WebSphere® Adapters

バージョン 7 リリース 0 Feature Pack 1

WebSphere Adapter for JDBC ユーザーズ・ガイド

バージョン 7 リリース 0

Feature Pack 1

WebSphere® Adapters

バージョン 7 リリース 0 Feature Pack 1

WebSphere Adapter for JDBC ユーザーズ・ガイド

バージョン 7 リリース 0

Feature Pack 1

IBM

お願い

本書および本書で紹介する製品をご使用になる前に、 149 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM WebSphere Adapter for JDBC バージョン 7 リリース 0 モディフィケーション 1 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： WebSphere® Adapters version 7 release 0 Feature Pack 1
WebSphere Adapter for JDBC User Guide
Version 7 Release 0 Feature Pack 1

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2010.6

目次

第 1 章 このリリースの新機能 1

第 2 章 Oracle テーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクトのユーザー定義型 3
ビジネス・オブジェクト 3
Create 操作 16
Update 操作 17
Outbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成 19
クエリー・ビジネス・オブジェクトの選択および構成 28
ビジネス・オブジェクト属性 33

第 3 章 テーブル・ビジネス・オブジェクト内の XML データ型 37
外部ソフトウェア依存関係の追加 37
ビジネス・オブジェクト属性 39
属性に関するアプリケーション固有情報 43
一般的な問題の解決策 52

第 4 章 Date、Time、および Timestamp の各データ型の形式のカスタマイズ 79
ビジネス・オブジェクト 79
Outbound 処理のデータベース・オブジェクトのディスカバー 92
Inbound 処理のデータベース・オブジェクトのディスカバー 96
属性に関するアプリケーション固有情報 100

第 5 章 テーブル間の親子関係の自動ディスカバリー 109
ビジネス・オブジェクト. 109
Outbound 処理のデータベース・オブジェクトのディスカバー 122
Inbound 処理のデータベース・オブジェクトのディスカバー 126
Outbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成 130
Inbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成 139

特記事項. 149
プログラミング・インターフェース情報 151
商標 151

第 1 章 このリリースの新機能

このバージョンには、ビジネスの柔軟性、ユーザー・エクスペリエンス、およびアダプターの性能を強化するためのいくつかの新機能が含まれています。

その他のサポート対象機能についての詳細な情報は、WebSphere® Adapter for JDBC インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/topic/com.ibm.wsadapters.jca.jdbc.doc/doc/stbp_jdb_welcome.html) で参照できます。このサイトは、定期的に最新の情報に更新されています。

WebSphere Adapter for JDBC でサポートされる新機能は、以下のとおりです。

- Oracle テーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクトのユーザー定義型
- テーブル・ビジネス・オブジェクト内の XML データ型
- Date、Time、および Timestamp の各データ型の形式のカスタマイズ
- テーブル間の親子関係の自動ディスカバリー

注: WebSphere Integration Developer では、1 つのバージョンのアダプターのみをワークスペースにインポートしてください。バージョン 7.0.0.1 のフィックスパックまたはバージョン 7.0.1.0 のフィーチャー・パックの、いずれかのアダプターが使用できます。

ランタイム環境のアプリケーション (EAR) には、1 つのバージョンの組み込み RAR ファイル (バージョン 7.0.0.1 のフィックスパックまたはバージョン 7.0.1.0 のフィーチャー・パックの、いずれかのアダプター) のみを含めるようにしてください。ノード・レベルがデプロイされたアダプターでも、1 つのバージョンのアダプターのみを組み込んでください。

第 2 章 Oracle テーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクトのユーザー定義型

このアダプターは、Oracle データベースのテーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクト内で配列、テーブル、構造体、ネストされた構造体などのユーザー定義型または複合データ型をサポートします。これらの型に対しては、型名および子属性の詳細が自動的にディスカバーされて、表示されます。

ユーザー定義タイプとして列が定義されている場合、外部サービス・ウィザードは、自動的にこれらの列をディスカバーし、ユーザー定義タイプの子ビジネス・オブジェクトを生成します。アダプターが、入力ビジネス・オブジェクト・インスタンスをユーザー定義タイプ・インスタンスへ変換し、ユーザー定義タイプ・インスタンスを使用して Outbound 処理の対応する操作を実行します。

ビジネス・オブジェクト

ビジネス・オブジェクトとは、データ、データ上で実行されるアクション、およびデータを処理するための追加の指示 (存在する場合) で構成される構造体のことです。WebSphere Adapter for JDBC は、ビジネス・オブジェクトを使用して、データベースのテーブルとビュー、データベース照会、ストアード・プロシージャー、およびストアード関数の結果を表現します。ビジネス・オブジェクトにより、データベースのオブジェクトの階層を作成し、無関係なテーブルをグループ化できます。コンポーネントはビジネス・オブジェクトを使用してアダプターと通信します。

アダプターによるビジネス・オブジェクトの使用法

統合アプリケーションは、ビジネス・オブジェクトを使用してデータベースにアクセスします。アダプターは、Outbound 要求のビジネス・オブジェクトを、データベースへアクセスするための JDBC API 呼び出しに変換します。Inbound イベントの場合、アダプターはイベントのデータをビジネス・オブジェクトに変換し、このビジネス・オブジェクトがアプリケーションに戻されます。

アダプターは、ビジネス・オブジェクトを使用してデータベース内の次のタイプのオブジェクトを表現します。

- テーブルとビュー
- シノニムとニックネーム
- ストアード・プロシージャーとストアード関数

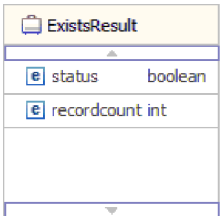
一部のビジネス・オブジェクトは、データベース・オブジェクトを表現しません。これらのビジネス・オブジェクトには、以下のものがあります。

- バッチ SQL ビジネス・オブジェクト。一連のユーザー定義の insert ステートメント、update ステートメント、および delete ステートメントを表します。
- クエリー・ビジネス・オブジェクト。データベースに対して実行されるユーザー定義 SQL 照会を表します。

- ラッパー・ビジネス・オブジェクト。このオブジェクトにより、無関係なテーブル・オブジェクトとビュー・オブジェクトを単一のビジネス・オブジェクトにグループ化でき、複数のストアード・プロシージャーを単一のビジネス・オブジェクトにグループ化することができます。

アダプターは、出力に一部のビジネス・オブジェクトを使用します。これらのビジネス・オブジェクトには、以下のものがあります。

- コンテナー・ビジネス・オブジェクト。RetrieveAll 操作からの出力が入ります。
- ExistsResult ビジネス・オブジェクト。これには、Exists 操作からの出力が入ります。



ビジネス・オブジェクト内でのデータの表現方法

テーブルまたはビュー・ビジネス・オブジェクトの場合

テーブルまたはビューの各列は、テーブル・ビジネス・オブジェクトまたはビュー・ビジネス・オブジェクトの単純属性により表現されます。*単純属性* とは、String、Integer、または Date などの単一値を表す属性です。その他の属性は、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表します。

同じビジネス・オブジェクトに含まれる単純属性を、別々のデータベース表に格納することはできませんが、次の状況が考えられます。

- データベース表に、対応するビジネス・オブジェクトに含まれる単純属性の数よりも多くの列が含まれる場合があります。つまり、データベースの列の一部が、ビジネス・オブジェクト内に表されていません。アプリケーションによるビジネス・オブジェクトの処理で必要な列のみを実際の設計に含める必要があります。
- ビジネス・オブジェクトに、対応するデータベース表に含まれる列の数よりも多くの単純属性が含まれる場合があります。つまり、ビジネス・オブジェクト内の属性の一部が、データベース内に表されていません。データベース内に列を持たない属性は、アプリケーション固有情報を持っていないか、デフォルト値が設定されているか、またはストアード・プロシージャーかストアード関数のパラメーターです。
- ビジネス・オブジェクトは、複数のデータベース表にまたがるビューを表すことができます。アダプターでは、Create、Update、および Delete 操作など、データベースに対する変更によって起動されるイベントを処理するときに、このようなビジネス・オブジェクトを使用できます。ただし、ビジネス・オブジェクトの要求を処理する場合には、Retrieve および RetrieveAll 要求に対してのみ、このようなビジネス・オブジェクトを使用できます。

テーブル・ビジネス・オブジェクトには、対応するデータベース表に基本キーがない場合でも、常に基本キーが設定されています。アダプターは、テーブル・ビジネ

ス・オブジェクトを取得するときに、基本キー属性で指定される列を使用します。データベースで外部キー参照が定義されている場合、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。例えば、親テーブルである CUSTOMER テーブルと子テーブルである ADDRESS テーブルがあるとします。データベース内で ADDRESS から CUSTOMER への外部キー参照を定義した場合、アダプターは自動的に親子関係をディスカバーし、外部キー参照を「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに表示します。外部キー参照が CUSTOMER から ADDRESS への場合、アダプターは自動的に「**単一カーディナリティー**」チェック・ボックスを選択し、外部キー参照を表示します。1 つのテーブルに複数の外部キー参照が定義されている場合、アダプターは外部キー関係を 1 つだけ生成します。

アダプターでは、複合の (つまり複数の) 基本キーが設定されている表をサポートしています。データベースに基本キーが 1 つ以上存在する場合、ウィザードは、テーブル・ビジネス・オブジェクトのそれらの列に基本キー・プロパティーを設定します。データベース表に基本キーが存在しない場合、外部サービス・ウィザードでは、そのビジネス・オブジェクトをディスカバーして構成するときに基本キー情報の入力を求めるプロンプトが出されます。例えば CUSTOMER (pkey1, pkey2) > ADDRESS (fkey1, fkey2) といった複合基本キーがある場合、アダプターは ADDRESS (fkey1) と CUSTOMER (pkey1)、ADDRESS (fkey2) と CUSTOMER (pkey2) をそれぞれ関連付けます。シーケンス列や ID 列など、固有のデータを持つ列を指定してください。ID 列 (Informix® では*シリアル列* と呼ばれる) は、データベースでテーブルの各行に自動的に固有の数値を生成する方法になります。テーブルには、ID 属性で定義される単一の列を作成できます。 ID 列の例として、オーダー番号、従業員番号、ストック番号、および問題番号などがあります。 ID 列は、DB2®、Informix および Microsoft® SQL Server のテーブルにのみ定義できます。

注: DB2 または Microsoft SQL Server データベースのいずれかの表に対してディスカバリー・プロセスを実行する場合に、その表で 1 つの列を ID 列として定義している場合、その表に対して生成されたビジネス・オブジェクトには、ID 列の固有 ID 属性は含まれていません。この場合、アプリケーション固有情報に属性を手動で追加することにより、生成されたビジネス・オブジェクトを編集する必要があります。これは、WebSphere Integration Developerのアセンブリー・エディターによって行うことができます。Informix データベースのテーブルに対してディスカバリー・プロセスを実行した場合は、固有 ID の属性を手動で追加する必要はありません。Informix の場合、生成されたビジネス・オブジェクトには、シリアル列の固有 ID 属性が含まれています。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.fffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

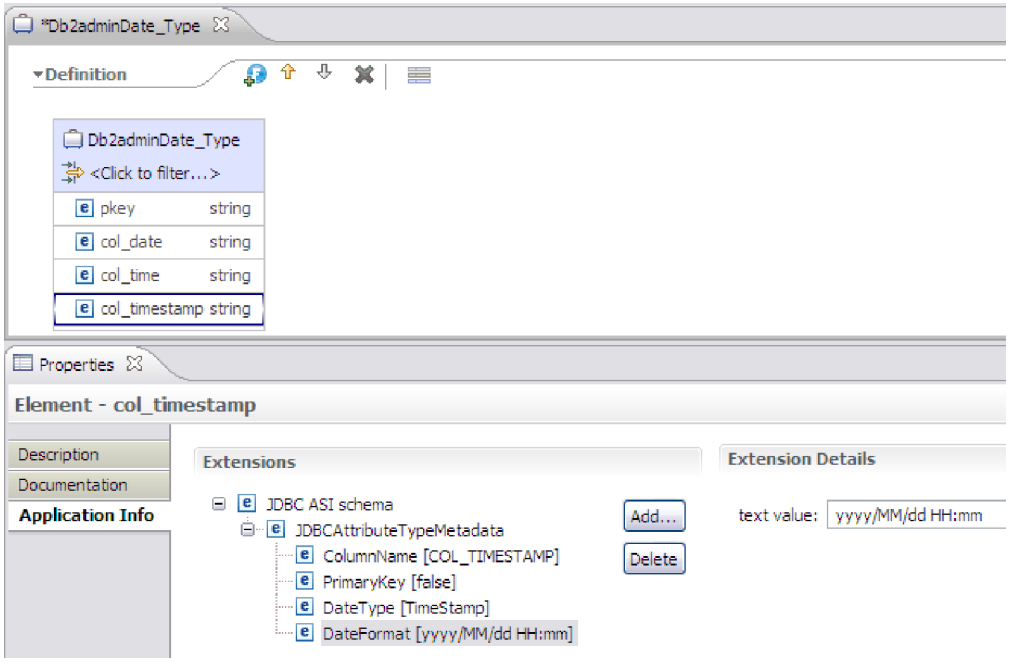


図 1. DateFormat アプリケーション固有情報とカスタマイズされた形式

テーブル・ビジネス・オブジェクトおよびビュー・ビジネス・オブジェクトは、Create、Update、Delete、Retrieve、RetrieveAll、Exists、および ApplyChanges の Outbound 操作をサポートします。階層型テーブル・ビジネス・オブジェクトに Exists 操作を実行すると、トップレベルのビジネス・オブジェクトのみが照会されます。

7 ページの図 2 に、ビジネス・オブジェクト・エディターに表示されたテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとに 1 つの属性が設定されています。表には子ビジネス・オブジェクトがないため、属性はすべて単純属性です。

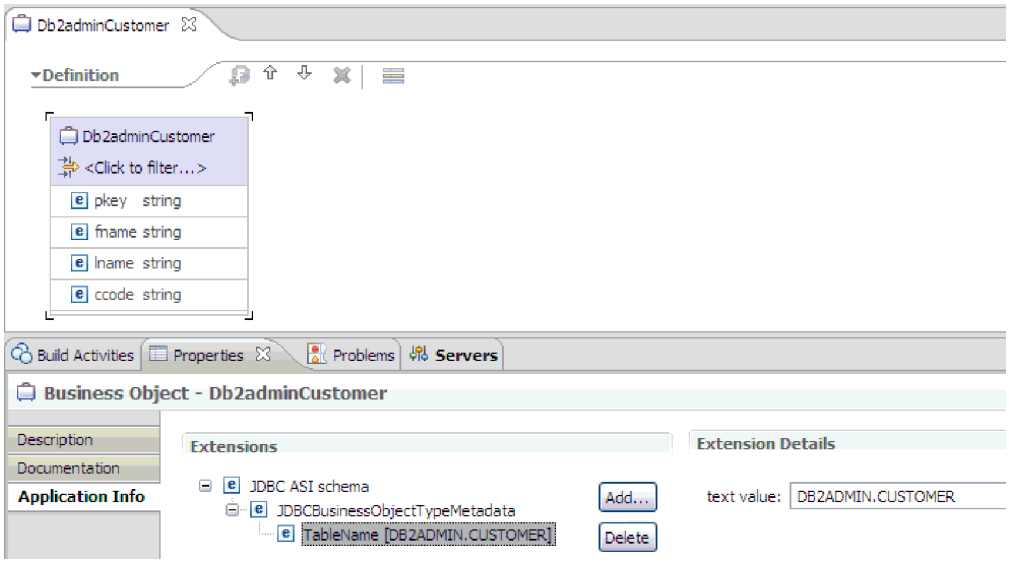


図 2. 子なしのテーブル・ビジネス・オブジェクト:

図 3 に、子テーブル・ビジネス・オブジェクトが 1 つあるテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとの単純属性に加えて、子ビジネス・オブジェクトを指す複合属性が設定されています。

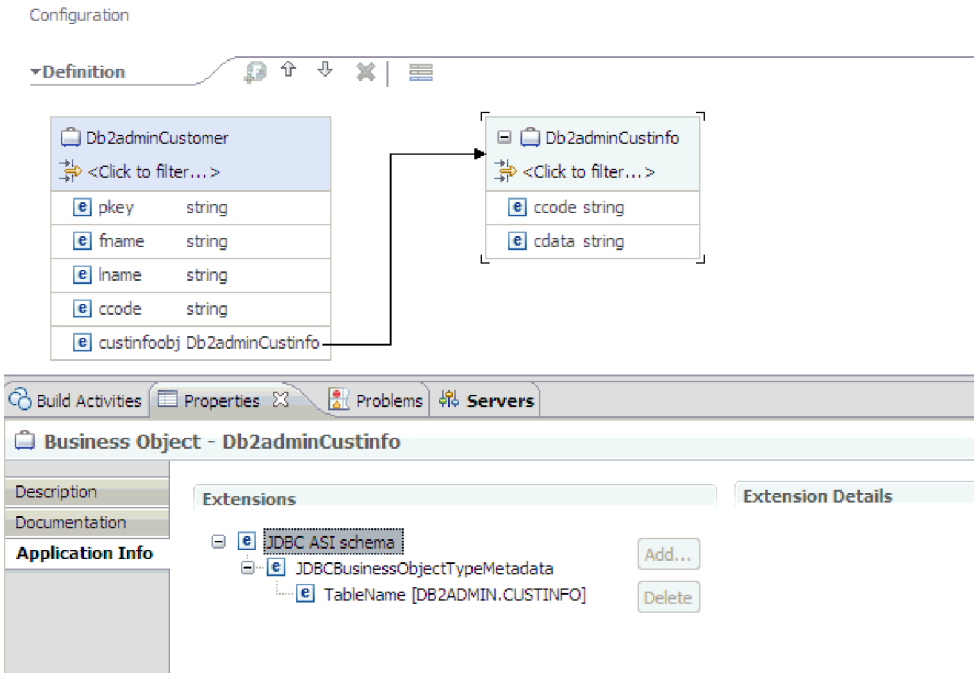


図 3. 子ビジネス・オブジェクトを 1 つ持つテーブル・ビジネス・オブジェクト:

Oracle データベースの場合、アダプターは、テーブル・ビジネス・オブジェクトで配列、テーブル、構造体、ネストされた構造体などのユーザー定義型または複合データ型をサポートします。これらの型に対しては、型名および子属性の詳細が自動的にディスカバーされて、表示されます。アダプターでは、テーブル・ビジネス・オブジェクトの子ビジネス・オブジェクトとしてこれらのデータ型が処理されま

す。

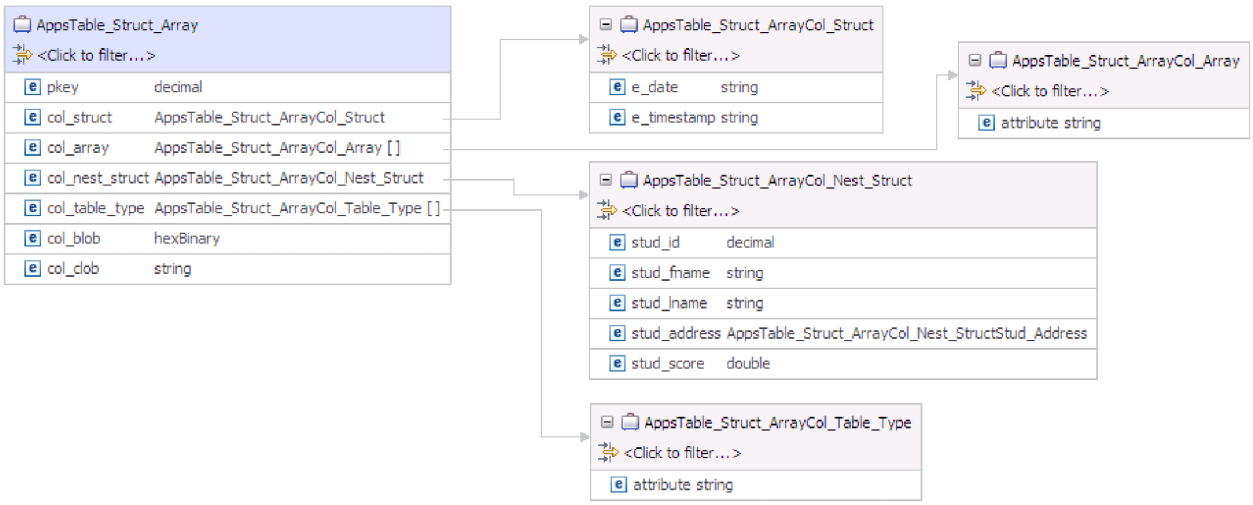


図 4. 列にユーザー定義型または複合型がある Oracle テーブル・ビジネス・オブジェクト

ストアード・プロシージャー・ビジネス・オブジェクトとストアード関数ビジネス・オブジェクトの場合

ストアード・プロシージャーまたはストアード関数のビジネス・オブジェクトでは、ストアード・プロシージャーまたはストアード関数のすべての入力パラメーターおよび出力パラメーターに、ビジネス・オブジェクトに対応する属性があります。入力または出力パラメーターのいずれかが、配列や構造体などの複合型である場合、対応するビジネス・オブジェクト属性は、配列または構造体の属性を含む子ビジネス・オブジェクトを持つ子ビジネス・オブジェクト型です。ストアード・プロシージャーが結果セットを戻した場合、戻された結果セットの属性を格納する子ビジネス・オブジェクトが作成されます。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

ストアード・プロシージャーとストアード関数のビジネス・オブジェクトは、Execute Outbound 操作をサポートします。

ストアード・プロシージャー・ビジネス・オブジェクトの構造を下のサンプル・ファイルに示します。ビジネス・オブジェクト ScottStrtValues および ScottStrtValuesStrt が、1 つの入力タイプと 2 つの出力タイプを持つストアード・プロシージャーから生成されます。出力パラメーターの 1 つは、構造体データ型です。外部サービス・ウィザードによって、構造体型のビジネス・オブジェクト ScottStrtValuesStrt が生成され、子オブジェクトとして親ビジネス・オブジェクト ScottStrtValues に追加されます。親ビジネス・オブジェクト内の構造体型の属性については、ChildBOType アプリケーション固有情報が Struct に設定され、型が構造体であることを示します。ChildBOTypeName アプリケーション固有情報は、データベース内のユーザー定義構造体型の値に設定されます。次の例は、ストアード・プロシージャーのスキーマを示しています。

ScottStrtValues ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvalues" xmlns:scottstrtvalues=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvalues" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt">
<import namespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt"
schemaLocation="ScottStrtvaluesStrt.xsd"/>
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvalues">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
<jdbcasi:MaxNumOfRetRS>0</jdbcasi:MaxNumOfRetRS>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>IP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="strt" type="scottstrtvaluesstrt:ScottStrtvaluesStrt"
```

```
minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
<jdbcasi:ChildBOType>STRUCT</jdbcasi:ChildBOType>
<jdbcasi:ChildBOTypeName>STRUCT1</jdbcasi:ChildBOTypeName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

ScottStrtValuesStrt ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvaluesStrt">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="name" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="title" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="dept_num" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
```

```
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

クエリー・ビジネス・オブジェクトの場合

データベース照会のビジネス・オブジェクトは、照会を実行する SQL ステートメントと、照会に必要なパラメーターを定義します。クエリー・ビジネス・オブジェクトでは、Outbound 操作 RetrieveAll がサポートされています。

例えば、次の SELECT ステートメントを実行するクエリー・ビジネス・オブジェクトがあるとします。

```
select C.pkey, C.fname, A.city from customer C, address A
    WHERE (C.pkey = A.custid) AND (C.fname like ?)
```

疑問符 (?) は、照会の入力パラメーターを示します。照会には複数のパラメーターを指定できます。各パラメーターは、SELECT ステートメントでは疑問符で示されています。サンプル・クエリー・ビジネス・オブジェクトの属性を表 1 に示します。クエリー・ビジネス・オブジェクトには、抽出される列ごとの単純属性、パラメーターごとの単純属性、およびパラメーター置換の後も WHERE 節を保持する、照会の WHERE 節の「プレースホルダー・オブジェクト」があります。

表 1. クエリー・ビジネス・オブジェクトの属性

ビジネス・オブジェクト属性	**説明**
pkey	Customer 表のデータベース列 PKEY に対応
fname	Customer 表のデータベース列 FNAME に対応
city	Address 表のデータベース列 CITY に対応
parameter1	パラメーター。SELECT ステートメントの ? (疑問符) ごとに 1 つのパラメーターがあります。複数のパラメーターを持つ SELECT ステートメントでは、後続のパラメーターに parameter2、parameter3 のようにして名前が付けられます。
jdbcwhereclause	WHERE 節のプレースホルダー・オブジェクト

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

以下の図に、ビジネス・オブジェクト・エディターに表示されたサンプル・クエリーのビジネス・オブジェクトを示します。

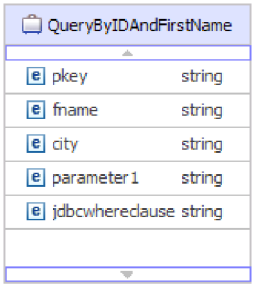


図 5. クエリー・ビジネス・オブジェクトの属性

この図は、クエリー・ビジネス・オブジェクト例のアプリケーション固有情報を示しています。SelectStatement アプリケーション固有情報には、SELECT ステートメントが含まれています。

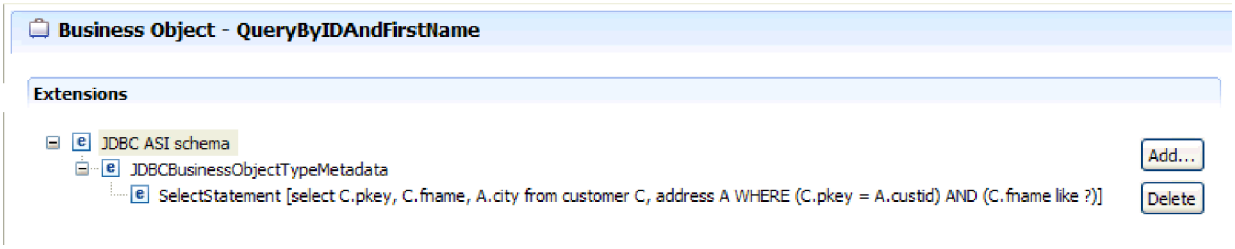


図 6. SELECT ステートメントは、ビジネス・オブジェクトのアプリケーション固有情報に保存されます。

Oracle データベースの場合、アダプターは、ビジネス・オブジェクトの照会結果で配列、テーブル、構造体、ネストされた構造体などの複合データ型をサポートします。アダプターでは、バッチおよびクエリーのビジネス・オブジェクトにおいて、これらの複合型をパラメーターとしてサポートしていません。

バッチ SQL ビジネス・オブジェクトの場合

バッチ SQL ビジネス・オブジェクトは、データベース・アクションを実行する INSERT、UPDATE、および DELETE SQL ステートメントと、そのステートメントで必要とされるパラメーターを定義します。バッチ SQL ビジネス・オブジェクトでは、Outbound 操作 Execute がサポートされています。

例えば、次の INSERT および DELETE ステートメントを実行するバッチ SQL ビジネス・オブジェクトがあるとします。

```
Insert into customer (pkey,ccode,fname,lname) values(?,?,?,?);
Delete From Customer where pkey=?
```

各疑問符 (?) は、ステートメントのパラメーターを示します。バッチ SQL ビジネス・オブジェクト内の各ステートメントには複数のパラメーターを指定できます。各パラメーターは、ステートメントでは疑問符で示されています。バッチ SQL ビ

ジネス・オブジェクトには複数のステートメントを含めることができ、それぞれが独自のパラメーター・セットを持ちます。図 7 に、それぞれが 1 つ以上のパラメーターを持つ INSERT ステートメントと DELETE ステートメントが定義されている、バッチ SQL ビジネス・オブジェクトのビジネス・オブジェクトの形式を示します。

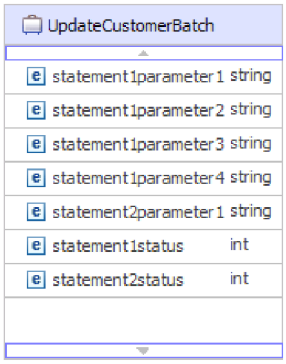


図 7. SQL ステートメントが 2 つのバッチ SQL ビジネス・オブジェクト

このビジネス・オブジェクトでは、statement1parameter1 や statement2parameter1 などの各ステートメントのパラメーターごとに属性が設定されています。また、statement1status や statement2status などの、各ステートメントの状況に関する属性もあります。ステートメント自体は、図 8 に示すように、ビジネス・オブジェクトについてのアプリケーション固有情報として格納されます。

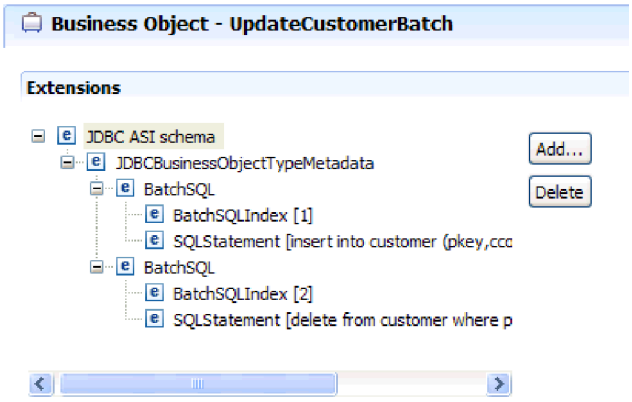


図 8. バッチ SQL ビジネス・オブジェクトのアプリケーション固有情報

ラッパー・ビジネス・オブジェクトの場合

ラッパー・ビジネス・オブジェクトにより、無関係の表の操作とビジネス・オブジェクトの表示を 1 回の操作で行うことができます。ラッパー・ビジネス・オブジェクトは、Create、Delete、Retrieve、および Update の Outbound 操作をサポートします。

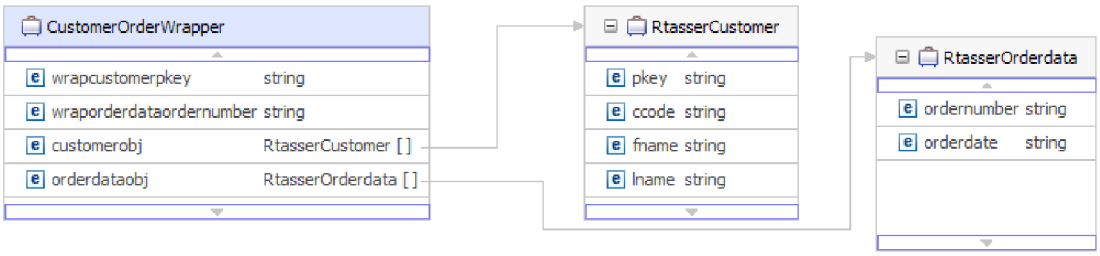


図 9. 2 つのテーブル・ビジネス・オブジェクトを含むラッパー・ビジネス・オブジェクト

ラッパー・ビジネス・オブジェクトには、子ビジネス・オブジェクトそれぞれの基本キーの単純属性が含まれています。フィールドの名前は「wrap」というストリングで、その後に、データベース表名とその表の基本キーの列名が続きます。ラッパー・ビジネス・オブジェクトには、テーブル・ビジネス・オブジェクトごとの複合属性も含まれています。属性の名前は、ストリング「obj」を付加した表名です。複合属性のタイプは、対応するテーブル・ビジネス・オブジェクトの名前です。

ビジネス・グラフ

アダプターの構成時に、ビジネス・グラフを生成するオプションを選択することもできます。バージョン 6.0.2 では、トップレベルの各ビジネス・オブジェクトがビジネス・グラフに含まれていますが、このビジネス・オブジェクトには、実行する操作に関する追加情報を指定するために、バージョン 6.0.2 でアプリケーションが使用できる動詞が組み込まれています。バージョン 7.0 では、ビジネス・グラフが必要になるのは以下の状況に限られます。

- Outbound ApplyChanges 操作を使用する必要がある場合
- バージョン 7.0 より前のバージョンの WebSphere Integration Developer で作成されたモジュールにビジネス・オブジェクトを追加する場合

ビジネス・グラフが存在する場合、ビジネス・グラフは処理されますが、ApplyChanges 以外のすべての操作で動詞は無視されます。

ビジネス・オブジェクトの作成方法

ビジネス・オブジェクトを作成するには、WebSphere Integration Developer から起動される外部サービス・ウィザードを使用します。このウィザードにより、データベースに接続し、データベース・オブジェクトがディスカバーされ、表示されます。ビジネス・オブジェクトを作成するデータベース・オブジェクトを選択します。例えば、調べるスキーマを指定します。指定されたスキーマで、テーブル、ビュー、ストアード・プロシージャー、ストアード関数、シノニム、およびニックネームを選択します。また、ビジネス・オブジェクトを追加で作成できます。例えば、データベースに対して実行されるユーザー定義の SELECT、INSERT、UPDATE、または DELETE ステートメントの結果を表すビジネス・オブジェクトを作成できます。このウィザードでは、親子関係と、無関係なビジネス・オブジェクトをまとめるラッパーを使用してビジネス・オブジェクト階層を作成できます。

必要なビジネス・オブジェクトを指定し、これらのオブジェクトの階層を定義すると、ウィザードにより、選択されたオブジェクトを表すビジネス・オブジェクトが生成されます。また、アダプターに必要なその他の成果物も生成されます。

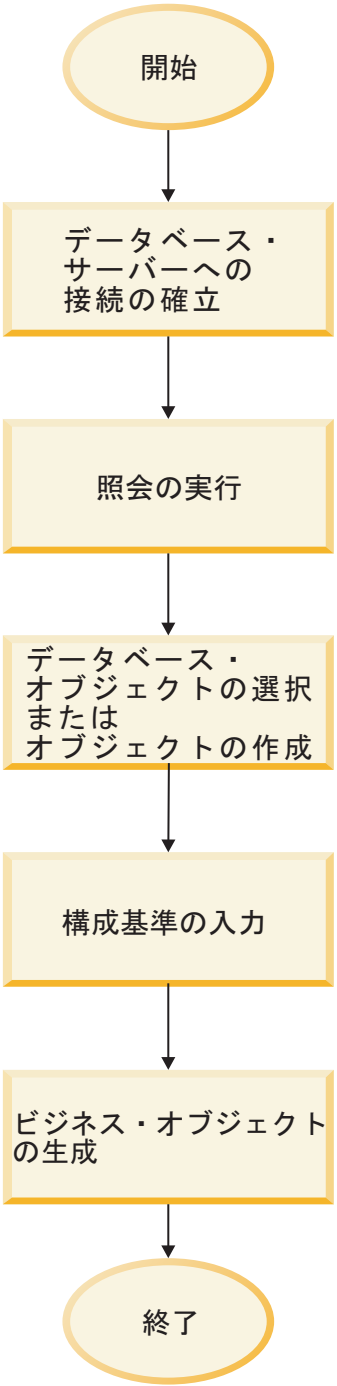


図 10. ビジネス・オブジェクトの作成方法

場合によっては、ウィザードで親子関係を完全に構成できないこともあります。これらの関係の場合は、WebSphere Integration Developer から起動するビジネス・オブジェクト・エディターを使用して、ウィザードによって作成されたビジネス・オブジェクト階層の定義を変更または完了します。詳しくは、WebSphere Integration Developer インフォメーション・センター (リンク: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp) で、ビジネス・オブジェクト・エディターによるビジネス・オブジェクトの変更方法を参照してください。

Create 操作

Create 操作は、要求内のビジネス・オブジェクトに対応したデータベース表に行を作成します。階層ビジネス・オブジェクトの場合は、Create 操作によってビジネス・オブジェクトが再帰的に全探索され、階層内の各ビジネス・オブジェクトに対応する行が作成されます。

Create 操作を処理するため、アダプターは次の操作を実行します。

1. ビジネス・オブジェクトがラッパーであるかどうかを確認します。トップレベルのビジネス・オブジェクトがラッパー・ビジネス・オブジェクトの場合、アダプターはこのビジネス・オブジェクトを無視します。ラッパー・オブジェクトの行は作成されません。
2. 所有関係を伴う単一カーディナリティーの各子ビジネス・オブジェクトを、データベース内に再帰的に挿入します。つまり、アダプターは、子および子とその孫に含まれるすべての子ビジネス・オブジェクトを作成します。

   ビジネス・オブジェクト定義上、ある属性がある単一カーディナリティーの子ビジネス・オブジェクトを表すものとされている場合に、その属性が空であると、アダプターはその属性を無視します。ただし、ビジネス・オブジェクト定義により、その属性が子を表すことが必要であるにもかかわらず、子を表していない場合には、アダプターはエラーを戻して処理を停止します。
3. 所有関係を伴わない単一カーディナリティーの各子ビジネス・オブジェクトの有無を検索し、確認します。検索が失敗して、子がデータベース内に存在しないことが示された場合、アダプターはエラーを返して処理を停止します。Retrieve 操作が成功した場合、アダプターは子ビジネス・オブジェクトを再帰的に更新します。

   注: データベースに子ビジネス・オブジェクトが存在する場合に、このアプローチが正しく機能するには、子ビジネス・オブジェクト内の基本キー属性の相互参照が、Create 操作時に正しく行われる必要があります。アプリケーション・データベースに子ビジネス・オブジェクトが存在しない場合、基本キー属性は設定してはいけません。
4. トップレベルのビジネス・オブジェクトをデータベース内に挿入するため、次の操作を実行します。
   a. トップレベルのビジネス・オブジェクトの各外部キー値を、対応する単一カーディナリティーの子ビジネス・オブジェクトの基本キー値に設定します。子ビジネス・オブジェクトの値は、データベース・シーケンスまたはカウンター、あるいはデータベース自体によって、子の作成時に設定される場合があります。そのため、このステップでは、アダプターが親をデータベースに挿入する前に、親の外部キー値を正しいものにします。
   b. データベースによって自動的に設定される属性のそれぞれに対して、新しい固有 ID 値を生成します。データベース・シーケンスまたはカウンターの名前は、属性のアプリケーション固有情報に格納されます。データベース・シーケンスまたはカウンターが属性に関連付けられている場合、アプリケーション・サーバーから渡された値があれば、アダプターによって生成された値によって上書きされます。
   c. トップレベルのビジネス・オブジェクトをデータベース内に挿入します。

注: アダプターは空の複合列をヌル列として処理します。その値がヌルに設定されているかどうかは関係ありません。

5. 複数カーディナリティーの子ビジネス・オブジェクトのすべてを、次のように処理します。
   a. それぞれの子の外部キー値を、親に含まれる対応する基本キー属性の値を参照するように設定します。親の基本キー値は、親の作成時に生成されている可能性があるため、アダプターがデータベースに子を挿入する前に、この処理によって、それぞれの子の外部キー値が正しくなるようにします。
   b. 複数カーディナリティーの子ビジネス・オブジェクトのすべてを、データベースに挿入します。

Update 操作

Update 操作は、ソース・ビジネス・オブジェクトを、トップレベルのソース・ビジネス・オブジェクトで指定された基本キーを使用してデータベースから検索されたビジネス・オブジェクトと比較することによって実行されます。

階層ビジネス・オブジェクトの更新時に、アダプターは次の操作を実行します。

1. ソース・ビジネス・オブジェクトの基本キー値を使用して、データベース内の対応するエンティティーを検索します。検索されたビジネス・オブジェクトは、データベース内のデータの現在の状態を正確に表したものです。

   検索が失敗した場合 (トップレベルのビジネス・オブジェクトがデータベース内に存在しないことを意味します)、アダプターは `RecordNotFoundException` エラーを戻し、更新は失敗します。

   検索に成功した場合、アダプターは、検索されたビジネス・オブジェクトをソース・ビジネス・オブジェクトと比較して、どの子ビジネス・オブジェクトに関してデータベースに変更を加える必要があるかを判別します。ただし、アダプターはソース・ビジネス・オブジェクトの単純属性の値と検索されたビジネス・オブジェクトの単純属性の値を比較しません。アダプターは、非キーの単純属性すべての値を更新します。

   トップレベルのビジネス・オブジェクトのすべての単純属性がキーを表している場合、アダプターはそのトップレベルのビジネス・オブジェクト用の更新照会を生成できません。この場合、アダプターは、警告を記録してから次に進みます。
2. トップレベルのビジネス・オブジェクトの子のうち、単一カーディナリティーのものすべてを再帰的に更新します。

   ビジネス・オブジェクト定義上、ある属性がある子ビジネス・オブジェクトを表すことが必須である場合には、その子ビジネス・オブジェクトがソース・ビジネス・オブジェクトと検索されたビジネス・オブジェクトの両方に存在している必要があります。存在しない場合、Update 操作は失敗し、アダプターはエラーを戻します。

   アダプターでは、所有関係にある単一カーディナリティーの子を、次のいずれかの方法で処理します。

- ソース・ビジネス・オブジェクトおよび検索したビジネス・オブジェクトの両方に子が存在する場合、アダプターはデータベース内の既存の子を更新する代わりに、既存の子を削除して新規の子を作成します。
- その子がソース・ビジネス・オブジェクトには存在するにもかかわらず、検索されたビジネス・オブジェクトには存在しない場合、アダプターはデータベース内にその子を再帰的に作成します。
- その子が検索されたビジネス・オブジェクトには存在するにもかかわらず、ソース・ビジネス・オブジェクトには存在しない場合、アダプターはデータベース内のその子を再帰的に削除します。

所有関係にない単一カーディナリティーの子に関しては、アダプターは、ソース・ビジネス・オブジェクトに存在するそのような子のすべてを、データベースから検索しようとします。アダプターは、子の検索に成功すると、その子ビジネス・オブジェクトにデータを取り込みますが、更新は行いません。これは、所有関係にない単一カーディナリティーの子はアダプターによって変更されることがないためです。

3. 検索されたビジネス・オブジェクトのすべての単純属性を更新します。ただし、ソース・ビジネス・オブジェクト内の対応する属性が指定されていない場合を除きます。

   更新されるビジネス・オブジェクトは一意である必要があるため、アダプターは、結果として 1 行のみが処理されることを確認します。複数の行が戻されている場合、アダプターはエラーを戻します。

   トップレベルのビジネス・オブジェクトがラッパー・ビジネス・オブジェクトの場合、これは無視されます。ラッパー・ビジネス・オブジェクトの更新は実行されません。
4. 検索されたビジネス・オブジェクトの複数カーディナリティーの子のそれぞれを、次のいずれかの方法で処理します。
   - その子がソース・ビジネス・オブジェクトの配列と検索されたビジネス・オブジェクトの配列の両方に存在する場合、アダプターはデータベース内でその子を再帰的に更新します。
   - その子がソース・ビジネス・オブジェクトの配列には存在しても、検索されたビジネス・オブジェクトの配列には存在しない場合、アダプターはデータベース内でその子を再帰的に作成します。
   - その子が検索されたビジネス・オブジェクトの配列には存在しても、ソース・ビジネス・オブジェクトの配列には存在しない場合、アダプターはデータベースからその子を再帰的に削除します。ただし、親に含まれているその子を表す属性のアプリケーション固有情報で、KeepRelationship プロパティーが `true` に設定されている場合を除きます。この場合、アダプターは、データベースからその子を削除しません。

NULL データと Update 操作

アダプターは、データベース表で列値が NULL のレコードを更新できます。例えば、Customer ビジネス・オブジェクトに、custid、ccode、fname、および lname という列があり、custid と ccode が複合キーを形成しているとします。複合キーとは、複数の属性を参照する基本キーであり、ビジネス・オブジェクトの一意性を定

義するときに使用されます。ccode が NULL の Customer レコードを更新できます。アダプターにより、Update 操作の更新照会が次のように生成されます。

```
update customer set fname=?, lname=? where custid=? and ccode is null
```

注: アダプターは空の複合列をヌル列として処理します。その値がヌルに設定されているかどうかは関係ありません。

Outbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成

モジュールで使用するテーブル、ビュー、およびシノニムまたはニックネームのビジネス・オブジェクトを選択して構成するには、ビジネス・オブジェクトの構成プロパティーを指定します。

始める前に

このタスクを実行するには、データベース内のデータの構造や、モジュールがどんなデータベース・オブジェクトにアクセスする必要があるかを理解しなければなりません。具体的には、以下の情報を認識しておく必要があります。

- テーブル、ビュー、およびシノニムまたはニックネームの構造 (必要な列や、データ型などの列属性を含む)。
- テーブル間の関係 (親子関係のカーディナリティーおよび所有権を含む)。

