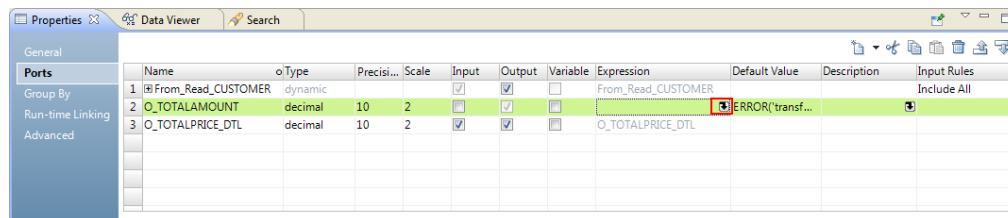
Developer tool は、Field という名前の新しいポートを作成します。

5. 新しいポートを選択し、各カラムの値を次のように変更します。
  - 名前: O\_TOTALAMOUNT
  - タイプ: decimal
  - 精度: 10
  - スケール: 2
  - 入力: 選択を解除してこのポートを出力専用ポートにします。

次の図は、アグリゲータ変換のポートを示しています。

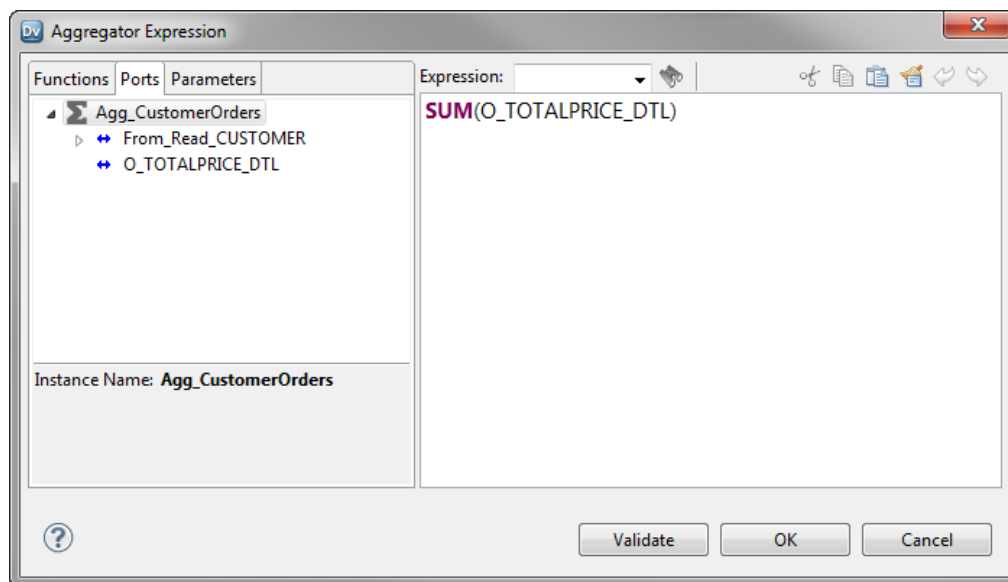


6. O\_TOTALAMOUNT ポートの式カラムで、**【開く】** ボタンをクリックします。



**【アグリゲータ式】** ウィンドウが表示されます。

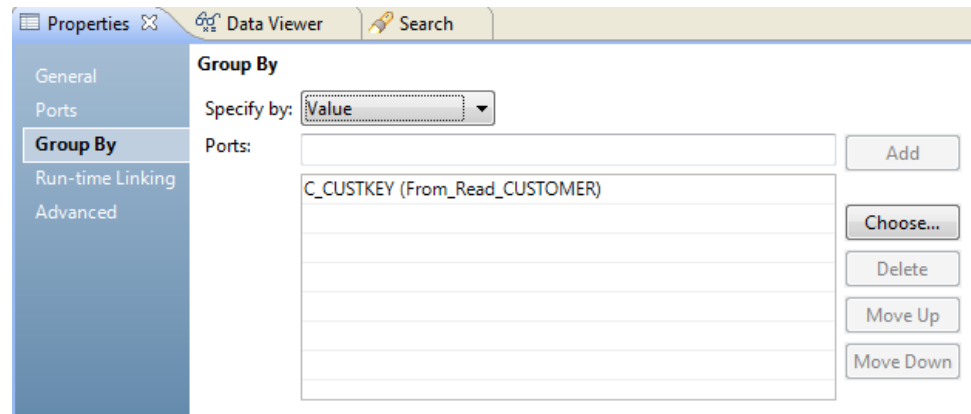
7. エディタの既存の式を、次の式で置換します。SUM(O\_TOTALPRICE\_DTL)



8. **【検証】** をクリックして式を検証します。
9. **【OK】** をクリックします。
10. **【OK】** をクリックして、**【アグリゲータ式】** エディタを終了します。
11. **【プロパティ】** ビューで **【グループ化】** タブをクリックします。
12. 市場区分別に合計価格を集計するグループ化ポートを、次の手順で指定します。
- 【指定元:】** ドロップダウンリストで **【値】** が選択されていることを確認します。
  - 【選択】** をクリックします。
- 【ポート】** ダイアログボックスが表示されます。

- c. C\_CUSTKEY の横のチェックボックスを選択し、[OK] をクリックします。

次の図は、選択されたグループ化ポートを示しています。



アグリゲータトランスフォーメーションデータをプレビューして、期待どおりの結果になっていることを確認できます。マッピングエディタでアグリゲータトランスフォーメーションを右クリックし、**【データビューアの実行】**を選択します。トランスフォーメーションによって計算されたデータが、**【データビューア】**ビューに表示されます。

	C_CUSTKEY	C_NAME	C_ADDRESS	C_NATIONKEY	C_PHONE	C_ACCTBAL	C_MKTSEGMENT	O_TOTALAMOUNT	O_TOTALPRICE_DTL
1	65536	Customer#000065536	QK9rk0yHs3...	14	24-965-688-5...	833.21	BUILDING	3320391.15	105991.01
2	131072	Customer#000131072	EHF8GcoL4...	9	19-862-247-6...	3090.02	BUILDING	1178715.91	52437.51
3	256	Customer#000000256	eJ6AggYh80...	10	20-229-271-4...	1299.92	HOUSEHOLD	2925500.20	61122.48
4	65792	Customer#000065792	DLwqCXA0h...	7	17-754-692-6...	8847.80	BUILDING	1145637.31	152952.65
5	512	Customer#000000512	e5 kymvjf6V...	2	12-144-416-6...	3937.58	BUILDING	847430.41	130631.83
6	131584	Customer#000131584	G 24DXCJ,x...	6	16-354-100-1...	1982.52	FURNITURE	3795211.12	189277.59

Row 1 to 1,000

## 手順 4.書き込みトランスフォーメーションの設定

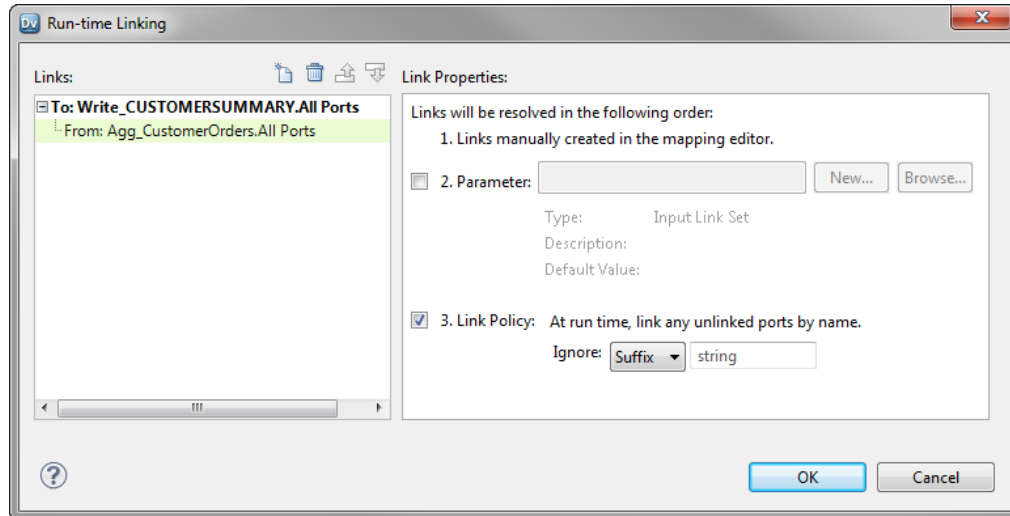
書き込みトランスフォーメーションを追加して、実行時にターゲットから直接、カラム変更を取得するように書き込みトランスフォーメーションを設定します。

1. CUSTOMERSUMMARY リレーショナルデータオブジェクトを書き込みトランスフォーメーションとして追加します。  
書き込みトランスフォーメーションが、Write\_CUSTOMERSUMMARY としてエディタに表示されます。
2. 書き込みトランスフォーメーションが、メタデータの変更を自動的に再インポートするように設定されていることを確認します。
  - a. **【プロパティ】** ビューで **【全般】** タブをクリックします。
  - b. **【入力ポートの同期】** が選択されていることを確認します。
3. 実行時にターゲットテーブルから直接カラムを取得するように、書き込みトランスフォーメーションを設定します。
  - a. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。
  - b. **【実行時に、データソースからデータオブジェクトのカラムを取得します】** を選択します。

## 手順 5.ランタイムリンクの作成と設定

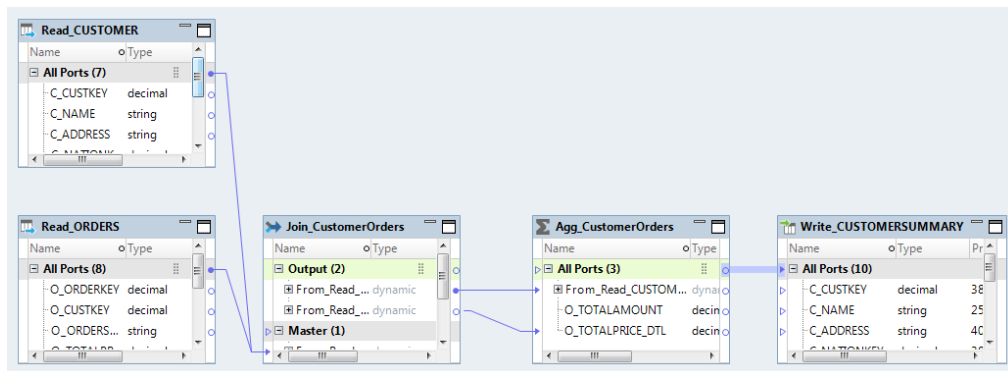
書き込みトランスフォーメーションに対するランタイムリンクを作成し、実行時にポート名でリンクを確立および解決するように、リンクポリシーを設定します。

1. Ctrl キーを押して、アグリゲータトランスフォーメーションから [すべてのポート] グループを、書き込みトランスフォーメーションの [すべてのポート] グループにドラッグします。  
[ランタイムリンク] ダイアログボックスが表示されます。
2. [リンクのプロパティ] 領域の [リンクポリシー] が選択されていることを確認します。これにより、実行時に自動的に名前がポートがリンクされます。



3. [OK] をクリックします。

Developer tool は、アグリゲータトランスフォーメーションと書き込みトランスフォーメーション間のランタイムリンクを作成します。



## 手順 6.マッピングの検証と実行

マッピングを検証して実行します。ターゲットデータオブジェクト内のデータをプレビューして結果を確認します。

1. マッピングエディタで、[編集] > [検証] をクリックします。
2. マッピングが有効であれば、[ファイル] > [保存] をクリックしてマッピングを保存します。
3. [実行] > [マッピング] をクリックします。

【マッピングの実行】 ウィンドウに、マッピング実行の進行状況が表示されます。 マッピングが実行され、出力がターゲットファイルに書き込まれます。

4. 【オブジェクトエクスプローラ】 ビューで、プロジェクト内の CUSTOMERSUMMARY データオブジェクトを探してダブルクリックします。

データオブジェクトがエディタに表示されます。

5. 【ウィンドウ】 > 【ビューの表示】 > 【データビューア】 をクリックします。

【データビューア】 ビューが表示されます。

6. 【データビューア】 ビューで、【実行】 をクリックします。

【データビューア】 ビューが起動して、データが表示されます。

この例では、C\_TOTALAMOUNT カラムに、集計された顧客注文の合計価格が表示されています。

Output								
Name: CUSTOMERSUMMARY								
	C_CUSTKEY	C_NAME	C_ADDRESS	C_NATIONKEY	C_PHONE	C_ACCTBAL	C_MKTSEGME...	C_TOTALAMOUNT
1	287	Customer#000...	KTsaTAJRC0e...	4	14-330-840-6321	1734.18	MACHINERY	701351.00
2	1055	Customer#000...	Z3AggyEMPM...	7	17-802-131-7180	639.93	HOUSEHOLD	1549236.00
3	32	Customer#000...	JD2xZzi Umld,D...	15	25-430-914-2194	3471.53	BUILDING	1336868.00
4	544	Customer#000...	Jv7vcn,oE,HEy...	5	15-572-651-1323	4974.68	AUTOMOBILE	2900638.00
5	289	Customer#000...	NUilehg0nVOk...	10	20-456-773-7693	-215.75	AUTOMOBILE	2893675.00
6	545	Customer#000...	AsYw6k,nDUQ...	10	20-849-123-8918	7505.33	AUTOMOBILE	975375.00
7	1057	Customer#000...	xyV8 FbW4xS,J...	24	34-750-735-1314	-377.11	AUTOMOBILE	2838452.00
8	34	Customer#000...	Q6G9wZ6dncz...	15	25-344-968-5422	8589.70	HOUSEHOLD	4295230.00
9	290	Customer#000...	8OIPT9G 8UqV...	4	14-458-625-5633	1811.35	MACHINERY	618490.00
10	1058	Customer#000...	R0NIEcSVDQ4r...	19	29-818-620-9637	6807.55	MACHINERY	1252089.00

## 手順 7. ソーススキーマ変更後のマッピングの実行

顧客データテーブルと顧客注文データテーブルを提供している部門が、各テーブルに新しいカラム Comments を追加します。 動的マッピングのカラム変更を確認して、マッピングを検証および再実行します。 ターゲットデータオブジェクト内のデータをプレビューして、更新された結果を確認します。

次の表に、新しい C\_COMMENT カラムで更新された CUSTOMER テーブルの各カラムとメタデータを示します。

名前	ネイティブタイプ	精度	スケール
C_CUSTKEY	number(p,s)	38	0
C_NAME	varchar2	25	0
C_ADDRESS	varchar2	40	0
C_NATIONKEY	number(p,s)	38	0
C_PHONE	varchar2	15	0
C_ACCTBAL	number(p,s)	10	2
C_MKTSEGMENT	varchar2	10	0
C_COMMENT	varchar2	117	0

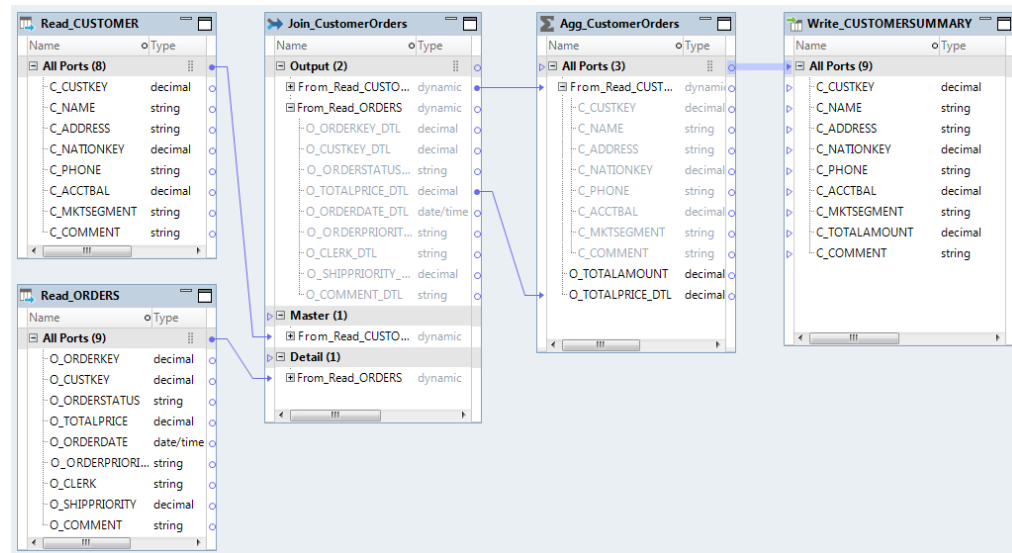


次の表に、新しい O\_COMMENT カラムで更新された ORDERS テーブルの各カラムとメタデータを示します。

名前	ネイティブタイプ	精度	スケール
O_ORDERKEY	number(p,s)	38	0
O_CUSTKEY	number(p,s)	38	0
O_ORDERSTATUS	varchar2	1	0
O_TOTALPRICE	number(p,s)	10	2
O_ORDERDATE	date	19	0
O_ORDERPRIORITY	varchar2	15	0
O_CLERK	varchar2	15	0
O_SHIPPRIORITY	number(p,s)	30	0
O_COMMENT	varchar2	79	0

1. マッピングエディタで、マッピングに対する変更を確認します。

読み取りおよび書き込みトランスフォーメーションによって、新しいカラムが自動的に反映されます。ジョイントランスフォーメーションおよびアグリゲータトランスフォーメーションの動的ポートには、新しいカラム C\_COMMENT と O\_COMMENT が、それぞれ対応する読み取りトランスフォーメーションから自動的に取り込まれます。



2. 変更されたマッピングを検証するには、**【編集】** > **【検証】** をクリックします。
3. マッピングが有効であれば、**【ファイル】** > **【保存】** をクリックしてマッピングを保存します。
4. **【実行】** > **【マッピング】** をクリックします。  
**【マッピングの実行】** ウィンドウに、マッピング実行の進行状況が表示されます。マッピングが実行され、出力がターゲットファイルに書き込まれます。
5. **【オブジェクトエクスプローラ】** ビューで、プロジェクト内の CUSTOMERSUMMARY データオブジェクトを探してダブルクリックします。

データオブジェクトがエディタに表示されます。

6. **【ウィンドウ】** > **【ビューの表示】** > **【データビューア】** をクリックします。  
**【データビューア】** ビューが表示されます。
7. **【データビューア】** ビューで、**【実行】** をクリックします。  
**【データビューア】** ビューが起動して、データが表示されます。
8. ソーススキーマの変更後、マッピングに期待どおりの結果が表示されていることを確認します。  
C\_TOTALAMOUNT カラムに、集計された顧客注文の合計価格が表示されています。

## 使用例: さまざまなソースおよびターゲットの動的マッピングの再利用

string 値の先頭と末尾の空白を削除するためにさまざまなデータファイルをクリーンする必要がある組織の開発者という立場を想定してみましょう。データファイルにはさまざまなカラム名と、複数の string 型カラムが含まれています。開発者は、さまざまなソースから文字列の先頭と末尾の空白を削除し、出力をさまざまなターゲットに書き込む動的マッピングを開発する必要があります。

### ソースファイル

ソースファイルは、先頭と末尾に空白を含む string 型データを含むフラットファイルです。読み取りトランスフォーマーのソースファイルには Customer\_FF および orders\_FF が含まれています。

この例では、以下の手順に従って、最初のマッピング実行で Customer\_FF ファイルから読み取り、2 番目のマッピング実行で orders\_FF ファイルから読み取っています。

#### Customer\_FF のカラムとデータ

Customer\_FF には、以下のカラムがあります。

```
C_Id  
C_Fullname  
C_title  
C_comp  
C_addr  
C_suite  
C_city  
C_state  
C_zip5  
C_country  
C_phone  
C_fax  
C_date  
C_email  
C_description
```

ここで、C\_ID および C\_zip5 カラムのデータ型は数値、その他のカラムのデータ型は string です。

Customer\_FF には、以下のデータが含まれています。

```
C_Id,C_Fullname,C_title,C_comp,C_addr,C_suite,C_city,C_state,C_zip5,C_country,C_phone,C_fax,C_date,C_email  
,C_description  
1, Smith John,Account Executive,DKR MANAGEMENT COMPANY INC,100 High Street,5406,Anytown,TN,22342,USA,  
4047668150,2124031386,31/08/1985,bwilliams@yahoo.com, ACTIVE  
2,Balasubramanian Krishna,Account Executive,EASTON & COMPANY,71 Congress Parkway,789,Bangalore,Karnataka,  
38103,India,4046345228,4151689756,29/10/1985,bmatthewc@univ.edu, ACTIVE  
3, Johnson Lars,Regional Sales Exec,GREATER BAY BANCORP,123 Snow St.,43543,St. Paul,MN,55103,USA,  
4046581534,6122945948,7/9/1992, ehpuniv.edu,INACTIVE
```

4,Zogby Kevin,Regional Sales Exec, HEWLETT-PACKARD,317 29th. St.,5856,San Francisco,CA,94116,USA,  
 4042662730,4155466814,7/8/1985,grobertwuniv.edu, ACTIVE  
 5,Franklin Roosevelt,Sales Representative,JAYD TRADING,1511 Wacker Dr,6334,Chicago,IL,60606,USA,  
 7703965851,2065075486,20/10/1982,trichard@univ.edu,INACTIVE  
 6, Cruz Emilio,Sales Representative,JEFFERSON-PILOT LIFE INSURANCE,700 Ponce de Leon Blvd,757,Miami,FL,  
 33134,USA,4043500799,2127655499,31/07/1983,ahelle@mailcity.com, ACTIVE  
 7, King BB,Sales Representative,KUWAIT PETROLEUM CORPORATION,18 Beale St,967,Memphis,TN,38103,USA,  
 4046243979,2151717120,27/09/1989, glizziem@univ.edu ,INACTIVE  
 8,Presley Elvis,Sales Representative,PRINCIPIA PARTNERS,45 N Green St.,43546,Tupelo,MS,38804,USA,  
 4043733125,3311313591,26/07/1992,, ACTIVE  
 9,Olson Floyd,Acct MGR., SOLITON ASSOCIATES INC.,21 Lake Harriet Pkwy,869790,Mineapolis,MN,55410,USA,  
 7706425402,3232429056,27/08/1993,,INACTIVE  
 10,Chu Steven,Account Executive,WQXR,2100 Sepulveda Blvd,3434,Los Angeles,CA,90049,USA,  
 4042319005,2126509756,29/09/1988,akennetha@univ.edu, ACTIVE

例えば、1 行目と 3 行目では名前の先頭にスペースがあります。

1, Smith John,  
 3, Johnson Lars,

## orders\_FF のカラムとデータ

orders\_FF には、以下のカラムがあります。

OrderID  
 Customer\_ID  
 Company  
 CompanyAddress  
 CompanyCity  
 CompanyState  
 CompanyZip  
 OrderContact  
 DeliveryAddress  
 DeliveryCity  
 DeliveryState  
 PaymentType  
 PaymentTerms  
 Title  
 DeliveryOption  
 DeliveryVendor  
 ConfirmationCode  
 OrderAmount  
 OrderType  
 ProductDescription

ここで、Customer\_ID カラムのデータ型は数値、その他のカラムのデータ型は string です。

orders\_FF には、以下のデータが含まれています。

0-5079,10110085,JOSEPH TAL LYON & ROSS,96 FISHER ROAD, MAHWAH,NJ,7430,PARKE PERSLEY OR RAYFORD LECROY,  
 96 FISHER ROAD,MAHWAH,NJ,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
 44162,\$21.00 ,Generic,O/L/B P/W L/S TAWNY SHIMMER .08 OZ.  
 0-6658,10110086,NRCA,10255 W.HIGGINS RD., ROSEMONT,IL,60018-5607,ROLANDA SORTO,10255 W.HIGGINS  
 RD.,ROSEMONT,IL,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
 44163,\$56.40 ,Generic,O-L-B PW LIPSTYLO LASTING PERFECTION .08 OZ.  
 0-8195,10110087,POND EQUITIES,4522 FT. HAMILTON PKWY., BROOKLYN,NY,11219, KONSTANTIN PEDDICORD,4522 FT.  
 HAMILTON PKWY.,BROOKLYN,NY,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
 44164,\$78.00 ,Generic,O/L/B P/W L/S TAWNY SHIMMER LASTING PERFECTION LIPSTYLO TAWNY SHIMMER .08 OZ.  
 0-9130,10110088, SCHRODER & COMPANY ,787 SEVENTH AVENUE, NEW YORK,NY,10019,GIORGIA  
 TWITCHELL,787 SEVENTH AVENUE,NEW YORK,NY,American Express,CHARGE,Account Executive,UPA,United Parcel  
 Service Air,44165,\$14.00 ,Generic,A/COL L PERFECTION L/S REF P SUPREME LASTING PERFECTION LIPSTYLO TAWNY  
 SHIMMER .08 OZ.  
 0-9352,10110089,YUASA TRADING COMPANY (AMERICA),150 EAST 52ND STREET,NEW YORK,NY,10005,STEFFI MCGLOWN,150  
 EAST 52ND STREET,NEW YORK,NY,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
 44166,\$54.00 ,Generic,O/L/B L PERFECTION REF LIPSTYLO COFFEE PEACH SUPREME .08 OZ.  
 0-9517,10110090,DAI ICHI KANGYO BANK,1 WORLD TRADE CENTRE SUITE 49 - 11,NEW YORK,NEW YORK,10048,AIKEN  
 DOBRICK,1 WORLD TRADE CENTRE SUITE 49 - 11,NEW YORK,NEW YORK,American Express,CHARGE,Account  
 Executive,UPR,United Parcel Service Red,44167,\$58.00 ,Generic,LASTING PERFECTION LIP COLOR HOLLYWOOD  
 GLAMOUR 1.7 G MAUVE ICE #752  
 0-9639,10110091,FIRST GLOBAL SECURITIES,614 EAST COLORADO BLVD.,PASADENA,CA,91101, KIRSTENI SIPPEL,614  
 EAST COLORADO BLVD.,PASADENA,CA,American Express,CHARGE,Account Executive,FSO,Federal Express Overnight,

```

44168,$24.00 ,Generic,A/COL L PERFECTION L/S REF P SUPREME .08 OZ.
0-9761,10110092,MILTON PARTNERS,56 MASON STREET, GREENWICH ,CT,6830,ORLANTA DYSON,56 MASON
STREET,GREENWICH,CT,American Express,CHARGE,Account Executive,UPI,United Parcel Service International,
44169,$75.20 ,Generic,LASTING PERFECTION LIPSTYLO PEACH SU .08 OZ.
0-9883,10110093, TAX ANALYSTS ,6830 N. FAIRFAX DRIVE,ARLINGTON,VA,22213,NEWLIN MCCART,6830 N. FAIRFAX
DRIVE,ARLINGTON,VA,American Express,CHARGE,Account Executive,FSO,Federal Express Overnight,
44170,$275.40 ,Generic,O/L/B L PERFECTION L/STYLO REF P SUPRE
0-5438,10110094,VECTORMEX,535 MADISON AVENUE,NEW YORK,NY,10022,LONNA HUGGINS,535 MADISON AVENUE,NEW
YORK,NY,American Express,CHARGE,Account Executive,FSO,Federal Express Overnight,
44171,$60.00 ,Generic,LASTING PERFECTION DOUBLE PERFORMANCE LIPSTICK PEACH SUPREME .08 OZ.

```

例えば、4 行目では会社名の先頭と末尾にスペースがあります。

```
0-9130,10110088, SCHRODER & COMPANY ,
```

## ターゲットファイル

ターゲットファイルは、マッピングが文字列値の先頭および末尾の空白を削除した後、データを書き出すフラットファイルです。ターゲットデータオブジェクトのターゲットファイルとして customerTrim.csv ファイルを作成します。

異なるデータソースを使用する場合は、パラメータを使用して実行時に出力ファイル名を変更します。データ統合サービスは、ターゲットファイル名のパラメータ値に基づいて出力ファイルを作成し、システム上の Informatica サービスがインストールされているターゲットディレクトリに保存します。

## 動的マッピング

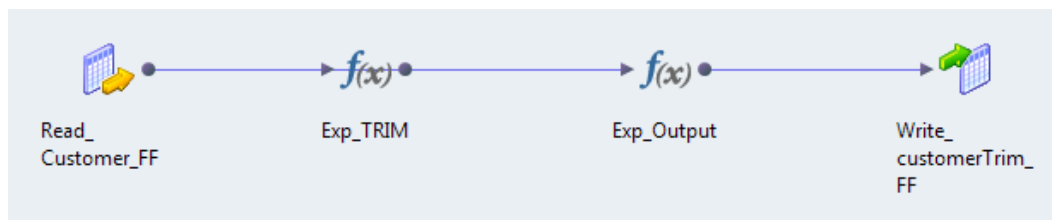
マッピング m\_Replication\_Template を作成して、次の動的マッピング機能を設定します。

- データオブジェクトのパラメータを使用して、さまざまなソースからデータを読み取る読み取りトランスフォーメーション
- 新しいカラムまたは変更されたカラムを渡すことができる、ダウストリームトランスフォーメーションの動的ポート
- 文字列の先頭および末尾の空白を削除する動的式を含む式トランスフォーメーション
- マッピングフローに基づいてターゲットカラムを作成し、ターゲットファイル名としてターゲットデータオブジェクト内のパラメータを使用する書き込みトランスフォーメーション

マッピングを実行すると、データ統合サービスによって以下のタスクが実行されます。

1. ソースデータオブジェクトのパラメータ値に基づいて適切なソースファイルからデータを読み取ります。
2. 新しいカラムおよび変更されたカラムを動的ポート経由でダウストリームトランスフォーメーションに渡します。
3. 動的式を展開し、動的ポート内の生成された各ポートについて式の関数を処理します。
4. マッピングフローに基づいて書き込みトランスフォーメーションでカラムを作成し、変換されたデータをパラメータ値に基づいて適切なターゲットファイルに書き込みます。

次の図は、マッピング内のオブジェクトを示しています。



マッピングには次のオブジェクトが含まれます。

## Read\_Customer\_FF

フラットファイルソースを表す読み取りトランスフォーメーション。フラットファイルには、先頭および末尾に空白を含む文字列データが含まれています。

## Exp\_TRIM

string 型ポートの先頭および末尾の空白を削除する動的式を含む式トランスフォーメーション。

## Exp\_Output

変換された string 型ポートとソースオブジェクトの残りのポートを含む式トランスフォーメーション。

## Write\_customerTrim\_FF

フラットファイルターゲットを表す書き込みトランスフォーメーション。マッピングは出力をフラットファイルターゲットに書き込みます。

## 手順 1. Read\_Customer\_FF 読み取りトランスフォーメーションの設定

リソースタイプのパラメータを使用してマッピング実行ごとにソースデータオブジェクトを変更するように、Read\_Customer\_FF 読み取りトランスフォーメーションを設定します。

1. Customer\_FF フラットファイルデータオブジェクトを表す読み取りトランスフォーメーションを追加します。

読み取りトランスフォーメーションが Read\_Customer\_FF としてエディタに表示されます。

2. **【プロパティ】** ビューで、**【データオブジェクト】** タブをクリックします。

3. **【指定元:】** ドロップダウンリストで、**【パラメータ】** を選択します。

4. **【新規】** をクリックして新しいパラメータを作成します。

**【パラメータ】** ダイアログボックスが表示されます。

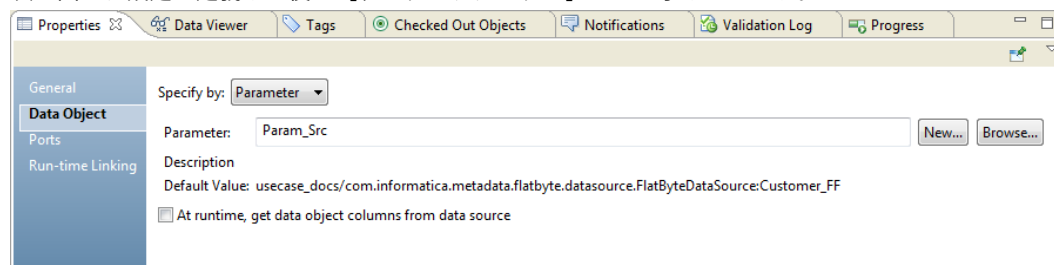
5. パラメータ名として、Param\_Src と入力します。

6. **【デフォルト値】** の **【参照】** をクリックします。

7. **【場所の選択】** ダイアログボックスで、デフォルト値として設定するデータオブジェクトを選択します。

サンプルのデフォルト値は MRS//Cust\_Dept/Customer\_FF です。ここで、MRS はモデルリポジトリサービス、Cust\_Dept は Customer\_FF データオブジェクトが格納されているプロジェクトです。パラメータの値はマッピングの実行時に変更できます。

次の図は、設定を定義した後の **【データオブジェクト】** タブを示しています。



## 手順 2. Exp\_TRIM 式トランスフォーメーションの設定

式トランスフォーメーション Exp\_TRIM をマッピングに追加し、文字列の先頭と末尾のスペースを削除するようにトランスフォーメーションを設定します。

1. 読み取りトランスフォーメーションからカラムを受信するように動的ポートを作成して、文字列ポートのみを含めるように入力ルールを定義します。

2. 動的出力ポートを作成し、文字列の先頭と末尾のスペースを削除するように動的式を定義します。

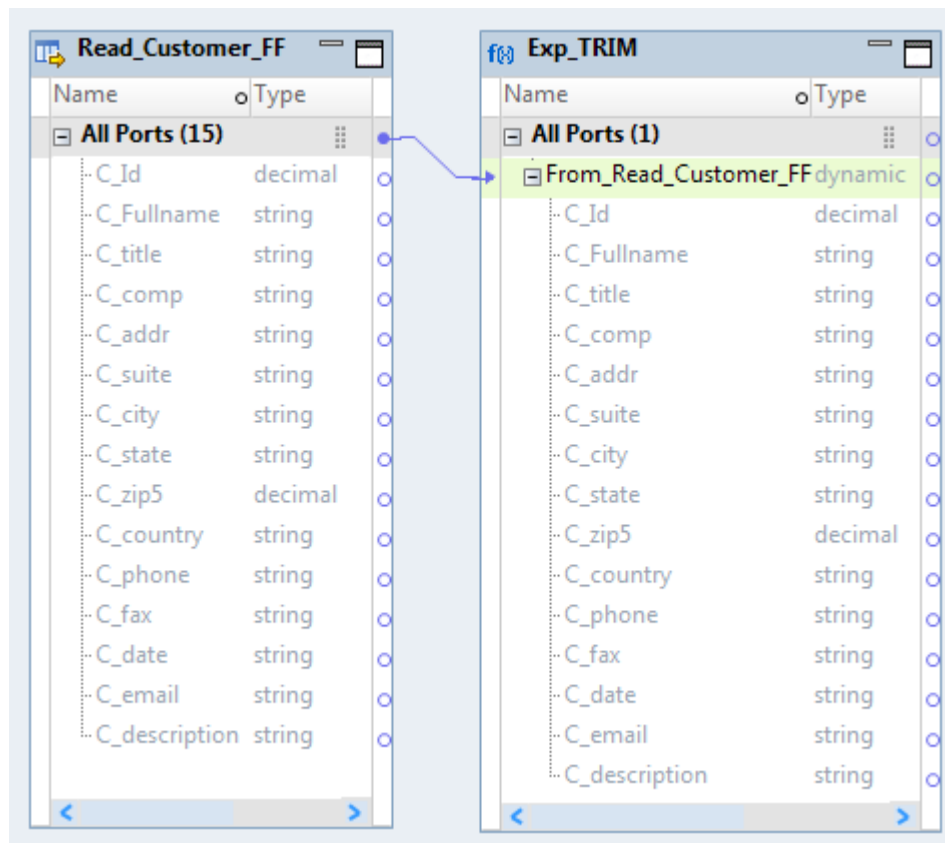
## 動的ポートの作成と入力ルールの定義

読み取りトランスフォーメーションからカラムを受信するように動的ポートを作成します。動的ポートに文字列ポートのみを含めるように入力ルールを定義します。

1. Read\_Customer\_FF トランスフォーメーションの [すべてのポート] グループを、Exp\_TRIM トランスフォーメーションの [すべてのポート] グループにドラッグします。

Developer tool は、Exp\_TRIM トランスフォーメーションに動的ポート From\_Read\_Customer\_FF を作成します。

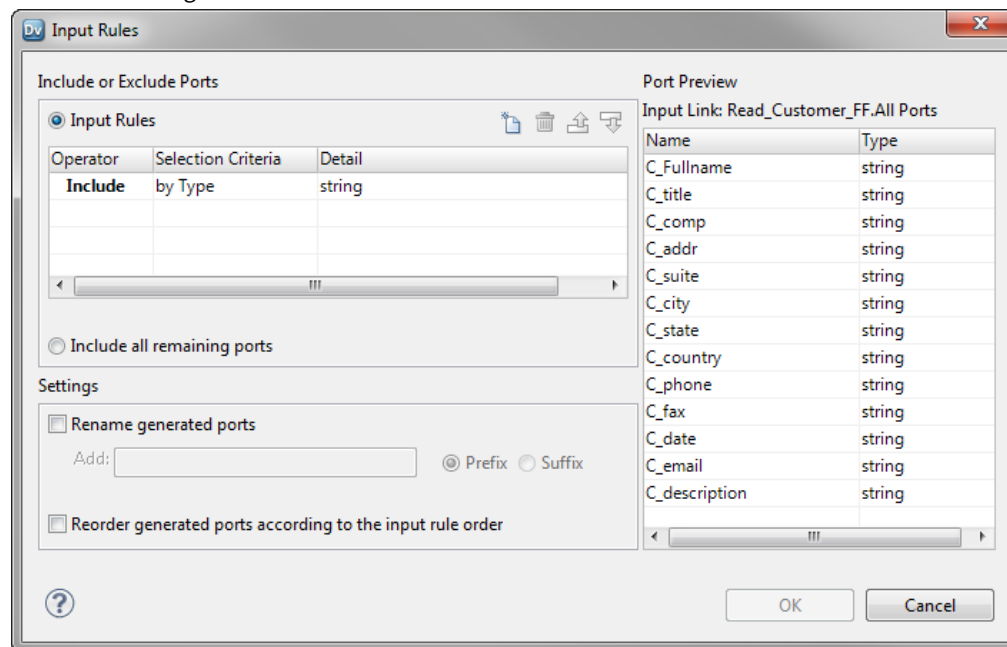
次の図は、読み取りトランスフォーメーションのすべてのポートを生成されたポートとして含む、Exp\_TRIM トランスフォーメーションの動的ポートを示しています。



