置換メソッド・リファレンス

このマニュアルは、**Oracle Data Integrator 置換メソッド**のリファレンス・マニュアルです。対象読者は、統合シナリオで、メソッドを使用して汎用のナレッジ・モジュールやプロシージャを作成する高度な開発者です。

このマニュアルの構成

このマニュアルには次の章が含まれています。

- **第 1 章「置換メソッドの使用」**では、置換メソッドを使用する場合と方法について説明します。
- **第 2 章「置換メソッド・リファレンス」**では、各メソッドの構文について詳しく説明します。

置換メソッドの使用

原則

ナレッジ・モジュールとプロシージャからアクセス可能なメソッドは、Java™で実装されている Oracle Data Integrator メソッドのダイレクト・コールです。これらのメソッドは通常、Oracle Data Integrator リポジトリに格納されているメタデータに対応するテキストの生成に使用されます。

一般構文

置換メソッドは、ナレッジ・モジュールまたはプロシージャのタスクの任意のテキストで使用されます。使用する構文は次のとおりです。

ここで

- 任意のテキスト: 使用するテクノロジの言語で記述された、タスクに関する任意のテキスト
- `Java 式`: 文字列の作成を可能にする任意の Java 式

Java 式の例:

```
odiRef.getTableName("WORK_TABLE") + "FUTURE"
```

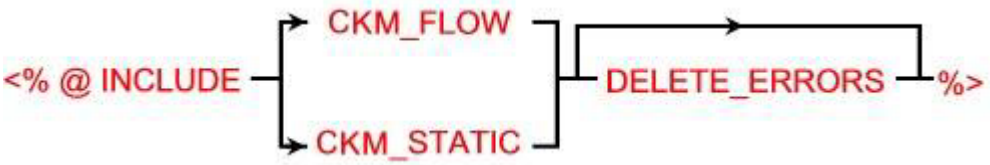
Oracle Data Integrator API は Java クラス OdiReference で実装されています。そのインスタンスである OdiRef はいつでも使用できます。このため、Data Integrator メソッド getFrom()をコールするには、odiRef.getFrom()と記述する必要があります。

注意: 前の構文 **snpRef<.method_name>**はまだサポートされていますが、廃止予定です。

CKM 固有の構文

次の構文は、チェック・プロシージャ（CKM）の実行をコールするために IKM で使用されます。

この構文は、処理のこの段階で、すべての CKM プロシージャ・コマンドを自動的にインクルードします。



この構文のオプションは次のとおりです。

- CKM_FLOW: インタフェースの「制御」タブで行った CKM の選択に応じて、フロー制御をトリガーします。
- CKM_STATIC: ターゲット・データストアの静的管理をトリガーします。データストアに対して静的な制約として定義および選択された制約がチェックされます。
- DELETE_ERRORS: このオプションを使用すると、検出されたエラーが自動的に抑制されます。

フレックスフィールドの使用

フレックスフィールドは、Oracle Data Integrator のオブジェクトのプロパティをカスタマイズ可能にするユーザー定義フィールドです。フレックスフィールドはオブジェクト・ウィンドウの**「フレックスフィールド」**タブで定義され、オブジェクト・ウィンドウの**「フレックスフィールド」**タブを使用して各オブジェクトに対して設定することができます。

Oracle Data Integrator の置換メソッドを使用してオブジェクトにアクセスする場合、フレックスフィールド・**コード**を指定すると、Oracle Data Integrator は、その**コード**を、オブジェクト・インスタンスのフレックスフィールド値で置き換えます。

例:

`<%=odiRef.getTable("L", "MY_DATASTORE_FIELD", "W")%>`の場合、現在の表のフレックスフィールド `MY_DATASTORE_FIELD` の値が返されます。

`<%=odiRef.getSrcTableList("", "[MY_DATASTORE_FIELD] ", ", ", "")%>`の場合、インタフェースの各ソース表のフレックスフィールド値が返されます。

フレックスフィールド値は、 getFlexFieldValue()メソッドを使用して取得することもできます。

重要: フレックスフィールドは、特定のオブジェクト・タイプにのみ存在します。「フレックスフィールド」タブがないオブジェクトではサポートされていません。

置換メソッド・リファレンス

置換メソッドのリスト

グローバル・メソッド

- getCatalogName
- getCatalogNameDefaultPSchema
- getColDefaultValue
- getContext
- getDataType
- getFlexFieldValue
- getInfo
- getJDBCConnection
- getObjectName
- getObjectNameDefaultPSchema
- getOption
- getPrevStepLog
- getSchemaName
- GetSchemaNameDefaultPSchema
- getSession
- getSessionVarList
- getStep
- getSysDate
- getUserExit
- setNbDelete
- setNbErrors
- setNbInsert
- setNbRows
- setNbUpdate

ジャーナル化メソッド

- getJrnFilter
- getJrnInfo
- getSubscriberList
- getTable
- getColList

ロード・メソッド

- getColList
- getFilter
- getFilterList
- getFrom
- getGrpBy
- getGrpByList
- getHaving
- getHavingList

- getJoin
- getJoinList
- getJRNFilter
- getJrnInfo
- getPop
- getSrcColList
- getSrcTablesList
- getTable
- getTargetTable
- getTargetColList

チェック・メソッド

- getAK
- getAKColList
- getCK
- getColList
- getFK
- getFKColList
- getNotNullCol
- getPK
- getPKColList
- getPop
- getTable
- getTargetTable
- getTargetColList

統合メソッド

- getColList
- getFilter
- getFilterList
- getFrom
- getGrpBy
- getGrpByList
- getHaving
- getHavingList
- getJoin
- getJoinList
- getJRNFilter
- getJrnInfo
- getPop
- getSrcColList
- getSrcTablesList
- getTable
- getTargetTable
- getTargetColList

リバース・エンジニアリング・メソッド

- getModel

Web サービス・メソッド

- hasPK
- nextAK
- nextCond
- nextFK

アクション・メソッド

詳細は、「アクションでの置換メソッドの使用」を参照してください。

- getAK
- getAKColList
- getCK
- getColList
- getColumn
- getFK
- getFKColList
- getIndex
- getIndexColList
- getNewColComment
- getNewTableComment
- getPK
- getPKColList
- getTable
- getTargetTable
- isColAttrChanged

グローバル・メソッド

getCatalogName()メソッド

使用方法

```
public java.lang.String getCatalogName(
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getCatalogName(
java.lang.String pLogicalSchemaName,
java.lang.String pContextCode,
java.lang.String pLocation)
public java.lang.String getCatalogName(
java.lang.String pLocation)
public java.lang.String getCatalogName()
```

説明

物理データ・カタログまたは作業カタログの名前を論理スキーマから取得するために使用します。

最初の構文が使用される場合、返されるカタログ名は現在のコンテキストに一致します。

2 番目の構文が使用される場合、返されるカタログ名は、pContextCode パラメータで指定されたコンテキストのカタログ名です。

3 番目の構文は、現在の論理スキーマ、現在のコンテキストでのデータ・カタログ（D）または作業カタログ（W）の名前を返します。

4 番目の構文は、現在のコンテキスト、現在の論理スキーマでのデータ・カタログ（D）の名前を返します。

パラメータ

パラメータ	タイプ	説明	
pLogicalSchemaName	文字列	論理スキーマの名前。	
pContextCode	文字列	強制適用されたスキーマのコンテキストのコード。	
pLocation	文字列	"W"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマの作業カタログを返します。
		"D"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマのデータ・カタログを返します。

例

定義されている物理スキーマ:
`Pluton.db_odi.dbo`

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner

この物理スキーマに関連付けられているもの: コンテキスト CTX_DEV の MSSQL_ODI

コール対象	戻り値
`<%=odiRef.getCatalogName("MSSQL_ODI", "CTX_DEV", "W")%>`	tempdb
`<%=odiRef.getCatalogName("MSSQL_ODI", "CTX_DEV", "D")%>`	db_odi

getCatalogNameDefaultPSchema()メソッド

使用方法

```
public java.lang.String getCatalogNameDefaultPSchema(
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getCatalogNameDefaultPSchema(
java.lang.String pLogicalSchemaName,
java.lang.String pContextCode,
java.lang.String pLocation)
public java.lang.String getCatalogNameDefaultPSchema(
java.lang.String pLocation)
public java.lang.String getCatalogNameDefaultPSchema()
```

説明

タプルに対応している物理スキーマ（論理スキーマ、コンテキスト）が関連付けられているデータ・サーバーの**デフォルト**の物理データ・カタログまたは作業カタログの名前を取得できます。コンテキストを指定しない場合、現在のコンテキストが使用されます。論理スキーマ名が指定されない場合、現在の論理スキーマが使用されます。pLocation が指定されない場合、データ・カタログが返されます。

パラメータ

パラメータ	タイプ	説明	
pLogicalSchemaName	文字列	論理スキーマの名前。	
pContextCode	文字列	強制適用されたスキーマのコンテキストのコード。	
pLocation	文字列	"W"	タプル（コンテキスト、論理スキーマ）に対応している物理スキーマもアタッチされているデータ・サーバーに関連付けられている、デフォルトの物理スキーマの作業カタログを返します。
		"D"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマのデータ・カタログを返します。

例

定義されている物理スキーマ:
`Pluton.db_odi.dbo`

データ・カタログ:	db_odi
データ・スキーマ:	dbo

作業カタログ:	tempdb
作業スキーマ:	temp_odi
デフォルト・スキーマか	**はい**

この物理スキーマに関連付けられているもの: コンテキスト CTX_DEV の MSSQL_ODI および Pluton.db_doc.doc

データ・カタログ:	db_doc
データ・スキーマ:	doc
作業カタログ:	tempdb
作業スキーマ:	temp_doc
デフォルト・スキーマか	**いいえ**

この物理スキーマに関連付けられているもの: コンテキスト CTX_DEV の MSSQL_DOC

コール対象	戻り値
`<%=odiRef.getCatalogNameDefaultPSchema("MSSQL_DOC", "CTX_DEV", "W")%>`	tempdb
`<%=odiRef.getCatalogNameDefaultPSchema("MSSQL_DOC", "CTX_DEV", "D")%>`	db_odi

getColDefaultValue() メソッド

使用方法

```
public java.lang.String getColDefaultValue()
```

説明

マッピングのターゲット列のデフォルト値を返します。

このメソッドはマッピング式で、<%%>タグなしで使用できます。このメソッド・コールは、生成されたコードに、列定義で設定したデフォルト値を挿入します。列タイプによっては、この値を引用符で囲みます。

例

'ターゲット列のデフォルト値は' +**'odiRef.getColDefaultValue()'**

getContext()メソッド

使用方法

```
public java.lang.String getContext(java.lang.String pPropertyName)
```

説明

このメソッドは現在の実行コンテキストに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	コンテキストの内部 ID。
CTX_NAME	コンテキストの名前。
CTX_CODE	コンテキストのコード。
CTX_DEFAULT	デフォルト・コンテキストなら 1、他のコンテキストの場合は 0 を返します。
<flexfield code>	この参照のフレックスフィールド値。

例

```
Current Context = <%=getContext("CTX_NAME")%>
```

getDataType()メソッド

使用方法

```
public java.lang.String getDataType(
java.lang.String pDataTypeName,
java.lang.String pDataTypeLength,
java.lang.String pDataTypePrecision)
```

説明

ソース・テクノロジおよびターゲット・テクノロジに関連付けられたパラメータに応じて、SQL データ型として varchar、数値または日付を使用する作成構文を返します。

パラメータ

パラメータ	タイプ	説明
pDataTypeName	文字列	次の表にリストされているデータ型の名前
pDataTypeLength	文字列	データ型の長さ
pDataTypePrecision	文字列	データ型の精度

次の表は、pDataTypeName で可能なすべての値のリストです。

値	説明
SRC_VARCHAR	ソースの varchar データ型に対応する構文を返します。
SRC_NUMERIC	ソースの数値データ型に対応する構文を返します。
SRC_DATE	ソースの日付データ型に対応する構文を返します。
DEST_VARCHAR	ターゲットの varchar データ型に対応する構文を返します。
DEST_NUMERIC	ターゲットの数値データ型に対応する構文を返します。
DEST_DATE	ターゲットの日付データ型に対応する構文を返します。

例

次のテクノロジの構文が次のようであるとします。

テクノロジ	Varchar	数値	日付
Oracle	varchar2(%L)	number(%L,%P)	date
MS SqlServer	varchar(%L)	numeric(%L,%P)	datetime
MS Access	Text(%L)	double	datetime

getDataType へのコールの例を次に示します。

コール	Oracle	MS SqlServer	MS Access
`<%=odiRef.getDataType("DEST_VARCHAR", "10", "")%>`	varchar2(10)	varchar(10)	Text(10)
`<%=odiRef.getDataType("DEST_VARCHAR", "10", "5")%>`	varchar2(10)	varchar(10)	Text(10)
`<%=odiRef.getDataType("DEST_NUMERIC", "10", "")%>`	number(10)	numeric(10)	double
`<%=odiRef.getDataType("DEST_NUMERIC", "10", "2")%>`	number(10,2)	numeric(10,2)	double

<%=odiRef.getDataType("DEST_NUMERIC", "", "")%>	number	numeric	double
<%=odiRef.getDataType("DEST_DATE", "", "")%>	date	datetime	datetime
<%=odiRef.getDataType("DEST_DATE", "10", "2")%>	date	datetime	datetime

getFlexFieldValue メソッド

使用方法

```
public java.lang.String getFlexFieldValue(java.lang.String pI_Instance,
java.lang.String pI_Object, java.lang.String pFlexFieldCode)
```

説明

このメソッドは、オブジェクト・インスタンスのフレックスフィールドの値を返します。

パラメータ

パラメータ	タイプ	説明
pI_Instance	文字列	オブジェクト・インスタンス・ウィンドウの**「バージョン」**タブに表示される、オブジェクト・インスタンスの**内部 ID**
pI_Object	文字列	そのオブジェクト・タイプの**オブジェクト**・ウィンドウの**「バージョン」**タブに表示される、オブジェクト・タイプの**内部 ID**
pPropertyName	文字列	値が返されるフレックスフィールド・コード

例

<%=odiRef.getFlexFieldValue("32001","2400","MY_DATASTORE_FIELD")%>

タイプが datastore（datastore の内部 ID は 2400）で、内部 ID が 32001 のオブジェクト・インスタンスの、フレックスフィールド MY_DATASTORE_FIELD の値を返します。

getJDBCConnection()メソッド

使用方法

```
java.sql.Connection getJDBCConnection(
java.lang.String pPropertyName)
```

説明

このメソッドは、現在のタスクのソースまたはターゲットの JDBC 接続を返します。

警告: このメソッドは文字列ではなく、JDBC 接続オブジェクトを返します。このオブジェクトはタスク内の Java コードに使用されることがあります。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	返される接続の名前。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
SRC	現在のタスクのソース接続。
DEST	現在タスクのターゲット接続。

例

この接続のソース接続を取得して、文を作成します。

```
java.sql.Connection sourceConnection = odiRef.getJDBCConnection("SRC");
java.sql.Statement s = sourceConnection.createStatement();
```

getInfo()メソッド

使用方法

```
public java.lang.String getInfo(java.lang.String pPropertyName)
```

説明

現在のタスクに関する一般情報を返す汎用メソッド。使用可能な情報のリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
I_SRC_SET	タスクがロード・ナレッジ・モジュールに所属する場合の、現在のソース・セットの内部識別子。
SRC_SET_NAME	タスクがロード・ナレッジ・モジュールに所属する場合の、現在のソース・セットの名前。
COLL_NAME	タスクがロード・ナレッジ・モジュールに所属する場合の、現在のロード・リソースの名前（C$）。
INT_NAME	タスクが文字列ロード、統合またはチェック・ナレッジ・モジュールに所属する場合の、現在の統合リソースの名前（I$）。
ERR_NAME	タスクがロード、統合またはチェック・ナレッジ・モジュールの一部である場合の、現在のエラー・リソースの名前（E$）。
TARG_NAME	タスクがロード、統合またはチェック・ナレッジ・モジュールの一部である場合の、ターゲット・リソースの名前（E$）。
SRC_CATALOG	ソース環境のデータ・カタログの名前。
SRC_SCHEMA	ソース環境のデータ・スキーマの名前。
SRC_WORK_CATALOG	ソース環境の作業カタログの名前。
SRC_WORK_SCHEMA	ソース環境の作業スキーマの名前。
DEST_CATALOG	ターゲット環境のデータ・カタログの名前。
DEST_SCHEMA	ターゲット環境のデータ・スキーマの名前。
DEST_WORK_CATALOG	ターゲット環境の作業カタログの名前。
DEST_WORK_SCHEMA	ターゲット環境の作業スキーマの名前。
SRC_TECHNO_NAME	ソース・テクノロジの名前。
SRC_CON_NAME	ソース接続の名前。
SRC_DSERV_NAME	ソース・マシンのデータ・サーバーの名前。
SRC_CONNECT_TYPE	ソース・マシンの接続タイプ。
SRC_IND_JNDI	JNDI URL フラグ
SRC_JAVA_DRIVER	ソース接続の JDBC ドライバの名前。
SRC_JAVA_URL	ソース接続の JDBC URL。
SRC_JNDI_AUTHENT	JNDI 認証タイプ。
SRC_JNDI_PROTO	JNDI ソース・プロトコル。

SRC_JNDI_FACTORY	JNDI ソース・ファクトリ。
SRC_JNDI_URL	ソース JNDI URL。
SRC_JNDI_RESSOURCE	アクセスされたソース JNDI リソース。
SRC_USER_NAME	ソース接続のユーザー名。
SRC_ENCODED_PASS	ソース接続の暗号化されたパスワード。
SRC_FETCH_ARRAY	ソース配列フェッチのサイズ。
SRC_BATCH_UPDATE	ソース・バッチ更新のサイズ。
SRC_EXE_CHANNEL	ソース接続の実行チャネル。
SRC_COL_ALIAS_WORD	ソース・テクノロジで列を別名と区別するために使用される語。
SRC_TAB_ALIAS_WORD	ソース・テクノロジで表を別名と区別するために使用される語。
SRC_DATE_FCT	ソース・テクノロジで現在の日付を返す関数。
SRC_DDL_NULL	ソースでの表の作成時にキーワード NULL に対して使用する定義を返します。
SRC_MAX_COL_NAME_LEN	ソース・テクノロジでの列名の最大文字数。
SRC_MAX_TAB_NAME_LEN	ソース・テクノロジでの表名の最大文字数。
SRC_REM_OBJ_PATTERN	ソース・テクノロジでのリモート・オブジェクトの置換モデル。
SRC_LOC_OBJ_PATTERN	ソース・テクノロジでのローカル・オブジェクト名の置換モデル。
DEST_TECHNO_NAME	ターゲット・テクノロジの名前。
DEST_CON_NAME	ターゲット接続の名前。
DEST_DSERV_NAME	ターゲット・マシンのデータ・サーバーの名前。
DEST_CONNECT_TYPE	ターゲット・マシンの接続タイプ。
DEST_IND_JNDI	ターゲット JNDI URL フラグ。
DEST_JAVA_DRIVER	ターゲット接続の JDBC ドライバの名前。
DEST_JAVA_URL	ターゲット接続の JDBC URL。
DEST_JNDI_AUTHENT	ターゲットの JNDI 認証タイプ。

DEST_JNDI_PROTO	JNDI ターゲット・プロトコル。
DEST_JNDI_FACTORY	JNDI ターゲット・ファクトリ。
DEST_JNDI_URL	ターゲットの JNDI URL。
DEST_JNDI_RESSOURCE	アクセスされるターゲット JNDI リソース。
DEST_USER_NAME	ターゲット接続用のユーザーの名前。
DEST_ENCODED_PASS	ターゲット接続の暗号化されたパスワード。
DEST_FETCH_ARRAY	ターゲット配列フェッチのサイズ。
DEST_BATCH_UPDATE	ターゲット・バッチ更新のサイズ。
DEST_EXE_CHANNEL	ターゲット接続の実行チャネル。
DEST_COL_ALIAS_WORD	ターゲット・テクノロジで列を別名と区別するために使用される語。
DEST_TAB_ALIAS_WORD	ターゲット・テクノロジで表を別名と区別するために使用される語。
DEST_DATE_FCT	ターゲット・テクノロジで現在の日付を返す関数。
DEST_DDL_NULL	ターゲットでの表の作成時にキーワード NULL に対して使用する定義を返す関数。
DEST_MAX_COL_NAME_LEN	ターゲット・テクノロジでの列の最大文字数。
DEST_MAX_TAB_NAME_LEN	ターゲット・テクノロジでの表名の最大文字数。
DEST_REM_OBJ_PATTERN	ターゲット・テクノロジでのリモート・オブジェクトの置換モデル。
DEST_LOC_OBJ_PATTERN	ターゲット・テクノロジでのローカル・オブジェクト名の置換モデル。
CT_ERR_TYPE	エラー・タイプ（F: フロー、S: 静的）。チェック・ナレッジ・モジュールの場合にのみ適用されます。
CT_ERR_ID	エラー識別子（静的管理では表番号、フロー制御ではインタフェース番号）。チェック・ナレッジ・モジュールの場合にのみ適用されます。
CT_ORIGIN	エラー元を識別する名前（静的管理の表名、またはプロジェクト・コードを接頭辞として付けたインタフェース）。チェック・ナレッジ・モジュールの場合にのみ適用されます。
JRN_NAME	ジャーナル化されたデータストアの名前。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューの名