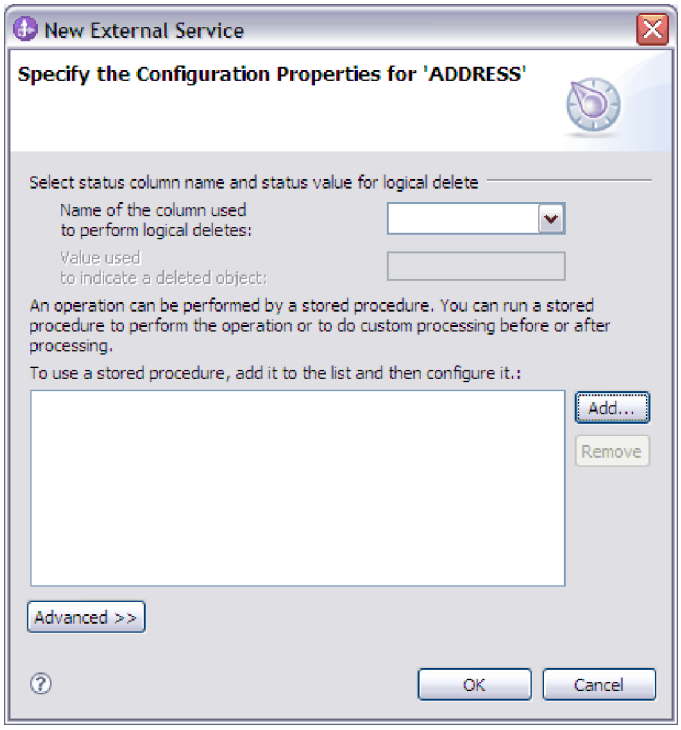
このタスクについて

このタスクは、外部サービス・ウィザードを介して実行されます。「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで作業を開始し、「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')ウィンドウ (構成するビジネス・オブジェクト固有のウィンドウ) で作業します。

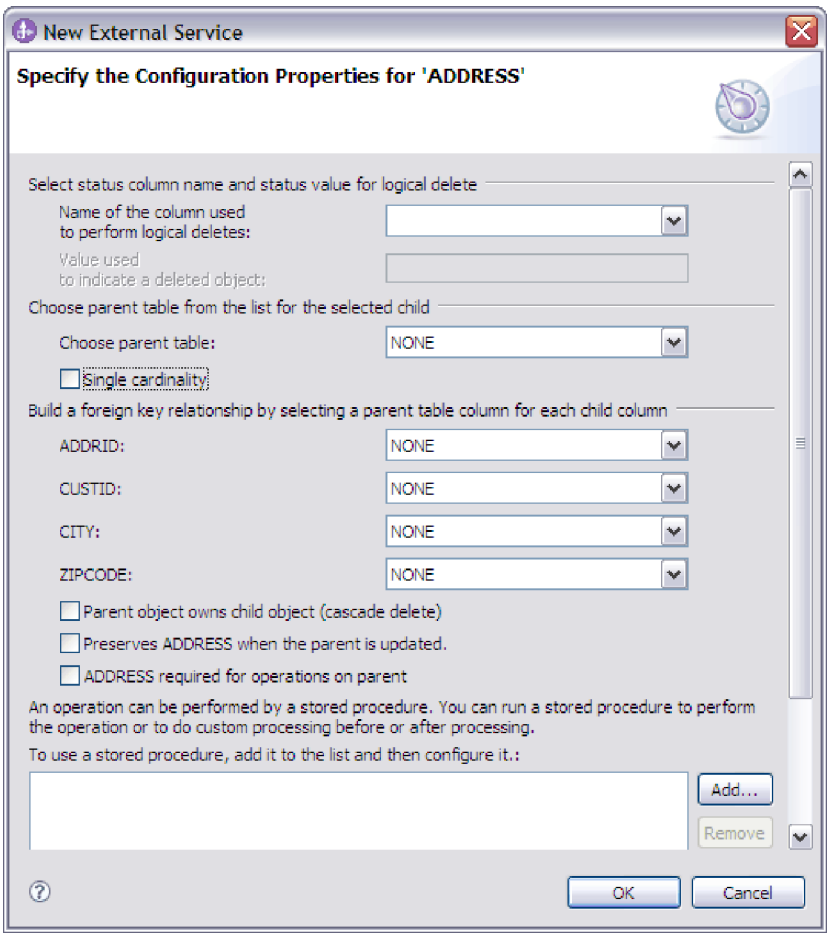
手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウの「**検出済みオブジェクト**」リストで、テーブル、ビュー、またはシノニムを 1 つ以上選択し、「**>**」 (追加) ボタンをクリックします。オブジェクトが「**選択済みオブジェクト**」リストに追加されます。

   以下の 2 つの図に、テーブル、ビュー、シノニム、またはニックネームのビジネス・オブジェクトの標準的な「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウを示します。最初の図に、選択する最初のテーブルまたはテーブル・グループの標準的なウィンドウを示します。



以下の図に、選択する後続のテーブルの標準的なウィンドウを示します。少なくとも 1 つのテーブルを選択して構成した後は、後続テーブルの「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに、テーブル間の親子階層をオプションで定義することができる領域が表示されます。



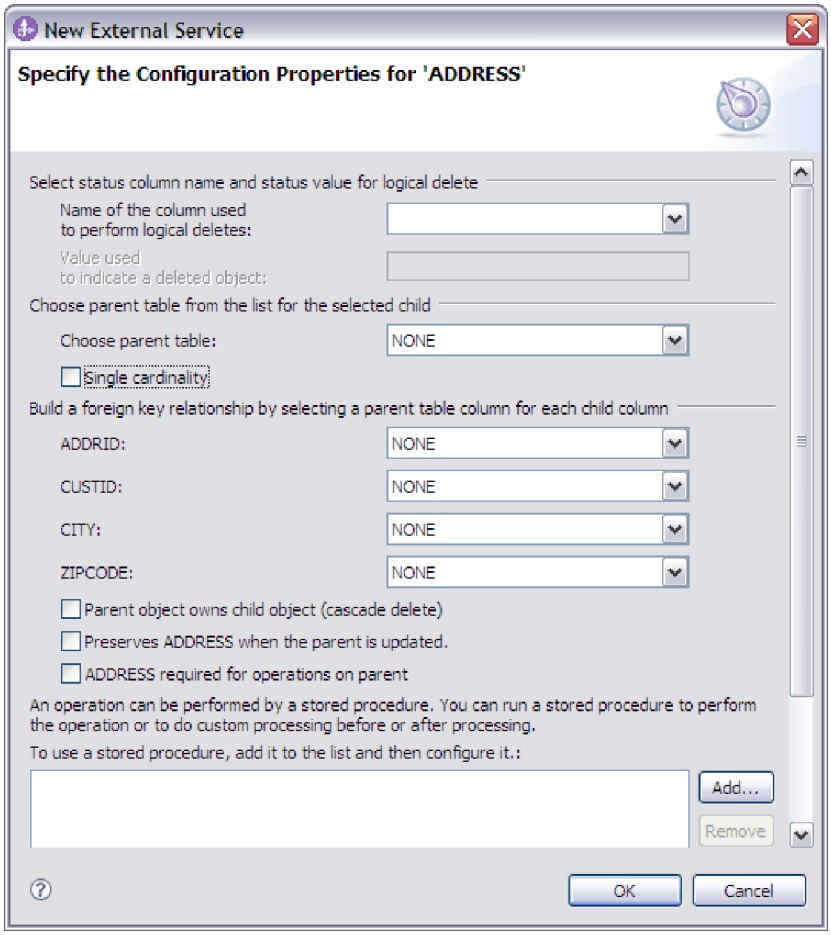
オブジェクトの構成時には、拡張構成を必要とする選択を行うと、このウィンドウに追加のフィールドが表示され、ウィンドウがスクロールされる場合があります。必ずウィンドウのすべてのフィールドを確認してから、「**OK**」をクリックしてください。

2. 論理削除を示すのに使用される列がテーブルにある場合は、次の手順に従います。
   a. 「**論理削除を実行するのに使用される列の名前**」フィールドで列名を選択します。
   b. 「**削除されたオブジェクトを示すために使用する値**」フィールドに、行が論理的に削除されていることを示す値を入力します。 この値については、データベース管理者に確認できます。
3. 「**テーブル *table_name* の基本キーの選択**」エリアが表示されたら、「**追加**」をクリックし、テーブル・ビジネス・オブジェクトの基本キーとして使用する列を選択してから、「**OK**」をクリックします。テーブルに複合キーがある場合は、複数の列を選択できます。 「**テーブル *table_name* の基本キーの選択**」エリアは、データベース表に基本キーとして指定された列が存在しない場合にのみ表示されます。各テーブル・ビジネス・オブジェクトには、関連付けられたデータベース表にキーがない場合でも、基本キーが定義されている必要があります。データベースで基本キーが定義されている場合、ウィンドウのこのセクションは表示されません。

4. オプション: ビジネス・オブジェクト間の親子関係を定義します。

   親子階層を作成する場合、まず親テーブルを構成して「エンタープライズ・システムでのオブジェクトの検索」ウィンドウに戻り、子テーブルを選択して構成します。

   以下の図に示す「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウの領域を使用して、親子関係を構成します。これらのフィールドは、構成する最初のテーブルの場合には表示されません。



   a. 「**親テーブルの選択**」フィールドで、構成する親テーブルの名前を選択します。 リストに親テーブルが表示されていない場合は、親テーブルがまだ構成されていません。戻って親オブジェクトを構成してから、子オブジェクトを構成してください。データベースで外部キー参照を定義した場合、親テーブルを選択すると、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。テーブルとその親のテーブル間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスが自動的に選択されます。

   b. 関係のカーディナリティーを指定します。

      - テーブルとその親テーブルの間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択します。単一カーディナリティー関係では、親はこのタイプの子ビジネス・オブジェク

トを 1 つのみ持つことができます。単一カーディナリティー関係は、所有関係を伴って実際の子を表すか、または所有関係を伴わずにルックアップ・テーブルまたはデータベース内の他の対等オブジェクトを表すために使用できます。

- テーブルに複数カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択しないでください。複数カーディナリティー関係では、親がこのタイプの子ビジネス・オブジェクトの配列を持つことができます。

c. 親と子の間に外部キー関係を作成するため、子の列ごとに、親テーブルの外部キーであるかどうかを指定します。

- 子の列が外部キーでない場合は、「なし」を選択します。
- 子の列が外部キーの場合は、その子の列に対応する親テーブルの列を選択します。

  注: ウィザードは、1 つの親テーブルのみを構成できます。子テーブルに複数の親テーブルがある場合は、ビジネス・オブジェクト・エディターを使用して、ウィザードを終了した後に残りの親テーブルを構成する必要があります。

d. 親オブジェクトが子オブジェクトを所有している場合、データベース内の子オブジェクトは親が削除されるときに削除されます。この子がその親によって所有されていることを示すには、「**親オブジェクトが子オブジェクトを所有する (カスケード削除)**」チェック・ボックスを選択します。あるいは、このオプションをクリアして、ルックアップ・テーブルなどの子オブジェクトが、親の削除時に削除されないようにします。

e. Update 操作の一環として子オブジェクトが削除されることのないようにするには、「**親の更新時に *child_table_name* を保持する**」チェック・ボックスを選択します。

  親テーブルが更新されると、アダプターは入力に存在する子ビジネス・オブジェクトを、データベースから返される子ビジネス・オブジェクトと比較します。デフォルトでは、アダプターは、入力ビジネス・オブジェクト内に存在しない、データベースから返されたすべての子オブジェクトを削除します。

f. デフォルトでは、子ビジネス・オブジェクトを指定せずに、親ビジネス・オブジェクトに対して操作を実行できます。親ビジネス・オブジェクトを変更対象として実行依頼するときに、その親ビジネス・オブジェクトで子ビジネス・オブジェクトが必ず指定されるようにしたい場合は、「***Child_table_name* は、親に対する操作で必須**」チェック・ボックスを選択します。

5. 操作を実行するには、アダプターによって生成される標準 SQL ステートメントを使用するか、あるいはデータベース内のストアード・プロシージャーまたはストアード関数を使用します。ストアード・プロシージャーまたはストアード関数を使用する場合は、以下の手順を実行します。

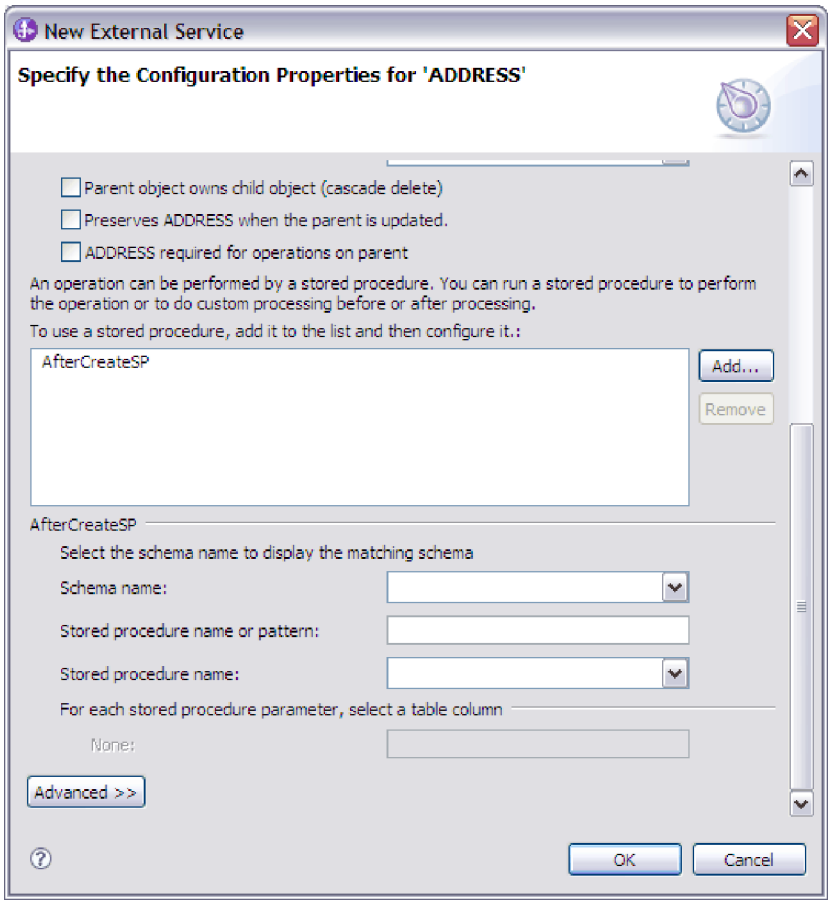
a. 「**追加**」をクリックします。

b. 「追加」ウィンドウで、実行するストアード・プロシージャーのタイプを選択します。 操作ごとに、その操作を実行するストアード・プロシージャーと、操作の前後に実行するストアード・プロシージャーを選択できます。例

えば、Create 操作の場合は、ストアード・プロシージャー CreateSP、BeforeCreateSP、および AfterCreateSP のどれでも指定できます。

注: RetrieveAllSP を指定してテーブルを構成する場合は、ストアード・プロシージャーが 1 つの結果セットのみを返すことを確認します。ストアード・プロシージャーの ResultSet ASI を true に設定して、実行時に例外「ストアード・プロシージャーに関連した結果セットが見つかりませんでした (No resultset found associated with the stored procedure)」、「結果セットが返されませんでした (No resultset returned)」、「複数の結果セットが返されました (More than one resultset returned)」のいずれも生成されないようにします。

c. 「**OK**」をクリックします。選択したストアード・プロシージャーのタイプが「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに表示されます。このウィンドウは、各ストアード・プロシージャーの構成用の領域を表示するために拡張されます。新しい領域を表示するには、スクロールダウンする必要がある場合もあります。

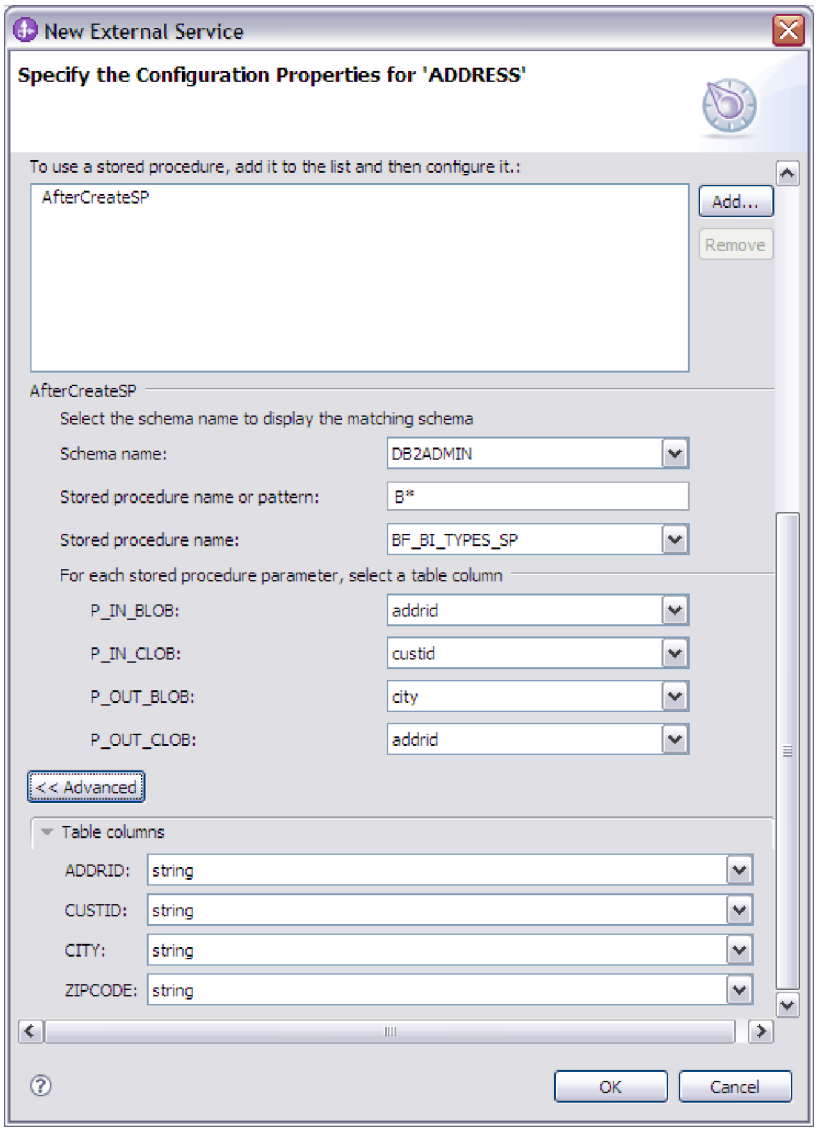


注: 階層ビジネス・オブジェクトにおいて、その階層のビジネス・オブジェクトごとにストアード・プロシージャーを実行する場合は、ストアード・プロシージャーを、トップレベルのビジネス・オブジェクトと、ビジネス・オブジェクトの各子ビジネス・オブジェクトまたは配列に別々に関連付ける必要があります。ストアード・プロシージャーをトップレベルのビジネス・オブ

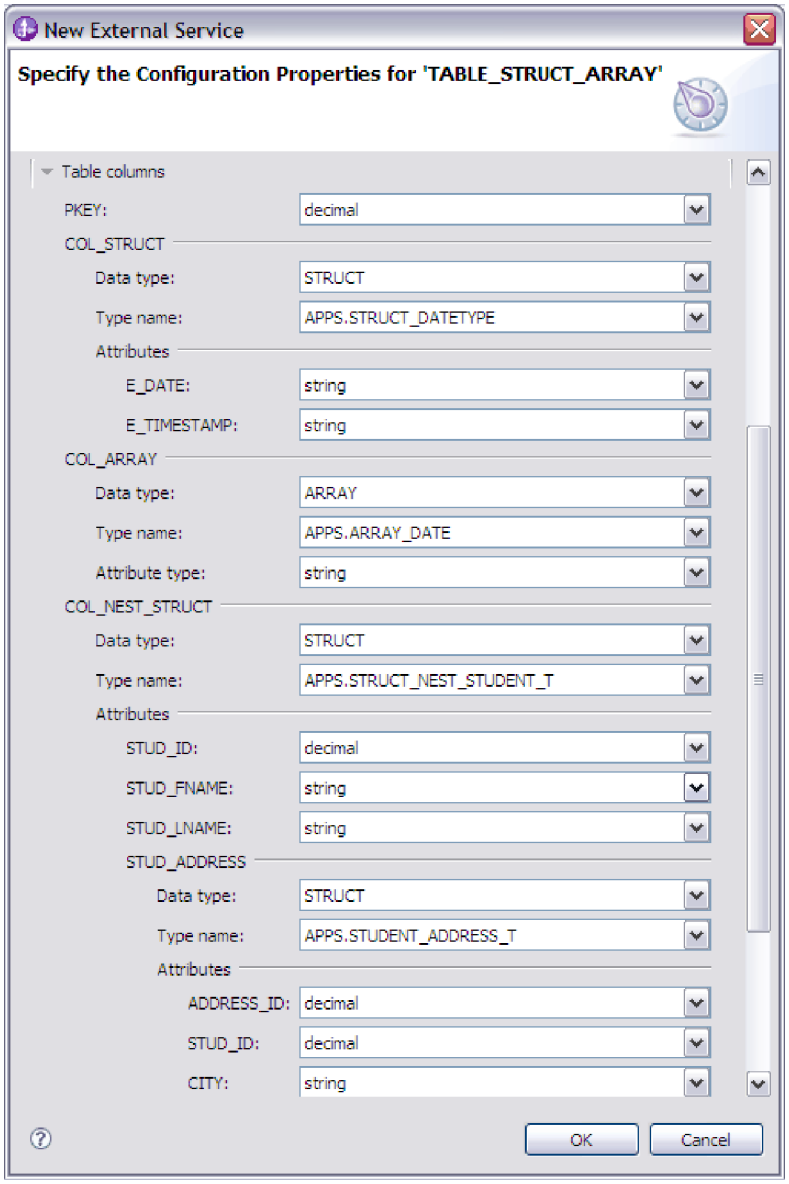
ジェクトに関連付けても、各子ビジネス・オブジェクトに関連付けないと、そのトップレベルのビジネス・オブジェクトはストアード・プロシージャーで処理されますが、子ビジネス・オブジェクトは標準 SQL 照会を使用して処理されます。

6. 選択したストアード・プロシージャー・タイプごとに、データベース内でのストアード・プロシージャーの名前を指定し、ビジネス・オブジェクトを構成します。
   a. 「**スキーマ名**」フィールドで、ストアード・プロシージャーが含まれるスキーマの名前を選択します。
   b. ストアード・プロシージャーまたはストアード関数の名前を指定します。
      1) 「**ストアード・プロシージャー名またはパターン**」フィールドで、ストアード・プロシージャーまたはストアード関数の名前を入力するか、または名前パターンを入力します。1 つの文字と一致させる場合は疑問符または下線 (? または _) を使用し、複数の文字と一致させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。
      2) 「**ストアード・プロシージャー名**」フィールドで、目的のプロシージャーの名前を選択します。

      「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウが拡張して、ストアード・プロシージャーを構成するための領域が表示されます。ウィザードは、データベース内のストアード・プロシージャーを調べることにより、パラメーターのリストを自動生成します。
   c. ストアード・プロシージャーのパラメーターごと (左側) に、そのパラメーターでストアード・プロシージャーに渡すテーブル列 (右側) を選択します。次の図に、ストアード・プロシージャーを構成した後のウィンドウの一部を示します。



7. テーブル内の各列のデータ型マッピングを指定するには、以下の手順を実行します。
   a. 「**拡張**」をクリックします。
   b. 「**テーブル列**」を展開します。テーブル内の列ごとに、デフォルトのデータ型マッピングが表示されます。 Oracle データベースにおいて、配列、構造体、ネストされた構造体、テーブルなど、何らかのユーザー定義型または複合データ型がテーブルに含まれている場合は、型名および子属性の詳細も自動的にディスカバーされて、表示されます。次の図に、複合データ型を含む Oracle テーブルの型名および子属性の詳細を示します。



c. マッピングを確認して、必要な場合は変更します。

8. ウィンドウのすべてのフィールドの操作が完了したら、「**OK**」をクリックします。ビジネス・オブジェクトの構成が保存されます。 定義したビジネス・オブジェクト (テーブル、ビュー、シノニム、およびニックネーム) が、「エンタープライズ・システムでのオブジェクトの検索」ウィンドウにリストされます。
9. 「**選択済みオブジェクト**」リストのオブジェクトの構成を変更するには、オブジェクト名を選択して、 (編集) アイコンをクリックします。

次のタスク

エンタープライズ・システムでのオブジェクトの検索ウィンドウで、他のタイプのビジネス・オブジェクトの選択と構成を続行します。完了したら、「**次へ**」をクリックして、グローバル・プロパティーを設定し、ラッパー・ビジネス・オブジェクトを構成します。

クエリー・ビジネス・オブジェクトの選択および構成

モジュールで使用するユーザー定義 SELECT ステートメントのクエリー・ビジネス・オブジェクトを選択および構成します。

始める前に

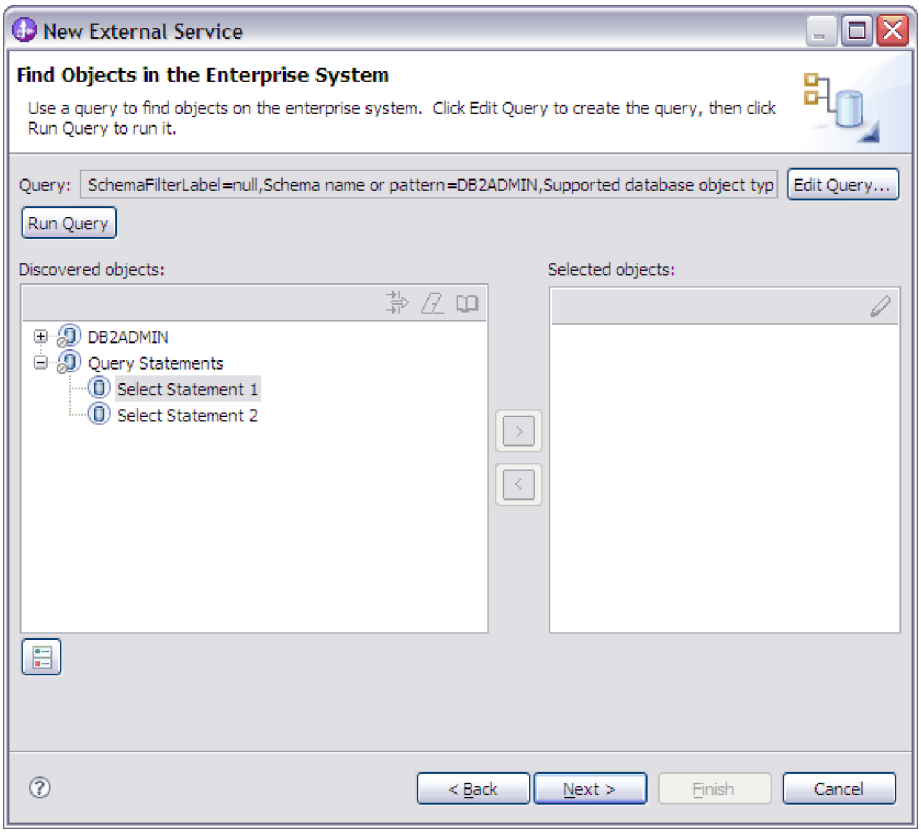
クエリー・ビジネス・オブジェクトを構成するには、テーブルおよびビューをはじめとする、データベース内のデータ構造がわかっていなければなりません。モジュールがアクセスするべき列の名前およびデータ型を知っておく必要があります。さらに、SQL SELECT ステートメントを記述できなければなりません。

このタスクについて

このタスクは、外部サービス・ウィザードを介して実行されます。「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで作業を開始し、「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')ウィンドウ (構成するビジネス・オブジェクト固有のウィンドウ) で作業します。

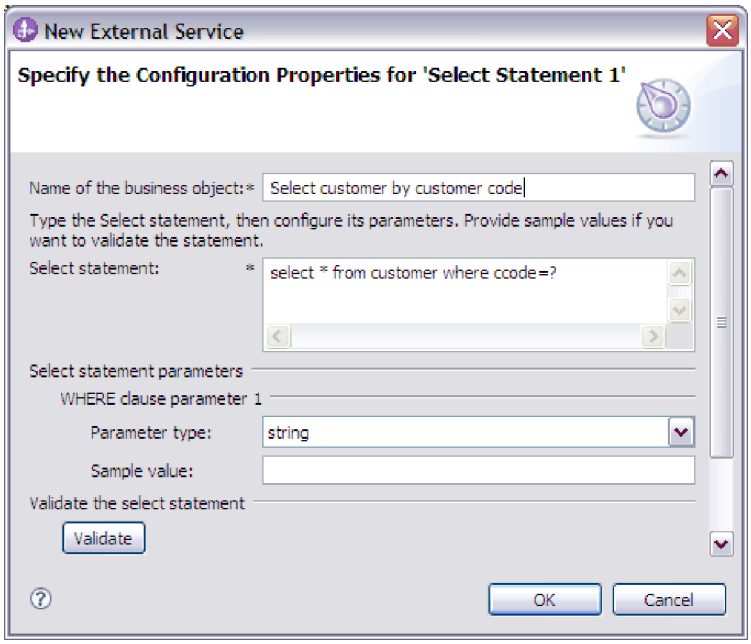
手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウの **「検出済みオブジェクト」**リストで、**「クエリー・ステートメント (Query Statements)」** ノードを展開します。 このノードには、「照会プロパティーの指定」ウィンドウで要求した各クエリー・ビジネス・オブジェクトのオブジェクト・テンプレート**「Select ステートメント *n*」**があります。例えば、前述のウィンドウでクエリー・ビジネス・オブジェクト数を 2 と指定した場合、**「ディスカバーされたオブジェクト (Discovered objects)」**リストには 2 つのオブジェクト・テンプレートが表示されます (以下の図を参照)。



2. オブジェクト・テンプレートを 1 つ以上選択し、「**>**」 (追加) ボタンをクリックして、オブジェクトを「**選択されたオブジェクト (Selected objects)**」リストに追加します。
3. 「**ビジネス・オブジェクトの名前**」フィールドに、ビジネス・オブジェクトの名前を入力します。 この名前には、スペースおよび各国語文字を使用できます。
4. 「**select ステートメント**」フィールドに、実行する SELECT ステートメントを入力します。各パラメーターは疑問符 (**?**) で示します。 次のサンプル SELECT ステートメントは、クエリー・ビジネス・オブジェクトの柔軟性を示しています。
   - `select * from customer where ccode=?`
   - `select * from customer where id=? and age=?`
   - `select * from customer where lname like ?`
   - `select C.pkey, C.fname, A.city from customer C, address A WHERE (C.pkey = A.custid) AND (C.fname like ?)`

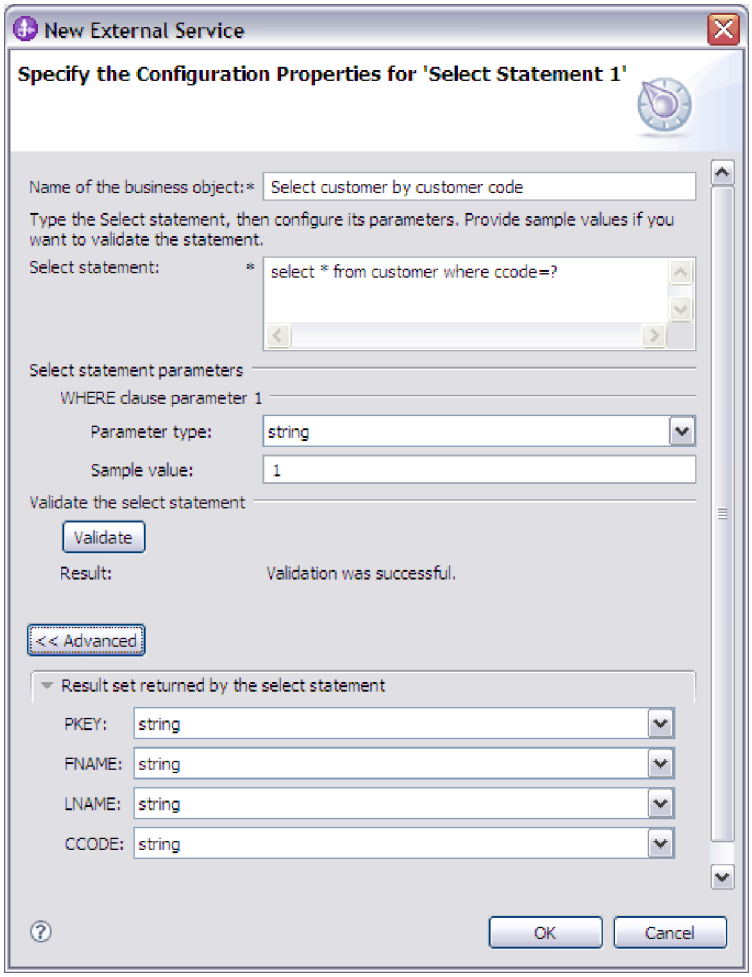
   注: SELECT ステートメントの FROM 節にネストされた SELECT ステートメントが含まれないようにしてください。
   ? を入力するごとに、ウィンドウが拡張して、そのパラメーターの WHERE 節を定義するための領域が表示されます。次の図は、単一のパラメーターを持つクエリー・ビジネス・オブジェクトの「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウを表しています。



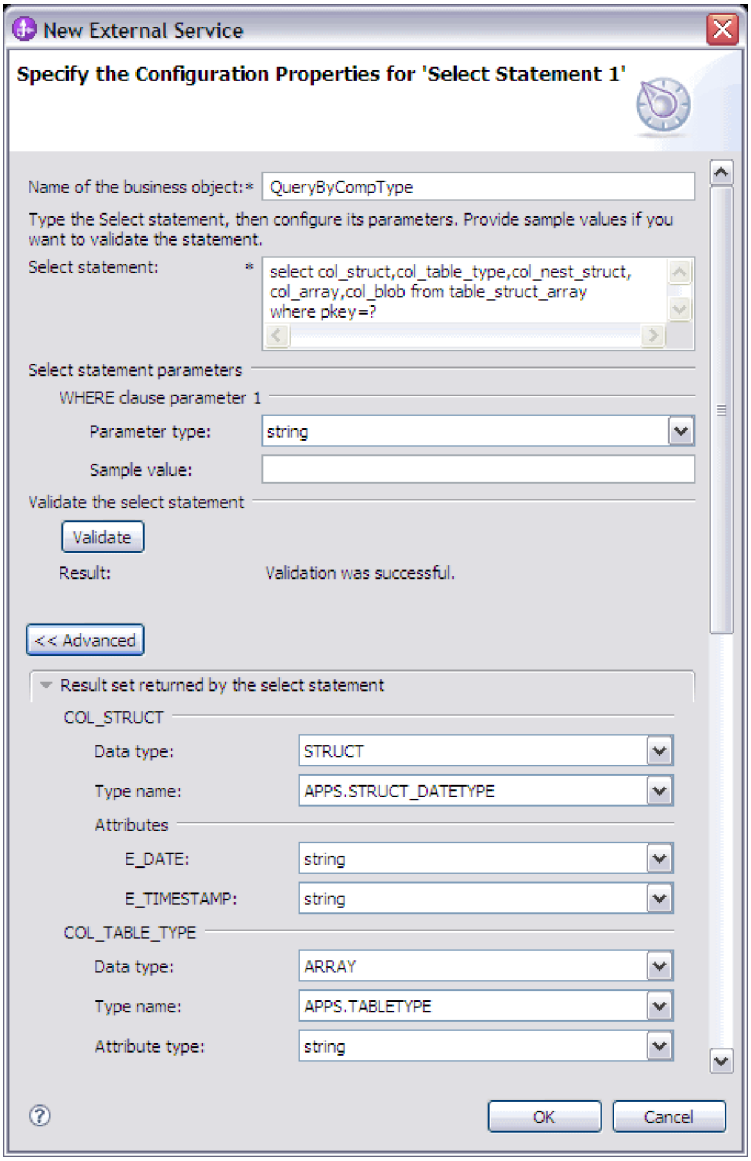
5. 「**WHERE 節パラメーター *n***」エリアに、 SELECT ステートメントの各パラメーターに関する情報を入力します。
   a. 「**パラメーター・タイプ**」フィールドで、パラメーターのデータ型を選択します。Oracle データベースの場合、アダプターはバッチおよびクエリーのビジネス・オブジェクトにおいて、配列、テーブル、構造体、ネストされた構造体などの複合型をパラメーターとしてサポートしていません。
   b. 「**サンプル値**」フィールドに、パラメーターのサンプル値を入力します。

   例えば、顧客の姓が格納されている列に対応するパラメーターの場合は、データ型として `string` を選択し、サンプル値 `Smith` を指定します。
6. 「**検証**」をクリックします。「**結果**」エリアに検証結果が表示されます。

   「**結果**」エリアに「検証は失敗しました。」と表示された場合は、入力した情報に問題があります。 「検証は失敗しました。」の後に表示されているデータベース・サーバーからのエラー・メッセージを参考にして、定義を訂正します。SELECT ステートメントの構文、パラメーターのデータ型、およびサンプル・データを確認してください。検証が成功した場合は、「**拡張**」ボタンが表示されます。
7. SELECT ステートメントから返される結果セット内の各列のデータ型マッピングを指定するには、以下の手順を実行します。
   a. 「**拡張**」をクリックします。
   b. 「**Select ステートメントによって返された結果セット**」を展開します。結果セット内の列ごとに、デフォルトのデータ型マッピングが表示されます。



Oracle データベースにおいて、配列、構造体、ネストされた構造体、テーブルなど、何らかの複合データ型が照会結果に含まれている場合は、型名および子属性の詳細も自動的にディスカバーされて、表示されます。以下の図は、Oracle テーブルの照会結果の型名および子属性の詳細を表しています。



c. マッピングを確認して、必要な場合は変更します。

8. 「**OK**」をクリックして、クエリー・ビジネス・オブジェクトの定義を保存します。

タスクの結果

定義したクエリー・ビジネス・オブジェクトが「エンタープライズ・システムでのオブジェクトの検索」ウィンドウにリストされます。

次のタスク

エンタープライズ・システムでのオブジェクトの検索ウィンドウで、他のタイプのビジネス・オブジェクトの選択と構成を続行します。完了したら、「**次へ**」をクリックして、グローバル・プロパティーを設定し、ラッパー・ビジネス・オブジェクトを構成します。

ビジネス・オブジェクト属性

ビジネス・オブジェクト属性は、ビジネス・オブジェクトの内容を定義するものであり、データベース・オブジェクトの列リストから作成されます。各属性は、名前、型、カーディナリティーなどのプロパティーを持ちます。外部サービス・ウィザードで、列名に属性名が設定されます。アダプターでは、属性のカーディナリティー、型、およびアプリケーション固有情報が追加されます。

ビジネス・オブジェクトは、属性で指定されるデータのコンテナーです。データベースのデータの構造はビジネス・オブジェクトによって定義されますが、データベースのデータはビジネス・オブジェクト属性内にあります。

表 2 に、ビジネス・オブジェクト属性のプロパティーをリストし、それらの解釈および設定値について説明します。

表 2. 属性プロパティー

プロパティー	解釈と設定値
Cardinality	ビジネス・オブジェクトのカーディナリティーを示す整数。1 つの子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す各ビジネス・オブジェクト属性は、カーディナリティーの値が単一 (1) または複数 (制限のない整数) になります。 単一カーディナリティー関係および複数カーディナリティー関係の両方で、親ビジネス・オブジェクトと子ビジネス・オブジェクトの間の関係は、その関係を保管するビジネス・オブジェクトのキー属性に含まれるアプリケーション固有情報によって記述されます。
Foreign Key	カーディナリティーが *n* の子ビジネス・オブジェクトの配列が検索されると、SELECT ステートメントの WHERE 文節で外部キーが使用されます。 RetrieveAll 操作は、キーおよび外部キーの使用を指定変更します。 **注:** アダプターでは、子ビジネス・オブジェクトを表す属性を外部キーとして指定することについては、サポートしていません。
Name	このプロパティーは、属性が単純属性の場合は、属性の固有の名前。属性が子ビジネス・オブジェクトの場合は、ビジネス・オブジェクトの名前を表します。
MinOccurs MaxOccurs	列が基本キーではなく、かつ NULL 値を取れない場合、MinOccurs および MaxOccurs 属性は必須であり、値は 1 以上に設定されます。
Primary Key	この属性が基本キーかどうかを示します。各ビジネス・オブジェクトで少なくとも 1 つの単純属性を基本キーとして指定する必要があります。 単純属性の基本キー・プロパティーを `true` に設定すると、アダプターは、ビジネス・オブジェクトの処理中に生成する SELECT および SQL UPDATE の各ステートメントの WHERE 文節にその属性を追加します。RetrieveAll 操作は、基本キーおよび外部キーの使用を指定変更します。 **注:** アダプターでは、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す属性を基本キー属性として指定することについてはサポートしていません。

表 2. 属性プロパティー (続き)

プロパティー	解釈と設定値
必須	属性が値を含む必要があるかどうかを指定します。カーディナリティーが単一 (1) のコンテナーに対して、このプロパティーが `true` に設定されている場合、アダプターでは、その親ビジネス・オブジェクトが、この属性に対応する子ビジネス・オブジェクトを含んでいる必要があります。Create、Update、および Delete 操作でアダプターに渡されるビジネス・オブジェクトは、子ビジネス・オブジェクトも含んでいなければなりません。単純属性のカーディナリティーは単一 (1) で、コンテナー属性のカーディナリティーは複数 (n) です。ビジネス・オブジェクトが必須属性に対して有効な値またはデフォルト値を持っていないと、アダプターでは Create 操作が失敗します。このオブジェクトに対するデータベースからの検索時に使用可能なデータがない場合も、Create 操作は失敗します。
タイプ	単純属性の場合、このプロパティーは属性の型 (Integer、String、Date、Timestamp、Boolean、Double、Float など) を指定します。サポートされる単純属性の型と、それらがマップされるデータベース・オブジェクトの JDBC タイプを表 3 に示します。 子ビジネス・オブジェクトを指定する属性の場合、このプロパティーはビジネス・オブジェクトの名前を指定します。

JDBC メタデータとして戻される各データベース・オブジェクトのタイプは、表 3 のリストにあるようにビジネス・オブジェクト属性タイプにマップされます。リストされている JDBC タイプのみがアダプターでサポートされます。リストされていないタイプの列は、ビジネス・オブジェクトに追加されません。その場合は、問題を説明する通知メッセージが生成されます (例: テーブル *yyyy* の列名 *xxxx* のタイプはサポートされていません。ビジネス・オブジェクトには追加されません)。

表 3. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ
BIT	Boolean
CHAR LONGVARCHAR VARCHAR	String
NUMERIC	Decimal Int
INTEGER SMALLINT TINYINT	Int
BIGINT	Long Int
TIME	String Time
TIMESTAMP	String DateTime

表 3. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ (続き)

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ
DATE	String Date Datetime
DECIMAL	Decimal
DOUBLE FLOAT	Double
REAL	Float
BLOB	hexBinary
CLOB	String
BINARY VARBINARY LONGBINARY	hexBinary
NCHAR NVARCHAR NTEXT	String
TEXT	String
RAW	hexBinary
MONEY SMALLMONEY	Decimal
STRUCT または ARRAY	アダプターは、これらのデータ型をテーブル・ビジネス・オブジェクトまたはクエリー・ビジネス・オブジェクトの子ビジネス・オブジェクトとして処理します。 **注:** アダプターが複合型をサポートするのは、Oracle テーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクトの場合のみです。テーブルに配列、構造体、ネストされた構造体、またはテーブルなどの複合データ型が含まれている場合は、型名およびサブ属性の詳細も自動的にディスカバーされて表示されます。 **注:** アダプターは空の複合列をヌルとして処理します。それがヌルに設定されているかどうかは関係ありません。

表 3. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ (続き)

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ

XML	String
	
	Oracle の場合、ランタイム環境で XML データ型を扱うには、さらに以下のライブラリーが必要です。
	xdb.jar
	xmlparserv2.jar
	
	また、JNDI データ・ソースを使用してデータベースに接続している場合は、データ・ソースの作成時に必ず xdb.jar と xmlparserv2.jar をクラスパスに追加してください。
	注: XML 型は、テーブル・ビジネス・オブジェクトでのみサポートされます。
	