2. 動的ポートを右クリックして、[入力ルール of 編集] を選択します。  
[入力ルール] ダイアログボックスが表示されます。
3. [選択条件] カラムから [タイプ] を選択します。
4. [詳細] ボタンをクリックして、含めるデータ型を選択します。
5. [入力ルールの詳細: タイプ別] ダイアログボックスで、リストから [string] データ型を選択します。

6. [ポートのプレビュー] 領域で、string 型ポートのみが表示されていることを確認します。

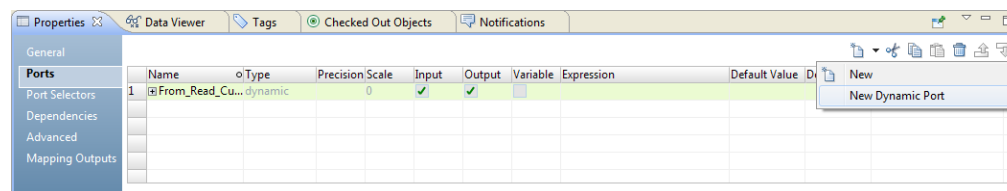
次の図は、[入カルール] ダイアログボックスの [ポートのプレビュー] 領域に表示された、更新された入カルールと string 型ポートを示しています。



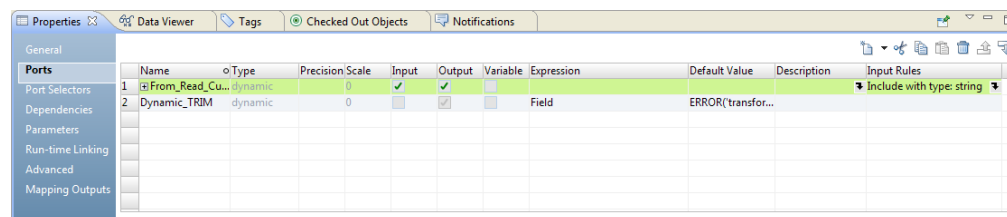
## 動的ポートの作成と動的式の定義

Exp\_TRIM トランスフォーメーションで出力のみのポートとして動的ポートを作成します。文字列の先頭と末尾のスペースを削除するように動的式を定義します。

1. Exp\_TRIM トランスフォーメーションの [プロパティ] ビューで、[ポート] タブをクリックします。
2. [新しい動的ポート] をクリックします。

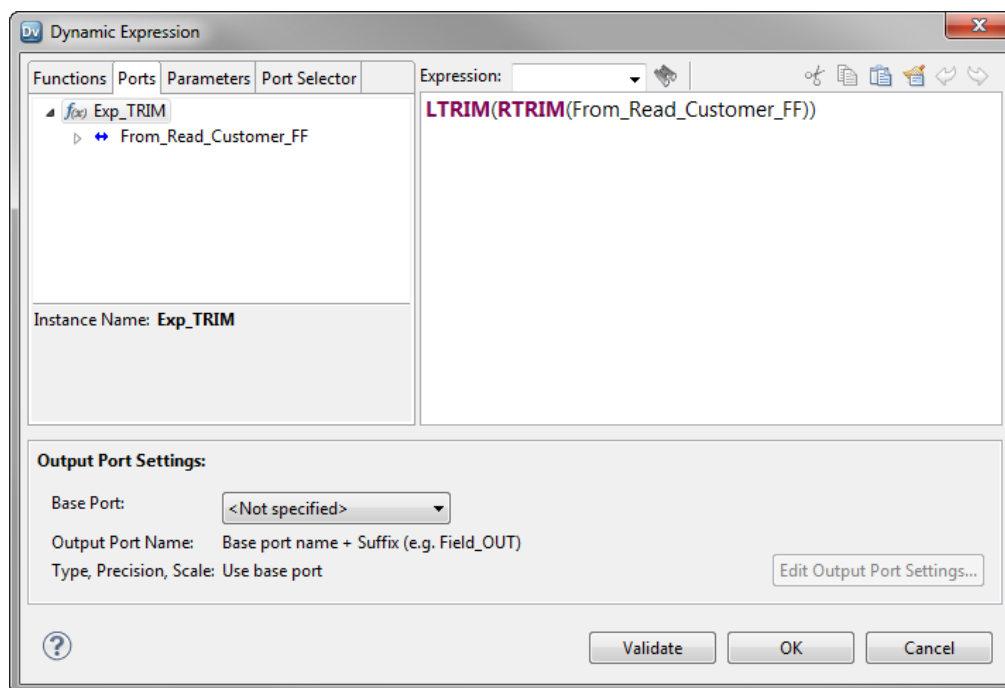


3. [入力] カラムをクリアして、このポートを出力専用ポートにします。
4. Dynamic\_TRIM として作成した動的ポートの名前を変更します。



5. Dynamic\_TRIM 動的ポートの [式] カラムで、[開く] ボタン (🔗) をクリックします。  
[動的式] ウィンドウが開きます。

6. エディタの既存の式を、次の式で置換します。LTRIM(RTRIM(From\_Read\_Customer\_FF))

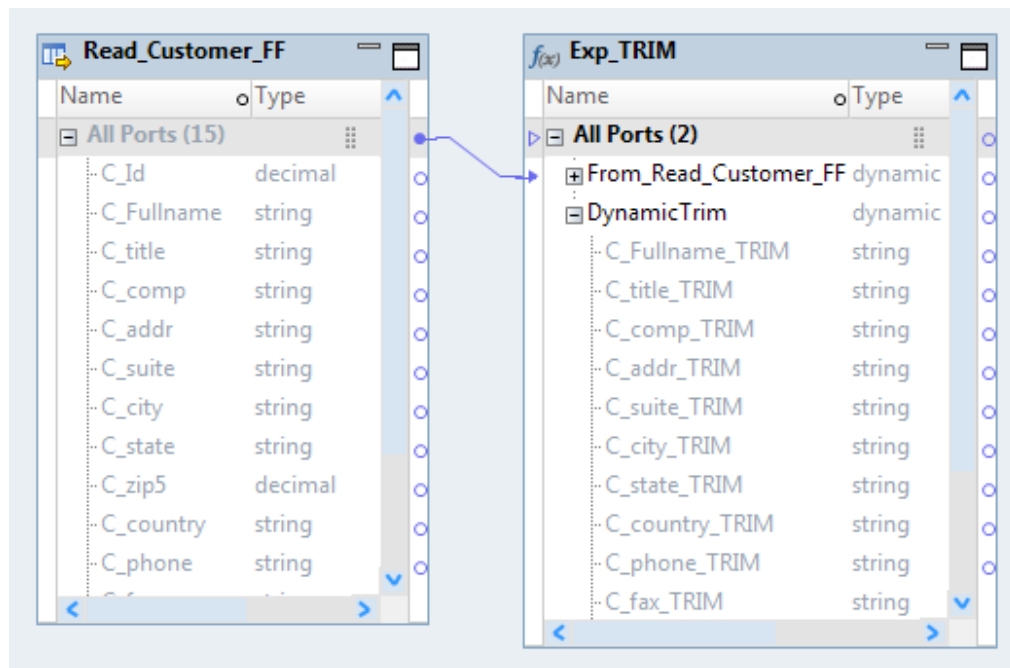


7. **【検証】** をクリックして式を検証します。
8. **【OK】** をクリックして、**【式の検証】** ダイアログボックスを終了します。
9. 式の出力ポートの名前を次のように変更します。
- 【出力ポート設定】** 領域で、ベースポートとして From\_Read\_Customer\_FF を選択します。
  - 【出力ポート設定の編集】** をクリックします。  
【出力ポート設定】 ダイアログボックスが表示されます。
  - 【名前】** リストで、**【ベースポート名+サフィックス】** を選択します。
  - 【サフィックス】** ボックスに「\_TRIM」と入力します。
  - 【OK】** をクリックします。



10. **[OK]** をクリックして、**[動的式]** エディタを終了します。

次の図は、Dynamic\_TRIM 動的ポートと、生成されたポートで名前変更されたものを示しています。



### 手順 3. Exp\_Output 式トランスフォーメーションの設定

式トランスフォーメーション Exp\_Output をマッピングに追加します。Exp\_TRIM トランスフォーメーションから出力ポートを取得するように動的ポートを作成します。読み取りトランスフォーメーションからポートを取得するように別の動的ポートを作成して、未使用のポートのみを含めるように入力ルールを定義します。

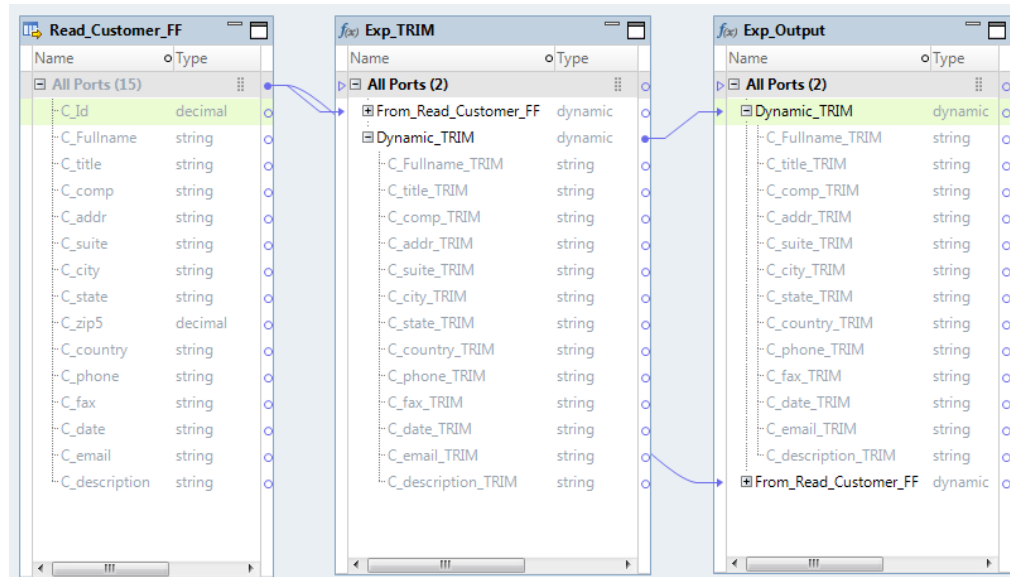
1. Exp\_TRIM トランスフォーメーションから、DynamicTrim 動的ポートを Exp\_Output トランスフォーメーションの **[すべてのポート]** グループにドラッグします。

Developer tool で、Exp\_Output トランスフォーメーションに動的ポート DynamicTrim が作成されます。

2. Read\_Customer\_FF トランスフォーメーションの **[すべてのポート]** グループを、Exp\_Output トランスフォーメーションの **[すべてのポート]** グループにドラッグします。

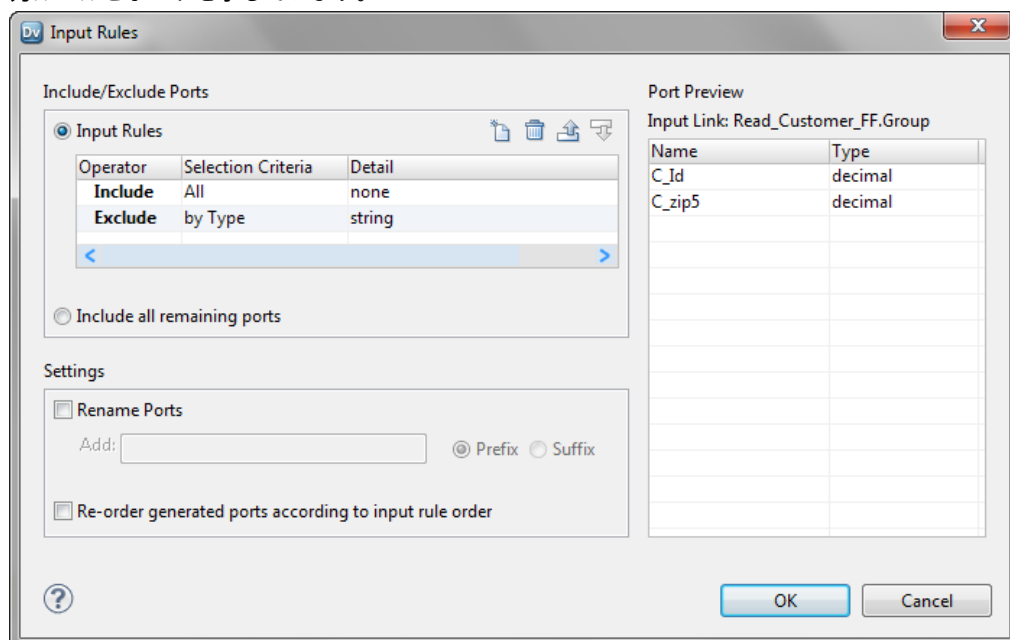
Developer tool で、Exp\_Output トランスフォーメーションに動的ポート From\_Read\_Customer\_FF が作成されます。

次の図は、Exp\_Output トランスフォーメーションの 2 つの動的ポートを示しています。



3. From\_Read\_CUSTOMER\_FF 動的ポートを右クリックし、**【入力規則の編集】** を選択します。  
**【入力規則】** ダイアログボックスが表示されます。
4. **【新規】** アイコンをクリックして、入力規則を追加します。
5. **【演算子】** カラムで **【除外】** を選択します。
6. **【選択条件】** カラムで **【タイプ】** を選択します。
7. **【詳細】** 矢印をクリックして、含めるデータ型を選択します。
8. **【入力規則の詳細: タイプ別】** ダイアログボックスで、リストから **【string】** データ型を選択します。
9. **【ポートのプレビュー】** 領域で、string 型ポートが表示されていないことを確認します。

次の図は、**【入力規則】** ダイアログボックスの **【ポートのプレビュー】** 領域に表示された、更新された入力規則とポートを示しています。





## 手順 4. Write\_customerTrim\_FF 書き込みトランスフォーメーションの設定

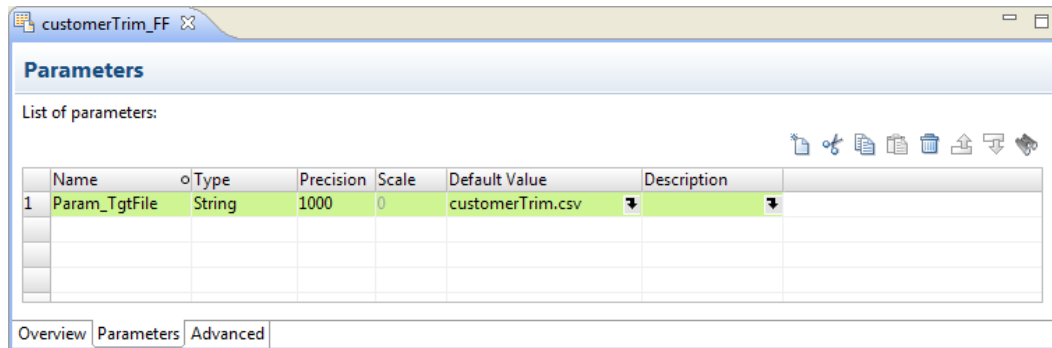
customerTrim\_FF データオブジェクトを作成し、出力ファイル名に string 型のパラメータを使用するように設定します。実行時に Exp\_Output トランスフォーメーションの列に基づいてターゲットファイルを作成するように Write\_customerTrim\_FF トランスフォーメーションを設定します。

### パラメータを使用するためのデータオブジェクトの設定

customerTrim\_FF データオブジェクトを作成して、マッピング内書き込みトランスフォーメーションとして追加します。出力ファイル名として string 型のパラメータを使用するようにデータオブジェクトを設定します。

- customerTrim.csv ファイルに基づいて、customerTrim\_FF データオブジェクトを作成します。
- パラメータを出力ファイルとして使用するには、以下の手順を実行します。
  - データオブジェクトの **【パラメータ】** タブで、**【新規】** ボタン (  ) をクリックして新しいパラメータを作成します。
  - 【名前】** 列で、パラメータ名を Param\_TgtFile に変更します。
  - 【デフォルト値】** 列で、**【開く】** ボタン (  ) をクリックします。  
**【パラメータ値の編集】** ウィンドウが表示されます。
  - デフォルトのファイル名の値として customerTrim.csv を入力し、**【OK】** をクリックします。
- customerTrim\_FF データオブジェクトを保存します。

次の図は、新しいパラメータが表示された **【パラメータ】** タブを示しています。

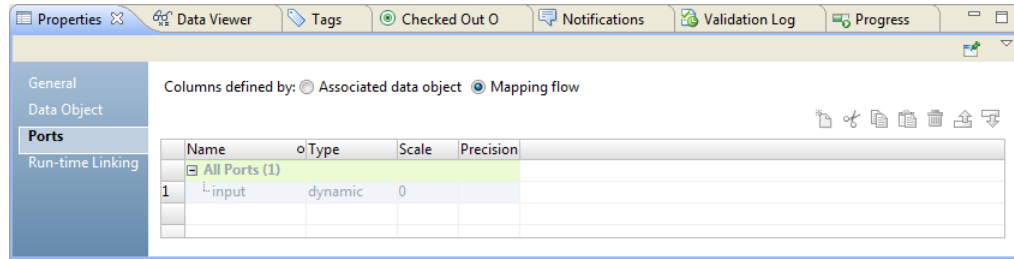


### マッピングフローからのターゲット列の作成

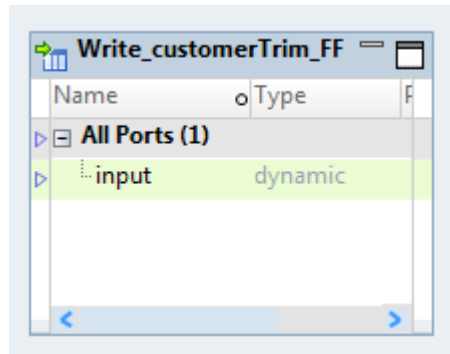
書き込みトランスフォーメーションをマッピングに追加して、Exp\_Output トランスフォーメーションの列に基づいて実行時にターゲットファイルを作成するように Write\_customerTrim\_FF トランスフォーメーションを設定します。

- customerTrim\_FF データオブジェクトを、書き込みトランスフォーメーションとしてマッピングに追加します。
- 書き込みトランスフォーメーションの **【プロパティ】** ビューで、**【ポート】** タブをクリックします。
- 【マッピングフロー】** オプションを選択してターゲットの列を定義します。  
Developer tool で、Write\_customerTrim\_FF トランスフォーメーションに動的ポート **input** が作成されます。

次の図は、オプションを選択した後の【ポート】タブを示しています。



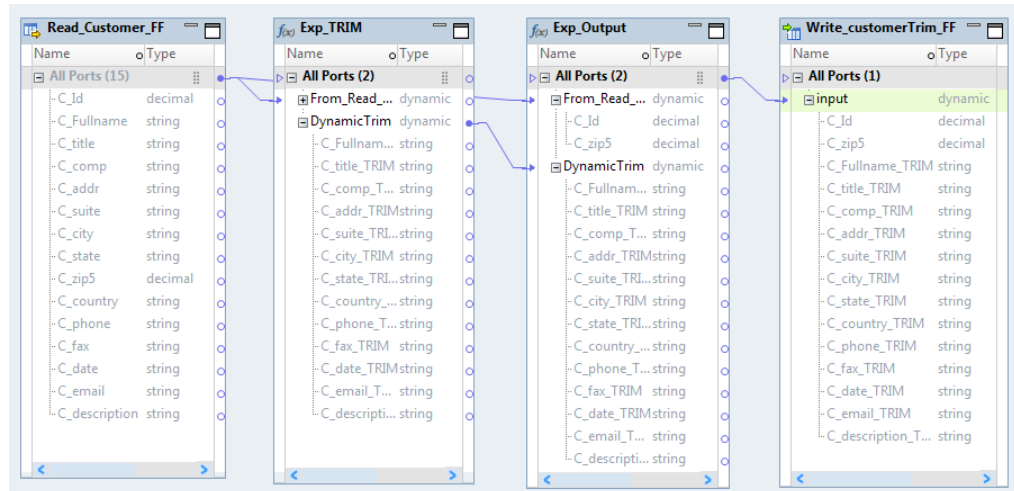
次の画像は、Write\_customerTrim\_FF トランスフォーメーションの新しい動的ポート **input** を示しています。



4. Exp\_Output トランスフォーメーションの【すべてのポート】グループを、Write\_customerTrim\_FF トランスフォーメーションの input ポートにドラッグします。

Developer tool でリンクが作成されて、Exp\_Output トランスフォーメーションの【すべてのポート】グループの各カラムから書き込みトランスフォーメーションの input 動的ポートへのフローが作成されます。

次の図は、書き込みトランスフォーメーションが設定された m\_Replication\_Template マッピングを示しています。



## 手順 5. マッピングの検証と保存

ソースデータオブジェクトとターゲットファイルのデフォルトのパラメータ値を使用して、m\_ReplicationTemplate マッピングを検証および実行し、結果を表示します。

1. マッピングエディタで、**[編集]** > **[検証]** をクリックします。
2. マッピングが有効であれば、**[ファイル]** > **[保存]** をクリックしてマッピングを保存します。

## 手順 6. 異なるソースおよびターゲットに対する動的マッピングの実行

動的マッピングを開発すると、マッピングを実行して異なるソースにアクセスし、パラメータ値に基づいて異なるターゲットに書き込めるようになります。

### Customer\_FF ソースのマッピングの実行

ソースデータオブジェクトとターゲットファイルのデフォルトのパラメータ値を使用して m\_ReplicationTemplate マッピングを実行し、結果を表示します。マッピングは Customer\_FF ソースファイルから読み取り、customerTrim.csv ターゲットファイルに書き込みます。

1. **[実行]** > **[マッピング]** をクリックします。  
**[マッピングの実行]** ウィンドウに、マッピング実行の進行状況が表示されます。マッピングが実行され、出力がターゲットファイルに書き込まれます。
2. ターゲットファイルに書き込まれた結果を確認するには、システム上の Informatica サービスがインストールされているターゲットディレクトリに移動します。

```
<Informatica Installation Directory>\tomcat\bin\target
```

3. customerTrim.csv ファイルを開き、文字列値の先頭と末尾に空白が含まれていないことを確認します。

ファイルの各行には、各カラムのデータがターゲットオブジェクトに現れる順番で表示されます（例：C\_id、C\_zip5、C\_Fullname、C\_title、C\_comp）。例えば、ファイルの先頭の 5 行には、各文字列値の先頭と末尾の空白が削除された次のようなデータが格納されます。

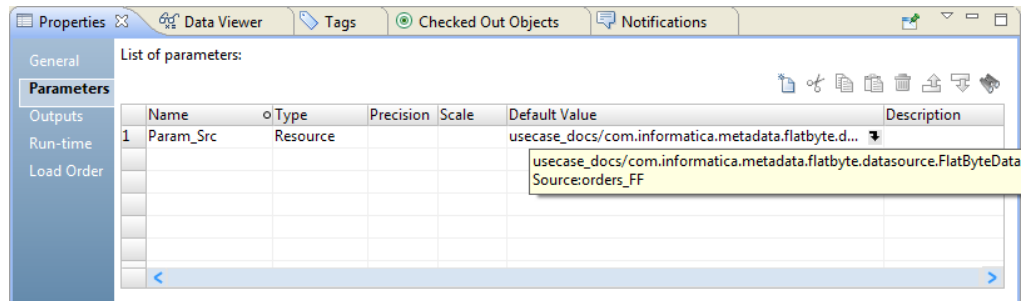
```
1,22342,Smith John,Account Executive,DKR MANAGEMENT COMPANY INC,100 High Street,5406,Anytown,TN,USA,
4047668150,2124031386,31/08/1985,bwilliams@yahoo.com,ACTIVE
2,38103,Balasubramanian Krishna,Account Executive,EASTON & COMPANY,71 Congress Parkway,
789,Bangalore,Karnataka,India,4046345228,4151689756,29/10/1985,bmatthewc@univ.edu,ACTIVE
3,55103,Johnson Lars,Regional Sales Exec,GREATER BAY BANCORP,123 Snow St.,43543,St. Paul,MN,USA,
4046581534,6122945948,7/9/1992,ehpuniv.edu,INACTIVE
4,94116,Zogby Kevin,Regional Sales Exec,HEWLETT-PACKARD,317 29th. St.,5856,San Francisco,CA,USA,
4042662730,4155466814,7/8/1985,grobertwuniv.edu,ACTIVE
5,60606,Franklin Roosevelt,Sales Representative,JAYD TRADING,1511 Wacker Dr,6334,Chicago,IL,USA,
7703965851,2065075486,20/10/1982,trichard@univ.edu,INACTIVE
```

### パラメータ値の変更

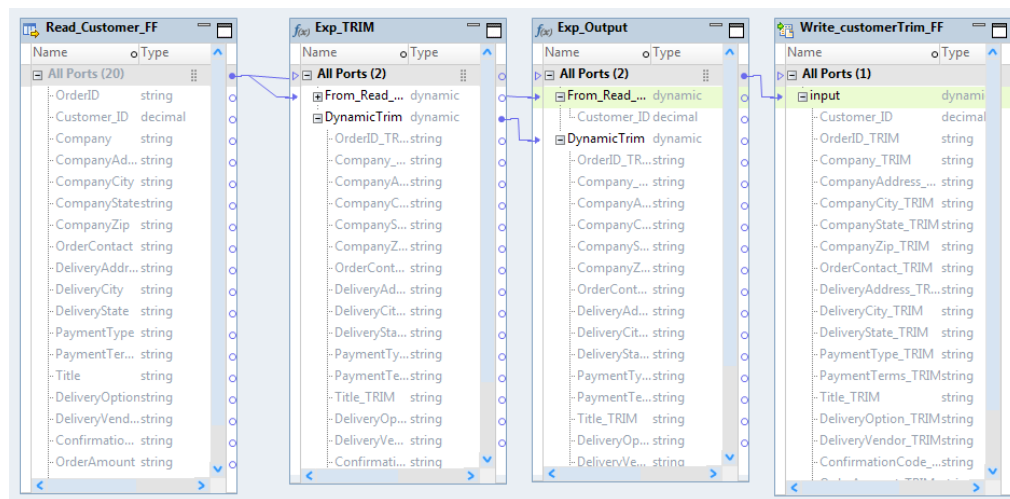
ソースデータオブジェクトのパラメータ値とターゲットデータオブジェクトの出力ファイル名を変更します。

1. ソースデータオブジェクトのパラメータ値を変更するには、以下の手順を実行します。
  - a. マッピングの **[プロパティ]** ビューで、**[パラメータ]** タブをクリックします。
  - b. ソースオブジェクトの Param\_Src パラメータを探します。
  - c. **[デフォルト値]** カラムで、**[開く]** ボタン (🔗) をクリックします。  
**[場所の選択]** ダイアログボックスが表示されます。
  - d. orders\_FF データオブジェクトを選択します。

次の図は、デフォルト値が更新されたマッピングの【パラメータ】タブを示しています。

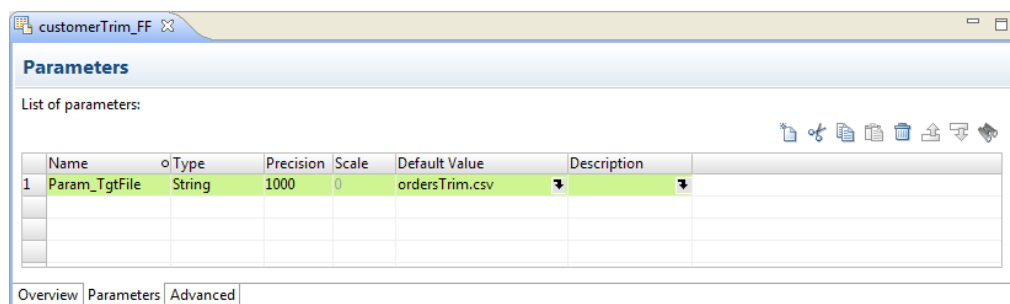


次の図は、読み取りトランスフォーメーションの orders\_FF データオブジェクトからのポートを反映するマッピングを示しています。動的ポートは、新しく生成されたポートを反映しています。



2. ターゲットファイル名のパラメータ値を変更するには、以下の手順を実行します。
  - a. customerTrim\_FF ターゲットデータオブジェクトを開きます。
  - b. データオブジェクトの【パラメータ】タブで、ターゲットファイル名を表す Param\_TgtFile パラメータを探します。
  - c. 【デフォルト値】カラムで、【開く】ボタン (🔍) をクリックします。  
【パラメータ値の編集】ウィンドウが表示されます。
  - d. デフォルトのファイル名の値を ordersTrim.csv に変更し、【OK】をクリックします。

次の図は、デフォルト値が更新された customerTrim\_FF データオブジェクトの【パラメータ】タブを示しています。



## orders\_FF ソースのマッピングの実行

マッピングを検証し、異なるソースおよびターゲットの m\_ReplicationTemplate マッピングを実行します。マッピングは orders\_FF ソースファイルから読み取り、ordersTrim.csv ターゲットファイルに書き込みます。

1. マッピングエディタで、**【編集】** > **【検証】** をクリックします。
2. マッピングが有効であれば、**【ファイル】** > **【保存】** をクリックしてマッピングを保存します。
3. **【実行】** > **【マッピング】** をクリックします。  
**【マッピングの実行】** ウィンドウに、マッピング実行の進行状況が表示されます。マッピングが実行され、出力がターゲットファイルに書き込まれます。
4. ターゲットファイルに書き込まれた結果を確認するには、システム上の Informatica サービスがインストールされているターゲットディレクトリに移動します。  
<Informatica Installation Directory>\tomcat\bin\target
5. ordersTrim.csv ファイルを開き、文字列値の先頭と末尾に空白が含まれていないことを確認します。

ファイルの各行には、各カラムのデータがターゲットオブジェクトに現れる順番で表示されます（例: Customer\_Id、Order\_ID、Company、CompanyAddress、CompanyCity）。例えば、ファイルの先頭の5行には、各文字列値の先頭と末尾の空白が削除された次のようなデータが格納されます。

```
10110085,0-5079,JOSEPH THAL LYON & ROSS,96 FISHER ROAD,MAHWAH,NJ,7430,PARKE PERSLEY OR RAYFORD LECROY,96  
FISHER ROAD,MAHWAH,NJ,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
44162,$21.00,Generic,0/L/B P/W L/S TAWNY SHIMMER .08 OZ.  
10110086,0-6658,NRCA,10255 W.HIGGINS RD.,ROSEMONT,IL,60018-5607,ROLANDA SORTO,10255 W.HIGGINS  
RD.,ROSEMONT,IL,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
44163,$56.40,Generic,0-L.B PW LIPSTYLO LASTING PERFECTION .08 OZ.  
10110087,0-8195,POND EQUITIES,4522 FT. HAMILTON PKWY.,BROOKLYN,NY,11219,KONSTANTIN PEDDICORD,4522 FT.  
HAMILTON PKWY.,BROOKLYN,NY,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
44164,$78.00,Generic,0/L/B P/W L/S TAWNY SHIMMER LASTING PERFECTION LIPSTYLO TAWNY SHIMMER .08 OZ.  
10110088,0-9130,SCHRÖDER & COMPANY,787 SEVENTH AVENUE,NEW YORK,NY,10019,GIORGIA TWITCHELL,787 SEVENTH  
AVENUE,NEW YORK,NY,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
44165,$14.00,Generic,A/COL L PERFECTION L/S REF P SUPREME LASTING PERFECTION LIPSTYLO TAWNY SHIMMER .08  
OZ.  
10110089,0-9352,YUASA TRADING COMPANY (AMERICA),150 EAST 52ND STREET,NEW YORK,NY,10005,STEFFI MCGLOWN,150  
EAST 52ND STREET,NEW YORK,NY,American Express,CHARGE,Account Executive,UPA,United Parcel Service Air,  
44166,$54.00,Generic,0/L/B L PERFECTION REF LIPSTYLO COFFEE PEACH SUPREME .08 OZ.
```

## 第 10 章

# マッピング管理

この章では、以下の項目について説明します。

- [マッピング管理の概要, 208 ページ](#)
- [マッピングジョブのプロパティの表示, 209 ページ](#)
- [マッピングジョブのサマリ統計の表示, 209 ページ](#)
- [マッピングジョブの詳細統計の表示, 209 ページ](#)
- [マッピングジョブのログの表示, 210 ページ](#)
- [デプロイ済みのマッピングジョブの再発行, 210 ページ](#)
- [マッピングジョブのキャンセル, 211 ページ](#)
- [拒否ファイル, 211 ページ](#)

## マッピング管理の概要

アドホックのマッピングジョブを実行する場合、またはデータ統合サービスにマッピングをデプロイする場合は、Monitoring ツールまたは Administrator ツールでジョブを監視できます。データ統合サービスがターゲットに行を書き込むことができない場合、拒否ファイルで行に関する情報を確認することもできます。ジョブを監視したり拒否ファイルを表示したりするには、適切な特権が必要です。

マッピングジョブは、以下の場所で監視することができます。

- Monitoring ツール。Developer tool で、**[進行状況]** ビューの **[メニュー]** ボタンをクリックし、**[ジョブの監視]** を選択します。マッピングジョブを実行するデータ統合サービスを選択して、**[OK]** をクリックします。Monitoring ツールが開きます。
- Administrator ツール。Administrator ツールでマッピングを監視するには、**[モニタ]** タブをクリックします。

マッピングジョブを監視する場合、ジョブのサマリ統計または実行統計を表示できます。**[サマリ統計]** ビューには、ドメイン内のマッピングジョブのステータスの概要がグラフィック表示されます。マッピングプロパティと統計の表示、ジョブログの表示、ジョブのキャンセル、デプロイ済みのマッピングの再発行を行うには、**[実行統計]** ビューを使用します。



## マッピングジョブのプロパティの表示

アドホックまたはデプロイ済みのマッピングジョブを監視する場合、ジョブのプロパティを表示できます。プロパティには、ジョブ ID、ジョブの開始ユーザー、ジョブの継続時間が含まれます。

1. **【実行統計】** ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. **【アドホックジョブ】** フォルダを選択するか、アプリケーションを展開して **【デプロイ済みのマッピングジョブ】** を選択します。

ジョブのリストが **【コンテンツ】** パネルに表示されます。**【コンテンツ】** パネルには、ジョブの名前、状態、ID、継続時間など、各種プロパティが表示されます。

4. **【コンテンツ】** パネルでジョブを選択します。  
**【詳細】** パネルにジョブのプロパティが表示されます。

## マッピングジョブのサマリ統計の表示

アドホックまたはデプロイ済みのマッピングジョブのスループットおよびリソース使用状況の統計を表示できます。

1. **【実行統計】** ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. **【アドホックジョブ】** フォルダを選択するか、アプリケーションを展開して **【デプロイ済みのマッピングジョブ】** を選択します。

ジョブのリストが **【コンテンツ】** パネルに表示されます。

4. **【コンテンツ】** パネルでジョブを選択します。  
**【詳細】** パネルにジョブのプロパティが表示されます。
5. **【詳細】** パネルで **【サマリ統計】** ビューをクリックします。

**【サマリ統計】** ビューには、ソースおよびターゲットのスループットおよびリソース使用状況の統計が表示されます。

必要に応じて、昇順または降順で統計をソートすることができます。列ヘッダーをクリックして昇順に列をソートします。再度列ヘッダーをクリックして降順に列をソートします。

## マッピングジョブの詳細統計の表示

個別のローカルプロセスで実行されている、アドホックまたはデプロイ済みのマッピングジョブのスループットおよびリソース使用状況のグラフを表示できます。1分を超えて実行されているジョブの詳細な統計が表示されます。

1. **【実行統計】** ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. **【アドホックジョブ】** フォルダを選択するか、アプリケーションを展開して **【デプロイ済みのマッピングジョブ】** を選択します。

ジョブのリストが [コンテンツ] パネルに表示されます。

4. [コンテンツ] パネルでジョブを選択します。

[詳細] パネルにジョブのプロパティが表示されます。

5. [詳細] パネルで [詳細統計] ビューをクリックします。

[詳細統計] ビューに、スループットのグラフとリソース使用状況のグラフが表示されます。

必要に応じて、[詳細統計] ビューで次のタスクを実行することもできます。

タスク	説明
グラフの拡大	グラフ上にカーソルを移動し、拡大鏡アイコンをクリックします。
拡大されたグラフの一部を拡大	カーソルをドラッグして拡大する領域を選択します。
スループットグラフの行とバイトの切り替え	[バイト] オプションまたは [行] オプションをクリックします。
スループットグラフにプロットする統計の選択	[スループット] フィールドで、表示するソースとターゲットを選択します。

## マッピングジョブのログの表示

ジョブ詳細を表示するジョブのログをダウンロードすることができます。

1. [実行統計] ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. [アドホックジョブ] フォルダを選択するか、アプリケーションを展開して [デプロイ済みのマッピングジョブ] を選択します。

ジョブのリストが [コンテンツ] パネルに表示されます。

4. [コンテンツ] パネルでジョブを選択します。

5. [アクション] > [選択したオブジェクトのログの表示] をクリックします。

ログファイルを開く、または保存するオプションのついたダイアログボックスが表示されます。

## デプロイ済みのマッピングジョブの再発行

マッピングジョブが失敗した場合は、デプロイ済みのマッピングジョブの再発行ができます。デプロイ済みのマッピングジョブを再発行する場合は、データ統合サービスでジョブが再度実行されます。