	前。
JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューの名前。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーの名前。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーの名前。
JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーの名前。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーの名前。
SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアの名前。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
SRC_DEF_CATALOG	ソース・データ・サーバー用のデフォルト・カタログ。
SRC_DEF_SCHEMA	ソース・データ・サーバー用のデフォルト・スキーマ。
SRC_DEFW_CATALOG	ソース・データ・サーバー用のデフォルト作業カタログ。
SRC_DEFW_SCHEMA	ソース・データ・サーバー用のデフォルト作業スキーマ。
DEST_DEF_CATALOG	ターゲット・データ・サーバー用のデフォルト・カタログ。
DEST_DEF_SCHEMA	ターゲット・データ・サーバー用のデフォルト・スキーマ。
DEST_DEFW_CATALOG	ターゲット・データ・サーバー用のデフォルト作業カタログ。
DEST_DEFW_SCHEMA	ターゲット・データ・サーバー用のデフォルト作業スキーマ。
SRC_LSCHEMA_NAME	ソース論理スキーマ名。
DEST_LSCHEMA_NAME	ターゲット論理スキーマ名。

例

現在のソース接続: **<%=odiRef.getInfo("SRC_CON_NAME")%>**
サーバー: **<%=odiRef.getInfo("SRC_DSERV_NAME")%>**

getObjectName()メソッド

使用方法

```
public java.lang.String getObjectName(
java.lang.String pMode,
java.lang.String pObjectName,
java.lang.String pLocation)
public java.lang.String getObjectName(
java.lang.String pMode,
java.lang.String pObjectName,
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getObjectName(
java.lang.String pMode,
java.lang.String pObjectName,
java.lang.String pLogicalSchemaName,
java.lang.String pContextName,
java.lang.String pLocation)
public java.lang.String getObjectName(
java.lang.String pObjectName,
java.lang.String pLocation)
public java.lang.String getObjectName(
java.lang.String pObjectName)
```

説明

カタログとスキーマを含む、物理オブジェクトのフルネームを返します。pMode パラメータは、使用する置換マスクを示します。

最初の構文は、現在のコンテキストでの現在の論理スキーマに応じて、オブジェクト名を作成します。

2 番目の構文は、現在のコンテキストで、pLogicalSchemaName パラメータに示された論理スキーマに応じて、オブジェクトの名前を作成します。

3 番目の構文は、pLogicalSchemaName および pContextName パラメータに示された論理スキーマとコンテキストから名前を作成します。

最初の構文は、現在のコンテキストでの現在の論理スキーマに応じて、ローカル・オブジェクト・マスクを使用してオブジェクト名を作成します（pMode = "L"）。

5 番目の構文は 4 番目と同じですが、pLocation = "D"です。

パラメータ

パラメータ	タイプ	説明	
pMode	文字列	"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。 "R"はリモート・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。 **注意**: リモート・オブジェクト・マスクを使用する場合、getObjectName は常にリモート・サーバーのデフォルト物理スキーマを使用してオブジェクト名前を解決します。	
pObjectName	文字列	有効なリソース名（表またはファイル）を表すすべての文字列。このオブジェクト名には接頭辞を付けておくことができます。この接頭辞は、実行時に、物理スキーマで定義された適切な一時オブジェクト接頭辞に置き換えられます。	
pLogicalSchemaName	文字列	オブジェクトの強制論理スキーマの名前。	
pContextName	文字列	オブジェクトの強制コンテキスト。	
pLocation	文字列	"W"	物理カタログのオブジェクトと、指定されたタプル（コンテキスト、論理スキーマ）に対応する作業物理スキーマの完全名を返します。
		"D"	物理カタログのオブジェクトと、指定されたタプル（コンテキスト、論理スキーマ）に対応するデータ物理スキーマの完全名を返します。

接頭辞

pObjectName パラメータで指定されたリソース名に接頭辞コードを付加することで、odi 一時オブジェクト名（エラー表または統合表、ジャーナル化トリガーなど）を生成することができます。

接頭辞のリストを次に示します。

接頭辞	説明
%INT_PRF	統合表の接頭辞（デフォルト値は「I$_」）。
%COL_PRF	ロード表の接頭辞（デフォルト値は「C$_」）。
%ERR_PRF	エラー表の接頭辞（デフォルト値は「E$_」）。
%JRN_PRF_TAB	ジャーナル化表の接頭辞（デフォルト値は「J$_」）。

%INT_PRF_VIE	ジャーナル化ビューの接頭辞（デフォルト値は「JV$_」）。
%INT_PRF_TRG	ジャーナル化トリガーの接頭辞（デフォルト値は「T$_」）。

一時オブジェクトは通常、作業物理スキーマに作成されることに注意してください。このため、接頭辞を使用して一時オブジェクトを作成するか、一時オブジェクトにアクセスする場合は、pLocation を W に設定します。

例

定義されている物理スキーマ:

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

コール対象	戻り値
`<%=odiRef.getObjectName("L", "EMP", "MSSQL_ODI", "CTX_DEV", "W")%>`	tempdb.temp_owner.EMP
`<%=odiRef.getObjectName("L", "EMP", "MSSQL_ODI", "CTX_DEV", "D")%>`	db_odi.dbo.EMP
`<%=odiRef.getObjectName("R", "%ERR_PRFEMP", "MSSQL_ODI", "CTX_DEV", "W")%>`	MyServer.tempdb.temp_owner.E$_EMP
`<%=odiRef.getObjectName("R", "EMP", "MSSQL_ODI", "CTX_DEV", "D")%>`	MyServer.db_odi.dbo.EMP

getObjectNameDefaultPSchema()メソッド

使用方法

```
public java.lang.String getObjectNameDefaultPSchema(
java.lang.String pMode,
java.lang.String pObjectName,
java.lang.String pLocation)
public java.lang.String getObjectNameDefaultPSchema(
java.lang.String pMode,
java.lang.String pObjectName,
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getObjectNameDefaultPSchema(
java.lang.String pMode,
```

```
java.lang.String pObjectName,
java.lang.String pLogicalSchemaName,
java.lang.String pContextName,
java.lang.String pLocation)
public java.lang.String getObjectNameDefaultPSchema(
java.lang.String pObjectName,
java.lang.String pLocation)
public java.lang.String getObjectNameDefaultPSchema(
java.lang.String pObjectName)
```

説明

このメソッドは getObjectName メソッドに類似しています。ただし、オブジェクト名は、物理スキーマがアタッチされるデータ・サーバーのデフォルト物理スキーマに対して導出されます。getObjectName では、オブジェクト名は物理スキーマ自身に対して導出されます。

- 詳細は、getObjectName を参照してください。

getOption()メソッド
getUserExit()メソッド

使用方法

```
public java.lang.String getOption(java.lang.String pOptionName)
public java.lang.String getUserExit(java.lang.String pOptionName)
```

説明

KM またはプロシージャのオプション（ユーザー・イグジットとも呼ばれます）の値を返します。getUserExit 構文は廃止予定で、互換性上の理由で残されているだけです。

パラメータ

パラメータ	タイプ	説明
pOptionName	文字列	リクエストされたオプションの名前が含まれている文字列。

例

MY_OPTION_1 オプションの値は`<%=odiRef.getOption("MY_OPTION_1")%>`

getPrevStepLog()メソッド

使用方法

```
public java.lang.String getPrevStepLog(java.lang.String pPropertyName)
```

説明

パッケージで最後に実行された手順についての情報を返します。リクエストされた情報は、pPropertyName パラメータを使用して指定されます。前の手順がない場合（たとえば、getPrevStepLog 手順がパッケージ外部から実行される場合）は、「前のステップがありません」という例外が発生します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	前の手順についてリクエストされたプロパティの名前が含まれる文字列。次に示す、有効プロパティのリストを参照してください。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
SESS_NO	セッションの番号。
NNO	パッケージ内部の手順の番号。実行された最初の手順は 0 です。
STEP_NAME	手順の名前。
STEP_TYPE	手順のタイプを示しているコード。次の値が返されます。 • F: インタフェース • VD: 変数宣言 • VS: 変数の設定/増分 • VE: 変数の評価 • V: 変数のリフレッシュ • T: プロシージャ • OE: OS コマンド • SE: odi ツール • RM: モデルのリバース・エンジニア • CM: モデルのチェック • CS: サブモデルのチェック • CD: データストアのチェック • JM: モデルのジャーナル化 • JD: データストアのジャーナル化
CONTEXT_NAME	手順が実行されたコンテキストの名前。
MAX_ERR	最大限許容されるエラーの数または割合

MAX_ERR_PRCT	最大エラー数が割合で表されている場合は 1、それ以外の場合は 0 を返します。
RUN_COUNT	この手順がこれまでに実行された回数。
BEGIN	手順が開始された日付と時刻。
END	手順が終了した日付と時刻。
DURATION	手順を実行するのにかかった時間（単位は秒）。
STATUS	前の手順の終了ステータスを示している 1 文字のコードを返します。状態 R（実行中）が返されることはありません。 • D: 終了（成功） • E: エラー • Q: キュー化 • W: 待機中 • M: 警告
RC	リターン・コード。0 はエラーがないことを示します。
MESSAGE	前の手順により返されたエラー・メッセージがある場合、そのメッセージ。エラーがない場合は空の文字列。
INSERT_COUNT	手順によって挿入された行数。
DELETE_COUNT	手順によって削除された行数。
UPDATE_COUNT	手順によって更新された行数。
ERROR_COUNT	品質管理手順の場合の、手順によって検出された誤っている行の数。

例

前の手順<%=odiRef.getPrevStepLog("STEP_NAME")%>の実行に要した時間は<%=odiRef.getPrevStepLog("DURATION")%>秒です。

getQuotedString()

使用方法

```
public java.lang.String getQuotedString(java.lang.String pString)
```

説明

このメソッドは引用符で囲まれた文字列を返します。引用符や、文字列に表示される可能性がある\n、\t などのエスケープ文字は維持されます。

このメソッドは、Java または Jython コードで、値として渡された文字列を保護するのに有効です。

パラメータ

パラメータ	タイプ	説明
pString	文字列	引用符で保護する文字列。

例

次の Java コードでは、有効な文字列値を生成するために getQuotedString メソッドが使用されています。

```
String condSqlOK = <%=odiRef.getQuotedString(odiRef.getCK("MESS"))%>;
String condSqlKO = <%=odiRef.getCK("MESS")%>;
```

条件のメッセージが"Error:\n Zero is not a valid value"である場合、生成されるコードは次のようになります。getQuotedString を使用しないと、\n が維持されず、改行になってしまうため、コードは正しくありません。

```
String condSqlOK = "Error:\n Zero is not a valid value";
String condSqlKO = "Error:
Zero is not a valid value";
```

getSchemaName()メソッド

使用方法

```
public java.lang.String getSchemaName(
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getSchemaName(
java.lang.String pLogicalSchemaName,
java.lang.String pContextCode,
java.lang.String pLocation)
public java.lang.String getSchemaName(
java.lang.String pLocation)
public java.lang.String getSchemaName()
```

説明

データ・スキーマまたは作業スキーマの物理名を論理スキーマから取得します。

最初の構文が使用される場合、返されるスキーマは現在のコンテキストに対応します。

2 番目の構文が使用される場合、返されるスキーマは pContextCode パラメータで指定されたコンテキストに対応します。

3 番目の構文は、現在の論理スキーマ、現在のコンテキストでのデータ・スキーマ（D）または作業スキーマ（W）の名前を返します。

4 番目の構文は、現在のコンテキスト、現在の論理スキーマでのデータ・スキーマ（D）の名前を返します。

パラメータ

パラメータ	タイプ	説明	
pLogicalSchemaName	文字列	スキーマの論理スキーマの名前。	
pContextCode	文字列	スキーマの強制コンテキスト。	
pLocation	文字列	"W"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマの作業スキーマを返します。
		"D"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマのデータ・スキーマを返します。

例

定義されている物理スキーマ:
Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

コール対象	戻り値
`<%=odiRef.getSchemaName("MSSQL_ODI", "CTX_DEV", "W")%>`	temp_owner
`<%=odiRef.getSchemaName("MSSQL_ODI", "CTX_DEV", "D")%>`	dbo

GetSchemaNameDefaultPSchema()メソッド

使用方法

```
public java.lang.String getSchemaNameDefaultPSchema(
java.lang.String pLogicalSchemaName,
java.lang.String pLocation)
public java.lang.String getSchemaNameDefaultPSchema(
java.lang.String pLogicalSchemaName,
java.lang.String pContextCode,
java.lang.String pLocation)
public java.lang.String getSchemaNameDefaultPSchema(
java.lang.String pLocation)
```

```
public java.lang.String getSchemaNameDefaultPSchema()
```

説明

タプルに対応している物理スキーマ（論理スキーマ、コンテキスト）が関連付けられているデータ・サーバーの**デフォルト**の物理データ・スキーマまたは作業スキーマの名前を取得できます。コンテキストを指定しない場合、現在のコンテキストが使用されます。論理スキーマ名が指定されない場合、現在の論理スキーマが使用されます。pLocation が指定されない場合、データ・スキーマが返されます。

パラメータ

パラメータ	タイプ	説明	
pLogicalSchemaName	文字列	論理スキーマの名前。	
pContextCode	文字列	強制適用されたスキーマのコンテキストのコード。	
pLocation	文字列	"W"	タプル（コンテキスト、論理スキーマ）に対応している物理スキーマもアタッチされているデータ・サーバーに関連付けられている、デフォルトの物理スキーマの作業スキーマを返します。
		"D"	タプル（コンテキスト、論理スキーマ）に対応する物理スキーマのデータ・スキーマを返します。

例

定義されている物理スキーマ:
`Pluton.db_odi.dbo`

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_odi
デフォルト・スキーマか	**はい**

この物理スキーマに関連付けられているもの: コンテキスト CTX_DEV の MSSQL_ODI および `Pluton.db_doc.doc`

データ・カタログ:	db_doc
データ・スキーマ:	doc
作業カタログ:	tempdb

作業スキーマ:	temp_doc
デフォルト・スキーマか	**いいえ**

この物理スキーマに関連付けられているもの: コンテキスト CTX_DEV の MSSQL_DOC

コール対象	戻り値
`<%=odiRef.getSchemaNameDefaultPSchema("MSSQL_DOC", "CTX_DEV", "W")%>`	temp_odi
`<%=odiRef.getSchemaNameDefaultPSchema("MSSQL_DOC", "CTX_DEV", "D")%>`	dbo

getSession()メソッド

使用方法

```
public java.lang.String getSession(java.lang.String pPropertyName)
```

説明

現在のセッションに関する概要を返す汎用メソッド。使用可能なプロパティのリストを**pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
SESS_NO	セッションの内部番号。
SESS_NAME	セッションの名前。
SCEN_VERSION	現在のシナリオのバージョン
CONTEXT_NAME	実行コンテキストの名前
CONTEXT_CODE	実行コンテキストのコード
AGENT_NAME	実行を担当する物理エージェントの名前
SESS_BEG	セッション開始の日付と時刻

USER_NAME	セッションを実行している odi ユーザー

例

現在のセッション: **<%=odiRef.getSession("SESS_NAME")%>**

getSessionVarList()メソッド

使用方法

```
public java.lang.String getSessionVarList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

説明

将来の使用のために予約されています。

パラメータ

将来の使用のために予約されています。

例

将来の使用のために予約されています。

getStep()メソッド

使用方法

```
public java.lang.String getStep(java.lang.String pPropertyName)
```

説明

現在の手順の概要を返す汎用メソッド。使用可能な情報のリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
SESS_NO	手順が所属するセッションの番号
NNO	セッションでの手順の番号
NB_RUN	実行の試行数。
STEP_NAME	手順名
STEP_TYPE	手順のタイプ
CONTEXT_NAME	実行コンテキストの名前
VAR_INCR	手順変数の増分
VAR_OP	変数を比較するために使用される演算子
VAR_VALUE	変数の強制値
OK_EXIT_CODE	成功の場合の終了コード
OK_EXIT	成功の場合にパッケージを終了
OK_NEXT_STEP	成功の場合の次の手順
OK_NEXT_STEP_NAME	成功の場合の次の手順の名前
KO_RETRY	失敗の場合の再試行数
KO_RETRY_INTERV	失敗の場合の試行間隔
KO_EXIT_CODE	失敗の場合の終了コード
KO_EXIT	失敗の場合にパッケージを終了
KO_NEXT_STEP	失敗の場合の次の手順
KO_NEXT_STEP_NAME	失敗の場合の次の手順の名前

例

現在の手順: **<%=odiRef.getStep("STEP_NAME")%>**

getSysDate()メソッド

使用方法

```
public java.lang.String getSysDate()
```

```
public java.lang.String getSysDate(pDateFormat)
```

説明

このメソッドは、セッションを実行しているマシンのシステム日付を返します。

パラメータ

パラメータ	タイプ	説明
pDateFormat	文字列	システム日付を返すために使用された日付書式。日付書式パターンに関する詳細は、「日付書式」を参照してください。

例

現在の年: `:<%=odiRef.getSysDate("y")%>`

getUser()メソッド

使用方法

```
public java.lang.String getUser(java.lang.String pPropertyName)
```

説明

現在のセッションを実行しているユーザーに関する概要を返す汎用メソッド。使用可能なプロパティのリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
I_USER	ユーザー識別子
USER_NAME	ユーザー名
IS_SUPERVISOR	ユーザーが supervisor（1）か、そうでない（0）かを示すブール値のフラグ

例

実行したユーザー: **<%=odiRef.getUser("USER_NAME")%>**

getOption()メソッド
getUserExit()メソッド

使用方法

```
public java.lang.String getOption(java.lang.String pOptionName)
public java.lang.String getUserExit(java.lang.String pOptionName)
```

説明

KM またはプロシージャのオプション（ユーザー・イグジットとも呼ばれます）の値を返します。getUserExit 構文は廃止予定で、互換性上の理由で残されているだけです。

パラメータ

パラメータ	タイプ	説明
pOptionName	文字列	リクエストされたオプションの名前が含まれている文字列。

例

MY_OPTION_1 オプションの値は**<%=odiRef.getOption("MY_OPTION_1")%>**

setNbInsert、setNbUpdate、setNbDelete、setNbErrors および setNbRows メソッド

使用方法

```
public java.lang.Void setNbInsert(public java.lang.Long)
public java.lang.Void setNbUpdate(public java.lang.Long)
public java.lang.Void setNbDelete(public java.lang.Long)
public java.lang.Void setNbErrors(public java.lang.Long)
public java.lang.Void setNbRows(public java.lang.Long)
```

説明

現在のタスクに設定されたこれらのメソッドにより、次の値が報告されます。

- 挿入された行数（**setNbInsert**）
- 更新された行数（**setNbUpdate**）

- 削除された行数（**setNbDelete**）
- エラーになった行数（**setNbErrors**）
- このタスク時に処理された行の総数（**setNbRows**）

これらの数は、処理された行の実際の数とは別に設定できます。

重要: このメソッドはJythonコードなどのスクリプト・エンジン・コマンドでのみ使用可能です。 <% %>タグで囲まないでください。

例

次のJythonの例では、挿入行数を定数値の50に設定し、エラーの行数を#DEMO.NbErrorsというODI変数から導出します。

```
InsertNumber=50
odiRef.setNbInsert(InsertNumber)
ErrorNumber=#DEMO.NbErrors
odiRef.setNbErrors(ErrorNumber)
```

setNbInsert、setNbUpdate、setNbDelete、setNbErrors および setNbRows Methods

使用方法

```
public java.lang.Void setNbInsert(public java.lang.Long)
public java.lang.Void setNbUpdate(public java.lang.Long)
public java.lang.Void setNbDelete(public java.lang.Long)
public java.lang.Void setNbErrors(public java.lang.Long)
public java.lang.Void setNbRows(public java.lang.Long)
```

説明

現在のタスクに設定されたこれらのメソッドにより、次の値が報告されます。

- 挿入された行数（**setNbInsert**）
- 更新された行数（**setNbUpdate**）
- 削除された行数（**setNbDelete**）
- エラーになった行数（**setNbErrors**）
- このタスク時に処理された行の総数（**setNbRows**）
- これらの数は、処理された行の実際の数とは別に設定できます。

重要: このメソッドはJythonコードなどのスクリプト・エンジン・コマンドでのみ使用可能です。 <% %>タグで囲まないでください。

例

次の Jython の例では、挿入行数を定数値の 50 に設定し、エラーの行数を#DEMO.NbErrors という ODI 変数から導出します。