	XML コンテンツの例:
	
	`<customer>`
	`<fname>John</fname>`
	`<lname>Smith</lname>`
	`</customer>`
	
	詳しくは、 77 ページの『Oracle での XML データ型のサポート』を参照してください。

第 3 章 テーブル・ビジネス・オブジェクト内の XML データ型

テーブル・ビジネス・オブジェクト内での XML 型が、アダプターでサポートされるようになりました。データベース内のテーブル列が XML 型のものである場合、XML アプリケーション固有情報は True に設定されます。XML メタデータ型は、String ビジネス・オブジェクト属性型にマップします。

外部ソフトウェア依存関係の追加

外部サービス・ウィザード がデータベース・サーバーと通信できるようにするには、データベースの特定ファイルのコピーが必要です。外部サービス・ウィザードを使用して、JDBC ドライバーと必要なネイティブ・システム・ライブラリー・ファイルが格納されている JAR ファイルの場所を指定します。

始める前に

このタスクを実行するには、WebSphere Integration Developer で外部サービス・ウィザードを実行します。

このタスクについて

モジュール構成時にこのタスクを実行するだけでなく、場合によっては、WebSphere Process Server または WebSphere Enterprise Service Bus にファイルをデプロイする必要があります。

手順

1. データベース管理者またはデータベース・ソフトウェアの Web サイトから、ご使用のデータベース・ソフトウェアおよびオペレーティング・システムの JDBC ドライバー固有ファイルまたはネイティブ・ライブラリーを入手します。 必要なファイルの種類は、データベース・サーバーによって異なります。JDBC ドライバー固有のファイルが JRE 1.6 と互換性があることを確認してください。次の表に、一般的なデータベース・ソフトウェアに必要な JDBC ドライバー・ファイルをリストします。

表 4. 共通データベース・ソフトウェアの JDBC ドライバー・ファイル

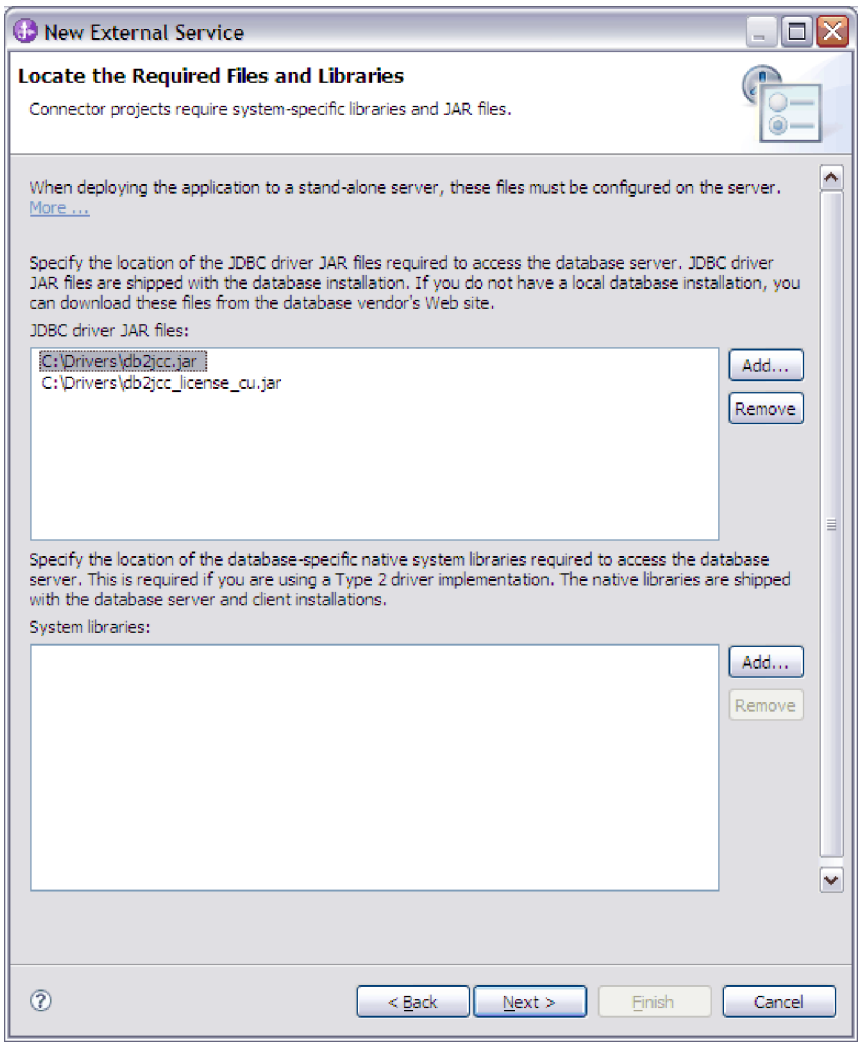
データベース・ソフトウェア	ドライバー	JDBC ドライバー・ファイル	ネイティブ・システム・ライブラリー
IBM® DB2 Universal Database for Linux®, UNIX®, and Windows®	IBM DB2 Universal (Type 4)	db2jcc.jar db2jcc_license_cu.jar	なし
IBM DB2 for z/OS	IBM DB2 Universal (Type 4)	db2jcc.jar db2jcc_license_cisuz.jar	なし

表 4. 共通データベース・ソフトウェアの JDBC ドライバー・ファイル (続き)

データベース・ソフトウェア	ドライバー	JDBC ドライバー・ファイル	ネイティブ・システム・ライブラリー
IBM DB2 for i5/OS®	IBM Toolbox for Java™ リモート・ドライバー (Type 4)	jt400.jar db2jcc_license_cisuz.jar	なし
	IBM Toolkit for Java ネイティブ・ドライバー* (Type 2)	db2_classes.jar	なし
IBM DB2 Universal Database for Linux, UNIX, and Windows	IBM DB2 Universal (Type 2)	db2java.zip	なし
Oracle	Thin ドライバー	ojdbc6.jar ランタイム環境で XML データ・タイプを扱うには、さらに以下の追加ライブラリーが必要です。 xdb.jar xmlparserv2.jar 詳しくは、 77 ページの『Oracle での XML データ型のサポート』を参照してください。	なし
Microsoft SQL Server 2005	Microsoft SQL Server 2005 for JDBC	sqljdbc.jar	なし

*IBM Toolkit for Java ネイティブ・ドライバーは、アダプター実行時のデータベースへの接続に使用することはできますが、ウィザード実行時の接続に使用することはできません。ディスカバリー・プロセス中は、IBM Toolbox for Java リモート・ドライバー、または、IBM DB2 Universal ドライバーを使用する必要があります。ただし、実行時にネイティブ・ドライバーを使用するようにモジュールを構成することはできます。これは、「サービス生成およびデプロイメント・プロパティーの指定」ウィンドウで行います。

2. 「必要なファイルおよびライブラリーの位置指定」ウィンドウで、プロジェクトに必要な JDBC ドライバー固有ファイルの場所を指定します。
   a. 「**JDBC ドライバー JAR ファイル**」フィールドで、「**追加**」をクリックして、JDBC ドライバー・ファイルを選択します。 JDBC ドライバーについて詳しくは、「JDBC ドライバーについてのよくある質問 (Frequently asked questions about JDBC drivers)」を参照してください。
   b. JDBC タイプ 2 ドライバーを使用する場合は、「**システム・ライブラリー**」フィールドで「**追加**」をクリックして、データベース・サーバーへのアクセスに必要なネイティブ・ システム・ライブラリーを追加します。 タイプ 4 の JDBC ドライバーを使用する場合のみ、このフィールドを空のままにします。



3. 「**次へ**」をクリックします。 処理方向の選択ウィンドウが表示されます。

タスクの結果

ウィザードに、データベース・サーバーと通信するために必要なファイルがあります。

次のタスク

外部サービス・ウィザードでの作業を続行します。次のステップでは、ウィザードがデータベースに接続するために必要となる情報を設定します。

ビジネス・オブジェクト属性

ビジネス・オブジェクト属性は、ビジネス・オブジェクトの内容を定義するものであり、データベース・オブジェクトの列リストから作成されます。各属性は、名前、型、カーディナリティーなどのプロパティーを持ちます。外部サービス・ウィザードで、列名に属性名が設定されます。アダプターでは、属性のカーディナリティー、型、およびアプリケーション固有情報が追加されます。

ビジネス・オブジェクトは、属性で指定されるデータのコンテナーです。データベースのデータの構造はビジネス・オブジェクトによって定義されますが、データベースのデータはビジネス・オブジェクト属性内にあります。

表 5 に、ビジネス・オブジェクト属性のプロパティーをリストし、それらの解釈および設定値について説明します。

表 5. 属性プロパティー

プロパティー	解釈と設定値
Cardinality	ビジネス・オブジェクトのカーディナリティーを示す整数。1 つの子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す各ビジネス・オブジェクト属性は、カーディナリティーの値が単一 (1) または複数 (制限のない整数) になります。 単一カーディナリティー関係および複数カーディナリティー関係の両方で、親ビジネス・オブジェクトと子ビジネス・オブジェクトの間の関係は、その関係を保管するビジネス・オブジェクトのキー属性に含まれるアプリケーション固有情報によって記述されます。
Foreign Key	カーディナリティーが *n* の子ビジネス・オブジェクトの配列が検索されると、SELECT ステートメントの WHERE 文節で外部キーが使用されます。 RetrieveAll 操作は、キーおよび外部キーの使用を指定変更します。 **注:** アダプターでは、子ビジネス・オブジェクトを表す属性を外部キーとして指定することについては、サポートしていません。
Name	このプロパティーは、属性が単純属性の場合は、属性の固有の名前。属性が子ビジネス・オブジェクトの場合は、ビジネス・オブジェクトの名前を表します。
MinOccurs MaxOccurs	列が基本キーではなく、かつ NULL 値を取れない場合、MinOccurs および MaxOccurs 属性は必須であり、値は 1 以上に設定されます。
Primary Key	この属性が基本キーかどうかを示します。各ビジネス・オブジェクトで少なくとも 1 つの単純属性を基本キーとして指定する必要があります。 単純属性の基本キー・プロパティーを `true` に設定すると、アダプターは、ビジネス・オブジェクトの処理中に生成する SELECT および SQL UPDATE の各ステートメントの WHERE 文節にその属性を追加します。RetrieveAll 操作は、基本キーおよび外部キーの使用を指定変更します。 **注:** アダプターでは、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す属性を基本キー属性として指定することについてはサポートしていません。

表 5. 属性プロパティー (続き)

プロパティー	解釈と設定値
必須	属性が値を含む必要があるかどうかを指定します。カーディナリティーが単一 (1) のコンテナーに対して、このプロパティーが `true` に設定されている場合、アダプターでは、その親ビジネス・オブジェクトが、この属性に対応する子ビジネス・オブジェクトを含んでいる必要があります。Create、Update、および Delete 操作でアダプターに渡されるビジネス・オブジェクトは、子ビジネス・オブジェクトも含んでいなければなりません。単純属性のカーディナリティーは単一 (1) で、コンテナー属性のカーディナリティーは複数 (n) です。ビジネス・オブジェクトが必須属性に対して有効な値またはデフォルト値を持っていないと、アダプターでは Create 操作が失敗します。このオブジェクトに対するデータベースからの検索時に使用可能なデータがない場合も、Create 操作は失敗します。
タイプ	単純属性の場合、このプロパティーは属性の型 (Integer、String、Date、Timestamp、Boolean、Double、Float など) を指定します。サポートされる単純属性の型と、それらがマップされるデータベース・オブジェクトの JDBC タイプを表 6 に示します。 子ビジネス・オブジェクトを指定する属性の場合、このプロパティーはビジネス・オブジェクトの名前を指定します。

JDBC メタデータとして戻される各データベース・オブジェクトのタイプは、表 6 のリストにあるようにビジネス・オブジェクト属性タイプにマップされます。リストされている JDBC タイプのみがアダプターでサポートされます。リストされていないタイプの列は、ビジネス・オブジェクトに追加されません。その場合は、問題を説明する通知メッセージが生成されます (例: テーブル *yyyy* の列名 *xxxx* のタイプはサポートされていません。ビジネス・オブジェクトには追加されません)。

表 6. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ
BIT	Boolean
CHAR LONGVARCHAR VARCHAR	String
NUMERIC	Decimal Int
INTEGER SMALLINT TINYINT	Int
BIGINT	Long Int
TIME	String Time
TIMESTAMP	String DateTime

表 6. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ (続き)

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ
DATE	String Date Datetime
DECIMAL	Decimal
DOUBLE FLOAT	Double
REAL	Float
BLOB	hexBinary
CLOB	String
BINARY VARBINARY LONGBINARY	hexBinary
NCHAR NVARCHAR NTEXT	String
TEXT	String
RAW	hexBinary
MONEY SMALLMONEY	Decimal
STRUCT または ARRAY	アダプターは、これらのデータ型をテーブル・ビジネス・オブジェクトまたはクエリー・ビジネス・オブジェクトの子ビジネス・オブジェクトとして処理します。 **注:** アダプターが複合型をサポートするのは、Oracle テーブル・ビジネス・オブジェクトおよびクエリー・ビジネス・オブジェクトの場合のみです。テーブルに配列、構造体、ネストされた構造体、またはテーブルなどの複合データ型が含まれている場合は、型名およびサブ属性の詳細も自動的にディスカバーされて表示されます。 **注:** アダプターは空の複合列をヌルとして処理します。それがヌルに設定されているかどうかは関係ありません。

表 6. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ (続き)

JDBC メタデータの列タイプ	ビジネス・オブジェクト属性のタイプ

XML	String
	
	Oracle の場合、ランタイム環境で XML データ型を扱うには、さらに以下のライブラリーが必要です。
	xdb.jar
	xmlparserv2.jar
	
	また、JNDI データ・ソースを使用してデータベースに接続している場合は、データ・ソースの作成時に必ず xdb.jar と xmlparserv2.jar をクラスパスに追加してください。
	注: XML 型は、テーブル・ビジネス・オブジェクトでのみサポートされます。
	
	XML コンテンツの例:
	`<customer>`
	`<fname>John</fname>`
	`<lname>Smith</lname>`
	`</customer>`
	
	詳しくは、77 ページの『Oracle での XML データ型のサポート』を参照してください。

属性に関するアプリケーション固有情報

ビジネス・オブジェクト属性のアプリケーション固有情報 (ASI) は、属性が単純属性であるか、子ビジネス・オブジェクト (または子ビジネス・オブジェクトの配列) を表す属性であるかによって異なります。子を表す属性のアプリケーション固有情報は、親子関係が子に格納されるか親に格納されるかによっても異なります。

単純属性のアプリケーション固有情報

単純属性では、アプリケーション固有情報の形式は、いくつかのパラメーターとその値で構成されています。単純属性に必要な唯一のパラメーターは列名です。単純属性のアプリケーション固有情報については、表 7で説明します。

表 7. 単純属性のアプリケーション固有情報

パラメーター	タイプ	説明	デフォルト値
BLOB	Boolean	この属性に対応するデータベース列が BLOB データ型であるかどうかを示します。BLOB データの表示中、アダプターはバイト数を 16 進値で表示します。属性の型は hexBinary です。 `True` に設定されている場合、列のデータ型は BLOB です。	なし

表 7. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
ByteArray	Boolean	列がバイナリー・データ型であるかどうかを示します。`True` の場合、アダプターはデータベースでバイナリー・データを読み取りおよび書き込みし、そのデータをストリングとしてアプリケーション・サーバーに送信します。アダプターは、ビジネス・オブジェクトにバイナリー・データを設定します。属性の型は hexBinary です。	`False`
ChildBOType	String	属性が複合データ型の場合は、このアプリケーション固有情報を使用して実際の型を指定します。 • Struct • Array • ResultSet	なし
ChildBOTypeName	String	ChildBOType アプリケーション固有情報の値が Struct または Array の場合、このパラメーターはユーザー定義タイプの名前を表します。この値は、大/小文字の区別があります。	
CLOB	Boolean	この属性に対応するデータベース列が CLOB データ型であるかどうかを示します。この値は、String 型の属性にのみ適用されます。 `True` の場合は、列のデータ型は CLOB です。 CLOB 属性は String 型で、その長さを使用して CLOB の長さを定義します。	なし
ColumnName	String	この属性に対応するデータベース列の名前。 これが唯一の必須パラメーターです。	なし

表 7. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
CopyAttribute	String	同一ビジネス・オブジェクトまたは親ビジネス・オブジェクト内から別の属性名を参照するユーザー指定値。 アプリケーション固有情報で設定されている値が、同一ビジネス・オブジェクト内の別の属性の名前を参照している場合、アダプターは、Create 操作でビジネス・オブジェクトをデータベースに追加する前に、その別の属性の値を使用してこの属性の値 (アプリケーション固有情報が定義されている属性) を設定します。 例えば、テーブルの新しい行の `contact` 列に、email 列と同じ値を組み込むには、`contact` 属性の CopyAttribute パラメーターに `email` を設定します。 値から子ビジネス・オブジェクトの属性を参照することはできませんが、親ビジネス・オブジェクト内の属性は参照できます。このためには、属性名の前に 2 つのピリオドを付加します。例えば、親ビジネス・オブジェクト内の `ccode` 属性を参照するには、`..ccode` と指定します。 アプリケーション固有情報にこのパラメーターを指定しないと、アダプターは、別の属性から値をコピーせずに現行属性の値を使用します。	なし
DateType	String	対応するエレメントが日付、時刻、またはタイム・スタンプであることを指定します。次の値のいずれかを指定してください。 • Date • Time • Timestamp DateType 型の属性の値を設定するときには、次の形式で設定します。 • Date には、*yyyy-MM-dd* を使用します。 • Time には、*hh:mm:ss* を使用します。 • Timestamp には、*yyyy-MM-dd hh:mm:ss.fffffffff* を使用します。 **注:** java.sql.Timestamp.valueOf () メソッドにより、JDBC タイム・スタンプ形式が Timestamp 値に変換され、その値がデータベースに挿入されます。java.sql.Timestamp.toString() メソッドは、データベースにある Timestamp 値をストリングに変換します。アダプターは、データベースに含まれる Timestamp 値を使用します。 Timestamp メソッドについての詳しい説明は、Sun の Web サイト (http://java.sun.com/j2se/1.5.0/docs/api/) で、 *Timestamp* を検索し、参照してください。	なし

表 7. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
DateFormat	String	Date、Time、および Timestamp の各データ型の形式をカスタマイズできます。アダプターは、Date、Time、または Timestamp の SQL データ型をストリングに変換する (およびその逆の変換を行う) 必要があるときに、このパラメーターを使用します。	なし
DecimalScale	Int	10 進データ型の位取りを指定します。例えば、`unscaledVal` × 10^{-scale} です。	なし
Dummy	Boolean	ダミー列を示します。`True` の場合、ダミー列の値は更新されず、データベースにも挿入されません。このアプリケーション固有情報を使用するのは、 1 つの列に複数の ForeignKey 値を構成したいときです。	なし

FixedChar	Boolean	テーブル内の列が VARCHAR 型ではなく CHAR 型である場合に、属性を固定長とするかどうかを指定します。例えば `true` に設定すると、ある特定の属性が CHAR 型の列にリンクされている場合、アダプターはデータベースを照会するときに、属性値をその属性の最大長までブランクで埋めます。
		
		ビジネス・オブジェクトの XSD ファイルでは、このパラメーターは手動で更新する必要があります。XML またはテキスト・エディターを使用してビジネス・オブジェクトを開いて XSD ファイルを編集し、以下の 2 つの変更を行ってください。
		1. オブジェクト属性の `<element>` タグにデフォルトで追加された `type="string"` を削除します。
		2. 次の例に示すように、`</element>` の前に新規に `<simpletype>` セクションを追加します。
		`<xsd:simpleType>`
		`<xsd:restriction base="xsd:string">`
		`<xsd:maxLength value="10"/>`
		`</xsd:restriction>`
		`</xsd:simpleType>`
		
		オブジェクト定義を保存して、更新後に XSD ファイル内で検証エラーが発生しないことを確認してください。
		
		この表の下にある、『ビジネス・オブジェクトの XSD ファイル内の FixedChar パラメーターの例』のセクションを参照してください。
			`false`

表 7. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
ForeignKey	String	このプロパティーの値は、親子関係が親ビジネス・オブジェクトに格納されるか、子ビジネス・オブジェクトに格納されるかによって異なります。 関係が親に格納される場合、子ビジネス・オブジェクトのタイプと、外部キー *(Child_BO_name/ Child_Property_Name)* として使用される子ビジネス・オブジェクト内の属性の名前の両方がこの値に含まれます。 この関係が子に保管される場合は、外部キーとして使用される親ビジネス・オブジェクト内の属性の名前のみを含むように値を設定します。 属性が外部キーではない場合は、アプリケーション固有情報にこのパラメーターを含めないでください。	なし
OrderBy	String	値が指定され、属性が子ビジネス・オブジェクト内に存在する場合は、アダプターは、検索照会の ORDER BY 文節でその属性の値を使用します。 アダプターは、子ビジネス・オブジェクトを昇順 (ASC) または降順 (DESC) で検索することができます。このパラメーターがアプリケーション固有情報に含まれていない場合、アダプターは検索順序を指定しません。	なし
PrimaryKey	Boolean	この属性に関連付けられている列が、データベース内の対応するテーブルの基本キーである場合、PrimaryKey パラメーターは `True` に設定されます。	なし
SPParameterType	String	ストアード・プロシージャーのタイプを指定します。 使用可能な値は、以下のとおりです。 • `IP` (入力のみ) • `OP` (出力のみ) • `IO` (入出力) • `RS` (結果セット)	なし

表 7. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
UniqueIdentifier (UID)	String	アダプターは、ビジネス・オブジェクトの固有 ID の生成に、このパラメーターを使用します。シーケンスと ID 列の生成がサポートされます (ID 列は、Informix では、*シリアル列* と呼ばれています)。DB2 は、シーケンスと ID 列の両方をサポートします。 ID 列は、データベースでテーブルの各行に自動的に固有の数値を生成する方法を提供します。 ID 列は、DB2 および Microsoft SQL Server の場合に定義でき、Informix の場合はシリアル列が定義できます。 このパラメーターの形式を以下に示します。 `UID=AUTO|`*`Sequence_Name`* DB2 または Microsoft SQL Server データベースのいずれかのテーブルに対してディスカバリー・プロセスを実行する場合、UID (固有 ID) 属性を手動で AUTO に設定する必要があります (例えば、`<UID>AUTO</UID>`)。 **注:** UID (固有 ID) 属性を手動で AUTO に設定する要件は、DB2 および Microsoft SQL Server の ID 列に特有のものです。この要件は、Informix のシリアル列には当てはまりません。Informix の場合、シリアル列の UID 属性は自動生成され、 <UID>SERIAL</UID> または <UID>SERIAL8</UID> のいずれかになります。 ID 列と同様、シーケンスも数値の自動生成に使用されます。データベースによるシーケンスおよび ID 列の使用について詳しくは、ご使用のデータベースの資料を参照してください。 シーケンスの場合、UID 属性にシーケンス名を設定します。シーケンスは、DB2 および Oracle データベースで定義できます。 属性で固有 ID が必要とされない場合は、このパラメーターをアプリケーション固有情報に含めないでください。	なし
XML	Boolean	データベース内のテーブル列が XML 型のものである場合、XML パラメーターは `True` に設定されます。	なし

属性のアプリケーション固有情報の形式は、XSD ファイルの以下の実例セクションに示します。

XSD ファイルのセクション例

```
   <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
   <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
   <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<xsd:simpleType>
```

```
   <xsd:restriction base="xsd:string">
   <xsd:maxLength value="10"/>
   </xsd:restriction>
</xsd:simpleType>
</element>
<element name="custCode" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
   <jdbcasi:ForeignKey>custinfoObj/custCode</jdbcasi:ForeignKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="firstName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>fname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lastName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>lname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
```

ビジネス・オブジェクトの XSD ファイル内の FixedChar パラメーターの例

```
<element name="primaryKey">
<annotation>
<appinfo source="WBI">
        <jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
                <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
                <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
                <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
        </jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<xsd:simpleType>
        <xsd:restriction base="xsd:string">
                <xsd:maxLength value="10"/>
        </xsd:restriction>
</xsd:simpleType>
</element>
```

子ビジネス・オブジェクトを参照する属性のアプリケーション固有情報

子ビジネス・オブジェクトを参照する属性には、2 つのアプリケーション固有情報パラメーターが使用されます (単純属性ではなく複合属性)。このアプリケーション固有情報を設定する場合は、表 8に示すパラメーターを指定します。

表 8. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報

パラメーター	タイプ	説明	デフォルト値
KeepRelationship	Boolean	True の場合、このパラメーターにより Update 操作中に子ビジネス・オブジェクトは削除されません。	なし

表 8. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
Ownership	Boolean	このパラメーターは、子ビジネス・オブジェクトが親によって所有されることを指定します。True の場合、子ビジネス・オブジェクトに対して Create、Update、および Delete 操作が許可されます。False の場合、どの更新も子ビジネス・オブジェクトには適用できません。親が作成されると、データベース内で関係の整合性が保持されるように、子が存在するかどうかが検証されます。	なし

ビジネス・オブジェクトの XSD ファイル内の ownership の例

```
<element minOccurs="0" name="addressObj" type="bons0:OutboundRtasserAddress"
 maxOccurs="unbounded">
  <annotation>
    <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
      <jdbcasi:Ownership>true</jdbcasi:Ownership>
    </jdbcasi:JDBCAttributeTypeMetadata>
    </appinfo>
  </annotation>
</element>


<element minOccurs="0" name="custinfoObj" type="bons1:OutboundRtasserCustinfo"
maxOccurs="1">
 <annotation>
  <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:Ownership>false</jdbcasi:Ownership>
   </jdbcasi:JDBCAttributeTypeMetadata>
  </appinfo>
 </annotation>
</element>
```

単一および複数カーディナリティー子ビジネス・オブジェクトの XSD ファイルのもう 1 つの例

単一または複数カーディナリティー子ビジネス・オブジェクトの XSD 定義ファイルの例をここに示します。エレメント custInfoObj は単一カーディナリティー子ビジネス・オブジェクトで、addressObj は複数カーディナリティー子ビジネス・オブジェクトです。

```
<element name="addressObj" minOccurs="1" type="Address:Address"
maxOccurs="unbounded">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>true</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
                        <element name="custInfoObj" minOccurs="0" type=
"CustInfo:CustInfo" maxOccurs="1">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
```

```
"urn:app:jdbc:asi">
                                <pasi:Ownership>false</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
```

操作のアプリケーション固有情報

アダプターは、操作レベルのアプリケーション固有情報を使用してデータベース内の情報の取得や更新などの操作を実行します。アダプターは、ビジネス・オブジェクトでの指定に従って、SQL 照会、ストアード・プロシージャー、またはストアード関数を使用してデータベース表を取得および更新します。

ビジネス・オブジェクトにストアード・プロシージャーまたはストアード関数を追加する場合、操作レベルのアプリケーション固有情報 (ASI) を、表 9に指定されているとおりに設定します。

表 9. 操作に関するアプリケーション固有情報

操作 ASI の StoredProcedure のパラメーター・エレメント	**ウィザードによって設定**	**説明**
Parameters	はい	ストアード・プロシージャーのパラメーターをリストします。
PropertyName	はい	選択したビジネス・オブジェクト属性の名前に設定します。
ResultSet	いいえ	ストアード・プロシージャーから結果セットが返される場合は、ビジネス・オブジェクト定義でこのパラメーターを `True` に設定します。
ReturnValue	はい	ストアード・プロシージャーに戻り値がある場合は、このパラメーターには次のいずれかの値が設定されます。 • ストリング `RS`。この値は、プロシージャーから結果セットが戻され、この結果セットを使用して、このビジネス・オブジェクトに対応する複数カーディナリティー・コンテナーが作成されることを示します。 • ビジネス・オブジェクト属性の名前。この値は、プロシージャーから戻される値が、実行時にビジネス・オブジェクトの特定の属性に割り当てられることを示します。 属性が別の子ビジネス・オブジェクトである場合、アダプターはエラーを返します。
StoredProcedure	はい	ストアード・プロシージャー名に設定します。
StoredProcedureType	はい	タイプのリストから選択します。 有効なストアード・プロシージャー・タイプについては、ストアード・プロシージャー・タイプを参照してください。

表 9. 操作に関するアプリケーション固有情報 (続き)

操作 ASI の StoredProcedure のパラメーター・エレメント	ウィザードによって設定	説明
タイプ	はい	ストアード・プロシージャーのパラメーターのタイプを設定します。使用可能な値は、以下のとおりです。 • IP (入力のみ) • OP (出力のみ) • IO (入出力) • RS (結果セット)

一般的な問題の解決策

ご使用のデータベースで WebSphere Adapter for JDBC を実行するときに発生する可能性のあるいくつかの問題と、その解決策および回避策を説明します。これらの問題および解決策は、ソフトウェア・サポート Web サイトの技術情報として文書化されている問題や解決策を補足するものです。

WebSphere Adaptersについての技術情報の詳細なリストについては、http://www.ibm.com/support/search.wss?tc=SSMKUK&rs=695&rank=8&dc=DB520+D800+D900+DA900+DA800+DB560&dtm を参照してください。

オブジェクトを選択に追加できない

問題

設計時に、JConnect ドライバーを使用してアダプターで Sybase からストアード・プロシージャーをインポートする場合、WebSphere Adapter for JDBC エンタープライズ・サービスのディスカバリー処理が失敗します。

注: この問題は、jTDS 1.2.2 ドライバーを使用する場合には発生しません。

次のメッセージが生成されます: オブジェクトを選択に追加できません: com.sybase.jdbc2.jdbc.SybSQLException:「CREATE TABLE」は、「tempdb」データベースの複数ステートメント・トランザクション内では許可されていません。

原因

エンタープライズ・サービスのディスカバリー処理において、自動コミットのプロパティー「**データベース接続時に自動コミットを設定**」が選択されておらず、Sybase データベースのストアード・プロシージャーのトランザクション・モードがデフォルト値の「非チェーン・モード」に設定されています。デフォルトの「非チェーン・モード」では、トランザクションを完了させるために、コミット・トランザクションやロールバック・トランザクションと対になった明示的な開始トランザクション・ステートメントが必要です。

解決策

ストアード・プロシージャーの定義を検討して、トランザクションを適切に処理するように変更できるかを判断してください。ストアード・プロシージャーの定義を変更できない場合、外部サービス・ウィザード の「ディスカバリー構成」ウィンドウから「**データベース接続時に自動コミットを設定**」を選択して、ディスカバリー・プロセスを再実行することができます。

「**データベース接続時に自動コミットを設定**」を選択すると、「非チェーン・モード」構成に関連したデフォルトの処理が自動的に指定変更されます。トランザクション・モードが Sybase データベースでどのように機能するかについて詳しくは、Sybase データベースの資料を参照してください。

注: 外部サービス・ウィザード の「ディスカバリー構成」ウィンドウから「**データベース接続時に自動コミットを設定**」を選択する場合、外部サービス・ウィザードの最後の画面でも「**データベース接続時に自動コミットを設定**」を選択する必要があります。この最後の画面の「**データベース接続時に自動コミットを設定**」の値は、実行時にアダプターが、データベースとの outbound 接続インスタンスを作成するために使用する管理接続ファクトリー・プロパティーに適用されます。

Oracle 9i または 10g データベースに 4K 以上の CLOB データ型を挿入できない

問題

4 K 以上の CLOB (文字ラージ・オブジェクト) を Oracle 9i または 10g データベースに挿入しようとすると、次の例外が生成されます。

- Oracle 9i: `ResourceAdapt E com.ibm.j2ca.jdbc.JDBCDBOperationHandler executePreparedCUDStatement CWYBC0301E: データベースでの操作が SQL 例外で失敗しました。失敗の理由はソケットから読み取るデータがありません (No more data to read from socket) です。`
- Oracle 10g: `ResourceAdapt E com.ibm.j2ca.jdbc.JDBCDBOperationHandler executePreparedCUDStatement CWYBC0301E: データベースでの操作が SQL 例外で失敗しました。失敗の理由は ORA-01460: インプリメントされていないまたは不当な変換が要求されました (ORA-01460: unimplemented or unreasonable conversion requested) です。`

原因

4K を超える CLOB を正しくサポートしない古いバージョンのドライバーを使用しています。

解決策

Oracle 10.1.0.2 以降のリリースの Oracle シン・ドライバーを使用してください。

生成された一部のビジネス・オブジェクトには Oracle データベース・オブジェクトの属性がない

問題

Oracle データベース・オブジェクトから生成される一部のビジネス・オブジェクトに、テーブル列の属性がありません。

原因

特定の条件下では、Oracle JDBC ドライバーはデータベース・オブジェクトの列情報を返しません。この問題については、以下のバグを現在 Oracle に提出しています。

- 2281705。シノニムがある場合、DATABASEMETADA.GETCOLUMNS は基盤となるテーブルを返さない
- 2696213。JDBC GETPROCEDURECOLUMNS はプロシージャーのシノニムの列を返さない

また、他のスキーマ内のオブジェクトを参照するプライベート・シノニムが使用されている場合、列情報は返されません。

解決策

シノニムを持つテーブルの場合は、テーブルのシノニムを使用してビジネス・オブジェクトを生成します。

プロシージャーのシノニムの場合は、シノニムの基盤となるオリジナルのプロシージャーを使用してビジネス・オブジェクトを生成します。

他のスキーマ内のオブジェクトを参照するプライベート・シノニムの場合は、オリジナルのテーブルを使用するか、現在のスキーマにシノニムを作成します。

アダプターを使用して JDBC (タイプ 2 またはタイプ 4) ユニバーサル・ドライバーにより IBM DB2 for z/OS® に接続する

問題と原因

DB2 for z/OS は、位置インデックスをデフォルトで使用し、列名を使用しないことによって、Adapter for JDBCが使用するストアード・プロシージャー・メタデータの照会をサポートします。解決策では、z/OS プラットフォームで DB2 と共に Adapter for JDBC を使用する手順を示しています。

解決策

Adapter for JDBCを使用して DB2 for z/OS に接続するには、以下の接続要件が満たされていることを確認してください。

- ユニバーサル JDBC ドライバーの物理表現は、db2jcc.jar ファイルです。このファイルへのパスが、クラス・パスに設定されていることを確認してください。
- データベース URL: タイプ 2 またはタイプ 4 のどちらのドライバーを使用するかを決定するには、接続の形式を検討します。

  タイプ 2: jdbc:db2:database
  (例: jdbc:db2:MyDB。MyDB はデータベース名。)

  タイプ 4: jdbc:db2://server:port/database
  (例: jdbc:db2://9.182.15.129:50000/MyDB。MyDB はデータベース名。)
- ドライバー・クラス: com.ibm.db2.jcc.DB2Driver。

タイプ 2 ドライバーとタイプ 4 ドライバーの両方で、ドライバー・クラスは同一です。

- クラス・パスに db2jcc_license_cisuz.jar ファイルのパスを設定します。

  タイプ 2 ドライバーとタイプ 4 ドライバーの両方で、ライセンス JAR ファイルは同一です。DB2 for z/OS サーバーおよび DB2 for i5/OS サーバーにアクセスするには、有効な DB2 Connect™ のライセンスが必要です。DB2 クライアントは、DB2 Connect ライセンスがなければ、zSeries® サーバーおよび iSeries® サーバーへの直接接続を提供しません。

  DB2 Connect のライセンス交付および使用法について詳しくは、以下のページを参照してください。
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0303zikopoulos1/0303zikopoulos1.html
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0301zikopoulos/0301zikopoulos.html

ウィザードを使用することによるストアード・プロシージャーのメタデータのインポートでは、問題が発生する可能性があります。Adapter for JDBC を使用して、ストアード・プロシージャーを使用し、DB2 からメタデータをインポートするには、DB2 を以下のステップで説明するように再構成する必要があります。前述のステップに加えて、以下のステップを実行してください。

- DB2 に APAR の PQ62695、PQ55393、PQ56616、PQ54605、PQ46183、および PQ62139 を適用します。
- アダプターでストアード・プロシージャーを使用したい場合は、下記のステップを実行します。これは、PQ62695 のフィックスの一部です。このフィックスでは、JDBC および ODBC 仕様で文書化されているスキーマ・メタデータ API に対応する結果セットを生成することが可能なストアード・プロシージャーが導入されています。

  これらのプロシージャーは、DB2 Universal Driver で提供される JDBC ドライバーおよび ODBC ドライバーによって使用されます。以下のステップを実行して、ストアード・プロシージャーのサポートを使用可能にします。

  1. APAR を適用します。
  2. ZPARM アセンブリー・ジョブ DSNTIJUZ 内の DESCSTAT 変数の値を確認します。 DESCSTAT 変数の値が NO である場合は、YES に変更します。

     注: DESCSTAT のデフォルトは、V7 では NO ですが、V8 では YES に変更されました。
  3. ZPARM モジュールを再アセンブルし、再初期化します。
  4. DSNTIJMS という名前の JCL ジョブを実行します。このメンバーは、db2prefix.SDSNSAMP データ・セット内にあります。
  5. DB2 を再始動します。

リモート DB2 データベースを用いた outbound サポートのための XA トランザクションの使用

Universal Driver を使用した Adapter for JDBC での XA トランザクションの使用

Adapter for JDBC および Universal ドライバーで XA トランザクションを使用してリモート DB2 データベースに接続するには、以下のバージョンのソフトウェアおよび構成プロパティーが必要です。

- DB2 バージョン: 8.2 以降
- JDBC ドライバー: UDB ドライバーのタイプ 4 およびタイプ 2
- XA データ・ソース名: com.ibm.db2.jcc.DB2XADataSource
- XA データベース名: これは、ローカル DB2 クライアントで構成されたリモート・データベース別名です。
- データベース URL: jdbc:db2://hostname:port/databasename
- JDBC ドライバー・クラス: com.ibm.db2.jcc.DB2Driver

リモート DB2 データベースを用いた Outbound サポートのための XA トランザクションの使用

リモート DB2 データベースを使用する WebSphere Adapter for JDBC 用の XA サポートの構成要件について、以下に示します。

リモート DB2 データベースでの XA トランザクションの使用

リモート DB2 データベースの追加

1. DB2 サーバーで db2admin (*DB2_InstallPath*¥SQLLIB¥BIN) コマンドを実行します。
2. DB2 構成アシスタントを開きます。
3. 「**表示 (View)**」 → 「**拡張表示 (Advanced View)**」に移動します。

次のステップを順番に実行してください。

1. **リモート・システムの追加**
   a. 「**システム**」タブを選択します。
   b. メニューから、「**選択済み**」 → 「**システムの追加 (Add System)**」を選択します。
   c. 「**システム名 (System name)**」フィールドで、ターゲット・データベースが配置されている物理マシン、サーバーシステム、またはワークステーションを指定します。サーバー・システム上のシステム名は、DB2SYSTEM DAS 構成パラメーターによって定義されます。ユーザーはこの値を使用する必要があります。
   d. 「**ホスト名**」フィールドに、ホスト名か、またはターゲット・データベースが存在するインターネット・プロトコル (IP) アドレスを入力します。
   e. 「**ノード名 (Node name)**」フィールドで、データベースが配置されているリモート・ノードのローカル・ニックネームを指定します。選択するノード名は、ノード・ディレクトリーまたは管理ノード・ディレクトリーに存在していないものにする必要があります。
   f. オペレーティング・システムを選択して、「**OK**」をクリックします。

2. **インスタンス・ノードの追加**
   a. 「**インスタンス・ノード (Instance Nodes)**」タブを選択します。
   b. メニューから、「**選択済み**」 → 「**インスタンス・ノードの追加 (Add Instance Node)**」を選択します。
   c. 「**システム名 (System name)**」フィールドで、ターゲット・データベースが配置されている物理マシン、サーバーシステム、またはワークステーションを指定します。リモート・システムの追加タスクで追加したシステムを選択します。
   d. 「**インスタンス名**」フィールドに、ターゲット・データベースが配置されているインスタンスの名前 (DB2 など) を入力します。
   e. 「**インスタンス・ノード名 (Instance node name)**」フィールドで、データベースが配置されているカタログされたシステム (ノード) の固有のニックネームを指定します。選択するノード名は、ノード・ディレクトリーまたは管理ノード・ディレクトリーに存在していないものにする必要があります。
   f. オペレーティング・システムを選択して、ホスト名を入力します。リモート・システムの追加タスクのステップ 4 と同じホスト名を使用します。
   g. リモート DB2 インスタンスが実行されているポート番号を入力します。
   h. 「**OK**」をクリックします。
3. **データベースの追加**
   a. 「**データベース**」タブを選択します。
   b. メニューから、「**選択済み**」 → 「**データベースの追加**」を選択します。
   c. 「**インスタンス・ノード (Instance node)**」フィールドで、インスタンス・ノードの追加タスクで作成したインスタンスを選択します。「**データベース名 (Database name)**」フィールドで追加するデータベースの名前を指定します。
   d. 「**別名 (Alias)**」フィールドで、ワークステーション上で実行中のアプリケーションが使用できるローカル・ニックネームを指定します。何も入力されない場合、別名はデータベース名と同じになります。別名は、固有でなければなりません。

      注: この別名値を、アダプターの XADatabaseName プロパティーに入力する必要があります。
4. **データベース接続のテスト**
   a. 「**データベース**」タブを選択します。
   b. データベースの追加タスクで追加されたデータベースを選択します。
   c. メニューから、「**選択済み**」 → 「**接続のテスト**」を選択します。
   d. 「**CLI**」チェック・ボックスを選択し、ユーザー ID とパスワードを入力して、「**接続のテスト**」をクリックします。これにより、接続の成功が戻されます。

イベント・テーブルのトランザクション (XID) 列を調べる

アダプターが送達は 1 回のみとして構成されている場合は、XID 列と共に状況列を使用して、イベントが処理されたかどうかを判別します。

- XID 列に 0 が含まれている場合、イベントはまだ処理対象として選出されていません。

- XID 列にトランザクション ID が含まれている (つまり 0 ではない) 場合、アダプターはイベントの処理を開始しましたが、処理は完了していません。イベントの処理中にアダプターまたはアプリケーション・サーバーが異常終了した場合は、この組み合わせが発生することがあります。トランザクション・マネージャーはリカバリー時にこれらのトランザクションを COMMIT または ROLLBACK します。

照会 SQL ステートメントからの予期しない結果の処理

照会から予期しない結果を受け取った場合は、トレースをオンにして、照会 SQL をログで確認してください。トレースをオンにすると、テスト・クライアントの場合に、不必要な属性をすべて*設定解除* し忘れていないかを確認できるため、特に便利です。また、入力ビジネス・オブジェクトが正確に入力されているかどうかを判断する場合にも、トレースをオンにすると効果的です。

XA トランザクションをサポートする SQL Server 2000 の構成

XA トランザクションをサポートするように SQL SERVER 2000 を構成するには、以下のようにします。

1. `Microsoft SQL Server 2000 Driver for JDBC¥SQLServer JTA¥` からパス *`sqlserver_install_directory`*`¥MSSQL¥Binn` に sqljdbc.dll をコピーします。
2. SQL クエリ アナライザーを開き、`Microsoft SQL Server 2000 Driver for JDBC¥SQLServer JTA¥instjdbc.sql` を実行します。

WebSphere Adapter for JDBC が SQL Server 2000 JDBC ドライバーによる SQL Server 2000 への接続に失敗する

原因

これは、SQL Server 2000 JDBC ドライバーでの制限です。

解決策

以下のいずれかの方法を使用してこの問題を解決できます。

- 値 `SelectMethod=Cursor` をデータベース URL プロパティーに付加する。例: jdbc:microsoft:sqlserver://127.0.0.1:1433;DatabaseName=Partner;SelectMethod=Cursor
- SQL Server 2005 JDBC ドライバーを使用する。 SQL Server 2005 JDBC ドライバーは、SQL Server 2000 および SQL Server 2005 の両方へのアクセスを提供します。詳しくは、Microsoft のサポート Web サイト (http://msdn.microsoft.com/en-us/data/aa937724.aspx) を参照してください。

複数の JDBC アダプター・エクスポート・コンポーネントが同一の SCA モジュールにある

複数の JDBC アダプター・エクスポート・コンポーネントが同一の SCA モジュールにある場合、これらのエクスポートでは、同じテーブルから同じイベント・レコードは取り出されず、そのレコードに対する操作は行われません。複数のエクスポートを定義して同一のイベント・レコードにアクセスする場合は、データベース・デッドロック・エラーなどの潜在的な問題が発生する可能性があります。複数の JDBC アダプター・エクスポート・コンポーネントが同一のイベント・テーブルか