1. [実行統計] ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. アプリケーションを展開して、[デプロイ済みのマッピングジョブ] を選択します。  
[コンテンツ] パネルにデプロイ済みのマッピングジョブのリストが表示されます。
4. デプロイ済みのマッピングジョブを選択します。

5. **【アクション】** > **【選択したジョブの再発行】** をクリックします。

## マッピングジョブのキャンセル

アドホックまたはデプロイ済みのマッピングジョブを実行中にキャンセルできます。例えば、応答を停止したジョブ、または完了までの所要時間が極端に長いジョブをキャンセルすることができます。

1. **【実行統計】** ビューをクリックします。
2. ドメインナビゲータで、データ統合サービスを展開します。
3. **【アドホックジョブ】** フォルダを選択するか、アプリケーションを展開して **【デプロイ済みのマッピングジョブ】** を選択します。  
ジョブのリストが **【コンテンツ】** パネルに表示されます。
4. **【コンテンツ】** パネルでジョブを選択します。
5. **【アクション】** > **【選択したジョブのキャンセル】** をクリックします。

## 拒否ファイル

マッピングの実行中、データ統合サービスはマッピングの中の各ターゲットインスタンスに対して拒否ファイルを作成します。データ統合サービスがターゲットに行を書き込むことができない場合、データ統合サービスは拒否された行を拒否ファイルに書き込みます。拒否ファイルおよびマッピングログには、その拒否の原因の判断に役立つ情報が含まれています。

拒否ファイルに拒否された行が含まれていない場合、データ統合サービスはマッピングの実行が終了したときに拒否ファイルを削除します。

マッピングを実行するたびに、データ統合サービスは拒否されたデータを拒否ファイルに追加します。問題の原因によっては、マッピングおよびターゲットデータベースを修正することで、それ以降のマッピングでの拒否の発生を防ぐことができます。

### 拒否ファイルの場所

データ統合サービスは、マッピング内のターゲットインスタンスごとに拒否ファイルを作成します。拒否ファイルはターゲットの拒否ファイルディレクトリに作成されます。

マッピング内のフラットファイルまたはリレーショナルなターゲットに関するランタイムプロパティで、ターゲットの拒否ファイルのディレクトリを設定します。デフォルトで、データ統合サービスは、RejectDir システムパラメータで定義されたディレクトリに、拒否ファイルを作成します。データ統合サービスは、拒否ファイルのファイル名をターゲットのインスタンス名に合わせて設定します。拒否ファイルのデフォルトのファイル名は<file\_name>.bad です。

データ統合サービスが1つのターゲットに複数のパーティションを作成する場合、データ統合サービスはパーティションごとに別々の拒否ファイルを作成し、そのファイル名は<file\_name><partition\_number>.bad という形式になります。例えば、3つのパーティションが、MyOutput1.bad、MyOutput2.bad、MyOutput3.bad という名前の拒否ファイルに書き込みを行うことがあります。

## 拒否ファイルの内容

拒否ファイルが見つかったら、拒否ファイルの使用するコードページを表示できるテキストエディタでファイルを読むことができます。

拒否ファイルには、ライタまたはターゲットデータベースに拒否されたデータの行が入っています。データ統合サービスの書き込みは、拒否ファイル内の行全体に対して行われます。ただし、問題は通常その行の 1 カラムに集中します。拒否された行のどのカラムに原因があったかの判定に役立つように、拒否ファイルには各カラムについての詳しい情報を提供するインジケータが含まれています。

拒否ファイルに含まれるインジケータは以下のとおりです。

### 行インジケータ

拒否ファイルの各行の最初のカラムは、行インジケータです。行インジケータは、挿入、更新、または拒否のいずれのマークが行に付いていたかを示します。

### カラムインジケータ

カラムインジケータは、カラムごとにデータの後に表示されます。カラムインジケータは、カラムに含まれているデータが有効か、オーバフローしているか、NULL か、または切り詰められているかを示します。

## 行インジケータ

リジェクトファイルの最初のカラムは行インジケータです。行インジケータは、データ行の更新戦略を定義するフラグです。

以下の表に、拒否ファイルの行インジケータを示します。

行インジケータ	意味	拒否元
0	挿入	writer またはターゲット
1	更新	writer またはターゲット
2	削除	writer またはターゲット
3	拒否。アップデートストラテジの式によって拒否のマークが付いている。	Writer
4	ロールバックされた挿入	Writer
5	ロールバックされた更新	Writer
6	ロールバックされた削除	Writer
7	コミットされた挿入	Writer
8	コミットされた更新	Writer
9	コミットされた削除	Writer

次に示す拒否ファイルのサンプルでは、各行で行インジケータが 0 になっています。これはその行の挿入アップデートストラテジを表しています。

```
0,D,1921,D,Nelson,D,William,D,415-541-5145,D
0,D,1922,D,Page,D,Ian,D,415-541-5145,D
0,D,1923,D,Osborne,D,Lyle,D,415-541-5145,D
```

0,D,1928,D,De Souza,D,Leo,D,415-541-5145,D  
0,D,2001123456789,0,S. MacDonald,D,Ira,D,415-541-514566,T

## カラムインジケータ

カラムインジケータは、各データカラムのあとに表示されます。カラムインジケータは、データが有効であったか、オーバーフローしていたか、NULL であったか、または切り詰められていたかを示します。

以下の表に、拒否ファイルのカラムインジケータを示します。

カラムインジケータ	データの種類	Writer による扱い
D	有効なデータ。	正しいデータ。writer はデータをターゲットデータベースに渡します。ターゲットは、重複キーなどのデータベースエラーが起こらない限りデータを受け入れます。
N	NULL。カラムに NULL が格納されている。	正しいデータ。writer はデータをターゲットに渡し、ターゲットデータベースが NULL 値を受け入れない場合はターゲットがデータをリジェクトします。
T	切り詰め。文字列データが、カラムに指定されている精度の範囲を超えているため、値は切り詰められました。	マッピングターゲットがオーバーフローまたは切り詰めをリジェクトするように設定している場合は不良データ。

拒否ファイル内では、NULL カラムはカンマでマークして示されます。次の例に、良好なデータに囲まれた NULL カラムを示します。

0,D,5,D,,N,5,D

各行インジケータの後にカラムインジケータ「D」も表示されます。次の例では、行インジケータ「0」の後にカラムインジケータ「D」が表示されています。

0,D,2001123456789,0,S. MacDonald,D,Ira,D,415-541-514566,T

行は、writer またはターゲットデータベースによって拒否されることがあります。ログを調べて、拒否された原因を判定します。

## 第 11 章

# PowerCenter へのエクスポート

この章では、以下の項目について説明します。

- [PowerCenter へのエクスポートの概要, 214 ページ](#)
- [PowerCenter のリリースの互換性, 215 ページ](#)
- [マプレットのエクスポート, 215 ページ](#)
- [パラメータを含むマッピングのエクスポート, 216 ページ](#)
- [PowerCenter へのエクスポートのオプション, 217 ページ](#)
- [PowerCenter へのオブジェクトのエクスポート, 218 ページ](#)
- [エクスポートの制限, 219 ページ](#)
- [PowerCenter へのエクスポートに関するルールおよびガイドライン, 221 ページ](#)
- [PowerCenter へのエクスポートのトラブルシューティング, 222 ページ](#)

## PowerCenter へのエクスポートの概要

オブジェクトを Developer tool からエクスポートして、PowerCenter<sup>®</sup>で使用できます。

以下のオブジェクトをエクスポートできます。

- マッピング。マッピングを PowerCenter マッピングまたはマプレットにエクスポートします。
- マプレット。マプレットを PowerCenter マプレットにエクスポートします。
- 論理データオブジェクトモデル。論理データオブジェクトモデルを PowerCenter マプレットにエクスポートします。

オブジェクトは PowerCenter リポジトリまたは XML ファイルにエクスポートします。オブジェクトを XML ファイルにエクスポートした場合、PowerCenter ユーザーはそのファイルを PowerCenter リポジトリにインポートできます。

オブジェクトをエクスポートするときに、PowerCenter のリリース、マッピングとマプレットを変換する方法、参照テーブルをエクスポートするかどうかなどのエクスポートオプションを指定します。

パラメータを含むマッピングおよびマプレットをエクスポートできます。パラメータは、マッピングを PowerCenter リポジトリにインポートするときにデフォルト値に解決されます。

メタデータを再利用するには、Developer tool または必要なエクスポートコマンドを使用して PowerCenter にエクスポートします。

モデルリポジトリから PowerCenter リポジトリにデータをエクスポートするには、次のタスクを実行します。

1. Developer tool または次のコマンドを使用して、モデルリポジトリオブジェクトをファイルにエクスポートします。  
`infacmd tools ExportObjects`  
または、`infacmd ipc ExportToPC` を直接実行してエクスポートすることもできます。
2. 次のコマンドを使用して、エクスポートファイルを PowerCenter ファイルに変換します。  
`infacmd ipc ExporttoPC`
3. PowerCenter または次のコマンドを使用して、オブジェクトをインポートします。  
`pmrep importObjects`

## PowerCenter のリリースの互換性

オブジェクトが特定の PowerCenter のリリースとの互換性を持つことを検証するには、PowerCenter のリリースの互換性レベルを設定します。互換性レベルは、Developer ツールで表示できるすべてのマッピング、マプレット、および論理データオブジェクトモデルに適用されます。

特定のリリースの PowerCenter と照らし合わせて検証を行うように Developer ツールを設定するか、リリースの互換性の検証をスキップするように Developer ツールを設定することができます。デフォルトでは、Developer ツールは PowerCenter のリリースと照らし合わせたオブジェクトの検証を行いません。

オブジェクトを PowerCenter にエクスポートする前に、互換性レベルを PowerCenter のリリースに設定します。互換性レベルを設定すると、マッピング、マプレット、または論理データオブジェクトモデルを検証するときに 2 つの検証チェックが実行されます。まず、オブジェクトが Developer ツールで有効であることが検証されます。オブジェクトが有効な場合は、オブジェクトが選択したリリースの PowerCenter へのエクスポートに有効であることが検証されます。【検証ログ】ビューで互換性エラーを表示できます。

### 互換性レベルの設定

互換性レベルを設定して、PowerCenter のリリースと照らし合わせてマッピング、マプレット、および論理データオブジェクトモデルを検証します。【なし】を選択した場合は、オブジェクトの検証時にリリースの互換性の検証がスキップされます。

1. 【編集】 > 【互換性レベル】をクリックします。
2. 互換性レベルを選択します。

メニューの選択した互換性レベルの横にドットが表示されます。互換性レベルは、Developer ツールで表示できるすべてのマッピング、マプレット、および論理データオブジェクトモデルに適用されます。

## マプレットのエクスポート

マプレットをエクスポートするとき、またはマッピングをマプレットとしてエクスポートするときに、エクスポートプロセスによってマプレット内のオブジェクトが作成されます。また、いくつかのマプレットオブジェクトの名前が変更されます。

エクスポートプロセスでは、以下のマプレットオブジェクトがエクスポート XML ファイルに作成されます。

### 式トランスフォーメーション

エクスポートプロセスによって、マプレットの各入力トランスフォーメーションのすぐ下と各出力トランスフォーメーションのすぐ上に式トランスフォーメーションが作成されます。式トランスフォーメーションには次のような名前が付けられます。

Expr\_<InputOrOutputTransformationName>

式トランスフォーメーションには、パススルーポートが含まれます。

### 出力トランスフォーメーション

マプレットをエクスポートしてターゲットを出力トランスフォーメーションに変換する場合は、ターゲットごとに出力トランスフォーメーションが作成されます。出力トランスフォーメーションには次のような名前が付けられます。

<MappletInstanceName>\_<TargetName>

エクスポートプロセスでは、以下のマプレットオブジェクトの名前がエクスポート XML ファイルで変更されません。

### マプレットの入力トランスフォーメーションおよび出力トランスフォーメーション

マプレットの入力トランスフォーメーションおよび出力トランスフォーメーションには次のような名前が付けられます。

<TransformationName>\_<InputOrOutputGroupName>

### マプレットポート

マプレットポートの名前は次のように変更されます。

<PortName>\_<GroupName>

## パラメータを含むマッピングのエクスポート

パラメータを含むマッピングやマプレットをエクスポートして、PowerCenter にインポートできます。

パラメータを含むマッピングやマプレットをエクスポートする場合、パラメータは PowerCenter にインポートするときにデフォルト値に解決されます。インポートでは、パラメータを含むすべての SQL 式を解決できません。

システムパラメータは、対応する PowerCenter システムパラメータに解決されます。PowerCenter に対応するシステムパラメータがない場合、システムパラメータ参照は、PowerCenter にインポートした後もマッピング内に残ります。マッピングを編集し、参照を変更する必要があります。

マッピング出力を PowerCenter にエクスポートすることはできません。マッピングにマッピング出力が含まれている場合、マッピングをインポートしても、PowerCenter では無効になります。



# PowerCenter へのエクスポートのオプション

オブジェクトを PowerCenter で使用するためにエクスポートするときは、エクスポートオプションを指定する必要があります。

以下の表に、エクスポートのオプションを示します。

オプション	説明
プロジェクト	オブジェクトのエクスポート元のモデルリポジトリ内のプロジェクト。
ターゲットリリース	PowerCenter のリリースバージョンです。
選択したオブジェクトをファイルにエクスポート	PowerCenter XML ファイルにオブジェクトをエクスポートします。このオプションを選択する場合は、エクスポート XML ファイルの名前と場所を指定します。
選択したオブジェクトを PowerCenter リポジトリにエクスポート	PowerCenter リポジトリにオブジェクトをエクスポートします。このオプションを選択する場合は、PowerCenter リポジトリに関する以下の接続の詳細を指定します。 <ul style="list-style-type: none"><li>- ホスト名。PowerCenter ドメインゲートウェイのホスト名です。</li><li>- ポート番号。PowerCenter ドメインゲートウェイの HTTP ポート番号。</li><li>- 認証タイプ。次のうちいずれかの値を選択します: Kerberos シングルサインオン、ネイティブ、または LDAP。</li><li>- ユーザー名。リポジトリユーザー名です。</li><li>- パスワード。リポジトリユーザー名のパスワード。</li></ul> <p>注: 認証タイプがネイティブまたは LDAP の場合は、ユーザー名とパスワードを指定します。</p> <ul style="list-style-type: none"><li>- セキュリティドメイン。認証タイプが LDAP の場合は、LDAP セキュリティドメイン名を指定します。存在しない場合は、「Native」と入力します。</li><li>- リポジトリ名。PowerCenter リポジトリ名です。</li></ul>
リポジトリフォルダーに送る	PowerCenter リポジトリ内の指定したフォルダーにオブジェクトをエクスポートします。
制御ファイルを使用	指定した <i>pmrep</i> 制御ファイルを使用して PowerCenter リポジトリにオブジェクトをエクスポートします。
エクスポートしたマッピングを PowerCenter マップレットに変換	Developer tool のマッピングを PowerCenter マップレットに変換します。Developer tool により、マッピングのソースおよびターゲットとして使用されるデータオブジェクトが PowerCenter マップレットの入力トランスフォーメーションおよび出力トランスフォーメーションに変換されます。
ターゲットマップレットの変換	マップレットのターゲットとして使用されるデータオブジェクトを PowerCenter マップレットの出力トランスフォーメーションに変換します。PowerCenter マップレットにターゲットを含めることはできません。エクスポートオブジェクトにターゲットを含むマップレットが含まれる場合にこのオプションを選択しないと、エクスポートプロセスは失敗します。
参照データのエクスポート	エクスポートするオブジェクトのトランスフォーメーションで使用する参照テーブルデータをエクスポートします。

オプション	説明
参照データの場所	Developer tool がエクスポートする参照テーブルデータの場所。参照テーブルデータは 1 つ以上のディクショナリファイルとしてエクスポートされます。Developer tool をホストするマシンで、ディレクトリへのパスを入力します。
コードページ	PowerCenter リポジトリのコードページ。

## PowerCenter へのオブジェクトのエクスポート

PowerCenter にマッピング、マップレット、または論理データオブジェクトモデルをエクスポートするときに、オブジェクトをファイルまたは PowerCenter リポジトリにエクスポートできます。

オブジェクトをエクスポートする前に、互換性レベルを適切な PowerCenter のリリースに設定します。オブジェクトがその PowerCenter のリリースとの互換性を持つことを検証します。

1. **【ファイル】** > **【エクスポート】** をクリックします。  
**【エクスポート】** ダイアログボックスが表示されます。
2. **【Informatica】** > **【PowerCenter】** を選択します。
3. **【次へ】** をクリックします。  
**【エクスポート先 PowerCenter】** ダイアログボックスが表示されます。
4. オブジェクトのエクスポート元のモデルリポジトリ内のプロジェクトを選択します。
5. オブジェクトのエクスポート先の PowerCenter リリースを選択します。
6. オブジェクトのエクスポート先を選択します。オブジェクトは、PowerCenter リポジトリの XML ファイルにエクスポートできます。
  - オブジェクトをファイルにエクスポートするには、XML ファイルの名前と場所を指定します。
  - オブジェクトを PowerCenter リポジトリにエクスポートするには、**【参照】** をクリックしてリポジトリへの接続の詳細を指定します。
7. PowerCenter リポジトリにエクスポートする場合は、PowerCenter リポジトリ内のフォルダーか、PowerCenter へのオブジェクトのインポート方法を定義する *pmrep* 制御ファイルを選択します。
8. **【エクスポートしたマッピングを PowerCenter マップレットに変換】** を選択し、PowerCenter で Developer ツールのマッピングをマップレットに変換します。
9. **【ターゲットマップレットの変換】** を選択し、マップレットでターゲットとして使用されるデータオブジェクトを PowerCenter マップレットの出力トランスフォーメーションに変換します。
10. **【参照データのエクスポート】** を選択し、エクスポートするオブジェクトのトランスフォーメーションで使用する参照テーブルデータをエクスポートします。
11. 参照データをエクスポートする場合は、Developer ツールがエクスポートする参照テーブルデータの場所を指定します。
12. PowerCenter リポジトリのコードページを選択します。
13. **【次へ】** をクリックします。  
エクスポートするオブジェクトを選択するように求められます。
14. エクスポートするオブジェクトを選択して、**【完了】** をクリックします。  
オブジェクトが選択した場所にエクスポートされます。

オブジェクトをファイルにエクスポートする場合は、そのファイルから PowerCenter リポジトリにオブジェクトをインポートできます。

参照テーブルデータをエクスポートする場合は、Informatica サービスをホストしているマシンの PowerCenter のディレクトリ構造に参照データファイルをコピーします。参照データファイルの場所は、モデルリポジトリ内の参照テーブルオブジェクトの場所に対応している必要があります。

例えば、参照データファイルを以下の場所にコピーします。

<PowerCenter インストールディレクトリ>\services\<モデルリポジトリプロジェクト名>\<フォルダー名>

## エクスポートの制限

モデルリポジトリオブジェクトを PowerCenter にエクスポートすると、一部のモデルリポジトリオブジェクトが PowerCenter リポジトリにエクスポートされない場合があります。PowerCenter で有効でない任意のオブジェクトを含むマッピングやマップレットをエクスポートすることはできません。

以下のオブジェクトは PowerCenter にエクスポートできません。

### 名前が長いオブジェクト

オブジェクト名が 80 文字を超える場合、PowerCenter ユーザーはマッピング、マップレット、またはマッピングかマップレット内のオブジェクトをインポートできません。

### 動的ポートが含まれているマッピングまたはマップレット

動的ポートが含まれているマッピングまたはマップレットはエクスポートできません。

### システムパラメータを使用しているディシジョントランスフォーメーションを含むマッピングまたはマップレット

ディシジョントランスフォーメーションのトランスフォーメーションスクリプトにシステムパラメータが含まれている場合、そのトランスフォーメーションを含むマッピングまたはマップレットをエクスポートすることはできません。エクスポート操作では、システムパラメータを PowerCenter が使用できる値に変換することはできません。システムパラメータを使用しているディシジョントランスフォーメーションを含むマッピングまたはマップレットをエクスポートするには、事前にそれらのシステムパラメータを適切な値で置換してください。

### マッピング出力を返すマッピングまたはマップレット

マッピングまたはマップレットでマッピング出力が返される場合、PowerCenter ユーザーはマッピングまたはマップレットをインポートできません。

### 特定の結合条件を持つジョイナトランスフォーメーションが含まれているマッピングまたはマップレット

PowerCenter で有効でない結合条件を持つジョイナトランスフォーメーションが含まれているマッピングおよびマップレットはエクスポートできません。PowerCenter では、ユーザーはマスタソースと明細ソースが等しいことに基づいて結合条件を定義します。Developer tool では、その他の結合条件を定義できます。例えば、マスタソースと明細ソースが等しいことまたは等しくないことに基づいて結合条件を定義できます。トランスフォーメーション式が含まれている結合条件を定義できます。また、ジョイナトランスフォーメーションで交差結合が実行されるようにする  $1=1$  などの結合条件も定義できます。

これらのタイプの結合条件は、PowerCenter では無効です。したがって、これらのタイプの結合条件を持つジョイナトランスフォーメーションが含まれているマッピングまたはマップレットを PowerCenter にエクスポートすることはできません。

#### **ポート名が変更されたルックアップトランスフォーメーションが含まれているマッピングまたはマップレット**

PowerCenter 統合サービスは、トランスフォーメーションのルックアップポートおよびルックアップ条件に基づいて、ルックアップソースに対してクエリを実行します。したがって、ルックアップトランスフォーメーションのポート名はルックアップソースのカラム名と一致している必要があります。

#### **特定のカスタム SQL クエリを使用するルックアップトランスフォーメーションが含まれているマッピングまたはマップレット**

Developer tool では、ルックアップトランスフォーメーション内の SQL クエリ構文を検証するために、PowerCenter とは異なるルールを使用します。Developer tool で作成されるカスタム SQL クエリで AS キーワードまたは計算済みフィールドを使用する場合、そのクエリは PowerCenter で有効ではありません。SQL クエリを使用するルックアップトランスフォーメーションがマッピングまたはマップレットに含まれていて、そのクエリで AS キーワードまたは計算済みフィールドが使用されている場合、そのマッピングまたはマップレットを PowerCenter にエクスポートすることはできません。

#### **PowerCenter で使用できないソースが含まれているマッピングまたはマップレット**

PowerCenter で使用できないソースが含まれているマッピングまたはマップレットをエクスポートする場合、マッピングまたはマップレットのエクスポートは失敗します。

以下のソースを持つマッピングまたはマップレットはエクスポートできません。

- 複合型ファイルデータオブジェクト
- DataSift
- Web コンテンツ-Kapow Katalyst

#### **データウェアハウスのソースまたはターゲットを含むマッピングまたはマップレット**

Greenplum、Netezza、または Teradata PT オブジェクトを含むマッピングまたはマップレットは PowerCenter にエクスポートできません。

#### **ポートを連結するマップレット**

複数グループ入力トランスフォーメーションが含まれているマップレットをエクスポートする場合に、異なる入力グループのポートが同じダウストリームトランスフォーメーションに接続されていると、エクスポートプロセスは失敗します。

#### **未接続のルックアップトランスフォーメーションを持つネストされたマップレット**

未接続のルックアップトランスフォーメーションを持つ別のマップレットが含まれているあらゆるタイプのマッピングまたはマップレットをエクスポートする場合、エクスポートプロセスは失敗します。

#### **SAP ソースが含まれているマッピング**

SAP ソースが含まれているマッピングをエクスポートすると、SAP ソースなしでマッピングがエクスポートされます。このマッピングを PowerCenter リポジトリにインポートすると、PowerCenter クライアントによってマッピングがソースなしでインポートされます。出力ウィンドウに、マッピングが無効であることを伝えるメッセージが表示されます。PowerCenter で SAP ソースを手動で作成してマッピングに追加する必要があります。

#### **Timestamp with Time Zone または Timestamp with Local Time Zone を含むマッピング**

Timestamp with Time Zone または Timestamp with Local Time Zone データ型のデータを含むマッピングを Developer tool からインポートすると、PowerCenter Client がマッピングの変換に失敗します。

# PowerCenter へのエクスポートに関するルールおよびガイドライン

Developer ツールと PowerCenter の違いにより、Developer ツールのオブジェクトの中には PowerCenter との互換性がないものもあります。

PowerCenter にオブジェクトをエクスポートする場合には、以下のルールおよびガイドラインに従います。

**PowerCenter のリリースを確認する。**

Developer ツールからエクスポートするオブジェクトがターゲットの PowerCenter リリースと互換性があることを確認します。

**オブジェクト名が一意であることを確認する。**

オブジェクトを PowerCenter リポジトリにエクスポートする場合、PowerCenter オブジェクトの名前がエクスポートされるオブジェクトと同じときは、PowerCenter オブジェクトが上書きされます。

**コードページに互換性があることを確認する。**

Developer ツールと PowerCenter で互換性のないコードページが使用されている場合、エクスポートプロセスは失敗します。

**精度モードを確認する。**

デフォルトでは、Developer ツールは高精度が有効になっている状態でマッピングおよびマプレットを実行しますが、PowerCenter は高精度が無効になっている状態でセッションを実行します。Developer ツールのマッピングと PowerCenter のセッションを異なる精度モードで実行すると、異なる結果になる場合があります。異なる結果にならないようにするには、オブジェクトを同じ精度モードで実行します。

**論理データオブジェクトに関連付けられているマッピングタイプを検証します。**

論理データオブジェクトの読み取りマッピングと、論理データオブジェクトの書き込みマッピングを PowerCenter にエクスポートすると、読み取りマッピングは Designer のマプレットの下に表示されます。エクスポートプロセスは、論理データオブジェクトに関連付けられている書き込みマッピングを無視します。

論理データオブジェクトの書き込みマッピングを PowerCenter にエクスポートすると、エクスポートプロセスは失敗します。

**参照データをコピーする。**

参照テーブルを使用するトランスフォーメーションとともにマッピングまたはマプレットをエクスポートする場合、その参照テーブルを、PowerCenter 統合サービスがアクセスできるディレクトリにコピーする必要があります。参照テーブルは、INFA\_CONTENT 環境変数で定義されたディレクトリにコピーします。INFA\_CONTENT が設定されていない場合は、参照テーブルを次の PowerCenter サービスディレクトリにコピーします。

`$INFA_HOME\services\<Developer ツールのプロジェクト名>\<Developer ツールのフォルダー名>`

# PowerCenter へのエクスポートのトラブルシューティング

名前が長いオブジェクトが含まれているマプレットをエクスポートするときにエクスポートプロセスが失敗する。

マプレットをエクスポートするとき、またはマッピングをマプレットとしてエクスポートするときに、エクスポートプロセスによってマプレット内のいくつかのオブジェクトが作成または名前変更されます。エクスポートプロセスでは、式トランスフォーメーションまたは出力トランスフォーメーションがエクスポート XML ファイルに作成されます。また、入力トランスフォーメーション、出力トランスフォーメーション、およびマプレットポートの名前が変更されます。

式トランスフォーメーションの名前を生成するために、入力トランスフォーメーションおよび出力トランスフォーメーションの名前に文字が付加されます。マプレットをエクスポートしてターゲットを出力トランスフォーメーションに変換する場合は、マプレットのインスタンス名とターゲット名を組み合わせることで出力トランスフォーメーション名が生成されます。入力トランスフォーメーション、出力トランスフォーメーション、およびマプレットポートの名前が変更されるとき、オブジェクト名にグループ名が付加されます。

既存のオブジェクトの名前が長い場合は、エクスポートされるオブジェクトの名前が、エクスポート XML ファイルまたは PowerCenter リポジトリにおけるオブジェクト名の制限である 80 文字を超えることがあります。オブジェクト名が 80 文字を超えると、内部エラーが発生してエクスポートプロセスが失敗します。

マプレットをエクスポートする場合に内部エラーが返されたときは、入力トランスフォーメーション、出力トランスフォーメーション、ターゲット、およびポートの名前を確認します。名前が長い場合は短くします。

## 第 12 章

# PowerCenter からのインポート

この章では、以下の項目について説明します。

- [PowerCenter からのインポートの概要, 223 ページ](#)
- [オーバーライドプロパティ, 224 ページ](#)
- [競合の解決, 226 ページ](#)
- [インポートサマリ, 227 ページ](#)
- [データ型の変換, 227 ページ](#)
- [トランスフォーメーションの変換, 228 ページ](#)
- [パラメータの変換, 235 ページ](#)
- [PowerCenter リポジトリの接続プロパティ, 237 ページ](#)
- [接続の割り当て, 238 ページ](#)
- [PowerCenter からのオブジェクトのインポート, 238 ページ](#)
- [インポートの制限, 240 ページ](#)
- [インポートパフォーマンス, 243 ページ](#)

## PowerCenter からのインポートの概要

PowerCenter リポジトリからモデルリポジトリにオブジェクトをインポートすることができます。インポートプロセスでは、PowerCenter リポジトリオブジェクトが検証されてモデルリポジトリオブジェクトに変換された後、インポートが実行されます。

PowerCenter からオブジェクトをインポートする前に、コマンドラインで `infacmd ipc genReuseReportfromPC` コマンドを実行して、ネイティブ環境および Hadoop 環境のモデルリポジトリで再利用できる PowerCenter マッピングの数を推定します。

PowerCenter からオブジェクトをインポートする場合は、インポートするオブジェクトおよびモデルリポジトリ内のターゲット場所を選択します。インポートプロセスには、インポート中にオブジェクト名の競合を解決するためのオプションがあります。

また、モデルリポジトリの接続を PowerCenter のオブジェクトに割り当てることもできます。複数の PowerCenter オブジェクトに同時に単一の接続を割り当てることができます。

パラメータを含むマッピングをインポートできます。再利用可能なトランスフォーメーションを含むマッピングをインポートする場合、インポートプロセスで PowerCenter マッピングパラメータがインポートされ、そのパラメータのバインド先となるトランスフォーメーションレベルのパラメータが生成されます。マッピングに再利用不可のトランスフォーメーションが含まれている場合、入力プロセスでマッピングレベルのパラメータが作成されます。

PowerCenter からマッピングをインポートすると、メタデータを再利用できます。Developer tool または必要なインポートコマンドを使用して PowerCenter からインポートします。

PowerCenter からモデルリポジトリにデータをインポートするには、次のタスクを実行します。

1. PowerCenter Client または次のコマンドを使用して、PowerCenter オブジェクトをファイルにエクスポートします。  
`pmrep ExportObject`
2. 次のコマンドを使用して、エクスポートファイルをモデルリポジトリファイルに変換します。  
`infacmd ipc importFromPC`
3. Developer tool または次のコマンドを使用して、オブジェクトをインポートします。  
`infacmd tools importObjects`

パイプライン、セッション、ワークフロー、ワークレットが複数あるマッピングを PowerCenter からモデルリポジトリにインポートできます。ワークフロー内のセッションは、モデルリポジトリにマッピングタスクとしてインポートされます。ワークフローは、モデルリポジトリ内にワークフローとしてインポートされます。ワークフロー内のワークレットは展開され、オブジェクトはモデルリポジトリにインポートされます。

インポートプロセスの完了後、インポートサマリを表示できます。

## オーバーライドプロパティ

インポートするときは、デフォルトでインポートプロセスがオーバーライドを保持します。インポートプロセス中に PowerCenter オブジェクトのオーバーライドプロパティを保持するか無視するかを選択できます。

PowerCenter では、いくつかのマッピングプロパティをオーバーライドできます。プロパティをオーバーライドするには、Workflow Manager の [マッピング] タブでセッションのプロパティを設定します。

セッションのオーバーライドでは、PowerCenter のセッションごとに値が異なるソースファイル名などのプロパティをオーバーライドできます。たとえば、[ソースファイル名] プロパティに値 'OriginalFile' を設定できます。セッション 1 はこの値を "Session1\_File" にオーバーライドし、セッション 2 はこの値を "Session2\_File" にオーバーライドできます。

PowerCenter マッピングにセッションオーバーライドが含まれている場合、インポートプロセスはモデルリポジトリにマッピングまたはデータオブジェクトパラメータを定義してセッションオーバーライドを内部で変換し、それをマッピングタスクでパラメータにバインドします。

モデルリポジトリでオーバーライドプロパティを保持する必要がある場合、次の情報を検討してください。

- インポートプロセスは再利用可能および再利用不可能なトランスフォーメーション、または PowerCenter オブジェクトの再利用可能なデータオブジェクトを作成します。
- PowerCenter マッピングがソースおよびターゲットプロパティをオーバーライドすると、PowerCenter マッピングと同じオーバーライドプロパティ値を持つデータオブジェクトが作成されます。インポートプロセスは PowerCenter オブジェクトの名前に番号を付与して、データオブジェクトが作成します。
- フラットファイルソース、フラットファイルターゲット、ルックアップトランスフォーメーション、リレーショナルソース、およびリレーショナルターゲットのオーバーライドプロパティのみを保持できます。



次の表に、モデルリポジトリに保持される PowerCenter オブジェクトのオーバーライドプロパティのリストを示します。

PowerCenter クライアントプロパティ	Developer tool プロパティ
<b>所有者名</b> リレーショナルソースのセッションプロパティ内にあります。	<b>所有者</b> マッピングインスタンスのランタイムプロパティ内にあります。
<b>ソーステーブル名</b> リレーショナルソースのセッションプロパティ内にあります。	<b>リソース</b> マッピングインスタンスのランタイムリソースプロパティ内にあります。
<b>拒否ファイルディレクトリ</b> リレーショナルターゲットのセッションプロパティ内にあります。	<b>拒否ファイルディレクトリ</b> マッピングインスタンスのランタイムプロパティ内にあります。
<b>拒否ファイル名</b> リレーショナルターゲットのセッションプロパティ内にあります。	<b>拒否ファイル名</b> マッピングインスタンスのランタイムプロパティ内にあります。
<b>ターゲットテーブル名</b> リレーショナルターゲットのセッションプロパティ内にあります。	<b>ターゲットテーブル名</b> マッピングインスタンスのランタイムリソースプロパティ内にあります。
Pre SQL リレーショナルソースまたはターゲットのセッションプロパティ内にあります。	Pre SQL マッピングインスタンスの詳細プロパティ内にあります。
Post SQL リレーショナルソースまたはターゲットのセッションプロパティ内にあります。	Post SQL マッピングインスタンスの詳細プロパティ内にあります。
Sql <b>クエリ</b> リレーショナルソースまたはターゲットのセッションプロパティ内にあります。	SQL <b>クエリ</b> マッピングインスタンスのクエリプロパティ内にあります。
<b>ユーザー定義結合</b> ソース修飾子プロパティ内にあります。	<b>結合</b> マッピングインスタンスの単純クエリプロパティ内にあります。
<b>ソースフィルタ</b> ソース修飾子プロパティ内にあります。	<b>フィルタ</b> マッピングインスタンスの単純クエリプロパティ内にあります。
<b>ルックアップ SQL オーバーライド</b> リレーショナルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>ルックアップ SQL オーバーライド</b> ルックアップクエリプロパティ内にあります。
<b>ルックアップテーブル名</b> リレーショナルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>リソースルックアップ</b> 詳細ランタイムリソースプロパティ内にあります。
<b>ルックアップソースフィルタ</b> リレーショナルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>フィルタクエリ</b> ルックアップクエリ単純フィルタプロパティ内にあります。
<b>ルックアップキャッシュのディレクトリ名</b> フラットファイルまたはリレーショナルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>ルックアップキャッシュのディレクトリ名</b> ルックアップランタイムプロパティ内にあります。 例: CacheDir (param)。
<b>ルックアップソースファイル名</b> フラットファイルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>ルックアップソースファイル名</b> パラメータプロパティによって指定されるルックアップデータオブジェクト内にあります。
<b>ルックアップソースファイルディレクトリ</b> フラットファイルルックアップトランスフォーメーションのセッションプロパティ内にあります。	<b>ルックアップソースファイルディレクトリ</b> パラメータプロパティによって指定されるルックアップデータオブジェクト内にあります。

PowerCenter クライアントプロパティ	Developer tool プロパティ
ルックアップカラム区切り文字フラットファイルルックアップトランスフォーメーションのセッションプロパティ内にあります。	区切り文字ルックアップデータオブジェクト詳細カラム区切り文字プロパティ内にあります。
ソースファイルディレクトリフラットファイルソースセッションプロパティ内にあります。	ソースファイルディレクトリデータオブジェクト詳細読み取りプロパティ内にあります。
ソースファイル名フラットファイルソースセッションプロパティ内にあります。	ソースファイル名データオブジェクト詳細読み取りプロパティ内にあります。
カラム区切り文字フラットファイルソースセッションプロパティ内にあります。	区切り文字データオブジェクト詳細カラム区切り文字プロパティ内にあります。
拒否ファイル名フラットファイルターゲットセッションプロパティ内にあります。	拒否ファイル名マッピングインスタンスのランタイムプロパティ内にあります。
カラム区切り文字フラットファイルターゲットセッションプロパティ内にあります。	区切り文字データオブジェクト詳細カラム区切り文字プロパティ内にあります。
マージファイルディレクトリフラットファイルターゲットセッションプロパティ内にあります。	マージファイルディレクトリデータオブジェクト詳細書き込みプロパティ内にあります。
マージファイル名フラットファイルターゲットセッションプロパティ内にあります。	マージファイル名データオブジェクト詳細書き込みプロパティ内にあります。
出力ファイルディレクトリフラットファイルターゲットセッションプロパティ内にあります。	出力ファイルディレクトリデータオブジェクト詳細書き込みプロパティ内にあります。
出力ファイル名フラットファイルターゲットセッションプロパティ内にあります。	出力ファイル名データオブジェクト詳細書き込みプロパティ内にあります。
拒否ファイルディレクトリフラットファイルターゲットセッションプロパティ内にあります。	拒否ファイルディレクトリマッピングインスタンスのランタイムプロパティ内にあります。

## 競合の解決

PowerCenter からオブジェクトをインポートするときにモデルリポジトリ内に同じ名前のオブジェクトが存在する場合の、オブジェクト名の競合を解決することができます。