```
InsertNumber=50
odiRef.setNbInsert(InsertNumber)
ErrorNumber=#DEMO.NbErrors
odiRef.setNbErrors(ErrorNumber)
```

setNbInsert、setNbUpdate、setNbDelete、setNbErrors および setNbRows Methods

使用方法

```
public java.lang.Void setNbInsert(public java.lang.Long)
public java.lang.Void setNbUpdate(public java.lang.Long)
public java.lang.Void setNbDelete(public java.lang.Long)
public java.lang.Void setNbErrors(public java.lang.Long)
public java.lang.Void setNbRows(public java.lang.Long)
```

説明

現在のタスクに設定されたこれらのメソッドにより、次の値が報告されます。

- 挿入された行数（**setNbInsert**）
- 更新された行数（**setNbUpdate**）
- 削除された行数（**setNbDelete**）
- エラーになった行数（**setNbErrors**）
- このタスク時に処理された行の総数（**setNbRows**）

これらの数は、処理された行の実際の数とは別に設定できます。

重要: このメソッドは Jython コードなどのスクリプト・エンジン・コマンドでのみ使用可能です。 <% %>タグで囲まないでください。

例

次の Jython の例では、挿入行数を定数値の 50 に設定し、エラーの行数を#DEMO.NbErrors という ODI 変数から導出します。

```
InsertNumber=50
odiRef.setNbInsert(InsertNumber)
ErrorNumber=#DEMO.NbErrors
odiRef.setNbErrors(ErrorNumber)
```

setNbInsert、setNbUpdate、setNbDelete、setNbErrors および setNbRows Methods

使用方法

```
public java.lang.Void setNbInsert(public java.lang.Long)
public java.lang.Void setNbUpdate(public java.lang.Long)
public java.lang.Void setNbDelete(public java.lang.Long)
public java.lang.Void setNbErrors(public java.lang.Long)
public java.lang.Void setNbRows(public java.lang.Long)
```

説明

現在のタスクに設定されたこれらのメソッドにより、次の値が報告されます。

- 挿入された行数（**setNbInsert**）
- 更新された行数（setNbUpdate）
- 削除された行数（setNbDelete）
- エラーになった行数（setNbErrors）
- このタスク時に処理された行の総数（setNbRows）

これらの数は、処理された行の実際の数とは別に設定できます。

重要: このメソッドは Jython コードなどのスクリプト・エンジン・コマンドでのみ使用可能です。 <% %>タグで囲まないでください。

例

次の Jython の例では、挿入行数を定数値の 50 に設定し、エラーの行数を`#DEMO.NbErrors` という ODI 変数から導出します。

```
InsertNumber=50
odiRef.setNbInsert(InsertNumber)
ErrorNumber=#DEMO.NbErrors
odiRef.setNbErrors(ErrorNumber)
```

setNbInsert、setNbUpdate、setNbDelete、setNbErrors および setNbRows Methods

使用方法

```
public java.lang.Void setNbInsert(public java.lang.Long)
public java.lang.Void setNbUpdate(public java.lang.Long)
public java.lang.Void setNbDelete(public java.lang.Long)
public java.lang.Void setNbErrors(public java.lang.Long)
public java.lang.Void setNbRows(public java.lang.Long)
```

説明

現在のタスクに設定されたこれらのメソッドにより、次の値が報告されます。

- 挿入された行数（**setNbInsert**）
- 更新された行数（**setNbUpdate**）
- 削除された行数（**setNbDelete**）
- エラーになった行数（**setNbErrors**）
- このタスク時に処理された行の総数（**setNbRows**）

これらの数は、処理された行の実際の数とは別に設定できます。

重要: このメソッドはJythonコードなどのスクリプト・エンジン・コマンドでのみ使用可能です。 <% %>タグで囲まないでください。

例

次のJythonの例では、挿入行数を定数値の50に設定し、エラーの行数を#DEMO.NbErrorsというODI変数から導出します。

```
InsertNumber=50
odiRef.setNbInsert(InsertNumber)
ErrorNumber=#DEMO.NbErrors
odiRef.setNbErrors(ErrorNumber)
```

ジャーナル化メソッド（JKM）

getJrnFilter()メソッド

使用方法

```
public java.lang.String getJrnFilter()
```

説明

現在のインタフェースのSQLジャーナル化フィルタを返します。ジャーナル化された表がソースにある場合、ロード・フェーズでこのメソッドを使用できます。ジャーナル化された表がステージング領域にある場合、統合時にこのメソッドを使用できます。

パラメータ

なし

例

```
<%=odiRef.getJrnFilter()%>
```

getJrnInfo()メソッド

使用方法

```
public java.lang.String getJrnInfo(java.lang.String pPropertyName)
```

説明

モデルやデータストアのジャーナル化の場合はJKM、インタフェースの場合はLKM/IKMに対するデータストアのジャーナル化に関する概要を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyNameの値

次の表は、pPropertyNameで可能な様々な値のリストです。

パラメータ値	説明
FULL_TABLE_NAME	ジャーナル化されたデータストアのフルネーム。
JRN_FULL_NAME	ジャーナル・データストアのフルネーム。
JRN_FULL_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
JRN_FULL_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
JRN_FULL_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
JRN_FULL_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
JRN_FULL_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
JRN_FULL_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
JRN_SUBSCRIBER	作業スキーマにおけるサブスクライバ表の名前。
JRN_NAME	ジャーナル化されたデータストアの名前。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューの名前。

JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューの名前。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーの名前。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーの名前。
JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーの名前。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーの名前。
JRN_SUBSCRIBER	サブスクライバの名前。
JRN_COD_MODE	ジャーナル化されたデータ・モデルのコード。
JRN_METHOD	ジャーナル化モード（一貫または単純）。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。

例

ジャーナル化される表: **<%=odiRef.getJrnInfo("FULL_TABLE_NAME")%>**

getSubscriberList()メソッド

使用方法

```
public java.lang.String getSubscriberList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getSubscriberList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

ジャーナル化された表のサブスクライバのリストを提供します。pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例 « 私の名前は[SUBSCRIBER]です。»
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
SUBSCRIBER	サブスクライバの名前

例

サブスクライバの表: **<%=odiRef.getSubscriberList("\nBegin List\n", "-[SUBSCRIBER]", "\n", "\nEnd of List\n")%>**

getTable()メソッド

使用方法

```
public java.lang.String getTable(
java.lang.String pMode,
java.lang.String pProperty,
java.lang.String pLocation)
```

```
public java.lang.String getTable(
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty)
```

説明

odi によって処理された一時的表および永続的表のフルネームを取得します。

パラメータ

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>pMode</td><td>文字列</td><td>"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。この値は、pMode が指定されていない場合に使用されます。
"R"はオブジェクト・マスクを使用してオブジェクトの完全パスを作成します。
"A"は自動。使用する適切なマスクを自動的に定義します。</td></tr>
<tr><td>pProperty</td><td>文字列</td><td>構築される表の名前を示すパラメータ。可能な値のリストを次に示します。
<table>
<tr><th>パラメータ値</th><th>説明</th></tr>
<tr><td>ID</td><td>データストア識別子。</td></tr>
<tr><td>TARG_NAME</td><td>ターゲット・データストアのフルネーム。アクションでは、このパラメータは、DDL コマンドによって処理された現在の表の名前を返します。</td></tr>
<tr><td>COLL_NAME</td><td>ロード・データストアのフルネーム。</td></tr>
<tr><td>INT_NAME</td><td>統合データストアのフルネーム。</td></tr>
<tr><td>ERR_NAME</td><td>エラー・データストアのフルネーム。</td></tr>
<tr><td>CHECK_NAME</td><td>エラー・サマリー・データストアの名前。</td></tr>
<tr><td>CT_NAME</td><td>チェック・データストアのフルネーム。</td></tr>
<tr><td>FK_PK_TABLE_NAME</td><td>外部キーによって参照されたデータストアのフルネーム。</td></tr>
</table>
</td></tr>
</table>

		JRN_NAME	ジャーナル化されたデータストアのフルネーム。
		JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
		JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
		JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
		JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
		JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
		JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
		SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアのフルネーム。
		CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
		CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
		CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
		CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
		<flexfield code>	現在のターゲット表のフレックスフィールド値。
pLocation	文字列	"W"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理作業スキーマのフルネームを返します。
		"D"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理データ・スキーマのフルネームを返します。

	"A"	odi がオブジェクトのデフォルトの場所を決定します。この値は、pLocation が指定されていない場合に使用されます。

例

定義されている要素:

物理スキーマ: Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner
ローカル・マスク:	%CATALOG.%SCHEMA.%OBJECT
リモート・マスク:	%DSERVER:%CATALOG.%SCHEMA.%OBJECT
ロード接頭辞:	CZ_
エラー接頭辞:	ERR_
統合接頭辞:	I$_

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

表の名前: CUSTOMER

コール対象	戻り値
`<%=odiRef.getTable("L", "COLL_NAME", "W")%>`	tempdb.temp_owner.CZ_0CUSTOMER
`<%=odiRef.getTable("R", "COLL_NAME", "D")%>`	MyServer:db_odi.dbo.CZ_0CUSTOMER
`<%=odiRef.getTable("L", "INT_NAME", "W")%>`	tempdb.temp_owner.I$_CUSTOMER
`<%=odiRef.getTable("R", "ERR_NAME", "D")%>`	MyServer:db_odi.dbo.ERR_CUSTOMER

getColList()メソッド

使用方法

```
public java.lang.String getColList(
java.lang.String pStart,
```

```
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pSelector)
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

列と式のリストを提供します。列リストはこのメソッドがコールされたフェーズにより異なります。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

ロード（LKM）

現在のソース環境で実行されたすべてのマッピング式と、ステージング領域で実行されたマッピング、フィルタ式、結合で使用された列。

インタフェースで execute というタグを付けられたマッピングのみが対象です。

- リストは POS、FILE_POS でソートされます。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

統合（IKM）

現在のインタフェースで execute というタグを付けられたすべての現在のマッピング式。

リストには、現在のインタフェースのターゲット表に（execute というタグを付けて）ロードされる各列につき、1 つの要素が含まれます。

- ロードされた表が一時表でない場合、リストは POS、FILE_POS でソートされます。
- ロードされた表が一時表である（参照にない）場合、リストはソートされません。

インタフェースのソースにジャーナル化されたデータストアがあり、それがステージング領域にある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

チェック（CKM）

ターゲット表のすべての列（静的管理またはフロー制御）

ターゲット表の列を、現在のインタフェースで記入された列から区別するには、MAP セレクタを使用する必要があります。

- リストはターゲット表の POS、FILE_POS でソートされます。

アクション

DDL コマンドが処理した表のすべての列。

変更されたか、追加したか、削除された列の場合、NEW および OLD セレクタを使用して、DDL コマンドによって処理される、変更された列の新バージョンまたは旧バージョンを取得することができます。

- リストは表の POS、FILE_POS でソートされます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。
CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。

SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外（*）。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外（*）。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外（*）。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外（*）。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。

PK	• LKM: 適用対象外（*）。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外（*）。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外（*）。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
UD1	• LKM: 適用対象外（*）。 • IKM: UD1 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD2	• LKM: 適用対象外（*）。 • IKM: UD2 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD3	• LKM: 適用対象外（*）。 • IKM: UD3 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD4	• LKM: 適用対象外（*）。 • IKM: UD4 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD5	• LKM: 適用対象外（*）。 • IKM: UD5 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。

	静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
NEW	• アクション: 表に追加された列。表の変更された列の新しいバージョン。
OLD	• アクション: 表から削除された列。表の変更された列の旧バージョン。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

（*）**重要な注意**: LKM で、*で示した一部のセレクタを使用することは、可能ですが非推奨です。インタフェースでソース上にマップされた列のみが返されます。**結果として、インタフェースにより、結果が正しくないことがあります。**
たとえば、UK セレクタでは、マップされないか、ソース上で実行されないキーの列は、このセレクタで返されません。

例

CUSTOMER 表に列 (CUST_ID, CUST_NAME, AGE)が含まれ、作成するコードが次のようである場合:

```
create table CUSTOMER (CUST_ID Numeric(10) null, CUST_NAME VARCHAR(50)
null, AGE Numeric(3) null)
```

次のように記述します。

```
create table CUSTOMER <%=odiRef.getColList("(", "[COL_NAME]
[SOURCE_CRE_DT] null", ", ", ")", "")%>
```

説明: getColList 関数が使用され、(CUST_ID numeric(10) null, CUST_NAME varchar(50) null, AGE numeric(3) null)が生成されます。先頭と末尾はカッコで、(column, data type, null)というパターンが、各列につきカンマで区切られて繰り返されます。このため、

- 関数の最初の文字**"("**は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ**"[COL_NAME] [SOURCE_CRE_DT] null"**は、このパターンを各列に対して繰り返すことを示します。キーワード[COL_NAME]および[SOURCE_CRE_DT]は、表「パターン属性リスト」の有効なキーワードに対する参照です。
- 3 番目のパラメータ**", "**は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字**")"**は、文字列が、文字列「)」で終わることを示します。
- 最後のパラメータ**""**は、パターンを各列に対して（選択内容なしで）繰り返すことを示します。

ロード・メソッド（LKM）

getColList()メソッド

使用方法

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pSelector)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

列と式のリストを提供します。列リストはこのメソッドがコールされたフェーズにより異なります。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

ロード（LKM）

現在のソース環境で実行されたすべてのマッピング式と、ステージング領域で実行されたマッピング、フィルタ式、結合で使用された列。

インタフェースで execute というタグを付けられたマッピングのみが対象です。

- リストは POS、FILE_POS でソートされます。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

統合（IKM）

現在のインタフェースで execute というタグを付けられたすべての現在のマッピング式。

リストには、現在のインタフェースのターゲット表に（execute というタグを付けて）ロードされる各列につき、1 つの要素が含まれます。

- ロードされた表が一時表でない場合、リストは POS、FILE_POS でソートされます。
- ロードされた表が一時表である（参照にない）場合、リストはソートされません。

インタフェースのソースにジャーナル化されたデータストアがあり、それがステージング領域にある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

チェック（CKM）

ターゲット表のすべての列（静的管理またはフロー制御）

ターゲット表の列を、現在のインタフェースで記入された列から区別するには、MAP セレクタを使用する必要があります。

- リストはターゲット表の POS、FILE_POS でソートされます。

アクション

DDL コマンドが処理した表のすべての列。

変更されたか、追加したか、削除された列の場合、NEW および OLD セレクタを使用して、DDL コマンドによって処理される、変更された列の新バージョンまたは旧バージョンを取得することができます。

- リストは表の POS、FILE_POS でソートされます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。

SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。
CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。

MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外（*）。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外（*）。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外（*）。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外（*）。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外（*）。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外（*）。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外（*）。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。

	• CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
UD1	• LKM: 適用対象外（*）。 • IKM: UD1 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD2	• LKM: 適用対象外（*）。 • IKM: UD2 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD3	• LKM: 適用対象外（*）。 • IKM: UD3 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD4	• LKM: 適用対象外（*）。 • IKM: UD4 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD5	• LKM: 適用対象外（*）。 • IKM: UD5 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。

SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
NEW	• アクション: 表に追加された列。表の変更された列の新しいバージョン。
OLD	• アクション: 表から削除された列。表の変更された列の旧バージョン。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

（*）重要な注意 : LKM で、*で示した一部のセレクタを使用することは、可能ですが非推奨です。インタフェースでソース上にマップされた列のみが返されます。**結果として、インタフェースにより、結果が正しくないことがあります。**
たとえば、UK セレクタでは、マップされないか、ソース上で実行されないキーの列は、このセレクタで返されません。

例

CUSTOMER 表に列 (CUST_ID, CUST_NAME, AGE)が含まれ、作成するコードが次のようである場合:

```
create table CUSTOMER (CUST_ID Numeric(10) null, CUST_NAME VARCHAR(50)
null, AGE Numeric(3) null)
```

次のように記述します。

```
create table CUSTOMER <%=odiRef.getColList("(", "[COL_NAME]
[SOURCE_CRE_DT] null", ", ", ")", "")%>
```

説明: getColList 関数が使用され、(CUST_ID numeric(10) null, CUST_NAME varchar(50) null, AGE numeric(3) null)が生成されます。先頭と末尾はカッコで、(column, data type, null)というパターンが、各列につきカンマで区切られて繰り返されます。このため、

- 関数の最初の文字**"("**は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ**"[COL_NAME] [SOURCE_CRE_DT] null"**は、このパターンを各列に対して繰り返すことを示します。キーワード[COL_NAME]および[SOURCE_CRE_DT]は、表「パターン属性リスト」の有効なキーワードに対する参照です。

- 3 番目のパラメータ**","**は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字**")"**は、文字列が、文字列「)」で終わることを示します。
- 最後のパラメータ**""**は、パターンを各列に対して（選択内容なしで）繰り返すことを示します。

getFilter()メソッド

使用方法

```
public java.lang.String getFilter()
```

説明

SQL フィルタ列を返します（ロード時はソースで、統合時はステージング領域で）。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getFilterList()メソッド

使用方法

```
public java.lang.String getFilterList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getFilterList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL フィルタの発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用中のナレッジ・モジュールにより異なります）で実行される各フィルタ式の要素が含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 リスト内で使用できるもののリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切るために使用されます。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	フィルタ内部識別子
EXPRESSION	フィルタ式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getFilterList 関数は、SELECT 句のフィルタを生成するために使用されます。フィルタは and で始まり、各フィルタごとに and で区切ってパターン（各フィルタの式）を繰り返します。このため、

- 関数の最初のパラメータ**"and"**は、文字列を、文字列 and で始めることを示します。

- 2 番目のパラメータ**"([EXPRESSION])"** は、このパターンを各フィルタに対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ**" and "**は、パターンの解釈された発生を、文字列 and で区切ることを示します。
- 関数の 4 番目のパラメータ**""**は、文字列が、指定された文字を伴わずに終わることを示します。

getFrom()メソッド

使用方法

```
public java.lang.String getFrom()
```

説明

FROM の SQL 文字列をソースの **SELECT** 句から取得します。**FROM** 文は、インタフェースで使用される表および結合（テクノロジの SQL 機能に応じて）から構築されます。このため、ISO の外部結合とカッコをサポートするテクノロジでは、getFrom()が次のような文字列を返すことがあります。

```
((CUSTOMER as CUS inner join CITY as CIT on (CUS.CITY_ID = CIT.CITY_ID))
left outer join SALES_PERSON as SP on (CUS.SALES_ID = SP.SALE_ID))
```

インタフェースのソースにジャーナル化されたデータストアがある場合、句内のソース表は、ジャーナル化されたソース・データストアにリンクされたデータ・ビューによって置き換えられます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getGrpBy()メソッド

使用方法

```
public java.lang.String getGrpBy()
```

説明

SQL GROUP BY 文字列を取得します（収集フェーズではソース上、統合フェーズではステージング領域上）。この文は、マッピング式に検出された集計変換から自動的に計算されます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getGrpByList()メソッド

使用方法

```
public java.lang.String getGrpByList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getGrpByList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL GROUP BY の発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用するナレッジ・モジュールにより異なります）の GROUP BY 文の要素が含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。

pMotif	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切るために使用されます。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	句の内部識別子。
EXPRESSION	グループ化文のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpByList("group by ", "[EXPRESSION]", " , ", "")%>
<%=odiRef.getHaving()%>
```

説明: getGrpByList 関数は、group by で始まり、各式ごとにコンマによって区切られたパターン（各グループ化式）が繰り返される **select** 命令の **group by** 句を生成するために使用されます。

- 関数の最初のパラメータ**"group by"**は、文字列の先頭に「group by」を置くことを示します。
- 2 番目のパラメータ"[EXPRESSION]"は、式によって各グループについてこのパターンを繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ","は、パターンの解釈された発生を、カンマで区切ることを示します。
- 関数の 4 番目のパラメータ""は、文字列が、指定された文字を伴わずに終わることを示します。

getHaving()メソッド

使用方法

```
public java.lang.String getHaving()
```

説明

SQL 文 HAVING を取得します（ロード時はソースで、統合時はステージング領域で）。この文は、検出された集計関数が含まれているフィルタ式から自動的に計算されます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getHavingList()メソッド