らレコードを取り出すためには、エクスポートごとに異なる EventTypeFilter (処理するイベント・タイプ) 条件を構成して、1 つのイベント・レコードが複数のエクスポートによって取り出されるリスクを回避する必要があります。

iSeries DB2 において異なる固有の名前があるストアード・プロシージャーに対して正しいビジネス・オブジェクトを作成できない

原因

DB2 iSeries JDBC ドライバー Toolbox for Java™ & JTOpen では、接続プロパティー・ソース「metadata source」を使用して、データベースからの DatabaseMetaData の取得方法が指定されます。これが「0」に設定されている場合、メタデータはオブジェクト情報取得 (ROI) データ・フローを使用して取得されます。これが「1」に設定されている場合、メタデータはシステム・ストアード・プロシージャーを呼び出すことにより取得されます。バージョン 6.1 以前の場合、デフォルトでは、metadata source は 1 に設定されていないため、データベース・メタデータはオブジェクト情報取得 (ROI) データ・フローを使用して取得されます。これが原因で問題が発生します。

DB2 iSeries バージョン 7.1 のデフォルトでは、ストアード・プロシージャーを呼び出すことによってデータベース・メタデータを取得します。

JTOpen ダウンロード Web サイト (http://sourceforge.net/projects/jt400/files/JTOpen-full/6.4/) から入手できる `jtopen_6_4_source¥com¥ibm¥as400¥access¥doc-files¥JDBCProperties.html` に、データベース・メタデータに関する詳細情報が記載されています。

解決策

metadata `source=1` を接続 URL に追加し、DB2 iSeries バージョン 6.1 以前への接続時には「;」の後にスペースがないようにします。例: jdbc:as400://wsbcas12.rtp.raleigh.ibm.com;metadata source=1

成果物を再生成せずにスキーマを変更する

スキーマのみが変更され、表名は変更されていない場合は、以下に説明されているように、Inbound 用の *.export ファイルまたは Outbound 処理用の *.import ファイルと、*.xsd ファイルを編集し、元のスキーマを新規のスキーマに変更します。

import ファイルまたは export ファイルを変更するには、以下のようにします。

1. Java パースペクティブにおいて、テキスト・エディターで *.import ファイルまたは *.export ファイルを開きます。例えば、 JDBCInboundInterface.export などです。

   以下は、JDBCInboundInterface.export のコード・スニペットです。

```
<connection type="com.ibm.j2ca.jdbc.inbound.JDBCActivationSpecWithXid"
listenerType="com.ibm.j2ca.base.ExtendedInboundListener"
selectorType=
"com.ibm.j2ca.extension.emd.runtime.StructuredDataFunctionSelector">
      <properties>
        <databaseURL>jdbc:oracle:thin:@localhost:1521:ORA92
</databaseURL>
        <databaseVendor>ORACLE</databaseVendor>
        <jdbcDriverClass>oracle.jdbc.driver.OracleDriver
```

```
</jdbcDriverClass>
        <password>jcajdbc</password>
        <returnDummyBOForSP>false</returnDummyBOForSP>
        <userName>jcajdbc</userName>
</properties>
    </connection>
```

2. 既存のユーザー名とパスワードを、新規のスキーマで使用されるユーザー名とパスワードに変更します。

*.xsd ファイルを変更するには、以下のようにします。

1. テキスト・エディターで *.xsd ファイルを開きます。例えば、JcajdbcCustomer.xsd などです。

   以下は、 JcajdbcCustomer.xsd のコード・スニペットです。

```
<jdbcasi:TableName>JCAJDBC.CUSTOMER</jdbcasi:TableName>
<jdbcasi:Operation>
<jdbcasi:Name>Retrieve</jdbcasi:Name>
<jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>BeforeRetrieveSP
</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>JCAJDBC.fn_beforeRetrievesSP
</jdbcasi:StoredProcedureName>
<jdbcasi:ReturnValue>RS</jdbcasi:ReturnValue>
<jdbcasi:Parameters>
<jdbcasi:Type>IP</jdbcasi:Type>
<jdbcasi:PropertyName>pkey</jdbcasi:PropertyName>
</jdbcasi:Parameters>
</jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>AfterRetrieveSP
</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>JCAJDBC.sp_afterRetrieveSP
</jdbcasi:StoredProcedureName>
<jdbcasi:Parameters>
<jdbcasi:Type>IP</jdbcasi:Type>
<jdbcasi:PropertyName>fname</jdbcasi:PropertyName>
</jdbcasi:Parameters>
<jdbcasi:Parameters>
<jdbcasi:Type>OP</jdbcasi:Type>
<jdbcasi:PropertyName>RS</jdbcasi:PropertyName>
</jdbcasi:Parameters>
</jdbcasi:StoredProcedures>
</jdbcasi:Operation>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
```

2. 表およびプロシージャーの名前のスキーマ名の接頭部 (BO で定義した場合) を新規のスキーマ名に更新します。例えば、JCAJDBC.CUSTOMER は <NewSchema>.CUSTOMER に変更し、JCAJDBC.fn_beforeRetrievesSP は <NewSchema>.fn_beforeRetrievesSP に変更し、JCAJDBC.sp_afterRetrieveSP は <NewSchema>.sp_afterRetrieveSP に変更します。

ストアード・プロシージャーの検証時に、結果セットに値がまったく含まれていない

問題

JDBC アダプターが MS SQL Server 2000 でストアード・プロシージャーを呼び出すように構成されている場合、「構文を検証... (validate the syntax...)」チェック・ボックスを選択した後で、ウィザードによって結果セットから値が返されません。

解決策

AutoCommit ='true' に設定するか、またはドライバーを SQL Server バージョン 2005 にアップグレードしてください。

MS SQL Server 用のストアード・プロシージャー名の BIDI サポート

問題

MS SQL Server では、ストアード・プロシージャー名が右から左に書かれる言語で記述されている場合、ストアード・プロシージャーのビジネス・オブジェクト構成ウィンドウが表示されません。ログ・ファイルに例外「java.lang.RuntimeException: com.microsoft.sqlserver.jdbc.SQLServerException: nvarchar 値「??」をデータ型 int に変換する際に変換に失敗しました (java.lang.RuntimeException: com.microsoft.sqlserver.jdbc.SQLServerException: Conversion failed when converting the nvarchar value '??' to data type int)」が表示されます。EMD ウィザードがこの例外を無視して完了した場合、対応するストアード・プロシージャーのビジネス・オブジェクトは生成されません。

解決策

ストアード・プロシージャー名は、左から右に書かれる言語で記述する必要があります。ストアード・プロシージャー名を表す言語を変更してください。

Oracle データベースのスキーマ名 BIDI サポート

問題

Oracle データベースでは、スキーマ名が右から左に書かれる言語で記述されている場合、ストアード・プロシージャーの検証が失敗します。ログ・ファイルに例外「java.sql.SQLException: ORA-06550: 行 xx、列 xx: PLS-00302: コンポーネント「xxx」を宣言する必要があります (java.sql.SQLException: ORA-06550: line xx, column xx: PLS-00302: component 'xxx' must be declared)」が表示されます。EMD ウィザードがこの例外を無視して完了した場合、対応するストアード・プロシージャーのビジネス・オブジェクトは生成されます。ただし、対応する Outbound 操作が処理されるときに、例外が生成されます。

解決策

スキーマ名は、左から右に書かれる言語で記述する必要があります。スキーマ名を表す言語を変更してください。

テーブル名または列名に SQL キーワードまたはその他の特殊文字が含まれるときに WebSphere Adapter for JDBC が失敗する

問題

テーブル名または列名に、スペース、単一引用符、SQL 予約キーワードなどの特殊文字が含まれていると、WebSphere Adapter for JDBC が失敗します。

原因

テーブル名に特殊文字が含まれるものがあり、例えば、Oracle サーバーでは、「TABLE」、「TABLE 2」、「table」、「table 2」、「'TABLE'」、「'TABLE 2'」などです。または、テーブル内の列に特殊文字が含まれるものがあり、例えば、「DESC」、「DESC 2」、「desc」、「desc 2」、「'COLUMN'」、「'COLUMN 2'」などです。

解決策

用語「TABLE」および「DESC」は SQL 予約キーワードです。これらのテーブル名または列名は SQL ステートメントの処理中に正しく扱われないため、 TableName または ColumnName のアプリケーション固有情報を編集して、テーブル名または列を二重引用符で囲む必要があります。

以下に、SQL キーワードおよび特殊文字 (スペース) を含むテーブル名および列名の例を示します。

```
<jdbcasi:TableName>YUANJS.table 2</jdbcasi:TableName>
<jdbcasi:ColumnName>DESC</jdbcasi:ColumnName>
```

テーブル名および列名を二重引用符で囲むように ASI を編集します。

```
<jdbcasi:TableName>"YUANJS.table 2"</jdbcasi:TableName>
<jdbcasi:ColumnName>"DESC"</jdbcasi:ColumnName>
```

WebSphere Adapter for JDBC および MySQL データベースのサポート

メタデータ・ディスカバリー中に、JDBC エンタープライズ・メタデータ・ディスカバリー (EMD) はまずデータベース固有の JDBC ドライバーからスキーマ情報を取得し、その後でスキーマの選択に応じて、他のデータベース・エンティティー (テーブル、プロシージャー、ビュー、シノニムなど) に関する情報を取得します。しかし、MySQL の場合、JDBC ドライバーはスキーマ情報を返しません。これは、MySQL の JDBC ドライバーの実装環境の場合、バージョン 5.0.4 より前のバージョンではデータベース・スキーマに関する情報を返さないためです。したがって、ディスカバリーおよび成果物の生成をそれ以上進めることはできません。

i5/OS でのテーブルおよびトリガーの生成用のサンプル SQL スクリプト

問題

JDBC J2EE コネクター・アーキテクチャー (JCA) のリソース・アダプターには、scripts_db2.sql という DB2 用のサンプル・スクリプトが付属しています。i5/OS 上の UDB では、 DB2 サンプル・スクリプトの SQL 構文が記述どおりに機能しません。

原因

UDB には、 SCHEMA.TABLE 形式の完全修飾スキーマ SQL が必要です。現在のスクリプトには、修飾されたスキーマは含まれません。

解決策

i5/OS で使用するために、テーブル・スクリプトとトリガー・スクリプトのテーブル名とトリガー名に完全修飾スキーマ名を含めるように scripts_db2.sql を変更します。例として、scripts_db2.sql からのテーブル作成スクリプトのサンプルを以下に示します。

```
CREATE TABLE customer
  (
   pkey VARCHAR(10) NOT NULL PRIMARY KEY,
   fname VARCHAR(20),
   lname VARCHAR(20),
   ccode VARCHAR(10)
);
```

i5/OS で使用するために、以下に示すように、スクリプトを変更して <schema_name> を完全修飾スキーマ名に置き換えます。

```
CREATE TABLE <schema_name>.customer
(
  pkey VARCHAR(10) NOT NULL PRIMARY KEY,
  fname VARCHAR(20),
  lname VARCHAR(20),
  ccode VARCHAR(10)
);
```

WebSphere Adapter for JDBC がテーブル内の行をロックしてしまい、それによって他のアプリケーションがタイムアウトする

問題

WebSphere Adapter for JDBC は、グローバル・トランザクションの一部として組み込まれていて、データベース・テーブル内の共有データにアクセスしていました。データベース・テーブル内のこの共有データには、他のアプリケーションもアクセスしていました。グローバル・トランザクションの一環として、WebSphere Adapter for JDBC は行をロックしたままにしていたため、他のアプリケーションが同じ共有データにアクセスしようとするとタイムアウトが発生しました。

解決策

共有データにアクセスする場合は、WebSphere Adapter for JDBC をグローバル・トランザクションから除去します。その代わりに、ビジネス・レベルで何らかの手段を使用して、ビジネス・データの整合性が保たれるようにしてください。例えば、共有データへの操作が成功したかどうかを示し、このデータへの重要な操作をログに記録する変数を使用します。

誤ったバイナリー・データが返される (照会 DB2/AS400)

問題

WebSphere Adapter for JDBC バージョン 6.1 を使用して DB2/AS400 に対する照会を実行すると、i5/OS ではバイナリー・フィールドが EBCDIC 文字として返されます。例えば、フィールドに 9910001761 ではなく 4040f9f9f1f0f0f0f1f7f6f1 が返されます。AS400 のフィールドは、 BINCHAR として定義されていて、CCSID 65535 のタグが付けられています。

原因

i5/OS データベースのフィールドには、 CCSID 65535 のタグが付けられています。Toolbox の JDBC ドライバーは、この CCSID を変換すべきでないフィールドとして認識します。

解決策

変換したいフィールドに、有効な CCSID のタグを付けます。あるいは、「translate binary」接続プロパティーを「true」に設定することもできます。これは、CCSID 65535 のタグが付けられたフィールドを含む、すべてのフィールドを変換するように JDBC ドライバーに指示するものです。これを簡単に行う方法としては、データベースへの接続時に使用される URL の終わりに「;translate binary=true」を追加します。

戻り値のあるストアード・プロシージャーの使用時のエラー

問題

処理するストアード・プロシージャーに戻り値がある場合、WebSphere Adapter for JDBC は失敗します。生成されるエラー・メッセージは、「プロシージャー 'GetPolicyCount' はパラメーター '@count' を予測していましたが、提供されませんでした。(Procedure 'GetPolicyCount' expects parameter '@count', which was not supplied.)」のようなものになります。

原因

アダプターは現在、戻り値があるストアード・プロシージャーの処理をサポートしていません。以下の例は、アダプターでサポートされないことが原因でエラーを引き起こしたエンタープライズ・メタデータ・ディスカバリーで生成された XSD 内の行を強調表示しています。

次の例では、状況値を返すストアード・プロシージャーを使用するときに、生成される XSD 内に、戻り値に対する以下のような参照 (太字で示しています) があります。

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/pocusergetpolicycount"
xmlns:pocusergetpolicycount=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/pocusergetpolicycount"
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi"
asiNSURI="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="PocuserGetpolicycount">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:TableName>POCUser.GetPolicyCount</jdbcasi:TableName>
<jdbcasi:StatusColumnName>returnvalue</jdbcasi:StatusColumnName>
<jdbcasi:StatusValue></jdbcasi:StatusValue>
<jdbcasi:Operation>
```

```
<jdbcasi:Name>Retrieve</jdbcasi:Name>
<jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>RetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>GetPolicyCount</jdbcasi:StoredProcedureName>
<jdbcasi:Parameters>
<jdbcasi:Type>OP</jdbcasi:Type>
<jdbcasi:PropertyName>returnvalue</jdbcasi:PropertyName>
</jdbcasi:Parameters>
<jdbcasi:Parameters>
<jdbcasi:Type>IP</jdbcasi:Type>
<jdbcasi:PropertyName>ipolicynum</jdbcasi:PropertyName>
</jdbcasi:Parameters>
</jdbcasi:StoredProcedures>
</jdbcasi:Operation>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="returnvalue" type="int" minOccurs="1" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>@RETURN_VALUE</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="ipolicynum" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>@iPolicyNum</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

解決策

ストアード・プロシージャーの実行をエラーなしで処理するために、戻り値へのすべての参照 (上記の太字で示したもの) を XSD ファイルから削除します。上記の例の XSD を変更したものは、以下のようになっているはずです。

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/pocusergetpolicycount"
xmlns:pocusergetpolicycount=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/pocusergetpolicycount"
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
 schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi"
asiNSURI="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="PocuserGetpolicycount">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
```

```
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:TableName>POCUser.GetPolicyCount</jdbcasi:TableName>
<jdbcasi:StatusColumnName>returnvalue</jdbcasi:StatusColumnName>
<jdbcasi:StatusValue></jdbcasi:StatusValue>
<jdbcasi:Operation>
<jdbcasi:Name>Retrieve</jdbcasi:Name>
<jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>RetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>GetPolicyCount</jdbcasi:StoredProcedureName>
<jdbcasi:Parameters>
<jdbcasi:Type>IP</jdbcasi:Type>
<jdbcasi:PropertyName>ipolicynum</jdbcasi:PropertyName>
</jdbcasi:Parameters>
</jdbcasi:StoredProcedures>
</jdbcasi:Operation>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="ipolicynum" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>@iPolicyNum</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

外部サービス・ウィザードの始動時にクラス・ローダー違反が発生する

問題

データ・パースペクティブでデータベースへの接続を使用した後に、外部サービス・ウィザードを使用することができません。ウィザードの 2 番目のパネルの最後に、例外

com.ibm.adapter.framework.api.ImportException が生成されます。
理由: pc:0 でクラス・ロードの制約に違反がありました
(クラス: oracle/jdbc/driver/OracleConnection
メソッド: getWrapper()Loracle/jdbc/OracleConnection;)。

このエラーは、以下の状態のどちらでも発生します。

- 外部サービス・ウィザードを通じてデータベースへの接続を確立すると、データ・パースペクティブからデータベースに接続しようとしたときにエラーが発生します。
- データ・パースペクティブを通じてデータベースへの接続を確立すると、外部サービス・ウィザードからデータベースに接続しようとしたときにエラーが発生します。

原因

データ・パースペクティブとウィザードは独自のクラス・ローダーを使用するため、エラーが発生します。DLL (JDBC ドライバーが使用するネイティブ・ライブラ

リー) は、データ・パースペクティブにいったんロードすると、ウィザードに再度ロードできなくなります。JVM には、どの時点でも 1 つのクラス・ローダーのみがネイティブ・ライブラリーをロード可能であるという、固有の制約事項があります。そのため、クラス・ローダー A が DLL B をロードした場合、クラス・ローダー A が解放され、ガーベッジ・コレクションが実行されるまで、その他のクラス・ローダーは DLL B をロードできません。実際にはユーザーはガーベッジ・コレクションを制御できないため、通常、別のクラス・ローダーで DLL B をロードする場合は、JVM を再始動する必要があるということになります。この制限は、既知のものであり、WebSphere Application Server では文書化されています。

解決策

このエラーが発生したときは、WebSphere Integration Developer を再始動することが唯一の解決策です。

Oracle 10g と共に XA を使用すると接続のクローズのエラーが発生する

問題

Oracle 10g を使用して XA トランザクションを実行するために Adapter for JDBC が使用されると、アダプターは接続のクローズの例外「javax.resource.ResourceException: Closed Connection」を戻します。

原因

これは、Oracle 10g データベース・ドライバーの既知の問題です。この問題については、Oracle で「3488761 Connection closed error from OracleConnection.getConnection() - 10G drivers」というバグが記録されています。

解決策

このバグは、Oracle 10g リリース 2 ドライバーで修正されています。予備手段としては、Oracle 9i JDBC Thin ドライバーを使用して、XA トランザクションのためにデータベースに接続します。

Oracle でのトランザクションの開始中のエラー

問題

Adapter for JDBCを使用して、Oracle データベースを使用する XA トランザクションを実行すると、次のエラーが生成されます。(WTRN0078E: トランザクション・マネージャーがトランザクションのリソースで start を呼び出そうとして、エラーが発生しました。エラー・コードは XAER_RMERR でした。(WTRN0078E: An attempt by the transaction manager to call start on a transactional resource has resulted in an error。The error code was XAER_RMERR.))

解決策

Oracle ディレクトリーに含まれているスクリプト initxa.sql および initjvm.sql を実行します。

```
<ORACLE_HOME>javavm¥install
file: initxa.sql
file: initjvm.sql
```

これらのスクリプトの実行に必要なアクセス権を取得するためには SYSOPER または SYSDBA として Oracle にログインする必要があるため、このアクティビティーは Oracle データベースの管理者によって実行される必要があると考えられます。

initxa.sql スクリプトは、データベースを XA 用に構成します。正常に実行されると、データベースは XA 向けに構成されます。このスクリプトは、一度目の試行で正常に実行される場合もあります。データベースのメモリー・スペースが小さすぎると、正常に実行されないことがあります。

これを修正するには、initjvm.sql スクリプトを実行します。これにより、調整が必要なパラメーターが示されます。パラメーターは、次のファイルに格納されます。

```
<ORACLE_HOME>¥database
file: init<DATABASE_SID>.ora
```

表 10に、通常増加させる必要のある 2 つのパラメーターを示します。ご使用の特定のデータベース構成では、異なるパラメーターの調整が必要となる可能性があります。

表 10. 標準的なパラメーター・サイズ

パラメーター名	最小値
java_pool_size	12000000
shared_pool_size	24000000

Outbound 処理中に ResourceException が発生する

ResourceException を受け取った場合は、根本原因のフィールドを調べて原因を判別します。よくある問題の根本原因を以下に示します。

- SQLException 例外

  SQLException にテキスト `UserID or password is invalid` が含まれている場合は、Outbound 接続に指定されたユーザー ID またはパスワードが正しくありません。

  例えば、次のようになります。

  ```
  javax.resource.ResourceException: [ibm][db2][jcc][t4][2013][11249] Connection
  authorization failure occurred.  Reason: User ID or Password invalid.
  ```

- ConnectException 例外

  ConnectException に、`connection refused` または `could not establish connection to the server` というテキストが含まれている場合は、データベース・サーバーが作動可能ではないか、または接続を妨げるネットワーク問題が発生している可能性があります。

  例えば、次のようになります。

  ```
  javax.resource.ResourceException: [ibm][db2][jcc][t4][2043][11550] Exception
  java.net.ConnectException: Error opening socket to server /9.26.237.55 on port
  50,000 with message: Connection refused: connect.
  ```

Inbound 処理中に ResourceException が発生する

この例外は、データベース接続時に繰り返される問題があることを示します。イベントをポーリングするには、アダプターはデータベースに接続する必要があります。接続が失敗した場合、アダプターは、構成済みの一定の時間を待機した後、データベースへの接続を再試行します。アダプターは、構成された回数だけ接続を試行した後、ポーリングを停止します。アダプターは、ポーリングを停止した場合、ResourceException を戻します。

テスト・クライアントで RetrieveAll 操作または Retrieve 操作を処理していると RecordNotFoundException が生成される

問題

WebSphere Integration Developer テスト・クライアントで RetrieveAll 操作または Retrieve 操作を処理するときに、WHERE 条件 (SELECT ステートメント内) において不要な属性がブランクのままであると、RetrieveAll 操作または Retrieve 操作は失敗し、「`RecordNotFoundException: Record not found in EIS`」という例外が生成されることがあります。

解決策

テスト・クライアントで、必要とされる属性の値を `<unset>` に設定します。RetrieveAll 操作を処理します。再び例外が生成される場合は、データベース表に一致するレコードが存在しないと考えられます。

WebSphere Adapter for JDBC が RecordNotFoundException をスローする

問題

アダプターがデータベースからデータを取得しようとしたときに、要求されたキーの項目がない場合、アダプターは、空またはヌルのオブジェクトを返すのではなく、「5/13/09 12:28:29:332 GST com.ibm.j2ca.base.exceptions.RecordNotFoundException」というような例外をスローします。

原因

RetrieveAll 操作を処理するときに、データベースから返すレコードがないと、アダプターは RecordNotFoundException を生成します。WebSphere Adapter for JDBC では、空の項目は正しく処理されません。

問題

IBM サポートに連絡して、WebSphere Adapter for JDBC の暫定フィックス V6.0.2.3IF10 を取得してください。このフィックスでは、管理接続ファクトリーに errorOnEmptyResultset という新規プロパティーが追加されています。このプロパティーのデフォルト値は、「true」で、RetrieveAll 操作に対して行が返されないと、RecordNotFoundException がスローされます。

RetrieveAll 操作の動作をオーバーライドするには、アプリケーションのデプロイ後に、管理コンソールを使用して errorOnEmptyResultset プロパティーの値を変更し、「False」に設定することができます。レコードが検出されないと、RetrieveAll 操作の処理後にアダプターは空のコンテナーを返します。このプロパティーは、このフィックスの EMD では構成できないため、管理コンソールで変更する必要があります。

以下に示すように、「errorOnEmptyResultset」プロパティーを *.import ファイルに追加することもできます。

```
<connection type="com.ibm.j2ca.jdbc.JDBCManagedConnectionFactory"
interactionType="com.ibm.j2ca.jdbc.JDBCInteractionSpec">
<properties>
<databaseURL>
jdbc:microsoft:sqlserver://localhost:1433;DatabaseName=adapter
</databaseURL>
<jdbcDriverClass>
com.microsoft.jdbc.sqlserver.SQLServerDriver</jdbcDriverClass>
<password>adrienne</password>
<userName>sa</userName>
<errorOnEmptyResultset>false</errorOnEmptyResultset>
</properties>
```

データの取得中にレコードが検出されない場合に JDBC JCA アダプター 6.x が例外をスローする

69 ページの『WebSphere Adapter for JDBC が RecordNotFoundException をスローする』 を参照してください。

ストアード・プロシージャーの ResultSet ASI に、結果セットを返すための手操作による介入が必要である

問題

WebSphere Adapter for JDBC エンタープライズ・メタデータ・ディスカバリー・プロセスは、結果セットを返し、動詞 ASI としてビジネス・オブジェクトに添付されるストアード・プロシージャーについては、ResultSet ASI を設定しません。

原因

Oracle データベースだけが、出力パラメーターとして結果セットを返します。その他のすべてのデータベースについては、ストアード・プロシージャーが結果セットを返すかどうかを判別する方法はありません。

解決策

結果セットを返し、動詞 ASI としてビジネス・オブジェクトに添付されるすべてのストアード・プロシージャーに対して、手動で ResultSet ASI を true に設定し、ストアード・プロシージャーが結果セットを返すように指示してください。

次の例で、ビジネス・オブジェクト内で ASI を設定する方法を示します。

```
<jdbcasi:Operation>
<jdbcasi:Name>RetrieveAll</jdbcasi:Name>
<jdbcasi:StoredProcedures>
```

```
<jdbcasi:StoredProcedureType>RetrieveAllSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>SCOTT.GETCUSTS1</jdbcasi:StoredProcedureName>
<jdbcasi:ResultSet>true</jdbcasi:ResultSet>
<jdbcasi:Parameters>
```

エンタープライズ・サービス・ディスカバリーは、生成したビジネス・オブジェクトに MaxNumOfRetRS および ResultSet ASI の値を保存できない

解決策

ウィザードで、まず「**ストアード・プロシージャーから返される結果セットの最大数**」フィールドの値を設定して、ストアード・プロシージャーを検証すると、「**ストアード・プロシージャーから返される結果セットの最大数**」フィールドの値は自動的に 0 にリセットされ、それまでの値は無視されます。生成されたビジネス・オブジェクトに値を正しく保存するには、まずストアード・プロシージャーを正常に検証してから、「**ストアード・プロシージャーから返される結果セットの最大数**」フィールドに値を指定してください。

『ストアード・プロシージャーおよびストアード関数の選択および構成』も参照してください。

WebSphere Adapter for JDBC が、単一のスキーマで同じ名前を持つ複数のストアード・プロシージャーをサポートしていない

『ストアード・プロシージャーおよびストアード関数の選択および構成』を参照してください。

ビジネス・オブジェクト属性の Date 型 ASI が、 Date および Time 型の列を参照する

問題

JDBC エンタープライズ・メタデータ・ディスカバリーは、Time、TimeStamp、および Date のすべてのデータベース列を、ビジネス・オブジェクト内のサービス・データ・オブジェクトの Date 型の属性にマップします。 SDO Date 型は、Date 型属性が、データベースに保管された実際の時間ではなく、 GMT で表示されるような方法で処理されます。値が SDO またはデータベースに設定されるときに、時間部分は消失します。

解決策

SQL の型が DATE、TIME、および TIMESTAMP である列について、JDBC エンタープライズ・メタデータ・ディスカバリーは対応する属性の型を String に設定し、DataType という名前の、属性のアプリケーション固有情報が追加されます。この値は、Date、Time、TimeStamp のいずれかに設定されます。適切な属性の型およびアプリケーション固有情報を持つビジネス・オブジェクト XSD を生成するには、JDBC エンタープライズ・メタデータ・ディスカバリーを再実行してください。

Outbound 中にビジネス・オブジェクトに日時の値を設定するときには、以下の形式を使用します。

- 日付形式は yyyy-mm-dd です。

- 時刻形式は hh:mm:ss です。
- タイム・スタンプ形式は yyyy-mm-dd hh:mm:ss です。

同様に、Inbound については、キー値がイベント・テーブル内の object_key 列について date または time 型である場合、上記で説明した形式で値を入力する必要があります。

XML データ型および XQuery が J2CA JDBC アダプターで直接サポートされない

「How J2CA Adapter works with XML datatype in DB2」を参照してください。

BPEL が特定のビジネス・フォールトを catch できない

「BPEL が特定のビジネス・フォールトを catch できない」を参照してください。

グローバル・トランザクションでの例外によって Outbound が失敗する

問題

Outbound がグローバル・トランザクションでの例外によって失敗し、トランザクションを正常にロールバックできません。

原因

バージョン 6.2 以降、WebSphere Adapter for JDBC は、ビジネス・オブジェクトで例外 (IntegrityConstraintViolationException、MatchesExceededLimitException、MissingDataException、MultipleMatchingRecordsException、ObjectNotFoundException、RecordNotFoundException、UniqueConstraintViolatedException など) をサポートしています。これらのいずれかの例外がスローされると、EIS バインディングによって、対応するフォールト・ビジネス・オブジェクト (IntegrityConstraintFault、MatchesExceededLimitFault、MissingDataFault、MultipleMatchingRecordsFault、ObjectNotFoundFault、RecordNotFoundFault、UniqueConstraintFault など) として例外がカプセル化されます。

EIS バインディングは、フォールト・ビジネス・オブジェクトを受け取るとすぐに、 ServiceRuntimeException の代わりに ServiceBusinessException をスローします。グローバル・トランザクションで、コンポーネントが ServiceRuntimeException をスローすると、トランザクション・マネージャーはグローバル・トランザクションをロールバックします。しかし、コンポーネントが ServiceBusinessException をスローすると、トランザクション・マネージャーはグローバル・トランザクションをコミットします。

解決策

1. .import ファイル内のすべてのフォールト・ビジネス・オブジェクト・バインディングを削除します。

```
<methodBinding
      inDataBindingType="com.ibm.xmlns.prod.websphere.j2ca.jdbc.db2admi
      ncustom erbg.Db2adminCustomerBGDataBinding"
      method="createDb2adminCustomerBG"
      outDataBindingType="com.ibm.xmlns.prod.websphere.j2ca.jdbc.db2adm
      incusto merbg.Db2adminCustomerBGDataBinding">
      <faultBinding fault="INTEGRITY_CONSTRAINT_VIOLATION"
      faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultData
      BindingI    mpl"/>
      <faultBinding fault="MISSING_DATA"
      faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultData
      BindingI    mpl"/>
      <faultBinding fault="OBJECT_NOTFOUND_EXCEPTION"
      faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultData
      BindingI    mpl"/>
      <faultBinding fault="UNIQUECONSTRAINT_VIOLATION"
      faultBindingType="com.ibm.j2ca.extension.emd.runtime.WBIFaultData
      BindingI    mpl"/>
      <interaction>
        <properties>
            <functionName>Create</functionName>
        </properties>
      </interaction>
  </methodBinding>
```

2. 以下に示すように、ビジネス・フォールトを処理するようにフォールト・ハンドラーを定義します。

```
DataObject bo = null;
  try {
   bo = (DataObject) locateService_JDBCOutboundInterfacePartner().invoke
("createDb2adminAddressBG",createDb2adminAddressBGInput);
  } catch (ServiceBusinessException e) {
   // TODO Auto-generated catch block
   throw new ServiceRuntimeException(e);
  }
```

JDBC アダプターの Outbound に例外がある場合は常に、ServiceRuntimeException が生成され、グローバル・トランザクションはロールバックされます。それ以外の場合、グローバル・トランザクションはコミットされます。

EAR ファイル内の 1 つ以上のモジュールとの JDBC アダプターの共用

組み込みデプロイメントによって、 EAR ファイル内の 1 つ以上のモジュールと JDBC アダプターを共用するには、以下のようにします。

1. WebSphere Integration Developer のビジネス・インテグレーション・パースペクティブで、「**ファイル**」>「**新規**」>「**外部サービス**」と選択します。
2. IBM WebSphere Adapter for JDBC を選択して、Outbound または Inbound サービスを作成します。各種のデータベースに接続するために、 JDBC ドライバーをアダプター・コネクター・プロジェクトに確実に追加します。 JDBC アダプターを各種のデータベースに接続したい場合に、すべての JDBC ドライバーを一度に JDBC アダプター・コネクター・プロジェクトに追加できます。
3. 既存のコネクター・プロジェクトを選択して、もう 1 つ Outbound または Inbound サービスを作成します。それまでに成果物用に作成したものと同じモジュールを使用します。

4. Inbound または Outbound サービスを作成したら、選択したモジュールの「依存関係」ページの「**J2EE**」エリアに、 JDBC アダプター・コネクター・プロジェクトのみがリストされているかどうかを確認します。

詳しくは、「EAR ファイル内の 1 つ以上のモジュールとの JDBC アダプターの共用」を参照してください。

イベント・テーブルでの複合キー用の object_key 列値の設定

問題

JDBC アダプターがポーリングするイベント・テーブルには、複合基本キーとして設定されている列がいくつかあります。JDBC アダプターがイベントをポーリングできるように object_key 列をイベント・テーブルに設定するには、どのような方法がよいですか。

解決策

次の例で、このシナリオについて説明します。以下に、データベース内の Employee テーブルを示します。

```
CREATE TABLE ADMIN.EMPLOYEE ( EMPLOYEEID VARCHAR (10) NOT NULL ,
JOBCODE VARCHAR (10) NOT NULL , EMPLOYEENAME VARCHAR (10) NOT NULL ,
AGE VARCHAR (10) NOT NULL , CONSTRAINT CC1182727951922 PRIMARY KEY (
EMPLOYEEID, JOBCODE) ) ;
```

EmployeeID と JobCode は両方とも、Employee テーブル内の複合キーです。

Employee テーブルにデータを挿入するには、以下の SQL ステートメントを使用します。

```
INSERT INTO employee (employeeid,jobcode,employeename,age) VALUES
('1','8','Mike','30');
```

「Employee」テーブルには、1 (EmployeeID) と 8 (JobCode) の両方に「Mike」という従業員のレコードがあります。

トリガーを使用してイベント・テーブルにレコードを挿入する場合は、以下のような SQL ステートメントが必要です。

```
INSERT INTO wbia_jdbc_eventstore (object_key, object_name,
object_function, event_priority, event_status)
VALUES ('1;8', 'AdminEmployeeBG', 'Create', 1, 0);
```

ESD を実行して「Employee」テーブルのビジネス・オブジェクトを生成した後、そのビジネス・オブジェクトから生成されるキー属性は両方とも、テーブル定義と同じ順序になります。正しい順序は、「EmployeeID」が先頭で、「JobCode」がその次です。

したがって、「object_key」の値をイベント・テーブルに挿入するときは、ビジネス・オブジェクト定義とテーブル定義を同じ順序にする必要があります。例えば、「EmployeeID」列の値が「1」で、「JobCode」列の値が「8」の場合、「object_key」の正しい値は、「8;1」ではなく、「1;8」です。

さらに、デリミッターとしてセミコロン文字 (;) を使用して、「object_key」列内の各複合キー値を区切る必要があります。

ユーザー定義イベント照会用の WebSphere Business Integration Adapter for JDBC での WebSphere Adapter for JDBC の使用

問題

ユーザー定義イベント照会に、WebSphere Adapter for JDBC の WBIA_JDBC_EventStore イベント・テーブルではなく、WebSphere Business Integration Adapter for JDBC の XWORLDS_EVENTS イベント・テーブルを使用すると、エラーが発生し、「フィールド object_function がありません (field object_function not there)」または「無効な列名です (Invalid column name)」というメッセージが生成されます。

原因

WebSphere Business Integration Adapters のイベント・テーブル XWORLDS_EVENTS に、フィールド「object_function」がありません。また、イベント・テーブル WBIA_JDBC_EVENTSTORE に、フィールド「connector_ID」がありません。

解決策

以下に、マイグレーション・ステップを示します。

1. connector_ID でテーブル「WBIA_JDBC_EVENTSTORE」を作成します。

   ```
   CREATE TABLE WBIA_JDBC_EventStore
   (
   event_id INTEGER NOT NULL PRIMARY KEY,
   xid VARCHAR(200),
   object_key VARCHAR(80) NOT NULL,
   object_name VARCHAR(40) NOT NULL,
   object_function VARCHAR(40) NOT NULL,
   event_priority INTEGER NOT NULL,
   event_time TIMESTAMP default CURRENT TIMESTAMP NOT NULL,
   event_status INTEGER NOT NULL,
   connector_ID VARCHAR(40),
   event_comment VARCHAR(100)
   );
   ```

   いくつかのイベントを JDBC アダプター・インスタンスからフィルター処理するために、「connector_ID」フィールドが使用されます。「connector_ID」によって、現在、connectorID フィルター処理を使用している WebSphere Business Integration Adapter のお客様がシームレスに JCA にマイグレーションすることができます。この機能により、お客様は同じタイプのイベントが数多くある場合にロード・バランシングを行うことができます。
2. 以下の SQL ステートメントを作成して実行します。

   ユーザー定義の照会:

   ```
   select event_id, object_key, object_name, object_verb as object_function ,
   connector_id from xworlds_events where event_status = 0
   ```

   ユーザー定義の update:

   ```
    update xworlds_events set event_status= 1 where event_id = ?
   ```

   ユーザー定義の delete:

   ```
   delete from xworlds_events where event_id= ?
   ```

WebSphere Adapter for JDBC の Inbound サービス用のアーカイブ・イベント機能の実装

「WebSphere Adapter for JDBC の Inbound サービス用のアーカイブ・イベント機能の実装」を参照してください。

JDBC アダプターが、Inbound 処理中にネイティブ・メソッド用のメソッドを検出できない

問題

JDBC アダプターが、Inbound 処理中にネイティブ・メソッド用のメソッドを検出できません。

原因

JDBC アダプターは、内部でネイティブ・メソッドを使用して、サポートされる各操作にマップします。Inbound 操作とネイティブ・メソッドの関係は、エクスポート・コンポーネント・プロパティーのメソッド・バインディングに定義されます。

JDBC アダプターは、プレフィックス「emitCreateAfterImage」と、アダプターのイベント・テーブルから取得されるオブジェクト名を使用することで、 Create 操作用のネイティブ・メソッドを構成します。アダプターのエクスポート・コンポーネントに定義されたネイティブ・メソッドが emitCreateAfterImageAdminCustomerBG で、アダプターのイベント・テーブルに挿入されたオブジェクト名が AdminCUSTOMERBG の場合は、2 つのネイティブ・メソッドの名前が互いに一致しないため、エラーが生成されます。

解決策

この問題を修正するには、次のいずれかの方法を使用できます。

- オブジェクト名として AdminCustomerBG をイベント・テーブルに挿入します。または
- アダプターのエクスポート・コンポーネントのプロパティーで、emitCreateAfterImageAdminCustomerBG を emitCreateAfterImageAdminCUSTOMERBG に置き換えます。

BLOB 列を含む結果セットから内容を取得中に、ドライバーによって NULL ポイント例外が生成される

問題

アダプターが DB2 ストアード・プロシージャーで返される BLOB 列を含む結果セットから内容を取得しようとすると、ドライバーによって NULL ポイント例外が生成されます。

原因

DB2 JCC ドライバーのバージョンが 4.7.85 です。

解決策

DB2 JCC ドライバー 3.50.152 を使用してください。

Oracle の Date 型が、Date ではなく dateTime にマップされる

問題

Oracle の一部のフィールドにおいて、Date データ型が Date ではなく dateTime にデフォルトでマップされます。

原因

これは、Oracle JDBC ドライバーの問題です。Oracle データベースにおける Date 型は、JDBC 仕様に定義されている Timestamp 型と似ています。 Date 型には、時間の情報も含まれています。アダプターは、ドライバーから戻されたデータ型に基づいて、JDBC 型を SDO 型にマップします。ドライバーが JDBC 型の Date を戻すと、アダプターはそれを SDO 型の Date にマップします。ドライバーが JDBC 型の Timestamp を戻すと、アダプターはそれを SDO 型の dateTime にマップします。

解決策

目的の型に手動でマップできます。

Oracle での XML データ型のサポート

問題 1

xmlparserv2.jar がアプリケーション・ライブラリーに追加されると、アプリケーションをデプロイしようとするときに管理コンソールで例外が生成されます。

原因

xmlparserv2.jar がアプリケーション・ライブラリーに追加されると、XML 処理インターフェースである DocumentBuilderFactory、SAXParserFactory、および TransformerFactory の実装として登録されます。xmlparserv2.jar が TransformerFactory の実装として登録されると、互換性の問題から、アプリケーションのデプロイメント・フェーズで以下の例外が生成されます。

```
java.lang.IllegalArgumentException
 at oracle.xml.jaxp.JXTransformer.setOutputProperty(JXTransformer.java:793)
 at org.eclipse.xsd.util.DefaultJAXPConfiguration.createTransformer
  (DefaultJAXPConfiguration.java:63)
 at org.eclipse.xsd.util.XSDResourceImpl.doSerialize(XSDResourceImpl.java:153)
 at org.eclipse.xsd.util.XSDResourceImpl.serialize(XSDResourceImpl.java:136)
 at org.eclipse.xsd.ecore.XSDEcoreBuilder.setAnnotations(XSDEcoreBuilder.java:3087)
 at com.ibm.ws.bo.BOModelBuilder.access$201(BOModelBuilder.java:103)
 at com.ibm.ws.bo.BOModelBuilder$1.run(BOModelBuilder.java:1778)
 at java.security.AccessController.doPrivileged(AccessController.java:202)
 at com.ibm.ws.bo.BOModelBuilder.setAnnotations(BOModelBuilder.java:1774)
 at org.eclipse.xsd.ecore.XSDEcoreBuilder.getEPackage(XSDEcoreBuilder.java:161)
 at com.ibm.ws.bo.BOModelBuilder.getEStructuralFeature(BOModelBuilder.java:983)
 at org.eclipse.xsd.ecore.XSDEcoreBuilder.generate(XSDEcoreBuilder.java:2709)
 at com.ibm.ws.bo.BOModelBuilder.generate(BOModelBuilder.java:340)
 at com.ibm.ws.bo.BOModelBuilder.generate(BOModelBuilder.java:650)
 at com.ibm.ws.bo.BOModelBuilder.build(BOModelBuilder.java:309)
 at com.ibm.ws.bo.BOModelHolder.loadModels(BOModelHolder.java:591)
 at com.ibm.ws.bo.BOModelHolder.loadAllModels(BOModelHolder.java:626)
```

```
at com.ibm.ws.bo.BOModelHolder.loadNamespace(BOModelHolder.java:545)
at com.ibm.ws.bo.BOEPackageRegistry.loadEPackage(BOEPackageRegistry.java:218)
at com.ibm.ws.bo.BOEPackageRegistry.getEPackage(BOEPackageRegistry.java:295)
```

解決策

<WPS_HOME>/java/jre/lib/ の下で jaxp.properties.sample を jaxp.properties にリネームすることで、互換性のある実装をクラス・ローダーにロードさせるようにします。さらに、jaxp.properties ファイルで、javax.xml.transform.TransformerFactory、javax.xml.parsers.SAXParserFactory、および javax.xml.parsers.DocumentBuilderFactory の各プロパティーのコメントを外します。サーバーを再始動します。

注: Oracle で、JDNI データ・ソースを使用してデータベースに接続している場合は、データ・ソースの作成時に必ず xdb.jar と xmlparserv2.jar をクラスパスに追加してください。

問題 2

Oracle (ドライバー・バージョン 11.1.0.7.0) で、XML データ型があるテーブルにおいて、Retrieve 操作および RetrieveAll 操作が正しくない結果を戻します。

原因

Oracle のドライバー・バージョン 11.1.0.7.0 では、複合型の XML 型属性の内容は正しく読み取られません。ドライバーからは常にヌルが戻されます。

解決策

ドライバー・バージョン 11.2.0.1.0 を使用してください。

第 4 章 Date、Time、および Timestamp の各データ型の形式のカスタマイズ

DateFormat パラメーターを使用して、Date、Time、および Timestamp の各データ型の形式をカスタマイズできます。アダプターは、Date、Time、または Timestamp の SQL データ型をストリングに変換する (およびその逆の変換を行う) 必要があるときに、このパラメーターを使用します。