以下の競合解決オプションから選択できます。

### ターゲットのオブジェクトの名前を変更する

デフォルトの命名規則で PowerCenter リポジトリオブジェクトの名前を変更し、インポートします。オブジェクトの名前を変更するオプションは、デフォルトの競合解決オプションです。

### ターゲットのオブジェクトを上書きする

モデルリポジトリオブジェクトのオブジェクトを PowerCenter リポジトリのオブジェクトで置き換えます。

### ターゲットのオブジェクトを再利用する

モデルリポジトリのオブジェクトをマッピングで再利用します。

**重要:** モデルリポジトリでは、競合を解決するためにマッピングとマップレットを区別することはありません。例えば、インポートしたマップレットと同じ名前のマッピングがリポジトリに含まれる場合、競合を解決するように要求されます。オブジェクトを置き換えることを選択した場合、インポートプロセスによってマッピングがマップレットで置き換えられます。

## インポートサマリ

PowerCenter オブジェクトをモデルリポジトリにインポートした後、インポートプロセスによってインポートサマリが作成されます。

変換エラーがある場合は、インポートサマリをファイルに保存できます。インポートサマリには、インポートのステータス、変換されなかったオブジェクトの数、インポート後に無効となったオブジェクトの数、変換エラーが含まれます。インポート後のオブジェクトを Developer ツールで検証して、検証エラーを表示することもできます。

## データ型の変換

一部の PowerCenter データ型はモデルリポジトリでは無効です。無効なデータ型の PowerCenter オブジェクトをインポートすると、モデルリポジトリの有効な対応するデータ型に変換されます。

PowerCenter のセッションパラメータ、ワークフローパラメータ、およびワークフロー変数をモデルリポジトリに変換することはできません。インポートプロセスは、セッション内のすべてのパラメータまたは変数参照を、モデルリポジトリ内の対応するマッピングタスク内の文字列表現にマップします。

以下の表に、インポートプロセスで、対応するモデルリポジトリデータに変換される PowerCenter リポジトリデータ型を示します。

PowerCenter リポジトリのデータ型	モデルリポジトリのデータ型
Real	Double
Small Int	Integer
Nstring	String
Ntext	Text

# トランスフォーメーションの変換

PowerCenter トランスフォーメーションは、互換性に基づいて変換されます。トランスフォーメーションの中にはモデルリポジトリと互換ではないものがあります。制約付きのその他のインポート。

制約付きでインポートするか、またはインポートに失敗した PowerCenter トランスフォーメーションについて、以下の表で説明します。

PowerCenter トランスフォーメーション	インポートアクション
アグリゲータ	制限付きでインポートされる。
デタマスキング	インポートが失敗する。
エクスターナルプロシージャ	インポートが失敗する。
HTTP	インポートが失敗する。
ID 解決	インポートが失敗する。
Java	制限付きでインポートされる。
ジョイナ	制限付きでインポートされる。
ルックアップ	制限付きでインポートされる。
ノーマライザ	制限付きでインポートされる。
ランク	制限付きでインポートされる。
シーケンスジェネレータ	制限付きでインポートされる。
ソータ	制限付きでインポートされる。
ソース修飾子	制限付きでインポートされる。ソースおよびソース修飾子トランスフォーメーションが 1 つのデータオブジェクトとして完全にインポートされる。
ストアドプロシージャ	インポートが失敗する。
トランザクションコントロール	インポートが失敗する。
SQL	制限付きでインポートされる。
共有体	制限付きでインポートされる。
構造化されていないデータ	インポートが失敗する。
XML パーサー	インポートが失敗する。
XML ジェネレータ	インポートが失敗する。

## トランスフォーメーションのプロパティの制限

一部の PowerCenter トランスフォーメーションは、トランスフォーメーションプロパティに基づき、制限付きでインポートされます。

インポートプロセスは、特定のトランスフォーメーションプロパティの互換性に基づいて、以下のいずれかのアクションを実行する場合があります。

- 無視。トランスフォーメーションプロパティを無視して、オブジェクトをインポートします。
- 内部的に変換。オブジェクトをトランスフォーメーションプロパティとともにインポートしますが、プロパティは Developer ツールで公開されません。
- インポートの失敗。オブジェクトのインポートに失敗し、マッピングは無効になります。

### アグリゲータトランスフォーメーション

以下の表に、アグリゲータトランスフォーメーションのプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
トランスフォーメーション範囲	無視。

### Java トランスフォーメーション

Java トランスフォーメーションでは、ポートは入力ポートまたは出力ポートのどちらかである必要があります。Java トランスフォーメーションに入力ポートと出力ポート両方がある場合、インポートは失敗します。

以下の表に、Java トランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
クラス名	無視。
関数識別子	無視。
トランザクションの生成	無視。
入力はブロック	無視。
パーティション化可能	無視。
言語	無視。
モジュール識別子	無視。
出力は確定的	無視。
出力が再現可能	無視。
パーティションごとに 1 つのスレッドを要求します	無視。

トランスフォーメーションのプロパティ	インポートアクション
実行時位置	無視。
アップデートストラテジトランスフォーメーション	無視。

## ジョイナトランスフォーメーション

以下の表に、ジョイナトランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
マスタでの NULL の順序付け	内部的に変換。
NULL の詳細な順序付け	内部的に変換。
トランスフォーメーション範囲	内部的に変換。

## ルックアップトランスフォーメーション

以下の表に、ルックアップトランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
キャッシュファイル名プレフィックス	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はインポート。
ルックアップキャッシュの初期化	無視。
ルックアップキャッシュディレクトリ名	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はインポート。
ルックアップキャッシュが有効	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はインポート。
ルックアップデータのキャッシュサイズ	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はインポート。
ルックアップのインデックスキャッシュサイズ	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はインポート。
ルックアップソースは静的です	無視。
ルックアップ SQL オーバライド	スタンドアロントランスフォーメーションとして変換された場合は無視、マッピング内で変換された場合はカスタム SQL クエリにインポート。

トランスフォーメーションのプロパティ	インポートアクション
ルックアップソースフィ ルタ	スタンドアロントランスフォーメーションとして変換された場合は無視、マ ッピング内で変換された場合はインポート。
ルックアップキャッシュ の事前作成	スタンドアロントランスフォーメーションとして変換された場合は無視、マ ッピング内で変換された場合はインポート。
ルックアップソースから の再キャッシュ	スタンドアロントランスフォーメーションとして変換された場合は無視、マ ッピング内で変換された場合はインポート。
古い場合に再キャッシュ	無視。
サブ秒の精度	無視。
動的キャッシュの同期	無視。

## ノーマライザトランスフォーメーション

ノーマライザトランスフォーメーションを Developer tool にインポートする場合、1 つの入力グループおよび少なくとも 1 つの出力グループを持つノーマライザトランスフォーメーションをインポートします。

マッピングの一部ではないノーマライザトランスフォーメーションをインポートする場合、Developer tool はノーマライザトランスフォーメーションの入力グループにすべての入力ポートを置きます。Developer tool は出力ポートのノーマライザトランスフォーメーションルールに基づいてデフォルトの出力グループを作成します。インポートするノーマライザトランスフォーメーションに出力ポートがない場合、Developer tool はインポート済みのノーマライザトランスフォーメーションにデフォルトの出力グループを作成します。

ノーマライザトランスフォーメーションがマッピングの一部である場合、Developer tool はマッピングのダウ  
ンストリームトランスフォーメーションまたはターゲットへのリンクに基づいて複数の出力グループを作成  
することがあります。複数グループのトランスフォーメーションからターゲットへのリンクに関するルールおよ  
びガイドラインの詳細については、『*Developer トランスフォーメーションガイド*』を参照してください。

再利用可能なノーマライザトランスフォーメーションを含むマッピングをインポートする場合、Developer  
tool はトランスフォーメーションを再利用可能としてインポートします。また Developer tool は、マッピ  
ング内の再利用可能なノーマライザトランスフォーメーションインスタンスを再利用不可能なトランスフォー  
メーションインスタンスで置換します。Developer tool は再利用不可能なノーマライザトランスフォーメー  
ションからダウンストリームトランスフォーメーションおよびターゲットへの新しいリンクを生成します。

PowerCenter で、ノーマライザトランスフォーメーションにキーポートが少なくとも 1 つ生成されます。  
Developer tool では、ノーマライザトランスフォーメーションに生成されたキーポートは含まれません。ノ  
ーマライザトランスフォーメーションを PowerCenter からインポートする場合、Developer tool は生成された  
キーポートを無視します。

以下の表に、ノーマライザトランスフォーメーションのプロパティのインポートアクションについて示します。

トランスフォーメーシ ョンのプロパティ	インポートアクション
リセット	無視。
リスタート	無視。

## リンクトランスフォーメーション

以下の表に、リンクトランスフォーメーションプロパティのインポートアクションについて示します。

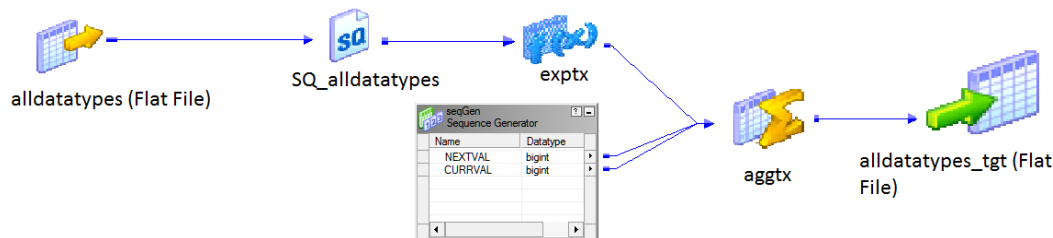
トランスフォーメーションのプロパティ	インポートアクション
トランスフォーメーション範囲	無視。

## シーケンスジェネレータトランスフォーメーション

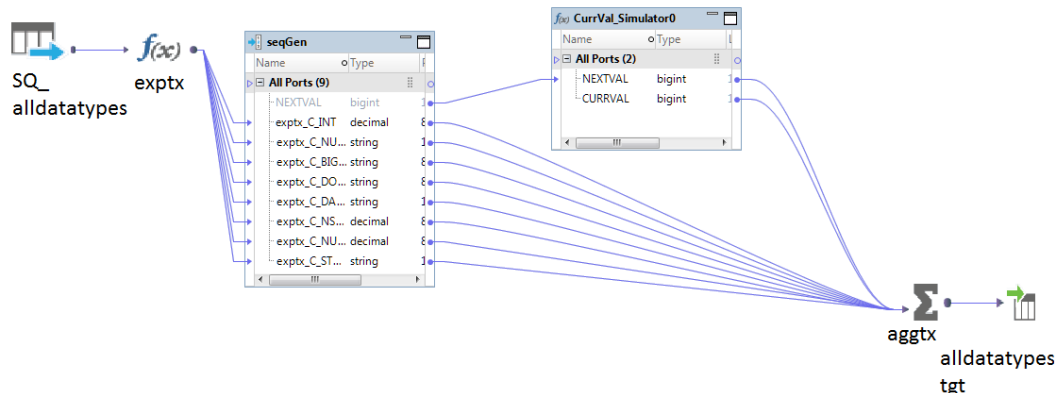
PowerCenter のシーケンスジェネレータトランスフォーメーションには 2 つのポート、CURRVAL および NEXTVAL があります。Developer tool では、シーケンスジェネレータトランスフォーメーションには出力ポート NEXTVAL のみがあります。マッピング内のシーケンスジェネレータトランスフォーメーションをインポートする場合、Developer tool は式トランスフォーメーションを作成して CURRVAL ポートの値を設定し、ダウンストリームトランスフォーメーションに渡します。

例えば、PowerCenter にシーケンスジェネレータ、アグリゲータ、および式トランスフォーメーションを持つマッピングがあるとします。ソースデータは、式トランスフォーメーション、アグリゲータトランスフォーメーション、ターゲットの順序で渡されます。シーケンスジェネレータトランスフォーメーションは、シーケンス番号を CURRVAL ポートを持つ各行に追加します。

次の図に、シーケンスジェネレータトランスフォーメーション、アグリゲータトランスフォーメーション、および式トランスフォーメーションを持つ PowerCenter マッピングを示します。



次の図に、Developer tool にインポートされたマッピングを示します。



マッピングをインポートする場合、Developer tool は NEXTVAL データをシーケンスジェネレータトランスフォーメーションから式トランスフォーメーションに渡して CURRVAL 値を設定します。

Developer tool は、マッピング内の再利用可能なシーケンスジェネレータトランスフォーメーションを再利用可能なシーケンスデータオブジェクトを持つ再利用不可能なトランスフォーメーションとしてインポートしま



す。マッピングの一部ではない再利用可能なシーケンスジェネレータトランスフォーメーションをインポートする場合、Developer tool は再利用可能なシーケンスデータオブジェクトを作成します。

以下の表に、シーケンスジェネレータトランスフォーメーションのプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
現在の値	無視。
キャッシュされる値の数	無視。

### ソータートランスフォーメーション

以下の表に、ソータートランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
トランスフォーメーション範囲	無視。

### ソース修飾子トランスフォーメーション

ソース修飾子トランスフォーメーションプロパティへのインポートアクションを以下の表に示します。

トランスフォーメーションのプロパティ	インポートアクション
ソートするポート数	無視。

### SQL トランスフォーメーション

以下の表に、SQL トランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
自動コミット	無視。
クラス名	無視。
接続タイプ	動的接続オブジェクトまたは完全な動的接続情報に設定されている場合、インポートに失敗する。
データベースタイプ	Sybase、Informix、または Teradata のインポートに失敗する。
関数識別子	無視。
トランザクションの生成	無視。

トランスフォーメーションのプロパティ	インポートアクション
入力ブロック	無視。
パーティション化可能	無視。
言語	無視。
プール内の最大接続数	無視。
モジュール識別子	無視。
出力は確定的	無視。
出力が再現可能	無視。
パーティションごとに1つのスレッドを要求します	無視。
実行時位置	無視。
SQL モード	スクリプトモードのインポートは失敗する。
トランスフォーメーション範囲	無視。
DB 接続エラーを重大として扱う	内部的に変換。
アップデートストラテジトランスフォーメーション	無視。
接続プールの使用	無視。

## 共有体トランスフォーメーション

以下の表に、共有体トランスフォーメーションプロパティのインポートアクションについて示します。

トランスフォーメーションのプロパティ	インポートアクション
クラス名	無視。
関数識別子	無視。
トランザクションの生成	無視。
入力ブロック	無視。
パーティション化可能	無視。
言語	無視。

トランスフォーメーションのプロパティ	インポートアクション
モジュール識別子	無視。
出力は確定的	無視。
出力が再現可能	無視。
パーティションごとに 1 つのスレッドを要求します	無視。
実行時位置	無視。
トランスフォーメーション範囲	無視。
アップデートストラテジトランスフォーメーション	無視。

## パラメータの変換

パラメータを含む PowerCenter マッピングまたはマプレットをインポートできます。

PowerCenter マッピングまたはマプレットにパラメータが含まれている場合、インポートプロセスで PowerCenter マッピングパラメータと、そのパラメータを参照する再利用可能なトランスフォーメーション間にパラメータバインディングが作成されます。インポートプロセスでは、トランスフォーメーションレベルのパラメータが生成されます。

PowerCenter からインポートする際は、パラメータ、変数またはオーバーライドを含むマッピングをインポートするための変換情報について考慮してください。

## パラメータを使用したマッピングのインポートのルールとガイドライン

パラメータを使用してマッピングをインポートするときは、次のルールおよびガイドラインを考慮してください。

**PowerCenter では、パラメータの初期値は不要です。**

初期値を指定しないで PowerCenter パラメータをインポートすると、インポートプロセスでパラメータのデータ型に基づいてパラメータのデフォルト値が割り当てられます。String データ型のパラメータのデフォルト値は、番号記号 (#) です。Numeric データ型のパラメータのデフォルト値は 0 です。Date/Time データ型のパラメータのデフォルト値は、mm/dd/yy の形式で 01/01/70 です。SQL パラメータのデフォルト値は空です。

**IsExprVar プロパティを使用した PowerCenter マッピングは、モデルリポジトリでは無効です。**

一部の PowerCenter マッピングパラメータでは、IsExprVar プロパティが有効になっています。このプロパティは、式を解析する前に PowerCenter 統合サービスでパラメータを展開する必要があることを示します。IsExprVar プロパティは、モデルリポジトリでは無効です。このパラメータプロパティが有効になっているマッピングをインポートする場合、変換には成功しますが、マッピングは無効になります。

Netezza および Teradata オブジェクト内のパラメータ化されたソース所有者または Teradata マッピング内のターゲットテーブル名のインポートは失敗します。

パラメータ化されたソース所有者名プロパティが含まれる Netezza および Teradata オブジェクトをインポートした場合、インポートプロセスによってプロパティが変換されません。インポートプロセスでは、Teradata マッピング内のパラメータ化されたターゲットテーブル名も無視されます。

## 変数を使用したマッピングのインポートのルールとガイドライン

変数を使用してマッピングをインポートするときは、次のルールおよびガイドラインを考慮してください。

**変数を使用した PowerCenter マッピングは、モデルリポジトリでは無効です。**

変数が含まれているマッピングをインポートしても、インポートプロセスで変数は変換されません。マッピングの変換に成功することもあります、そのマッピングはモデルリポジトリでは無効になります。マッピングを変更して、モデルリポジトリの変数の代わりにパラメータを使用できます。

## オーバーライドのインポート時のルールとガイドライン

オーバーライドをインポートするときは、次のルールおよびガイドラインを考慮してください。

**インポートプロセスは、セッションオーバーライドに対応するパラメータのタイプに変換します。**

SQL ベースのオーバーライドをモデルリポジトリにインポートすると、インポートプロセスはオーバーライドを SQL パラメータのタイプに変換します。例えば、PowerCenter マッピングをモデルリポジトリにインポートすると、Pre SQL および Post SQL が SQL パラメータのタイプに変更されます。また、ユーザー定義の結合およびソースフィルタは SQL パラメータのタイプに変換されます。

残りのセッションオーバーライドプロパティは、文字列または対応するパラメータのタイプに変換されます。例えば、拒否ファイルは文字列パラメータのタイプに変換されます。

PowerCenter では、セッションオーバーライドプロパティでユーザー定義の結合、SQL クエリ、Post SQL など文字列型のマッピングパラメータを使用している場合、参照されるパラメータはインポートプロセスで文字列型として取り込まれます。インポート後、SQL タイプのプロパティに割り当てられたマッピングパラメータのタイプを手動で変更できます。

## システムパラメータ変換

一部のシステム定義パラメータを含む PowerCenter マッピングまたはマップレットをインポートできます。パラメータがモデルリポジトリで有効なシステム定義パラメータと一致する場合、インポートプロセスでパラメータがインポートされます。

インポートプロセスでは、システム定義パラメータと、そのパラメータを参照する再利用可能なトランスフォーメーション間にパラメータバインディングが作成されます。

PowerCenter マッピングに、モデルリポジトリ内に相当するシステム定義パラメータがないシステム定義パラメータがある場合、変換は失敗しません。インポートプロセスで、そのパラメータ名のマッピングプロパティがプロパティ値としてコピーされます。ただし、インポートされたマッピングは有効ではありません。ユーザー定義パラメータを作成してプロパティ値を置き換えるか、マッピングロジックを変更することができます。

次のシステム定義パラメータをインポートできます。

- \$PMMappingName
- \$PMIntegrationServiceName
- \$PMRepositoryUserName
- \$SESSSTARTTIME
- \$SYSDATE

次の PowerCenter システム定義パラメータはインポートできません。

- \$PMFolderName
- \$PMRepositoryServiceName
- \$PMSessionName
- \$PMSessionRunMode
- \$PMTAB\_ALL\_DATA\_TYPES@TableName
- \$PMTGT\_ALL\_DATA\_TYPES@TableName
- \$PMWorkflowName
- \$PMWorkflowRunId
- \$PMWorkflowRunInstanceName

**注:** PowerCenter リポジトリ内のシステムパラメータにプロパティを割り当てると、インポートプロセスがシステムパラメータを変換できない場合があります。インポートプロセスはモデルリポジトリでデフォルトのパラメータ値を設定することがあります。インポートされたマッピング内のプロパティ値は編集できません。

## PowerCenter リポジトリの接続プロパティ

PowerCenter リポジトリからオブジェクトをインポートする際には、接続プロパティをリポジトリに指定する必要があります。Developer tool はこのインポートプロパティを使用して、PowerCenter リポジトリに接続します。

以下の表は、インポートプロパティの説明です。

プロパティ	説明
ホスト名	PowerCenter ドメインゲートウェイのホスト名です。
ポート番号	PowerCenter ドメインゲートウェイの HTTP ポート番号です。
リリース番号	PowerCenter のリリースバージョンです。
認証タイプ	PowerCenter リポジトリへの接続に必要なユーザー認証のタイプです。次のうちいずれかの値を選択します: Kerberos シングルサインオン、ネイティブ、または LDAP。 <b>注:</b> 認証タイプがネイティブまたは LDAP の場合は、ユーザー名とパスワードを指定します。
ユーザー名	PowerCenter リポジトリのユーザー名です。
パスワード	PowerCenter リポジトリのユーザー名のパスワードです。
セキュリティドメイン	認証タイプが LDAP の場合は、LDAP セキュリティドメイン名を指定します。それ以外の場合は、ネイティブと入力します。
リポジトリ名	PowerCenter リポジトリ名です。
コードページ	PowerCenter リポジトリのコードページです。

## 接続の割り当て

PowerCenter からデータソースおよびその他のオブジェクトをインポートする場合、データソース接続タイプを割り当てることができます。

例えば、Oracle データベースに接続する PowerCenter のソース定義、ターゲット定義、またはルックアップトランスフォーメーションを作成できます。Developer tool でこれらのオブジェクトをインポートする場合、**【接続の割り当て】** ダイアログボックスを使用して、各オブジェクトの接続タイプを指定します。

PowerCenter リポジトリオブジェクトの接続タイプを選択する場合、次のいずれかの方法を使用して、接続を割り当てます。

**単一の接続を複数の PowerCenter オブジェクトに同時に割り当てます。**

割り当てられた接続を持たないすべてのソース、すべてのターゲット、すべてのルックアップトランスフォーメーション、またはすべてのオブジェクトに単一の接続を割り当てることができます。または、単一の接続を、特定の名前のパターンと一致する名前を持つすべてのオブジェクトに割り当てることができます。**【選択】** リストからオプションを選択し、**【接続の割り当て】** をクリックします。

**単一の接続を、複数の異なるオブジェクトタイプの PowerCenter オブジェクトに割り当てます。**

**【カスタム】** オプションを **【選択】** リストで選択し、複数の PowerCenter オブジェクトを選択して、**【接続を割り当て】** をクリックします。

PowerCenter オブジェクトに接続を割り当てます。

PowerCenter オブジェクトを選択し、**【接続名】** カラムの **【開く】** ボタンをクリックします。

元の接続タイプとは異なるオブジェクトに接続を割り当てることができます。この操作は、Developer tool で元の接続タイプがサポートされていない場合などに行います。元の接続タイプとは異なる接続タイプを割り当てる場合、Developer tool に接続タイプの不一致の警告が表示されます。

警告を無視して続行するように選択すると、インポートに成功し、インポートしたデータソースに新しい接続が割り当てられます。選択した接続のメタデータが、インポートしたデータソースのスキーマと一致している場合、ソースまたはターゲットが有効になります。

**注:** Developer tool でサポートされていない接続タイプで作成されたルックアップトランスフォーメーションの場合、元の接続タイプに関する情報が使用できないため、Developer tool に接続の不一致に関する警告が表示されません。

## PowerCenter からのオブジェクトのインポート

PowerCenter リポジトリからモデルリポジトリにオブジェクトをインポートすることができます。

PowerCenter からオブジェクトをインポートする前に、ターゲットモデルリポジトリに接続します。

1. **【ファイル】** > **【インポート】** を選択します。

**【インポート】**ダイアログボックスが表示されます。

2. **【Informatica】** > **【PowerCenter】** を選択します。

3. **【次へ】** をクリックします。

**【PowerCenter からのインポート】**ダイアログボックスが表示されます。

4. PowerCenter リポジトリの接続パラメータを指定します。

5. **【テスト接続】** をクリックします。

PowerCenter リポジトリへの接続がテストされます。

6. PowerCenter リポジトリへの接続が成功したら、**[OK]** をクリックします。**[次へ]** をクリックします。  
PowerCenter リポジトリのフォルダーが表示され、インポートするオブジェクトを選択するように求めるメッセージが表示されます。
7. インポートするオブジェクトを 1 つ以上選択します。
8. **[次へ]** をクリックします。
9. モデルリポジトリ内でオブジェクトのインポート先の場所を選択します。
10. オブジェクト名の競合に対する競合解決オプションを選択します。ターゲットモデルリポジトリで、オブジェクトの名前の変更、置換、または再利用を選択します。
  - デフォルトの命名規則を使用している PowerCenter リポジトリオブジェクトの名前を変更してからモデルリポジトリにインポートするには、**[ターゲットのオブジェクトの名前を変更する]** オプションを選択します。オブジェクトの名前を変更するオプションは、デフォルトの競合解決オプションです。
  - PowerCenter リポジトリオブジェクトを持つモデルリポジトリオブジェクトを置換するには、**[ターゲットのオブジェクトを上書きする]** オプションを選択します。
  - PowerCenter オブジェクトをインポートするのではなく、マッピング内のモデルリポジトリのオブジェクトを再利用するには、**[ターゲットのオブジェクトを再利用する]** オプションを選択します。
11. **[次へ]** をクリックします。  
PowerCenter オブジェクトと依存オブジェクトが表示されます。
12. 再利用可能な PowerCenter ソース、ターゲット、トランスフォーメーションのオーバーライドプロパティを無視するには、**[オーバーライドされたプロパティを無視する]** をクリックします。デフォルトでは、プロセスで上書きプロパティが保持されます。
13. IBM DB2 オブジェクトをインポートする場合は、DB2 オブジェクトタイプを選択します。以下のいずれかのオブジェクトタイプを選択できます。LOW、z/OS、i5/OS。
14. **[次へ]** をクリックします。
15. PowerCenter リポジトリオブジェクトに対するモデルリポジトリの接続の詳細を指定します。
16. **[接続の選択]** ダイアログボックスが開き接続を選択し、**[OK]** をクリックします。
17. **[次へ]** をクリックします。  
Developer ツールによってインポートの概要が作成され、PowerCenter オブジェクトとインポートされる依存オブジェクトが一覧表示されます。
18. **[変換チェック]** をクリックし、オブジェクトが有効なモデルリポジトリオブジェクトとしてインポートできるかどうかを検証します。  
変換チェックのサマリレポートに、変換チェックの結果が表示されます。
19. **[OK]** をクリックします。**[完了]** をクリックします。  
インポート中、進捗情報が表示されます。PowerCenter オブジェクトおよび依存オブジェクトがモデルリポジトリにインポートされ、最終的なインポートサマリが生成されます。
20. 変換エラーがある場合、**[保存]** をクリックして、インポートサマリを保存するファイル名を指定します。

# インポートの制限

PowerCenter リポジトリからモデルリポジトリにオブジェクトをインポートするときは、PowerCenter オブジェクトに適用される制限とガイドラインがあります。

次の PowerCenter オブジェクトに特定の制限が適用されます。

- ソースおよびターゲット
- トランスフォーメーション
- マッピング
- 関数
- マッピング変数
- セッションのプロパティ
- ワークフローとワークレット
- タスク

## ソースおよびターゲットの制限

PowerCenter オブジェクトをインポートするときは、次のソースおよびターゲットの制限が適用されます。

- PowerCenter の 9.1.0 以前のリリースからソースまたはターゲットをインポートする場合、インポートプロセスでは、オブジェクトに関連付けられている接続タイプが有効かを検証できません。
- PowerCenter リポジトリのバージョンが 9.5.0 以前の場合、IBM DB2 のソースデータベース名またはターゲット名の先頭には、DB2 タイプを設定するために "DB2" を付けます。
- フラットファイルソースの行の区切り文字が無効な場合、インポートプロセスによってデフォルト値に変更されます。
- PowerCenter から Salesforce ソースまたはターゲットをインポートすることはできません。
- PowerCenter から Teradata ソースをインポートした場合、インポートプロセスで次のプロパティが無視されます。
  - カラムの [非 NULL] プロパティ
  - 出力は確定的
  - 出力が再現可能
  - ソートするポート数
  - 外部キー
- PowerCenter から Teradata ターゲットをインポートした場合、インポートプロセスで次のプロパティが無視されます。
  - カラムの [非 NULL] プロパティ
  - 更新オーバーライド
  - ターゲットテーブルプレフィックス
  - 外部キー
- PowerCenter から Netezza ソースをインポートした場合、インポートプロセスで次のプロパティが無視されます。
  - 外部キー
  - 出力は確定的



- 出力が再現可能
- ソートするポート数
- PowerCenter から Netezza ターゲットをインポートした場合、インポートプロセスで次のプロパティが無視されます。
  - 外部キー
  - 更新オーバーライド
- PowerCenter から Netezza ソースをインポートすると、その Netezza ソースは新しく作成された Netezza データオブジェクトの下に同じ名前でインポートされます。一方、読み取り操作を作成した場合、読み取り操作内のソーストランスフォーメーションはインポートしたソース名とは異なる番号が付加された一意の名前になります。
- オーバーライドされた SQL クエリ（ソース修飾子のカラムのサブセットを選択する）が設定されたソース修飾子を使用してマッピングをインポートすると、統合サービスはマッピングを実行できません。この問題を解決するには、余分なポートを削除し、Developer tool で物理データオブジェクトのソース内のすべてのカラムが含まれるように SQL クエリを一致させていることを確認します。

## トランスフォーメーション制限

PowerCenter オブジェクトをインポートするときには次のトランスフォーメーション制限が適用されます。

- トランスフォーメーションに含まれる式は、最大 4,000 文字とします。
- SQL トランスフォーメーションのデータベースタイプは、インポートプロセス中に ODBC に変換されます。
- トランスフォーメーションのデータキャッシュサイズまたはインデックスキャッシュサイズを無効な値に設定すると、インポートプロセスによってその値は Auto に変更されます。
- 空のルックアップキャッシュディレクトリ名を使用してマッピングをモデルリポジトリにインポートすると、インポートされたマッピングは無効になります。この問題を解決するには、Developer tool のルックアップトランスフォーメーションのランタイムプロパティで、ルックアップキャッシュディレクトリ名に有効な値を入力します。

## マッピングの制限

PowerCenter オブジェクトをインポートするときには次のマッピングの制限が適用されます。

- サポートされているオブジェクトまたはトランスフォーメーションを含むマッピングをインポートすると、ターゲットロード順を保持したまま、PowerCenter マッピング内のパイプラインごとに個別のモデルリポジトリマッピングが作成されます。  
このインポートプロセスの動作は、ワークフロー内のセッションで複数のパイプラインがあるマッピングを実行する場合でも同じです。各マッピング名には、ターゲットロード順のとおりマッピングを実行するために必要な順序を示す番号が付加されます。
- モデルリポジトリでサポートされていないオブジェクトがマッピングに含まれている場合、パイプラインはモデルリポジトリに単一のマッピングとしてインポートされます。  
サポートされていないトランスフォーメーションまたはオブジェクトがあるパイプラインは、リンクが壊れた状態でモデルリポジトリにインポートされます。この場合、ターゲットロード順によっては、インポートされたマッピングの数が PowerCenter マッピング内のパイプラインの総数よりも小さくなる場合があります。PowerCenter 内のパイプラインに、モデルリポジトリでサポートされていないトランスフォーメーションまたはオブジェクトが含まれている場合、パイプラインはオブジェクトが未接続となって破損し、モデルリポジトリの単一のマッピング内に表示されます。

## 関数の制限

PowerCenter オブジェクトをインポートするときは、SetVariable および SetMax 関数はインポートできません。代わりに、モデルリポジトリでマッピング出力になるようにこれらの関数を変換できます。

## マッピング変数の制限

マッピング変数を PowerCenter からモデルリポジトリにインポートすることはできません。

## セッションのプロパティ制約

次の表に、PowerCenter からモデルリポジトリにインポートできるセッションプロパティのリストを示します。

PowerCenter セッションプロパティ	Developer tool マッピングタスクプロパティ
セッションログファイル名	マッピングタスクログファイル名
セッションログファイルディレクトリ	マッピングタスクログファイルディレクトリ
[リカバリ戦略] > [タスクを失敗としてワークフローを続行]	[タスクのリカバリストラテジ] > [スキップ]
[リカバリ戦略] > [タスクの再開がサポートされている]	[タスクリカバリストラテジ] > [再開]
Java クラスパス	Java クラスパス
高精度 10 進演算を有効にする	高精度
セッションソート順	ソート順序
日時形式文字列	デフォルトの日時形式
[セッションログの保存方法] > [タイムスタンプによってセッションログを保存する]	[マッピングタスクログ保存タイプ] > [マッピングタスクタイムスタンプ]
[セッションログの保存方法] > [実行によってセッションログを保存する]	[マッピングタスクログ保存タイプ] > [マッピングタスク実行]
これらの実行のセッションログの保存	次の実行でマッピングタスクログを保存
トレースのオーバーライド	トレースレベルのオーバーライド
HA リカバリを有効にする	リカバリの有効化

残りのセッションプロパティは、インポートではサポートされていません。サポートされていないプロパティをインポートすると、デフォルト値が表示される場合があります。

## ワークフローの制限

PowerCenter オブジェクトをインポートするときには次のワークフロー制限が適用されます。

- PowerCenter では、ワークフローリンクを右クリックすると、Workflow Manager でリンク条件を設定できます。モデルリポジトリにワークフローをインポートした後、その条件はコメントとしてのみ表示されるので、必要な条件に基づいてマッピングを手動で変換する必要があります。
- ワークフローパラメータ、ワークフロー変数、およびセッションパラメータは、PowerCenter からモデルリポジトリへのインポートではサポートされていません。
- ネストされたワークレットを PowerCenter からインポートすることはできません。

## タスクの制限

PowerCenter オブジェクトをインポートするときは、モデルリポジトリ内の開始、セッション、コマンド、ワークレット、および終了タスクをインポートできます。

## インポートパフォーマンス

68MB を超えるマッピングをインポートする場合は、コマンドラインから行うと最適なパフォーマンスが得られます。infacmd ipc ImportFromPC コマンドを使用して、インポートのパフォーマンスを最適化できます。

PowerCenter からインポートする際にパフォーマンスを高めるには、infacmd ipc genReuseReportFromPC コマンドにオプション-BlockSize を追加します。

すべてのフォルダをインポートする代わりに、必要なマッピングだけを PowerCenter からモデルリポジトリにインポートできます。特にユーザーがメモリ不足エラーを受け取った場合、レポートの生成に推奨される Java ヒープサイズは 4GB で、ブロックサイズは 100 です。複雑なマッピングの場合は、ブロックサイズ値を小さくすることもできます。

## 第 13 章

# パフォーマンスのチューニング

この章では、以下の項目について説明します。

- [Performance Tuning Overview, 244](#) ページ
- [最適化方式, 245](#) ページ
- [Optimizer Levels, 249](#) ページ
- [Setting the Optimizer Level for a Developer Tool Mapping, 250](#) ページ
- [デプロイ済みのマッピングに対する最適化レベルの設定, 251](#) ページ

## Performance Tuning Overview

The Data Integration Service optimizes mappings to improve the performance of a mapping.

The Data Integration Service can perform the following optimizations:

Filter data to reduce the number of rows to be processed.

The Data Integration Service applies optimization methods in an attempt to reduce the amount of data to process. When you run a mapping, you can choose an optimizer level that determines which optimization methods the Data Integration Service can apply to the mapping. For example, the Data Integration Service can use early selection optimization to move a filter closer to the source. It can use pushdown optimization to push transformation logic to a database. It can use the cost-based optimization method to change the join processing order.

The Data Integration Service can apply multiple optimization methods to a mapping at the same time. For example, the Data Integration Service applies the early projection, predicate optimization, early selection, branch pruning, or push-into optimization methods when you select the normal optimizer level.

Determine the partitioning strategy to maximize parallel processing.

If you have the partitioning option, the Data Integration Service can maximize parallelism for mappings. The Data Integration Service dynamically determines the partitioning strategy for mappings. The partitioning strategy includes the location of partition points, the optimal number of partitions for each pipeline stage, and the partitioning types that best redistribute data across each partition point. For more information about partitioning, see [「パーティション化されたマッピングの概要」 \(ページ 278\)](#).

You can also set constraints on relational sources, logical data objects, physical data objects, and virtual tables in a mapping to filter unnecessary rows. The Data Integration Service can process constraints to improve mapping performance.

# 最適化方式

各種の最適化方式を適用することによって、マッピングで処理される行数が削減されます。マッピングの最適化レベルを設定して、適用される最適化方式を制限できます。

データ統合サービスでは、以下の最適化方式を適用できます。

- プッシュダウンの最適化
- 初期プロジェクション最適化
- 初期選択最適化
- ブランチ刈り込み最適化
- 最適化にプッシュイン
- 述部最適化
- グローバル述部最適化
- コストベース最適化
- データシップ結合最適化
- 準結合最適化

データ統合サービスでは、マッピングに複数の最適化方式を同時に適用できます。例えば、ノーマル最適化レベルを選択すると、データ統合サービスは、初期プロジェクション最適化、述部最適化、グローバル述部最適化、ブランチ刈り込み最適化、および初期選択最適化またはプッシュイン最適化方式を適用します。

## 初期プロジェクション最適化方法

データ統合サービスで初期プロジェクション最適化方式を適用すると、未使用のポートが特定され、それらのポート間のリンクが削除されます。

初期プロジェクション最適化方式では、トランスフォーメーション間を移動するデータの量を少なくすることによってパフォーマンスを向上させます。データ統合サービスは、マッピングを処理するときに、マッピング内のすべての接続ポートのデータをトランスフォーメーション間で移動します。大きく複雑なマッピング、またはネストされたマプレットを使用するマッピングには、データをターゲットに提供しないポートがある場合があります。データ統合サービスにより、データをターゲットに提供しないポートが特定されます。未使用のポートが特定されたら、すべての未使用のポート間のリンクがマッピングから削除されます。

すべてのリンクが削除されるわけではありません。例えば、以下のリンクは削除されません。

- 副次効果のあるトランスフォーメーションに接続されているリンク。
- ABORT()関数または ERROR()関数の呼び出し、メールの送信、またはストアドプロシージャの呼び出しを実行するトランスフォーメーションに接続されているリンク。

トランスフォーメーション内のすべてのポートが未使用であると判断されたら、データが最も少ないポートへのリンクを除くすべてのトランスフォーメーションリンクが削除されます。未使用のトランスフォーメーションはマッピングから削除されません。

この最適化方式は、Developer ツールではデフォルトで有効になっています。

## Early Selection Optimization Method

When the Data Integration Service applies the early selection optimization method, it splits, moves, or removes the Filter transformations in a mapping. It moves filters up the mapping closer to the source.