使用方法

```
public java.lang.String getHavingList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getHavingList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL HAVING の発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用中のナレッジ・モジュールにより異なります）で実行される各 HAVING 式の要素が 1 つ含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。

pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	句の内部識別子。
EXPRESSION	having 式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpByList("group by ", "[EXPRESSION]", " , ", "")%>
<%=odiRef.getHavingList("having ", "([EXPRESSION])", " and ", "")%>
```

説明: getHavingList 関数は、select 命令の having 句を生成するために使用されます。この句は having で始まり、各式ごとに and で区切ってパターン（各集計フィルタ式）を繰り返します。

- 関数の最初のパラメータ**"having "**は、文字列の先頭に `having` を置くことを示します。
- 2 番目のパラメータ**"([EXPRESSION])"**は、このパターンを各フィルタに対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ**" and "**は、パターンの解釈された発生を、文字列 and で区切ることを示します。
- 関数の 4 番目のパラメータ**""**は、文字列が、指定された文字を伴わずに終わることを示します。

getJoin()メソッド

使用方法

```
public java.lang.String getJoin()
```

説明

SQL 結合文字列を取得します（ロード時はソースで、統合時はステージング領域で）。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getJoinList()メソッド

使用方法

```
public java.lang.String getJoinList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getJoinList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェース内の SQL 結合の発生のリストを提供して、WHERE 句内に位置付けます。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」

		を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	結合の内部識別子。
EXPRESSION	結合式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoinList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getJoinList 関数は、SELECT 文の WHERE 部分に置く結合式を生成するために使用されます。式は and で始まり、各結合ごとに and で区切ってパターン（各結合の式）を繰り返します。このため、

- 関数の最初のパラメータ**"and"**は、文字列を and で始めることを示します。
- 2 番目のパラメータ"([EXPRESSION])"は、このパターンを各結合に対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ" and "は、パターンの解釈された発生を、and で区切ることを示します（and の前後のスペースに注意してください）。
- 関数の 4 番目のパラメータ**""**は、文字列が、指定された文字を伴わずに終わることを示します。

getJrnInfo()メソッド

使用方法

```
public java.lang.String getJrnInfo(java.lang.String pPropertyName)
```

説明

モデルやデータストアのジャーナル化の場合は JKM、インタフェースの場合は LKM/IKM に対するデータストアのジャーナル化に関する概要を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
FULL_TABLE_NAME	ジャーナル化されたデータストアのフルネーム。
JRN_FULL_NAME	ジャーナル・データストアのフルネーム。
JRN_FULL_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
JRN_FULL_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
JRN_FULL_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
JRN_FULL_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
JRN_FULL_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
JRN_FULL_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
JRN_SUBSCRIBER	作業スキーマにおけるサブスクライバ表の名前。
JRN_NAME	ジャーナル化されたデータストアの名前。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューの名前。
JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューの名前。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーの名前。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーの名前。

JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーの名前。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーの名前。
JRN_SUBSCRIBER	サブスクライバの名前。
JRN_COD_MODE	ジャーナル化されたデータ・モデルのコード。
JRN_METHOD	ジャーナル化モード（一貫または単純）。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。

例

ジャーナル化される表: **<%=odiRef.getJrnInfo("FULL_TABLE_NAME")%>**

getJrnFilter()メソッド

使用方法

```
public java.lang.String getJrnFilter()
```

説明

現在のインタフェースの SQL ジャーナル化フィルタを返します。ジャーナル化された表がソースにある場合、ロード・フェーズでこのメソッドを使用できます。ジャーナル化された表がステージング領域にある場合、統合時にこのメソッドを使用できます。

パラメータ

なし

例

<%=odiRef.getJrnFilter()%>

getPop()メソッド

使用方法

```
public java.lang.String getPop(java.lang.String pPropertyName)
```

説明

現在のインタフェースの概要を返す汎用メソッド。使用可能な情報のリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
I_POP	インタフェースの内部番号。
FOLDER	インタフェースのフォルダの名前。
POP_NAME	インタフェースの名前。
IND_WORK_TARG	ステージング領域の位置フラグ。
LSCHEMA_NAME	インタフェースのステージング領域である論理スキーマの名前。
DESCRIPTION	インタフェースの説明。
WSTAGE	ターゲット・データストアの性質を示しているフラグ: • E: ターゲット・データストアは既存の表です（一時表ではありません）。 • N: ターゲット・データストアはデータ・スキーマ内の一時表です。 • W: ターゲット・データストアは作業スキーマ内の一時表です。
TABLE_NAME	ターゲット表の名前。
KEY_NAME	更新キーの名前。
DISTINCT_ROWS	重複抑制フラグ。
OPT_CTX	インタフェースの最適化コンテキストの名前。
TARG_CTX	インタフェースの実行コンテキストの名前。

MAX_ERR	許容される最大エラー数。
MAX_ERR_PRCT	割合によるエラー・インジケータ。
IKM	統合ナレッジ・モジュールの名前。
LKM	ロード・ナレッジ・モジュールの名前。
CKM	チェック・ナレッジ・モジュールの名前。
HAS_JRN	インタフェースのソースにジャーナル化された表がある場合、1 を返します。それ以外の場合は 0 を返します。
<flexfield code>	インタフェースのフレックスフィールド値。

例

現在のインタフェースは**<%=odiRef.getPop("POP_NAME")%>**で、論理スキーマ**<%=odiRef.getInfo("L_SCHEMA_NAME")%>**上で実行されています。

getSrcColList()メソッド

使用方法

```
public java.lang.String getSrcColList( java.lang.String pStart,
java.lang.String pUnMappedPattern,
java.lang.String pMappedPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

説明

このメソッドは LKM と IKM に用意されていて、列のリストのプロパティを返します。このリストは LKM（ソース）または IKM（ステージング領域）によって処理されたソースのすべての列を含みます。リストは、ソース表内の列位置を基準にソートされます。

表示されるプロパティは、列がマップされているかどうかにより異なります。列がマップされている場合、返されるプロパティは pMappedPattern パターンで定義されます。列がマップされていない場合、返されるプロパティは pUnMappedPattern パターンで定義されます。

パターンで使用できる属性の詳細は、「パターン属性リスト」に詳述されます。属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。たとえば、「My string [COL_NAME] is a column」です。

pMappedPattern または pUnMappedPattern パラメータがリストの各要素について解釈され、繰り返されます。パターンは pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pUnMappedPattern	文字列	列がマップされていない場合、パターンはリスト内に現れるたびに繰り返されます。
pMappedPattern	文字列	列がマップされている場合、パターンはリスト内に現れるたびに繰り返されます。
pSeparator	文字列	このパラメータはパターンを区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
ALIAS_NAME	列の名前。COL_NAME と異なり、この属性はオプションのテクノロジ・デリミタを付けずに列名を返します。列名に、たとえばスペースが含まれる場合、これらのデリミタが表示されます。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。

CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース（LKM）またはステージング領域（IKM）上で実行された、式のテキスト（マッピング・フィールドに入力したもの）。列がマップされていない場合、このパラメータは空の文字列を返します。
CX_COL_NAME	サポートされていません。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット（IKM）またはステージング領域（LKM）のテクノロジのデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
<flexfield code>	現在の列のフレックスフィールド値。

例

ソース・ファイルに類似した表を作成する場合:

```
create table <%=odiRef.getTable("L","COLL_NAME", "D")%>_F
(
<%=odiRef.getSrcColList("","[COL_NAME] [DEST_CRE_DT]","[COL_NAME]
[DEST_CRE_DT]",",\n","")%>
)
```

getSrcTablesList()メソッド

使用方法

```
public java.lang.String getSrcTablesList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getSrcTablesList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースのソース表のリストを提供します。このメソッドはSELECT 命令に FROM 句を作成するために使用できます。ただし、これでなく getFrom()メソッドを使用することをお薦めします。

pPattern パターンはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

属性	説明
I_TABLE	現在のソース表の odi 内部番号（番号がある場合）。
MODEL_NAME	現在のソース表のモデルの名前（名前がある場合）。
SUB_MODEL_NAME	現在のソース表のサブモデルの名前（名前がある場合）。
TECHNO_NAME	ソース・データストアのテクノロジの名前。
LSCHEMA_NAME	ソース表の論理スキーマ。
TABLE_NAME	ソース・データストアの論理名。
RES_NAME	リソースの物理アクセス名（ファイル名または JMS キュー、表の物理名など）。 インタフェースのソースにジャーナル化されたデータストアがある場合、句内のソース表は、ジャーナル化されたソース・データストアにリンクされたデータ・ビューによって置き換えられます。
CATALOG	ソース・データストアのカタログ（実行時に解決）。
WORK_CATALOG	ソース・データストアの作業カタログ。
SCHEMA	ソース・データストアのスキーマ（実行時に解決）。
WORK_SCHEMA	ソース・データストアの作業スキーマ。
TABLE_ALIAS	表リストに表示されるデータストアの別名（別名がある場合）。
POP_TAB_ALIAS	現在のインタフェースに表示されるデータストアの別名（別名がある場合）。
TABLE_TYPE	データストア・ソースのタイプ（データストア・ソースがある場合）。
DESCRIPTION	ソース・データストアの説明（説明がある場合）。
R_COUNT	ソース・データストアのレコードの数（判明している場合）。
FILE_FORMAT	ファイル形式（判明している場合）。
FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。

XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。
FILE_FIRST_ROW	無視するヘッダーの行数（ヘッダーがある場合）。
FILE_DEC_SEP	データストアのデフォルトの小数点記号（デフォルトがある場合）。
METADATA	現在のリソースのメタデータの odi 形式の説明（説明がある場合）。
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
<flexfield code>	現在の表のフレックスフィールド値。

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getSrcTablesList("", "[CATALOG].[SCHEMA].[TABLE_NAME] as
[POP_TAB_ALIAS]", ", ", "")%>
where (1=1)
<%=odiRef.getJoinList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getSrcTablesList 関数が、ソース内の各表についてカンマで区切られたパターンを繰り返す SELECT 文の FROM 句を生成するパターン（CATALOG.SCHEMA.TABLE_NAME as POP_TAB_ALIAS）を繰り返すために使用されます。

- 関数の最初のパラメータ""は、文字列を、特定の文字で開始しないことを示します。
- 関数の 2 番目のパラメータ"[CATALOG].[SCHEMA].[TABLE_NAME] as [POP_TAB_ALIAS]"は、このパターンを各ソース表について繰り返すことを示します。キーワード[CATALOG]、[SCHEMA]、[TABLE_NAME]および[POP_TAB_ALIAS]は、表「パターン属性リスト」の有効なキーワードを参照します。
- 3 番目のパラメータ","は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
    - 関数の 4 番目のパラメータ""は、文字列が、指定された文字を伴わずに終わることを示します。

getTable()メソッド

使用方法

```
public java.lang.String getTable(
java.lang.String pMode,
```

```
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty)
```

説明

odi によって処理された一時的表および永続的表のフルネームを取得します。

パラメータ

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>pMode</td><td>文字列</td><td>"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。この値は、pMode が指定されていない場合に使用されます。
"R"はオブジェクト・マスクを使用してオブジェクトの完全パスを作成します。
"A"は自動。使用する適切なマスクを自動的に定義します。</td></tr>
<tr><td>pProperty</td><td>文字列</td><td>構築される表の名前を示すパラメータ。可能な値のリストを次に示します。
<table>
<tr><th>パラメータ値</th><th>説明</th></tr>
<tr><td>ID</td><td>データストア識別子。</td></tr>
<tr><td>TARG_NAME</td><td>ターゲット・データストアのフルネーム。アクションでは、このパラメータは、DDL コマンドによって処理された現在の表の名前を返します。</td></tr>
<tr><td>COLL_NAME</td><td>ロード・データストアのフルネーム。</td></tr>
<tr><td>INT_NAME</td><td>統合データストアのフルネーム。</td></tr>
<tr><td>ERR_NAME</td><td>エラー・データストアのフルネーム。</td></tr>
<tr><td>CHECK_NAME</td><td>エラー・サマリー・データストアの名前。</td></tr>
<tr><td>CT_NAME</td><td>チェック・データストアのフルネーム。</td></tr>
<tr><td>FK_PK_TABLE_NAME</td><td>外部キーによって参照されたデータストアのフルネーム。</td></tr>
</table>
</td></tr>
</table>

		JRN_NAME	ジャーナル化されたデータストアのフルネーム。
		JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
		JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
		JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
		JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
		JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
		JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
		SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアのフルネーム。
		CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
		CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
		CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
		CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
		<flexfield code>	現在のターゲット表のフレックスフィールド値。
pLocation	文字列	"W"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理作業スキーマのフルネームを返します。
		"D"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理データ・スキーマのフルネームを返します。
		"A"	odi がオブジェクトのデフォルトの場所を決定します。この値は、pLocation が指定されていない場合に使用されます。

	は、pLocation が指定されていない場合に使用されます。

例

定義されている要素:

物理スキーマ: Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner
ローカル・マスク:	%CATALOG.%SCHEMA.%OBJECT
リモート・マスク:	%DSERVER:%CATALOG.%SCHEMA.%OBJECT
ロード接頭辞:	CZ_
エラー接頭辞:	ERR_
統合接頭辞:	I$_

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

表の名前: CUSTOMER

コール対象	戻り値
`<%=odiRef.getTable("L", "COLL_NAME", "W")%>`	tempdb.temp_owner.CZ_0CUSTOMER
`<%=odiRef.getTable("R", "COLL_NAME", "D")%>`	MyServer:db_odi.dbo.CZ_0CUSTOMER
`<%=odiRef.getTable("L", "INT_NAME", "W")%>`	tempdb.temp_owner.I$_CUSTOMER
`<%=odiRef.getTable("R", "ERR_NAME", "D")%>`	MyServer:db_odi.dbo.ERR_CUSTOMER

getTargetTable()メソッド

使用方法

```
public java.lang.String getTargetTable(java.lang.String pPropertyName)
```

説明

現在のターゲット表の概要を返す汎用メソッド。使用可能なデータのリストを **pPropertyName の値**の表に示します。

アクションでは、このメソッドは DDL コマンドによって処理されている表に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
I_TABLE	データストアの内部識別子。
MODEL_NAME	現在のデータストアのモデルの名前。
SUB_MODEL_NAME	現在のデータストアのサブモデルの名前。
TECHNO_NAME	ターゲット・テクノロジの名前。
LSCHEMA_NAME	ターゲット論理スキーマの名前。
TABLE_NAME	ターゲット・データストアの名前。
RES_NAME	ターゲット・リソースの物理名。
CATALOG	カタログ名。
WORK_CATALOG	作業カタログの名前。
SCHEMA	スキーマ名。
WORK_SCHEMA	作業スキーマの名前。
TABLE_ALIAS	現在のデータストアの別名。
TABLE_TYPE	データストアのタイプ。
DESCRIPTION	現在のインタフェースの説明。
TABLE_DESC	現在のインタフェースのターゲット・データストアの説明。DDL コマンドの場合、現在の表の説明。
R_COUNT	現在のデータストアの行数。
FILE_FORMAT	現在のデータストア（ファイル）の形式。

FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。
XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。
FILE_FIRST_ROW	ファイル（ファイル）の先頭の、無視する行数。
FILE_DEC_SEP	小数点記号（ファイル）。
METADATA_DESC	データストアのメタデータの説明（ファイル）
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
TABLE_DESC	表の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
WS_NAME	データ・サービス: このデータストアのモデル用に生成された Web サービスの名前。
WS_NAMESPACE	データ・サービス: Web サービスの XML ネームスペース。
WS_JAVA_PACKAGE	データ・サービス: Web サービス用に生成された Java パッケージ。
WS_ENTITY_NAME	データ・サービス: Web サービスでこのデータストアに対して使用されるエンティティ名。
WS_DATA_SOURCE	データ・サービス: このデータストアの Web サービスに対して指定されたデータソース。
<flexfield code>	現在の表のフレックスフィールド値。

例

現在の表: **<%=odiRef.getTargetTable("RES_NAME")%>**

getTargetColList()メソッド

使用方法

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getTargetColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースのターゲット表の列のリストを提供します。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column of the target
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子:

	1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。

ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、`NOT NULL` を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、`DEFAULT <デフォルト値>`を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。

NULL	• LKM: 適用対象外。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを

	設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

例

```
create table TARGET_COPY <%=odiRef.getTargetColList("(", "[COL_NAME]
[DEST_DT] null", ", ", ")", "")%>
```

コントロール・メソッド（CKM）

getAK()メソッド

使用方法

```
public java.lang.String getAK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの代替キーに関する情報を返します。現在のタスクに alternate key というタグが付けられている場合、チェック・ナレッジ・モジュールからのみアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている代替キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	AK 制約の内部番号。
KEY_NAME	代替キーの名前。
MESS	代替キーの制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成された AK のフルネーム。
<flexfield code>	この AK のフレックスフィールドの値。

例

表の代替キーの名前: **<%=odiRef.getAK("KEY_NAME")%>**

getAKColList()メソッド

使用方法

```
public java.lang.String getAKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getAKColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

現在チェックされている代替キーの列と式のリストを提供します。

リストの各要素について、pPattern パラメータが解釈され、繰り返されます。前の要素からは pSeparator パラメータによって区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の代替キーの各列の要素が含まれます。現在のタスクに alternate key というタグが付けられている場合、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された代替キーの列のリストを、キー内の位置の順に並べて返します。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン・シーケンス内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キー列の名前。
COL_HEADING	キー列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始位置（固定ファイル）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるグループ化記号。
SOURCE_DT	列のデータ型のコード。

SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	現在の列のフレックスフィールド値。

例

CUSTOMER 表に代替キーAK_CUSTOMER (CUST_ID, CUST_NAME)があり、作成するコードが次のようである場合:

```
create table T_AK_CUSTOMER (CUST_ID numeric(10) not null, CUST_NAME
varchar(50) not null)
```

次のように記述します。

```
create table T_<%=odiRef.getAK("KEY_NAME")%> <%=odiRef.getAKColList("(",
"[COL_NAME] [DEST_CRE_DT] not null", ", ", ")")%>
```

説明: getAKColList 関数は、(CUST_ID numeric(10) not null, CUST_NAME varchar(50) not null)の部分を生成するために使用されます。これはカッコで開始および停止し、代替キーの各列について、カンマで区切られたパターン（列、データ型、NOT NULL）を繰り返します。このため、

- 関数の最初のパラメータ"("は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ"[COL_NAME] [DEST_CRE_DT] not null"は、このパターンを代替キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[DEST_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ", "は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字")"は、文字列が、文字列「)」で終わることを示します。

getCK()メソッド

使用方法

```
public java.lang.String getCK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの条件に関する情報を返します。現在のタスクに condition というタグが付けられている場合のみ、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されているチェック制約に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている現在の文字列。

次の表は、pPropertyName で許容される様々な値のリストです。

パラメータの値。	説明
ID	チェック制約の内部番号。
COND_ALIAS	SQL 文に使用する表の別名。
COND_NAME	条件の名前。
COND_TYPE	条件のタイプ。
COND_SQL	条件の SQL 文。
MESS	チェック制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成されたチェック制約のフルネーム。
COND_SQL_DDL	表の別名がない条件の SQL 文。
<flexfield code>	このチェック制約のフレックスフィールド値。

例

現在の条件のコール: **<%=snpRep.getCK("COND_NAME")%>**

```
insert into MY_ERROR_TABLE
select *
from MY_CHECKED_TABLE
where (not (<%=odiRef.getCK("COND_SQL")%>))
```

getColList()メソッド

使用方法

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pSelector)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

列と式のリストを提供します。列リストはこのメソッドがコールされたフェーズにより異なります。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

ロード（LKM）

現在のソース環境で実行されたすべてのマッピング式と、ステージング領域で実行されたマッピング、フィルタ式、結合で使用された列。

インタフェースで execute というタグを付けられたマッピングのみが対象です。

- リストは POS、FILE_POS でソートされます。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

統合（IKM）

現在のインタフェースで execute というタグを付けられたすべての現在のマッピング式。

リストには、現在のインタフェースのターゲット表に（execute というタグを付けて）ロードされる各列につき、1 つの要素が含まれます。

- ロードされた表が一時表でない場合、リストは POS、FILE_POS でソートされます。
- ロードされた表が一時表である（参照にない）場合、リストはソートされません。

インタフェースのソースにジャーナル化されたデータストアがあり、それがステージング領域にある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

チェック（CKM）

ターゲット表のすべての列（静的管理またはフロー制御）

ターゲット表の列を、現在のインタフェースで記入された列から区別するには、MAP セレクタを使用する必要があります。

- リストはターゲット表の POS、FILE_POS でソートされます。

アクション

DDL コマンドが処理した表のすべての列。

変更されたか、追加したか、削除された列の場合、NEW および OLD セレクタを使用して、DDL コマンドによって処理される、変更された列の新バージョンまたは旧バージョンを取得することができます。

- リストは表の POS、FILE_POS でソートされます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または!