ビジネス・オブジェクト

ビジネス・オブジェクトとは、データ、データ上で実行されるアクション、およびデータを処理するための追加の指示 (存在する場合) で構成される構造体のことです。WebSphere Adapter for JDBC は、ビジネス・オブジェクトを使用して、データベースのテーブルとビュー、データベース照会、ストアード・プロシージャー、およびストアード関数の結果を表現します。ビジネス・オブジェクトにより、データベースのオブジェクトの階層を作成し、無関係なテーブルをグループ化できます。コンポーネントはビジネス・オブジェクトを使用してアダプターと通信します。

アダプターによるビジネス・オブジェクトの使用法

統合アプリケーションは、ビジネス・オブジェクトを使用してデータベースにアクセスします。アダプターは、Outbound 要求のビジネス・オブジェクトを、データベースへアクセスするための JDBC API 呼び出しに変換します。Inbound イベントの場合、アダプターはイベントのデータをビジネス・オブジェクトに変換し、このビジネス・オブジェクトがアプリケーションに戻されます。

アダプターは、ビジネス・オブジェクトを使用してデータベース内の次のタイプのオブジェクトを表現します。

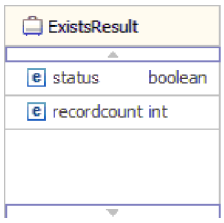
- テーブルとビュー
- シノニムとニックネーム
- ストアード・プロシージャーとストアード関数

一部のビジネス・オブジェクトは、データベース・オブジェクトを表現しません。これらのビジネス・オブジェクトには、以下のものがあります。

- バッチ SQL ビジネス・オブジェクト。一連のユーザー定義の insert ステートメント、update ステートメント、および delete ステートメントを表します。
- クエリー・ビジネス・オブジェクト。データベースに対して実行されるユーザー定義 SQL 照会を表します。
- ラッパー・ビジネス・オブジェクト。このオブジェクトにより、無関係なテーブル・オブジェクトとビュー・オブジェクトを単一のビジネス・オブジェクトにグループ化でき、複数のストアード・プロシージャーを単一のビジネス・オブジェクトにグループ化することができます。

アダプターは、出力に一部のビジネス・オブジェクトを使用します。これらのビジネス・オブジェクトには、以下のものがあります。

- コンテナー・ビジネス・オブジェクト。RetrieveAll 操作からの出力が入ります。
- ExistsResult ビジネス・オブジェクト。これには、Exists 操作からの出力が入ります。



ビジネス・オブジェクト内でのデータの表現方法

テーブルまたはビュー・ビジネス・オブジェクトの場合

テーブルまたはビューの各列は、テーブル・ビジネス・オブジェクトまたはビュー・ビジネス・オブジェクトの単純属性により表現されます。*単純属性* とは、String、Integer、または Date などの単一値を表す属性です。その他の属性は、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表します。

同じビジネス・オブジェクトに含まれる単純属性を、別々のデータベース表に格納することはできませんが、次の状況が考えられます。

- データベース表に、対応するビジネス・オブジェクトに含まれる単純属性の数よりも多くの列が含まれる場合があります。つまり、データベースの列の一部が、ビジネス・オブジェクト内に表されていません。アプリケーションによるビジネス・オブジェクトの処理で必要な列のみを実際の設計に含める必要があります。
- ビジネス・オブジェクトに、対応するデータベース表に含まれる列の数よりも多くの単純属性が含まれる場合があります。つまり、ビジネス・オブジェクト内の属性の一部が、データベース内に表されていません。データベース内に列を持たない属性は、アプリケーション固有情報を持っていないか、デフォルト値が設定されているか、またはストアード・プロシージャーかストアード関数のパラメーターです。
- ビジネス・オブジェクトは、複数のデータベース表にまたがるビューを表すことができます。アダプターでは、Create、Update、および Delete 操作など、データベースに対する変更によって起動されるイベントを処理するときに、このようなビジネス・オブジェクトを使用できます。ただし、ビジネス・オブジェクトの要求を処理する場合には、Retrieve および RetrieveAll 要求に対してのみ、このようなビジネス・オブジェクトを使用できます。

テーブル・ビジネス・オブジェクトには、対応するデータベース表に基本キーがない場合でも、常に基本キーが設定されています。アダプターは、テーブル・ビジネス・オブジェクトを取得するときに、基本キー属性で指定される列を使用します。データベースで外部キー参照が定義されている場合、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。例えば、親テーブルである CUSTOMER テーブルと子テーブルである ADDRESS テーブルがあるとします。データベース内で ADDRESS から CUSTOMER への外部キー参照を定義した場合、アダプターは自動的に親子関係をディスカバーし、外部キー参照を「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウ

ィンドウに表示します。外部キー参照が CUSTOMER から ADDRESS への場合、アダプターは自動的に「**単一カーディナリティー**」チェック・ボックスを選択し、外部キー参照を表示します。1 つのテーブルに複数の外部キー参照が定義されている場合、アダプターは外部キー関係を 1 つだけ生成します。

アダプターでは、複合の (つまり複数の) 基本キーが設定されている表をサポートしています。データベースに基本キーが 1 つ以上存在する場合、ウィザードは、テーブル・ビジネス・オブジェクトのそれらの列に基本キー・プロパティーを設定します。データベース表に基本キーが存在しない場合、外部サービス・ウィザードでは、そのビジネス・オブジェクトをディスカバーして構成するときに基本キー情報の入力を求めるプロンプトが出されます。例えば CUSTOMER (pkey1, pkey2) > ADDRESS (fkey1, fkey2) といった複合基本キーがある場合、アダプターは ADDRESS (fkey1) と CUSTOMER (pkey1)、ADDRESS (fkey2) と CUSTOMER (pkey2) をそれぞれ関連付けます。シーケンス列や ID 列など、固有のデータを持つ列を指定してください。ID 列 (Informix では*シリアル列* と呼ばれる) は、データベースでテーブルの各行に自動的に固有の数値を生成する方法になります。テーブルには、ID 属性で定義される単一の列を作成できます。 ID 列の例として、オーダー番号、従業員番号、ストック番号、および問題番号などがあります。 ID 列は、DB2、Informix および Microsoft SQL Server のテーブルにのみ定義できます。

注: DB2 または Microsoft SQL Server データベースのいずれかの表に対してディスカバリー・プロセスを実行する場合に、その表で 1 つの列を ID 列として定義している場合、その表に対して生成されたビジネス・オブジェクトには、ID 列の固有 ID 属性は含まれていません。この場合、アプリケーション固有情報に属性を手動で追加することにより、生成されたビジネス・オブジェクトを編集する必要があります。これは、WebSphere Integration Developerのアセンブリー・エディターによって行うことができます。Informix データベースのテーブルに対してディスカバリー・プロセスを実行した場合は、固有 ID の属性を手動で追加する必要はありません。Informix の場合、生成されたビジネス・オブジェクトには、シリアル列の固有 ID 属性が含まれています。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

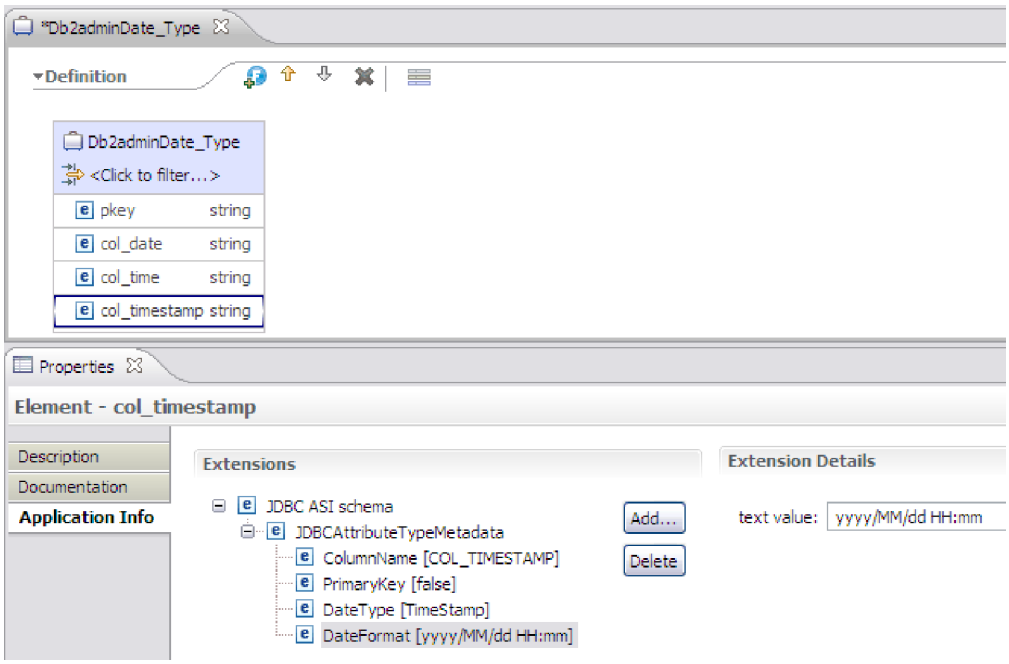


図 11. DateFormat アプリケーション固有情報とカスタマイズされた形式

テーブル・ビジネス・オブジェクトおよびビュー・ビジネス・オブジェクトは、Create、Update、Delete、Retrieve、RetrieveAll、Exists、および ApplyChanges の Outbound 操作をサポートします。階層型テーブル・ビジネス・オブジェクトに Exists 操作を実行すると、トップレベルのビジネス・オブジェクトのみが照会されます。

83 ページの図 12 に、ビジネス・オブジェクト・エディターに表示されたテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとに 1 つの属性が設定されています。表には子ビジネス・オブジェクトがないため、属性はすべて単純属性です。

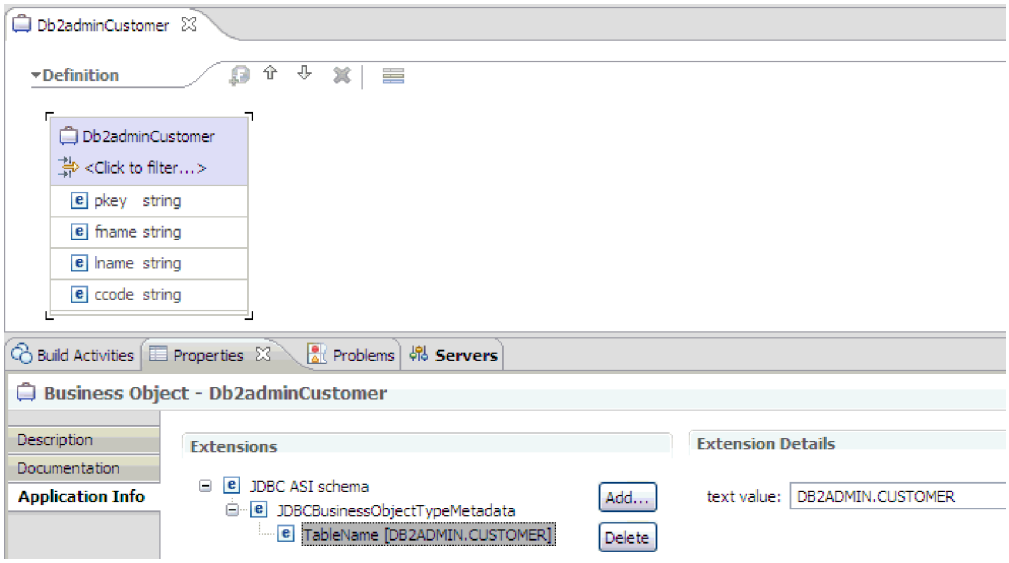


図 12. 子なしのテーブル・ビジネス・オブジェクト：

図 13 に、子テーブル・ビジネス・オブジェクトが 1 つあるテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとの単純属性に加えて、子ビジネス・オブジェクトを指す複合属性が設定されています。

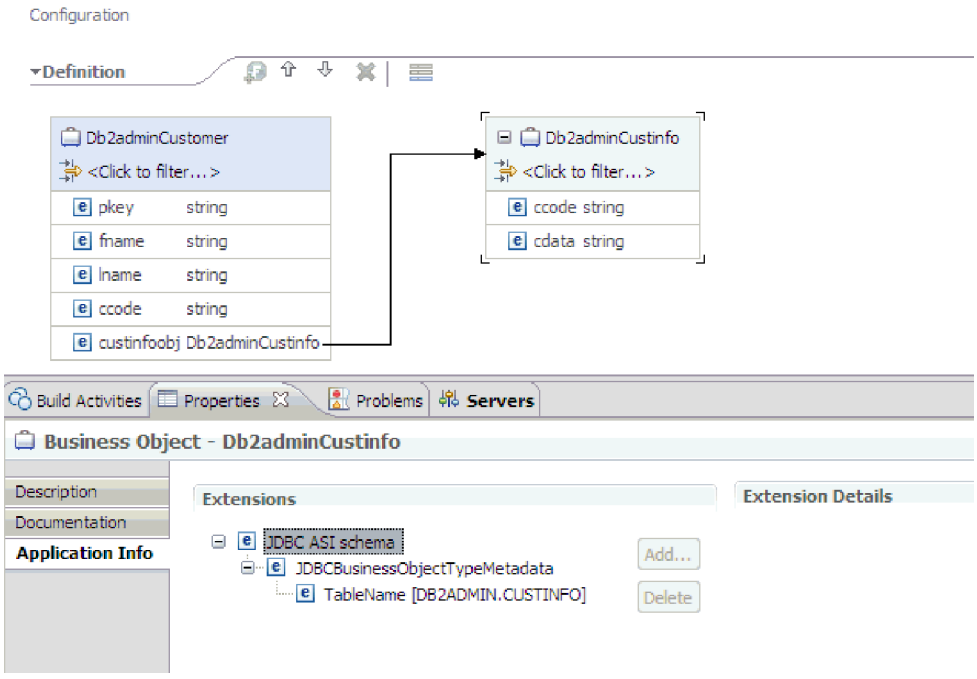


図 13. 子ビジネス・オブジェクトを 1 つ持つテーブル・ビジネス・オブジェクト：

Oracle データベースの場合、アダプターは、テーブル・ビジネス・オブジェクトで配列、テーブル、構造体、ネストされた構造体などのユーザー定義型または複合データ型をサポートします。これらの型に対しては、型名および子属性の詳細が自動的にディスカバーされて、表示されます。アダプターでは、テーブル・ビジネス・オブジェクトの子ビジネス・オブジェクトとしてこれらのデータ型が処理されま

す。

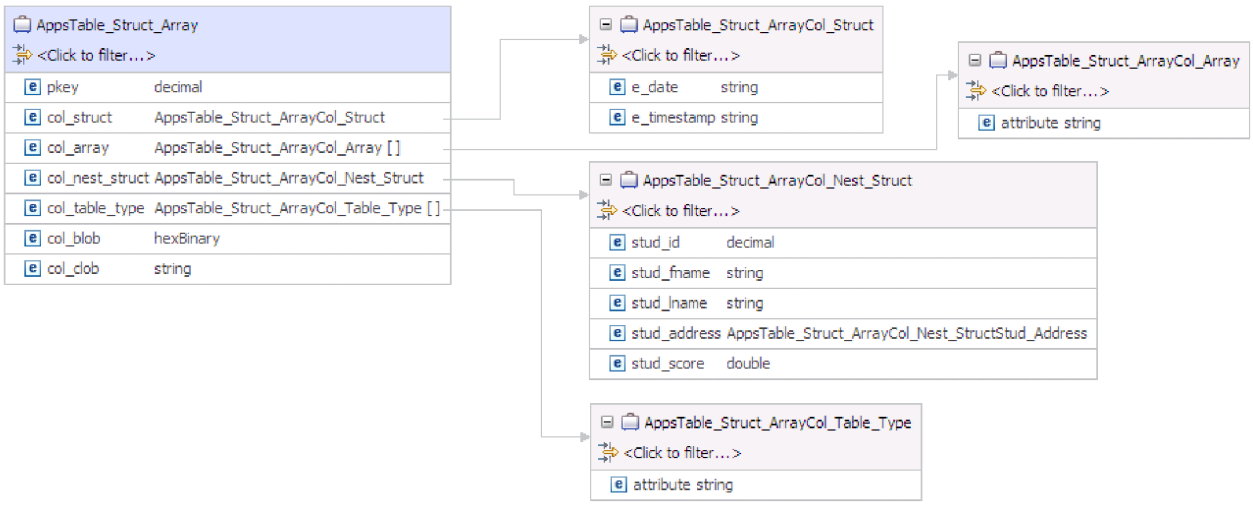


図 14. 列にユーザー定義型または複合型がある Oracle テーブル・ビジネス・オブジェクト

ストアード・プロシージャー・ビジネス・オブジェクトとストアード関数ビジネス・オブジェクトの場合

ストアード・プロシージャーまたはストアード関数のビジネス・オブジェクトでは、ストアード・プロシージャーまたはストアード関数のすべての入力パラメーターおよび出力パラメーターに、ビジネス・オブジェクトに対応する属性があります。入力または出力パラメーターのいずれかが、配列や構造体などの複合型である場合、対応するビジネス・オブジェクト属性は、配列または構造体の属性を含む子ビジネス・オブジェクトを持つ子ビジネス・オブジェクト型です。ストアード・プロシージャーが結果セットを戻した場合、戻された結果セットの属性を格納する子ビジネス・オブジェクトが作成されます。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

ストアード・プロシージャーとストアード関数のビジネス・オブジェクトは、Execute Outbound 操作をサポートします。

ストアード・プロシージャー・ビジネス・オブジェクトの構造を下のサンプル・ファイルに示します。ビジネス・オブジェクト ScottStrtValues および ScottStrtValuesStrt が、1 つの入力タイプと 2 つの出力タイプを持つストアード・プロシージャーから生成されます。出力パラメーターの 1 つは、構造体データ型です。外部サービス・ウィザードによって、構造体型のビジネス・オブジェクト ScottStrtValuesStrt が生成され、子オブジェクトとして親ビジネス・オブジェクト ScottStrtValues に追加されます。親ビジネス・オブジェクト内の構造体型の属性については、ChildBOType アプリケーション固有情報が Struct に設定され、型が構造体であることを示します。ChildBOTypeName アプリケーション固有情報は、データベース内のユーザー定義構造体型の値に設定されます。次の例は、ストアード・プロシージャーのスキーマを示しています。

ScottStrtValues ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvalues" xmlns:scottstrtvalues=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvalues" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt">
<import namespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt"
schemaLocation="ScottStrtvaluesStrt.xsd"/>
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvalues">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
<jdbcasi:MaxNumOfRetRS>0</jdbcasi:MaxNumOfRetRS>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>IP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="strt" type="scottstrtvaluesstrt:ScottStrtvaluesStrt"
```

```
minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
<jdbcasi:ChildBOType>STRUCT</jdbcasi:ChildBOType>
<jdbcasi:ChildBOTypeName>STRUCT1</jdbcasi:ChildBOTypeName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

ScottStrtValuesStrt ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvaluesStrt">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="name" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="title" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="dept_num" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
```

```
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

クエリー・ビジネス・オブジェクトの場合

データベース照会のビジネス・オブジェクトは、照会を実行する SQL ステートメントと、照会に必要なパラメーターを定義します。クエリー・ビジネス・オブジェクトでは、Outbound 操作 RetrieveAll がサポートされています。

例えば、次の SELECT ステートメントを実行するクエリー・ビジネス・オブジェクトがあるとします。

```
select C.pkey, C.fname, A.city from customer C, address A
    WHERE (C.pkey = A.custid) AND (C.fname like ?)
```

疑問符 (?) は、照会の入力パラメーターを示します。照会には複数のパラメーターを指定できます。各パラメーターは、SELECT ステートメントでは疑問符で示されています。サンプル・クエリー・ビジネス・オブジェクトの属性を表 11 に示します。クエリー・ビジネス・オブジェクトには、抽出される列ごとの単純属性、パラメーターごとの単純属性、およびパラメーター置換の後も WHERE 節を保持する、照会の WHERE 節の「プレースホルダー・オブジェクト」があります。

表 11. クエリー・ビジネス・オブジェクトの属性

ビジネス・オブジェクト属性	**説明**
pkey	Customer 表のデータベース列 PKEY に対応
fname	Customer 表のデータベース列 FNAME に対応
city	Address 表のデータベース列 CITY に対応
parameter1	パラメーター。SELECT ステートメントの ? (疑問符) ごとに 1 つのパラメーターがあります。複数のパラメーターを持つ SELECT ステートメントでは、後続のパラメーターに parameter2、parameter3 のようにして名前が付けられます。
jdbcwhereclause	WHERE 節のプレースホルダー・オブジェクト

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

以下の図に、ビジネス・オブジェクト・エディターに表示されたサンプル・クエリーのビジネス・オブジェクトを示します。

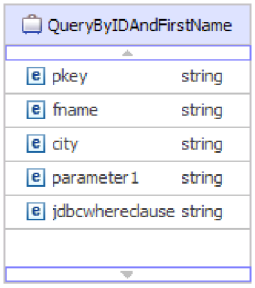


図 15. クエリー・ビジネス・オブジェクトの属性

この図は、クエリー・ビジネス・オブジェクト例のアプリケーション固有情報を示しています。SelectStatement アプリケーション固有情報には、SELECT ステートメントが含まれています。

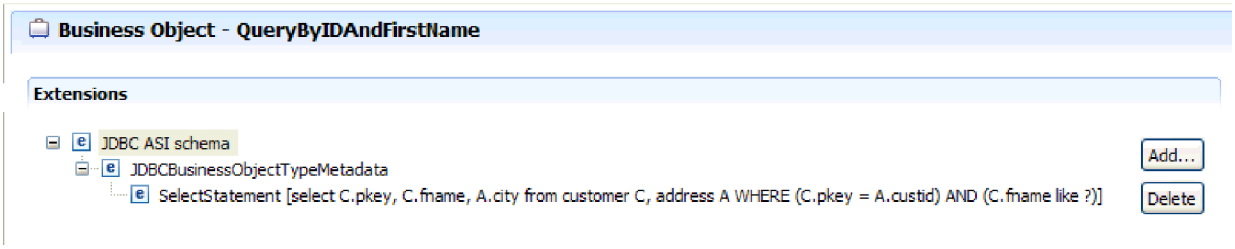


図 16. SELECT ステートメントは、ビジネス・オブジェクトのアプリケーション固有情報に保存されます。

Oracle データベースの場合、アダプターは、ビジネス・オブジェクトの照会結果で配列、テーブル、構造体、ネストされた構造体などの複合データ型をサポートします。アダプターでは、バッチおよびクエリーのビジネス・オブジェクトにおいて、これらの複合型をパラメーターとしてサポートしていません。

バッチ SQL ビジネス・オブジェクトの場合

バッチ SQL ビジネス・オブジェクトは、データベース・アクションを実行する INSERT、UPDATE、および DELETE SQL ステートメントと、そのステートメントで必要とされるパラメーターを定義します。バッチ SQL ビジネス・オブジェクトでは、Outbound 操作 Execute がサポートされています。

例えば、次の INSERT および DELETE ステートメントを実行するバッチ SQL ビジネス・オブジェクトがあるとします。

```
Insert into customer (pkey,ccode,fname,lname) values(?,?,?,?);
Delete From Customer where pkey=?
```

各疑問符 (?) は、ステートメントのパラメーターを示します。バッチ SQL ビジネス・オブジェクト内の各ステートメントには複数のパラメーターを指定できます。各パラメーターは、ステートメントでは疑問符で示されています。バッチ SQL ビ

ジネス・オブジェクトには複数のステートメントを含めることができ、それぞれが独自のパラメーター・セットを持ちます。図 17 に、それぞれが 1 つ以上のパラメーターを持つ INSERT ステートメントと DELETE ステートメントが定義されている、バッチ SQL ビジネス・オブジェクトのビジネス・オブジェクトの形式を示します。

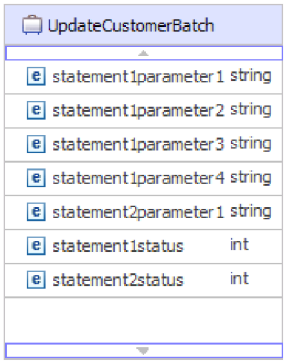


図 17. SQL ステートメントが 2 つのバッチ SQL ビジネス・オブジェクト

このビジネス・オブジェクトでは、statement1parameter1 や statement2parameter1 などの各ステートメントのパラメーターごとに属性が設定されています。また、statement1status や statement2status などの、各ステートメントの状況に関する属性もあります。ステートメント自体は、図 18 に示すように、ビジネス・オブジェクトについてのアプリケーション固有情報として格納されます。

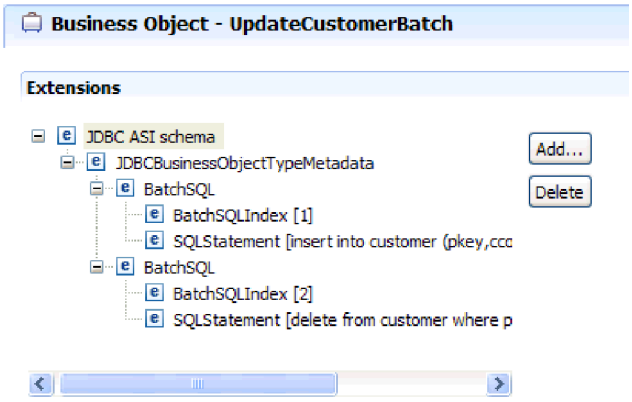


図 18. バッチ SQL ビジネス・オブジェクトのアプリケーション固有情報

ラッパー・ビジネス・オブジェクトの場合

ラッパー・ビジネス・オブジェクトにより、無関係の表の操作とビジネス・オブジェクトの表示を 1 回の操作で行うことができます。ラッパー・ビジネス・オブジェクトは、Create、Delete、Retrieve、および Update の Outbound 操作をサポートします。

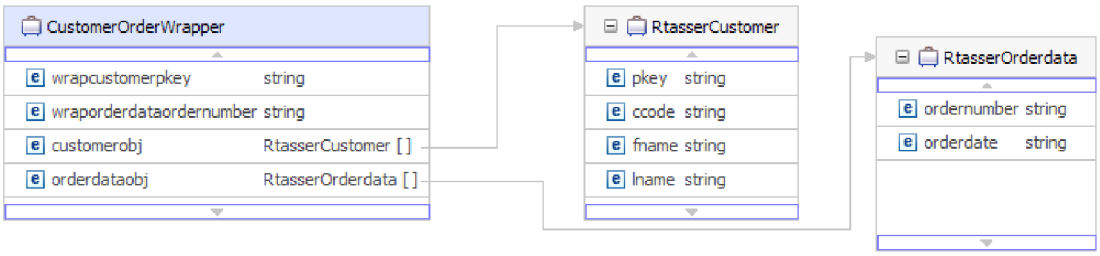


図 19. 2 つのテーブル・ビジネス・オブジェクトを含むラッパー・ビジネス・オブジェクト

ラッパー・ビジネス・オブジェクトには、子ビジネス・オブジェクトそれぞれの基本キーの単純属性が含まれています。フィールドの名前は「wrap」というストリングで、その後に、データベース表名とその表の基本キーの列名が続きます。ラッパー・ビジネス・オブジェクトには、テーブル・ビジネス・オブジェクトごとの複合属性も含まれています。属性の名前は、ストリング「obj」を付加した表名です。複合属性のタイプは、対応するテーブル・ビジネス・オブジェクトの名前です。

ビジネス・グラフ

アダプターの構成時に、ビジネス・グラフを生成するオプションを選択することもできます。バージョン 6.0.2 では、トップレベルの各ビジネス・オブジェクトがビジネス・グラフに含まれていますが、このビジネス・オブジェクトには、実行する操作に関する追加情報を指定するために、バージョン 6.0.2 でアプリケーションが使用できる動詞が組み込まれています。バージョン 7.0 では、ビジネス・グラフが必要になるのは以下の状況に限られます。

- Outbound ApplyChanges 操作を使用する必要がある場合
- バージョン 7.0 より前のバージョンの WebSphere Integration Developer で作成されたモジュールにビジネス・オブジェクトを追加する場合

ビジネス・グラフが存在する場合、ビジネス・グラフは処理されますが、ApplyChanges 以外のすべての操作で動詞は無視されます。

ビジネス・オブジェクトの作成方法

ビジネス・オブジェクトを作成するには、WebSphere Integration Developer から起動される外部サービス・ウィザードを使用します。このウィザードにより、データベースに接続し、データベース・オブジェクトがディスカバーされ、表示されます。ビジネス・オブジェクトを作成するデータベース・オブジェクトを選択します。例えば、調べるスキーマを指定します。指定されたスキーマで、テーブル、ビュー、ストアード・プロシージャー、ストアード関数、シノニム、およびニックネームを選択します。また、ビジネス・オブジェクトを追加で作成できます。例えば、データベースに対して実行されるユーザー定義の SELECT、INSERT、UPDATE、または DELETE ステートメントの結果を表すビジネス・オブジェクトを作成できます。このウィザードでは、親子関係と、無関係なビジネス・オブジェクトをまとめるラッパーを使用してビジネス・オブジェクト階層を作成できます。

必要なビジネス・オブジェクトを指定し、これらのオブジェクトの階層を定義すると、ウィザードにより、選択されたオブジェクトを表すビジネス・オブジェクトが生成されます。また、アダプターに必要なその他の成果物も生成されます。

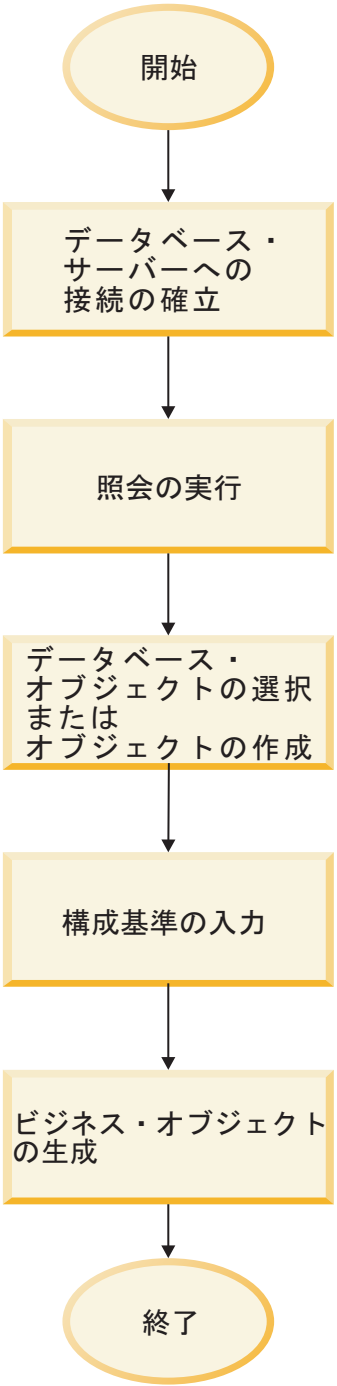


図 20. ビジネス・オブジェクトの作成方法

場合によっては、ウィザードで親子関係を完全に構成できないこともあります。これらの関係の場合は、WebSphere Integration Developer から起動するビジネス・オブジェクト・エディターを使用して、ウィザードによって作成されたビジネス・オブジェクト階層の定義を変更または完了します。詳しくは、WebSphere Integration Developer インフォメーション・センター (リンク: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp) で、ビジネス・オブジェクト・エディターによるビジネス・オブジェクトの変更方法を参照してください。

Outbound 処理のデータベース・オブジェクトのディスカバー

データベースに接続した後は、データベース・オブジェクトを検索する照会を実行します。ディスカバーされたオブジェクトのツリーを参照して、データベース内のオブジェクトの構造を理解します。また、フィルターを使用して、参照したいデータベース・オブジェクトのみを表示します。ユーザー定義データベース・クエリーおよびユーザー定義バッチ SQL ステートメントとして作成するビジネス・オブジェクトの数を定義します。

始める前に

データベースのアクセスに必要なプログラムのデータ要件を知っていなければなりません。例えば、データベースに関する以下の情報が必要になります。

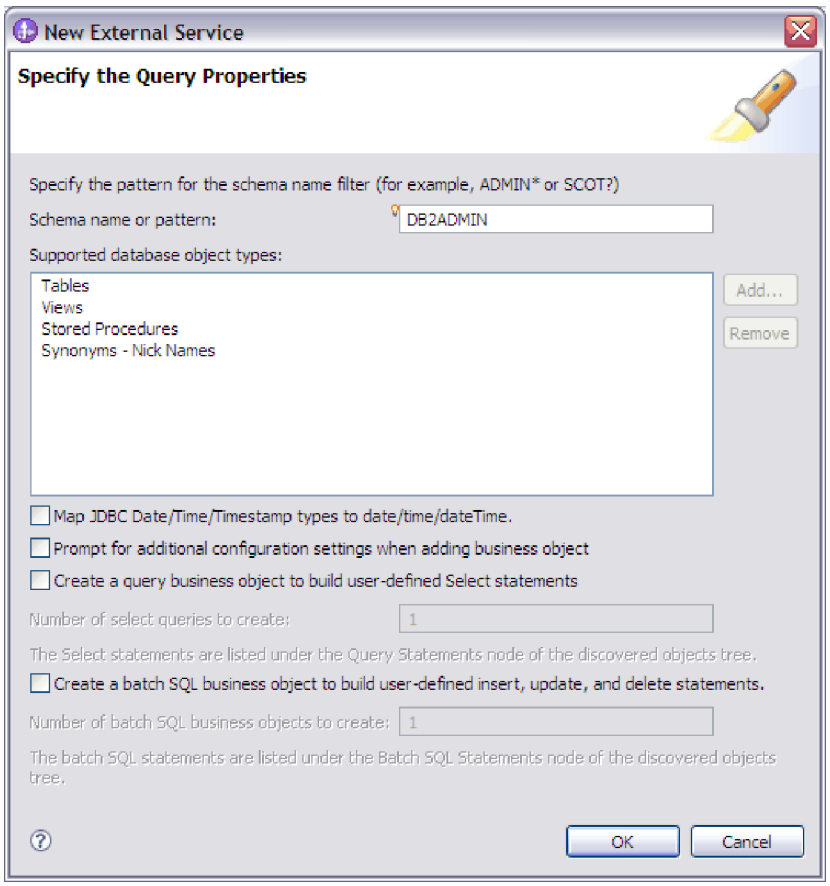
- モジュールはどのスキーマにアクセスする必要があるか
- これらのスキーマ内のどのタイプのデータベース・オブジェクトにアクセスする必要があるか
- どのテーブル、ビュー、シノニムまたはニックネーム、ストアード・プロシージャーまたはストアード関数にアクセスする必要があるか
- 作成する必要があるクエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトの数。パラメーター値およびそのパラメーターのサンプル・データベース値を含む

このタスクについて

この作業は、外部サービス・ウィザードの「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで開始します。

手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**照会の編集 (Edit Query)**」をクリックします。 照会プロパティーの指定ウィンドウが表示されます。



「照会プロパティーの指定」ウィンドウを使用して、次のタスクを実行します。

- データベース・スキーマのサブセットを検索することにより、検索時間を削減します。
- 1 つ以上のタイプのデータベース・オブジェクトを検索から除外します。
- データベース内の情報からは自動的に判別できないアプリケーション固有情報の入力をウィザードがユーザーに求めるようにします。
- 作成するクエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトの数を指定します。
- JDBC のデータ型 Date、Time、および Timestamp を date、time、および dateTime にマップします。

注: バージョン 6.1.x、フィックスパック 2 では、作成したいラッパー・ビジネス・オブジェクトの数もこのウィンドウで指定できました。バージョン 6.2.x では、ウィザードは後で wrapper 情報の入力を求めます。

2. 取得されるデータベース・スキーマの数を制限するには、「**スキーマ名またはパターン**」にスキーマの名前またはスキーマ名パターンを入力します。1 文字に対応させる場合は疑問符または下線 (? または _) を使用し、複数文字に対応させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。 クエリーを実行すると、そのストリングで始まるスキーマか、またはスキーマ名パターンに一致するスキーマのみが表示されます。スキーマ名パターン

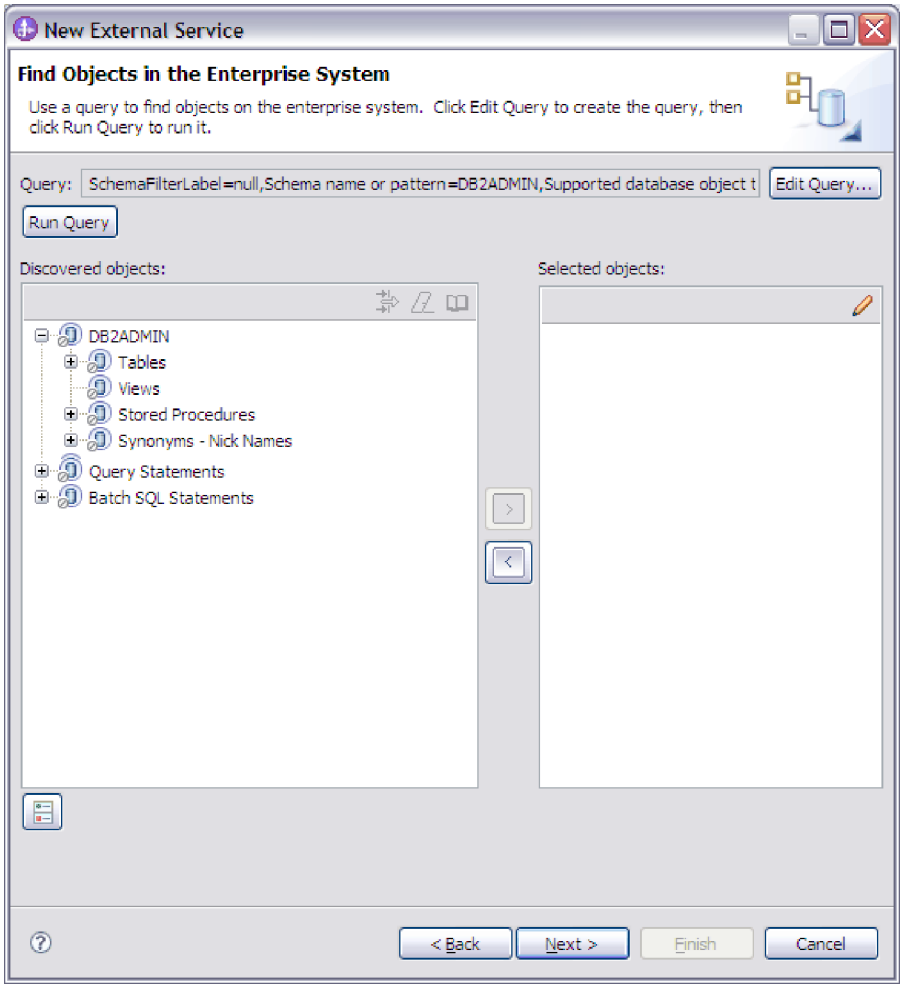
を指定しない場合は、データベース内のすべてのスキーマが表示されます。データベースに多数のスキーマがある場合は、フィルターを使用してディスカバリー・プロセスを高速化できます。

3. 検索から 1 つ以上のタイプのオブジェクトを除外するには、除外したいオブジェクトのタイプ (テーブル、ビュー、ストアード・プロシージャーおよびストアード関数、シノニムまたはニックネーム) を「**サポートされるデータベース・オブジェクト・タイプ**」フィールドで選択して、「**除去**」をクリックします。そのオブジェクト・タイプをもう一度追加するには、「**追加**」をクリックします。 特定のタイプのデータベース・オブジェクトにのみアクセスする必要がある場合は、必要のないオブジェクトを除外することで、ディスカバリー・プロセスを高速化できます。
4. データ型が Date、Time、および Timestamp であるテーブル、ストアード・プロシージャー、およびストアード関数の各オブジェクトは、デフォルトでは String データ型にマップされます。これらのオブジェクトを、JDBC ドライバーでサポートされる実際のデータ型 (Date、Time、Datetime など) にマップするには、「**JDBC の Date/Time/Timestamp の型を date/time/dateTime にマップ**」チェック・ボックスを選択します。

   注: デフォルトのデータ型マッピングは、JDBC ドライバーのバージョンによって異なります。例えば、Oracle JDBC ドライバーを使用する場合、Date データ型は、Date ではなく dateTime データ型にマップされます。このような場合は、「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウで、適切なデータ型を手動で選択する必要があります。
5. 「**ビジネス・オブジェクトの追加時に追加構成設定のプロンプトを出す**」チェック・ボックスを選択します。 これにより、作成するビジネス・オブジェクトのリストにデータベース・オブジェクトを追加すると、そのオブジェクトに関してユーザーが構成可能なすべてのアプリケーション固有情報を入力するよう求められます。例えば、このオプションを選択した場合は、ビジネス・オブジェクトの単純親子階層を作成するプロセスが、順を追ってウィザードで示されます。 2 つの異なるテーブルの属性を参照する 2 つの属性をテーブル・ビジネス・オブジェクトが持つ (つまり、2 つの親ビジネス・オブジェクトを持つ) 階層が必要な場合は、アセンブリー・エディターで構成を実行します。このエディターは、WebSphere Integration Developer から起動されるツールです。また、データベースで外部キー参照が定義されている場合は、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。

   重要: このオプションを選択しない場合、ウィザードは必須情報のみを入力するようプロンプトを出します。アセンブリー・エディターを使用して、ビジネス・オブジェクトの構成を実行する必要があります。また、データベースで外部キー参照を定義していない場合、アダプターは親子関係を自動的には生成しません。
6. ユーザー定義データベース照会を実行するビジネス・オブジェクトを作成するには、「**ユーザー定義の Select ステートメントを作成するためのクエリー・ビジネス・オブジェクトを作成する**」を選択してから、作成するクエリー・ビジネス・オブジェクトの数を入力します。 ここではビジネス・オブジェクトの数のみを指定します。このビジネス・オブジェクトについては、後で名前およびその他の詳細の入力をウィザードから求められます。

7. 一連の SQL ステートメントを実行するビジネス・オブジェクトを作成するには、**「ユーザー定義の insert、update、および delete ステートメントを作成するためのバッチ SQL ビジネス・オブジェクトを作成する」**を選択して、作成するバッチ SQL ビジネス・オブジェクトの数を入力します。 ここではビジネス・オブジェクトの数のみを指定します。このビジネス・オブジェクトについては、後で名前およびその他の詳細の入力をウィザードから求められます。
8. **「OK」**をクリックして、データベース・クエリーへの変更を保存します。
9. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、**「クエリーの実行 (Run Query)」**をクリックします。これによりこのクエリーを使用してデータベース・オブジェクトがディスカバーされ、クエリーおよびバッチ SQL ビジネス・オブジェクトのテンプレートが作成されます。 標準的な照会の実行結果を次の図に示します。



注: 有効期限が切れたデータベース接続を復元するには、外部サービス・ウィザードを再始動します。

「検出済みオブジェクト」ペインには、ディスカバーされたデータベース・オブジェクトがリストされます。

10. **「検出済みオブジェクト」**リストで、スキーマ・ノードおよびその下の**「テーブル」**、**「ビュー (Views)」**、**「ストアード・プロシージャー (Stored**

Procedures)」、「**シノニム - ニックネーム (Synonyms - Nicknames)**」の各ノードを展開するには、「**+**」 (正符号) をクリックします。これにより、ウィザードによってディスカバーされたデータベース・オブジェクトが表示されます。

11. 「**クエリー・ステートメント (Query Statements)**」および「**バッチ SQL ステートメント (Batch SQL Statements)**」のノードを展開するには、「**+**」 (正符号) をクリックします。これによって、クエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトのテンプレートが表示されます。

タスクの結果

アダプター、およびクエリー・ビジネス・オブジェクトとバッチ SQL ビジネス・オブジェクトのビジネス・オブジェクト・テンプレートを使用してアクセスできる、データベース・オブジェクトがウィザードによって表示されました。

次のタスク

外部サービス・ウィザードでの作業を続行します。次の手順では、モジュールで使用するオブジェクトの選択、各ビジネス・オブジェクトの構成、およびビジネス・オブジェクトの階層の作成を行います。

Inbound 処理のデータベース・オブジェクトのディスカバー

接続プロパティーを構成した後は、データベース・オブジェクトを検索するクエリーを実行します。ディスカバーされたオブジェクトのツリーを参照して、データベース内のオブジェクトの構造を理解します。また、フィルターを使用して、参照したいデータベース・オブジェクトのみを表示します。