The Data Integration Service might split a Filter transformation if the filter condition is a conjunction. For example, the Data Integration Service might split the filter condition "A>100 AND B<50" into two

simpler conditions, "A>100" and "B<50." When the Data Integration Service splits a filter, it moves the simplified filters up the mapping pipeline, closer to the source. The Data Integration Service moves the filters up the pipeline separately when it splits the filter.

When you choose the normal or full optimizer level in the Developer tool, the early selection optimization method is enabled by default. When you use the auto optimizer level, the early selection optimization method is enabled in the following cases:

- The mapping runs in the native environment or on the Blaze engine.
- The mapping contains a data source that supports pushing filters to the source.

You can disable early selection if the optimization does not increase performance.

The Data Integration Service ignores early selection optimization if a transformation that appears before the Filter transformation has side effects. The Data Integration Service cannot determine if the SQL transformation, Web Service Consumer transformation, and Java transformation have side effects. You can configure early selection optimization for these transformations if they do not have side effects.

## ブランチ刈り込み最適化方式

データ統合サービスでは、マッピングにおいてターゲットに行をまったく渡さないトランスフォーメーションにブランチ刈り込み最適化方式を適用できます。

データ行に関してフィルタ条件が FALSE に評価されると、データ統合サービスはフィルタトランスフォーメーションを削除することがあります。例えば、マッピングに、2つのリレーショナルソースのデータをフィルタリングする2つのフィルタトランスフォーメーションが存在するとします。一方のフィルタトランスフォーメーションには「Country=US」というフィルタ条件が設定されており、もう一方のフィルタトランスフォーメーションには「Country=Canada」というフィルタ条件が設定されているとします。共有体トランスフォーメーションは、これら2つのリレーショナルソースを結合するもので、「Country=US」というフィルタ条件が設定されているとします。データ統合サービスは、フィルタ条件「Country=Canada」が設定されたフィルタトランスフォーメーションをマッピングから削除することがあります。

ブランチ刈り込み最適化方式は、最適化レベルをノーマルまたは完全に設定した場合に Developer ツールでフォルトで有効になります。この最適化でパフォーマンスが向上しない場合は、最適化レベルを最小またはなしに設定してブランチ刈り込みを無効にすることができます。

## 述部最適化方式

データ統合サービスで述部最適化方式を適用すると、マッピングで生成された述部式が調べられ、マッピングのパフォーマンス向上のために式を簡略化するか書き直すことができるかどうか判断されます。

データ統合サービスは、マッピングを実行するときに、マッピングソースに対するクエリを生成し、マッピングログックおよびマッピング内のトランスフォーメーションに基づいてクエリ結果に対して操作を実行します。クエリおよび操作には、多くの場合、述部式が含まれます。述部式は、データが満たす必要がある条件を表します。述部式の例としては、フィルタトランスフォーメーションとジョイナトランスフォーメーションのフィルタ条件と結合条件があります。

また、述部最適化方式では、マッピングのパフォーマンス向上のために、マッピングで述部式をできる限り早期に適用するように試行されます。

データ統合サービスでは、既存の述部式からリレーションを推測し、新しい述部式を作成します。例えば、マッピングに結合条件「A=B」を持つジョイナトランスフォーメーションおよびフィルタ条件「A>5」を持つフィルタトランスフォーメーションが含まれているとします。データ統合サービスは、「B>5」を結合条件に追加できます。

データ統合サービスでは、述部最適化方式と初期選択最適化方式の両方をマッピングに適用できる場合、それらの方式が適用されます。例えば、データ統合サービスで述部最適化方式を使用して新しいフィルタ条件が作成されるとき、初期選択方式を使用してマッピングのアップストリームへの条件の移動も試行されます。両方

の最適化方式を適用すると、一方の方式のみを適用したときと比べてマッピングのパフォーマンスが向上します。

データ統合サービスで述部最適化方式が適用されるのは、それによってパフォーマンスが向上する場合です。適用することでマッピングの結果が変わったりマッピングのパフォーマンスが低下したりする場合は、この方式は適用されません。データ統合サービスでは、デフォルトでこの最適化方式が適用されます。

## コストベースの最適化方式

コストベースの最適化では、データ統合サービスによって、マッピングが評価されて意味的に同等のマッピングが生成され、最適なパフォーマンスでマッピングが実行されます。コストベースの最適化では、隣接する内部結合や完全な外部結合の操作を実行するマッピングの実行時間が短縮されます。

意味的に同等のマッピングとは、同じ関数を実行して同じ結果になるマッピングです。意味的に同等のマッピングを生成するために、データ統合サービスでは、元のマッピングがフラグメントに分割されます。次に、最適化できるマッピングのフラグメントが特定されます。

最適化中に、フラグメント内のトランスフォーメーションの追加、削除、または順序変更が行われる場合があります。最適化したフラグメントが元のフラグメントと同じ結果になることが検証され、最適化したフラグメントを使用する代替マッピングが形成されます。

データ統合サービスは、ソートされたマージ結合のパフォーマンスが、ネストされたループ結合のパフォーマンスよりも高いと判断した場合に、ソートされたマージ結合を適用することもできます。ソートされたマージ結合では、結合を実行する前に、ソート順を使用して2つのデータセットを整列します。ネストされたループ結合では、ネストされたループを使用して2つのデータセットを結合します。データをソートするためのコストが、ネストされたループ結合を処理するためのコストより低い場合、データ統合サービスは、ソースのソート情報を使用したり、ソータートランスフォーメーションを作成したりできます。

元のマッピングと意味的に同等のすべてのマッピングまたはほぼすべてのマッピングが生成されます。プロファイリング統計またはデータベース統計を使用して、元のマッピングおよび各代替マッピングのコストを計算します。次に、最も早く実行するマッピングを特定します。最適な代替マッピングに対して検証チェックが実行され、そのマッピングが有効で元のマッピングと同じ結果になることが確認されます。

最適な代替マッピングがメモリにキャッシュされます。マッピングを実行するときに、代替マッピングが取得されて元のマッピングの代わりに実行されます。

この方式は、Developer tool ではデフォルトで無効になっています。

## データシップ結合最適化方式

データシップ結合最適化方式では、大きいデータセットに隣接する小さいデータセットを特定し、結合処理の時間を短縮しようとします。データ統合サービスは、2つのテーブルのサイズが大幅に異なる場合、データシップ結合最適化方式を適用しようとします。

例えば、データ統合サービスは、データシップ結合最適化方式を適用して、10,000 行のマスタテーブルと 1,000,000 行の詳細テーブルを結合できます。データ統合サービスは、データシップ結合を実行するために、大きい詳細テーブルを含むデータベースに一時ステー징テーブルを作成します。次に、データ統合サービスは、小さいマスタテーブルを一時テーブルにコピーし、一時テーブルのデータと大きい詳細テーブルのデータを結合します。データ統合サービスが結合操作を実行すると、データベースでジョイナトランスフォーメーションロジックが処理されます。

データシップ結合最適化方式を適用する前に、データ統合サービスは分析を実行し、データシップ結合最適化が可能か、また実行する価値があると考えられるかを判断します。分析によって、この方式でパフォーマンスが向上する可能性が高いと判断されたら、この方式がマッピングに適用されます。その後、マッピングが再分析されて、データシップ結合最適化を行う機会が他にもあるかどうか判断されます。必要に応じて、最適化がさらに実行されます。

この方式は、Developer ツールではデフォルトで無効になっています。



## データシップ結合でパフォーマンスを向上させるための要件

データシップ結合最適化方式では、パフォーマンスが常に向上するとは限りません。データシップ結合最適化によるマッピングのパフォーマンスに影響する要因を次に示します。

- ジョイナトランスフォーメーションのマスタソースの行が明細ソースよりも大幅に少なくなければならない。
- 明細ソースが最適化の正当性を保証できるほど大幅に大きくなければならない。明細ソースが十分に大きくない場合、データ統合サービスは、データシップ結合最適化方式を適用せずに、マスタソースおよび明細ソースからすべてのデータを読み取る方が速いと判断します。

## データシップ結合最適化に関するルールとガイドライン

データ統合サービスでは、次の要件を満たすジョイナトランスフォーメーションにデータシップ結合最適化を適用できます。

- 結合タイプがノーマル、マスタ外部、または明細外部である。
- 明細パイプラインがリレーショナルソースから生成されている。
- ジョイナトランスフォーメーション範囲が「すべての入力」である（マッピングでターゲットベースのコミットが使用される場合）。
- マスタパイプラインと明細パイプラインでトランスフォーメーションが共有されていない。
- 明細ソースとジョイナトランスフォーメーションの間のブランチがマッピングに含まれていない。
- 結合の詳細サイドを含むデータベースが Unicode エンコーディングをサポートしない IBM DB2 データベースの場合、データ統合サービスはデータシップ結合最適化方式の適用に失敗します。

## 準結合最適化方式

準結合最適化方式では、マッピングで結合操作を変更することで、ソースから抽出されるデータの量の削減が試行されます。

データ統合サービスでは、一方の入力グループに他方よりも多くの行が含まれている場合、結合条件に基づいて、小さい方のグループに一致するものがない行が大きい方のグループに多数含まれているときに、準結合最適化方式がジョイナトランスフォーメーションに適用されます。データ統合サービスは、小さい方のグループから行を読み取り、大きい方のグループで一致する行を見つけて結合操作を実行することで、1つの結合オペランドのデータセットのサイズを小さくしようとします。データセットのサイズを小さくすると、データ統合サービスで大きい方のグループソースから不要な行が読み取られなくなるため、マッピングのパフォーマンスが向上します。データ統合サービスによって、結合条件が大きい方のグループソースに移動され、小さい方のグループと一致する行のみが読み取られます。

準結合最適化方式を適用する前に、データ統合サービスは分析を実行し、準結合最適化が可能か、また実行する価値があると考えられるかを判断します。分析によって、この方式でパフォーマンスが向上する可能性が高いと判断されたら、この方式がマッピングに適用されます。その後、マッピングが再分析されて、準結合最適化を行う機会が他にもあるかどうか判断されます。必要に応じて、最適化がさらに実行されます。

この方式は、Developer ツールではデフォルトで無効になっています。

## 準結合最適化でパフォーマンスを向上させるための要件

準結合最適化方式では、パフォーマンスが常に向上するとは限りません。準結合最適化によるマッピングのパフォーマンスに影響する要因を次に示します。

- ジョイナトランスフォーメーションのマスタソースの行が明細ソースよりも大幅に少なくなければならない。



- 明細ソースが最適化の正当性を保証できるだけの大きさでなければならない。準結合最適化方式を適用すると、マッピング処理にオーバーヘッド時間が加わります。明細ソースが小さい場合は、準結合最適化の適用にかかる時間が、明細ソース内のすべての行の処理にかかる時間を超える可能性があります。
- Data Integration Service で、通常の結合操作と準結合操作にかかる時間を正確に比較できるように、ジョイナトランスフォーメーションのソース行数の統計を取得できなければならない。

## 準結合最適化に関するルールとガイドライン

Data Integration Service では、以下の要件を満たすジョイナトランスフォーメーションに準結合最適化を適用できます。

- 結合タイプがノーマル、マスタ外部、または明細外部である。ジョイナトランスフォーメーションでは、完全外部結合は実行できません。
- 明細パイプラインがリレーショナルソースから生成されている。
- 結合条件が有効なソート-マージ-結合条件である。つまり、各句が1つのマスタポートと1つの明細ポートの等式である必要があります。複数の句がある場合は、AND で結合する必要があります。
- ジョイナトランスフォーメーション範囲が [すべての入力] である（マッピングでターゲットベースのコミットが使用されない場合）。
- マスタパイプラインと明細パイプラインでトランスフォーメーションが共有されていない。
- 明細ソースとジョイナトランスフォーメーションの間のブランチがマッピングに含まれていない。

## 最適化されたマッピングの表示

最適化されたマッピングを表示して、最適化方式がマッピングにどのように影響するかを判断できます。

- ▶ エディタの空の領域を右クリックし、**[最適化されたマッピングの表示]** をクリックします。  
データ統合サービスにより最適化されたマッピングが生成されます。  
**注:** 最適化されたマッピングでは、データをプレビューできません。

# Optimizer Levels

The Data Integration Service optimizes mappings based on the optimizer level that you configure. Configure the optimizer level property when you want the mapping to use an optimizer level other than the default.

以下の最適化レベルのいずれかを選択できます。

### 自動

データ統合サービスは、実行モードとマッピングコンテンツに基づいて最適化を適用します。

### なし

データ統合サービスは最適化は適用されません。

### 最小

データ統合サービスは初期プロジェクション最適化方式を適用します。

## ノーマル

データ統合サービスは、初期プロジェクション、初期選択、ブランチ刈り込み、プッシュイン、グローバル述部、述部の最適化方式を適用します。

## 完全

データ統合サービスは、コストベース、初期プロジェクション、初期選択、ブランチ刈り込み、述部、プッシュイン、準結合、データシップ結合の最適化方式を適用します。

デフォルトは「自動」です。

When you run the mapping from the **Run** menu or mapping editor in the Developer tool, the Data Integration Service applies the optimizer level in the mapping configuration. When you run the mapping from the command line, the Data Integration Service applies the optimization level from the mapping deployment properties in the application.

**注:** The Data Integration Service does not apply the pushdown optimization method with an optimizer level. You can configure pushdown optimization for a mapping in the mapping run-time properties.

## 関連項目：

- [「プッシュダウンの最適化の概要」 \(ページ 252\)](#)

# Setting the Optimizer Level for a Developer Tool Mapping

When you run a mapping through the Run menu or mapping editor, the Developer tool runs the mapping with the optimizer level you set in the mapping configuration. To run the mapping with a different optimizer level, run the mapping from the **Run Configurations** dialog box.

1. Open the mapping.
2. Select **Run > Open Run Dialog**.  
The **Run Configurations** dialog box appears.
3. Select a mapping configuration that contains the optimizer level you want to apply or create a mapping configuration.
4. Click the **Advanced** tab.
5. Change the optimizer level.
6. Click **Apply**.
7. Click **Run** to run the mapping.

The Developer tool runs the mapping with the optimizer level in the selected mapping configuration.

# デプロイ済みのマッピングに対する最適化レベルの設定

アプリケーションのマッピングのデプロイメントのプロパティを変更して、コマンドラインから実行するマッピングの最適化レベルを設定します。

マッピングがアプリケーションに含まれている必要があります。

1. マッピングを含むアプリケーションを開きます。
2. **【詳細】** タブをクリックします。
3. 最適化レベルを選択します。
4. アプリケーションを保存します。

最適化レベルを変更した後、アプリケーションを再デプロイする必要があります。

## 第 14 章

# プッシュダウンの最適化

この章では、以下の項目について説明します。

- [プッシュダウンの最適化の概要, 252 ページ](#)
- [プッシュダウンタイプ, 253 ページ](#)
- [トランスフォーメーションプッシュロジック, 255 ページ](#)
- [ソースへのプッシュダウンの最適化, 256 ページ](#)
- [プッシュダウンの最適化の式, 260 ページ](#)
- [データ統合サービスとソースの出力の比較, 277 ページ](#)

## プッシュダウンの最適化の概要

データ統合サービスでプッシュダウンの最適化を適用すると、トランスフォーメーションロジックがソースにプッシュされます。データ統合サービスはトランスフォーメーションロジックを SQL クエリに変換し、その SQL クエリをデータベースに送信します。トランスフォーメーションを処理する SQL クエリはソースデータベースで実行されます。

ソースデータベースがデータ統合サービスよりも高速にトランスフォーメーションロジックを処理できる場合、プッシュダウンの最適化により、マッピングのパフォーマンスが向上します。データ統合サービスがソースから読み取るデータも少なくなります。

データ統合サービスがソース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決まります。データ統合サービスは、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。

プッシュダウンの最適化をマッピングに設定すると、ソースからターゲットまで、またはソースデータベースにプッシュできないダウンストリームトランスフォーメーションに達するまで、最適化されたマッピングが分析されます。データ統合サービスでは、トランスフォーメーションロジックがプッシュダウンされたソースごとに SELECT 文を生成して実行します。次に、Data Integration Service はこの SQL クエリの結果が読み込み、マッピングの残りのトランスフォーメーションを処理します。

関連項目：

- [「Optimizer Levels」 \(ページ 249\)](#)

## プッシュダウンタイプ

マッピングのランタイムプロパティでプッシュダウンタイプを選択した場合、データ統合サービスはプッシュダウンの最適化をマッピングに適用します。

以下のプッシュダウンタイプを選択することができます。

- なし。マッピングのプッシュダウンタイプを選択しません。
- ソース。データ統合サービスは、ソースデータベースに、できるだけ多くのトランスフォーメーションロジックのプッシュダウンを試みます。
- 全体。データ統合サービスは、トランスフォーメーションロジック全体をソースデータベースにプッシュします。

プッシュダウンタイプの文字列パラメータを作成し、以下のパラメータ値を使用することもできます。

- フル
- ソース
- なし

## 完全なプッシュダウンの最適化

データ統合サービスは、完全なプッシュダウンの最適化を適用するときに、マッピングのすべて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ます。マッピングランタイムプロパティで完全なプッシュダウンを設定できます。

完全なプッシュダウンの最適化は、ソースとターゲットが同じデータベースにある場合や、アグリゲータトランスフォーメーションやフィルタトランスフォーメーションなどのトランスフォーメーションをソースデータベースで処理してデータの移動量を減らす場合に最適です。例えば、マッピングに Teradata ソースと Teradata ターゲットが含まれている場合、完全なプッシュダウンの最適化を設定して、処理するすべてのトランスフォーメーションロジックを Teradata ソースデータベースから Teradata ターゲットデータベースにプッシュします。

完全なプッシュダウン用にアップデートストラテジトランスフォーメーションを含むマッピングを設定する場合、マッピングのプッシュダウン互換性を判断する必要があります。

次の場合、データ統合サービスはアップデートストラテジトランスフォーメーションを含むマッピングをプッシュダウンできます。

- アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションが、異なるキーを持つ複数の行を受信する場合。
- アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションが、並べ替えできる同じキーを持つ複数の行を受信する場合。

次の場合、データ統合サービスはアップデートストラテジトランスフォーメーションを含むマッピングをプッシュダウンできません。

- アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションが、並べ替えできない同じキーを持つ複数の行を受信する場合。

マッピングでプッシュダウン互換性パラメータを使用することもできます。次のパラメータ値を使用できます。

- noMultipleRowsWithSameKeyOnTarget
- reorderAllowedForMultipleRowsWithSameKey
- reorderNotAllowedForRowsWithSameKey

データ統合サービスで完全なプッシュダウンの最適化を使用できるソースは次のとおりです。

- Amazon Redshift
- Greenplum
- IBM DB2
- Microsoft SQL Server
- Netezza
- Oracle
- SAP HANA
- Snowflake
- Teradata

## 完全なプッシュダウンの最適化に関するルールおよびガイドライン

完全なプッシュダウンの最適化を設定する場合は、以下のルールとガイドラインを考慮してください。

- データ統合サービスは、マッピング内のすべてのトランスフォーメーションロジックを IBM DB2、Microsoft SQL Server、Oracle、および ODBC ソース（Amazon Redshift、Greenplum、Netezza、Teradata、SAP HANA など）にプッシュできます。
- アップデートストラテジトランスフォーメーションを含むマッピングに対して完全なプッシュダウンの最適化を設定する場合、「更新または挿入」ストラテジは Amazon Redshift、Oracle、Teradata でのみ使用できます。

## ソースプッシュダウン

データ統合サービスは、ソースプッシュダウンを適用するときにソースからターゲットへのマッピングを分析します。これを行わないと、ダウンストリームトランスフォーメーションに達するまで、ソースデータベースにプッシュできません。

データ統合サービスは、データベースにプッシュできる各トランスフォーメーションのトランスフォーメーションロジックに基づいて、SELECT 文を生成、実行します。次に、統合サービスではこの SQL クエリの結果が読み込まれ、残りのトランスフォーメーションが処理されます。

ソースとターゲットが異なるデータベースに存在している場合、ソースプッシュダウンを使用するようにマッピングを設定できます。例えば、マッピングに Teradata ソースと Oracle ターゲットが含まれている場合、ソースプッシュダウンを設定して、処理する一部のトランスフォーメーションロジックを Teradata ソースにプッシュできます。

## プッシュダウンの設定

マッピングランタイムプロパティで、プッシュダウンの最適化のためのマッピングを設定できます。

1. マッピングを開きます。
2. **【プロパティ】** タブで、**【ランタイム】** を選択します。

3. プッシュダウンタイプを選択するか、プッシュダウンパラメータを割り当てます。
  - **なし**。データ統合サービスはマッピングロジックをソースデータベースにプッシュダウンしません。
  - **完全**。データ統合サービスはマッピングロジック全体をソースデータベースにプッシュダウンします。
  - **ソース**。データ統合サービスは、ターゲットを除くすべてのマッピングロジックをソースデータベースにプッシュダウンします。
  - **パラメータの割り当て**。プッシュダウンタイプに設定したパラメータを選択するか、新しいパラメータを作成して、**[OK]** をクリックします。
4. 完全なプッシュダウンの最適化を選択していて、マッピングにアップデートストラテジトランスフォーメーションが含まれている場合は、必要に応じて、プッシュダウン互換性オプションを選択するか、プッシュダウン互換性パラメータを割り当てることができます。
  - **複数の行に同じキーがない**。アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションは、同じキーを持つ複数の行を受け取ります。データ統合サービスは、アップデートストラテジトランスフォーメーションをプッシュダウンできます。
  - **同じキーの複数の行を並べ替えることができる**。アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションは、並べ替えが可能な同じキーの複数の行を受け取ります。データ統合サービスは、アップデートストラテジトランスフォーメーションをプッシュダウンできます。
  - **同じキーの複数の行を並べ替えることができない**。アップデートストラテジトランスフォーメーションに接続されているターゲットトランスフォーメーションは、並べ替えが不可能な同じキーの複数の行を受け取ります。データ統合サービスは、アップデートストラテジトランスフォーメーションをプッシュダウンできません。
  - **パラメータの割り当て**。プッシュダウン互換性に設定したパラメータを選択するか、パラメータを作成して、**[OK]** をクリックします。

## トランスフォーメーションプッシュロジック

データ統合サービスは、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ソースデータベースにプッシュします。データ統合サービスがソース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決まります。データ統合サービスは、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。

データ統合サービスは、以下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できます。

- アグリゲータ
- 式
- フィルタ
- ジョイナ
- ルックアップ
- ソータ
- 共有体

データ統合サービスは、以下の場合にソースにトランスフォーメーションロジックをプッシュできません。

- ソースがバイナリデータ型のカラムを含む場合。

- ソースは、フィルタ条件、または式またはジョイナトランスフォーメーションロジックのユーザー定義ジョインを含む、カスタマイズされたデータオブジェクトです。
- ソースは、別のデータベース管理システムにあるか、ジョイナまたは共有体トランスフォーメーションロジックに別の接続を使用しています。
- ルックアップ一致ポリシーが「すべての行を返す」に設定されていません。

データ統合サービスは、ソースにプッシュできないマッピングロジックを処理します。

## ソースへのプッシュダウンの最適化

データ統合サービスはトランスフォーメーションロジックを別のソース、例えばリレーショナルソースやデータベース固有の ODBC ドライバを使用するソースにプッシュできます。データ統合サービスがプッシュするトランスフォーメーションロジックのタイプは、ソースタイプによって異なります。

データ統合サービスは、トランスフォーメーションロジックを次のタイプのソースにプッシュできます。

- リレーショナルソース
- ネイティブのデータベースドライバを使用するソース
- PowerExchange®の非リレーショナルソース
- データベース固有の ODBC ドライバを使用するソース
- SAP ソース

## リレーショナルソースへのプッシュダウンの最適化

データ統合サービスは、ネイティブドライバまたはデータベース固有の ODBC ドライバを使用して、トランスフォーメーションロジックをリレーショナルソースにプッシュできます。

データ統合サービスは、アグリゲータ、式、フィルタ、ジョイナ、ソータ、および共有体トランスフォーメーションロジックを以下のリレーショナルソースにプッシュできます。

- Amazon Redshift
- Greenplum
- Hive
- IBM DB2
- Microsoft SQL Server
- Oracle
- SAP HANA
- Sybase
- Teradata

リレーショナルソースにアグリゲータトランスフォーメーションロジックをプッシュすると、パススルーポートがグループ別ポートの場合はパススルーポートが有効になります。トランスフォーメーション言語には、アグリゲータトランスフォーメーションで使用できる集計関数が含まれています。



以下の表では、IBM DB2 リレーショナルソースで有効な集計関数を示しています。

集計関数	DB2-LUW	DB2i	DB2z/os
AVG	○	○	○
COUNT	○	○	○
FIRST	×	×	×
LAST	×	×	×
MAX	○	○	○
MEDIAN	×	×	×
MIN	○	○	○
PERCENTILE	×	×	×
STDDEV	○	○	○
SUM	○	○	○
VARIANCE	○	○	○

以下の表では、Amazon Redshift、Greenplum、Hive、MSSQL、Oracle、Sybase、および Teradata リレーショナルソースで有効な集計関数を示しています。

集計関数	Amazon Redshift	Greenplum	Hive	MSSQL	Oracle	Sybase	Teradata
AVG	○	○	○	○	○	○	○
COUNT	○	○	○	○	○	○	○
FIRST	×	×	×	×	×	×	×
LAST	×	×	×	×	×	×	×
MAX	○	○	○	○	○	○	○
MEDIAN	×	×	×	×	○	×	×
MIN	○	○	○	○	○	○	○
PERCENTILE	×	×	×	×	×	×	×
STDDEV	○	○	○	○	○	×	○
SUM	○	○	○	○	○	○	○
VARIANCE	○	○	○	○	○	×	○

リレーショナルソースには NULL 値を扱うためのデフォルト設定があります。デフォルトでは、一部のデータベースは NULL 値を他の値よりも低く扱い、一部のデータベースは NULL 値を他の値よりも高く扱います。ソ

ースにデフォルトの NULL の順序付けがあるときは、ソータートランスフォーメーションロジックをリレーショナルソースにプッシュして正確な結果を得ることができます。

重複しない出力行にソータートランスフォーメーションを設定すると、大文字と小文字を区別するソートを有効にしてトランスフォーメーションロジックを DB2、Sybase、および Oracle のソースにプッシュする必要があります。

データ統合サービスは、Decimal データ型を含む関数を Hive ソースにプッシュすることができません。

## ネイティブソースへのプッシュダウンの最適化

ネイティブドライバを使用してトランスフォーメーションロジックをリレーショナルソースにプッシュするときに、データ統合サービスはネイティブデータベース SQL を使用する SQL 文を生成します。

データ統合サービスは、アグリゲータ、式、フィルタ、ジョイナ、ソータ、および共有体の各トランスフォーメーションロジックを、次のネイティブソースにプッシュできます。

- IBM DB2 for Linux、UNIX、および Windows (「DB2 for LUW」)
- Microsoft SQL Server。データ統合サービスは、Windows 上で実行される場合、Microsoft SQL Server へのネイティブ接続を使用できます。
- Oracle

データ統合サービスは、フィルタトランスフォーメーションロジックを次のネイティブソースにプッシュできます。

- IBM DB2 for i5/OS
- IBM DB2 for z/OS

## PowerExchange 非リレーショナルソースへのプッシュダウンの最適化

z/OS システム上の PowerExchange 非リレーショナルデータソースの場合、Data Integration Service はフィルタトランスフォーメーションロジックを PowerExchange にプッシュします。PowerExchange は、ソースが処理できるクエリにロジックを変換します。

Data Integration Service は、フィルタトランスフォーメーションロジックを次のタイプの非リレーショナルソースにプッシュできます。

- IBM IMS
- シーケンシャルデータセット
- VSAM

## ODBC ソースへのプッシュダウンの最適化

データ統合サービスは、トランスフォーメーションロジックを、データベース固有の ODBC ドライバを使用するデータベースにプッシュできます。**【その他】**として ODBC プロバイダを選択した場合、データ統合サービスは、トランスフォーメーションロジックをソースにプッシュできません。

データベース固有の ODBC ドライバを使用してソースに接続する場合、データ統合サービスはネイティブデータベース SQL を使用して SQL 文を生成します。

ODBC プロバイダは、ODBC 接続オブジェクトで指定できます。

以下の ODBC 接続タイプオブジェクトに対して、特定の ODBC プロバイダを設定できます。

- Amazon Redshift

- Greenplum
- Microsoft SQL Server
- Netezza
- SAP HANA
- Snowflake
- Sybase ASE
- Teradata

## SAP ソースへのプッシュダウンの最適化

Data Integration Service は、カラム名、演算子、リテラル文字列を含む式について、フィルタトランスフォーメーションロジックを SAP ソースにプッシュできます。Data Integration Service がトランスフォーメーションロジックを SAP にプッシュする際、式のリテラル文字列は SAP のデータ型に変換されます。

Data Integration Service は、TO\_DATE が DATS、TIMS、または ACCP データ型の文字列を以下のいずれかの日付フォーマットに変換する場合、TO\_DATE 関数を含むフィルタトランスフォーメーションロジックをプッシュできます。

- 'MM/DD/YYYY'
- 'YYYY/MM/DD'
- 'YYYY-MM-DD HH24:MI:SS'
- 'YYYY/MM/DD HH24:MI:SS'
- 'MM/DD/YYYY HH24:MI:SS'

Data Integration Service は、TO\_DATE 関数が DATS、TIMS、または ACCP 以外のデータ型に適用される場合、または、Data Integration Services が SAP にプッシュできないフォーマットに TO\_DATE が文字列が変換される場合、そのトランスフォーメーションロジックを処理します。Data Integration Service は、TO\_DATE 以外の Informatica 関数を含むトランスフォーメーションロジックを処理します。Data Integration Service は、TO\_DATE 以外の Informatica 関数を含むトランスフォーメーションロジックを処理します。

フィルタトランスフォーメーション式には、AND または OR で区切って複数の条件を指定できます。条件が複数の SAP テーブルに適用される場合、Data Integration Service は、SAP データオブジェクトで Open SQL ABAP ジョイン構文が使用されているときに、トランスフォーメーションロジックを SAP にプッシュできません。SAP データオブジェクトの読み取り操作で Select 構文モードを設定します。

## SAP データ型の例外

ソースがフィルタトランスフォーメーションロジックを処理できず、トランスフォーメーション式に以下のデータ型が含まれる場合、データ統合サービスがフィルタトランスフォーメーションロジックを処理します。

- RAW
- LRAW
- LCHR

# プッシュダウンの最適化の式

トランスフォーメーションにソースがサポートする演算子と関数が含まれる場合、Data Integration Service 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にプッシュできます。Data Integration Service は、データベース内の対応する演算子および関数を判断することによって、トランスフォーメーション式をクエリに変換します。対応する演算子または関数が存在しない場合、Data Integration Service がトランスフォーメーションロジックを処理します。

ソースで ODBC 接続が使用されていて、ODBC 接続オブジェクト内にデータベース固有の ODBC プロバイダを設定した場合、Data Integration Service はそのソースをネイティブソースタイプと見なします。

## 関数

Informatica 関数は、z/OS の非リレーショナルソースには使用できません。以下の表では、IBM DB2 ソースのプッシュダウンの最適化に使用できる Informatica 関数を示しています。

関数	DB2 for i5/OS <sup>1</sup>	DB2 for LUW	DB2 for z/OS <sup>1</sup>
ABORT ()	×	×	×
ABS()	×	○	×
ADD_TO_DATE()	○	○	○
AES_DECRYPT()	×	×	×
AES_ENCRYPT()	×	×	×
ASCII()	○	○	○
AVG()	○	○	○
CEIL()	○	○	○
CHOOSE()	×	×	×
CHR()	×	○	×
CHRCODE()	×	○	○
COMPRESS()	×	×	×
CONCAT()	○	○	○
COS()	○	○	○
COSH()	○	○	○
COUNT()	○	○	○
CRC32()	×	×	×
CREATE_TIMESTAMP_TZ()	×	×	×
CUME()	×	×	×

関数	DB2 for i5/OS <sup>1</sup>	DB2 for LUW	DB2 for z/OS <sup>1</sup>
DATE_COMPARE()	○	○	○
DATE_DIFF()	×	×	×
DECODE()	×	○	×
DECODE_BASE64()	×	×	×
DECOMPRESS()	×	×	×
ENCODE_BASE64()	×	×	×
ERROR()	×	×	×
EXP()	×	○	×
FIRST()	×	×	×
FLOOR()	×	○	×
FV()	×	×	×
GET_DATE_PART()	○	○	○
GET_TIMESTAMP()	×	×	×
GET_TIMEZONE()	×	×	×
GREATEST()	×	×	×
IIF()	×	○	×
IN()	×	○	×
INDEXOF()	×	×	×
INITCAP()	×	×	×
INSTR()	○	○	○
IS_DATE()	×	×	×
IS_NUMBER()	×	×	×
IS_SPACES()	×	×	×
ISNULL()	○	○	○
LAST()	×	×	×
LAST_DAY()	×	×	×
LEAST()	×	×	×

関数	DB2 for i5/OS <sup>1</sup>	DB2 for LUW	DB2 for z/OS <sup>1</sup>
LENGTH()	○	○	○
LN()	○	○	○
LOG()	○	○	○
LOWER()	○	○	○
LPAD()	×	×	×
LTRIM()	○	○	○
MAKE_DATE_TIME()	×	×	×
MAX()	○	○	○
MD5()	×	×	×
MEDIAN()	×	×	×
METAPHONE()	×	×	×
MIN()	○	○	○
MOD()	○	○	○
MOVINGAVG()	×	×	×
MOVINGSUM()	×	×	×
NPER()	×	×	×
PERCENTILE()	×	×	×
PMT()	×	×	×
POWER()	○	○	○
PV()	×	×	×
RAND()	×	×	×
RATE()	×	×	×
REG_EXTRACT()	×	×	×
REG_MATCH()	×	×	×
REG_REPLACE	×	×	×
REPLACECHR()	×	×	×
REPLACESTR()	×	×	×

関数	DB2 for i5/OS <sup>1</sup>	DB2 for LUW	DB2 for z/OS <sup>1</sup>
REVERSE()	×	×	×
ROUND(DATE)	×	×	○
ROUND(NUMBER)	○	○	○
RPAD()	×	×	×
RTRIM()	○	○	○
SET_DATE_PART()	×	×	×
SIGN()	○	○	○
SIN()	○	○	○
SINH()	○	○	○
SOUNDEX()	×	○ <sup>1</sup>	×
SQRT()	×	○	×
STDDEV()	○	○	○
SUBSTR()	○	○	○
SUM()	○	○	○
SYSTIMESTAMP()	○	○	○
TAN()	○	○	○
TANH()	○	○	○
TO_BIGINT	○	○	○
TO_CHAR(DATE)	○	○	○
TO_CHAR(NUMBER)	○	○ <sup>2</sup>	○
TO_DATE()	○	○	○
TO_DECIMAL()	○	○ <sup>3</sup>	○
TO_DECIMAL38()	×	×	×
TO_FLOAT()	○	○	○
TO_INTEGER()	○	○	○
TO_TIMESTAMP_TZ()	×	×	×
TRUNC(DATE)	×	×	×

関数	DB2 for i5/OS <sup>1</sup>	DB2 for LUW	DB2 for z/OS <sup>1</sup>
TRUNC(NUMBER)	○	○	○
UPPER()	○	○	○
VARIANCE()	○	○	○

<sup>1</sup> データ統合サービスは、フィルタトランスフォーメーションロジックに含まれる場合にのみこれらの関数をプッシュできます。

<sup>2</sup> この関数が decimal または float の引数を取り、フィルタトランスフォーメーションロジックに含まれる場合にのみ、データ統合サービスはこの関数をプッシュできます。

<sup>3</sup> この関数が string の引数を取り、フィルタトランスフォーメーションロジックに含まれる場合にのみ、データ統合サービスはこの関数をプッシュできます。

次の表では、Amazon Redshift、Hive、Microsoft SQL Server、Oracle、SAP、SAP HANA、Snowflake、および Sybase ASE ソースのプッシュダウンの最適化に使用できる Informatica 関数を示しています。

関数	Amazon Redshift	Hive	Microsoft SQL Server	Oracle	SAP <sup>1</sup>	SAP HANA	Snowflake	Sybase ASE
ABORT ()	×	×	×	×	×	×	×	×
ABS()	○	○	○	○	×	○	○	○
ADD_TO_DATE()	○	×	○	○	×	×	○	○
AES_DECRYPT()	×	×	×	×	×	×	×	×
AES_ENCRYPT()	×	×	×	×	×	×	×	×
ASCII()	×	×	○	○	×	×	○	○
AVG()	○	○	○	○	×	○	○	○
CEIL()	○	○	○	○	×	○	○	○
CHOOSE()	×	×	×	×	×	×	×	×
CHR()	○	×	○	○	×	○	○	○
CHRCODE()	×	×	○	○	×	○	○	○
COMPRESS()	×	×	×	×	×	×	×	×
CONCAT()	○	○	○	○	×	○	○	○
COS()	○	○	○	○	×	○	×	○
COSH()	×	×	○	○	×	○	○	○
COUNT()	○	○	○	○	×	○	×	○
CRC32()	×	×	×	×	×	×	×	×