	2. 論理和: OR または| | 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。

CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、`NOT NULL` を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、`DEFAULT <デフォルト値>`を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外（*）。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外（*）。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外（*）。

	• IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外（*）。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外（*）。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外（*）。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外（*）。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
UD1	• LKM: 適用対象外（*）。 • IKM: UD1 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD2	• LKM: 適用対象外（*）。 • IKM: UD2 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD3	• LKM: 適用対象外（*）。 • IKM: UD3 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD4	• LKM: 適用対象外（*）。 • IKM: UD4 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD5	• LKM: 適用対象外（*）。 • IKM: UD5 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM:

	フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
NEW	• アクション: 表に追加された列。表の変更された列の新しいバージョン。
OLD	• アクション: 表から削除された列。表の変更された列の旧バージョン。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

（*）重要な注意: LKM で、*で示した一部のセレクタを使用することは、可能ですが非推奨です。インタフェースでソース上にマップされた列のみが返されます。**結果として、インタフェースにより、結果が正しくないことがあります。**
たとえば、UK セレクタでは、マップされないか、ソース上で実行されないキーの列は、このセレクタで返されません。

例

CUSTOMER 表に列 (CUST_ID, CUST_NAME, AGE)が含まれ、作成するコードが次のようである場合:

```
create table CUSTOMER (CUST_ID Numeric(10) null, CUST_NAME VARCHAR(50)
null, AGE Numeric(3) null)
```

次のように記述します。

```
create table CUSTOMER <%=odiRef.getColList("(", "[COL_NAME]
[SOURCE_CRE_DT] null", ", ", ")", "")%>
```

説明: getColList 関数が使用され、(CUST_ID numeric(10) null, CUST_NAME varchar(50) null, AGE numeric(3) null)が生成されます。先頭と末尾はカッコで、(column, data type, null)というパターンが、各列につきカンマで区切られて繰り返されます。このため、

- 関数の最初の文字**"("**は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ**"[COL_NAME] [SOURCE_CRE_DT] null"**は、このパターンを各列に対して繰り返すことを示します。キーワード[COL_NAME]および[SOURCE_CRE_DT]は、表「パターン属性リスト」の有効なキーワードに対する参照です。
- 3 番目のパラメータ**", "**は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字**")"**は、文字列が、文字列「)」で終わることを示します。
- 最後のパラメータ**""**は、パターンを各列に対して（選択内容なしで）繰り返すことを示します。

getFK()メソッド

使用方法

```
public java.lang.String getFK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの外部キー（または結合、参照）に関する情報を返します。現在のタスクに reference というタグが付けられている場合のみ、ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている外部キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	参照制約の内部番号。
FK_NAME	参照制約の名前。
FK_TYPE	参照制約のタイプ。

FK_ALIAS	参照表の別名（複合式の場合のみ使用）。
PK_ALIAS	参照された表の別名（複合式の場合のみ使用）。
ID_TABLE_PK	参照された表の内部番号。
PK_I_MOD	参照されたモデルの番号。
PK_CATALOG	参照された表のカタログ。
PK_SCHEMA	参照された表の物理スキーマ。
PK_TABLE_NAME	参照された表の名前。
COMPLEX_SQL	結合句の複合 SQL 文（該当する場合）。
MESS	参照制約のエラー・メッセージ。
FULL_NAME	ローカル・オブジェクトマスクで生成された外部キーのフルネーム。
<flexfield code>	この参照のフレックスフィールド値。

例

表の現在の参照キーの名前は**<%=odiRef.getFK("FK_NAME")%>**です。これは、表**<%=odiRef.getFK("PK_TABLE_NAME")%>**を参照しており、この表はスキーマ**<%=odiRef.getFK("PK_SCHEMA")%>**にあります。

getFKColList()メソッド

使用方法

```
public java.lang.String getFKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getFKColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

参照制約（外部キー）の列部分のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の外部キーの各列の要素が 1 つ含まれます。現在のタスクに reference というタグが付けられている場合のみ、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された外部キーの列のリストを、キー内の位置の順に並べて返します。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キーの列の名前。
COL_HEADING	キーの列のヘッダー。
COL_DESC	キーの列の説明。
POS	キーの列の位置。
LONGC	キーの列の長さ（精度）。
SCALE	キーの列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理オクテット数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）

CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列のレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
PK_I_COL	参照列の内部識別子。
PK_COL_NAME	参照キー列の名前。
PK_COL_HEADING	参照キー列のヘッダー。
PK_COL_DESC	参照キー列の説明。
PK_POS	参照列の位置。
PK_LONGC	参照列の長さ。
PK_SCALE	参照列の精度。
PK_FILE_POS	参照列の開始（索引）。
PK_BYTES	参照列の物理オクテット数。

PK_FILE_END_POS	参照列の終了（FILE_POS+BYTES）。
PK_IND_WRITE	参照列の書込み権限フラグ。
PK_COL_MANDATORY	参照列の必須文字（0: NULL を許可、1: NOT NULL）
PK_CHECK_FLOW	参照された列のフロー制御フラグ（0: チェックしない、1: チェック）。
PK_CHECK_STAT	参照された列の静的管理フラグ（0: チェックしない、1: チェック）。
PK_COL_FORMAT	参照された列の論理形式。
PK_COL_DEC_SEP	参照された列の小数点記号。
PK_REC_CODE_LIST	参照された列に維持されたレコード・コードのリスト。
PK_COL_NULL_IF_ERR	参照された列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
PK_DEF_VALUE	参照された列のデフォルト値。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	参照している表の現在の列のフレックスフィールド値。

例

CUSTOMER.COUNTRY_ID = CITY.ID_COUNT and CUSTOMER.CITY_ID = CITY.ID_CIT で CUSTOMER 表が CITY 表を参照している場合、

句:

```
(CUS.COUNTRY_ID = CITY.ID_COUNT and CUS.CITY_ID = CITY.ID_CIT)
```

は、次のようにも記述できます。

```
<%=odiRef.getFKColList("(", "CUS.[COL_NAME] = CITY.[PK_COL_NAME]", " and
", ")" )%>
```

説明:getFKColList 関数は、外部キーの各列上でループして、カッコで始まってカッコで終わり、**and** で区切られたパターンを外部キーの各列に対して繰り返す句を生成するために使用されます。このため、

- 関数の最初のパラメータ**"("**は、文字列を、「(」で始めることを示します。
  - 2 番目のパラメータ**"CUS.[COL_NAME] = CITY.[PK_COL_NAME]"** は、このパターンを外部キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[PK_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
  - 3 番目のパラメータ**" and "**は、パターンの発生を、文字列 and で区切ることを示します。
  - 関数の 4 番目の文字**")"**は、文字列が、「)」で終わることを示します。

getNotNullCol()メソッド

使用方法

```
public java.lang.String getNotNullCol(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの非 NULL 列に関する情報を返します。現在のタスクに mandatory というタグが付けられている場合、チェック・ナレッジ・モジュールからアクセスできます。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	現在の列の odi 内部識別子。
COL_NAME	非 NULL 列の名前。
MESS	標準エラー・メッセージ。
<flexfield code>	現在の非 NULL 列のフレックスフィールド値。

例

```
insert into...
select *
from ...
where <%=odiRef.getNotNullCol("COL_NAME")%> is null
```

getPK()メソッド

使用方法

```
public java.lang.String getPK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの主キーに関する情報を返します。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている主キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	PK 制約の内部番号。
KEY_NAME	主キーの名前。
MESS	主キー制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成された PK のフルネーム。
<flexfield code>	主キーのフレックスフィールド値。

例

表の主キーの名前: **<%=odiRef.getPK("KEY_NAME")%>**

getPKColList()メソッド

使用方法

```
public java.lang.String getPKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

説明

チェックされている主キーの列と式のリストを提供します。

リストの各要素について、pPattern パラメータが解釈され、繰り返されます。前の要素からは pSeparator パラメータによって区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の主キーの各列の要素が含まれます。現在のタスクに primary key というタグが付けられている場合、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された主キーの列のリストを、キー内の位置の順に並べて返します。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン・シーケンス内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キー列の名前。
COL_HEADING	キー列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始位置（固定ファイル）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。

CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるグループ化記号。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	現在の列のフレックスフィールド値。

例

CUSTOMER 表に主キーPK_CUSTOMER (CUST_ID, CUST_NAME)があり、作成するコードが次のようである場合:

```
create table T_PK_CUSTOMER (CUST_ID numeric(10) not null, CUST_NAME
varchar(50) not null)
```

次のように記述します。

```
create table T_<%=odiRef.getPK("KEY_NAME")%> <%=odiRef.getPKColList("(",
"[COL_NAME] [DEST_CRE_DT] not null", ", ", ")")%>
```

説明: getPKColList 関数は、(CUST_ID numeric(10) not null, CUST_NAME varchar(50) not null) の部分を生成するために使用されます。これはカッコで開始および停止し、代替キーの各列について、カンマで区切られたパターン（列、データ型、NOT NULL）を繰り返します。このため、

- 関数の最初のパラメータ"("は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ"[COL_NAME] [DEST_CRE_DT] not null"は、このパターンを主キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[DEST_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ", "は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字")"は、文字列が、文字列「)」で終わることを示します。

getPop()メソッド

使用方法

```
public java.lang.String getPop(java.lang.String pPropertyName)
```

説明

現在のインタフェースの概要を返す汎用メソッド。使用可能な情報のリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
I_POP	インタフェースの内部番号。
FOLDER	インタフェースのフォルダの名前。
POP_NAME	インタフェースの名前。
IND_WORK_TARG	ステージング領域の位置フラグ。
LSCHEMA_NAME	インタフェースのステージング領域である論理スキーマの名前。
DESCRIPTION	インタフェースの説明。
WSTAGE	ターゲット・データストアの性質を示しているフラグ: • E: ターゲット・データストアは既存の表です（一時表ではありません）。

	• N: ターゲット・データストアはデータ・スキーマ内の一時表です。 • W: ターゲット・データストアは作業スキーマ内の一時表です。
TABLE_NAME	ターゲット表の名前。
KEY_NAME	更新キーの名前。
DISTINCT_ROWS	重複抑制フラグ。
OPT_CTX	インタフェースの最適化コンテキストの名前。
TARG_CTX	インタフェースの実行コンテキストの名前。
MAX_ERR	許容される最大エラー数。
MAX_ERR_PRCT	割合によるエラー・インジケータ。
IKM	統合ナレッジ・モジュールの名前。
LKM	ロード・ナレッジ・モジュールの名前。
CKM	チェック・ナレッジ・モジュールの名前。
HAS_JRN	インタフェースのソースにジャーナル化された表がある場合、1 を返します。それ以外の場合は 0 を返します。
<flexfield code>	インタフェースのフレックスフィールド値。

例

現在のインタフェースは**<%=odiRef.getPop("POP_NAME")%>**で、論理スキーマ**<%=odiRef.getInfo("L_SCHEMA_NAME")%>**上で実行されています。

getTable()メソッド

使用方法

```
public java.lang.String getTable(
java.lang.String pMode,
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty)
```

説明

odi によって処理された一時的表および永続的表のフルネームを取得します。

パラメータ

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>pMode</td><td>文字列</td><td>"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。この値は、pMode が指定されていない場合に使用されます。
"R"はオブジェクト・マスクを使用してオブジェクトの完全パスを作成します。
"A"は自動。使用する適切なマスクを自動的に定義します。</td></tr>
<tr><td>pProperty</td><td>文字列</td><td>構築される表の名前を示すパラメータ。可能な値のリストを次に示します。
<table>
<tr><th>パラメータ値</th><th>説明</th></tr>
<tr><td>ID</td><td>データストア識別子。</td></tr>
<tr><td>TARG_NAME</td><td>ターゲット・データストアのフルネーム。アクションでは、このパラメータは、DDL コマンドによって処理された現在の表の名前を返します。</td></tr>
<tr><td>COLL_NAME</td><td>ロード・データストアのフルネーム。</td></tr>
<tr><td>INT_NAME</td><td>統合データストアのフルネーム。</td></tr>
<tr><td>ERR_NAME</td><td>エラー・データストアのフルネーム。</td></tr>
<tr><td>CHECK_NAME</td><td>エラー・サマリー・データストアの名前。</td></tr>
<tr><td>CT_NAME</td><td>チェック・データストアのフルネーム。</td></tr>
<tr><td>FK_PK_TABLE_NAME</td><td>外部キーによって参照されたデータストアのフルネーム。</td></tr>
<tr><td>JRN_NAME</td><td>ジャーナル化されたデータストアのフルネーム。</td></tr>
<tr><td>JRN_VIEW</td><td>ジャーナル化されたデータストアにリンクされたビューのフルネーム。</td></tr>
<tr><td>JRN_DATA_VIEW</td><td>ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。</td></tr>
<tr><td>JRN_TRIGGER</td><td>ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。</td></tr>
</table>
</td></tr>
</table>

		JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
		JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
		JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
		SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアのフルネーム。
		CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
		CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
		CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
		CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
		<flexfield code>	現在のターゲット表のフレックスフィールド値。
pLocation	文字列	"W"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理作業スキーマのフルネームを返します。
		"D"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理データ・スキーマのフルネームを返します。
		"A"	odi がオブジェクトのデフォルトの場所を決定します。この値は、pLocation が指定されていない場合に使用されます。

例

定義されている要素:

物理スキーマ: Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo

作業カタログ:	tempdb
作業スキーマ:	temp_owner
ローカル・マスク:	%CATALOG.%SCHEMA.%OBJECT
リモート・マスク:	%DSERVER:%CATALOG.%SCHEMA.%OBJECT
ロード接頭辞:	CZ_
エラー接頭辞:	ERR_
統合接頭辞:	I$_

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

表の名前: CUSTOMER

コール対象	戻り値
`<%=odiRef.getTable("L", "COLL_NAME", "W")%>`	tempdb.temp_owner.CZ_0CUSTOMER
`<%=odiRef.getTable("R", "COLL_NAME", "D")%>`	MyServer:db_odi.dbo.CZ_0CUSTOMER
`<%=odiRef.getTable("L", "INT_NAME", "W")%>`	tempdb.temp_owner.I$_CUSTOMER
`<%=odiRef.getTable("R", "ERR_NAME", "D")%>`	MyServer:db_odi.dbo.ERR_CUSTOMER

getTargetTable()メソッド

使用方法

```
public java.lang.String getTargetTable(java.lang.String pPropertyName)
```

説明

現在のターゲット表の概要を返す汎用メソッド。使用可能なデータのリストを **pPropertyName の値**の表に示します。

アクションでは、このメソッドは DDL コマンドによって処理されている表に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
I_TABLE	データストアの内部識別子。
MODEL_NAME	現在のデータストアのモデルの名前。
SUB_MODEL_NAME	現在のデータストアのサブモデルの名前。
TECHNO_NAME	ターゲット・テクノロジの名前。
LSCHEMA_NAME	ターゲット論理スキーマの名前。
TABLE_NAME	ターゲット・データストアの名前。
RES_NAME	ターゲット・リソースの物理名。
CATALOG	カタログ名。
WORK_CATALOG	作業カタログの名前。
SCHEMA	スキーマ名。
WORK_SCHEMA	作業スキーマの名前。
TABLE_ALIAS	現在のデータストアの別名。
TABLE_TYPE	データストアのタイプ。
DESCRIPTION	現在のインタフェースの説明。
TABLE_DESC	現在のインタフェースのターゲット・データストアの説明。DDL コマンドの場合、現在の表の説明。
R_COUNT	現在のデータストアの行数。
FILE_FORMAT	現在のデータストア（ファイル）の形式。
FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。
XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。

FILE_FIRST_ROW	ファイル（ファイル）の先頭の、無視する行数。
FILE_DEC_SEP	小数点記号（ファイル）。
METADATA_DESC	データストアのメタデータの説明（ファイル）
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
TABLE_DESC	表の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
WS_NAME	データ・サービス: このデータストアのモデル用に生成された Web サービスの名前。
WS_NAMESPACE	データ・サービス: Web サービスの XML ネームスペース。
WS_JAVA_PACKAGE	データ・サービス: Web サービス用に生成された Java パッケージ。
WS_ENTITY_NAME	データ・サービス: Web サービスでこのデータストアに対して使用されるエンティティ名。
WS_DATA_SOURCE	データ・サービス: このデータストアの Web サービスに対して指定されたデータソース。
<flexfield code>	現在の表のフレックスフィールド値。

例

現在の表: **<%=odiRef.getTargetTable("RES_NAME")%>**

getTargetColList()メソッド

使用方法

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getTargetColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースのターゲット表の列のリストを提供します。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column of the target
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。

DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。

	• CKM: 適用対象外。
REW	• LKM: 適用対象外。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

例

```
create table TARGET_COPY <%=odiRef.getTargetColList("(", "[COL_NAME]
[DEST_DT] null", ", ", ")", "")%>
```

統合メソッド（IKM）

getColList()メソッド

使用方法

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pSelector)
```

```
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

列と式のリストを提供します。列リストはこのメソッドがコールされたフェーズにより異なります。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

ロード（LKM）

現在のソース環境で実行されたすべてのマッピング式と、ステージング領域で実行されたマッピング、フィルタ式、結合で使用された列。

インタフェースで execute というタグを付けられたマッピングのみが対象です。

- リストは POS、FILE_POS でソートされます。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

統合（IKM）

現在のインタフェースで execute というタグを付けられたすべての現在のマッピング式。

リストには、現在のインタフェースのターゲット表に（execute というタグを付けて）ロードされる各列につき、1 つの要素が含まれます。

- ロードされた表が一時表でない場合、リストは POS、FILE_POS でソートされます。
- ロードされた表が一時表である（参照にない）場合、リストはソートされません。

インタフェースのソースにジャーナル化されたデータストアがあり、それがステージング領域にある場合、3 つのジャーナル化擬似列 `JRN_FLG`、`JRN_DATE` および `JRN_SUBSCRIBER` は、ジャーナル化されたソース・データストアの列として追加されます。

チェック（CKM）

ターゲット表のすべての列（静的管理またはフロー制御）

ターゲット表の列を、現在のインタフェースで記入された列から区別するには、MAP セレクタを使用する必要があります。

- リストはターゲット表の POS、FILE_POS でソートされます。

アクション

DDL コマンドが処理した表のすべての列。

変更されたか、追加したか、削除された列の場合、NEW および OLD セレクタを使用して、DDL コマンドによって処理される、変更された列の新バージョンまたは旧バージョンを取得することができます。

- リストは表の POS、FILE_POS でソートされます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。

	<SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。

DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。
CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、`NOT NULL` を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、`DEFAULT` <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外（*）。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。

UPD	• LKM: 適用対象外（*）。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外（*）。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外（*）。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外（*）。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外（*）。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外（*）。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
UD1	• LKM: 適用対象外（*）。 • IKM: UD1 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD2	• LKM: 適用対象外（*）。 • IKM: UD2 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD3	• LKM: 適用対象外（*）。 • IKM: UD3 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD4	• LKM: 適用対象外（*）。 • IKM: UD4 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD5	• LKM: 適用対象外（*）。

	• IKM: UD5 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
NEW	• アクション: 表に追加された列。表の変更された列の新しいバージョン。
OLD	• アクション: 表から削除された列。表の変更された列の旧バージョン。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

（*）重要な注意: LKM で、*で示した一部のセレクタを使用することは、可能ですが非推奨です。インタフェースでソース上にマップされた列のみが返されます。**結果として、インタフェースにより、結果が正しくないことがあります。**
たとえば、UK セレクタでは、マップされないか、ソース上で実行されないキーの列は、このセレクタで返されません。

例

CUSTOMER 表に列 (CUST_ID, CUST_NAME, AGE)が含まれ、作成するコードが次のようである場合:

```
create table CUSTOMER (CUST_ID Numeric(10) null, CUST_NAME VARCHAR(50)
null, AGE Numeric(3) null)
```

次のように記述します。

```
create table CUSTOMER <%=odiRef.getColList("(", "[COL_NAME]
[SOURCE_CRE_DT] null", ", ", ")", "")%>
```

説明: getColList 関数が使用され、(CUST_ID numeric(10) null, CUST_NAME varchar(50) null, AGE numeric(3) null)が生成されます。先頭と末尾はカッコで、(column, data type, null)というパターンが、各列につきカンマで区切られて繰り返されます。このため、

- 関数の最初の文字**"("**は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ**"[COL_NAME] [SOURCE_CRE_DT] null"**は、このパターンを各列に対して繰り返すことを示します。キーワード[COL_NAME]および[SOURCE_CRE_DT]は、表「パターン属性リスト」の有効なキーワードに対する参照です。
- 3 番目のパラメータ**", "**は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字**")"**は、文字列が、文字列「)」で終わることを示します。
- 最後のパラメータ**""**は、パターンを各列に対して（選択内容なしで）繰り返すことを示します。

getFilter()メソッド

使用方法