始める前に

データベースのアクセスに必要なプログラムのデータ要件を知っていなければなりません。例えば、データベースに関する以下の情報が必要になります。

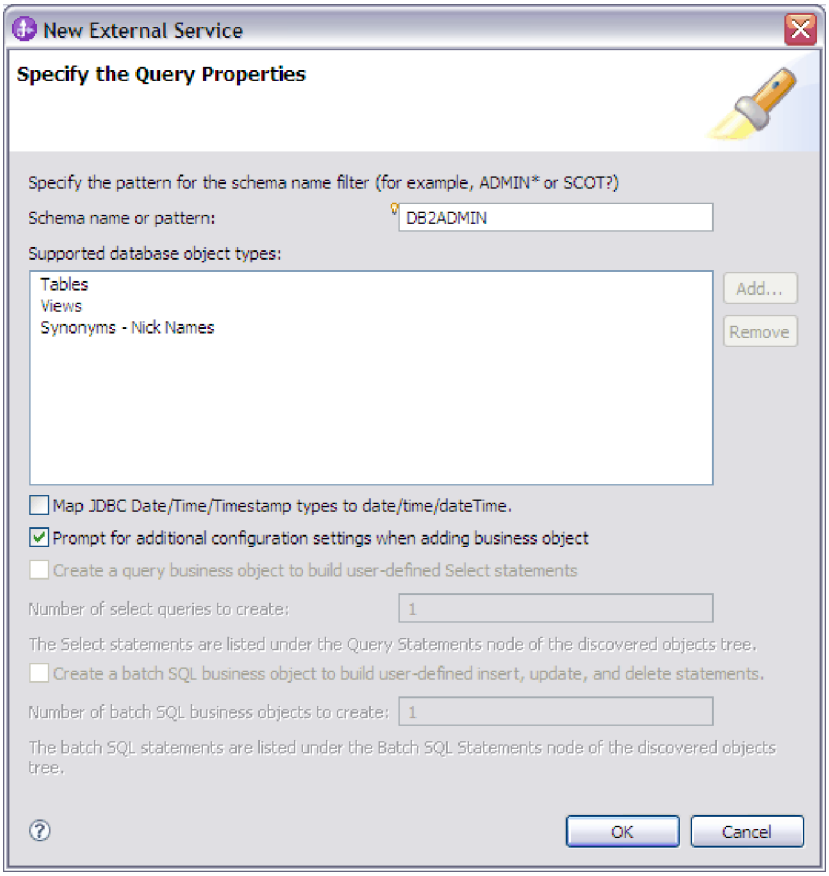
- モジュールはどのスキーマにアクセスする必要があるか
- これらのスキーマ内のどのタイプのデータベース・オブジェクトにアクセスする必要があるか

このタスクについて

この作業は、外部サービス・ウィザードの「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで開始します。

手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**照会の編集 (Edit Query)**」をクリックします。 照会プロパティーの指定ウィンドウが表示されます。



注: 「ユーザー定義の select ステートメントを作成するためのクエリー・ビジネス・オブジェクトを作成する」および**「ユーザー定義の insert、update、および delete ステートメントを作成するためのバッチ SQL ビジネス・オブジェクトを作成する」**の各オプションは、Outbound 処理にのみ使用できます。

「照会プロパティーの指定」ウィンドウを使用して、次のタスクを実行します。

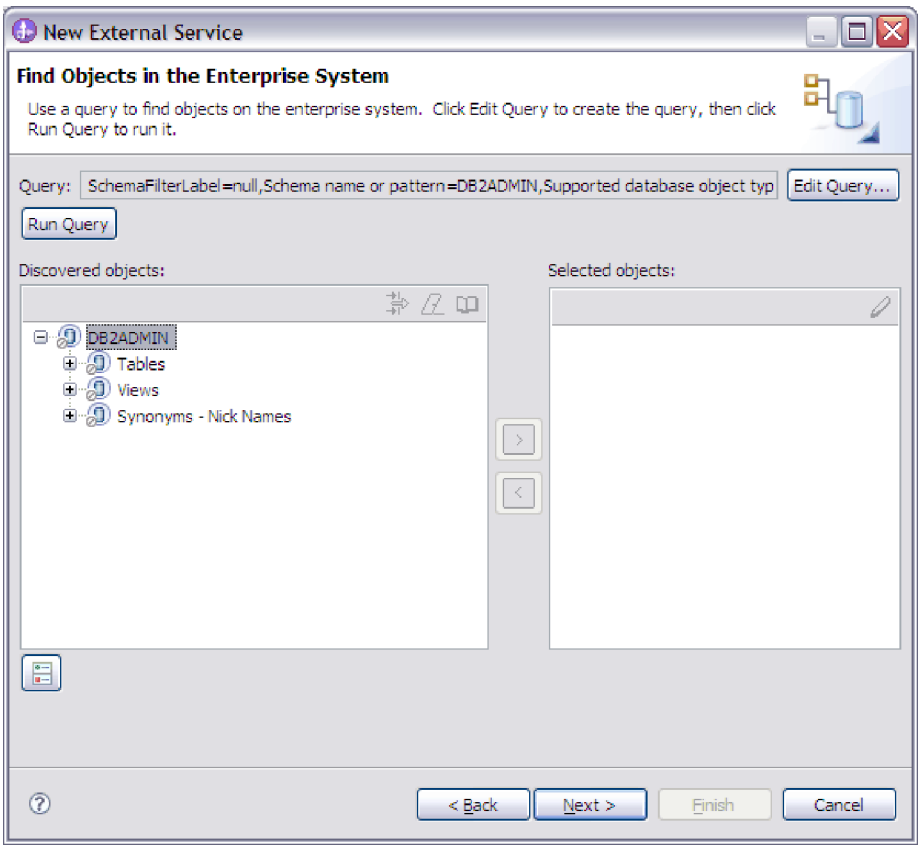
- データベース・スキーマのサブセットを検索することにより、検索時間を削減します。
- 1 つ以上のタイプのデータベース・オブジェクトを検索から除外します。
- データベース内の情報からは自動的に判別できないアプリケーション固有情報の入力をウィザードがユーザーに求めるようにします。
- JDBC のデータ型 Date、Time、および Timestamp を date、time、および dateTime にマップします。

2. 取得されるデータベース・スキーマの数を制限するには、**「スキーマ名またはパターン」**にスキーマの名前またはスキーマ名パターンを入力します。1 文字に対応させる場合は疑問符または下線 (? または _) を使用し、複数文字に対応させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。 クエリーを実行すると、そのストリングで始まるスキーマか、またはスキーマ名パターンに一致するスキーマのみが表示されます。スキーマ名パターンを指定しない場合は、データベース内のすべてのスキーマが表示されます。データベースに多数のスキーマがある場合は、フィルターを使用してディスカバリー・プロセスを高速化できます。

3. 検索から 1 つ以上のタイプのオブジェクトを除外するには、除外したいオブジェクトのタイプ (テーブル、ビュー、およびシノニムまたはニックネーム) を「**サポートされるデータベース・オブジェクト・タイプ**」フィールドで選択して、「**除去**」をクリックします。そのオブジェクト・タイプをもう一度追加するには、「**追加**」をクリックします。 アクセスする必要のないオブジェクト・タイプがデータベースに含まれている場合は、それらを除外すると、ディスカバリー・プロセスを高速化できます。
4. データ型が Date、Time、および Timestamp であるテーブル・オブジェクトは、デフォルトでは String データ型にマップされます。これらのオブジェクトを、JDBC ドライバーでサポートされる実際のデータ型 (Date、Time、Datetime など) にマップするには、「**JDBC の Date/Time/Timestamp の型を date/time/dateTime にマップ**」チェック・ボックスを選択します。

   注: デフォルトのデータ型マッピングは、JDBC ドライバーのバージョンによって異なります。例えば、Oracle JDBC ドライバーを使用する場合、Date データ型は、Date ではなく dateTime データ型にマップされます。このような場合は、「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウで、適切なデータ型を手動で選択する必要があります。
5. 「**ビジネス・オブジェクトの追加時に追加構成設定のプロンプトを出す**」チェック・ボックスを選択します。 これにより、作成するビジネス・オブジェクトのリストにデータベース・オブジェクトを追加すると、そのオブジェクトに関してユーザーが構成可能なすべてのアプリケーション固有情報を入力するよう求められます。例えば、このオプションを選択した場合は、ビジネス・オブジェクトの単純親子階層を作成するプロセスが、順を追ってウィザードで示されます。 2 つの異なるテーブルの属性を参照する 2 つの属性をテーブル・ビジネス・オブジェクトが持つ (つまり、2 つの親ビジネス・オブジェクトを持つ) 階層が必要な場合は、アセンブリー・エディターで構成を実行します。このエディターは、WebSphere Integration Developer から起動されるツールです。また、データベースで外部キー参照が定義されている場合は、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。

   重要: このオプションを選択しない場合、ウィザードは必須情報のみを入力するようプロンプトを出します。アセンブリー・エディターを使用して、ビジネス・オブジェクトの構成を実行する必要があります。また、データベースで外部キー参照を定義していない場合、アダプターは親子関係を自動的には生成しません。
6. 「**OK**」をクリックして、クエリーへの変更を保存します。
7. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**クエリーの実行 (Run Query)**」をクリックします。これによりこのクエリーを使用してデータベース・オブジェクトがディスカバーされます。 標準的な照会の実行結果を次の図に示します。



「**検出済みオブジェクト**」ペインには、ディスカバーされたオブジェクトがリストされます。テーブル、ビュー、シノニム/ニックネームは、スキーマ名によってソートされます。

8. 「**検出済みオブジェクト**」リストで、スキーマ・ノードおよびその下の「**テーブル**」、「**ビュー (Views)**」、「**シノニム - ニックネーム (Synonyms - Nicknames)**」の各ノードを展開するには、「**+**」 (正符号) をクリックします。これにより、ウィザードによってディスカバーされたデータベース・オブジェクトが表示されます。

タスクの結果

アダプターを使用してアクセスできるデータベース・オブジェクトがウィザードによってディスカバーされました。

次のタスク

外部サービス・ウィザードでの作業を続行します。次の手順では、モジュールで使用するオブジェクトの選択、各ビジネス・オブジェクトの構成、およびビジネス・オブジェクトの階層の作成を行います。

属性に関するアプリケーション固有情報

ビジネス・オブジェクト属性のアプリケーション固有情報 (ASI) は、属性が単純属性であるか、子ビジネス・オブジェクト (または子ビジネス・オブジェクトの配列) を表す属性であるかによって異なります。子を表す属性のアプリケーション固有情報は、親子関係が子に格納されるか親に格納されるかによっても異なります。

単純属性のアプリケーション固有情報

単純属性では、アプリケーション固有情報の形式は、いくつかのパラメーターとその値で構成されています。単純属性に必要な唯一のパラメーターは列名です。単純属性のアプリケーション固有情報については、表 12で説明します。

表 12. 単純属性のアプリケーション固有情報

パラメーター	タイプ	説明	デフォルト値
BLOB	Boolean	この属性に対応するデータベース列が BLOB データ型であるかどうかを示します。BLOB データの表示中、アダプターはバイト数を 16 進値で表示します。属性の型は hexBinary です。 True に設定されている場合、列のデータ型は BLOB です。	なし
ByteArray	Boolean	列がバイナリー・データ型であるかどうかを示します。True の場合、アダプターはデータベースでバイナリー・データを読み取りおよび書き込みし、そのデータをストリングとしてアプリケーション・サーバーに送信します。アダプターは、ビジネス・オブジェクトにバイナリー・データを設定します。属性の型は hexBinary です。	False
ChildBOType	String	属性が複合データ型の場合は、このアプリケーション固有情報を使用して実際の型を指定します。 • Struct • Array • ResultSet	なし
ChildBOTypeName	String	ChildBOType アプリケーション固有情報の値が Struct または Array の場合、このパラメーターはユーザー定義タイプの名前を表します。この値は、大/小文字の区別があります。	
CLOB	Boolean	この属性に対応するデータベース列が CLOB データ型であるかどうかを示します。この値は、String 型の属性にのみ適用されます。 True の場合は、列のデータ型は CLOB です。 CLOB 属性は String 型で、その長さを使用して CLOB の長さを定義します。	なし
ColumnName	String	この属性に対応するデータベース列の名前。 これが唯一の必須パラメーターです。	なし

表 12. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
CopyAttribute	String	同一ビジネス・オブジェクトまたは親ビジネス・オブジェクト内から別の属性名を参照するユーザー指定値。 アプリケーション固有情報で設定されている値が、同一ビジネス・オブジェクト内の別の属性の名前を参照している場合、アダプターは、Create 操作でビジネス・オブジェクトをデータベースに追加する前に、その別の属性の値を使用してこの属性の値 (アプリケーション固有情報が定義されている属性) を設定します。 例えば、テーブルの新しい行の `contact` 列に、email 列と同じ値を組み込むには、`contact` 属性の CopyAttribute パラメーターに `email` を設定します。 値から子ビジネス・オブジェクトの属性を参照することはできませんが、親ビジネス・オブジェクト内の属性は参照できます。このためには、属性名の前に 2 つのピリオドを付加します。例えば、親ビジネス・オブジェクト内の `ccode` 属性を参照するには、`..ccode` と指定します。 アプリケーション固有情報にこのパラメーターを指定しないと、アダプターは、別の属性から値をコピーせずに現行属性の値を使用します。	なし
DateType	String	対応するエレメントが日付、時刻、またはタイム・スタンプであることを指定します。次の値のいずれかを指定してください。 • Date • Time • Timestamp DateType 型の属性の値を設定するときには、次の形式で設定します。 • Date には、*yyyy-MM-dd* を使用します。 • Time には、*hh:mm:ss* を使用します。 • Timestamp には、*yyyy-MM-dd hh:mm:ss.fffffffff* を使用します。 **注:** java.sql.Timestamp.valueOf () メソッドにより、JDBC タイム・スタンプ形式が Timestamp 値に変換され、その値がデータベースに挿入されます。java.sql.Timestamp.toString() メソッドは、データベースにある Timestamp 値をストリングに変換します。アダプターは、データベースに含まれる Timestamp 値を使用します。 Timestamp メソッドについての詳しい説明は、Sun の Web サイト (http://java.sun.com/j2se/1.5.0/docs/api/) で、 *Timestamp* を検索し、参照してください。	なし

表 12. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
DateFormat	String	Date、Time、および Timestamp の各データ型の形式をカスタマイズできます。アダプターは、Date、Time、または Timestamp の SQL データ型をストリングに変換する (およびその逆の変換を行う) 必要があるときに、このパラメーターを使用します。	なし
DecimalScale	Int	10 進データ型の位取りを指定します。例えば、`unscaledVal` × 10^{-scale} です。	なし
Dummy	Boolean	ダミー列を示します。`True` の場合、ダミー列の値は更新されず、データベースにも挿入されません。このアプリケーション固有情報を使用するのは、 1 つの列に複数の ForeignKey 値を構成したいときです。	なし

FixedChar	Boolean	テーブル内の列が VARCHAR 型ではなく CHAR 型である場合に、属性を固定長とするかどうかを指定します。例えば `true` に設定すると、ある特定の属性が CHAR 型の列にリンクされている場合、アダプターはデータベースを照会するときに、属性値をその属性の最大長までブランクで埋めます。
		
		ビジネス・オブジェクトの XSD ファイルでは、このパラメーターは手動で更新する必要があります。XML またはテキスト・エディターを使用してビジネス・オブジェクトを開いて XSD ファイルを編集し、以下の 2 つの変更を行ってください。
		1. オブジェクト属性の `<element>` タグにデフォルトで追加された `type="string"` を削除します。
		2. 次の例に示すように、`</element>` の前に新規に `<simpletype>` セクションを追加します。
		`<xsd:simpleType>`
		`<xsd:restriction base="xsd:string">`
		`<xsd:maxLength value="10"/>`
		`</xsd:restriction>`
		`</xsd:simpleType>`
		
		オブジェクト定義を保存して、更新後に XSD ファイル内で検証エラーが発生しないことを確認してください。
		
		この表の下にある、『ビジネス・オブジェクトの XSD ファイル内の FixedChar パラメーターの例』のセクションを参照してください。
			`false`

表 12. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
ForeignKey	String	このプロパティーの値は、親子関係が親ビジネス・オブジェクトに格納されるか、子ビジネス・オブジェクトに格納されるかによって異なります。 関係が親に格納される場合、子ビジネス・オブジェクトのタイプと、外部キー *(Child_BO_name/ Child_Property_Name)* として使用される子ビジネス・オブジェクト内の属性の名前の両方がこの値に含まれます。 この関係が子に保管される場合は、外部キーとして使用される親ビジネス・オブジェクト内の属性の名前のみを含むように値を設定します。 属性が外部キーではない場合は、アプリケーション固有情報にこのパラメーターを含めないでください。	なし
OrderBy	String	値が指定され、属性が子ビジネス・オブジェクト内に存在する場合は、アダプターは、検索照会の ORDER BY 文節でその属性の値を使用します。 アダプターは、子ビジネス・オブジェクトを昇順 (ASC) または降順 (DESC) で検索することができます。このパラメーターがアプリケーション固有情報に含まれていない場合、アダプターは検索順序を指定しません。	なし
PrimaryKey	Boolean	この属性に関連付けられている列が、データベース内の対応するテーブルの基本キーである場合、PrimaryKey パラメーターは `True` に設定されます。	なし
SPParameterType	String	ストアード・プロシージャーのタイプを指定します。 使用可能な値は、以下のとおりです。 • `IP` (入力のみ) • `OP` (出力のみ) • `IO` (入出力) • `RS` (結果セット)	なし

表 12. 単純属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
UniqueIdentifier (UID)	String	アダプターは、ビジネス・オブジェクトの固有 ID の生成に、このパラメーターを使用します。シーケンスと ID 列の生成がサポートされます (ID 列は、Informix では、*シリアル列* と呼ばれています)。DB2 は、シーケンスと ID 列の両方をサポートします。 ID 列は、データベースでテーブルの各行に自動的に固有の数値を生成する方法を提供します。 ID 列は、DB2 および Microsoft SQL Server の場合に定義でき、Informix の場合はシリアル列が定義できます。 このパラメーターの形式を以下に示します。 `UID=AUTO|Sequence_Name` DB2 または Microsoft SQL Server データベースのいずれかのテーブルに対してディスカバリー・プロセスを実行する場合、UID (固有 ID) 属性を手動で AUTO に設定する必要があります (例えば、`<UID>AUTO</UID>`)。 **注:** UID (固有 ID) 属性を手動で AUTO に設定する要件は、DB2 および Microsoft SQL Server の ID 列に特有のものです。この要件は、Informix のシリアル列には当てはまりません。Informix の場合、シリアル列の UID 属性は自動生成され、 <UID>SERIAL</UID> または <UID>SERIAL8</UID> のいずれかになります。 ID 列と同様、シーケンスも数値の自動生成に使用されます。データベースによるシーケンスおよび ID 列の使用について詳しくは、ご使用のデータベースの資料を参照してください。 シーケンスの場合、UID 属性にシーケンス名を設定します。シーケンスは、DB2 および Oracle データベースで定義できます。 属性で固有 ID が必要とされない場合は、このパラメーターをアプリケーション固有情報に含めないでください。	なし
XML	Boolean	データベース内のテーブル列が XML 型のものである場合、XML パラメーターは `True` に設定されます。	なし

属性のアプリケーション固有情報の形式は、XSD ファイルの以下の実例セクションに示します。

XSD ファイルのセクション例

```
   <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
   <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
   <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<xsd:simpleType>
```

```
   <xsd:restriction base="xsd:string">
   <xsd:maxLength value="10"/>
   </xsd:restriction>
</xsd:simpleType>
</element>
<element name="custCode" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
   <jdbcasi:ForeignKey>custinfoObj/custCode</jdbcasi:ForeignKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="firstName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>fname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lastName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>lname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
```

ビジネス・オブジェクトの XSD ファイル内の FixedChar パラメーターの例

```
<element name="primaryKey">
<annotation>
<appinfo source="WBI">
        <jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
                <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
                <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
                <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
        </jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<xsd:simpleType>
        <xsd:restriction base="xsd:string">
                <xsd:maxLength value="10"/>
        </xsd:restriction>
</xsd:simpleType>
</element>
```

子ビジネス・オブジェクトを参照する属性のアプリケーション固有情報

子ビジネス・オブジェクトを参照する属性には、2 つのアプリケーション固有情報パラメーターが使用されます (単純属性ではなく複合属性)。このアプリケーション固有情報を設定する場合は、表 13に示すパラメーターを指定します。

表 13. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報

パラメーター	タイプ	説明	デフォルト値
KeepRelationship	Boolean	True の場合、このパラメーターにより Update 操作中に子ビジネス・オブジェクトは削除されません。	なし

表 13. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報 (続き)

パラメーター	タイプ	説明	デフォルト値
Ownership	Boolean	このパラメーターは、子ビジネス・オブジェクトが親によって所有されることを指定します。True の場合、子ビジネス・オブジェクトに対して Create、Update、および Delete 操作が許可されます。False の場合、どの更新も子ビジネス・オブジェクトには適用できません。親が作成されると、データベース内で関係の整合性が保持されるように、子が存在するかどうかが検証されます。	なし

ビジネス・オブジェクトの XSD ファイル内の ownership の例

```
<element minOccurs="0" name="addressObj" type="bons0:OutboundRtasserAddress"
 maxOccurs="unbounded">
  <annotation>
    <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
      <jdbcasi:Ownership>true</jdbcasi:Ownership>
    </jdbcasi:JDBCAttributeTypeMetadata>
    </appinfo>
  </annotation>
</element>


<element minOccurs="0" name="custinfoObj" type="bons1:OutboundRtasserCustinfo"
maxOccurs="1">
 <annotation>
  <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:Ownership>false</jdbcasi:Ownership>
   </jdbcasi:JDBCAttributeTypeMetadata>
  </appinfo>
 </annotation>
</element>
```

単一および複数カーディナリティー子ビジネス・オブジェクトの XSD ファイルのもう 1 つの例

単一または複数カーディナリティー子ビジネス・オブジェクトの XSD 定義ファイルの例をここに示します。エレメント custInfoObj は単一カーディナリティー子ビジネス・オブジェクトで、addressObj は複数カーディナリティー子ビジネス・オブジェクトです。

```
<element name="addressObj" minOccurs="1" type="Address:Address"
maxOccurs="unbounded">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>true</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
                        <element name="custInfoObj" minOccurs="0" type=
"CustInfo:CustInfo" maxOccurs="1">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
```

```
"urn:app:jdbc:asi">
                                <pasi:Ownership>false</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
```

操作のアプリケーション固有情報

アダプターは、操作レベルのアプリケーション固有情報を使用してデータベース内の情報の取得や更新などの操作を実行します。アダプターは、ビジネス・オブジェクトでの指定に従って、SQL 照会、ストアード・プロシージャー、またはストアード関数を使用してデータベース表を取得および更新します。

ビジネス・オブジェクトにストアード・プロシージャーまたはストアード関数を追加する場合、操作レベルのアプリケーション固有情報 (ASI) を、表 14に指定されているとおりに設定します。

表 14. 操作に関するアプリケーション固有情報

操作 ASI の StoredProcedure のパラメーター・エレメント	**ウィザードによって設定**	**説明**
Parameters	はい	ストアード・プロシージャーのパラメーターをリストします。
PropertyName	はい	選択したビジネス・オブジェクト属性の名前に設定します。
ResultSet	いいえ	ストアード・プロシージャーから結果セットが返される場合は、ビジネス・オブジェクト定義でこのパラメーターを `True` に設定します。
ReturnValue	はい	ストアード・プロシージャーに戻り値がある場合は、このパラメーターには次のいずれかの値が設定されます。 • ストリング `RS`。この値は、プロシージャーから結果セットが戻され、この結果セットを使用して、このビジネス・オブジェクトに対応する複数カーディナリティー・コンテナーが作成されることを示します。 • ビジネス・オブジェクト属性の名前。この値は、プロシージャーから戻される値が、実行時にビジネス・オブジェクトの特定の属性に割り当てられることを示します。 属性が別の子ビジネス・オブジェクトである場合、アダプターはエラーを返します。
StoredProcedure	はい	ストアード・プロシージャー名に設定します。
StoredProcedureType	はい	タイプのリストから選択します。 有効なストアード・プロシージャー・タイプについては、ストアード・プロシージャー・タイプを参照してください。

表 14. 操作に関するアプリケーション固有情報 (続き)

操作 ASI の StoredProcedure のパラメーター・エレメント	**ウィザードによって設定**	**説明**
タイプ	はい	ストアード・プロシージャーのパラメーターのタイプを設定します。使用可能な値は、以下のとおりです。 • IP (入力のみ) • OP (出力のみ) • IO (入出力) • RS (結果セット)

第 5 章 テーブル間の親子関係の自動ディスカバリー

データベースで外部キー参照を定義した場合、ビジネス・オブジェクトを選択して構成すると、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。

ビジネス・オブジェクト

ビジネス・オブジェクトとは、データ、データ上で実行されるアクション、およびデータを処理するための追加の指示 (存在する場合) で構成される構造体のことです。WebSphere Adapter for JDBC は、ビジネス・オブジェクトを使用して、データベースのテーブルとビュー、データベース照会、ストアード・プロシージャー、およびストアード関数の結果を表現します。ビジネス・オブジェクトにより、データベースのオブジェクトの階層を作成し、無関係なテーブルをグループ化できます。コンポーネントはビジネス・オブジェクトを使用してアダプターと通信します。

アダプターによるビジネス・オブジェクトの使用法

統合アプリケーションは、ビジネス・オブジェクトを使用してデータベースにアクセスします。アダプターは、Outbound 要求のビジネス・オブジェクトを、データベースへアクセスするための JDBC API 呼び出しに変換します。Inbound イベントの場合、アダプターはイベントのデータをビジネス・オブジェクトに変換し、このビジネス・オブジェクトがアプリケーションに戻されます。

アダプターは、ビジネス・オブジェクトを使用してデータベース内の次のタイプのオブジェクトを表現します。

- テーブルとビュー
- シノニムとニックネーム
- ストアード・プロシージャーとストアード関数

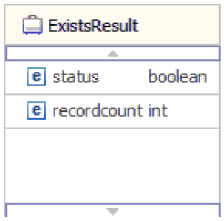
一部のビジネス・オブジェクトは、データベース・オブジェクトを表現しません。これらのビジネス・オブジェクトには、以下のものがあります。

- バッチ SQL ビジネス・オブジェクト。一連のユーザー定義の insert ステートメント、update ステートメント、および delete ステートメントを表します。
- クエリー・ビジネス・オブジェクト。データベースに対して実行されるユーザー定義 SQL 照会を表します。
- ラッパー・ビジネス・オブジェクト。このオブジェクトにより、無関係なテーブル・オブジェクトとビュー・オブジェクトを単一のビジネス・オブジェクトにグループ化でき、複数のストアード・プロシージャーを単一のビジネス・オブジェクトにグループ化することができます。

アダプターは、出力に一部のビジネス・オブジェクトを使用します。これらのビジネス・オブジェクトには、以下のものがあります。

- コンテナー・ビジネス・オブジェクト。RetrieveAll 操作からの出力が入ります。

- ExistsResult ビジネス・オブジェクト。これには、Exists 操作からの出力が入ります。



ビジネス・オブジェクト内でのデータの表現方法

テーブルまたはビュー・ビジネス・オブジェクトの場合

テーブルまたはビューの各列は、テーブル・ビジネス・オブジェクトまたはビュー・ビジネス・オブジェクトの単純属性により表現されます。*単純属性* とは、String、Integer、または Date などの単一値を表す属性です。その他の属性は、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表します。

同じビジネス・オブジェクトに含まれる単純属性を、別々のデータベース表に格納することはできませんが、次の状況が考えられます。

- データベース表に、対応するビジネス・オブジェクトに含まれる単純属性の数よりも多くの列が含まれる場合があります。つまり、データベースの列の一部が、ビジネス・オブジェクト内に表されていません。アプリケーションによるビジネス・オブジェクトの処理で必要な列のみを実際の設計に含める必要があります。
- ビジネス・オブジェクトに、対応するデータベース表に含まれる列の数よりも多くの単純属性が含まれる場合があります。つまり、ビジネス・オブジェクト内の属性の一部が、データベース内に表されていません。データベース内に列を持たない属性は、アプリケーション固有情報を持っていないか、デフォルト値が設定されているか、またはストアード・プロシージャーかストアード関数のパラメーターです。
- ビジネス・オブジェクトは、複数のデータベース表にまたがるビューを表すことができます。アダプターでは、Create、Update、および Delete 操作など、データベースに対する変更によって起動されるイベントを処理するときに、このようなビジネス・オブジェクトを使用できます。ただし、ビジネス・オブジェクトの要求を処理する場合には、Retrieve および RetrieveAll 要求に対してのみ、このようなビジネス・オブジェクトを使用できます。

テーブル・ビジネス・オブジェクトには、対応するデータベース表に基本キーがない場合でも、常に基本キーが設定されています。アダプターは、テーブル・ビジネス・オブジェクトを取得するときに、基本キー属性で指定される列を使用します。データベースで外部キー参照が定義されている場合、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。例えば、親テーブルである CUSTOMER テーブルと子テーブルである ADDRESS テーブルがあるとします。データベース内で ADDRESS から CUSTOMER への外部キー参照を定義した場合、アダプターは自動的に親子関係をディスカバーし、外部キー参照を「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに表示します。外部キー参照が CUSTOMER から ADDRESS への場合、

アダプターは自動的に「**単一カーディナリティー**」チェック・ボックスを選択し、外部キー参照を表示します。1 つのテーブルに複数の外部キー参照が定義されている場合、アダプターは外部キー関係を 1 つだけ生成します。

アダプターでは、複合の (つまり複数の) 基本キーが設定されている表をサポートしています。データベースに基本キーが 1 つ以上存在する場合、ウィザードは、テーブル・ビジネス・オブジェクトのそれらの列に基本キー・プロパティーを設定します。データベース表に基本キーが存在しない場合、外部サービス・ウィザードでは、そのビジネス・オブジェクトをディスカバーして構成するときに基本キー情報の入力を求めるプロンプトが出されます。例えば CUSTOMER (pkey1, pkey2) > ADDRESS (fkey1, fkey2) といった複合基本キーがある場合、アダプターは ADDRESS (fkey1) と CUSTOMER (pkey1)、ADDRESS (fkey2) と CUSTOMER (pkey2) をそれぞれ関連付けます。シーケンス列や ID 列など、固有のデータを持つ列を指定してください。ID 列 (Informix では*シリアル列* と呼ばれる) は、データベースでテーブルの各行に自動的に固有の数値を生成する方法になります。テーブルには、ID 属性で定義される単一の列を作成できます。 ID 列の例として、オーダー番号、従業員番号、ストック番号、および問題番号などがあります。 ID 列は、DB2、Informix および Microsoft SQL Server のテーブルにのみ定義できます。

注: DB2 または Microsoft SQL Server データベースのいずれかの表に対してディスカバリー・プロセスを実行する場合に、その表で 1 つの列を ID 列として定義している場合、その表に対して生成されたビジネス・オブジェクトには、ID 列の固有 ID 属性は含まれていません。この場合、アプリケーション固有情報に属性を手動で追加することにより、生成されたビジネス・オブジェクトを編集する必要があります。これは、WebSphere Integration Developerのアセンブリー・エディターによって行うことができます。Informix データベースのテーブルに対してディスカバリー・プロセスを実行した場合は、固有 ID の属性を手動で追加する必要はありません。Informix の場合、生成されたビジネス・オブジェクトには、シリアル列の固有 ID 属性が含まれています。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

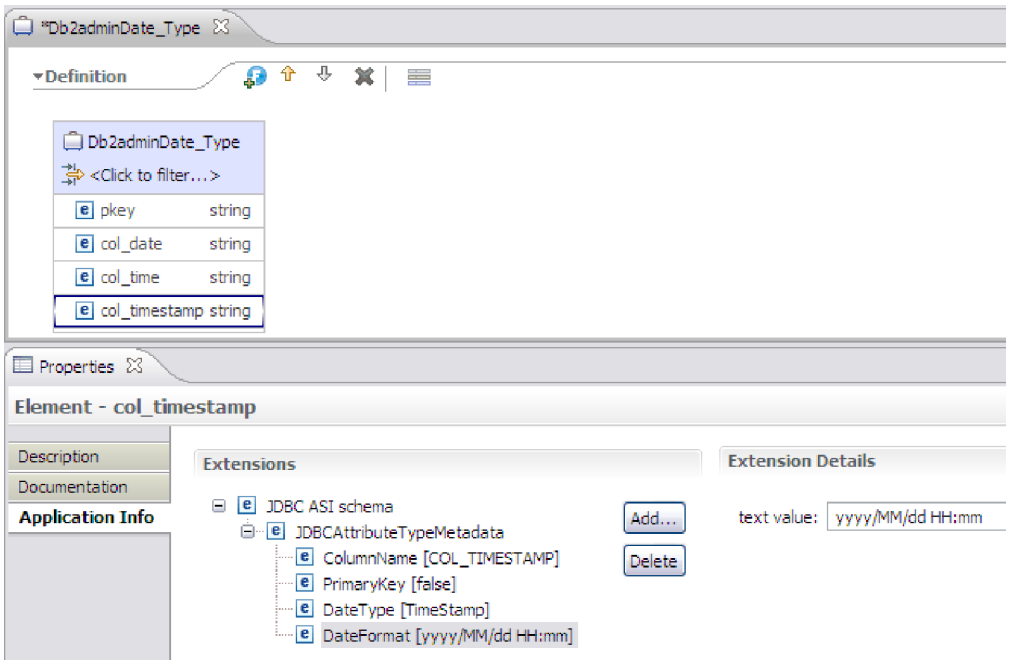


図 21. DateFormat アプリケーション固有情報とカスタマイズされた形式

テーブル・ビジネス・オブジェクトおよびビュー・ビジネス・オブジェクトは、Create、Update、Delete、Retrieve、RetrieveAll、Exists、および ApplyChanges の Outbound 操作をサポートします。階層型テーブル・ビジネス・オブジェクトに Exists 操作を実行すると、トップレベルのビジネス・オブジェクトのみが照会されます。

113 ページの図 22 に、ビジネス・オブジェクト・エディターに表示されたテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとに 1 つの属性が設定されています。表には子ビジネス・オブジェクトがないため、属性はすべて単純属性です。

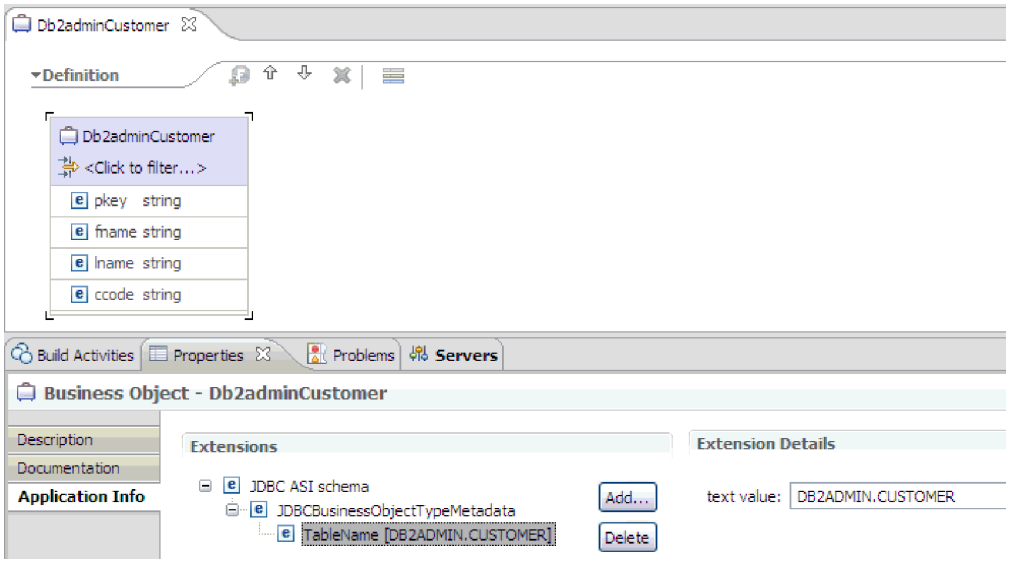


図 22. 子なしのテーブル・ビジネス・オブジェクト:

図 23 に、子テーブル・ビジネス・オブジェクトが 1 つあるテーブル・ビジネス・オブジェクトを示します。このビジネス・オブジェクトでは、データベース表の列ごとの単純属性に加えて、子ビジネス・オブジェクトを指す複合属性が設定されています。

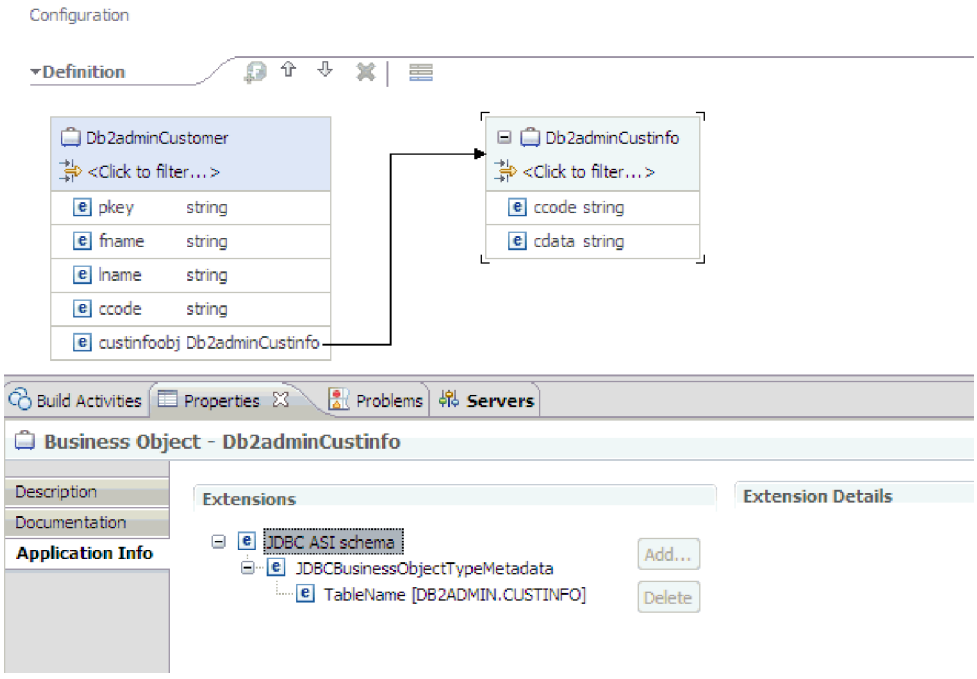


図 23. 子ビジネス・オブジェクトを 1 つ持つテーブル・ビジネス・オブジェクト:

Oracle データベースの場合、アダプターは、テーブル・ビジネス・オブジェクトで配列、テーブル、構造体、ネストされた構造体などのユーザー定義型または複合データ型をサポートします。これらの型に対しては、型名および子属性の詳細が自動的にディスカバーされて、表示されます。アダプターでは、テーブル・ビジネス・オブジェクトの子ビジネス・オブジェクトとしてこれらのデータ型が処理されま

す。

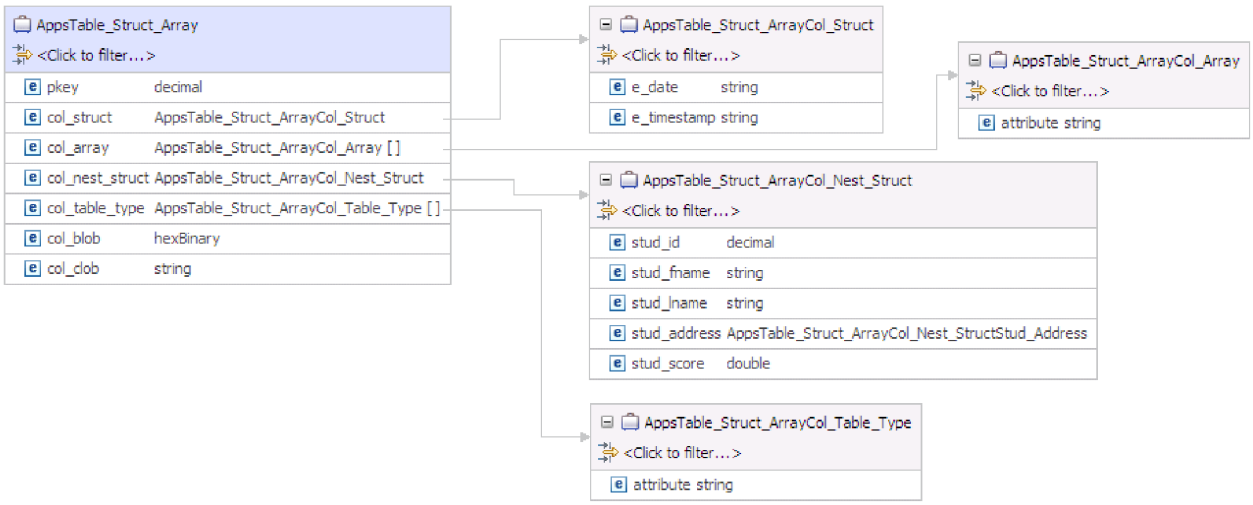


図 24. 列にユーザー定義型または複合型がある Oracle テーブル・ビジネス・オブジェクト

ストアード・プロシージャー・ビジネス・オブジェクトとストアード関数ビジネス・オブジェクトの場合

ストアード・プロシージャーまたはストアード関数のビジネス・オブジェクトでは、ストアード・プロシージャーまたはストアード関数のすべての入力パラメーターおよび出力パラメーターに、ビジネス・オブジェクトに対応する属性があります。入力または出力パラメーターのいずれかが、配列や構造体などの複合型である場合、対応するビジネス・オブジェクト属性は、配列または構造体の属性を含む子ビジネス・オブジェクトを持つ子ビジネス・オブジェクト型です。ストアード・プロシージャーが結果セットを戻した場合、戻された結果セットの属性を格納する子ビジネス・オブジェクトが作成されます。

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.ffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

ストアード・プロシージャーとストアード関数のビジネス・オブジェクトは、Execute Outbound 操作をサポートします。

ストアード・プロシージャー・ビジネス・オブジェクトの構造を下のサンプル・ファイルに示します。ビジネス・オブジェクト ScottStrtValues および ScottStrtValuesStrt が、1 つの入力タイプと 2 つの出力タイプを持つストアード・プロシージャーから生成されます。出力パラメーターの 1 つは、構造体データ型です。外部サービス・ウィザードによって、構造体型のビジネス・オブジェクト ScottStrtValuesStrt が生成され、子オブジェクトとして親ビジネス・オブジェクト ScottStrtValues に追加されます。親ビジネス・オブジェクト内の構造体型の属性については、ChildBOType アプリケーション固有情報が Struct に設定され、型が構造体であることを示します。ChildBOTypeName アプリケーション固有情報は、データベース内のユーザー定義構造体型の値に設定されます。次の例は、ストアード・プロシージャーのスキーマを示しています。

ScottStrtValues ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvalues" xmlns:scottstrtvalues=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvalues" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt">
<import namespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt"
schemaLocation="ScottStrtvaluesStrt.xsd"/>
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvalues">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
<jdbcasi:MaxNumOfRetRS>0</jdbcasi:MaxNumOfRetRS>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>IP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="strt" type="scottstrtvaluesstrt:ScottStrtvaluesStrt"
```

```
minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
<jdbcasi:ChildBOType>STRUCT</jdbcasi:ChildBOType>
<jdbcasi:ChildBOTypeName>STRUCT1</jdbcasi:ChildBOTypeName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

ScottStrtValuesStrt ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="ScottStrtvaluesStrt">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="name" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="title" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="dept_num" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
```

```
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

クエリー・ビジネス・オブジェクトの場合

データベース照会のビジネス・オブジェクトは、照会を実行する SQL ステートメントと、照会に必要なパラメーターを定義します。クエリー・ビジネス・オブジェクトでは、Outbound 操作 RetrieveAll がサポートされています。

例えば、次の SELECT ステートメントを実行するクエリー・ビジネス・オブジェクトがあるとします。

```
select C.pkey, C.fname, A.city from customer C, address A
    WHERE (C.pkey = A.custid) AND (C.fname like ?)
```

疑問符 (?) は、照会の入力パラメーターを示します。照会には複数のパラメーターを指定できます。各パラメーターは、SELECT ステートメントでは疑問符で示されています。サンプル・クエリー・ビジネス・オブジェクトの属性を表 15 に示します。クエリー・ビジネス・オブジェクトには、抽出される列ごとの単純属性、パラメーターごとの単純属性、およびパラメーター置換の後も WHERE 節を保持する、照会の WHERE 節の「プレースホルダー・オブジェクト」があります。