関数	Amazon Redshift	Hive	Microsoft SQL Server	Oracle	SAP <sup>1</sup>	SAP HANA	Snowflake	Sybase ASE
CREATE_TIMESTAMP_TZ()	×	×	×	○	×	×	×	×
CUME()	×	×	○	×	×	×	×	×
DATE_COMPARE()	○	×	○	○	×	○	×	○
DATE_DIFF()	○	×	×	×	×	○	○	×
DECODE()	○	○	○	○	×	○	○	○
DECODE_BASE64()	×	×	×	×	×	×	×	×
DECOMPRESS()	×	×	×	×	×	×	×	×
ENCODE_BASE64()	×	×	×	×	×	×	×	×
ERROR()	×	×	×	×	×	×	×	×
EXP()	○	○	○	○	×	×	○	○
FIRST()	×	×	×	×	×	×	○	×
FLOOR()	○	○	○	○	×	○	○	○
FV()	×	×	×	×	×	×	×	×
GET_DATE_PART()	○	×	○	○	×	○	○	○
GET_TIMESTAMP()	×	×	×	○	×	×	×	×
GET_TIMEZONE()	×	×	×	×	×	×	×	×
GREATEST()	×	×	×	○	×	×	×	×
IIF()	○	○	○	○	×	○	×	○
IN()	○	×	○	○	×	×	×	○
INDEXOF()	×	×	×	×	×	×	×	×
INITCAP()	○	×	×	○	×	×	○	×
INSTR()	○	×	○	○	×	×	×	○
IS_DATE()	×	×	×	×	×	×	×	×

関数	Amazon Redshift	Hive	Microsoft SQL Server	Oracle	SAP <sup>1</sup>	SAP HANA	Snowflake	Sybase ASE
IS_NUMBER()	×	×	×	×	×	×	×	×
IS_SPACES()	×	×	×	×	×	×	×	×
ISNULL()	○	○	○	○	×	○	×	○
LAST()	×	×	×	×	×	×	×	×
LAST_DAY()	○	×	×	○	×	○	×	×
LEAST()	×	×	×	○	×	×	×	×
LENGTH()	○	○	○	○	×	○	○	○
LN()	○	○	○	○	×	○	○	○
LOG()	×	○	○	○	×	○	○	○
LOWER()	○	○	○	○	×	○	○	○
LPAD()	○	○	×	○	×	○	○	×
LTRIM()	○	○	○	○	×	○	○	×
MAKE_DATE_TIME()	×	×	×	×	×	×	×	×
MAX()	○	○	○	○	×	○	○	○
MD5()	×	×	×	×	×	×	×	×
MEDIAN()	×	×	×	○	×	×	○	×
METAPHONE()	×	×	×	×	×	×	×	×
MIN()	○	○	○	○	×	○	○	○
MOD()	○	○	○	○	×	○	○	○
MOVINGAVG()	×	×	×	×	×	×	×	×
MOVINGSUM()	×	×	×	×	×	×	×	×
NPER()	×	×	×	×	×	×	×	×
PERCENTILE()	×	×	×	×	×	×	×	×
PMT()	×	×	×	×	×	×	×	×
POWER()	○	○	○	○	×	○	○	○
PV()	×	×	×	×	×	×	×	×

関数	Amazon Redshift	Hive	Microsoft SQL Server	Oracle	SAP <sup>1</sup>	SAP HANA	Snowflake	Sybase ASE
RAND()	×	×	×	×	×	×	×	×
RATE()	×	×	×	×	×	×	×	×
REG_EXTRACT()	×	×	×	×	×	×	×	×
REG_MATCH()	×	×	×	×	×	×	×	×
REG_REPLACE	×	×	×	×	×	×	×	×
REPLACECHR()	×	×	×	×	×	×	×	×
REPLACESTR()	×	×	×	×	×	×	×	×
REVERSE()	×	×	×	×	×	×	×	×
ROUND(DATE)	×	×	×	○	×	×	○	×
ROUND(NUMBER)	○	○	○	○	×	○	○	○
RPAD()	○	○	×	○	×	○	○	×
RTRIM()	○	○	○	○	×	○	○	○
SET_DATE_PART()	×	×	×	×	×	×	×	×
SIGN()	○	×	○	○	×	○	○	○
SIN()	○	○	○	○	×	○	×	○
SINH()	×	×	○	○	×	○	○	○
SOUNDEX()	×	×	○	○	×	×	×	○
SQRT()	○	○	○	○	×	○	○	○
STDDEV()	○	×	○	○	×	○	○	○
SUBSTR()	○	○	○	○	×	○	○	○
SUM()	○	○	○	○	×	○	×	○
SYSTIMESTAMP()	○	×	○	○	×	○ <sup>2</sup>	×	○
TAN()	○	×	○	○	×	○	×	○
TANH()	×	×	○	○	×	×	○	○
TO_BIGINT	○	○	○	○	×	○	○	○

関数	Amazon Redshift	Hive	Microsoft SQL Server	Oracle	SAP <sup>1</sup>	SAP HANA	Snowflake	Sybase ASE
TO_CHAR(DATE)	○	×	○	○	×	○	○	○
TO_CHAR(NUMBER)	○	×	○	○	×	○	○	○
TO_DATE()	○	×	○	○	○	○	○	○
TO_DECIMAL()	○	×	○	○	×	○	○	○
TO_DECIMAL38()	○	○	○	○	○	○	○	○
TO_FLOAT()	○	×	○	○	×	○	○	○
TO_INTEGER()	○	○	○	○	×	○	○	○
TO_TIMESTAMP_TZ()	×	×	×	○	×	×	×	×
TRUNC(DATE)	○	×	×	○	×	○	○	×
TRUNC(NUMBER)	○	×	○	○	×	○	○	○
UPPER()	○	○	○	○	×	○	○	○
VARIANCE()	○	×	○	○	×	×	○	○

<sup>1</sup>. データ統合サービスは、これらの関数がフィルタトランスフォーメーションロジックに含まれる場合にのみ、これらをソースにプッシュできます。

<sup>2</sup>. SYSTIMESTAMP() は SS 引数のみをサポートしています。

次の表では、Greenplum、Netezza、および Teradata ソースのプッシュダウンの最適化に使用できる Informatica 関数を示しています。

関数	Greenplum	Netezza	Teradata
ABORT ()	×	×	いいえ
ABS()	○	○	はい
ADD_TO_DATE()	○	○	はい
AES_DECRYPT()	×	×	いいえ
AES_ENCRYPT()	×	×	いいえ
ASCII()	○	はい	×
AVG()	○	○	はい
CEIL()	○	○	はい

関数	Greenplum	Netezza	Teradata
CHOOSE()	×	×	いいえ
CHR()	○	はい	×
CHRCODE()	○	はい	×
COMPRESS()	×	×	いいえ
CONCAT()	○	○	はい
COS()	○	○	はい
COSH()	○	○	はい
COUNT()	○	○	はい
CRC32()	×	×	いいえ
CREATE_TIMESTAMP_TZ()	×	×	いいえ
CUME()	×	×	いいえ
DATE_COMPARE()	○	○	はい
DATE_DIFF()	×	×	いいえ
DECODE()	○	○	はい
DECODE_BASE64()	×	×	いいえ
DECOMPRESS()	×	×	いいえ
ENCODE_BASE64()	×	×	いいえ
ERROR()	×	×	いいえ
EXP()	○	○	はい
FIRST()	×	×	いいえ
FLOOR()	○	○	はい
FV()	×	×	いいえ
GET_DATE_PART()	○	○	はい
GET_TIMESTAMP()	×	×	いいえ
GET_TIMEZONE()	×	×	いいえ
GREATEST()	×	×	いいえ
IIF()	○	○	はい

関数	Greenplum	Netezza	Teradata
IN()	×	×	○
INDEXOF()	×	×	いいえ
INITCAP()	○	はい	×
INSTR()	×	○	はい
IS_DATE()	×	×	いいえ
IS_NUMBER()	×	×	いいえ
IS_SPACES()	×	×	いいえ
ISNULL()	×	×	○
LAST()	×	×	いいえ
LAST_DAY()	×	○	×
LEAST()	×	×	いいえ
LENGTH()	○	○	はい
LN()	○	×	○
LOG()	○	○	はい
LOWER()	○	○	はい
LPAD()	○	はい	×
LTRIM()	○	○	はい
MAKE_DATE_TIME()	×	×	いいえ
MAX()	○	○	はい
MD5()	×	×	いいえ
MEDIAN()	×	×	いいえ
METAPHONE()	×	×	いいえ
MIN()	○	○	はい
MOD()	○	○	はい
MOVINGAVG()	×	×	いいえ
MOVINGSUM()	×	×	いいえ
NPER()	×	×	いいえ

関数	Greenplum	Netezza	Teradata
PERCENTILE()	×	×	いいえ
PMT()	×	×	いいえ
POWER()	○	○	はい
PV()	×	×	いいえ
RAND()	×	×	いいえ
RATE()	×	×	いいえ
REG_EXTRACT()	×	×	いいえ
REG_MATCH()	×	×	いいえ
REG_REPLACE	×	×	いいえ
REPLACECHR()	×	×	いいえ
REPLACESTR()	×	×	いいえ
REVERSE()	×	×	いいえ
ROUND(DATE)	×	×	いいえ
ROUND(NUMBER)	○	○	はい
RPAD()	○	はい	×
RTRIM()	○	○	はい
SET_DATE_PART()	×	×	いいえ
SIGN()	○	○	はい
SIN()	○	○	はい
SINH()	○	○	はい
SOUNDEX()	×	×	いいえ
SQRT()	○	○	はい
STDDEV()	○	○	はい
SUBSTR()	○	○	はい
SUM()	○	○	はい
SYSTIMESTAMP()	○	はい	×
TAN()	○	○	はい

関数	Greenplum	Netezza	Teradata
TANH()	○	○	はい
TO_BIGINT	○	○	はい
TO_CHAR(DATE)	○	○	はい
TO_CHAR(NUMBER)	○	○	はい
TO_DATE()	○	○	はい
TO_DECIMAL()	○	○	はい
TO_DECIMAL38()	○	○	はい
TO_FLOAT()	○	○	はい
TO_INTEGER()	○	○	はい
TO_TIMESTAMP_TZ()	×	×	いいえ
TRUNC(DATE)	○	はい	×
TRUNC(NUMBER)	○	○	はい
UPPER()	○	○	はい
VARIANCE()	○	○	はい

## Amazon Redshift 関数の例外

特定の条件下において、データ統合サービスは、サポートされている関数を Amazon Redshift データベースにプッシュできません。

- プッシュダウン最適化を使用するには、**【複数の一致】** ルックアップトランスフォーメーションプロパティの値を **【すべての行を返す】** に設定する必要があります。
- TRUNC(DATE)を Amazon Redshift にプッシュするには、日付および形式引数を定義する必要があります。定義しないと、エージェントはこの関数を Amazon Redshift にプッシュしません。
- Amazon Redshift の集計関数が受け取る引数は 1 つだけで、それは集計関数のフィールドセットです。フィルタ条件の引数は無視されます。また、ターゲットにマップされているすべてのフィールドが GROUP BY 句に指定されていることを確認します。
- TO\_DATE()を Amazon Redshift にプッシュするには、文字列および形式引数を定義する必要があります。
- TO\_CHAR()を Amazon Redshift にプッシュするには、日付および形式引数を定義する必要があります。
- SYSTIMESTAMP()に形式を指定して、SYSTIMESTAMP を Amazon Redshift にプッシュしないでください。Amazon Redshift データベースは完全なタイムスタンプを返します。
- INSTR()を Amazon Redshift にプッシュするには、文字列、search\_value、および開始引数のみを定義します。Amazon Redshift では、オカレンスおよび comparison\_type 引数はサポートされていません。
- TO\_BIGINT および TO\_INTEGER を Amazon Redshift にプッシュすると、フラグ引数は無視されます。
- IN()を Amazon Redshift にプッシュすると、CaseFlag 引数は無視されます。



- ADD\_TO\_DATE()関数の一部として NS 形式を使用する場合、エージェントではこの関数が Amazon Redshift にプッシュされません。
- 次の形式のいずれかを TO\_CHAR()および TO\_DATE()関数の一部として使用する場合、エージェントではこれらの関数が Amazon Redshift にプッシュされません。
  - NS
  - SSSS
  - SSSSS
  - RR
- TRUNC(DATE)、GET\_DATE\_PART()、および DATE\_DIFF()を Amazon Redshift にプッシュするには、次の形式を使用する必要があります。
  - D
  - DDD
  - HH24
  - MI
  - MM
  - MS
  - SS
  - US
  - YYYY

## Hive 関数の例外

特定の条件下において、データ統合サービスは、サポートされている関数を Hive ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、データ統合サービスは、Hive ソースのトランスフォーメーションロジックを処理します。

- 2 番目の引数として、LTRIM にスペースが含まれる。
- 2 番目の引数として、RTRIM にスペースが含まれる。

以下の関数を date データ型とともに使用する場合、データ統合サービスでは Hive ソースのトランスフォーメーションロジックを処理できません。

- CONCAT
- MAX
- MIN
- ROUND
- TO\_BIGINIT
- TO\_INTEGER

## IBM DB2 関数の例外

特定の条件下において、Data Integration Service は、サポートされている関数を IBM DB2 for i5/OS、DB2 for LUW、および DB2 for z/OS ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、Data Integration Service は、IBM DB2 ソースのトランスフォーメーションロジックを処理します。

- ADD\_TO\_DATE または GET\_DATE\_PART が、ミリ秒またはナノ秒の精度で結果を返す場合。
- LTRIM が 2 つ以上の引数を含む場合。
- RTRIM が 2 つ以上の引数を含む場合。
- TO\_BIGINT が、文字列を DB2 for LUW ソースの Bigint 値に変換される場合。
- TO\_CHAR が日付を文字列値に変換し、DB2 でサポートされていない形式を指定する場合。
- TO\_DATE が文字列値を日付に変換し、DB2 でサポートされていない形式を指定する場合。
- TO\_DECIMAL が scale 引数なしで文字列を decimal 値に変換する場合。
- TO\_FLOAT が文字列を倍精度浮動小数点数に変換する場合。
- TO\_INTEGER が文字列を DB2 for LUW ソースの整数値に変換する場合。

## Microsoft SQL Server 関数の例外

特定の条件下において、Data Integration Service はサポートされている関数を Microsoft SQL Server ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、Data Integration Service は、Microsoft SQL Server ソースのトランスフォーメーションロジックを処理します。

- IN が CaseFlag 引数を含む場合。
- INSTR が 4 つ以上の引数を含む場合。
- LTRIM が 2 つ以上の引数を含む場合。
- RTRIM が 2 つ以上の引数を含む場合。
- TO\_BIGINT が 2 つ以上の引数を含む場合。
- TO\_INTEGER が 2 つ以上の引数を含む場合。

## Netezza 関数の例外

特定の条件下において、データ統合サービスは、サポートされている関数を Netezza ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、データ統合サービスは、Netezza ソースのトランスフォーメーションロジックを処理します。

- SYSTIMESTAMP は、YYYY-MM-DD HH24:MI:SS.US 形式の日付を含みます。
- TO\_CHAR(DATE)と TO\_DATE()は、サブ秒精度の YYYY-MM-DD HH24:MI:SS.US 形式の日付を含みます。

## Oracle 関数の例外

特定の条件下において、Data Integration Service は、サポートされている関数を Oracle ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、Data Integration Service は、Oracle ソースのトランスフォーメーションロジックを処理します。

- ADD\_TO\_DATE または GET\_DATE\_PART が、サブ秒の精度で結果を返す場合。
- ROUND が、秒またはサブ秒に値を丸める場合。
- SYSTIMESTAMP がミリ秒の精度で日付と時刻を返す場合。
- TRUNC が秒またはサブ秒を切り詰める場合。

## ODBC 関数の例外

IN 関数の CaseFlag 引数がゼロ以外の値の場合、データ統合サービスは、ODBC のトランスフォーメーションロジックを処理します。

**注:** ODBC 接続オブジェクトのプロパティがデータベース固有の ODBC プロバイダを含む場合、データ統合サービスはソースをネイティブソースタイプと見なします。

接続オブジェクトで ODBC プロバイダを **【その他】** に指定した場合、データ統合サービスは EXP()関数を Teradata ソースにプッシュできません。EXP()関数をプッシュするには、ODBC プロバイダを **【Teradata】** に設定してください。

## Sybase ASE 関数の例外

特定の条件下において、Data Integration Service は、サポートされている関数を Sybase ASE ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、Data Integration Service は、Sybase ASE ソースのトランスフォーメーションロジックを処理します。

- IN が CaseFlag 引数を含む場合。
- INSTR が 3 つ以上の引数を含む場合。
- LTRIM が 2 つ以上の引数を含む場合。
- RTRIM が 2 つ以上の引数を含む場合。
- TO\_BIGINT が 2 つ以上の引数を含む場合。
- TO\_INTEGER が 2 つ以上の引数を含む場合。
- TRUNC(Numbers)が 2 つ以上の引数を含む場合。

## Teradata 関数の例外

特定の条件下において、データ統合サービスは、サポートされている関数を Teradata ソースにプッシュできません。

サポートされている関数が以下のロジックで式に含まれる場合、データ統合サービスは、Teradata ソースのトランスフォーメーションロジックを処理します。

- ADD\_TO\_DATE が、YEAR および MONTH 以外の属性を含む場合。
- IN が CaseFlag 引数を含む場合。
- INSTR が 3 つ以上の引数を含む場合。

- LTRIM が 2 つ以上の引数を含む場合。
- ROUND が 2 つ以上の引数を含む場合。
- RTRIM が 2 つ以上の引数を含む場合。

## 演算子

以下の表に、Informatica の演算子を使用できるかどうかをソースタイプ別に示します。各カラムには、データ統合サービスが演算子をソースにプッシュできるかどうかを示されます。

**注:** 非リレーショナルソースは、z/OS 上の IMS、VSAM、およびシーケンシャルデータセットです。

演算子	DB 2 for LUW	DB2 for i5/OS または z/OS	Amazon Redshift	Greenplum	Hive	Microsoft SQL Server	非リレーショナル*	Oracle	SAP*	SAP HANA	Sybase ASE	Teradata
+ - *	○	○	○	○	○	○	○	○	×	○	○	○
/	○	○	○	○	○	○	×	○	×	○	○	○
%	○	○	○	○	○	○	×	○	×	○	○	○
	○	○	○	○	○	○	×	○	×	○	○	○
= > < >= <=	○	○	○	○	○	○	○	○	○	○	○	○
<>	○	○	○	○	○	○	×	○	○	○	○	○
!=	○	○	○	○	○	○	○	○	○	○	○	○
^=	○	○	○	○	○	○	×	○	○	○	○	○
AND OR	○	○	○	○	○	○	○	○	○	○	○	○
NOT	○	○	○	○	○	○	×	○	×	○	○	○

\* データ統合サービスは、フィルタトランスフォーメーションロジックに含まれる場合にのみこれらの演算子をプッシュできます。

# データ統合サービスとソースの出力の比較

データ統合サービスとソースでは、同じトランスフォーメーションロジックを処理した場合でも、異なる結果になる場合があります。データ統合サービスがトランスフォーメーションロジックをソースにプッシュした場合、トランスフォーメーションロジックの出力は異なる可能性があります。

トランスフォーメーションロジックの出力が異なる可能性があるのは、次のような場合です。

## 大文字小文字の区別

大文字小文字の区別の取り扱い方は、データ統合サービスとデータベースとで異なる場合があります。例えば、データ統合サービスは大文字小文字を区別するクエリが使用し、データベースは使用しない場合があります。フィルタトランスフォーメーションでフィルタ条件として `IIF(col_varchar2 = 'CA', TRUE, FALSE)` が使用されているとします。'CA' に一致する行を返すデータベースが必要です。ただし、大文字と小文字を区別しないデータベースにこのトランスフォーメーションロジックをプッシュすると、値 'Ca'、'ca'、'cA'、および 'CA' に一致する行が返されます。

## 数値が文字値に変換される

同じ数値を文字値に変換するにしても、データ統合サービスとデータベースとで変換形式が異なる可能性があります。データベースでは、数値を許容できない文字形式に変換されてしまう可能性があります。例えば、テーブルに数値 1234567890 が含まれているとします。データ統合サービスでこの数を文字値に変換した場合、文字「1234567890」が挿入されます。ただし、この数はデータベースで、「1.2E9」に変換されます。これら 2 つの文字列は同じ値を示しています。

## TO\_CHAR 関数および TO\_DATE 関数の日付フォーマット

データ統合サービスは、TO\_CHAR 関数または TO\_DATE 関数をデータベースにプッシュする場合、関数で日付フォーマットが使用されます。日付または時刻の値を比較するには、TO\_DATE 関数を使用します。TO\_CHAR を使用して日付または時刻の値を比較する場合、1 桁の月、日、または 1 時間などの値に、スペースまたは先頭のゼロが追加されます。データベースがスペースまたは先頭のゼロを追加すると、データベースの比較結果はデータ統合サービスの結果と異なる可能性があります。

## 精度

データ統合サービスとデータベースとで、特定のデータ型の精度が異なる場合があります。トランスフォーメーションデータ型が使用するデフォルトの数値精度は、ネイティブデータ型とは異なる場合があります。データベースで使用されている精度がデータ統合サービスとは異なる場合、結果が変わる可能性があります。

## SYSTIMESTAMP 関数

SYSTIMESTAMP を使用した場合、データ統合サービスはサービスプロセスを実行するノードの現在の日付と時刻を返します。しかし、トランスフォーメーションロジックをデータベースに渡した場合、データベースは、データベースのホストマシンの現在の日付と時刻を返します。データベースをホストしているマシンのタイムゾーンが、データ統合サービスを実行するマシンのタイムゾーンと同じでない場合、結果に相違が生じることがあります。

SYSTIMESTAMP を IBM DB2 または Sybase ASE データベースにプッシュし、SYSTIMESTAMP のフォーマットを指定した場合、データベースではそのフォーマットは無視され、完全なタイムスタンプが返されます。

## LTRIM、RTRIM、または SOUNDEX 関数

LTRIM、RTRIM、または SOUNDEX をデータベースにプッシュした場合、引数 (') はそのデータベースでは NULL として扱われますが、データ統合サービスではスペースとして扱われます。

## Oracle ソース上の LAST\_DAY 関数

LAST\_DAY を Oracle にプッシュした場合、秒までの日付が Oracle によって返されます。入力日付にサブ秒が含まれている場合、Oracle は日付を秒まで切り詰めます。

## 第 15 章

# パーティション化されたマッピング

この章では、以下の項目について説明します。

- [パーティション化されたマッピングの概要, 278 ページ](#)
- [各パイプラインステージごとに 1 つのスレッド, 279 ページ](#)
- [各パイプラインステージごとに複数のスレッド, 280 ページ](#)
- [パーティション化されたフラットファイルソース, 282 ページ](#)
- [パーティション化されたリレーショナルソース, 283 ページ](#)
- [パーティション化されたフラットファイルターゲット, 285 ページ](#)
- [パーティション化されたリレーショナルターゲット, 289 ページ](#)
- [パーティション化されたトランスフォーメーション, 291 ページ](#)
- [パーティション化されたマッピングにおける順序の維持, 294 ページ](#)
- [マッピングの最大並行処理のオーバーライド, 295 ページ](#)
- [パーティション化されたマッピングのトラブルシューティング, 299 ページ](#)

## パーティション化されたマッピングの概要

パーティション化オプションがある場合は、管理者はデータ統合サービスによりマッピング実行時の並列処理を最大化することができます。並列処理を最大化すると、データ統合サービスによって基になるデータが動的にパーティションに分割され、すべてのパーティションが同時に処理されます。

マッピングが大規模なデータセットを処理する場合、または複雑な計算を実行するトランスフォーメーションを含む場合は、マッピングの処理に時間がかかり、データのスループットが低下する可能性があります。これらのマッピングでパーティション化を有効にする場合、データ統合サービスは追加のスレッドを使用してマッピングを処理し、これによりパフォーマンスが最適化される可能性があります。

パーティション化を有効にするには、管理者と開発者が以下のタスクを実行します。

**管理者は、データ統合サービスの最大並行処理を、Administrator ツールで 2 以上の値に設定します。**

最大並行処理は、単一のパイプラインステージを処理する並行スレッドの最大数を決定します。管理者は、マッピングを実行するノードで使用可能な CPU の数に基づいて【最大並行処理】プロパティ値を増やします。

必要に応じて、開発者は、マッピングの並行処理の最大値を Developer ツールで変更できます。

デフォルトでは、各マッピングの【最大並行処理】プロパティは【自動】に設定されています。各マッピングは、データ統合サービスに定義されている最大並行処理値を使用します。

開発者は、マッピングランタイムプロパティ内の最大並行処理値を変更することにより、特定のマッピングの最大値を定義できます。データ統合サービスとマッピングで設定されている最大並行処理の整数値が異なる場合、データ統合サービスではこれらの最小値が使用されます。

マッピングでパーティション化が無効になっている場合、データ統合サービスはマッピングをパイプラインステージに区切り、1つのスレッドを使用して各ステージを処理します。

マッピングでパーティション化が有効になっている場合、データ統合サービスは、複数のスレッドを使用して各マッピングパイプラインステージを処理します。

データ統合サービスでは、入力および出力として物理データを持つマッピングにパーティションを作成できます。データ統合サービスは複数のパーティションを使用してマッピング実行中に次のアクションを完了できます。

- フラットファイル、IBM DB2 for LUW、または Oracle ソースからの読み取り。
- トランスフォーメーションの実行。
- フラットファイル、IBM DB2 for LUW、または Oracle ターゲットへの書き込み。

## 各パイプラインステージごとに1つのスレッド

最大並列処理数が1に設定されている場合、パーティション化は無効になっています。データ統合サービスは、マッピングを複数のパイプラインステージに分割し、1つのスレッドを使用して各ステージを処理します。

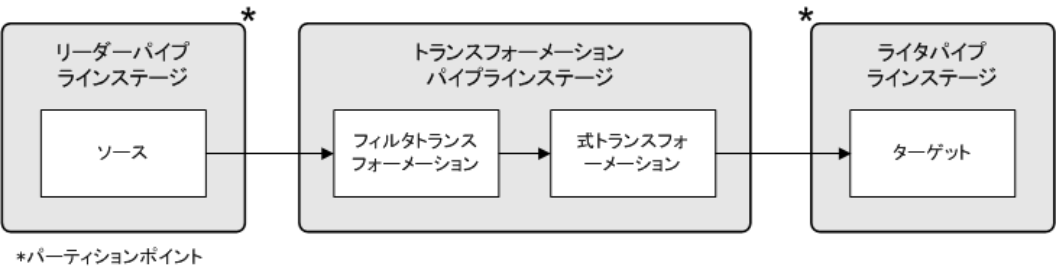
各マッピングには、1つ以上のパイプラインが含まれています。パイプラインは、読み取りトランスフォーメーションと、読み取りトランスフォーメーションからデータを受信するすべてのトランスフォーメーションで構成されています。データ統合サービスは、マッピングパイプラインをパイプラインステージに分けてから、各パイプラインステージの抽出、トランスフォーメーション、およびロードを並行して実行します。

パーティションポイントはパイプラインの境界をマークし、パイプラインをステージに分割します。どのマッピングパイプラインに対しても、データ統合サービスは、読み取りトランスフォーメーションの後、および書き込みトランスフォーメーションが複数のパイプラインステージを作成する前に、パーティションポイントを追加します。

各パイプラインステージは以下のいずれかのスレッドを実行します。

- データ統合サービスがソースからデータを抽出する方法を制御する読み取りスレッド。
- データ統合サービスがパイプラインのデータを処理する方法を制御するトランスフォーメーションスレッド。
- データ統合サービスがデータをターゲットにロードする方法を制御する書き込みスレッド。

以下の図に、読み取りパイプラインステージ、トランスフォーメーションパイプラインステージ、および書き込みパイプラインステージに区切られたマッピングを示します。



パイプラインには3つのステージが含まれるため、データ統合サービスでは3セットの行を同時に処理してマッピングのパフォーマンスを最適化することができます。例えば、読み取りスレッドが3番目の行セットを処理している間、トランスフォーメーションスレッドは2番目の行セットを処理し、書き込みスレッドは1番目の行セットを処理します。

以下の表に、複数のスレッドが3セットの行を同時に処理する方法を示します。

読み取りスレッド	トランスフォーメーションスレッド	書き込みスレッド
行セット 1	-	-
行セット 2	行セット 1	-
行セット 3	行セット 2	行セット 1
行セット 4	行セット 3	行セット 2
行セット n	行セット (n-1)	行セット (n-2)

マッピングパイプラインに複雑な計算を実行するトランスフォーメーションが含まれている場合、トランスフォーメーションパイプラインステージの処理には時間がかかる可能性があります。パフォーマンスを最適化するには、一部のトランスフォーメーションが追加のトランスフォーメーションパイプラインステージを作成する前に、データ統合サービスがパーティションポイントを追加します。

## 各パイプラインステージごとに複数のスレッド

最大並列処理が2以上の値に設定されている場合、パーティション化が有効になっています。データ統合サービスはマッピングをパイプラインステージに分割し、複数のスレッドを使用して各ステージを処理します。任意のパイプラインステージにおけるスレッドの数は、そのステージにおけるパーティションの数と一致します。

並行処理を最大にすると、データ統合サービスは実行時に以下のタスクを動的に実行します。

### データをパーティションに分割する。

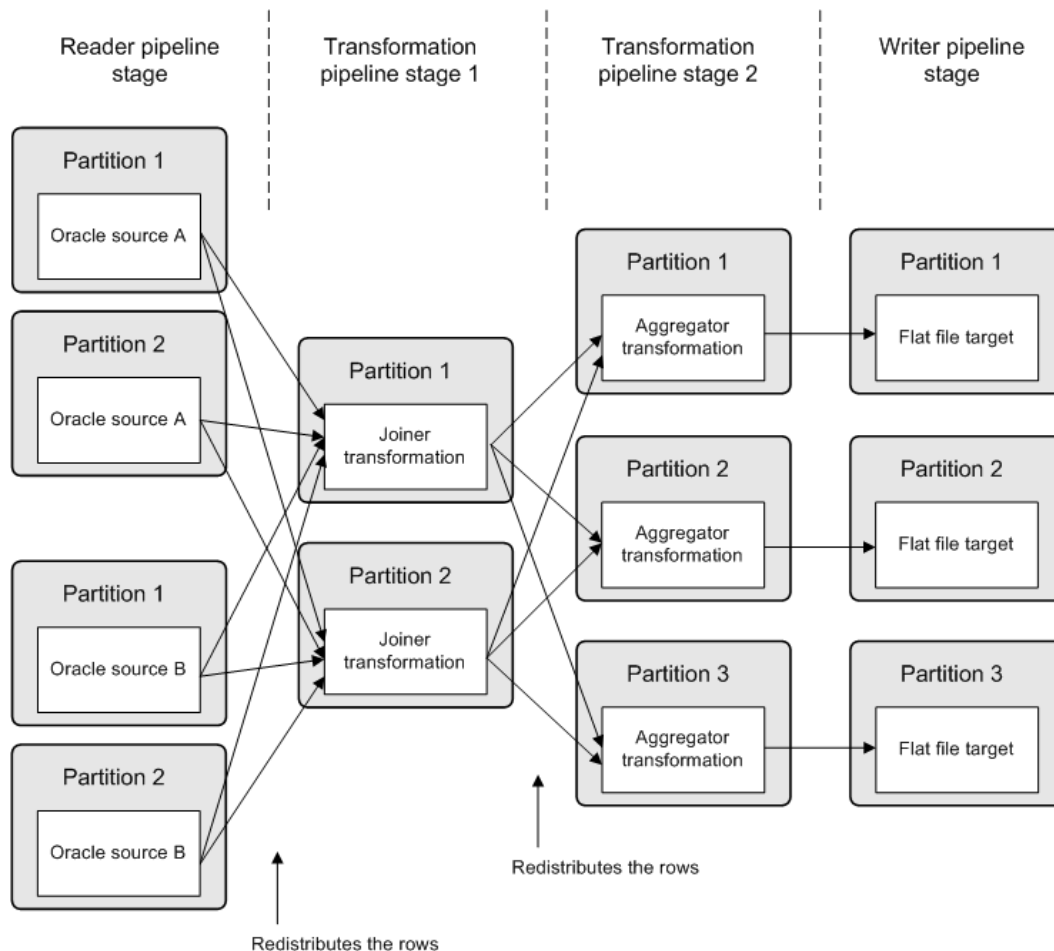
データ統合サービスは基になるデータを動的にパーティションに分割し、各パーティションを同時に実行します。データ統合サービスは、各パイプラインステージの最適なスレッド数を判断します。1つのパイプラインステージで使用されるスレッドの数は、最大並列処理数を超えることはできません。データ統合サービスは、パイプラインステージごとに異なる数のスレッドを使用できます。



パーティションポイント全体にわたってデータを再分散する。

データ統合サービスは、トランスフォーメーション要件に基づいて、パーティションポイント全体にわたってデータを再分配する最善の方法を動的に決定します。

以下の図に、パイプラインステージごとに複数のパーティションにデータを分配するマッピングの例を示します。



上図では、データ統合サービスの最大並列処理の値は3です。マッピングの最大並列処理は自動です。データ統合サービスは、マッピングを4つのパイプラインステージに分け、合計12個のスレッドを使用してマッピングを実行します。データ統合サービスは、各パイプラインステージで以下のタスクを実行します。

- 読み取りパイプラインステージでは、データ統合サービスは、Oracle データベースシステムに問い合わせ、ソース A とソース B の両方のソーステーブルに2つのデータベースパーティションが存在することを確認します。データ統合サービスは、各データベースパーティションについて1つの読み取りスレッドを使用します。
- 最初のトランスフォーメーションパイプラインステージで、データ統合サービスは、データを再配分して結合条件を満たす行を2つのスレッドに分割します。
- 2番目のトランスフォーメーションパイプラインステージで、データ統合サービスは、このアグリゲータトランスフォーメーションには3つのスレッドが最適であると判断します。そして、データを再配分して、集計式を満たす行を3つのスレッドに分割します。
- 書き込みパイプラインステージで、データ統合サービスは、ターゲットパーティションポイント全体にわたって行を再配分する必要がありません。単一パーティション内のすべての行は、ターゲットパーティションポイントを通じて、そのパーティション内に留まります。

# パーティション化されたフラットファイルソース

パーティション化が有効になっているマッピングでフラットファイルソースを読み取る場合、データ統合サービスは、複数のスレッドを使用してファイルソースを読み取ることができます。

データ統合サービスは、以下のフラットファイルソースタイプ用のパーティションを作成できます。

- 直接ファイル
- 間接ファイル
- ファイルのディレクトリ
- コマンド
- Hadoop 分散ファイルシステム（HDFS）のファイルまたはファイルのディレクトリ

データ統合サービスで複数のスレッドを使用してファイルソースを読み取る場合、そのソースに対して複数の同時接続が作成されます。デフォルトでは、データ統合サービスによって行の順番が保持されることはありません。ファイルやファイルリストの行を順番に読み取ることがないためです。複数のスレッドで1つのファイルソースを読み取る場合に行順序を保持するには、同時読み取りのパーティション化を設定します。

データ統合サービスで複数のスレッドを使用して直接ファイルを読み取る場合は、複数の読み取りスレッドが作成され、ファイルが同時に読み取られます。

データ統合サービスで、複数のスレッドを使用して間接ファイルまたはファイルのディレクトリを読み取る場合は、複数の読み取りスレッドが作成され、リストまたはディレクトリのファイルが同時に読み取られます。データ統合サービスで複数のスレッドを使用して1つのファイルが読み取られる場合もあります。また、データ統合サービスで1つのスレッドを使用して、リストやディレクトリの複数のファイルが読み取られる場合もあります。

## 同時読み込みのパーティション化

複数のスレッドが単一のファイルソースから読み取りを行う場合に、行の順序を保持するには、フラットファイルデータオブジェクトの**【同時読み込みのパーティション化】**プロパティを設定して順序を保持します。

フラットファイルデータオブジェクトの**【詳細】**プロパティの**【同時読み取りのパーティション化】**プロパティを設定します。**【ランタイム: 読み取り】**セクションでプロパティを見つけます。

**【同時読み込みのパーティション化】**プロパティの以下のいずれかのオプションを選択します。

### スループットの最適化

複数のパーティションが1つのファイルソースから読み取る場合、データ統合サービスでは行の順番が保持されません。このオプションは、複数のパーティションが1つのファイルソースから読み込む順序が重要でない場合に使用します。

デフォルトのオプションです。

### 相対順序を維持する

各パーティションが読み取る入力行のソート順が保持されます。

以下の表に、2つのパーティションが読み取る10行を含むファイルソースのソート順の例を示します。

パーティション	読み込まれる行
Partition #1	1、3、5、8、9
Partition #2	2、4、6、7、10

### 絶対順序を維持する

すべてのパーティションが読み取るすべての入力行のソート順が保持されます。バッチトランスフォーメーションを含むパススルーマッピングでは、ターゲットに書き込まれる行の順序は、入力行と同じ順序になります。

以下の表に、2つのパーティションが読み取る10行を含むファイルソースのソート順の例を示します。

パーティション	読み込まれる行
Partition #1	1、2、3、4、5
Partition #2	6、7、8、9、10

## パーティション化されたリレーショナルソース

パーティション化が有効になっているマッピングで IBM DB2 for LUW または Oracle ソースから読み取りを行う場合、データ統合サービスは複数のスレッドを使用してリレーショナルソースを読み取ることができます。データ統合サービスは、各スレッド用にデータベースへの個別の接続を作成します。