```
public java.lang.String getFilter()
```

説明

SQL フィルタ列を返します（ロード時はソースで、統合時はステージング領域で）。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getFilterList()メソッド

使用方法

```
public java.lang.String getFilterList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getFilterList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL フィルタの発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用中のナレッジ・モジュールにより異なります）で実行される各フィルタ式の要素が含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 リスト内で使用できるもののリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切るために使用されます。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	フィルタ内部識別子
EXPRESSION	フィルタ式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getFilterList 関数は、SELECT 句のフィルタを生成するために使用されます。フィルタは and で始まり、各フィルタごとに and で区切ってパターン（各フィルタの式）を繰り返します。このため、

- 関数の最初のパラメータ**"and"**は、文字列を、文字列 and で始めることを示します。
- 2 番目のパラメータ"([EXPRESSION])" は、このパターンを各フィルタに対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ" and "は、パターンの解釈された発生を、文字列 and で区切ることを示します。
- 関数の 4 番目のパラメータ""は、文字列が、指定された文字を伴わずに終わることを示します。

getFrom()メソッド

使用方法

```
public java.lang.String getFrom()
```

説明

FROM の SQL 文字列をソースの **SELECT** 句から取得します。**FROM** 文は、インタフェースで使用される表および結合（テクノロジの SQL 機能に応じて）から構築されます。このため、ISO の外部結合とカッコをサポートするテクノロジでは、getFrom()が次のような文字列を返すことがあります。

```
((CUSTOMER as CUS inner join CITY as CIT on (CUS.CITY_ID = CIT.CITY_ID))
left outer join SALES_PERSON as SP on (CUS.SALES_ID = SP.SALE_ID))
```

インタフェースのソースにジャーナル化されたデータストアがある場合、句内のソース表は、ジャーナル化されたソース・データストアにリンクされたデータ・ビューによって置き換えられます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getGrpBy()メソッド

使用方法

```
public java.lang.String getGrpBy()
```

説明

SQL GROUP BY 文字列を取得します（収集フェーズではソース上、統合フェーズではステージング領域上）。この文は、マッピング式に検出された集計変換から自動的に計算されます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getGrpByList()メソッド

使用方法

```
public java.lang.String getGrpByList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getGrpByList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL GROUP BY の発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用するナレッジ・モジュールにより異なります）の GROUP BY 文の要素が含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pMotif	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切るために使用されます。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	句の内部識別子。
EXPRESSION	グループ化文のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
```

```
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpByList("group by ", "[EXPRESSION]", " , ", "")%>
<%=odiRef.getHaving()%>
```

説明: getGrpByList 関数は、group by で始まり、各式ごとにコンマによって区切られたパターン（各グループ化式）が繰り返される **select** 命令の **group by** 句を生成するために使用されます。

- 関数の最初のパラメータ**"group by"**は、文字列の先頭に「group by」を置くことを示します。
- 2 番目のパラメータ"[EXPRESSION]"は、式によって各グループについてこのパターンを繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ","は、パターンの解釈された発生を、カンマで区切ることを示します。
- 関数の 4 番目のパラメータ""は、文字列が、指定された文字を伴わずに終わることを示します。

getHaving()メソッド

使用方法

```
public java.lang.String getHaving()
```

説明

SQL 文 HAVING を取得します（ロード時はソースで、統合時はステージング領域で）。この文は、検出された集計関数が含まれているフィルタ式から自動的に計算されます。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getHavingList()メソッド

使用方法

```
public java.lang.String getHavingList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getHavingList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースの SQL HAVING の発生のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、ソースまたはターゲット上（使用中のナレッジ・モジュールにより異なります）で実行される各 HAVING 式の要素が 1 つ含まれます。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	句の内部識別子。
EXPRESSION	having 式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
```

```
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpByList("group by ", "[EXPRESSION]", " , ", "")%>
<%=odiRef.getHavingList("having ", "([EXPRESSION])", " and ", "")%>
```

説明: getHavingList 関数は、select 命令の having 句を生成するために使用されます。この句は having で始まり、各式ごとに and で区切ってパターン（各集計フィルタ式）を繰り返します。

- 関数の最初のパラメータ**"having "**は、文字列の先頭に having を置くことを示します。
- 2 番目のパラメータ**"([EXPRESSION])"**は、このパターンを各フィルタに対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ**" and "**は、パターンの解釈された発生を、文字列 and で区切ることを示します。
- 関数の 4 番目のパラメータ**""**は、文字列が、指定された文字を伴わずに終わることを示します。

getJoin()メソッド

使用方法

```
public java.lang.String getJoin()
```

説明

SQL 結合文字列を取得します（ロード時はソースで、統合時はステージング領域で）。

パラメータ

なし

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoin()%>
<%=odiRef.getFilter()%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

getJoinList()メソッド

使用方法

```
public java.lang.String getJoinList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getJoinList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェース内の SQL 結合の発生のリストを提供して、WHERE 句内に位置付けます。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
ID	結合の内部識別子。
EXPRESSION	結合式のテキスト

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getFrom()%>
where (1=1)
<%=odiRef.getJoinList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
```

```
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getJoinList 関数は、SELECT 文の WHERE 部分に置く結合式を生成するために使用されます。式は and で始まり、各結合ごとに and で区切ってパターン（各結合の式）を繰り返します。このため、

- 関数の最初のパラメータ**"and"**は、文字列を and で始めることを示します。
- 2 番目のパラメータ**"([EXPRESSION])"**は、このパターンを各結合に対して繰り返すことを示します。キーワード[EXPRESSION]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ**" and "**は、パターンの解釈された発生を、and で区切ることを示します（and の前後のスペースに注意してください）。
- 関数の 4 番目のパラメータ**""**は、文字列が、指定された文字を伴わずに終わることを示します。

getJrnInfo()メソッド

使用方法

```
public java.lang.String getJrnInfo(java.lang.String pPropertyName)
```

説明

モデルやデータストアのジャーナル化の場合は JKM、インタフェースの場合は LKM/IKM に対するデータストアのジャーナル化に関する概要を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
FULL_TABLE_NAME	ジャーナル化されたデータストアのフルネーム。
JRN_FULL_NAME	ジャーナル・データストアのフルネーム。
JRN_FULL_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
JRN_FULL_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
JRN_FULL_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。

JRN_FULL_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
JRN_FULL_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
JRN_FULL_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
JRN_SUBSCRIBER	作業スキーマにおけるサブスクライバ表の名前。
JRN_NAME	ジャーナル化されたデータストアの名前。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューの名前。
JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューの名前。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーの名前。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーの名前。
JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーの名前。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーの名前。
JRN_SUBSCRIBER	サブスクライバの名前。
JRN_COD_MODE	ジャーナル化されたデータ・モデルのコード。
JRN_METHOD	ジャーナル化モード（一貫または単純）。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。

例

ジャーナル化される表: **<%=odiRef.getJrnInfo("FULL_TABLE_NAME")%>**

getJrnFilter()メソッド

使用方法

```
public java.lang.String getJrnFilter()
```

説明

現在のインタフェースの SQL ジャーナル化フィルタを返します。ジャーナル化された表がソースにある場合、ロード・フェーズでこのメソッドを使用できます。ジャーナル化された表がステージング領域にある場合、統合時にこのメソッドを使用できます。

パラメータ

なし

例

```
<%=odiRef.getJrnFilter()%>
```

getPop()メソッド

使用方法

```
public java.lang.String getPop(java.lang.String pPropertyName)
```

説明

現在のインタフェースの概要を返す汎用メソッド。使用可能な情報のリストを **pPropertyName の値**の表に示します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
I_POP	インタフェースの内部番号。
FOLDER	インタフェースのフォルダの名前。

POP_NAME	インタフェースの名前。
IND_WORK_TARG	ステージング領域の位置フラグ。
LSCHEMA_NAME	インタフェースのステージング領域である論理スキーマの名前。
DESCRIPTION	インタフェースの説明。
WSTAGE	ターゲット・データストアの性質を示しているフラグ: • E: ターゲット・データストアは既存の表です（一時表ではありません）。 • N: ターゲット・データストアはデータ・スキーマ内の一時表です。 • W: ターゲット・データストアは作業スキーマ内の一時表です。
TABLE_NAME	ターゲット表の名前。
KEY_NAME	更新キーの名前。
DISTINCT_ROWS	重複抑制フラグ。
OPT_CTX	インタフェースの最適化コンテキストの名前。
TARG_CTX	インタフェースの実行コンテキストの名前。
MAX_ERR	許容される最大エラー数。
MAX_ERR_PRCT	割合によるエラー・インジケータ。
IKM	統合ナレッジ・モジュールの名前。
LKM	ロード・ナレッジ・モジュールの名前。
CKM	チェック・ナレッジ・モジュールの名前。
HAS_JRN	インタフェースのソースにジャーナル化された表がある場合、1 を返します。それ以外の場合は 0 を返します。
<flexfield code>	インタフェースのフレックスフィールド値。

例

現在のインタフェースは**<%=odiRef.getPop("POP_NAME")%>**で、論理スキーマ**<%=odiRef.getInfo("L_SCHEMA_NAME")%>**上で実行されています。

getSrcColList()メソッド

使用方法

```
public java.lang.String getSrcColList( java.lang.String pStart,
java.lang.String pUnMappedPattern,
```

```
java.lang.String pMappedPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

説明

このメソッドは LKM と IKM に用意されていて、列のリストのプロパティを返します。このリストは LKM（ソース）または IKM（ステージング領域）によって処理されたソースのすべての列を含みます。リストは、ソース表内の列位置を基準にソートされます。

表示されるプロパティは、列がマップされているかどうかにより異なります。列がマップされている場合、返されるプロパティは pMappedPattern パターンで定義されます。列がマップされていない場合、返されるプロパティは pUnMappedPattern パターンで定義されます。

パターンで使用できる属性の詳細は、「パターン属性リスト」に詳述されます。属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。たとえば、「My string [COL_NAME] is a column」です。

pMappedPattern または pUnMappedPattern パラメータがリストの各要素について解釈され、繰り返されます。パターンは pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 JRN_FLG、JRN_DATE および JRN_SUBSCRIBER は、ジャーナル化されたソース・データストアの列として追加されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pUnMappedPattern	文字列	列がマップされていない場合、パターンはリスト内に現れるたびに繰り返されます。
pMappedPattern	文字列	列がマップされている場合、パターンはリスト内に現れるたびに繰り返されます。
pSeparator	文字列	このパラメータはパターンを区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
ALIAS_NAME	列の名前。COL_NAME と異なり、この属性はオプションのテクノロ

	ジ・デリミタを付けずに列名を返します。列名に、たとえばスペースが含まれる場合、これらのデリミタが表示されます。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース（LKM）またはステージング領域（IKM）上で実行された、式のテキスト（マッピング・フィールドに入力したもの）。列がマップされていない場合、このパラメータは空の文字列を返します。
CX_COL_NAME	サポートされていません。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット（IKM）またはステージング領域（LKM）のテクノロジのデータ型に変換された、列のデータ型のコード。

DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
<flexfield code>	現在の列のフレックスフィールド値。

例

ソース・ファイルに類似した表を作成する場合:

```
create table <%=odiRef.getTable("L","COLL_NAME", "D")%>_F
(
<%=odiRef.getSrcColList("","[COL_NAME] [DEST_CRE_DT]","[COL_NAME]
[DEST_CRE_DT]",",\n","")%>
)
```

getSrcTablesList()メソッド

使用方法

```
public java.lang.String getSrcTablesList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getSrcTablesList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースのソース表のリストを提供します。このメソッドは SELECT 命令に FROM 句を作成するために使用できます。ただし、これでなく getFrom()メソッドを使用することをお薦めします。

pPattern パターンはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

属性	説明
I_TABLE	現在のソース表の odi 内部番号（番号がある場合）。
MODEL_NAME	現在のソース表のモデルの名前（名前がある場合）。
SUB_MODEL_NAME	現在のソース表のサブモデルの名前（名前がある場合）。
TECHNO_NAME	ソース・データストアのテクノロジの名前。
LSCHEMA_NAME	ソース表の論理スキーマ。
TABLE_NAME	ソース・データストアの論理名。
RES_NAME	リソースの物理アクセス名（ファイル名または JMS キュー、表の物理名など）。 インタフェースのソースにジャーナル化されたデータストアがある場合、句内のソース表は、ジャーナル化されたソース・データストアにリンクされたデータ・ビューによって置き換えられます。
CATALOG	ソース・データストアのカタログ（実行時に解決）。
WORK_CATALOG	ソース・データストアの作業カタログ。
SCHEMA	ソース・データストアのスキーマ（実行時に解決）。
WORK_SCHEMA	ソース・データストアの作業スキーマ。

TABLE_ALIAS	表リストに表示されるデータストアの別名（別名がある場合）。
POP_TAB_ALIAS	現在のインタフェースに表示されるデータストアの別名（別名がある場合）。
TABLE_TYPE	データストア・ソースのタイプ（データストア・ソースがある場合）。
DESCRIPTION	ソース・データストアの説明（説明がある場合）。
R_COUNT	ソース・データストアのレコードの数（判明している場合）。
FILE_FORMAT	ファイル形式（判明している場合）。
FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。
XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。
FILE_FIRST_ROW	無視するヘッダーの行数（ヘッダーがある場合）。
FILE_DEC_SEP	データストアのデフォルトの小数点記号（デフォルトがある場合）。
METADATA	現在のリソースのメタデータの odi 形式の説明（説明がある場合）。
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
<flexfield code>	現在の表のフレックスフィールド値。

例

```
insert into <%=odiRef.getTable("L", "COLL_NAME", "W")%>
select <%=odiRef.getColList("", "[EXPRESSION]", ", ", "", "INS=1")%>
from <%=odiRef.getSrcTablesList("", "[CATALOG].[SCHEMA].[TABLE_NAME] as
[POP_TAB_ALIAS]", ", ", "")%>
where (1=1)
<%=odiRef.getJoinList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getFilterList("and ","([EXPRESSION])"," and ","")%>
<%=odiRef.getGrpBy()%>
<%=odiRef.getHaving()%>
```

説明: getSrcTablesList 関数が、ソース内の各表についてカンマで区切られたパターンを繰り返す SELECT 文の FROM 句を生成するパターン（CATALOG.SCHEMA.TABLE_NAME as POP_TAB_ALIAS）を繰り返すために使用されます。

- 関数の最初のパラメータ""は、文字列を、特定の文字で開始しないことを示します。
- 関数の 2 番目のパラメータ"[CATALOG].[SCHEMA].[TABLE_NAME] as [POP_TAB_ALIAS]"は、このパターンを各ソース表について繰り返すことを示します。キーワード[CATALOG]、[SCHEMA]、[TABLE_NAME]および[POP_TAB_ALIAS]は、表「パターン属性リスト」の有効なキーワードを参照します。
- 3 番目のパラメータ","は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目のパラメータ""は、文字列が、指定された文字を伴わずに終わることを示します。

getTable()メソッド

使用方法

```
public java.lang.String getTable(
java.lang.String pMode,
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty)
```

説明

odi によって処理された一時的表および永続的表のフルネームを取得します。

パラメータ

パラメータ	タイプ	説明
pMode	文字列	"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。この値は、pMode が指定されていない場合に使用されます。 "R"はオブジェクト・マスクを使用してオブジェクトの完全パスを作成します。 "A"は自動。使用する適切なマスクを自動的に定義します。
pProperty	文字列	構築される表の名前を示すパラメータ。可能な値のリストを次に示します。

パラメータ値	説明
ID	データストア識別子。
TARG_NAME	ターゲット・データストアのフルネーム。アクションでは、このパラメータは、DDL コマンドによって処理された現在の表の名前を返します。
COLL_NAME	ロード・データストアのフルネーム。
INT_NAME	統合データストアのフルネーム。
ERR_NAME	エラー・データストアのフルネーム。
CHECK_NAME	エラー・サマリー・データストアの名前。
CT_NAME	チェック・データストアのフルネーム。
FK_PK_TABLE_NAME	外部キーによって参照されたデータストアのフルネーム。
JRN_NAME	ジャーナル化されたデータストアのフルネーム。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアのフルネーム。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。

		CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。
		CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
		CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
		<flexfield code>	現在のターゲット表のフレックスフィールド値。
pLocation	文字列	"W"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理作業スキーマのフルネームを返します。
		"D"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理データ・スキーマのフルネームを返します。
		"A"	odi がオブジェクトのデフォルトの場所を決定します。この値は、pLocation が指定されていない場合に使用されます。

例

定義されている要素:

物理スキーマ: Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner
ローカル・マスク:	%CATALOG.%SCHEMA.%OBJECT
リモート・マスク:	%DSERVER:%CATALOG.%SCHEMA.%OBJECT
ロード接頭辞:	CZ_
エラー接頭辞:	ERR_
統合接頭辞:	I$_

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

表の名前: CUSTOMER

コール対象	戻り値
<%=odiRef.getTable("L", "COLL_NAME", "W")%>	tempdb.temp_owner.CZ_0CUSTOMER
<%=odiRef.getTable("R", "COLL_NAME", "D")%>	MyServer:db_odi.dbo.CZ_0CUSTOMER
<%=odiRef.getTable("L", "INT_NAME", "W")%>	tempdb.temp_owner.I$_CUSTOMER
<%=odiRef.getTable("R", "ERR_NAME", "D")%>	MyServer:db_odi.dbo.ERR_CUSTOMER

getTargetTable()メソッド

使用方法

```
public java.lang.String getTargetTable(java.lang.String pPropertyName)
```

説明

現在のターゲット表の概要を返す汎用メソッド。使用可能なデータのリストを **pPropertyName の値**の表に示します。

アクションでは、このメソッドは DDL コマンドによって処理されている表に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
I_TABLE	データストアの内部識別子。
MODEL_NAME	現在のデータストアのモデルの名前。
SUB_MODEL_NAME	現在のデータストアのサブモデルの名前。
TECHNO_NAME	ターゲット・テクノロジの名前。
LSCHEMA_NAME	ターゲット論理スキーマの名前。

TABLE_NAME	ターゲット・データストアの名前。
RES_NAME	ターゲット・リソースの物理名。
CATALOG	カタログ名。
WORK_CATALOG	作業カタログの名前。
SCHEMA	スキーマ名。
WORK_SCHEMA	作業スキーマの名前。
TABLE_ALIAS	現在のデータストアの別名。
TABLE_TYPE	データストアのタイプ。
DESCRIPTION	現在のインタフェースの説明。
TABLE_DESC	現在のインタフェースのターゲット・データストアの説明。DDL コマンドの場合、現在の表の説明。
R_COUNT	現在のデータストアの行数。
FILE_FORMAT	現在のデータストア（ファイル）の形式。
FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。
XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。
FILE_FIRST_ROW	ファイル（ファイル）の先頭の、無視する行数。
FILE_DEC_SEP	小数点記号（ファイル）。
METADATA_DESC	データストアのメタデータの説明（ファイル）
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
TABLE_DESC	表の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。

WS_NAME	データ・サービス: このデータストアのモデル用に生成された Web サービスの名前。
WS_NAMESPACE	データ・サービス: Web サービスの XML ネームスペース。
WS_JAVA_PACKAGE	データ・サービス: Web サービス用に生成された Java パッケージ。
WS_ENTITY_NAME	データ・サービス: Web サービスでこのデータストアに対して使用されるエンティティ名。
WS_DATA_SOURCE	データ・サービス: このデータストアの Web サービスに対して指定されたデータソース。
<flexfield code>	現在の表のフレックスフィールド値。

例

現在の表: **<%=odiRef.getTargetTable("RES_NAME")%>**

getTargetColList()メソッド

使用方法

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getTargetColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

```
public java.lang.String getTargetColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

インタフェースのターゲット表の列のリストを提供します。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column of the target
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。

SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。

COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。
<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM:

	フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

例

```
create table TARGET_COPY <%=odiRef.getTargetColList("(", "[COL_NAME]
[DEST_DT] null", ", ", ")", "")%>
```

リバース・メソッド（RKM）

getModel()メソッド

使用方法

```
public java.lang.String getModel(java.lang.String pPropertyName)
```

説明

このメソッドは、パーソナライズされたリバース・エンジニアリングの処理時に、現在のデータ・モデルに関する情報を返します。使用可能なデータのリストを **pPropertyName の値**の表に示します。