表 15. クエリー・ビジネス・オブジェクトの属性

ビジネス・オブジェクト属性	**説明**
pkey	Customer 表のデータベース列 PKEY に対応
fname	Customer 表のデータベース列 FNAME に対応
city	Address 表のデータベース列 CITY に対応
parameter1	パラメーター。SELECT ステートメントの ? (疑問符) ごとに 1 つのパラメーターがあります。複数のパラメーターを持つ SELECT ステートメントでは、後続のパラメーターに parameter2、parameter3 のようにして名前が付けられます。
jdbcwhereclause	WHERE 節のプレースホルダー・オブジェクト

ビジネス・オブジェクトに Date、Time、または Timestamp のデータ型が含まれている場合、これらの型の形式は DateFormat アプリケーション固有情報でカスタマイズできます。日付の形式は、java.text.SimpleDateFormat に定義されたパターンに従っている必要があります。 SQL の Date、Time、または Timestamp をストリングに変換する (およびその逆の変換を行う) 必要があるときに、DateFormat アプリケーション固有情報でこれらの型がカスタマイズされている場合、アダプターはこのカスタマイズされた形式を使用します。例えば、日付を dd/MM/yy 形式で、タイム・スタンプを yyyy/MM/dd HH:mm 形式でそれぞれ指定できます。DateFormat アプリケーション固有情報を指定しない場合、アダプターはデフォルトの形式を使用します。Date 型のデフォルトの形式は、「yyyy-MM-dd」、Timestamp 型は、「yyyy-mm-dd hh:mm:ss.fffffffff」、Time 型は、「HH:mm:ss」です。

注: Timestamp 型の形式は、JDBC 規格で定義され、SimpleDateFormat パターンには従いません。

以下の図に、ビジネス・オブジェクト・エディターに表示されたサンプル・クエリーのビジネス・オブジェクトを示します。

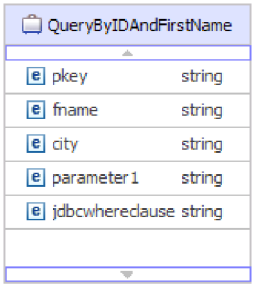


図 25. クエリー・ビジネス・オブジェクトの属性

この図は、クエリー・ビジネス・オブジェクト例のアプリケーション固有情報を示しています。SelectStatement アプリケーション固有情報には、SELECT ステートメントが含まれています。

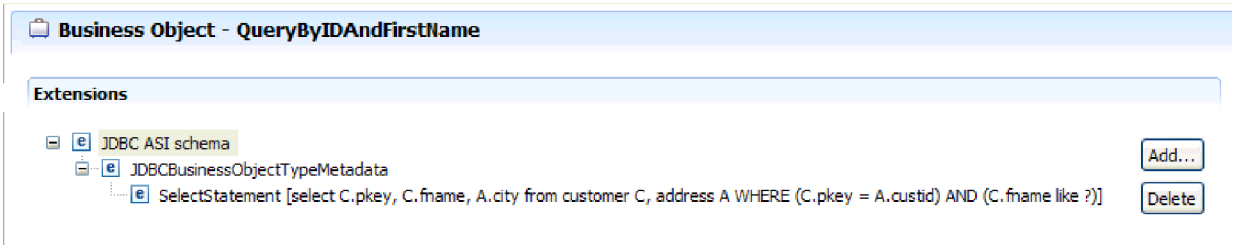


図 26. SELECT ステートメントは、ビジネス・オブジェクトのアプリケーション固有情報に保存されます。

Oracle データベースの場合、アダプターは、ビジネス・オブジェクトの照会結果で配列、テーブル、構造体、ネストされた構造体などの複合データ型をサポートします。アダプターでは、バッチおよびクエリーのビジネス・オブジェクトにおいて、これらの複合型をパラメーターとしてサポートしていません。

バッチ SQL ビジネス・オブジェクトの場合

バッチ SQL ビジネス・オブジェクトは、データベース・アクションを実行する INSERT、UPDATE、および DELETE SQL ステートメントと、そのステートメントで必要とされるパラメーターを定義します。バッチ SQL ビジネス・オブジェクトでは、Outbound 操作 Execute がサポートされています。

例えば、次の INSERT および DELETE ステートメントを実行するバッチ SQL ビジネス・オブジェクトがあるとします。

```
Insert into customer (pkey,ccode,fname,lname) values(?,?,?,?);
Delete From Customer where pkey=?
```

各疑問符 (?) は、ステートメントのパラメーターを示します。バッチ SQL ビジネス・オブジェクト内の各ステートメントには複数のパラメーターを指定できます。各パラメーターは、ステートメントでは疑問符で示されています。バッチ SQL ビ

ジネス・オブジェクトには複数のステートメントを含めることができ、それぞれが独自のパラメーター・セットを持ちます。図 27 に、それぞれが 1 つ以上のパラメーターを持つ INSERT ステートメントと DELETE ステートメントが定義されている、バッチ SQL ビジネス・オブジェクトのビジネス・オブジェクトの形式を示します。

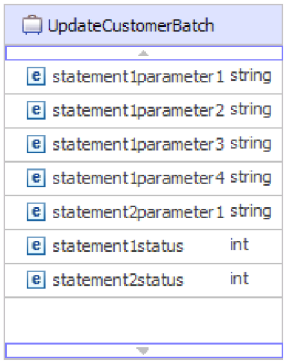


図 27. SQL ステートメントが 2 つのバッチ SQL ビジネス・オブジェクト

このビジネス・オブジェクトでは、statement1parameter1 や statement2parameter1 などの各ステートメントのパラメーターごとに属性が設定されています。また、statement1status や statement2status などの、各ステートメントの状況に関する属性もあります。ステートメント自体は、図 28 に示すように、ビジネス・オブジェクトについてのアプリケーション固有情報として格納されます。

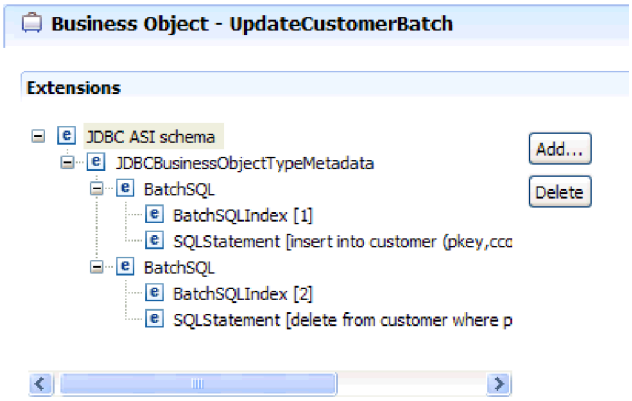


図 28. バッチ SQL ビジネス・オブジェクトのアプリケーション固有情報

ラッパー・ビジネス・オブジェクトの場合

ラッパー・ビジネス・オブジェクトにより、無関係の表の操作とビジネス・オブジェクトの表示を 1 回の操作で行うことができます。ラッパー・ビジネス・オブジェクトは、Create、Delete、Retrieve、および Update の Outbound 操作をサポートします。

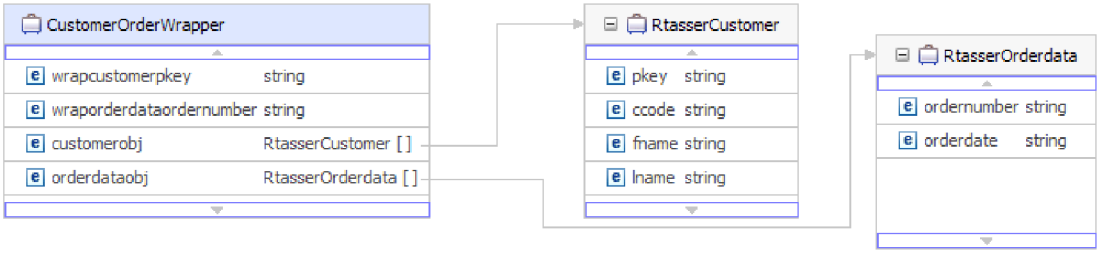


図 29. 2 つのテーブル・ビジネス・オブジェクトを含むラッパー・ビジネス・オブジェクト

ラッパー・ビジネス・オブジェクトには、子ビジネス・オブジェクトそれぞれの基本キーの単純属性が含まれています。フィールドの名前は「wrap」というストリングで、その後に、データベース表名とその表の基本キーの列名が続きます。ラッパー・ビジネス・オブジェクトには、テーブル・ビジネス・オブジェクトごとの複合属性も含まれています。属性の名前は、ストリング「obj」を付加した表名です。複合属性のタイプは、対応するテーブル・ビジネス・オブジェクトの名前です。

ビジネス・グラフ

アダプターの構成時に、ビジネス・グラフを生成するオプションを選択することもできます。バージョン 6.0.2 では、トップレベルの各ビジネス・オブジェクトがビジネス・グラフに含まれていますが、このビジネス・オブジェクトには、実行する操作に関する追加情報を指定するために、バージョン 6.0.2 でアプリケーションが使用できる動詞が組み込まれています。バージョン 7.0 では、ビジネス・グラフが必要になるのは以下の状況に限られます。

- Outbound ApplyChanges 操作を使用する必要がある場合
- バージョン 7.0 より前のバージョンの WebSphere Integration Developer で作成されたモジュールにビジネス・オブジェクトを追加する場合

ビジネス・グラフが存在する場合、ビジネス・グラフは処理されますが、ApplyChanges 以外のすべての操作で動詞は無視されます。

ビジネス・オブジェクトの作成方法

ビジネス・オブジェクトを作成するには、WebSphere Integration Developer から起動される外部サービス・ウィザードを使用します。このウィザードにより、データベースに接続し、データベース・オブジェクトがディスカバーされ、表示されます。ビジネス・オブジェクトを作成するデータベース・オブジェクトを選択します。例えば、調べるスキーマを指定します。指定されたスキーマで、テーブル、ビュー、ストアード・プロシージャー、ストアード関数、シノニム、およびニックネームを選択します。また、ビジネス・オブジェクトを追加で作成できます。例えば、データベースに対して実行されるユーザー定義の SELECT、INSERT、UPDATE、または DELETE ステートメントの結果を表すビジネス・オブジェクトを作成できます。このウィザードでは、親子関係と、無関係なビジネス・オブジェクトをまとめるラッパーを使用してビジネス・オブジェクト階層を作成できます。

必要なビジネス・オブジェクトを指定し、これらのオブジェクトの階層を定義すると、ウィザードにより、選択されたオブジェクトを表すビジネス・オブジェクトが生成されます。また、アダプターに必要なその他の成果物も生成されます。

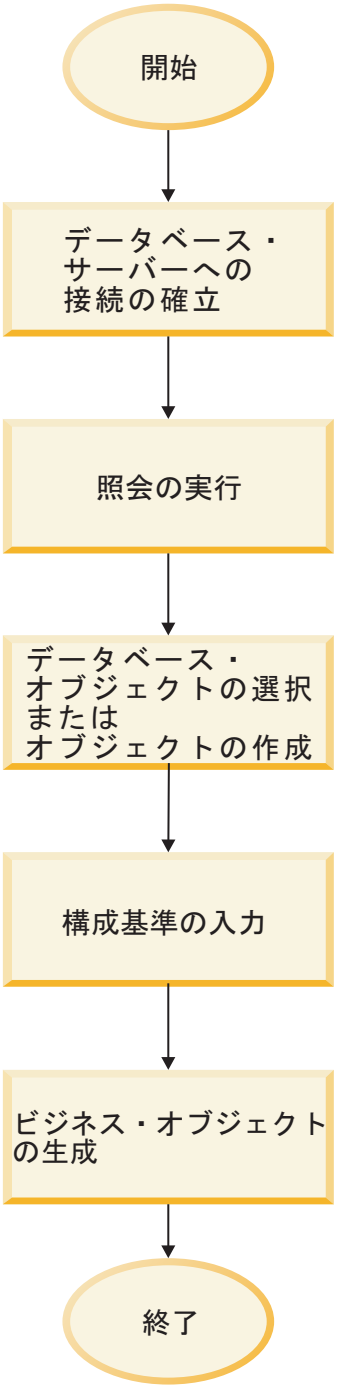


図 30. ビジネス・オブジェクトの作成方法

場合によっては、ウィザードで親子関係を完全に構成できないこともあります。これらの関係の場合は、WebSphere Integration Developer から起動するビジネス・オブジェクト・エディターを使用して、ウィザードによって作成されたビジネス・オブジェクト階層の定義を変更または完了します。詳しくは、WebSphere Integration Developer インフォメーション・センター (リンク: http://publib.boulder.ibm.com/infocenter/dmndhelp/v7r0mx/index.jsp) で、ビジネス・オブジェクト・エディターによるビジネス・オブジェクトの変更方法を参照してください。

Outbound 処理のデータベース・オブジェクトのディスカバー

データベースに接続した後は、データベース・オブジェクトを検索する照会を実行します。ディスカバーされたオブジェクトのツリーを参照して、データベース内のオブジェクトの構造を理解します。また、フィルターを使用して、参照したいデータベース・オブジェクトのみを表示します。ユーザー定義データベース・クエリーおよびユーザー定義バッチ SQL ステートメントとして作成するビジネス・オブジェクトの数を定義します。

始める前に

データベースのアクセスに必要なプログラムのデータ要件を知っていなければなりません。例えば、データベースに関する以下の情報が必要になります。

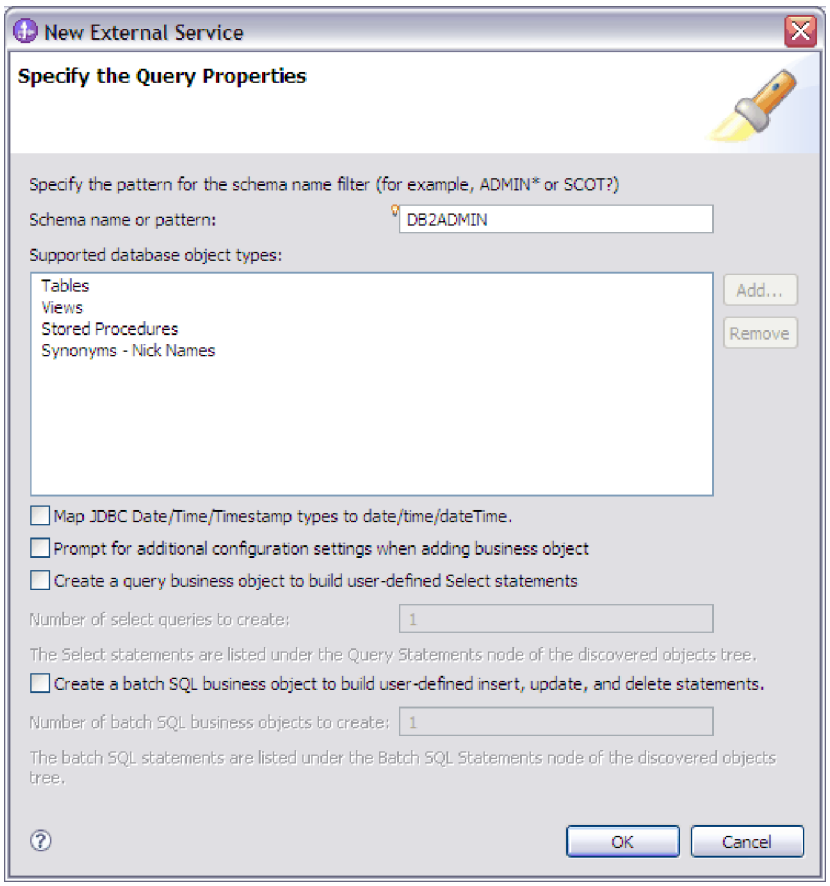
- モジュールはどのスキーマにアクセスする必要があるか
- これらのスキーマ内のどのタイプのデータベース・オブジェクトにアクセスする必要があるか
- どのテーブル、ビュー、シノニムまたはニックネーム、ストアード・プロシージャーまたはストアード関数にアクセスする必要があるか
- 作成する必要があるクエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトの数。パラメーター値およびそのパラメーターのサンプル・データベース値を含む

このタスクについて

この作業は、外部サービス・ウィザードの「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで開始します。

手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**照会の編集 (Edit Query)**」をクリックします。 照会プロパティーの指定ウィンドウが表示されます。



「照会プロパティーの指定」ウィンドウを使用して、次のタスクを実行します。

- データベース・スキーマのサブセットを検索することにより、検索時間を削減します。
- 1 つ以上のタイプのデータベース・オブジェクトを検索から除外します。
- データベース内の情報からは自動的に判別できないアプリケーション固有情報の入力をウィザードがユーザーに求めるようにします。
- 作成するクエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトの数を指定します。
- JDBC のデータ型 Date、Time、および Timestamp を date、time、および dateTime にマップします。

注: バージョン 6.1.x、フィックスパック 2 では、作成したいラッパー・ビジネス・オブジェクトの数もこのウィンドウで指定できました。バージョン 6.2.x では、ウィザードは後で wrapper 情報の入力を求めます。

2. 取得されるデータベース・スキーマの数を制限するには、「**スキーマ名またはパターン**」にスキーマの名前またはスキーマ名パターンを入力します。1 文字に対応させる場合は疑問符または下線 (? または _) を使用し、複数文字に対応させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。 クエリーを実行すると、そのストリングで始まるスキーマか、またはスキーマ名パターンに一致するスキーマのみが表示されます。スキーマ名パターン

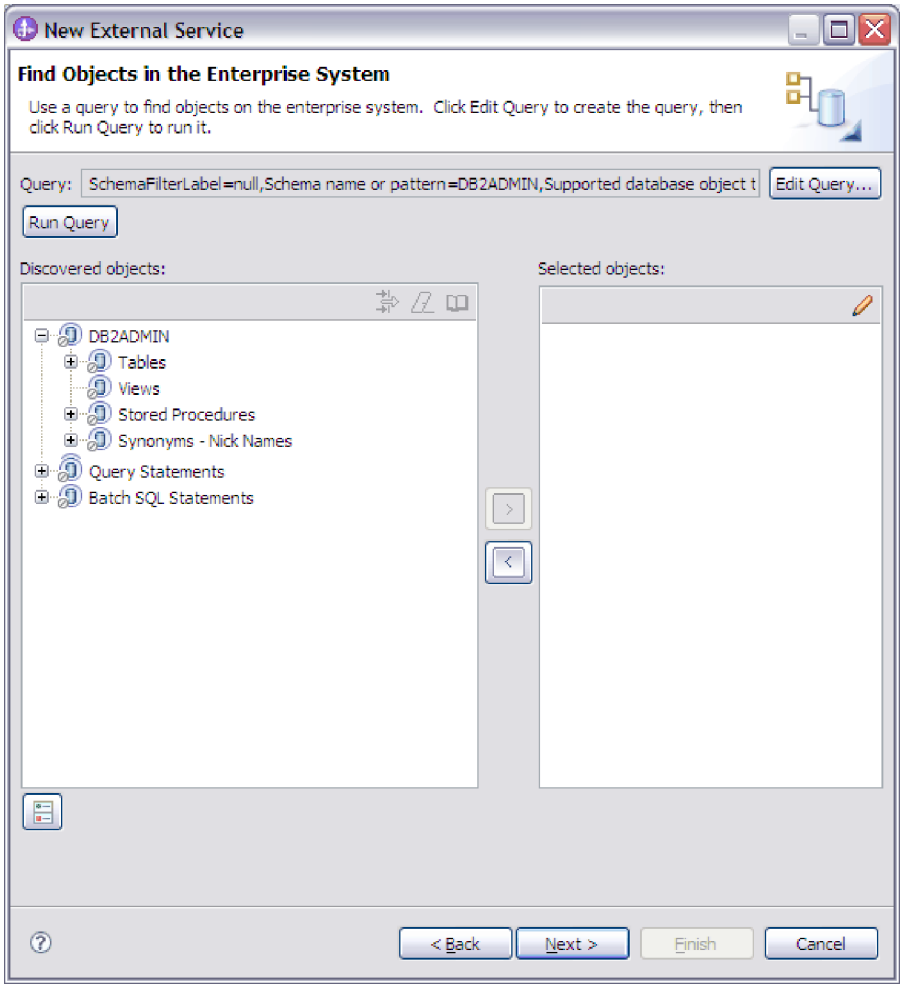
を指定しない場合は、データベース内のすべてのスキーマが表示されます。データベースに多数のスキーマがある場合は、フィルターを使用してディスカバリー・プロセスを高速化できます。

3. 検索から 1 つ以上のタイプのオブジェクトを除外するには、除外したいオブジェクトのタイプ (テーブル、ビュー、ストアード・プロシージャーおよびストアード関数、シノニムまたはニックネーム) を「**サポートされるデータベース・オブジェクト・タイプ**」フィールドで選択して、「**除去**」をクリックします。そのオブジェクト・タイプをもう一度追加するには、「**追加**」をクリックします。 特定のタイプのデータベース・オブジェクトにのみアクセスする必要がある場合は、必要のないオブジェクトを除外することで、ディスカバリー・プロセスを高速化できます。
4. データ型が Date、Time、および Timestamp であるテーブル、ストアード・プロシージャー、およびストアード関数の各オブジェクトは、デフォルトでは String データ型にマップされます。これらのオブジェクトを、JDBC ドライバーでサポートされる実際のデータ型 (Date、Time、Datetime など) にマップするには、「**JDBC の Date/Time/Timestamp の型を date/time/dateTime にマップ**」チェック・ボックスを選択します。

   注: デフォルトのデータ型マッピングは、JDBC ドライバーのバージョンによって異なります。例えば、Oracle JDBC ドライバーを使用する場合、Date データ型は、Date ではなく dateTime データ型にマップされます。このような場合は、「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウで、適切なデータ型を手動で選択する必要があります。
5. 「**ビジネス・オブジェクトの追加時に追加構成設定のプロンプトを出す**」チェック・ボックスを選択します。 これにより、作成するビジネス・オブジェクトのリストにデータベース・オブジェクトを追加すると、そのオブジェクトに関してユーザーが構成可能なすべてのアプリケーション固有情報を入力するよう求められます。例えば、このオプションを選択した場合は、ビジネス・オブジェクトの単純親子階層を作成するプロセスが、順を追ってウィザードで示されます。 2 つの異なるテーブルの属性を参照する 2 つの属性をテーブル・ビジネス・オブジェクトが持つ (つまり、2 つの親ビジネス・オブジェクトを持つ) 階層が必要な場合は、アセンブリー・エディターで構成を実行します。このエディターは、WebSphere Integration Developer から起動されるツールです。また、データベースで外部キー参照が定義されている場合は、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。

   重要: このオプションを選択しない場合、ウィザードは必須情報のみを入力するようプロンプトを出します。アセンブリー・エディターを使用して、ビジネス・オブジェクトの構成を実行する必要があります。また、データベースで外部キー参照を定義していない場合、アダプターは親子関係を自動的には生成しません。
6. ユーザー定義データベース照会を実行するビジネス・オブジェクトを作成するには、「**ユーザー定義の Select ステートメントを作成するためのクエリー・ビジネス・オブジェクトを作成する**」を選択してから、作成するクエリー・ビジネス・オブジェクトの数を入力します。 ここではビジネス・オブジェクトの数のみを指定します。このビジネス・オブジェクトについては、後で名前およびその他の詳細の入力をウィザードから求められます。

7. 一連の SQL ステートメントを実行するビジネス・オブジェクトを作成するには、**「ユーザー定義の insert、update、および delete ステートメントを作成するためのバッチ SQL ビジネス・オブジェクトを作成する」**を選択して、作成するバッチ SQL ビジネス・オブジェクトの数を入力します。 ここではビジネス・オブジェクトの数のみを指定します。このビジネス・オブジェクトについては、後で名前およびその他の詳細の入力をウィザードから求められます。
8. **「OK」**をクリックして、データベース・クエリーへの変更を保存します。
9. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、**「クエリーの実行 (Run Query)」**をクリックします。これによりこのクエリーを使用してデータベース・オブジェクトがディスカバーされ、クエリーおよびバッチ SQL ビジネス・オブジェクトのテンプレートが作成されます。 標準的な照会の実行結果を次の図に示します。



注: 有効期限が切れたデータベース接続を復元するには、外部サービス・ウィザードを再始動します。

「検出済みオブジェクト」ペインには、ディスカバーされたデータベース・オブジェクトがリストされます。

10. **「検出済みオブジェクト」**リストで、スキーマ・ノードおよびその下の**「テーブル」**、**「ビュー (Views)」**、**「ストアード・プロシージャー (Stored**

Procedures)」、「**シノニム - ニックネーム (Synonyms - Nicknames)**」の各ノードを展開するには、「**+**」 (正符号) をクリックします。これにより、ウィザードによってディスカバーされたデータベース・オブジェクトが表示されます。

11. 「**クエリー・ステートメント (Query Statements)**」および「**バッチ SQL ステートメント (Batch SQL Statements)**」のノードを展開するには、「**+**」 (正符号) をクリックします。これによって、クエリー・ビジネス・オブジェクトおよびバッチ SQL ビジネス・オブジェクトのテンプレートが表示されます。

タスクの結果

アダプター、およびクエリー・ビジネス・オブジェクトとバッチ SQL ビジネス・オブジェクトのビジネス・オブジェクト・テンプレートを使用してアクセスできる、データベース・オブジェクトがウィザードによって表示されました。

次のタスク

外部サービス・ウィザードでの作業を続行します。次の手順では、モジュールで使用するオブジェクトの選択、各ビジネス・オブジェクトの構成、およびビジネス・オブジェクトの階層の作成を行います。

Inbound 処理のデータベース・オブジェクトのディスカバー

接続プロパティーを構成した後は、データベース・オブジェクトを検索するクエリーを実行します。ディスカバーされたオブジェクトのツリーを参照して、データベース内のオブジェクトの構造を理解します。また、フィルターを使用して、参照したいデータベース・オブジェクトのみを表示します。

始める前に

データベースのアクセスに必要なプログラムのデータ要件を知っていなければなりません。例えば、データベースに関する以下の情報が必要になります。

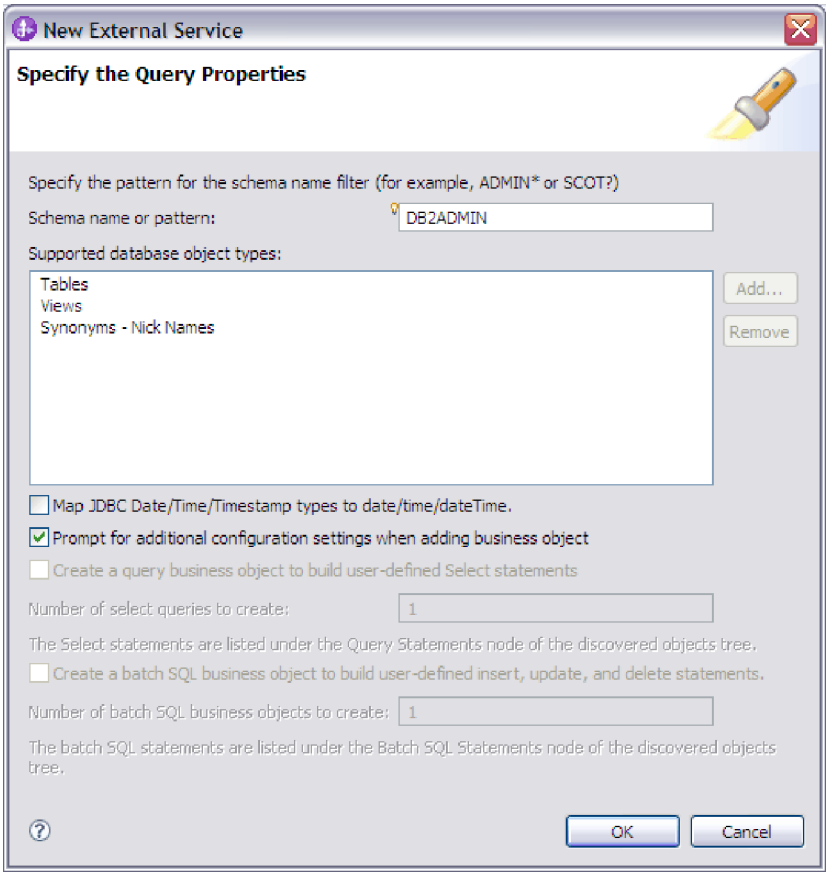
- モジュールはどのスキーマにアクセスする必要があるか
- これらのスキーマ内のどのタイプのデータベース・オブジェクトにアクセスする必要があるか

このタスクについて

この作業は、外部サービス・ウィザードの「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで開始します。

手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**照会の編集 (Edit Query)**」をクリックします。 照会プロパティーの指定ウィンドウが表示されます。



注: 「ユーザー定義の select ステートメントを作成するためのクエリー・ビジネス・オブジェクトを作成する」および**「ユーザー定義の insert、update、および delete ステートメントを作成するためのバッチ SQL ビジネス・オブジェクトを作成する」**の各オプションは、Outbound 処理にのみ使用できます。

「照会プロパティーの指定」ウィンドウを使用して、次のタスクを実行します。

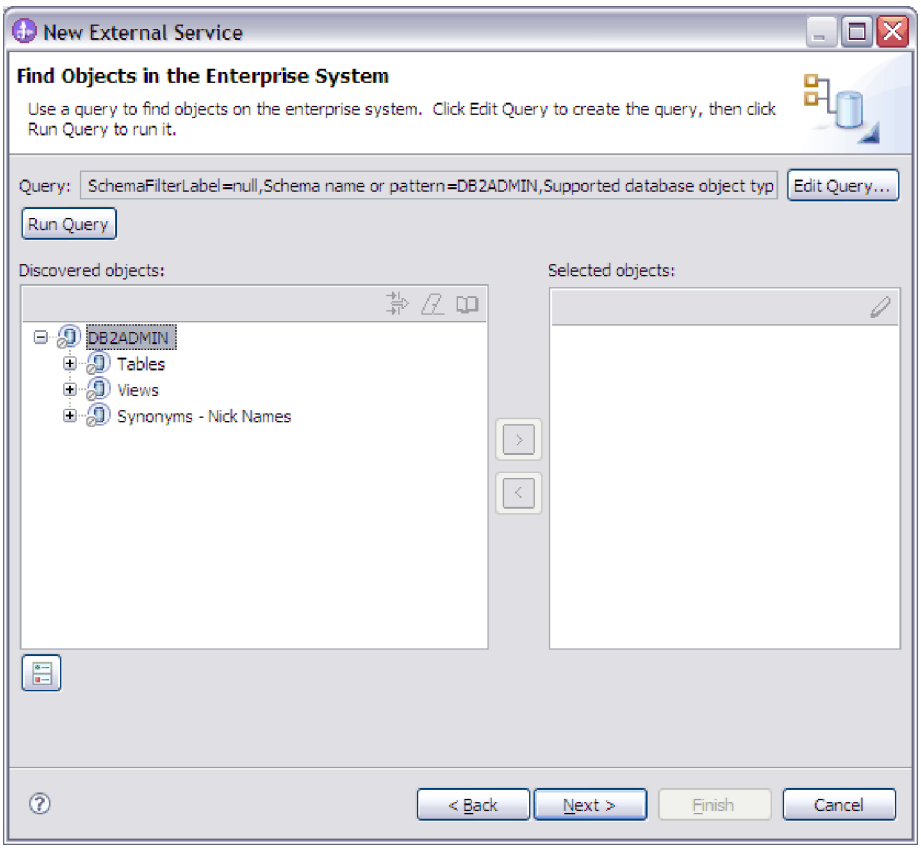
- データベース・スキーマのサブセットを検索することにより、検索時間を削減します。
- 1 つ以上のタイプのデータベース・オブジェクトを検索から除外します。
- データベース内の情報からは自動的に判別できないアプリケーション固有情報の入力をウィザードがユーザーに求めるようにします。
- JDBC のデータ型 Date、Time、および Timestamp を date、time、および dateTime にマップします。

2. 取得されるデータベース・スキーマの数を制限するには、**「スキーマ名またはパターン」**にスキーマの名前またはスキーマ名パターンを入力します。1 文字に対応させる場合は疑問符または下線 (? または _) を使用し、複数文字に対応させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。 クエリーを実行すると、そのストリングで始まるスキーマか、またはスキーマ名パターンに一致するスキーマのみが表示されます。スキーマ名パターンを指定しない場合は、データベース内のすべてのスキーマが表示されます。データベースに多数のスキーマがある場合は、フィルターを使用してディスカバリー・プロセスを高速化できます。

3. 検索から 1 つ以上のタイプのオブジェクトを除外するには、除外したいオブジェクトのタイプ (テーブル、ビュー、およびシノニムまたはニックネーム) を「**サポートされるデータベース・オブジェクト・タイプ**」フィールドで選択して、「**除去**」をクリックします。そのオブジェクト・タイプをもう一度追加するには、「**追加**」をクリックします。 アクセスする必要のないオブジェクト・タイプがデータベースに含まれている場合は、それらを除外すると、ディスカバリー・プロセスを高速化できます。
4. データ型が Date、Time、および Timestamp であるテーブル・オブジェクトは、デフォルトでは String データ型にマップされます。これらのオブジェクトを、JDBC ドライバーでサポートされる実際のデータ型 (Date、Time、Datetime など) にマップするには、「**JDBC の Date/Time/Timestamp の型を date/time/dateTime にマップ**」チェック・ボックスを選択します。

   注: デフォルトのデータ型マッピングは、JDBC ドライバーのバージョンによって異なります。例えば、Oracle JDBC ドライバーを使用する場合、Date データ型は、Date ではなく dateTime データ型にマップされます。このような場合は、「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウで、適切なデータ型を手動で選択する必要があります。
5. 「**ビジネス・オブジェクトの追加時に追加構成設定のプロンプトを出す**」チェック・ボックスを選択します。 これにより、作成するビジネス・オブジェクトのリストにデータベース・オブジェクトを追加すると、そのオブジェクトに関してユーザーが構成可能なすべてのアプリケーション固有情報を入力するよう求められます。例えば、このオプションを選択した場合は、ビジネス・オブジェクトの単純親子階層を作成するプロセスが、順を追ってウィザードで示されます。 2 つの異なるテーブルの属性を参照する 2 つの属性をテーブル・ビジネス・オブジェクトが持つ (つまり、2 つの親ビジネス・オブジェクトを持つ) 階層が必要な場合は、アセンブリー・エディターで構成を実行します。このエディターは、WebSphere Integration Developer から起動されるツールです。また、データベースで外部キー参照が定義されている場合は、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。

   重要: このオプションを選択しない場合、ウィザードは必須情報のみを入力するようプロンプトを出します。アセンブリー・エディターを使用して、ビジネス・オブジェクトの構成を実行する必要があります。また、データベースで外部キー参照を定義していない場合、アダプターは親子関係を自動的には生成しません。
6. 「**OK**」をクリックして、クエリーへの変更を保存します。
7. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで、「**クエリーの実行 (Run Query)**」をクリックします。これによりこのクエリーを使用してデータベース・オブジェクトがディスカバーされます。 標準的な照会の実行結果を次の図に示します。



「**検出済みオブジェクト**」ペインには、ディスカバーされたオブジェクトがリストされます。テーブル、ビュー、シノニム/ニックネームは、スキーマ名によってソートされます。

8. 「**検出済みオブジェクト**」リストで、スキーマ・ノードおよびその下の「**テーブル**」、「**ビュー (Views)**」、「**シノニム - ニックネーム (Synonyms - Nicknames)**」の各ノードを展開するには、「**+**」 (正符号) をクリックします。これにより、ウィザードによってディスカバーされたデータベース・オブジェクトが表示されます。

タスクの結果

アダプターを使用してアクセスできるデータベース・オブジェクトがウィザードによってディスカバーされました。

次のタスク

外部サービス・ウィザードでの作業を続行します。次の手順では、モジュールで使用するオブジェクトの選択、各ビジネス・オブジェクトの構成、およびビジネス・オブジェクトの階層の作成を行います。

Outbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成

モジュールで使用するテーブル、ビュー、およびシノニムまたはニックネームのビジネス・オブジェクトを選択して構成するには、ビジネス・オブジェクトの構成プロパティーを指定します。

始める前に

このタスクを実行するには、データベース内のデータの構造や、モジュールがどんなデータベース・オブジェクトにアクセスする必要があるかを理解しなければなりません。具体的には、以下の情報を認識しておく必要があります。

- テーブル、ビュー、およびシノニムまたはニックネームの構造 (必要な列や、データ型などの列属性を含む)。
- テーブル間の関係 (親子関係のカーディナリティーおよび所有権を含む)。

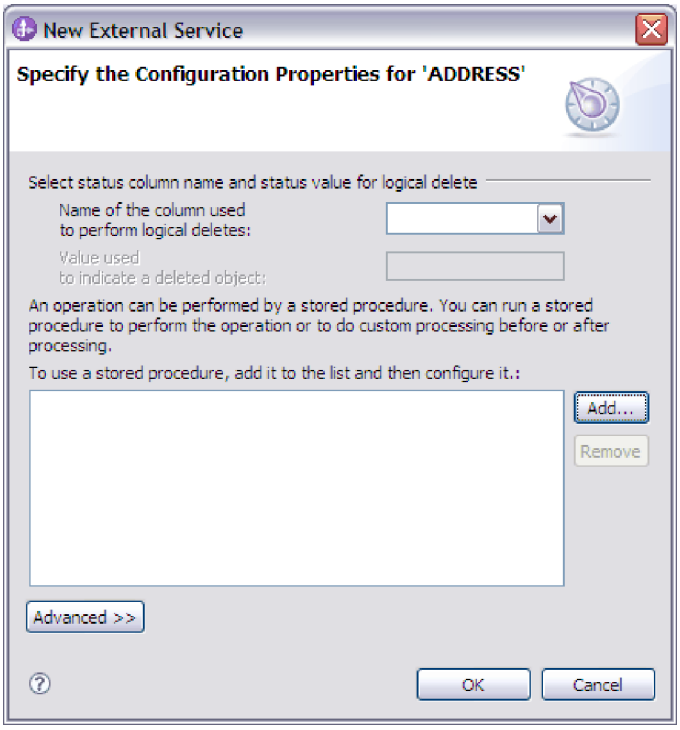
このタスクについて

このタスクは、外部サービス・ウィザードを介して実行されます。「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで作業を開始し、「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')ウィンドウ (構成するビジネス・オブジェクト固有のウィンドウ) で作業します。

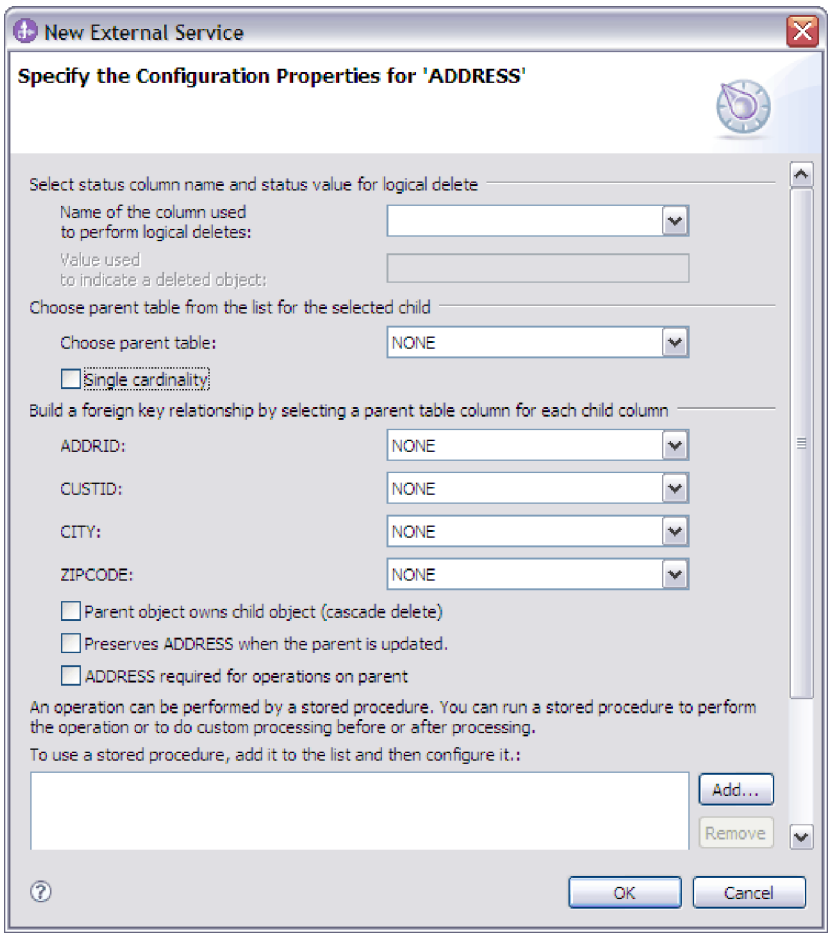
手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウの「**検出済みオブジェクト**」リストで、テーブル、ビュー、またはシノニムを 1 つ以上選択し、「**>**」 (追加) ボタンをクリックします。オブジェクトが「**選択済みオブジェクト**」リストに追加されます。

   以下の 2 つの図に、テーブル、ビュー、シノニム、またはニックネームのビジネス・オブジェクトの標準的な「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウを示します。最初の図に、選択する最初のテーブルまたはテーブル・グループの標準的なウィンドウを示します。



以下の図に、選択する後続のテーブルの標準的なウィンドウを示します。少なくとも 1 つのテーブルを選択して構成した後は、後続テーブルの「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに、テーブル間の親子階層をオプションで定義することができる領域が表示されます。



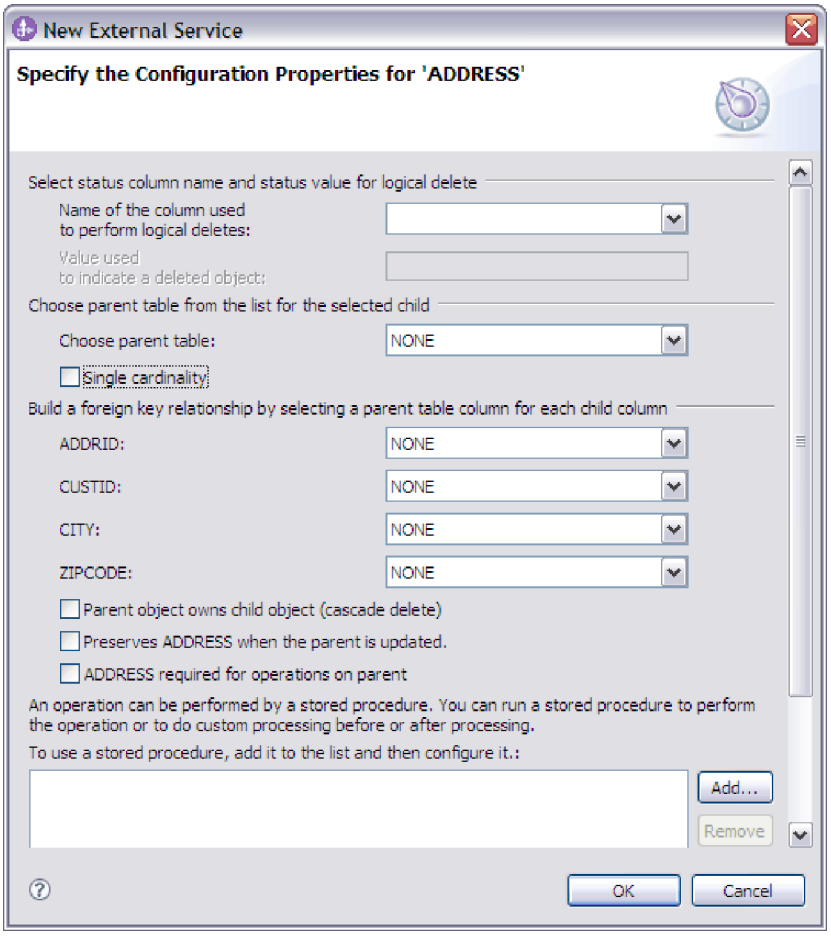
オブジェクトの構成時には、拡張構成を必要とする選択を行うと、このウィンドウに追加のフィールドが表示され、ウィンドウがスクロールされる場合があります。必ずウィンドウのすべてのフィールドを確認してから、「**OK**」をクリックしてください。