**注:** マッピングが DB2 for LUW または Oracle 以外のリレーショナルソースから読み取りを行う場合、データ統合サービスは1つのスレッドを使用してソースから読み取りを行います。データ統合サービスは、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。

データ統合サービスでは DB2 for LUW または Oracle データベースシステムに対してパーティション情報のクエリを実行します。ソーステーブルでデータベースのパーティション化がサポートされている場合、データ統合サービスでは、複数のスレッドを使用してパーティション化されたデータを、データベースの対応するノードから読み取ることができます。データ統合サービスでは、各 reader スレッドに対して SQL クエリが生成されます。

データ統合サービスが使用する reader スレッドの数は以下の状況によって異なります。

### データベースパーティションの数は、最大並行処理値以下です。

データ統合サービスは、各データベースパーティションについて1つの読み取りスレッドを使用します。データ統合サービスは1つのデータベースパーティションを各 reader スレッドに分散します。

複合パーティション化を使用する Oracle ソースの場合、データ統合サービスは各データベースのサブパーティションに1つの読み取りスレッドを使用します。例えば、Oracle ソースに3つのパーティションが含まれ、各パーティションに2つのサブパーティションが含まれる場合、データ統合サービスは6つの読み取りスレッドを使用します。

### データベースパーティションの数が並列処理の最大値を超えている。

データ統合サービスは、最大並行処理値で定義されている数の読み取りスレッドを使用します。データ統合サービスは複数のデータベースパーティションを一部の reader スレッドに分散します。例えば、DB2 for LUW ソースに5つのデータベースパーティションがあり、最大並行処理値が3に設定されているとします。データ統合サービスは3つの読み取りスレッドを使用します。データ統合サービスは、2つのデータベースパーティションを、1番目の reader スレッドと2番目の reader スレッドに分散します。サービスは1つのデータベースパーティションを3番目の reader スレッドに分散します。

### データベースパーティションがない。

データ統合サービスは1つのスレッドを使用してソースから読み取りを行います。データ統合サービスは、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。

## パーティション化のリレーショナル接続タイプ

データ統合サービスでは複数のスレッドを使用して、データベースへの接続で使用する接続タイプに基づいて、DB2 for LUW または Oracle のリレーショナルソースを読み取ることができます。

DB2 for LUW または Oracle のデータベースに接続するには、以下のどの接続タイプでも使用できます。

- DB2 for LUW 接続または Oracle 接続
- JDBC 接続
- ODBC 接続

複数のスレッドを使用して DB2 for LUW または Oracle のリレーショナルソースを読み取るには、リレーショナルデータオブジェクトが DB2 for LUW または Oracle の接続を使用する必要があります。

DB2 for LUW または Oracle のリレーショナルデータオブジェクトが JDBC または ODBC 接続を使用する場合、データ統合サービスはスレッドを 1 つ使用してソースを読み取ります。データ統合サービスは、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。

## パーティション化されたリレーショナルソースの SQL クエリ

データ統合サービスは、複数のスレッドを使用してリレーショナルソースを読み取る際に、読み取りスレッドごとに SQL クエリを生成します。

データベースソースのデータベースパーティションの方が並列処理の最大値よりも多い場合、データ統合サービスは reader スレッド全体にわたってデータを分散します。データ統合サービスは、複数のデータベースパーティションから読み取る SQL クエリを生成できます。Oracle ソースにサブパーティションが含まれる場合、データ統合サービスは複数のデータベースサブパーティションから読み取る SQL クエリを生成できます。

### DB2 for LUW または Oracle ソースの例

最大並行処理値は 3 で、リレーショナルソースには 5 つのデータベースパーティションがあります。データ統合サービスでデータベースパーティションに対して SQL クエリが実行されている場合、最初と 2 番目の reader スレッドは 2 つのデータベースパーティションからデータを受け取ります。3 番目の reader スレッドは、1 つのデータベースパーティションからデータを受け取ります。この例では、読み取りトランスフォーメーション内の単純クエリで個別選択オプションが有効になっていません。

DB2 for LUW ソースを使用する場合は、データ統合サービスで、最初の reader スレッドに対して以下の文と同様の SQL 文が生成されます。

```
SELECT <column list> FROM <table name>  
WHERE (nodenumber(<column 1>)=0 OR nodenumber(<column 1>) = 3)
```

Oracle ソースを使用する場合は、データ統合サービスで、最初の reader スレッドに対して以下の文と同様の SQL 文が生成されます。

```
SELECT <column list> FROM <table name> PARTITION <database_partition1 name> UNION ALL  
SELECT <column list> FROM <table name> PARTITION <database_partition4 name> UNION ALL
```

### Oracle ソースとサブパーティションの例

Oracle ソースには、1 から 5 の 5 つのパーティションと、各パーティションに a と b の 2 つのサブパーティションがあります。最大並行処理値は 3 です。最初の reader スレッドは、4 つのデータベースサブパーティションからデータを受け取ります。2 番目と 3 番目の reader スレッドは、3 つのデータベースサブパーティションからデータを受け取ります。この例では、読み取りトランスフォーメーション内の単純クエリで個別選択オプションが有効になっていません。

データ統合サービスで、最初の reader スレッドに対して、以下の文と同様の SQL 文が生成されます。

```
SELECT <column list> FROM <table name> SUBPARTITION <database_subpartition1_a name> UNION ALL  
SELECT <column list> FROM <table name> SUBPARTITION <database_subpartition1_b name> UNION ALL
```

```
SELECT <column list> FROM <table name> SUBPARTITION <database_subpartition4_a name> UNION ALL  
SELECT <column list> FROM <table name> SUBPARTITION <database_subpartition4_b name> UNION ALL
```

## リレーショナルソースパーティションに関するルールおよびガイドライン

リレーショナルソースからの読み取りを実行するマッピングのパーティション化を有効にする場合は、以下のルールおよびガイドラインを考慮してください。

- データ統合サービスは、1 スレッドを使用してソースを読み取りますが、以下の状況では、複数のスレッドを使用して残りのマッピングパイプラインステージを処理できます。
  - マッピングが DB2 for LUW または Oracle 以外のリレーショナルソースから読み取る。
  - マッピングが JDBC または ODBC 接続を使用して DB2 for LUW または Oracle ソースから読み取る。
  - マッピングが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する。
  - 読み取りトランスフォーメーションで単純クエリを使用して、ソートの基準となるポートを選択するか、ユーザー定義の結合を設定する。
  - 読み取りトランスフォーメーションで詳細クエリを使用して、カスタム SQL クエリを作成する。
- 読み取りトランスフォーメーションで単純クエリを使用してヒントを作成、異なる値を選択、またはソースフィルタを入力する場合は、データ統合サービスで複数のスレッドを使用してソースを読み取ることができます。データ統合サービスで、ヒント、異なる値、またはソースフィルタが、各パーティション用に生成された SQL クエリに追加されます。

## パーティション化されたフラットファイルターゲット

パーティション化が有効になっているマッピングがフラットファイルターゲットに書き込む場合、データ統合サービスは複数のスレッドを使用してファイルターゲットに書き込むことができます。

データ統合サービスは、Hadoop 分散ファイルシステム (HDFS) にフラットファイルまたはファイルのパーティションを作成できます。

ファイル出力タイプまたはコマンド出力タイプを持つようにフラットファイルデータオブジェクトを設定できます。フラットファイルデータオブジェクトがファイル出力タイプである場合、データ統合サービスはターゲットデータをフラットファイルに書き込みます。複数のスレッドがフラットファイルターゲットに書き込むと、各スレッドは個別のファイルに対してターゲット出力を書き込みます。各スレッドは以下の形式でファイルに名前を付けます。

```
<output_file_name><partition_number>.out
```

例えば、3 つのスレッドが MyOutput1.out、MyOutput2.out、および MyOutput3.out という名前のファイルに書き込むとします。

パフォーマンスが最適化されるように複数の出力ファイルディレクトリを設定したり、単一のマージファイルに書き込むようにフラットファイルデータオブジェクトを設定したりできます。

フラットファイルデータオブジェクトがコマンド出力タイプである場合、データ統合サービスはターゲットデータをフラットファイルまたはマージファイルの代わりにコマンドまたはマージコマンドに出力します。複数のパーティションがフラットファイルターゲットに書き込む場合は、単一のパーティションのターゲットデータを処理するか、すべてのターゲットパーティションのマージデータを処理するようにコマンドを設定できます。

## パーティション化されたファイルターゲットに対応するための出力ファイルディレクトリの最適化

デフォルトでは、フラットファイルデータオブジェクトの出力タイプがファイルの場合、各スレッドがターゲット出力を個々のファイルに書き込みます。複数のスレッドで1つのファイルターゲットに書き込む場合に最適なパフォーマンスを実現するには、複数の出力ファイルディレクトリを設定します。

複数のスレッドが単一のディレクトリに書き込む場合、入力/出力 (I/O) 競合によりマッピングにボトルネックが発生することがあります。I/O 競合は、複数のスレッドがファイルシステムに同時にデータを書き込む場合に発生する可能性があります。

複数のディレクトリを設定すると、ラウンドロビン方式で各スレッドに対して出力ディレクトリが割り当てられます。例えば、`directoryA` と `directoryB` をターゲットディレクトリとして使用するようフラットファイルデータオブジェクトを設定したとします。データ統合サービスがファイルターゲットへの書き込みに4つのスレッドを使用する場合、1つ目と3つ目の書き込みスレッドはターゲットファイルを `directoryA` に書き込みます。2つ目と4つ目の書き込みスレッドは、ターゲットファイルを `directoryB` に書き込みます。

データ統合サービスがターゲットへの書き込みに複数のスレッドを使用しない場合は、出力ファイルは、最初に一覧表示されたディレクトリに書き込まれます。

フラットファイルデータオブジェクトの **【詳細】** プロパティで、出力ファイルディレクトリを設定します。**【ランタイム: 書き込み】** セクションで、**【出力ファイルディレクトリ】** プロパティを見つけます。デフォルトでは、このプロパティは、データ統合サービスに対して定義されているシステムパラメータ値を使用するように設定されます。データ統合サービスの **【ターゲットディレクトリ】** プロパティ用に管理者がセミコロンで区切られた複数のディレクトリを入力した場合は、デフォルトの `TargetDir` システムパラメータ値を使用してください。

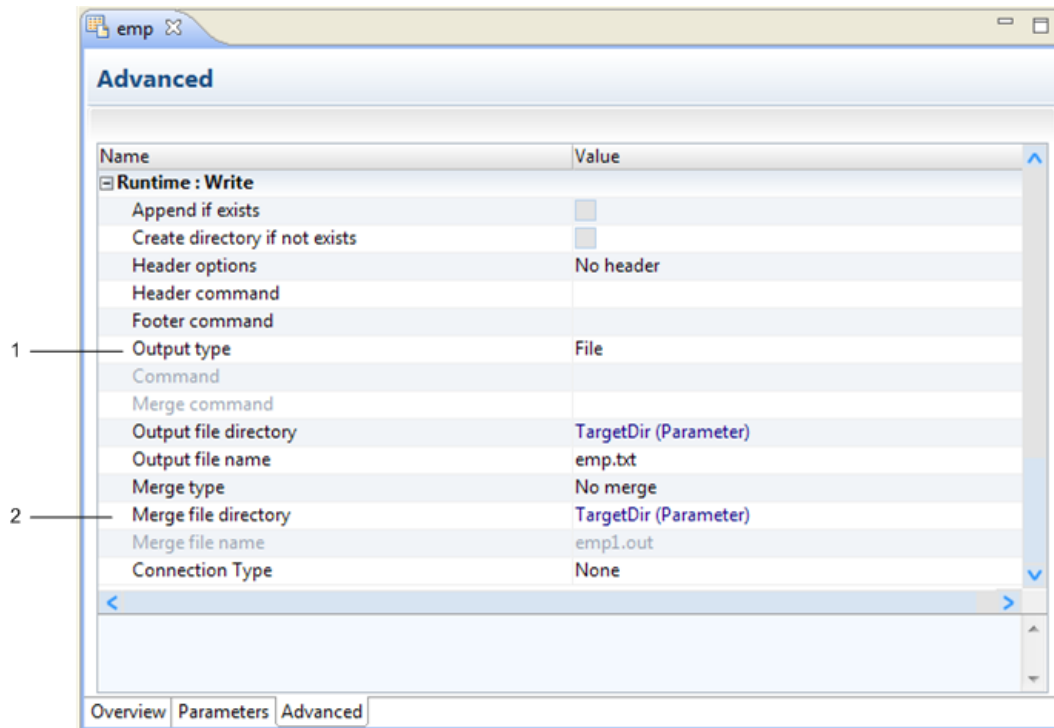
異なる値を入力し、フラットファイルデータオブジェクトに固有の出力ファイルディレクトリを複数設定できます。プロパティ、またはプロパティに割り当てられたユーザー定義のパラメータ用に、セミコロンで区切られた複数のディレクトリを入力します。

## パーティション化されたファイルターゲットのマージオプション

デフォルトでは、フラットファイルデータオブジェクトの出力タイプがファイルの場合、各スレッドがターゲット出力を個々のファイルに書き込みます。パーティションのターゲットデータはマージできます。ターゲットデータをマージすると、データ統合サービスはすべてのターゲットパーティションに対して1つのマージファイルを作成します。

マージオプションは、フラットファイルデータオブジェクトの **【詳細】** プロパティで設定します。**【ランタイム: 書き込み】** セクションで、マージプロパティを見つけます。

以下の図に、フラットファイルデータオブジェクトの「詳細」プロパティのマージオプションを示します。



1. ファイル出力タイプ
2. マージオプション

【マージタイプ】 プロパティには、以下のいずれかのオプションを選択します。

#### マージなし

データ統合サービスは、ターゲット出力を各パーティションの個別のファイルに同時に書き込みます。

デフォルトのオプションです。

#### シーケンシャル

データ統合サービスは各パーティションに対して出力ファイルを作成してから、それらを1つのマージファイルにマージします。データ統合サービスは、出力ファイル名と出力ファイルディレクトリ値を使用して個々のターゲットファイルを作成します。データ統合サービスは、各書き込みスレッドの完了順に、各パーティションの出力データをマージファイルに順次追加します。例えば、パーティション2の書き込みスレッドがパーティション1のスレッドよりも前に完了した場合、データ統合サービスは、パーティション2、パーティション1の順序でデータをマージファイルに追加します。

#### ファイルリスト

データ統合サービスは、各パーティションに対してターゲットファイルを作成し、個々のファイルのパスを含むファイルリストを作成します。データ統合サービスは、出力ファイル名と出力ファイルディレクトリ値を使用して個々のターゲットファイルを作成します。ターゲットファイルをマージディレクトリまたはマージディレクトリの下ディレクトリに書き込んだ場合、ファイルリストには相対パスが含まれます。それ以外の場合は、ファイルリストに絶対パスが含まれます。別のマッピングでターゲットファイルをソースファイルとして使用する場合は、このファイルをソースファイルとして使用します。

#### 同時

データ統合サービスは、すべてのターゲットパーティションのデータをマージファイルに同時に書き込みます。パーティションごとに中間ファイルが作成されることはありません。データ統合サービスはすべて



のパーティションのデータをマージファイルに同時に書き込むため、マージファイル内のデータの順序が連続していない可能性があります。

フラットファイルデータオブジェクトを設定してターゲットデータをマージする場合、オプションで **【マージファイルディレクトリ】** および **【マージファイル名】** プロパティのデフォルト値を編集できます。

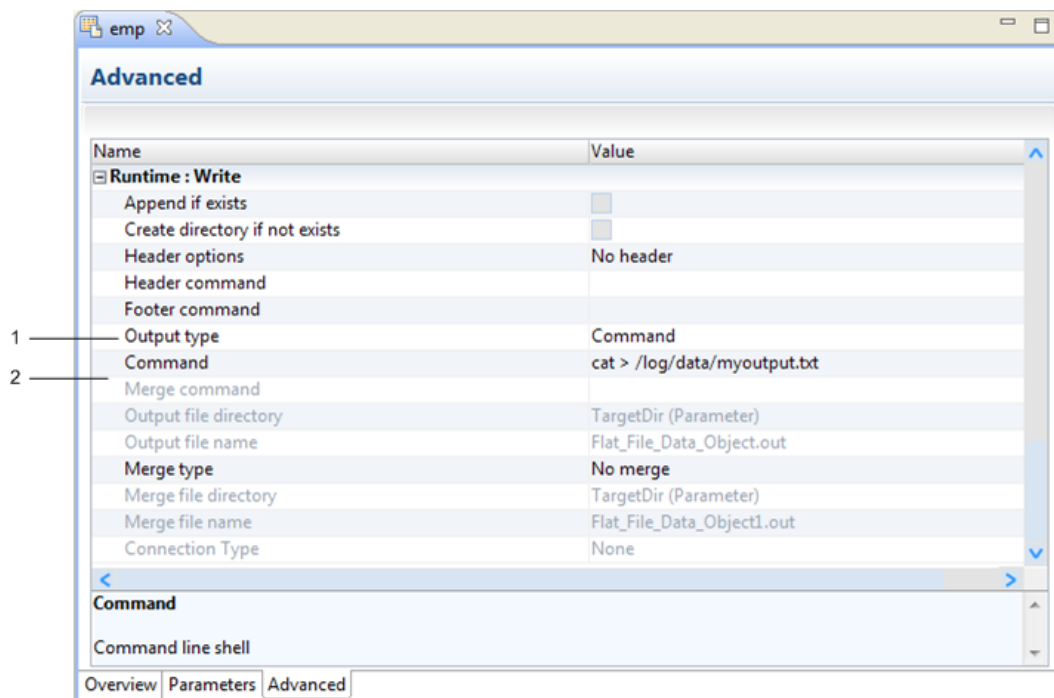
フラットファイルデータオブジェクトを設定してターゲットデータをマージし、データ統合サービスがターゲット用のパーティションを作成しない場合、データ統合サービスはマージオプションを無視します。サービスは **【出力ファイル名】** および **【出力ファイルディレクトリ】** プロパティで定義されたファイルに書き込みます。

## パーティション化されたファイルターゲットのコマンド

フラットファイルデータオブジェクトの出力タイプがコマンドの場合、コマンドを使用して、1つのパーティションのターゲットデータの処理や、マッピング内のすべてのターゲットパーティションのマージデータの処理が行えます。データ統合サービスはデータを、フラットファイルやマージファイルではなく、コマンドまたはマージコマンドに送信します。

フラットファイルデータオブジェクトの **【詳細】** プロパティで、パーティションのデータを処理するコマンドを設定します。**【ランタイム: 書き込み】** セクションで、コマンドプロパティを見つけます。

以下の図に、コマンドを使用して1つのパーティションのターゲットデータを処理するように設定されたフラットファイルデータオブジェクトを示します。



1. コマンド出力タイプ
2. コマンドのオプション

UNIX では、有効な任意の UNIX コマンドまたはシェルスクリプトを使用します。Windows では、有効な任意の DOS またはバッチファイルを使用します。

コマンドを使用して次のタイプのターゲットデータを処理できます。



### 単一パーティションのデータ

各 writer スレッドが個別に実行するコマンドを入力します。各スレッドは、マッピングが実行されるときにターゲットデータをコマンドに送信します。各スレッドは、異なるデータセットで同じコマンドを実行します。

コマンドを入力するときは、マッピングを実行するオペレーティングシステムを考慮する必要があります。例えば、コマンド `cat > /log/data/myoutput.txt` を入力すると、複数のスレッドが同じファイルに書き込むため、オペレーティングシステムのエラーが発生する可能性があります。コマンド `cat >> /log/data/myoutput.txt` を入力すると、複数のスレッドがデータを同じファイルに付加するため、オペレーティングシステムエラーが発生する可能性は少なくなります。

単一パーティションのターゲットデータをコマンドに送信するには、**【出力タイプ】** プロパティで **【コマンド】** を選択し、**【マージタイプ】** プロパティで **【マージなし】** を選択します。**【コマンド】** プロパティのコマンドを入力します。

### すべてのパーティションのデータのマージ

すべての writer スレッドのデータのマージを処理するマージコマンドを入力します。データ統合サービスは、マージデータを処理するためのコマンドに、コンカレントマージタイプを使用する必要があります。マッピングを実行するときに、各スレッドは同時にターゲットデータをマージコマンドに送信します。マージコマンドは、すべてのデータで 1 回実行します。コマンドでは、ターゲットデータの順序が維持されない場合があります。

すべてのパーティションのマージデータをマージコマンドに送信するには、**【出力タイプ】** プロパティで **【コマンド】** を選択し、**【マージタイプ】** プロパティで **【コンカレント】** を選択します。**【マージコマンド】** プロパティのコマンドを入力します。

## パーティション化されたリレーショナルターゲット

パーティション化が有効になっているマッピングが IBM DB2 for LUW または Oracle ターゲットへ書き込みを行う場合、データ統合サービスは複数スレッドを使用してリレーショナルターゲットに書き込むことができます。データ統合サービスは、各スレッド用にデータベースへの個別の接続を作成します。

**注:** マッピングが DB2 for LUW または Oracle 以外のリレーショナルターゲットに書き込む場合、データ統合サービスは 1 つのスレッドを使用してターゲットに書き込みます。データ統合サービスは、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。

ターゲットが Oracle である場合、データ統合サービスは最大並行処理値で定義されている数の書き込みスレッドを使用します。Oracle リレーショナルテーブルにパーティションがある場合、このデータベースはデータを正しいパーティションにルーティングします。

ターゲットが DB2 for LUW の場合、データ統合サービスはパーティション情報に DB2 for LUW システムを必要とします。データ統合サービスはパーティション化されたデータをターゲットデータベース内の対応するノードにロードします。

データ統合サービスが DB2 for LUW ターゲットに使用する writer スレッドの数は、以下の状況によって異なります。

#### データベースパーティションの数は、最大並行処理値以下です。

データ統合サービスは、データベースパーティションごとに 1 つの書き込みスレッドを使用します。各 writer スレッドは 1 つのデータベースパーティションに書き込みます。

**データベースパーティションの数が並列処理の最大値を超えている。**

データ統合サービスは、最大並行処理値で定義されている数の書き込みスレッドを使用します。データ統合サービスは複数のデータベースパーティションを一部の writer スレッドに分散します。例えば、DB2 for LUW ターゲットに 5 つのデータベースパーティションがあり、最大並行処理値が 3 に設定されているとします。データ統合サービスは 3 つの書き込みスレッドを使用します。データ統合サービスは、2 つのデータベースパーティションを、1 番目の writer スレッドと 2 番目の writer スレッドに分散します。サービスは 1 つのデータベースパーティションを 3 番目の writer スレッドに分散します。

**データベースパーティションがない。**

データ統合サービスは、最大並行処理値で定義されている数の書き込みスレッドを使用します。

ただし、ターゲットにデータベースパーティションがある場合は、ロードパフォーマンスを最適化することができます。この場合、各 writer スレッドは、データベースパーティションを含む DB2 for LUW ノードに接続します。すべてのスレッドが単一のマスターノードに接続するのではなく、writer スレッドが別々の DB2 for LUW ノードに接続するため、パフォーマンスは向上します。

## パーティション化のリレーショナル接続タイプ

データ統合サービスは複数のスレッドを使用して、データベースへの接続で使用する接続タイプに基づいて、DB2 for LUW または Oracle のリレーショナルターゲットに書き込むことができます。

DB2 for LUW または Oracle のデータベースに接続するには、以下のどの接続タイプでも使用できます。

- DB2 for LUW 接続または Oracle 接続
- JDBC 接続
- ODBC 接続

複数のスレッドを使用して DB2 for LUW または Oracle のリレーショナルターゲットに書き込むには、リレーショナルデータオブジェクトが DB2 for LUW 接続または Oracle 接続を使用する必要があります。

DB2 for LUW または Oracle のリレーショナルデータオブジェクトが JDBC 接続または ODBC 接続を使用する場合、データ統合サービスは 1 つのスレッドを使用してターゲットに書き込みます。データ統合サービスは、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。

## リレーショナルターゲットパーティションに関するルールおよびガイドライン

リレーショナルターゲットへの書き込みを行うマッピングのパーティション化を有効にする場合は、以下のルールとガイドラインを考慮してください。

- データ統合サービスは、1 つのスレッドを使用してターゲットに書き込みますが、以下の状況では、残りのマッピングパイプラインステージ用に複数のスレッドを使用できます。
  - マッピングが DB2 for LUW や Oracle 以外のリレーショナルターゲットに書き込む。
  - マッピングが JDBC または ODBC 接続を使用して DB2 for LUW または Oracle のターゲットに書き込む。
- DB2 for LUW ターゲットテーブルパーティションキーが小数のカラムである場合は、マッピングの高精度を有効にします。パーティションキーが小数のカラムであり、マッピングに対して高精度を有効にしない場合は、データ統合サービスでセッションが失敗することがあります。

# パーティション化されたトランスフォーメーション

パーティション化が有効になっているマッピングにパーティション化をサポートするトランスフォーメーションが含まれる場合、データ統合サービスは複数のスレッドを使用してそのトランスフォーメーションを実行できます。

データ統合サービスはトランスフォーメーションにパーティションポイントを追加する必要があるかどうか決定し、そのトランスフォーメーションのパイプラインステージの最適なスレッド数を決定します。また、データ統合サービスはパーティションポイントでデータを再分散する必要があるかどうかを決定します。例えば、データ統合サービスはアグリゲータトランスフォーメーションでデータを再分散して集計式の行をグループ化することがあります。

一部のトランスフォーメーションは、パーティション化をサポートしません。パーティション化が有効になっているマッピングにパーティション化をサポートしないトランスフォーメーションが含まれる場合、データ統合サービスは1つのスレッドを使用してそのトランスフォーメーションを実行します。データ統合サービスは、複数のスレッドを使用して残りのマッピングパイプラインステージを実行できます。

以下のトランスフォーメーションは、パーティション化をサポートしません。

- 関連付け
- 統合
- 例外
- 一致（フィールド一致分析に設定されている場合）
- REST Web サービスコンシューマ
- 未接続のルックアップ
- Web サービスコンシューマ

## パーティション化されたトランスフォーメーションに関する制限

パーティション化をサポートしている一部のトランスフォーメーションには、特有の設定が必要です。パーティション化が有効になっているマッピングにサポートされていない設定があるトランスフォーメーションが含まれる場合、データ統合サービスは1つのスレッドを使用してそのトランスフォーメーションを実行します。データ統合サービスは複数のスレッドを使用して残りのマッピングパイプラインステージを処理します。

以下のトランスフォーメーションには、パーティション化をサポートする特有の設定が必要です。

- アグリゲータトランスフォーメーションには、グループ化ポートを含める必要があります。アグリゲータトランスフォーメーションにパススルーポートを含めることはできません。アグリゲータトランスフォーメーションに、行ごとに現在の合計と平均を計算する数値関数を含めることはできません。
- 式トランスフォーメーションに、以下のタイプの関数または変数を含めることはできません。
  - 行ごとに現在の合計と平均を計算する数値関数。
  - 複数のスレッドでトランスフォーメーションが処理された場合に、異なる結果を返す可能性がある特殊関数。
  - 前の行の値に依存するローカル変数。
- ディジション、Java、および SQL トランスフォーメーションでは、**【パーティション化可能】** プロパティを有効にする必要があります。
- ジョイナトランスフォーメーションには、等価演算子を使用する結合条件が含まれていることが必要です。結合条件に複数の等価条件が含まれる場合、各条件は AND 演算子を使用して結合する必要があります。
- ランクトランスフォーメーションには、グループ化ポートを含める必要があります。

## トランスフォーメーションのキャッシュのパーティション化

キャッシュのパーティション化では、各パーティションごとに、アグリゲータ、ジョイナ、ランク、ルックアップ、またはソータの各トランスフォーメーションを処理する個別のキャッシュが作成されます。キャッシュのパーティション化中、各パーティションは異なるデータを別々のキャッシュに保存します。各キャッシュには、そのパーティションが必要とする行が含まれます。

各スレッドが個々のキャッシュに並行してクエリを実行するため、キャッシュをパーティション化することでマッピングのパフォーマンスが最適化されます。データ統合サービスがマッピング用のパーティションを作成する場合、データ統合サービスは、パーティション化されたアグリゲータ、ジョイナ、ランク、およびソータートランスフォーメーションに対して常にキャッシュのパーティション化を使用します。データ統合サービスは、パーティション化されたルックアップトランスフォーメーションに対しては、キャッシュのパーティション化を使用する場合があります。

データ統合サービスは、以下の状況で、接続されたルックアップトランスフォーメーションに対してキャッシュのパーティション化を使用します。

- ルックアップ条件に等価演算子だけが含まれる。
- 接続されたルックアップトランスフォーメーションがリレーショナルテーブルのデータを検索する場合、データベースは大文字と小文字を区別して比較するように設定されています。

例えば、ルックアップ条件に文字列ポートが含まれ、データベースが大文字と小文字を区別して比較するように設定されていない場合、データ統合サービスはキャッシュのパーティション化を使用しません。

データ統合サービスがルックアップトランスフォーメーションに対してキャッシュのパーティション化を使用しない場合、ルックアップトランスフォーメーションを実行するすべてのスレッドは同じキャッシュを共有します。各スレッドは同じキャッシュに連続してクエリを実行します。

**注:** データ統合サービスは、1つのスレッドを使用して未接続のルックアップトランスフォーメーションを実行するため、未接続のルックアップトランスフォーメーションにキャッシュのパーティション化を使用しません。

## パーティション化されたキャッシュのキャッシュサイズ

データ統合サービスはアグリゲータ、ジョイナ、ランク、ルックアップ、およびソータートランスフォーメーションに対してキャッシュのパーティション化を使用する場合、パーティション全体にキャッシュサイズを分割します。

キャッシュサイズは、トランスフォーメーションの詳細プロパティで設定します。キャッシュサイズはバイト数で入力するか、**【自動】**を選択してデータ統合サービスによって実行時に自動で決定されるようにすることができます。

サイズを数値で指定した場合、データ統合サービスは実行時にトランスフォーメーションスレッド数でキャッシュサイズを分割します。例えば、トランスフォーメーションキャッシュサイズを 2,000,000 バイトに設定したとします。データ統合サービスは 4つのスレッドを使用してトランスフォーメーションを実行します。各スレッドが最大 500,000 バイトをキャッシュサイズとして使用するよう、キャッシュサイズ値が分割されます。

**【自動】**を選択すると、データ統合サービスはトランスフォーメーションのキャッシュサイズを実行時に決定します。次に、そのキャッシュサイズをトランスフォーメーションスレッドの数で分割します。

## パーティション化に対応するためのキャッシュディレクトリの最適化

アグリゲータ、ジョイナ、ランク、およびソータートランスフォーメーションでキャッシュのパーティション化を行うときのパフォーマンスを最適化するため、複数のキャッシュディレクトリを設定します。

データ統合サービスがキャッシュのパーティション化を使用し、オーバーフローした値をキャッシュファイルに格納する必要がある場合、トランスフォーメーションスレッドはキャッシュディレクトリに書き込みを行います。複数のスレッドが単一のディレクトリに書き込む場合、入力/出力 (I/O) 競合によりマッピングにボト

ルネックが発生することがあります。I/O 競合は、複数のスレッドがファイルシステムに同時にデータを書き込む場合に発生する可能性があります。

複数のキャッシュディレクトリを設定すると、データ統合サービスは各トランスフォーメーションスレッドのキャッシュディレクトリをラウンドロビン方式で決定します。例えば、ディレクトリ A とディレクトリ B をキャッシュディレクトリとして使用するアグリゲータトランスフォーメーションを設定するとします。このアグリゲータトランスフォーメーションを実行するためにデータ統合サービスが 4 つのスレッドを使用する場合、最初と 3 つ目のトランスフォーメーションスレッドはオーバーフローした値を directoryA のキャッシュファイルに格納します。2 つ目と 4 つ目のトランスフォーメーションスレッドは、オーバーフローした値をディレクトリ B のキャッシュファイルに格納します。

データ統合サービスがアグリゲータ、ジョイナ、ランク、またはソータートランスフォーメーションでキャッシュのパーティション化を使用しない場合は、オーバーフローした値は最初にリストされたディレクトリのキャッシュファイルに格納されます。

**注:** ルックアップトランスフォーメーションでは、1 つのキャッシュディレクトリしか使用できません。

アグリゲータ、ジョイナ、またはランクトランスフォーメーションの [詳細] プロパティの **【キャッシュディレクトリ】** プロパティで、キャッシュディレクトリを設定します。ソータートランスフォーメーションの [詳細] プロパティの **【作業ディレクトリ】** プロパティで、キャッシュディレクトリを設定します。デフォルトでは、**【キャッシュディレクトリ】** プロパティと **【作業ディレクトリ】** プロパティは、データ統合サービスに定義されているシステムパラメータ値を使用するように設定されます。データ統合サービスの **【キャッシュディレクトリ】** または **【一時ディレクトリ】** プロパティ用に管理者がセミコロンで区切られた複数のディレクトリを入力した場合は、デフォルトの CacheDir または TempDir システムパラメータ値を使用してください。

異なる値を入力し、トランスフォーメーションに固有のキャッシュディレクトリを複数設定できます。プロパティ、またはプロパティに割り当てられたユーザー定義のパラメータ用に、セミコロンで区切られた複数のディレクトリを入力します。

## トランスフォーメーションのためのパーティション化の無効化

パーティション化されたディシジョン、Java、または SQL トランスフォーメーションは、各マッピング実行で同じ結果を返さないことがあります。これらのトランスフォーメーションのパーティション化を無効にすることで、データ統合サービスが 1 つのスレッドを使用してトランスフォーメーションを処理するようにできます。データ統合サービスは複数のスレッドを使用して残りのマッピングパイプラインステージを処理します。

Java または SQL トランスフォーメーションでは、**【パーティション化可能】** 詳細プロパティがデフォルトで選択されます。この詳細プロパティをクリアし、トランスフォーメーションのパーティション化を無効にします。

ディシジョントランスフォーメーションでは、**【パーティション化可能】** 詳細プロパティはデフォルトでクリアされます。この詳細プロパティを選択して、トランスフォーメーションのパーティション化を有効にします。

トランスフォーメーションのためにパーティション化を無効にする理由は、トランスフォーメーションのタイプによって異なります。

### ディシジョントランスフォーメーション

数値関数を使用するディシジョントランスフォーメーションのパーティション化を無効にする場合があります。数値関数の CUME、MOVINGSUM、および MOVINGAVG は、行単位で現在の合計および平均を計算します。パーティション化されたディシジョントランスフォーメーションにこれらのいずれかの関数が含まれる場合、各スレッドは関数を個別に処理します。各関数は、すべてのデータではなく、データのサブセットを使用して結果を計算します。したがって、CUME、MOVINGSUM、または MOVINGAVG 関数を使用するパーティション化されたトランスフォーメーションは、マッピングを実行するたびに、同じ計算結果を返さない場合があります。

### Java トランスフォーメーション

Java コードで Java トランスフォーメーションを 1 つのスレッドで処理する必要がある場合に、Java トランスフォーメーションのパーティション化を無効にします。



## SQL トランスフォーメーション

SQL クエリで SQL トランスフォーメーションを 1 つのスレッドで処理する必要がある場合に、SQL トランスフォーメーションのパーティション化を無効にします。または、SQL トランスフォーメーションのパーティション化を無効にすることで、データベースへの接続を 1 つだけに限定することもできます。

# パーティション化されたマッピングにおける順序の維持

ソート済みフラットファイルソース、ソート済みリレーショナルソース、またはソータートランスフォーメーションを使用したマッピングにおける順序を確立できます。マッピングにパーティションポイントを追加する場合、データ統合サービスがデータを再分散し、マッピング内ですでに確立されていた順序が失われることがあります。パーティション化されたマッピングで順序を保持するには、一部のトランスフォーメーションと書き込みトランスフォーメーションで行順序が保持されるように指定する必要があります。

以下のマッピングオブジェクトについては、入力データの行順序を維持するように指定できます。

- 式トランスフォーメーション
- Java 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
- SQL トランスフォーメーション
- 書き込みトランスフォーメーション[かきこみとらんすふおーめーしょん]

例えば、ソート順で書き込まれているデータに依存するデータベーストリガがリレーショナルターゲットに含まれる場合、行順序を保持するように書き込みトランスフォーメーションを設定します。

行順序を保持するように書き込みトランスフォーメーションを設定すると、データ統合サービスは 1 つのスレッドを使用してターゲットに書き込みます。ソート済みの入力を使用するアグリゲータトランスフォーメーションが書き込みトランスフォーメーションの前にある場合、データ統合サービスは 1 つのスレッドを使用してアグリゲータトランスフォーメーションとターゲットの両方を処理します。

他のすべてのトランスフォーメーションで行順序を保持するように設定するとき、データ統合サービスは順序を保持するためのトランスフォーメーションパイプラインステージの最適なスレッド数を判断します。

行順序を保持するためにトランスフォーメーションの設定に使用する方法は、以下のオブジェクトタイプによって異なります。

### 式、シーケンスジェネレータ、または SQL トランスフォーメーション

式、シーケンスジェネレータ、または SQL トランスフォーメーションの **【詳細】** プロパティで、**【行順序を保持】** プロパティを選択します。

### Java トランスフォーメーション

Java トランスフォーメーションの **【詳細】** プロパティで、**【ステートレス】** プロパティを選択します。

### 書き込みトランスフォーメーション[かきこみとらんすふおーめーしょん]

書き込みトランスフォーメーションの **【詳細】** プロパティで、**【行順序を保持】** プロパティを選択します。

## 安定ソートの維持

パーティション化されたマッピングで順序を維持する場合、データ統合サービスは安定ソートを実行しません。データ統合サービスは、ソートキーに基づいて行の順序を維持します。ただし、複数の行でソートキーの値が等しい場合、それらの行の相対順序は入力と出力とで一致しないことがあります。

例えば、パーティションが有効化されたマッピングで、次のデータを含むソート済みのフラットファイルソースを読み取るとします。

```
Order_ID,Item_ID,Item,Quantity,Price
45,000468,ItemD,5,0.56
45,123456,ItemA,5,3.04
41,456789,ItemB,2,12.02
43,123456,ItemA,3,3.04
```