注意: このメソッドはソース接続（リバース・エンジニアリングされるデータ・サーバー）上でも、ターゲット接続（odi リポジトリ）上でも使用できます。ターゲット接続では、コンテキストから独立したプロパティのみを指定できます（たとえば、スキーマ名やカタログ名は使用できません）。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
ID	現在のモデルの内部識別子。
MOD_NAME	現在のモデルの名前。
LSCHEMA_NAME	現在のモデルの論理スキーマの名前。
MOD_TEXT	現在のモデルの説明。
REV_TYPE	リバース・エンジニアリングのタイプ: S は標準リバース、C はカスタマイズ。
REV_UPDATE	モデルの更新フラグ。
REV_INSERT	モデルの挿入フラグ。
REV_OBJ_PATT	リバースするオブジェクト用のマスク。
REV_OBJ_TYPE	このモデルでリバース・エンジニアするオブジェクト・タイプのリスト。これは、セミコロンで区切られた、オブジェクト・タイプ・コードのリストです。有効なコードは次のとおりです。 • T: 表 • V: ビュー • Q: キュー • SY: システム表 • AT: 表の別名 • SY: シノニム

TECH_INT_NAME	現在のモデルのテクノロジの内部名。
LAGENT_NAME	リバース・エンジニアリング用の論理実行エージェントの名前。
REV_CONTEXT	リバースの実行コンテキスト。
REV_ALIAS_LTRIM	別名生成で抑制される文字。
CKM	チェック・ナレッジ・モジュール。
RKM	リバース・エンジニアリング・ナレッジ・モジュール。
SCHEMA_NAME	現在のリバース・コンテキストでのデータ・スキーマの物理名。
WSCHEMA_NAME	現在のリバース・コンテキストでの作業スキーマの物理名。
CATALOG_NAME	現在のリバース・コンテキストでのデータ・カタログの物理名。
WCATALOG_NAME	現在のリバース・コンテキストでの作業カタログの物理名。
<flexfield code>	現在のモデルのフレックスフィールドの値。

例

リバースするオブジェクトのマスクの一部である表のリストを取得します。

```
select TABLE_NAME,
     RES_NAME,
     replace(TABLE_NAME, '<%=odiRef.getModel("REV_ALIAS_LTRIM")%>' , '')
     ALIAS,
     TABLE_DESC
from MY_TABLES
```

ここで、

```
TABLE_NAME like '<%=odiRef.getModel("REV_OBJ_PATT")%>'
```

Web サービス・メソッド（SKM）

hasPK()メソッド

使用方法

```
public java.lang.Boolean hasPK()
```

説明

このメソッドはブーリアンを返します。Web サービスが生成されているデータストアに主キーがある場合、戻り値は true です。

このメソッドは SKM でのみ使用できます。

例

```
<% if (odiRef.hasPK()) { %>
     There is a PK :
     <%=odiRef.getPK("KEY_NAME")%> : <%=odiRef.getPKColList("{",
     "\u0022[COL_NAME]\u0022", ", ", "}")%>
<% } else {%>
     There is NO PK.
<% } %>
```

nextAK()メソッド

使用方法

```
public java.lang.Boolean nextAK()
```

説明

このメソッドは、Web サービスが生成されているデータストアの次の代替キー（AK）に移ります。

最初にコールされると、このメソッドは true を返し、現在の AK をデータストアの最初の AK に置きます。データストアに AK がない場合は false を返します。

次回以降のコールは、現在の AK をデータストアの次の AK に置き、true を返します。次の AK がない場合は false を返します。

このメソッドは SKM でのみ使用できます。

例

次の例では、データストアのすべての AK に対して反復しています。while ループの繰返しごとに、getAK と getAKColList メソッドは、データストアの各種の AK に関する情報を返します。

```
<% while (odiRef.nextAK()) { %>
    <%=odiRef.getAK("KEY_NAME")%>
        Columns <%=odiRef.getAKColList("{", "\u0022[COL_NAME]\u0022", ",
", "}")%>
        メッセージ: <%=odiRef.getAK("MESS")%>
<% } %>
```

nextCond()メソッド

使用方法

```
public java.lang.Boolean nextCond()
```

説明

このメソッドは、Web サービスが生成されているデータストアの次の条件（チェック制約）に移ります。

最初にコールされると、このメソッドは true を返し、現在の条件をデータストアの最初の条件に置きます。データストアに条件がない場合は false を返します。

次回以降のコールは、現在の条件をデータストアの次の条件に置き、true を返します。次の条件がない場合は false を返します。

このメソッドは SKM でのみ使用できます。

例

次の例では、データストアのすべての条件に対して反復しています。while ループの繰返しごとに、getCK メソッドは、データストアの各種の条件に関する情報を返します。

```
<% while (odiRef.nextCond()) { %>
    <%=odiRef.getCK("COND_NAME")%>
        SQL :<%=odiRef.getCK("COND_SQL")%>
        MESS :<%=odiRef.getCK("MESS")%>
<% } %>
```

nextFK()メソッド

使用方法

```
public java.lang.Boolean nextFK()
```

説明

このメソッドは、Web サービスが生成されているデータストアの次の外部キー（FK）に移ります。

最初にコールされると、このメソッドは true を返し、現在の FK をデータストアの最初の FK に置きます。データストアに FK がない場合は false を返します。

次回以降のコールは、現在の FK をデータストアの次の FK に置き、true を返します。次の FK がない場合は false を返します。

このメソッドは SKM でのみ使用できます。

例

次の例では、データストアのすべての FK に対して反復しています。while ループの繰返しごとに、getFK と getFKColList メソッドは、データストアの各種の FK に関する情報を返します。

```
<% while (odiRef.nextFK()) { %>
    FK : <%=odiRef.getFK("FK_NAME")%>
        参照されている表 : <%=odiRef.getFK("PK_TABLE_NAME")%>
        Columns <%=odiRef.getFKColList("{", "\u0022[COL_NAME]\u0022", ",
", "}")%>
```

```
        メッセージ : <%=odiRef.getFK("MESS")%>
<% } %>
```

アクション・メソッド

アクションでの置換メソッドの使用

概要

アクションは DDL 操作（表の作成、参照のドロップなど）に対応しています。各アクションには、DDL 操作を実行するために必要なコマンドに対応する、いくつかの**アクション行**が含まれます（たとえば、表を削除するには、先にその制約をすべて削除する必要があります）。

アクション行のコード

アクション行には、アクション・グループのテクノロジで有効な文が含まれます。プロシージャやナレッジ・モジュール・コマンドと異なり、これらの文は 1 つの接続を使用します（SELECT ... INSERT 文は使用できません）。ナレッジ・モジュールのスタイルでは、アクションは置換メソッドを使用して DDL コードを汎用にします。

たとえば、アクション行に、表のチェック制約を削除するための次のコードが含まれることがあります。

```
ALTER TABLE <%=odiRef.getTable("L", "TARG_NAME", "A") %> DROP CONSTRAINT
<%=odiRef.getCK("COND_NAME") %>
```

アクション・コール・メソッド

アクション・コール・メソッドは、アクション行でのみ使用できます。他の置換メソッドと異なり、テキストの生成には使用されず、コンテキストに適切なアクションを生成するために使用されます。

たとえば、Drop Table DDL 操作を実行するためには、まず表を参照している外部キーをすべて削除する必要があります。
表の削除アクションでは、最初のアクション行は `dropReferringFKs()` アクション・コール・メソッドを使用して、現在の表の各外部キーについて外部キーの削除アクションを自動的に生成します。このコールは、`<% odiRef.dropReferringFKs(); %>` というコードでアクション行を作成することによって実行されます。

アクション・コール・メソッドをコールするための構文は次のとおりです。

```
<% odiRef.method_name(); %>
```

注意: アクション・コール・メソッドはアクション行 1 行に 1 つのみである必要があります。=記号を前に付けずにコールしてください。また、最後にセミコロンが必要です。

アクションでは、次のアクション・コール・メソッドを使用できます。

- **addAKs()**: 現在の表のすべての代替キーに対して、代替キーの追加アクションをコールします。
- **dropAKs()**: 現在の表のすべての代替キーに対して、代替キーの削除アクションをコールします。

- **addPK()**: 現在の表の主キーに対して、主キーの追加アクションをコールします。
- **dropPK()**: 現在の表の主キーに対して、主キーの削除アクションをコールします。
- **createTable()**: 現在の表に対して表の作成アクションをコールします。
- **dropTable()**: 現在の表に対して表の削除アクションをコールします。
- **addFKs()**: 現在の表のすべての外部キーに対して、外部キーの追加アクションをコールします。
- **dropFKs()**: 現在の表のすべての外部キーに対して、外部キーの削除アクションをコールします。
- **enableFKs()**: 現在の表のすべての外部キーに対して、外部キーの有効化アクションをコールします。
- **disableFKs()**: 現在の表のすべての外部キーに対して、外部キーの無効化アクションをコールします。
- **addReferringFKs()**: 現在の表を参照しているすべての外部キーに対して、外部キーの追加アクションをコールします。
- **dropReferringFKs()**: 現在の表を参照しているすべての外部キーに対して、外部キーの削除アクションをコールします。
- **enableReferringFKs()**: 現在の表を参照しているすべての外部キーに対して、外部キーの有効化アクションをコールします。
- **disableReferringFKs()**: 現在の表を参照しているすべての外部キーに対して、外部キーの無効化アクションをコールします。
- **addChecks()**: 現在の表のすべてのチェック制約に対して、チェック制約の追加アクションをコールします。
- **dropChecks()**: 現在の表のすべてのチェック制約に対して、チェック制約の削除アクションをコールします。
- **addIndexes()**: 現在の表のすべての索引に対して、索引の追加アクションをコールします。
- **dropIndexes()**: 現在の表のすべての索引に対して、索引の削除アクションをコールします。
- **modifyTableComment()**: 現在の表に対して表のコメントの変更アクションをコールします。
- **AddColumnsComment()**: 現在の表のすべての列に対して列のコメントの変更アクションをコールします。

getAK()メソッド

使用方法

```
public java.lang.String getAK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの代替キーに関する情報を返します。現在のタスクに alternate key というタグが付けられている場合、チェック・ナレッジ・モジュールからのみアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている代替キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	AK 制約の内部番号。
KEY_NAME	代替キーの名前。
MESS	代替キーの制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成された AK のフルネーム。
<flexfield code>	この AK のフレックスフィールドの値。

例

表の代替キーの名前: **<%=odiRef.getAK("KEY_NAME")%>**

getAKColList()メソッド

使用方法

```
public java.lang.String getAKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getAKColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

現在チェックされている代替キーの列と式のリストを提供します。

リストの各要素について、pPattern パラメータが解釈され、繰り返されます。前の要素からは pSeparator パラメータによって区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の代替キーの各列の要素が含まれます。現在のタスクに alternate key というタグが付けられている場合、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された代替キーの列のリストを、キー内の位置の順に並べて返します。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン・シーケンス内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キー列の名前。
COL_HEADING	キー列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始位置（固定ファイル）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。

CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるグループ化記号。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	現在の列のフレックスフィールド値。

例

CUSTOMER 表に代替キーAK_CUSTOMER (CUST_ID, CUST_NAME)があり、作成するコードが次のようである場合:

```
create table T_AK_CUSTOMER (CUST_ID numeric(10) not null, CUST_NAME
varchar(50) not null)
```

次のように記述します。

```
create table T_<%=odiRef.getAK("KEY_NAME")%> <%=odiRef.getAKColList("(",
"[COL_NAME] [DEST_CRE_DT] not null", ", ", ")")%>
```

説明: getAKColList 関数は、(CUST_ID numeric(10) not null, CUST_NAME varchar(50) not null) の部分を生成するために使用されます。これはカッコで開始および停止し、代替キーの各列について、カンマで区切られたパターン（列、データ型、NOT NULL）を繰り返します。このため、

- 関数の最初のパラメータ"("は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ"[COL_NAME] [DEST_CRE_DT] not null"は、このパターンを代替キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[DEST_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ", "は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字")"は、文字列が、文字列「)」で終わることを示します。

getCK()メソッド

使用方法

```
public java.lang.String getCK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの条件に関する情報を返します。現在のタスクに condition というタグが付けられている場合のみ、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されているチェック制約に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている現在の文字列。

次の表は、pPropertyName で許容される様々な値のリストです。

パラメータの値。	説明
ID	チェック制約の内部番号。
COND_ALIAS	SQL 文に使用する表の別名。
COND_NAME	条件の名前。
COND_TYPE	条件のタイプ。
COND_SQL	条件の SQL 文。
MESS	チェック制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成されたチェック制約のフルネーム。

COND_SQL_DDL	表の別名がない条件の SQL 文。
<flexfield code>	このチェック制約のフレックスフィールド値。

例

現在の条件のコール: **<%=snpRep.getCK("COND_NAME")%>**

```
insert into MY_ERROR_TABLE
select *
from MY_CHECKED_TABLE
where (not (<%=odiRef.getCK("COND_SQL")%>))
```

getColList()メソッド

使用方法

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd,
java.lang.String pSelector)
```

代替可能な構文:

```
public java.lang.String getColList(
java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pSelector)
public java.lang.String getColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

列と式のリストを提供します。列リストはこのメソッドがコールされたフェーズにより異なります。

pPattern パラメータはリストの各要素（pSelector パラメータに応じて選択）ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

ロード（LKM）

現在のソース環境で実行されたすべてのマッピング式と、ステージング領域で実行されたマッピング、フィルタ式、結合で使用された列。

インタフェースで execute というタグを付けられたマッピングのみが対象です。

- リストは POS、FILE_POS でソートされます。

インタフェースのソースにジャーナル化されたデータストアがある場合、3 つのジャーナル化擬似列 `JRN_FLG`、`JRN_DATE` および `JRN_SUBSCRIBER` は、ジャーナル化されたソース・データストアの列として追加されます。

統合（IKM）

現在のインタフェースで execute というタグを付けられたすべての現在のマッピング式。

リストには、現在のインタフェースのターゲット表に（execute というタグを付けて）ロードされる各列につき、1 つの要素が含まれます。

- ロードされた表が一時表でない場合、リストは POS、FILE_POS でソートされます。
- ロードされた表が一時表である（参照にない）場合、リストはソートされません。

インタフェースのソースにジャーナル化されたデータストアがあり、それがステージング領域にある場合、3 つのジャーナル化擬似列 `JRN_FLG`、`JRN_DATE` および `JRN_SUBSCRIBER` は、ジャーナル化されたソース・データストアの列として追加されます。

チェック（CKM）

ターゲット表のすべての列（静的管理またはフロー制御）

ターゲット表の列を、現在のインタフェースで記入された列から区別するには、MAP セレクタを使用する必要があります。

- リストはターゲット表の POS、FILE_POS でソートされます。

アクション

DDL コマンドが処理した表のすべての列。

変更されたか、追加したか、削除された列の場合、NEW および OLD セレクタを使用して、DDL コマンドによって処理される、変更された列の新バージョンまたは旧バージョンを取得することができます。

- リストは表の POS、FILE_POS でソートされます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。

pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。
pSelector	文字列	次の形式を使用して、最初のリストの要素をフィルタできるブール式を指定する文字列。 <SELECTOR> <演算子> <SELECTOR>など。カッコを使用できます。 使用できる演算子: 1. 否定: NOT または! 2. 論理和: OR または|| 3. 論理積: AND または&& 例: (INS AND UPD) OR TRG 有効なセレクタについては、表「セレクタの説明」で説明します。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。

COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。
CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
JDBC_TYPE	データ・サービス: ドライバによって返される列の JDBC タイプ。

<flexfield code>	現在の列のフレックスフィールド値。

セレクタの説明

パラメータ値	説明
INS	• LKM: 適用対象外（*）。 • IKM: 挿入でマークされたマッピング式のみ。 • CKM: 適用対象外。
UPD	• LKM: 適用対象外（*）。 • IKM: 更新でマークされたマッピング式のみ。 • CKM: 適用対象外。
TRG	• LKM: 適用対象外（*）。 • IKM: ターゲット上で実行されたマッピング式のみ。 • CKM: ターゲット上で実行されたマッピング式。
NULL	• LKM: 適用対象外（*）。 • IKM: NULL 値可能でない列をロードするすべてのマッピング式。 • CKM: NULL 値を許容しないすべてのターゲット列。
PK	• LKM: 適用対象外（*）。 • IKM: 主キー列をロードするすべてのマッピング式。 • CKM: 主キーの一部であるすべてのターゲット列。
UK	• LKM: 適用対象外（*）。 • IKM: 現在のインタフェースに対して選択された更新キー列をロードするすべてのマッピング式。 • CKM: 適用対象外。
REW	• LKM: 適用対象外（*）。 • IKM: 読取り専用フラグが設定されている列をロードするすべてのマッピング式。 • CKM: 読出し専用フラグが選択されていないすべてのターゲット列。
UD1	• LKM: 適用対象外（*）。 • IKM: UD1 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD2	• LKM: 適用対象外（*）。 • IKM: UD2 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。

UD3	• LKM: 適用対象外（*）。 • IKM: UD3 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD4	• LKM: 適用対象外（*）。 • IKM: UD4 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
UD5	• LKM: 適用対象外（*）。 • IKM: UD5 としてマークされた列をロードするすべてのマッピング式。 • CKM: 適用対象外。
MAP	• LKM: 適用対象外。 • IKM: 適用対象外。 • CKM: フロー制御: 現在のインタフェースに式を使用してロードされたターゲット表のすべての列。 静的管理: ターゲット表のすべての列。
SCD_SK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: サロゲート・キー」とマーキングされたすべての列。
SCD_NK	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 自然キー」とマーキングされたすべての列。
SCD_UPD	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時に上書き」とマーキングされたすべての列。
SCD_INS	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 変更時の行の追加」とマーキングされたすべての列。
SCD_FLAG	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 現在のレコード・フラグ」とマーキングされたすべての列。
SCD_START	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 開始タイムスタンプ」とマーキングされたすべての列。
SCD_END	• LKM、CKM、IKM: データ・モデル定義で「SCD の動作: 終了タイムスタンプ」とマーキングされたすべての列。
NEW	• アクション: 表に追加された列。表の変更された列の新しいバージョン。
OLD	• アクション: 表から削除された列。表の変更された列の旧バージョン。
WS_INS	• SKM: 列は、データ・サービスを使用した INSERT を許可するフラグを設定されています。
WS_UPD	• SKM: 列は、データ・サービスを使用した UDATE を許可するフラグを設

	定されています。
WS_SEL	• SKM: 列は、データ・サービスを使用した SELECT を許可するフラグを設定されています。

（*）重要な注意: LKM で、*で示した一部のセレクタを使用することは、可能ですが非推奨です。インタフェースでソース上にマップされた列のみが返されます。**結果として、インタフェースにより、結果が正しくないことがあります。**
たとえば、UK セレクタでは、マップされないか、ソース上で実行されないキーの列は、このセレクタで返されません。

例

CUSTOMER 表に列 (CUST_ID, CUST_NAME, AGE)が含まれ、作成するコードが次のようである場合:

```
create table CUSTOMER (CUST_ID Numeric(10) null, CUST_NAME VARCHAR(50)
null, AGE Numeric(3) null)
```

次のように記述します。

```
create table CUSTOMER <%=odiRef.getColList("(", "[COL_NAME]
[SOURCE_CRE_DT] null", ", ", ")", "")%>
```

説明: getColList 関数が使用され、(CUST_ID numeric(10) null, CUST_NAME varchar(50) null, AGE numeric(3) null)が生成されます。先頭と末尾はカッコで、(column, data type, null)というパターンが、各列につきカンマで区切られて繰り返されます。このため、

- 関数の最初の文字**"("**は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ**"[COL_NAME] [SOURCE_CRE_DT] null"**は、このパターンを各列に対して繰り返すことを示します。キーワード[COL_NAME]および[SOURCE_CRE_DT]は、表「パターン属性リスト」の有効なキーワードに対する参照です。
- 3 番目のパラメータ**", "**は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字**")"**は、文字列が、文字列「)」で終わることを示します。
- 最後のパラメータ**""**は、パターンを各列に対して（選択内容なしで）繰り返すことを示します。

getColumn()メソッド

使用方法

```
public java.lang.String getColumn(
java.lang.String pPattern,
java.lang.String pSelector)
public java.lang.String getColumn(
java.lang.String pPattern)
```

説明

アクションでは、アクションにより処理されている列に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPattern	文字列	列に合せて整理された値のパターン。 パターンで使用できる属性のリストを、表「パターン属性リスト」に示します。 属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSelector	文字列	セレクタは次のいずれかの値をとれます。 • NEW: 変更された列の新しいバージョンまたは新しい列を返します。 • OLD: 変更された列の古いバージョンまたは削除された列を返します。 セレクタを省略すると、すべての削除アクションに対して OLD に設定されます。それ以外の場合は NEW に設定されます。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	列の名前。
COL_HEADING	列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。

CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースなしに設定、2= 非アクティブ・トレースなしに設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	ソース上で実行される式のテキスト（マッピングに入力された式、またはステージング領域上で実行された式を作成する列前）。
CX_COL_NAME	ステージング領域の上の現在の式のコンテナとして使用されている列の計算された名前。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
MANDATORY_CLAUSE	列が必須の場合、NOT NULL を返します。そうでない場合、そのテクノロジの NULL キーワードを返します。
DEFAULT_CLAUSE	デフォルト値が存在する場合、DEFAULT <デフォルト値>を返します。それ以外の場合、空の列を返します。
COL_DESC	列の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
<flexfield code>	現在の列のフレックスフィールド値。

getFK()メソッド

使用方法