2. 論理削除を示すのに使用される列がテーブルにある場合は、次の手順に従います。
   a. 「**論理削除を実行するのに使用される列の名前**」フィールドで列名を選択します。
   b. 「**削除されたオブジェクトを示すために使用する値**」フィールドに、行が論理的に削除されていることを示す値を入力します。 この値については、データベース管理者に確認できます。
3. 「**テーブル *table_name* の基本キーの選択**」エリアが表示されたら、「**追加**」をクリックし、テーブル・ビジネス・オブジェクトの基本キーとして使用する列を選択してから、「**OK**」をクリックします。テーブルに複合キーがある場合は、複数の列を選択できます。 「**テーブル *table_name* の基本キーの選択**」エリアは、データベース表に基本キーとして指定された列が存在しない場合にのみ表示されます。各テーブル・ビジネス・オブジェクトには、関連付けられたデータベース表にキーがない場合でも、基本キーが定義されている必要があります。データベースで基本キーが定義されている場合、ウィンドウのこのセクションは表示されません。

4. オプション: ビジネス・オブジェクト間の親子関係を定義します。

   親子階層を作成する場合、まず親テーブルを構成して「エンタープライズ・システムでのオブジェクトの検索」ウィンドウに戻り、子テーブルを選択して構成します。

   以下の図に示す「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウの領域を使用して、親子関係を構成します。これらのフィールドは、構成する最初のテーブルの場合には表示されません。



   a. 「**親テーブルの選択**」フィールドで、構成する親テーブルの名前を選択します。 リストに親テーブルが表示されていない場合は、親テーブルがまだ構成されていません。戻って親オブジェクトを構成してから、子オブジェクトを構成してください。データベースで外部キー参照を定義した場合、親テーブルを選択すると、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。テーブルとその親のテーブル間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスが自動的に選択されます。

   b. 関係のカーディナリティーを指定します。

      - テーブルとその親テーブルの間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択します。単一カーディナリティー関係では、親はこのタイプの子ビジネス・オブジェク

トを 1 つのみ持つことができます。単一カーディナリティー関係は、所有関係を伴って実際の子を表すか、または所有関係を伴わずにルックアップ・テーブルまたはデータベース内の他の対等オブジェクトを表すために使用できます。

- テーブルに複数カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択しないでください。複数カーディナリティー関係では、親がこのタイプの子ビジネス・オブジェクトの配列を持つことができます。

c. 親と子の間に外部キー関係を作成するため、子の列ごとに、親テーブルの外部キーであるかどうかを指定します。
   - 子の列が外部キーでない場合は、「なし」を選択します。
   - 子の列が外部キーの場合は、その子の列に対応する親テーブルの列を選択します。

     注: ウィザードは、1 つの親テーブルのみを構成できます。子テーブルに複数の親テーブルがある場合は、ビジネス・オブジェクト・エディターを使用して、ウィザードを終了した後に残りの親テーブルを構成する必要があります。

d. 親オブジェクトが子オブジェクトを所有している場合、データベース内の子オブジェクトは親が削除されるときに削除されます。この子がその親によって所有されていることを示すには、「**親オブジェクトが子オブジェクトを所有する (カスケード削除)**」チェック・ボックスを選択します。あるいは、このオプションをクリアして、ルックアップ・テーブルなどの子オブジェクトが、親の削除時に削除されないようにします。

e. Update 操作の一環として子オブジェクトが削除されることのないようにするには、「**親の更新時に *child_table_name* を保持する**」チェック・ボックスを選択します。

   親テーブルが更新されると、アダプターは入力に存在する子ビジネス・オブジェクトを、データベースから返される子ビジネス・オブジェクトと比較します。デフォルトでは、アダプターは、入力ビジネス・オブジェクト内に存在しない、データベースから返されたすべての子オブジェクトを削除します。

f. デフォルトでは、子ビジネス・オブジェクトを指定せずに、親ビジネス・オブジェクトに対して操作を実行できます。親ビジネス・オブジェクトを変更対象として実行依頼するときに、その親ビジネス・オブジェクトで子ビジネス・オブジェクトが必ず指定されるようにしたい場合は、「***Child_table_name* は、親に対する操作で必須**」チェック・ボックスを選択します。

5. 操作を実行するには、アダプターによって生成される標準 SQL ステートメントを使用するか、あるいはデータベース内のストアード・プロシージャーまたはストアード関数を使用します。ストアード・プロシージャーまたはストアード関数を使用する場合は、以下の手順を実行します。

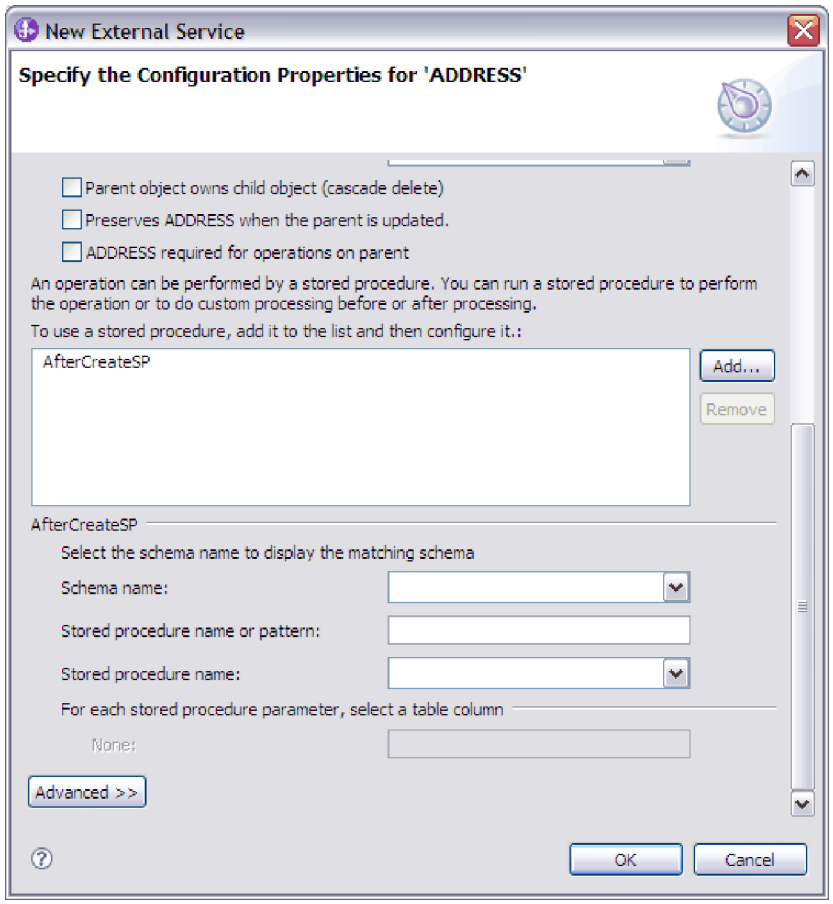
   a. 「**追加**」をクリックします。

   b. 「追加」ウィンドウで、実行するストアード・プロシージャーのタイプを選択します。 操作ごとに、その操作を実行するストアード・プロシージャーと、操作の前後に実行するストアード・プロシージャーを選択できます。例

えば、Create 操作の場合は、ストアード・プロシージャー CreateSP、BeforeCreateSP、および AfterCreateSP のどれでも指定できます。

注: RetrieveAllSP を指定してテーブルを構成する場合は、ストアード・プロシージャーが 1 つの結果セットのみを返すことを確認します。ストアード・プロシージャーの ResultSet ASI を `true` に設定して、実行時に例外「ストアード・プロシージャーに関連した結果セットが見つかりませんでした (`No resultset found associated with the stored procedure`)」、「結果セットが返されませんでした (`No resultset returned`)」、「複数の結果セットが返されました (`More than one resultset returned`)」のいずれも生成されないようにします。

c. 「**OK**」をクリックします。選択したストアード・プロシージャーのタイプが「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに表示されます。このウィンドウは、各ストアード・プロシージャーの構成用の領域を表示するために拡張されます。新しい領域を表示するには、スクロールダウンする必要がある場合もあります。

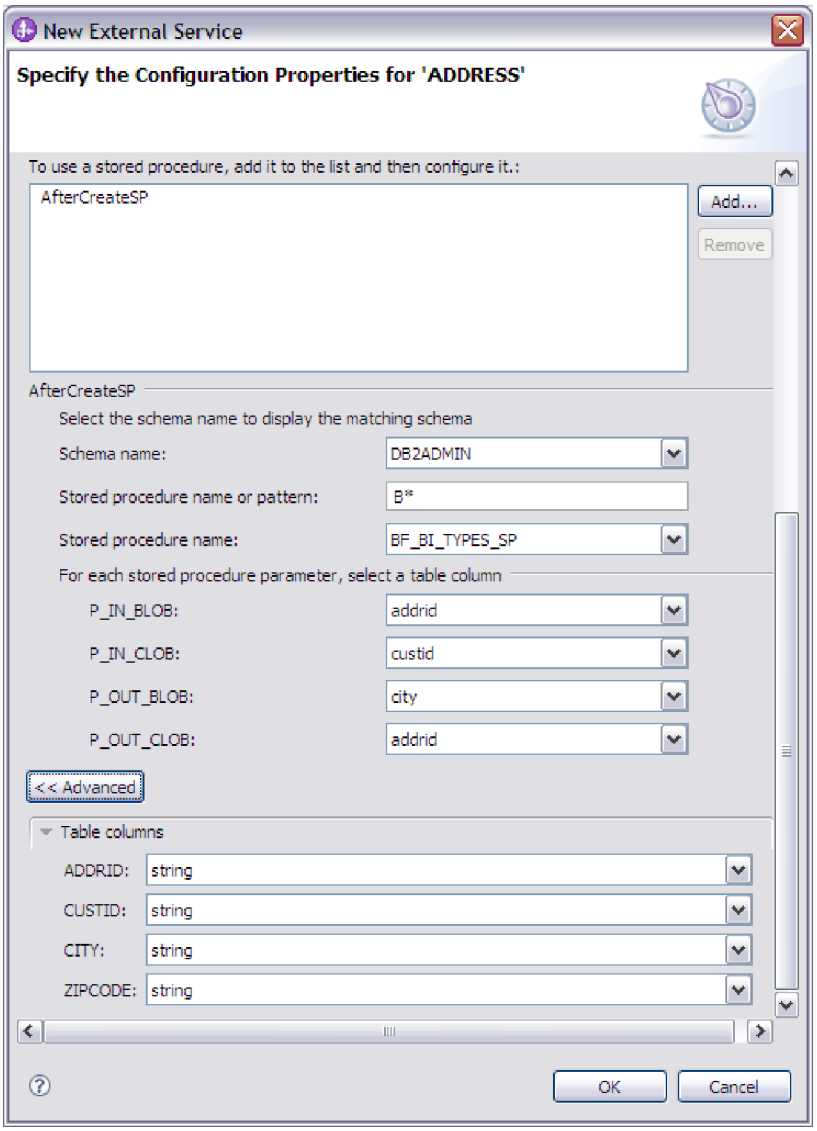


注: 階層ビジネス・オブジェクトにおいて、その階層のビジネス・オブジェクトごとにストアード・プロシージャーを実行する場合は、ストアード・プロシージャーを、トップレベルのビジネス・オブジェクトと、ビジネス・オブジェクトの各子ビジネス・オブジェクトまたは配列に別々に関連付ける必要があります。ストアード・プロシージャーをトップレベルのビジネス・オブ

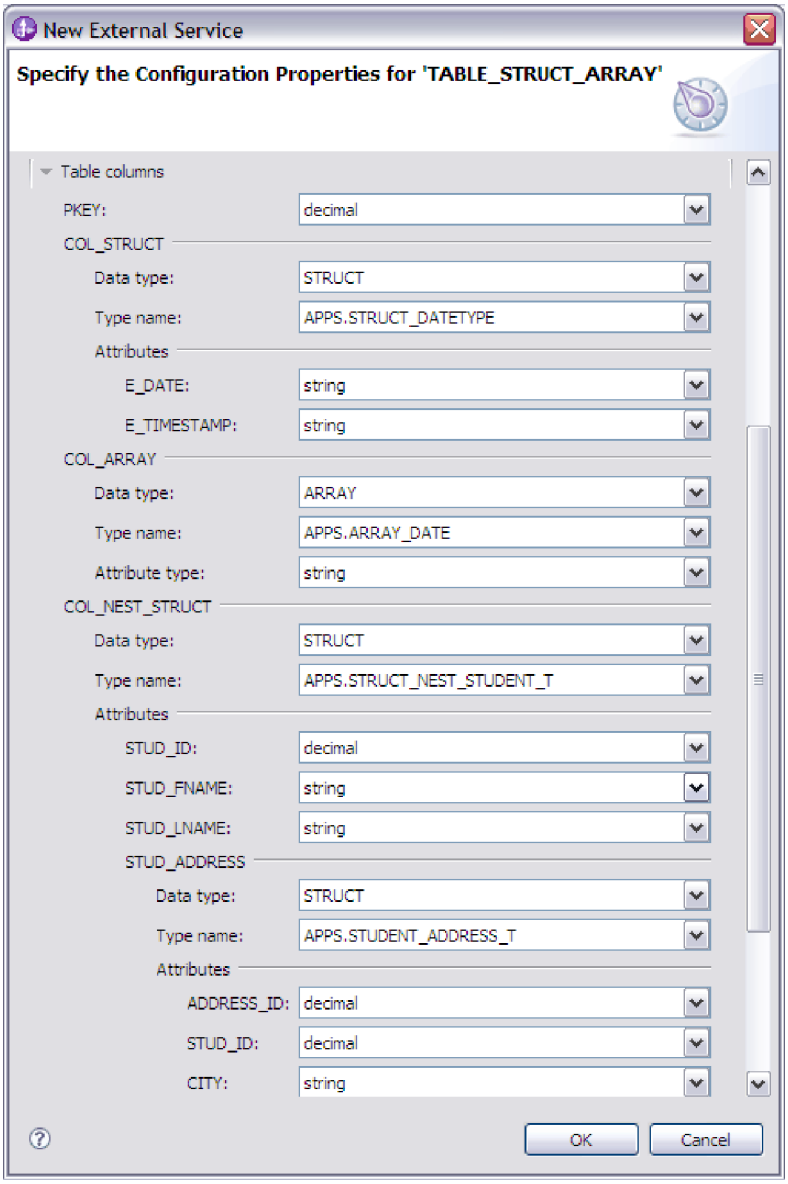
ジェクトに関連付けても、各子ビジネス・オブジェクトに関連付けないと、そのトップレベルのビジネス・オブジェクトはストアード・プロシージャーで処理されますが、子ビジネス・オブジェクトは標準 SQL 照会を使用して処理されます。

6. 選択したストアード・プロシージャー・タイプごとに、データベース内でのストアード・プロシージャーの名前を指定し、ビジネス・オブジェクトを構成します。
   a. 「**スキーマ名**」フィールドで、ストアード・プロシージャーが含まれるスキーマの名前を選択します。
   b. ストアード・プロシージャーまたはストアード関数の名前を指定します。
      1) 「**ストアード・プロシージャー名またはパターン**」フィールドで、ストアード・プロシージャーまたはストアード関数の名前を入力するか、または名前パターンを入力します。1 つの文字と一致させる場合は疑問符または下線 (? または _) を使用し、複数の文字と一致させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。
      2) 「**ストアード・プロシージャー名**」フィールドで、目的のプロシージャーの名前を選択します。

      「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウが拡張して、ストアード・プロシージャーを構成するための領域が表示されます。ウィザードは、データベース内のストアード・プロシージャーを調べることにより、パラメーターのリストを自動生成します。
   c. ストアード・プロシージャーのパラメーターごと (左側) に、そのパラメーターでストアード・プロシージャーに渡すテーブル列 (右側) を選択します。次の図に、ストアード・プロシージャーを構成した後のウィンドウの一部を示します。



7. テーブル内の各列のデータ型マッピングを指定するには、以下の手順を実行します。
   a. 「**拡張**」をクリックします。
   b. 「**テーブル列**」を展開します。テーブル内の列ごとに、デフォルトのデータ型マッピングが表示されます。 Oracle データベースにおいて、配列、構造体、ネストされた構造体、テーブルなど、何らかのユーザー定義型または複合データ型がテーブルに含まれている場合は、型名および子属性の詳細も自動的にディスカバーされて、表示されます。次の図に、複合データ型を含む Oracle テーブルの型名および子属性の詳細を示します。



c. マッピングを確認して、必要な場合は変更します。

8. ウィンドウのすべてのフィールドの操作が完了したら、「**OK**」をクリックします。ビジネス・オブジェクトの構成が保存されます。 定義したビジネス・オブジェクト (テーブル、ビュー、シノニム、およびニックネーム) が、「エンタープライズ・システムでのオブジェクトの検索」ウィンドウにリストされます。
9. 「**選択済みオブジェクト**」リストのオブジェクトの構成を変更するには、オブジェクト名を選択して、 (編集) アイコンをクリックします。

次のタスク

エンタープライズ・システムでのオブジェクトの検索ウィンドウで、他のタイプのビジネス・オブジェクトの選択と構成を続行します。完了したら、「**次へ**」をクリックして、グローバル・プロパティーを設定し、ラッパー・ビジネス・オブジェクトを構成します。

Inbound 処理のテーブル、ビュー、およびシノニムまたはニックネームの選択および構成

モジュールで使用するテーブル、ビュー、およびシノニムまたはニックネームのビジネス・オブジェクトを選択および構成します。Inbound 処理の場合、これらはイベントで送達されるビジネス・オブジェクトです。

始める前に

このタスクを実行するには、データベース内のデータの構造や、モジュールがどんなデータベース・オブジェクトにアクセスする必要があるかを理解しなければなりません。具体的には、以下の情報を認識しておく必要があります。

- テーブル、ビュー、およびシノニムまたはニックネームの構造 (必要な列や、データ型などの列属性を含む)。
- テーブル間の関係 (親子関係のカーディナリティーおよび所有権を含む)。

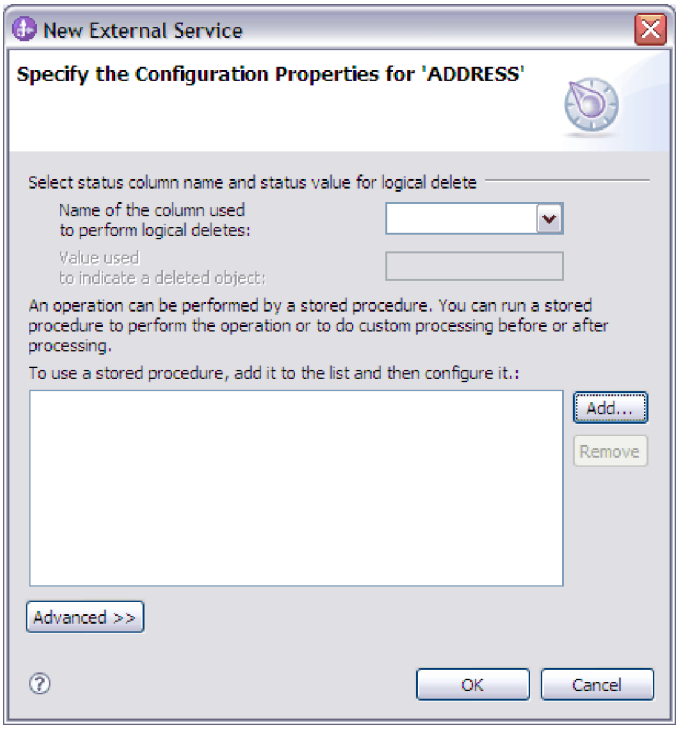
このタスクについて

このタスクは、外部サービス・ウィザードを介して実行されます。「エンタープライズ・システムでのオブジェクトの検索」ウィンドウで作業を開始し、「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')ウィンドウ (構成するビジネス・オブジェクト固有のウィンドウ) で作業します。

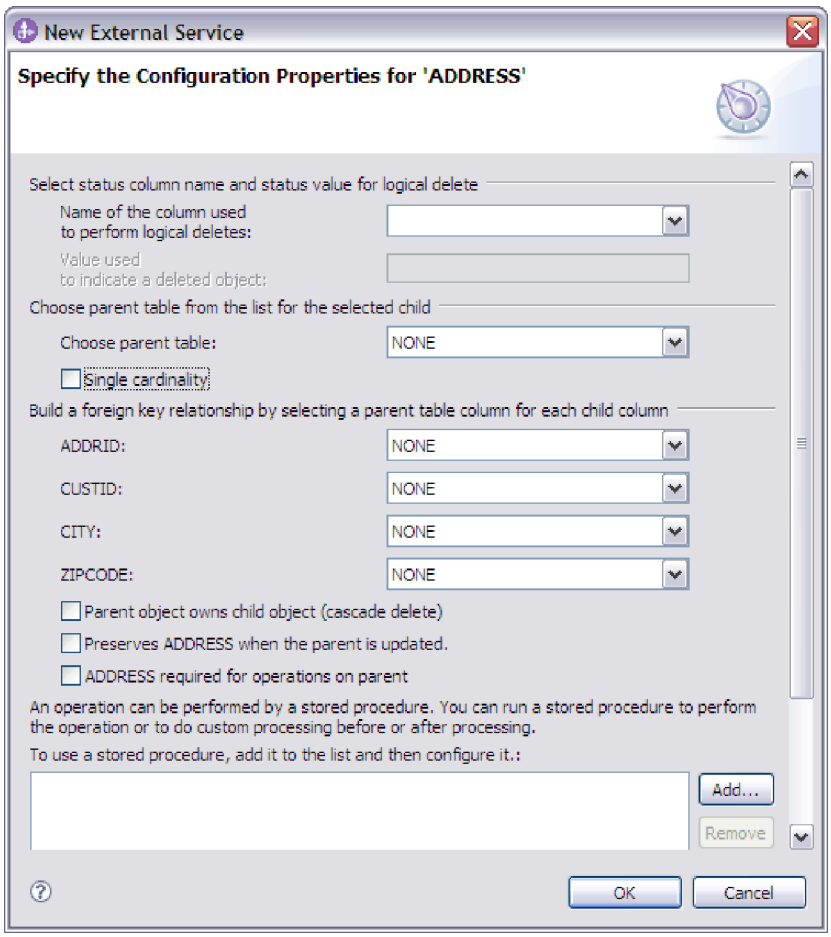
手順

1. 「エンタープライズ・システムでのオブジェクトの検索」ウィンドウの「**検出済みオブジェクト**」リストで、テーブル、ビュー、またはシノニムを 1 つ以上選択し、「**>**」 (追加) ボタンをクリックします。オブジェクトが「**選択済みオブジェクト**」リストに追加されます。

   以下の 2 つの図に、テーブル、ビュー、シノニム、またはニックネームのビジネス・オブジェクトの標準的な「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウを示します。最初の図に、選択する最初のテーブルまたはテーブル・グループの標準的なウィンドウを示します。



以下の図に、選択する後続のテーブルの標準的なウィンドウを示します。少なくとも 1 つのテーブルを選択して構成した後は、後続テーブルの「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに、テーブル間の親子階層をオプションで定義することができる領域が表示されます。



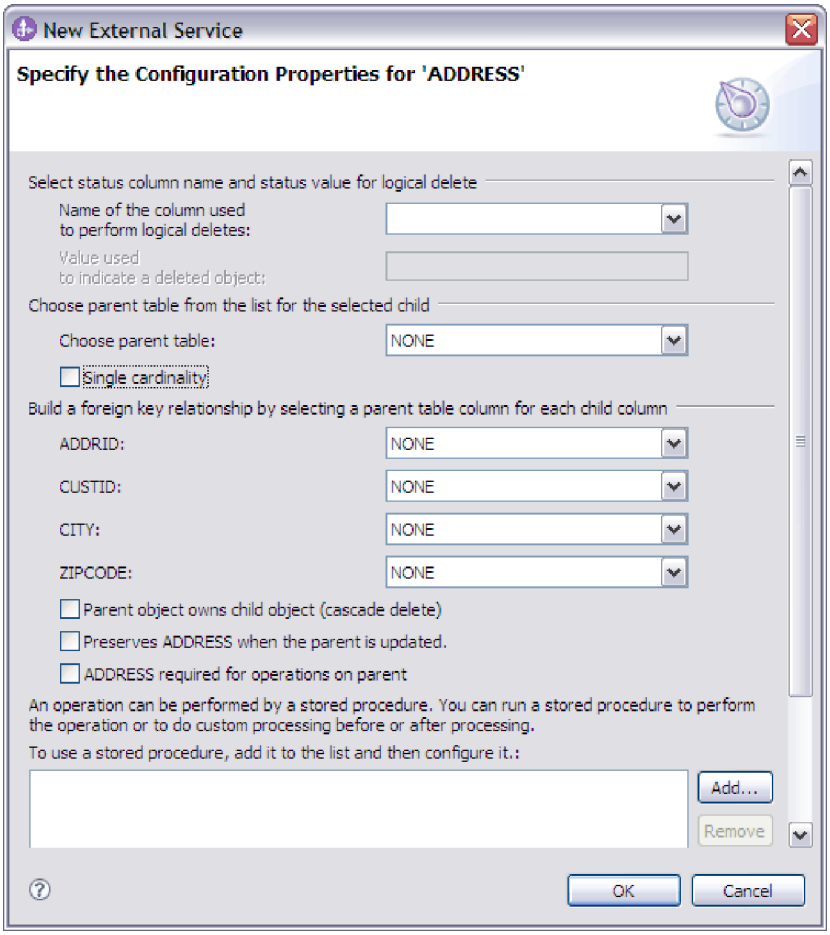
オブジェクトの構成時には、拡張構成を必要とする選択を行うと、このウィンドウに追加のフィールドが表示され、ウィンドウがスクロールされる場合があります。必ずウィンドウのすべてのフィールドを確認してから、「**OK**」をクリックしてください。

2. 論理削除を示すのに使用される列がテーブルにある場合は、次の手順に従います。
   a. 「**論理削除を実行するのに使用される列の名前**」フィールドで列名を選択します。
   b. 「**削除されたオブジェクトを示すために使用する値**」フィールドに、行が論理的に削除されていることを示す値を入力します。 この値については、データベース管理者に確認できます。
3. 「**テーブル *table_name* の基本キーの選択**」エリアが表示されたら、「**追加**」をクリックし、テーブル・ビジネス・オブジェクトの基本キーとして使用する列を選択してから、「**OK**」をクリックします。テーブルに複合キーがある場合は、複数の列を選択できます。 「**テーブル *table_name* の基本キーの選択**」エリアは、データベース表に基本キーとして指定された列が存在しない場合にのみ表示されます。各テーブル・ビジネス・オブジェクトには、関連付けられたデータベース表にキーがない場合でも、基本キーが定義されている必要があります。データベースで基本キーが定義されている場合、ウィンドウのこのセクションは表示されません。

4. オプション: ビジネス・オブジェクト間の親子関係を定義します。

   親子階層を作成する場合、まず親テーブルを構成して「エンタープライズ・システムでのオブジェクトの検索」ウィンドウに戻り、子テーブルを選択して構成します。

   以下の図に示す「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウの領域を使用して、親子関係を構成します。これらのフィールドは、構成する最初のテーブルの場合には表示されません。



   a. 「**親テーブルの選択**」フィールドで、構成する親テーブルの名前を選択します。 リストに親テーブルが表示されていない場合は、親テーブルがまだ構成されていません。戻って親オブジェクトを構成してから、子オブジェクトを構成してください。データベースで外部キー参照を定義した場合、親テーブルを選択すると、アダプターはテーブル間の親子関係を自動的にディスカバーして表示します。テーブルとその親のテーブル間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスが自動的に選択されます。

   b. 関係のカーディナリティーを指定します。

      - テーブルとその親テーブルの間に単一カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択します。単一カーディナリティー関係では、親はこのタイプの子ビジネス・オブジェク

トを 1 つのみ持つことができます。単一カーディナリティー関係は、所有関係を伴って実際の子を表すか、または所有関係を伴わずにルックアップ・テーブルまたはデータベース内の他の対等オブジェクトを表すために使用できます。

- テーブルに複数カーディナリティー関係がある場合は、「**単一カーディナリティー**」チェック・ボックスを選択しないでください。複数カーディナリティー関係では、親がこのタイプの子ビジネス・オブジェクトの配列を持つことができます。

c. 親と子の間に外部キー関係を作成するため、子の列ごとに、親テーブルの外部キーであるかどうかを指定します。

- 子の列が外部キーでない場合は、「なし」を選択します。
- 子の列が外部キーの場合は、その子の列に対応する親テーブルの列を選択します。

注: ウィザードは、1 つの親テーブルのみを構成できます。子テーブルに複数の親テーブルがある場合は、ビジネス・オブジェクト・エディターを使用して、ウィザードを終了した後に残りの親テーブルを構成する必要があります。

d. 親オブジェクトが子オブジェクトを所有している場合、データベース内の子オブジェクトは親が削除されるときに削除されます。この子がその親によって所有されていることを示すには、「**親オブジェクトが子オブジェクトを所有する (カスケード削除)**」チェック・ボックスを選択します。あるいは、このオプションをクリアして、ルックアップ・テーブルなどの子オブジェクトが、親の削除時に削除されないようにします。

e. Update 操作の一環として子オブジェクトが削除されることのないようにするには、「**親の更新時に *child_table_name* を保持する**」チェック・ボックスを選択します。

親テーブルが更新されると、アダプターは入力に存在する子ビジネス・オブジェクトを、データベースから返される子ビジネス・オブジェクトと比較します。デフォルトでは、アダプターは、入力ビジネス・オブジェクト内に存在しない、データベースから返されたすべての子オブジェクトを削除します。

f. デフォルトでは、子ビジネス・オブジェクトを指定せずに、親ビジネス・オブジェクトに対して操作を実行できます。親ビジネス・オブジェクトを変更対象として実行依頼するときに、その親ビジネス・オブジェクトで子ビジネス・オブジェクトが必ず指定されるようにしたい場合は、「***Child_table_name* は、親に対する操作で必須**」チェック・ボックスを選択します。

5. 操作を実行するには、アダプターによって生成される標準 SQL ステートメントを使用するか、あるいはデータベース内のストアード・プロシージャーまたはストアード関数を使用します。ストアード・プロシージャーまたはストアード関数を使用する場合は、以下の手順を実行します。

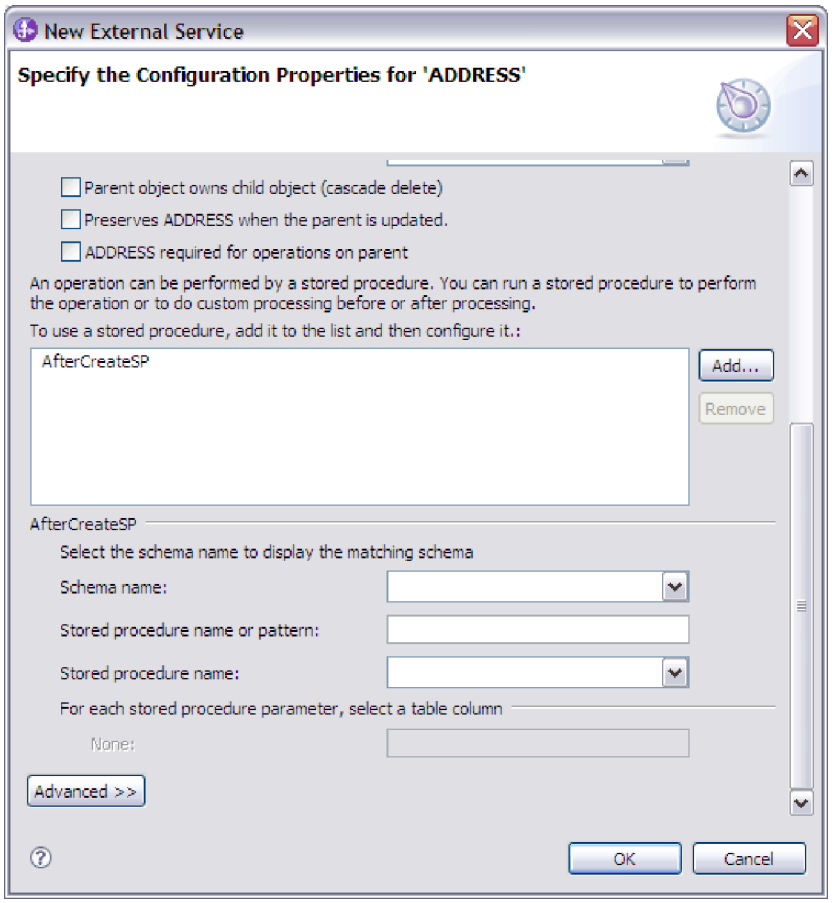
a. 「**追加**」をクリックします。

b. 「追加」ウィンドウで、実行するストアード・プロシージャーのタイプを選択します。 操作ごとに、その操作を実行するストアード・プロシージャー

と、操作の前後に実行するストアード・プロシージャーを選択できます。例えば、Create 操作の場合は、ストアード・プロシージャー CreateSP、BeforeCreateSP、および AfterCreateSP のどれでも指定できます。

注: RetrieveAllSP を指定してテーブルを構成する場合は、ストアード・プロシージャーが 1 つの結果セットのみを返すことを確認します。ストアード・プロシージャーの ResultSet ASI を true に設定して、実行時に例外「ストアード・プロシージャーに関連した結果セットが見つかりませんでした (No resultset found associated with the stored procedure)」、「結果セットが返されませんでした (No resultset returned)」、「複数の結果セットが返されました (More than one resultset returned)」のいずれも生成されないようにします。

c. 「**OK**」をクリックします。選択したストアード・プロシージャーのタイプが「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウに表示されます。このウィンドウは、各ストアード・プロシージャーの構成用の領域を表示するために拡張されます。新しい領域を表示するには、スクロールダウンする必要がある場合もあります。

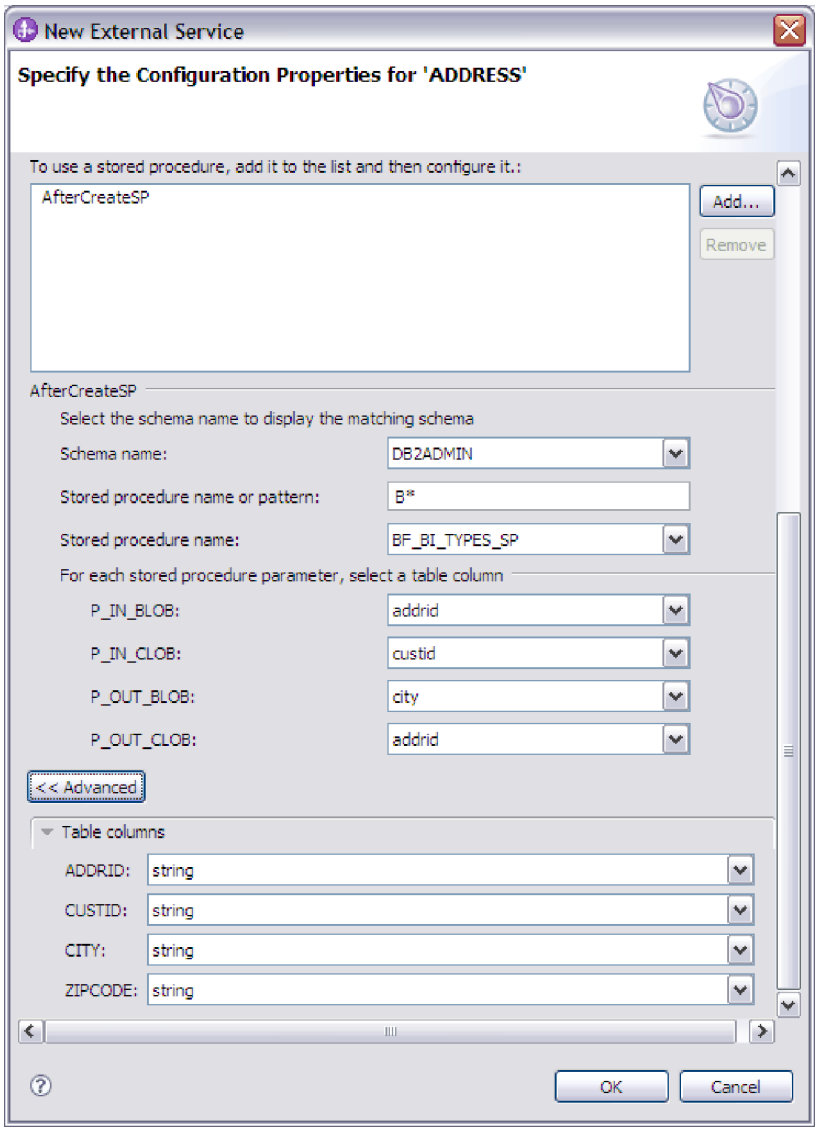


注: 階層ビジネス・オブジェクトにおいて、その階層のビジネス・オブジェクトごとにストアード・プロシージャーを実行する場合は、ストアード・プロシージャーを、トップレベルのビジネス・オブジェクトと、ビジネス・オブジェクトの各子ビジネス・オブジェクトまたは配列に別々に関連付ける必

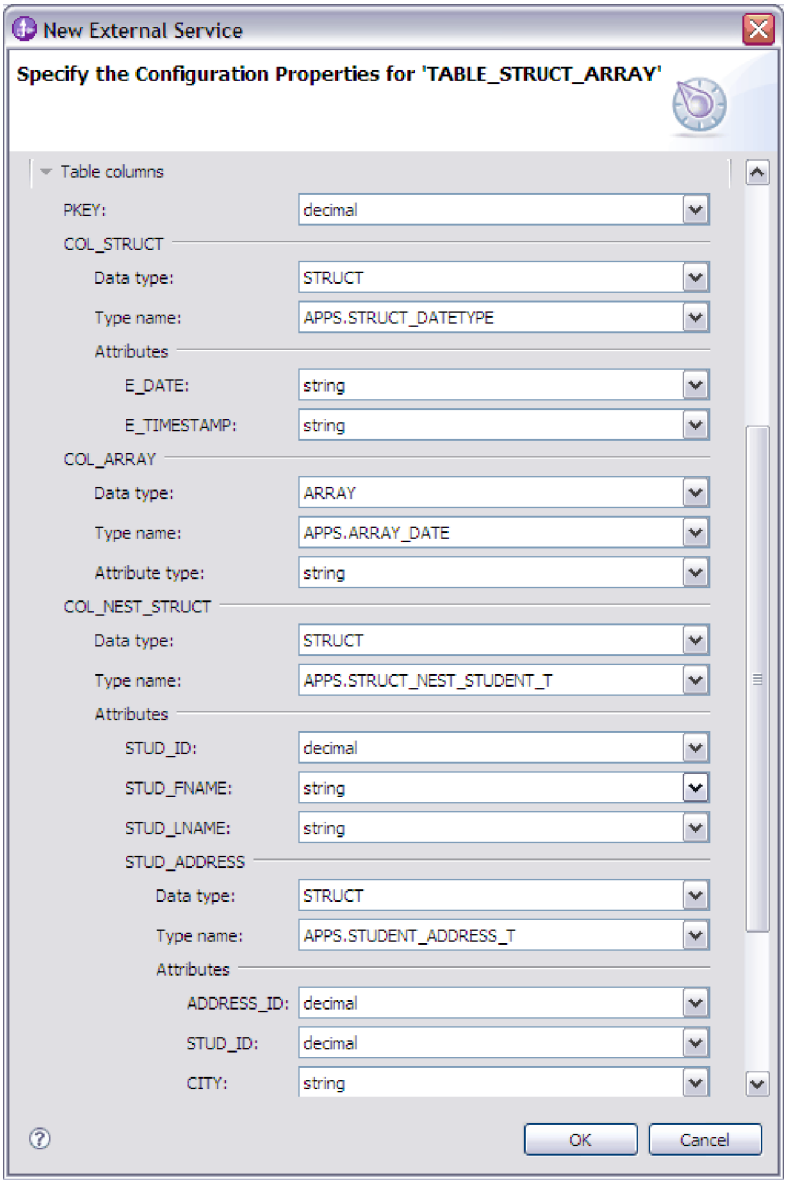
要があります。ストアード・プロシージャーをトップレベルのビジネス・オブジェクトに関連付けても、各子ビジネス・オブジェクトに関連付けないと、そのトップレベルのビジネス・オブジェクトはストアード・プロシージャーで処理されますが、子ビジネス・オブジェクトは標準 SQL 照会を使用して処理されます。

6. 選択したストアード・プロシージャー・タイプごとに、データベース内でのストアード・プロシージャーの名前を指定し、ビジネス・オブジェクトを構成します。
   a. 「**スキーマ名**」フィールドで、ストアード・プロシージャーが含まれるスキーマの名前を選択します。
   b. ストアード・プロシージャーまたはストアード関数の名前を指定します。
      1) 「**ストアード・プロシージャー名またはパターン**」フィールドで、ストアード・プロシージャーまたはストアード関数の名前を入力するか、または名前パターンを入力します。1 つの文字と一致させる場合は疑問符または下線 (? または _) を使用し、複数の文字と一致させる場合はアスタリスクまたはパーセント記号 (* または %) を使用します。
      2) 「**ストアード・プロシージャー名**」フィールドで、目的のプロシージャーの名前を選択します。

      「「オブジェクト」の構成プロパティーの指定 (Specify the Configuration Properties for 'object')」ウィンドウが拡張して、ストアード・プロシージャーを構成するための領域が表示されます。ウィザードは、データベース内のストアード・プロシージャーを調べることにより、パラメーターのリストを自動生成します。
   c. ストアード・プロシージャーのパラメーターごと (左側) に、そのパラメーターでストアード・プロシージャーに渡すテーブル列 (右側) を選択します。 次の図に、ストアード・プロシージャーを構成した後のウィンドウの一部を示します。



7. テーブル内の各列のデータ型マッピングを指定するには、以下の手順を実行します。
   a. 「**拡張**」をクリックします。
   b. 「**テーブル列**」を展開します。テーブル内の列ごとに、デフォルトのデータ型マッピングが表示されます。 Oracle データベースにおいて、配列、構造体、ネストされた構造体、テーブルなど、何らかのユーザー定義型または複合データ型がテーブルに含まれている場合は、型名および子属性の詳細も自動的にディスカバーされて、表示されます。次の図に、複合データ型を含む Oracle テーブルの型名および子属性の詳細を示します。



c. マッピングを確認して、必要な場合は変更します。

注: テーブル内の基本キーのデータ型が Date または Timestamp である場合、event_table 内の object_key のフォーマットは「yyyy-mm-dd hh-mm-ss」である必要があります。

8. ウィンドウのすべてのフィールドの操作が完了したら、「**OK**」をクリックします。ビジネス・オブジェクトの構成が保存されます。 定義したビジネス・オブジェクト (テーブル、ビュー、シノニム、およびニックネーム) が、「エンタープライズ・システムでのオブジェクトの検索」ウィンドウにリストされます。
9. 「**選択済みオブジェクト**」リストのオブジェクトの構成を変更するには、オブジェクト名を選択して、 (編集) アイコンをクリックします。
10. 必要なすべてのビジネス・オブジェクトを選択および構成したら、「**次へ**」をクリックして、グローバル・プロパティーを設定し、ラッパー・ビジネス・オブジェクトを構成します。

次のタスク

エンタープライズ・システムでのオブジェクトの検索ウィンドウで、他のタイプのビジネス・オブジェクトの選択と構成を続行します。完了したら、「**次へ**」をクリックして、グローバル・プロパティーを設定し、ラッパー・ビジネス・オブジェクトを構成します。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒242-8502
神奈川県大和市下鶴間 1623 番 14 号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
Department 2Z4A/SOM1
294 Route 100
Somers, NY 10589-0100
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向性および指針に関するすべての記述は、予告なく変更または撤回される場合があります。これらは目標および目的を提示するものにすぎません。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを

経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物にも、次のように、著作権表示を入れていただく必要があります。「(c) (お客様の会社名) (西暦年).このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 (c) Copyright IBM Corp. _年を入れる_. All rights reserved.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

プログラミング・インターフェース情報は、プログラムを使用してアプリケーション・ソフトウェアを作成する際に役立ちます。

一般使用プログラミング・インターフェースにより、お客様はこのプログラム・ツール・サービスを含むアプリケーション・ソフトウェアを書くことができます。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

警告:

診断、修正、調整情報は、変更される場合がありますので、プログラミング・インターフェースとしては使用しないでください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc.の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

この製品には、Eclipse Project (http://www.eclipse.org) により開発されたソフトウェアが含まれています。

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21