このマッピングには、降順のソートキーとして Order\_ID が指定されたソータートランスフォーメーションが含まれています。データ統合サービスでは、複数のスレッドを使用してソータートランスフォーメーションを実行する場合、Order\_ID の値が同一の行の相対順序は維持されません。例えば、マージされたターゲットファイルに次の順序で行を書き出すとします。

```
Order_ID,Item_ID,Item,Quantity,Price
45,123456,ItemA,5,3.04
45,000468,ItemD,5,0.56
43,123456,ItemA,3,3.04
41,456789,ItemB,2,12.02
```

安定ソートを維持するには、マッピングの **【最大並行処理】** ランタイムプロパティを 1 に設定して、マッピングのパーティション化を無効にします。

## マッピングの最大並行処理のオーバーライド

デフォルトでは、各マッピングの **【最大並行処理】** プロパティは **【自動】** に設定されています。各マッピングは、データ統合サービスに定義されている最大並行処理値を使用します。最大並行処理値をオーバーライドし、特定のマッピングの最大値を定義できます。

データ統合サービスとマッピングで設定されている最大並行処理の整数値が異なる場合、データ統合サービスではこれらの最小値が使用されます。

以下の理由のため、マッピングの **【最大並行処理】** プロパティをオーバーライドした方がよい場合があります。

**CPU の処理能力を超えるスレッドとなる複雑なマッピングを実行する。**

完全なパイプラインのマッピングに実行できる並行処理スレッドの合計数は、並行処理値にパイプラインステージの数を掛けた値になります。各パーティションポイントには、追加のパイプラインステージが追加されます。複数のアグリゲータまたはジョイナートランスフォーメーションを使用する複雑なマッピングには、多くのパイプラインステージがある場合があります。多数のパイプラインステージにより、データ統合サービスで CPU の処理能力を超える数のスレッドが使用される可能性があります。

**各パイプラインステージで並行処理スレッドが少ない方が、マッピングのパフォーマンスがよくなります。**

単一のマッピングを少ない並行処理スレッドで実行すると、データ統合サービスで追加のジョブを実行するために使用できるスレッドが増えます。

**トランスフォーメーションの推奨並行処理値を定義する必要がある。**

マッピングの最大並行処理をオーバーライドする場合、マッピング内の特定のトランスフォーメーションの推奨並行処理値を定義できます。多数のポートが含まれるトランスフォーメーションや複雑な計算を実行するトランスフォーメーションでは、パフォーマンスを最適化するために推奨並行処理値を定義することをお勧めします。

アドレスバリデータまたは一致トランスフォーメーションに対して実行インスタンス数の値を定義する必要があります。

マッピングの最大並行処理をオーバーライドした場合、データ統合サービスは、マッピング内のアドレスバリデータまたは一致トランスフォーメーションに対する実行インスタンス数の値を考慮します。実行インスタンス数の値を定義して、トランスフォーメーションのパフォーマンスを最適化することをお勧めします。

## トランスフォーメーションのための提案された並行処理

マッピングの**【最大並行処理】**ランタイムプロパティをオーバーライドする場合、マッピングランタイムプロパティの特定のトランスフォーメーションの**【提案された並行処理】**プロパティを定義できます。

データ統合サービスにより、トランスフォーメーションのパーティション化が可能な限り、トランスフォーメーションパイプラインステージのスレッド数について提案された並行処理値が考慮されます。例えば、行順序を保持するようにマッピングを設定した場合、データ統合サービスはトランスフォーメーション用に1つのスレッドを使用することが必要になる場合があります。

マッピングの**【最大並行処理】**ランタイムプロパティが**【自動】**に設定されていると、マッピングのトランスフォーメーションの提案された並行処理値を定義できません。トランスフォーメーションのための提案された並行処理の値を定義した後に、マッピングの最大並行処理の値を**【自動】**に設定した場合、データ統合サービスは提案された並行処理の値を無視します。

多数のポートが含まれるトランスフォーメーションや複雑な計算を実行するトランスフォーメーションでは、パフォーマンスを最適化するために提案された並行処理値を定義することをお勧めします。

例えば、パーティション化が有効になっているマッピングが小さなデータセットを処理する場合、データ統合サービスは式トランスフォーメーションパイプラインステージの処理にはスレッドが1つあれば十分だと判断するかもしれません。ただし、式トランスフォーメーションに多数の複雑な計算が含まれていると、トランスフォーメーションパイプラインステージの処理に時間がかかる可能性もあります。提案された並行処理値には、1より大きく、マッピングまたはデータ統合サービスに定義された最大並行処理値より小さい値を入力できます。データ統合サービスは式トランスフォーメーションのスレッド数の提案された並行処理値を使用します。

マッピングの最大並行処理をオーバーライドする場合、トランスフォーメーションの**【提案された並行処理】**プロパティの次の値を設定できます。

提案された並行処理値	説明
1	データ統合サービスは、1つのスレッドを使用してトランスフォーメーションを実行します。
自動	データ統合サービスは、マッピングおよびデータ統合サービスに対して定義された最大並行処理を考慮します。データ統合サービスは最低値を使用して、トランスフォーメーションを実行するスレッドの最適な数を決定します。 各トランスフォーメーションのデフォルトです。
1より大きい	データ統合サービスにより、トランスフォーメーションに定義された提案された並行処理、マッピングに定義された最大並行処理、およびデータ統合サービスに定義された最大並行処理が考慮されます。データ統合サービスは最低値を使用して、トランスフォーメーションを実行するスレッドの数を決定します。

次のトランスフォーメーションのマッピングランタイムプロパティの**【提案された並行処理】**プロパティを定義できます。

- アグリゲータ



- 式
- フィルタ
- Java
- 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
- ソータ
- SQL
- 共有体
- アップデートストラテジ

## アドレスバリデータおよび一致トランスフォーメーションの実行インスタンス数

マッピングの【最大並行処理】ランタイムプロパティをオーバーライドした場合、データ統合サービスは、アドレスバリデータまたは一致トランスフォーメーションに対して定義された【実行インスタンス数】詳細プロパティの値を考慮します。

データ統合サービスは、トランスフォーメーションがパーティション化可能な限り、実行インスタンス数の値を考慮してトランスフォーメーションパイプラインステージのスレッド数を決定します。例えば、行順序を保持するようにマッピングを設定した場合、データ統合サービスはトランスフォーメーション用に1つのスレッドを使用することが必要になる場合があります。

ID 照合分析用にトランスフォーメーションを設定した場合、一致トランスフォーメーション上の実行インスタンスの数を増やすことができます。フィールド一致分析用にトランスフォーメーションを設定した場合、一致トランスフォーメーション上の実行インスタンスの数を増やすことはできません。フィールド一致分析では、一致トランスフォーメーションは単一の実行インスタンスを使用します。

マッピングの【最大並行処理】ランタイムプロパティが【自動】に設定されている場合、データ統合サービスは、アドレスバリデータまたは一致トランスフォーメーションに対して定義された実行インスタンス数の値を無視します。

マッピングの最大並行処理をオーバーライドする場合、アドレスバリデータまたは一致トランスフォーメーションの **【実行インスタンス数】** 詳細プロパティの以下の値を設定できます。

実行インスタンス数の値	説明
1	データ統合サービスは、1つのスレッドを使用してトランスフォーメーションを実行します。アドレスバリデータトランスフォーメーションのデフォルト。
自動	データ統合サービスは、マッピングおよびデータ統合サービスに対して定義された最大並行処理を考慮します。データ統合サービスは最低値を使用して、トランスフォーメーションを実行するスレッドの最適な数を決定します。 ID 照合分析での一致トランスフォーメーションのデフォルト。
1 より大きい	データ統合サービスは、トランスフォーメーションに対して定義された実行インスタンス数、マッピングに対して定義された最大並行処理、およびデータ統合サービスに対して定義された最大並行処理を考慮します。データ統合サービスは最低値を使用して、トランスフォーメーションを実行するスレッドの数を決定します。

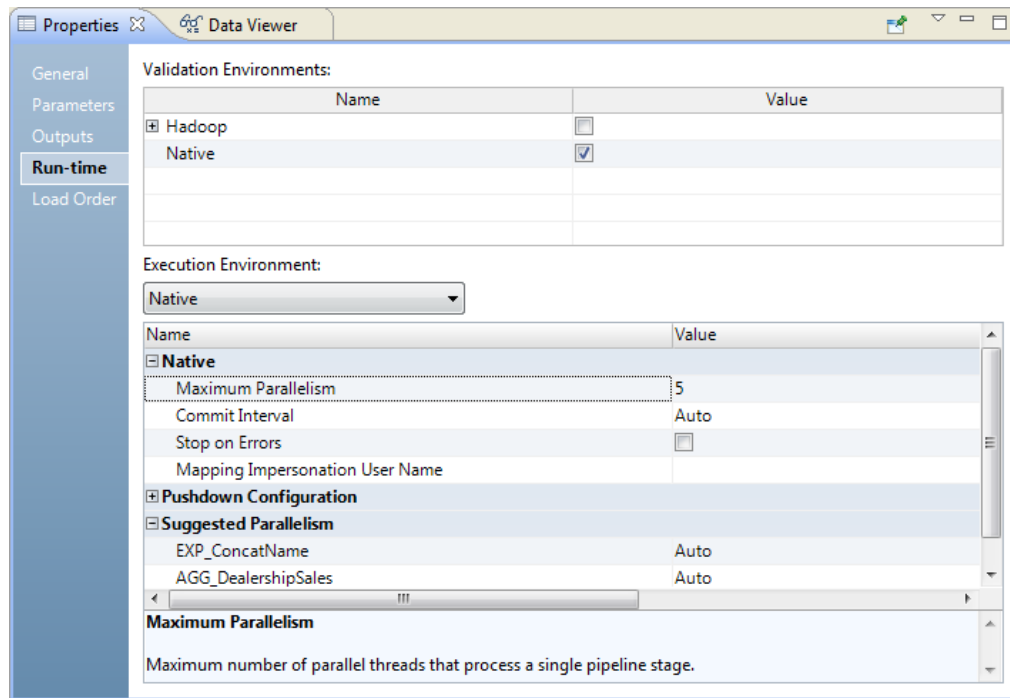
**注:** また、データ統合サービスは、アドレスバリデータトランスフォーメーションの最適なスレッド数を計算する際に、コンテンツ管理サービスの **【最大アドレスオブジェクト数】** プロパティも考慮します。 **【最大アドレスオブジェクト数】** プロパティは、マッピング内で同時に実行できるアドレス検証インスタンスの最大数を決定します。 **【最大アドレスオブジェクト数】** の値は、データ統合サービスの **【最大並行処理】** の値以上である必要があります。

## 並行処理の最大値のオーバーライド

最大並行処理値をオーバーライドするには、マッピングランタイムプロパティで最大並行処理を、2 以上、かつデータ統合サービス用に設定されている値よりも小さい整数値に設定します。

1. マッピングを開きます。
2. **【プロパティ】** ビューで **【ランタイム】** タブをクリックします。
3. **【実行環境】** に **【ネイティブ】** を選択します。
4. **【最大並行処理】** プロパティには、2 以上、かつデータ統合サービス用に設定されている値よりも小さい整数値を入力します。  
あるいは、ユーザー定義のパラメータをプロパティに割り当て、続いてパラメータセットまたはパラメータファイル内でパラメータ値を定義することもできます。
5. マッピングの特定のトランスフォーメーションに提案された並行処理値を定義するには、**【提案された並行処理】** セクションでトランスフォーメーションに対して 1 より大きい整数値を入力します。

次の画像は、トランスフォーメーションにオーバーライドされる [最大並行処理] 値およびデフォルトの [提案された並行処理] 値を持つマッピングを示しています。



6. マッピングを保存します。
7. ID 照合分析に設定されているアドレスバリデータまたは一致トランスフォーメーションに実行インスタンス値を定義するには、次の手順を完了します。
  - a. アドレスバリデータまたは一致トランスフォーメーションを開きます。
  - b. [詳細] ビューで、[実行インスタンス数] プロパティに 1 より大きい整数値を入力します。
  - c. トランスフォーメーションを保存します。

## パーティション化されたマッピングのトラブルシューティング

次に示す状況に対する解決方法は、パーティション化されたマッピングのトラブルシューティングで役に立つことがあります。

データ統合サービスの最大並行処理プロパティが 2 以上の値に設定されていて、マッピングの最大並行処理プロパティが [自動] に設定されている。しかし、マッピングが実行される時にはパーティション化が無効になっている。

複数のパーティションを持つマッピングを実行するためには、データ統合サービスに割り当てられたライセンスにパーティション化が含まれている必要があります。ライセンスにパーティション化が含まれない場合、データ統合サービスは以下のメッセージをマッピングログに書き込み、1 つのスレッドを使用して各マッピングパイプラインステージを処理します。

WARNING: The Integration Service cannot create partitions for the mapping because the license assigned to the Integration Service does not include partitioning.  
 INFO: Partitioning is disabled for the mapping.

## 第 16 章

# Developer tool の命名規則

この章では、以下の項目について説明します。

- [トランスフォーメーションの命名規則, 300 ページ](#)
- [オブジェクトタイプの命名規則, 302 ページ](#)
- [ワークフローオブジェクトの命名規則, 302 ページ](#)

## トランスフォーメーションの命名規則

次の表に、トランスフォーメーションタイプの標準化された命名規則を示します。

トランスフォーメーションのタイプ	推奨される命名規則
関連付け	AST_<トランスフォーメーション名>
アドレスバリデータ	AGG_<トランスフォーメーション名>
アグリゲータ	AGG_<トランスフォーメーション名>
関連付け	AST_<トランスフォーメーション名>
不良レコードの例外	EXC_<トランスフォーメーション名>
大文字小文字変換プログラム	CCO_<トランスフォーメーション名>
分類子	CLA_<トランスフォーメーション名>
比較	CMP_<トランスフォーメーション名>
統合	CNS_<トランスフォーメーション名>
データマスキング	DMK_<トランスフォーメーション名>
データプロセッサ	DPR_<トランスフォーメーション名>
ディシジョン	DEC_<トランスフォーメーション名>
重複レコードの例外	EXC_<トランスフォーメーション名>
式	EXP_<トランスフォーメーション名>

トランスフォーメーションのタイプ	推奨される命名規則
フィルタ	FIL_<トランスフォーメーション名>
Java	JTX_<トランスフォーメーション名>
ジョイナ	JNR_<トランスフォーメーション名>
キージェネレータ	KGN_<トランスフォーメーション名>
ラベラー	LAB_<トランスフォーメーション名>
ルックアップ	LKP_<トランスフォーメーション名>
一致	MAT_<トランスフォーメーション名>
マージ	MRG_<トランスフォーメーション名>
ノーマライザ	NRM_<トランスフォーメーション名>
パーサー	PRS_<トランスフォーメーション名>
ランク	RNK_<トランスフォーメーション名>
読み取り	SRC_<トランスフォーメーション名>
REST プロバイダ	RESTP_<トランスフォーメーション名>
REST Web サービスコンシューマ	RESTWS_<トランスフォーメーション名>
ルーター	RTR_<トランスフォーメーション名>
シーケンスジェネレータ	SEQ_<トランスフォーメーション名>
ソーター	SRT_<トランスフォーメーション名>
SQL	SQL_<トランスフォーメーション名>
標準化	STD_<トランスフォーメーション名>
共有体	UN_<トランスフォーメーション名>
アップデートストラテジ	UPD_<トランスフォーメーション名>
Web サービスコンシューマ	WSC_<トランスフォーメーション名>
加重平均	WAV_<トランスフォーメーション名>
書き込み	WRT_<更新タイプ>_<ターゲット名>

# オブジェクトタイプの命名規則

次の表に、リポジトリオブジェクトの標準化された命名規則を示します。

Developer tool オブジェクト	推奨される命名規則
アプリケーション	APP_<アプリケーション名>
カスタマイズデータオブジェクト	CDO_<データオブジェクト名>
論理データオブジェクト	LDO_<データオブジェクト名>
論理データオブジェクトモデル	LDOM_<モデル名>
マッピング	M_<プロセス>_<ソースシステム>_<ターゲット名>
マップレット	MPLT_<マップレット名>
物理データオブジェクト	PDO_<データオブジェクト名>
プロファイル	PRFL_<プロファイル名>
ルール	Rule_<ルール名>
スコアカード	SCD_<スコアカード名>
SQL データサービス	SDS_<データサービス名>
ターゲット	T_<ターゲット名>
仮想スキーマ	VS_<スキーマ名>
仮想ストアドプロシージャ	VSP_<プロシージャ名>
仮想テーブル	VT_<テーブル名>
Web サービス	WS_<Web サービス名>

# ワークフローオブジェクトの命名規則

ワークフローオブジェクトには、イベント、タスク、ゲートウェイがあります。次の表に、ワークフローオブジェクトの標準化された命名規則を示します。

ワークフローオブジェクト	推奨される命名規則
割り当てタスク	AST_<説明>
コマンドタスク	CMT_<説明>
排他ゲートウェイタスク	EXG_<説明>

ワークフローオブジェクト	推奨される命名規則
ヒューマンタスク	HT_<例外テーブル名>_<説明>
マッピングタスク	MT_<説明>
通知タスク	NTF_<説明>
ワークフロー	WF_<ワークフロー名>

# 索引

## A

application 要素  
パラメータファイル [62](#)

## C

CUME 関数  
パーティション化の制限事項 [293](#)

## E

early selection optimization  
description [245](#)

## F

full optimization level  
description [249](#)

## H

Hadoop  
拒否ファイルディレクトリ [25](#)  
実行環境 [25](#)  
Hive ソース  
SQL クエリのパラメータ [87](#)

## I

IBM DB2 for LUW ソース  
パーティション化 [283](#)  
IBM DB2 for LUW ターゲット  
パーティション化 [289](#)  
IBM DB2 ソース  
プッシュダウンの最適化 [258](#)  
infacmd  
パラメータセットの使用 [81](#)

## J

Java トランスフォーメーション  
パーティション化 [291](#)  
パーティション化の無効化 [293](#)

## M

Microsoft SQL Server ソース  
プッシュダウンの最適化 [258](#)

Microsoft SQL Server ソース (続く)  
プッシュダウンの最適化 [258](#)  
minimal optimization level  
description [249](#)  
MOVINGAVG 関数  
パーティション化の制限事項 [293](#)  
MOVINGSUM 関数  
パーティション化の制限事項 [293](#)

## N

normal optimization level  
description [249](#)

## O

optimization  
early selection optimization method [245](#)  
optimization levels  
description [249](#)  
Oracle ソース  
パーティション化 [283](#)  
プッシュダウンの最適化 [258](#)

## P

performance tuning  
early selection optimization method [245](#)  
optimization levels [249](#)  
PowerCenter からのインポート  
インポートの制限 [240](#)  
インポートパフォーマンス [243](#)  
オブジェクトのインポート [238](#)  
オプション [237](#)  
競合の解決 [226](#)  
システム定義パラメータ [236](#)  
トランスフォーメーションのタイプの変換 [228](#)  
パラメータの変換 [235](#)  
概要 [223](#)  
PowerCenter へのエクスポート  
オブジェクトのエクスポート [218](#)  
オプション [217](#)  
互換性レベルの設定 [215](#)  
トラブルシューティング [222](#)  
パラメータの変換 [216](#)  
リリースの互換性 [215](#)  
ルールおよびガイドライン [221](#)  
概要 [214](#)  
制限 [219](#)  
project 要素  
パラメータファイル [61](#)



## S

SAP HANA ソース  
  プッシュダウンの最適化 [258](#)  
SAP ソース  
  プッシュダウンの最適化 [259](#)  
SQL トランスフォーメーション  
  パーティション化 [291](#)  
  パーティション化の無効化 [293](#)  
SQL 文  
  Hive でのパラメータ [87](#)  
  パラメータ [97](#)  
  ヒント [99](#)  
Sybase ASE ソース  
  プッシュダウンの最適化 [258](#)

## あ

アグリゲータトランスフォーメーション  
  キャッシュのパーティション化 [292](#)  
  パーティション化 [291](#)  
  複数のキャッシュディレクトリ [292](#)  
アドレスバリデータトランスフォーメーション  
  実行インスタンス [297](#)  
安定ソート順  
  パーティション化されたマッピングにおける維持 [295](#)

## い

一致トランスフォーメーション  
  実行インスタンス [297](#)  
インスタンス値  
  パラメータの設定 [54](#), [75](#)

## え

エクスポート  
  PowerCenter [214](#)  
エラー  
  トランスフォーメーション [27](#)  
  ライター [27](#)  
  リーダー [27](#)  
エラー時の停止  
  マッピング [27](#)

## お

オーバーライド  
  トランスフォーメーションパラメータ [67](#)  
  パラメータ [65](#)  
オブジェクト  
  命名規則 [302](#)

## か

カスタムクエリ  
  パラメータ [89](#)  
カスタムマッピング設定  
  作成 [34](#)  
仮想テーブルマッピング  
  パラメータの設定 [58](#)

## き

キャッシュサイズ  
  パーティション [292](#)  
キャッシュディレクトリ  
  最適化 [292](#)  
  複数のディレクトリ [292](#)  
キャッシュのパーティション化  
  キャッシュサイズ [292](#)  
  説明 [292](#)  
行インジケータ  
  拒否ファイル [212](#)

## こ

コストベース最適化  
  説明 [247](#)  
コミット間隔  
  ターゲット [26](#)

## さ

最大並行処理  
  上書き [296](#)  
  説明 [280](#)  
最適化  
  コストベースの最適化方式 [247](#)  
  準結合最適化方式 [248](#)  
  初期プロジェクション最適化方法 [245](#)  
  データシップ結合最適化方式 [247](#)  
  ブランチ刈り込み最適化方式 [246](#)  
  マッピングのパフォーマンスの方式 [245](#)

## し

式  
  パラメータの使用 [89](#)  
  プッシュダウンの最適化 [260](#)  
式トランスフォーメーション  
  動的式 [141](#)  
  パーティション化 [291](#)  
  [マッピング出力] ビュー [103](#)  
システム定義パラメータ  
  インポート [236](#)  
実行インスタンス  
  アドレスバリデータトランスフォーメーション [297](#)  
  一致トランスフォーメーション [297](#)  
実行時のターゲットの作成  
  ルールおよびガイドライン [137](#)  
集計タイプ  
  マッピング出力 [102](#)  
出力式  
  マップレットでの設定 [116](#)  
出力バインディング  
  ダイアログボックスの説明 [112](#)  
[出力] ビュー  
  説明 [102](#)  
出力ファイルディレクトリ  
  最適化 [286](#)  
  複数のディレクトリ [286](#)  
準結合最適化  
  説明 [248](#)  
順序  
  パーティション化されたマッピングにおける維持 [294](#)  
  パーティション化されたマッピングの安定ソート [295](#)

ジョイナトランスフォーメーション  
  キャッシュのパーティション化 [292](#)  
  パーティション化 [291](#)  
  複数のキャッシュディレクトリ [292](#)  
初期プロジェクト最適化  
  説明 [245](#)  
処理スレッド  
  マッピング [279](#)

## す

スコープ  
  ポートセクタ [152](#)  
スレッド  
  処理用マッピング [279](#)

## せ

生成されたポート  
  概要 [139](#)  
  名前の変更 [166](#)  
  並べ替え [166](#)  
生成されたポートの順序変更  
  動的マッピング [149](#)  
  例 [150](#)  
生成されたポートの名前の変更  
  例 [146](#)  
生成されたマップレット  
  概要 [44](#)  
  検証エラー [44](#)  
  作成 [45](#)  
  ルールおよびガイドライン [44](#)  
制約  
  行の更新 [28](#)  
  行の削除 [28](#)  
  行の挿入 [28](#)  
  作成 [40](#)  
  ターゲットのロード順 [28](#)  
セグメント  
  コピー [21](#)  
  マッピング内 [21](#)  
選択条件  
  ポートセクタ [152](#)  
選択ルール  
  動的マッピング [153](#)  
  ポートセクタ [152](#)  
  例 [154](#)

## そ

ソース  
  パーティション化されたフラットファイル [282](#)  
  パーティション化されたリレーショナル [283](#)  
ソータートランスフォーメーション  
  キャッシュのパーティション化 [292](#)  
ソートリストパラメータ  
  説明 [99](#)  
ソート順  
  パーティション化されたマッピングにおける維持 [294](#)

## た

ターゲット  
  パーティション化されたフラットファイル [285](#)

ターゲット (続く)  
  パーティション化されたリレーショナル [289](#)  
  マージファイル [286](#)  
ターゲットロード順の制約  
  作成 [40](#)  
  説明 [28](#)  
  ルールおよびガイドライン [29](#)  
  例 [29](#)

## て

提案された並行処理  
  トランスフォーメーション [296](#)  
ディシジョントランスフォーメーション  
  パーティション化 [291](#)  
  パーティション化の無効化 [293](#)  
データオブジェクト  
  実行時のカラムの取得 [130](#)  
データシップ結合最適化  
  説明 [247](#)  
データオブジェクト操作  
  説明 [16](#)  
デフォルトのマッピング設定  
  作成 [34](#)

## と

同時読み込みのパーティション化  
  説明 [282](#)  
動的式  
  概要 [141](#)  
  作成 [171](#)  
  パラメータ [142](#)  
  例 [141](#)  
  式パラメータ [142](#)  
動的ポート  
  概要 [139](#)  
  作成 [165](#)  
  設定 [166](#)  
動的ポートおよび生成されたポート  
  設定 [140](#)  
  動的マッピング  
    トランスフォーメーション [140](#)  
動的マッピング  
  ポートセクタ [153](#)  
  ポートを含めるまたは除外する [144](#)  
  概要 [125](#)  
  生成されたポート [139](#)  
  生成されたポートの順序変更 [149](#)  
  生成されたポートの名前の変更 [146](#)  
  設計時リンク [154](#)  
  設定の概要 [126](#)  
  ソース [129](#)  
  ソースデータオブジェクトのパラメータ化 [132](#)  
  ソースポート名のリストア [148](#)  
  ソース名のパラメータ化 [131](#), [132](#)  
  ターゲットオブジェクト [133](#)  
  ターゲットデータオブジェクトのパラメータ化 [138](#)  
  ターゲット名のパラメータ化 [138](#)  
  動的ポート [139](#)  
  動的ポートの作成 [165](#)  
  トラブルシューティング [159](#)  
  入力ルール [143](#), [145](#)  
  入力ルールの定義 [166](#)  
  残りのすべてのポートを含める [145](#)  
  パラメータ [128](#)

## 動的マッピング (続く)

- ポートおよびリンク [127](#)
- ランタイムリンク [156](#), [157](#)
- ランタイムリンクの作成 [178](#)
- リンクの解決 [156](#)
- 開発と実行 [162](#)
- 検査 [181](#)
- 実行 [182](#)
- 書き込みトランスフォーメーションの設定 [174](#)
- 選択ルール [153](#)
- 入力ルール [143](#), [145](#)

## 動的マッピングのコンポーネント

- データソース [126](#)
- ルール [128](#)

## 動的マッピングのルール

- 概要 [128](#)

## トラブルシューティング

- PowerCenter へのオブジェクトのエクスポート [222](#)

## 動的マッピング [159](#)

## トランスフォーメーション

- 説明 [20](#)
- 提案された並行処理 [27](#), [296](#)
- パーティション化 [291](#)
- 命名規則 [300](#)

# に

## 入力リンクセットパラメータ

- 説明 [99](#)

# ね

## ネスト

- 式パラメータ [94](#)

# は

## パーティション化の制限事項

- Java トランスフォーメーション [291](#)
- SQL トランスフォーメーション [291](#)
- アグリゲータトランスフォーメーション [291](#)
- 式トランスフォーメーション [291](#)
- ジョイナトランスフォーメーション [291](#)
- 数値関数 [293](#)
- ディシジョントランスフォーメーション [291](#)
- リンクトランスフォーメーション [291](#)
- リレーショナルソース [284](#), [285](#)
- リレーショナルターゲット [290](#)

## パーティションポイント

- 説明 [279](#)

## パイプラインステージ

- 説明 [279](#)

## パイプラインのマッピング

- 説明 [279](#)

## バインディング

- マッピングへのマップレット出力 [112](#), [114](#)

## パーティション化

- IBM DB2 for LUW ソース [283](#)
- IBM DB2 for LUW ターゲット [289](#)
- Java トランスフォーメーション [291](#)
- Oracle ソース [283](#)
- SQL トランスフォーメーション [291](#)
- アグリゲータトランスフォーメーション [291](#)
- アドレスバリデータトランスフォーメーション [297](#)
- 安定ソート順の維持 [295](#)

## パーティション化 (続く)

- 一致トランスフォーメーション [297](#)
- キャッシュ [292](#)
- キャッシュサイズ [292](#)
- 行順序を保持 [294](#)
- 拒否ファイル [211](#)
- 最大並行処理 [280](#)
- 式トランスフォーメーション [291](#)
- ジョイナトランスフォーメーション [291](#)
- ディシジョントランスフォーメーション [291](#)
- 同時読み取り [282](#)
- トラブルシューティング [299](#)
- トランスフォーメーション [291](#)
- フラットファイルソース [282](#)
- フラットファイルターゲット [285](#)
- マージされたファイルターゲット [286](#)
- マッピング [280](#)
- マッピング用に減らす [295](#)
- リンクトランスフォーメーション [291](#)
- リレーショナル接続タイプ [284](#), [290](#)

## パフォーマンスのチューニング

- コストベースの最適化方式 [247](#)
- 最適化方式 [245](#)
- 述部最適化方式 [246](#)
- 準結合最適化方式 [248](#)
- 初期プロジェクション最適化方法 [245](#)
- データシップ結合最適化方式 [247](#)
- プッシュダウンの最適化 [252](#)
- ブランチ刈り込み最適化方式 [246](#)

## パラメータ

- リレーショナルテーブルのプロパティ [132](#)
- PowerCenter からのインポート [235](#)
- PowerCenter へのエクスポート [216](#)
- SQL 文 [97](#), [99](#)
- インスタンス値 [75](#)
- カスタムクエリ [89](#)
- 仮想テーブルマッピング [58](#)
- 式 [89](#)
- ソースデータオブジェクト [132](#)
- ターゲットデータオブジェクト [138](#)
- テーブル名とリソース [96](#)
- 動的マッピング [128](#)
- フィルタ条件 [89](#), [98](#)
- フラットファイルソース [131](#)
- フラットファイルの区切り文字 [95](#)
- ポート式 [91](#)
- マップレット [55](#)
- 論理データオブジェクト [57](#)
- 圧縮コーデック [86](#)
- 圧縮形式 [86](#)
- 結合条件 [98](#)
- 設定方法 [68](#)

## パラメータ

- 日付フォーマット [52](#)
- パラメータインスタンス値
- 設定 [54](#)
- パラメータセット
- 作成 [76](#)
- 概要 [59](#)

## パラメータのオーバーライド

- パラメータ階層の使用 [65](#)
- マッピング内 [66](#)

## パラメータのバインド

- インスタンス値 [75](#)
- パラメータ階層の使用 [65](#)

## パラメータファイル

- application 要素 [62](#)
- Developer tool からのエクスポート [64](#)

パラメータファイル (続く)

- project 要素 [61](#)
- XML スキーマ定義 [60](#)
- 作成 [64](#)
- サンプル [63](#)
- ヒント [62](#)
- マッピングの実行 [60](#), [81](#)
- ルール [62](#)
- 構造 [60](#)
- 目的 [60](#)

パラメータ階層 [65](#)

## ひ

日付パラメータ

- 有効な形式 [52](#)

非リレーショナルソース

- プッシュダウンの最適化 [258](#)

## ふ

フィルタ条件

- パラメータ [98](#)

プッシュダウンの最適化

- リレーショナルソース [256](#), [258](#)
- Greenplum ソース [258](#)
- IBM DB2 ソース [258](#)
- Microsoft SQL Server ソース [258](#)
- ODBC ソース [258](#)
- Oracle ソース [258](#)
- SAP HANA ソース [258](#)
- SAP ソース [259](#)
- Sybase ASE ソース [258](#)
- z/OS の非リレーショナルソース [258](#)
- 概要 [252](#)
- 式 [260](#)
- ソース [256](#)
- プッシュダウンタイプ [253](#)
- リレーショナルソース [256](#), [258](#)
- 演算子 [276](#)
- 関数 [260](#)

プッシュダウンの最適化方式

- ソースプッシュダウン [254](#)
- プッシュダウンの設定 [254](#)
- 完全なプッシュダウン [253](#)

フラットファイルソース

- パーティション化 [282](#)

フラットファイルターゲット

- パーティション化 [285](#)
- パーティションの統合 [286](#)
- 複数の出力ディレクトリ [286](#)
- 拒否ファイル [211](#)

フラットファイルデータオブジェクト

- 拒否ファイル [211](#)

フラットファイルの区切り文字

- パラメータの使用 [95](#)

ブランチ刈り込み最適化

- 説明 [246](#)

## へ

並行処理

- 最大 [26](#), [27](#)
- 推奨 [27](#)
- マッピング用に減らす [295](#)

## ほ

ポート

- 接続の検証 [23](#)

ポートセレクト

- 選択ルール [152](#), [153](#)
- 作成 [170](#)
- 説明 [152](#)
- 選択ルール [152](#), [153](#)
- 動的式で [141](#)
- 例 [154](#)

ポートのプレビュー

- ポートセレクト [152](#)

ポートリストパラメータ

- 説明 [99](#)

## ま

マッピング

- SQL クエリから [118](#)
- オブジェクト [15](#)
- オブジェクトの検証 [23](#)
- オブジェクトの接続 [39](#)
- オブジェクトの追加 [38](#)
- 概要 [14](#)
- 偽装 [27](#)
- 最大並行処理 [280](#)
- 最適化方式 [245](#)
- 作成 [38](#)
- 式の検証 [23](#)
- 述部最適化方式 [246](#)
- 処理スレッド [279](#)
- 制約の作成 [40](#)
- 接続の検証 [23](#)
- ターゲットロード順の制約 [28](#)
- タブ [21](#)
- パーティション化 [280](#)
- パーティションポイント [279](#)
- パイプライン [279](#)
- パラメータ [53](#)
- パラメータの検証 [23](#)
- 表示 [21](#)
- 並行処理の減少 [295](#)
- マッピング設定 [31](#), [32](#)
- ランタイムプロパティ [24](#)
- ワークフロー [38](#)
- 開発 [38](#)
- 拒否ファイル [211](#)
- 検証 [22](#), [41](#)
- 最適化されたマッピング [249](#)
- 実行中 [41](#)
- 詳細オプション [35](#), [37](#)
- 同期の検証 [23](#)

マッピング、動的

- トラブルシューティング [159](#)

マッピング管理

- 概要 [208](#)

マッピング偽装

- ユーザー名 [27](#)

マッピングセグメント

- 説明 [21](#)

マッピングパラメータ

- infacmd [81](#)
- 概要 [49](#)
- 仮想テーブルマッピング [58](#)
- システム [50](#)
- タイプ [51](#)

マッピングパラメータ (続く)

ユーザー定義 [51](#)

割り当て先 [82](#)

作成する場所 [53](#)

マッピングの検証

パラメータを使用

前提条件 [79](#)

マッピングの実行

パラメータを使用

Developer tool [78](#), [80](#)

infacmd [81](#)

コマンドライン [81](#)

マッピングパラメータとして公開

説明 [54](#), [55](#)

タスクの説明 [74](#)

マッピングビュー

説明 [21](#)

マッピング出力

システム定義 [101](#)

集計タイプ [102](#)

出力式 [103](#)

[出力] ビュー [102](#)

定義 [102](#)

マプレットへのバインディング [110](#)

ユーザー定義 [102](#)

論理データオブジェクト [114](#)

概要 [101](#)

マッピング設定

プロパティ [32](#)

作成 [34](#)

マプレット

およびルール仕様 [46](#)

概要 [42](#)

検証 [48](#)

生成 [44](#)

説明 [20](#)

タイプ [43](#)

パラメータ

オーバーライド [67](#)

パラメータの使用 [55](#)

ルール [48](#)

マプレット出力

定義 [115](#)

マッピング出力へのバインディング [114](#)

マッピングへのバインディング [110](#)

マッピングへのバインド [112](#)

マッピングへのバインド方法 [116](#)

マプレットパラメータ

例 [56](#)

マプレット

PowerCenter へのエクスポート [215](#)

作成 [48](#)

出力 [44](#)

入力 [43](#)

## め

命名規則

オブジェクトタイプ [302](#)

トランスフォーメーション [300](#)

## も

文字列パラメータ

SQL 文 [98](#)

文字列パラメータ (続く)

精度の制限 [51](#)

## ゆ

ユーザー名

マッピング偽装 [27](#)

## ら

リンクトランスフォーメーション

キャッシュのパーティション化 [292](#)

パーティション化 [291](#)

複数のキャッシュディレクトリ [292](#)

ランタイムリンク

概要 [156](#)

動的マッピング [157](#)

リンクポリシー [157](#)

例 [159](#)

作成 [178](#)

## り

リレーショナルソース

パーティション化 [283](#)

パーティション化の制限事項 [284](#), [285](#)

ブッシュダウンの最適化 [258](#)

リレーショナルターゲット

パーティション化 [289](#)

パーティション化の制限事項 [290](#)

拒否ファイル [211](#)

リレーショナルデータオブジェクト

パーティション化 [283](#), [289](#)

拒否ファイル [211](#)

リンクの解決

動的マッピング [156](#)

## る

ルールおよびガイドライン

実行時のターゲットの作成 [137](#)

ルール仕様 [46](#)

ルックアップトランスフォーメーション

キャッシュのパーティション化 [292](#)

## れ

例

生成されたポートの順序変更 [150](#)

生成されたポートの名前の変更 [146](#)

動的式 [141](#)

動的マッピングの選択ルール [154](#)

ランタイムリンク [159](#)

## ろ

ロード順

制約 [28](#)

論理データオブジェクト

パラメータの使用 [57](#)

マッピング出力 [114](#)