```
public java.lang.String getFK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの外部キー（または結合、参照）に関する情報を返します。現在のタスクに reference というタグが付けられている場合のみ、ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている外部キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	参照制約の内部番号。
FK_NAME	参照制約の名前。
FK_TYPE	参照制約のタイプ。
FK_ALIAS	参照表の別名（複合式の場合のみ使用）。
PK_ALIAS	参照された表の別名（複合式の場合のみ使用）。
ID_TABLE_PK	参照された表の内部番号。
PK_I_MOD	参照されたモデルの番号。
PK_CATALOG	参照された表のカタログ。
PK_SCHEMA	参照された表の物理スキーマ。
PK_TABLE_NAME	参照された表の名前。
COMPLEX_SQL	結合句の複合 SQL 文（該当する場合）。
MESS	参照制約のエラー・メッセージ。
FULL_NAME	ローカル・オブジェクトマスクで生成された外部キーのフルネーム。
<flexfield code>	この参照のフレックスフィールド値。

例

表の現在の参照キーの名前は**<%=odiRef.getFK("FK_NAME")%>**です。これは、表**<%=odiRef.getFK("PK_TABLE_NAME")%>** を参照しており、この表はスキーマ**<%=odiRef.getFK("PK_SCHEMA")%>**にあります。

getFKColList()メソッド

使用方法

```
public java.lang.String getFKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

代替可能な構文:

```
public java.lang.String getFKColList(
java.lang.String pPattern,
java.lang.String pSeparator)
```

説明

参照制約（外部キー）の列部分のリストを提供します。

pPattern パラメータはリストの各要素ごとに解釈され、繰り返されて、前の要素からパラメータ pSeparator で区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の外部キーの各列の要素が 1 つ含まれます。現在のタスクに reference というタグが付けられている場合のみ、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された外部キーの列のリストを、キー内の位置の順に並べて返します。

代替可能な構文では、未設定のパラメータはすべて空の文字列として設定されます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このパラメータは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン文字列内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このパラメータは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キーの列の名前。
COL_HEADING	キーの列のヘッダー。
COL_DESC	キーの列の説明。
POS	キーの列の位置。
LONGC	キーの列の長さ（精度）。
SCALE	キーの列のスケール。
FILE_POS	列の開始（索引）。
BYTES	列の物理オクテット数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列のレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるセパレータ。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。

SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
PK_I_COL	参照列の内部識別子。
PK_COL_NAME	参照キー列の名前。
PK_COL_HEADING	参照キー列のヘッダー。
PK_COL_DESC	参照キー列の説明。
PK_POS	参照列の位置。
PK_LONGC	参照列の長さ。
PK_SCALE	参照列の精度。
PK_FILE_POS	参照列の開始（索引）。
PK_BYTES	参照列の物理オクテット数。
PK_FILE_END_POS	参照列の終了（FILE_POS+BYTES）。
PK_IND_WRITE	参照列の書込み権限フラグ。
PK_COL_MANDATORY	参照列の必須文字（0: NULL を許可、1: NOT NULL）
PK_CHECK_FLOW	参照された列のフロー制御フラグ（0: チェックしない、1: チェック）。
PK_CHECK_STAT	参照された列の静的管理フラグ（0: チェックしない、1: チェック）。
PK_COL_FORMAT	参照された列の論理形式。
PK_COL_DEC_SEP	参照された列の小数点記号。
PK_REC_CODE_LIST	参照された列に維持されたレコード・コードのリスト。
PK_COL_NULL_IF_ERR	参照された列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
PK_DEF_VALUE	参照された列のデフォルト値。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。

<flexfield code>	参照している表の現在の列のフレックスフィールド値。

例

CUSTOMER.COUNTRY_ID = CITY.ID_COUNT and CUSTOMER.CITY_ID = CITY.ID_CIT で CUSTOMER 表が CITY 表を参照している場合、

句:

```
(CUS.COUNTRY_ID = CITY.ID_COUNT and CUS.CITY_ID = CITY.ID_CIT)
```

は、次のようにも記述できます。

```
<%=odiRef.getFKColList("(", "CUS.[COL_NAME] = CITY.[PK_COL_NAME]", " and
", ")" )%>
```

説明:getFKColList 関数は、外部キーの各列上でループして、カッコで始まってカッコで終わり、**and** で区切られたパターンを外部キーの各列に対して繰り返す句を生成するために使用されます。このため、

- 関数の最初のパラメータ**"("**は、文字列を、「(」で始めることを示します。
- 2 番目のパラメータ"CUS.[COL_NAME] = CITY.[PK_COL_NAME]" は、このパターンを外部キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[PK_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ" and "は、パターンの発生を、文字列 and で区切ることを示します。
- 関数の 4 番目の文字")"は、文字列が、「)」で終わることを示します。

getIndex()メソッド

使用方法

```
public java.lang.String getIndex(java.lang.String pPropertyName)
```

説明

アクションでは、このメソッドは、DDL コマンドによって現在処理されている索引に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	索引の内部番号。

KEY_NAME	索引の名前。
FULL_NAME	ローカル・オブジェクト・マスクで生成された索引のフルネーム。
<flexfield code>	この索引のフレックスフィールドの値。

getIndexColList()メソッド

使用方法

```
public java.lang.String getIndexColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

説明

アクションでは、このメソッドは、DDL コマンドによって処理された索引の列のリストを、索引内の位置の順に並べて返します。

リストの各要素について、pPattern パラメータが解釈され、繰り返されます。前の要素からは pSeparator パラメータによって区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の索引の各列の要素が含まれます。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン・シーケンス内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	索引列の名前。
COL_HEADING	索引列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始位置（固定ファイル）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。
CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるグループ化記号。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコー

	ド。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	現在の列のフレックスフィールド値。

getNewColComment()メソッド

使用方法

```
public java.lang.String getNewColComment()
```

説明

アクションでは、このメソッドは、列のコメントの変更アクションで、DDL コマンドによって処理されている列の新しいコメントを返します。

getNewTableComment()メソッド

使用方法

```
public java.lang.String getNewTableComment()
```

説明

アクションでは、このメソッドは、表のコメントの変更アクションで、DDL コマンドによって処理されている表の新しいコメントを返します。

getPK()メソッド

使用方法

```
public java.lang.String getPK(java.lang.String pPropertyName)
```

説明

このメソッドはチェック・プロシージャの間、データストアの主キーに関する情報を返します。

アクションでは、このメソッドは、DDL コマンドによって現在処理されている主キーに関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
ID	PK 制約の内部番号。
KEY_NAME	主キーの名前。
MESS	主キー制約に関するエラー・メッセージ。
FULL_NAME	ローカル・オブジェクト・マスクで生成された PK のフルネーム。
<flexfield code>	主キーのフレックスフィールド値。

例

表の主キーの名前: **<%=odiRef.getPK("KEY_NAME")%>**

getPKColList()メソッド

使用方法

```
public java.lang.String getPKColList( java.lang.String pStart,
java.lang.String pPattern,
java.lang.String pSeparator,
java.lang.String pEnd)
```

説明

チェックされている主キーの列と式のリストを提供します。

リストの各要素について、pPattern パラメータが解釈され、繰り返されます。前の要素からは pSeparator パラメータによって区切られます。生成された文字列は pStart から始まり、pEnd で終わります。

このリストには、現在の主キーの各列の要素が含まれます。現在のタスクに primary key というタグが付けられている場合、チェック・ナレッジ・モジュールからアクセスできます。

アクションでは、このメソッドは、DDL コマンドによって処理された主キーの列のリストを、キー内の位置の順に並べて返します。

パラメータ

パラメータ	タイプ	説明
pStart	文字列	このシーケンスは、生成する文字列の始まりの目印です。
pPattern	文字列	パターンはリスト内に現れるたびに繰り返されます。 パターンで使用できる属性のリストは、表「パターン属性リスト」を参照してください。 各属性は、パターン・シーケンス内に出現するたびにその値に置き換えられます。属性は大カッコ（[ ]）で囲む必要があります。 例: My string [COL_NAME] is a column
pSeparator	文字列	このパラメータは、各パターンを前のパターンから区切ります。
pEnd	文字列	このシーケンスは、生成する文字列の終わりの目印です。

パターン属性リスト

次の表は、異なるパラメータ値を、それに関連付けられた説明とともにあげたものです。

パラメータ値	説明
I_COL	列の内部識別子。
COL_NAME	キー列の名前。
COL_HEADING	キー列のヘッダー。
COL_DESC	列の説明。
POS	列の位置。
LONGC	列の長さ（精度）。
SCALE	列のスケール。
FILE_POS	列の開始位置（固定ファイル）。
BYTES	列の物理バイト数。
FILE_END_POS	列の終了（FILE_POS+BYTES）。
IND_WRITE	列の書込み権限フラグ。
COL_MANDATORY	列の必須文字（0: NULL を許可、1: NOT NULL）
CHECK_FLOW	列のフロー制御フラグ（0: チェックしない、1: チェック）。

CHECK_STAT	列の静的管理フラグ（0: チェックしない、1: チェック）。
COL_FORMAT	列の論理形式。
COL_DEC_SEP	列の小数点記号。
REC_CODE_LIST	列に維持されたレコード・コードのリスト。
COL_NULL_IF_ERR	列の処理フラグ（0 = 拒否、1 = アクティブ・トレースを NULL に設定、2= 非アクティブ・トレースを NULL に設定）。
DEF_VALUE	列のデフォルト値。
EXPRESSION	未使用。
CX_COL_NAME	未使用。
ALIAS_SEP	別名に対して（テクノロジにより）使用されるグループ化記号。
SOURCE_DT	列のデータ型のコード。
SOURCE_CRE_DT	列のデータ型に対する表作成構文。
SOURCE_WRI_DT	列の書込み可能データ型に対する表作成構文。
DEST_DT	ターゲット・テクノロジ上のデータ型に変換された、列のデータ型のコード。
DEST_CRE_DT	ターゲット・テクノロジ上のデータ型に変換された列のデータ型に対する表作成構文。
DEST_WRI_DT	ターゲット・テクノロジ上のデータ型に変換された、列の書込み可能なデータ型に対する表作成構文。
SCD_COL_TYPE	データ・モデルでこの列の緩やかに変化するディメンションに対して定義された動作。
<flexfield code>	現在の列のフレックスフィールド値。

例

CUSTOMER 表に主キーPK_CUSTOMER (CUST_ID, CUST_NAME)があり、作成するコードが次のようである場合:

```
create table T_PK_CUSTOMER (CUST_ID numeric(10) not null, CUST_NAME
varchar(50) not null)
```

次のように記述します。

```
create table T_<%=odiRef.getPK("KEY_NAME")%> <%=odiRef.getPKColList("(",
"[COL_NAME] [DEST_CRE_DT] not null", ", ", ")")%>
```

説明: getPKColList 関数は、(CUST_ID numeric(10) not null, CUST_NAME varchar(50) not null) の部分を生成するために使用されます。これはカッコで開始および停止し、代替キーの各列について、カンマで区切られたパターン（列、データ型、NOT NULL）を繰り返します。このため、

- 関数の最初のパラメータ″(″は、文字列を、文字列「(」で始めることを示します。
- 2 番目のパラメータ″ [COL_NAME] [DEST_CRE_DT] not null″は、このパターンを主キーの各列に対して繰り返すことを示します。キーワード[COL_NAME]および[DEST_CRE_DT]は、表「パターン属性リスト」の有効なキーワードを参照しています。
- 3 番目のパラメータ″, ″は、パターンの解釈された発生を、文字列「,」で区切ることを示します。
- 関数の 4 番目の文字")"は、文字列が、文字列「)」で終わることを示します。

getTable()メソッド

使用方法

```
public java.lang.String getTable(
java.lang.String pMode,
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty,
java.lang.String pLocation)
public java.lang.String getTable(
java.lang.String pProperty)
```

説明

odi によって処理された一時的表および永続的表のフルネームを取得します。

パラメータ

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>pMode</td><td>文字列</td><td>"L"はローカル・オブジェクト・マスクを使用してオブジェクトの完全パスを作成します。この値は、pMode が指定されていない場合に使用されます。
"R"はオブジェクト・マスクを使用してオブジェクトの完全パスを作成します。
"A"は自動。使用する適切なマスクを自動的に定義します。</td></tr>
<tr><td>pProperty</td><td>文字列</td><td>構築される表の名前を示すパラメータ。可能な値のリストを次に示します。
<table>
<tr><th>パラメータ値</th><th>説明</th></tr>
<tr><td>ID</td><td>データストア識別子。</td></tr>
</table>
</td></tr>
</table>

TARG_NAME	ターゲット・データストアのフルネーム。アクションでは、このパラメータは、DDL コマンドによって処理された現在の表の名前を返します。
COLL_NAME	ロード・データストアのフルネーム。
INT_NAME	統合データストアのフルネーム。
ERR_NAME	エラー・データストアのフルネーム。
CHECK_NAME	エラー・サマリー・データストアの名前。
CT_NAME	チェック・データストアのフルネーム。
FK_PK_TABLE_NAME	外部キーによって参照されたデータストアのフルネーム。
JRN_NAME	ジャーナル化されたデータストアのフルネーム。
JRN_VIEW	ジャーナル化されたデータストアにリンクされたビューのフルネーム。
JRN_DATA_VIEW	ジャーナル化されたデータストアにリンクされたデータ・ビューのフルネーム。
JRN_TRIGGER	ジャーナル化されたデータストアにリンクされたトリガーのフルネーム。
JRN_ITRIGGER	ジャーナル化されたデータストアにリンクされた挿入トリガーのフルネーム。
JRN_UTRIGGER	ジャーナル化されたデータストアにリンクされた更新トリガーのフルネーム。
JRN_DTRIGGER	ジャーナル化されたデータストアにリンクされた削除トリガーのフルネーム。
SUBSCRIBER_TABLE	サブスクライバ・リストが含まれているデータストアのフルネーム。
CDC_SET_TABLE	CDC セットのリストが含まれている表のフルネーム。
CDC_TABLE_TABLE	CDC セットを使用してジャーナル化された表のリストが含まれている表のフルネーム。

		CDC_SUBS_TABLE	CDC セットのサブスクライバのリストが含まれている表のフルネーム。
		CDC_OBJECTS_TABLE	ジャーナル化パラメータとオブジェクトが含まれている表のフルネーム。
		<flexfield code>	現在のターゲット表のフレックスフィールド値。
pLocation	文字列	"W"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理作業スキーマのフルネームを返します。
		"D"	物理カタログのオブジェクトと、現在のタプル（コンテキスト、論理スキーマ）に対応する物理データ・スキーマのフルネームを返します。
		"A"	odi がオブジェクトのデフォルトの場所を決定します。この値は、pLocation が指定されていない場合に使用されます。

例

定義されている要素:

物理スキーマ: Pluton.db_odi.dbo

データ・カタログ:	db_odi
データ・スキーマ:	dbo
作業カタログ:	tempdb
作業スキーマ:	temp_owner
ローカル・マスク:	%CATALOG.%SCHEMA.%OBJECT
リモート・マスク:	%DSERVER:%CATALOG.%SCHEMA.%OBJECT
ロード接頭辞:	CZ_
エラー接頭辞:	ERR_
統合接頭辞:	I$_

この物理スキーマを関連付けている論理スキーマ: MSSQL_ODI（コンテキストは CTX_DEV）

表の名前: CUSTOMER

コール対象	戻り値
`<%=odiRef.getTable("L", "COLL_NAME",`	tempdb.temp_owner.CZ_0CUSTOMER

"W")%> <%=odiRef.getTable("R", "COLL_NAME", "D")%>	MyServer:db_odi.dbo.CZ_0CUSTOMER
<%=odiRef.getTable("L", "INT_NAME", "W")%>	tempdb.temp_owner.I$_CUSTOMER
<%=odiRef.getTable("R", "ERR_NAME", "D")%>	MyServer:db_odi.dbo.ERR_CUSTOMER

getTargetTable()メソッド

使用方法

```
public java.lang.String getTargetTable(java.lang.String pPropertyName)
```

説明

現在のターゲット表の概要を返す汎用メソッド。使用可能なデータのリストを **pPropertyName の値**の表に示します。

アクションでは、このメソッドは DDL コマンドによって処理されている表に関する情報を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	リクエストされたプロパティの名前が含まれている文字列。

pPropertyName の値

次の表は、pPropertyName で可能な値のリストです。

パラメータ値	説明
I_TABLE	データストアの内部識別子。
MODEL_NAME	現在のデータストアのモデルの名前。
SUB_MODEL_NAME	現在のデータストアのサブモデルの名前。
TECHNO_NAME	ターゲット・テクノロジの名前。
LSCHEMA_NAME	ターゲット論理スキーマの名前。
TABLE_NAME	ターゲット・データストアの名前。
RES_NAME	ターゲット・リソースの物理名。
CATALOG	カタログ名。

WORK_CATALOG	作業カタログの名前。
SCHEMA	スキーマ名。
WORK_SCHEMA	作業スキーマの名前。
TABLE_ALIAS	現在のデータストアの別名。
TABLE_TYPE	データストアのタイプ。
DESCRIPTION	現在のインタフェースの説明。
TABLE_DESC	現在のインタフェースのターゲット・データストアの説明。DDL コマンドの場合、現在の表の説明。
R_COUNT	現在のデータストアの行数。
FILE_FORMAT	現在のデータストア（ファイル）の形式。
FILE_SEP_FIELD	フィールド・セパレータ（ファイル）。
XFILE_SEP_FIELD	16 進表記のフィールド・セパレータ（ファイル）
SFILE_SEP_FIELD	フィールド・セパレータ文字列（ファイル）。
FILE_ENC_FIELD	フィールドの開始および終了文字（ファイル）。
FILE_SEP_ROW	レコード・セパレータ（ファイル）。
XFILE_SEP_ROW	16 進表記のレコード・セパレータ（ファイル）。
SFILE_SEP_ROW	レコード・セパレータ文字列（ファイル）。
FILE_FIRST_ROW	ファイル（ファイル）の先頭の、無視する行数。
FILE_DEC_SEP	小数点記号（ファイル）。
METADATA_DESC	データストアのメタデータの説明（ファイル）
OLAP_TYPE	データストア定義で指定された OLAP タイプ。
IND_JRN	データストアが CDC に含まれることを示すフラグ。
JRN_ORDER	一貫したジャーナル化のための CDC セット内でのデータストアの順序。
TABLE_DESC	表の説明（コメント）。引用符と二重引用符はスペースに置き換えられます。
WS_NAME	データ・サービス: このデータストアのモデル用に生成された Web サービスの名前。
WS_NAMESPACE	データ・サービス: Web サービスの XML ネームスペース。

WS_JAVA_PACKAGE	データ・サービス: Web サービス用に生成された Java パッケージ。
WS_ENTITY_NAME	データ・サービス: Web サービスでこのデータストアに対して使用されるエンティティ名。
WS_DATA_SOURCE	データ・サービス: このデータストアの Web サービスに対して指定されたデータソース。
<flexfield code>	現在の表のフレックスフィールド値。

例

現在の表: **<%=odiRef.getTargetTable("RES_NAME")%>**

isColAttrChanged()メソッド

使用方法

```
public java.lang.Boolean
isColAttrChanged(java.lang.String pPropertyName)
```

説明

このメソッドは、列属性またはコメントを変更するための列アクションで使用できます。パラメータとして渡された列属性が変更されたかどうかを示すブール値を返します。

パラメータ

パラメータ	タイプ	説明
pPropertyName	文字列	属性コード（次を参照）。

次の表は、pPropertyName で可能な様々な値のリストです。

パラメータ値	説明
DATATYPE	列のデータ型、長さまたは精度が変更されています。
LENGTH	列の長さが変更されています（たとえば、VARCHAR(10)がVARCHAR(12)に変更されています）。
PRECISION	列の精度が変更されています（たとえば、DECIMAL(10,3)がDECIMAL(10,4)に変更されています）。
COMMENT	列コメントが変更されています。
NULL_TO_NOTNULL	列の NULL 値可能属性が、NULL から NOT NULL に変更されています。

NOTNULL_TO_NULL	列の NULL 値可能属性が、NOT NULL から NULL に変更されています。
NULL	列の NULL 値可能属性が変更されています。
DEFAULT	列のデフォルト値が変更されています。

例

```
<% if (odiRef.IsColAttrChanged("DEFAULT") ) { %>
     /* Column default attribute haschanged. */
<% } %>
```