IBM WebSphere Business Integration
Adapters

Adapter for i2 Active Data Warehouse ユーザーズ・ガイド

Adapter バージョン 2.5.x

IBM WebSphere Business Integration
Adapters

Adapter for i2 Active Data Warehouse ユーザーズ・ガイド

Adapter バージョン 2.5.x

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、 167 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 Business Integration Adapter for i2 Active Data Warehouse (5724-H39) バージョン 2.5.x に適用されます。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

http://www.ibm.com/jp/manuals/main/mail.html

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

http://www.ibm.com/jp/manuals/ の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

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： IBM WebSphere Business Integration Adapters
Adapter for i2 Active Data Warehouse User Guide
Adapter Version 2.5.x

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2004.7

この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財）日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

本書について

IBM(R) WebSphere(R) Business Integration Adapter ポートフォリオは、主要な e-business テクノロジー、エンタープライズ・アプリケーション、レガシー、およびメインフレーム・システムに統合コネクティビティーを提供します。本製品には、コンポーネントをカスタマイズ、作成、および管理するためのツールとテンプレートが含まれており、これにより、ビジネス・プロセスの統合を実現します。

本書では、Adapter for i2 Active Data Warehouse のインストール、構成、ビジネス・オブジェクト開発、およびトラブルシューティングについて説明します。

対象読者

本書は、お客様のサイトでコネクターを使用するコンサルタント、開発者、およびシステム管理者を対象にしています。

関連文書

資料の完全セットにより、すべての WebSphere Business Integration Adapters に共通な機能とコンポーネントについての説明が提供されます。これには、特定のコンポーネントに関する参考資料も含まれます。

本書では、2 つの資料「システム・インストール・ガイド *(Windows 版)*」または「システム・インストール・ガイド *(UNIX 版)*」と「*WebSphere InterChange Server* システム・インプリメンテーション・ガイド」が頻繁に参照されています。本書を印刷する場合には、これらの資料の印刷も必要となります。

以下のサイトから、関連資料をインストールすることができます。

- 一般的なアダプター情報が必要な場合、アダプターを WebSphere Message Broker (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、WebSphere Business Integration Message Broker) とともに使用する場合、およびアダプターを WebSphere Application Server とともに使用する場合は、以下のサイトを参照してください。

  http://www.ibm.com/websphere/integration/wbiadapters/infocenter

- アダプターを InterChange Server とともに使用する場合は、以下のサイトを参照してください。

  http://www.ibm.com/websphere/integration/wicserver/infocenter

  http://www.ibm.com/websphere/integration/wbicollaborations/infocenter

- Message Broker (WebSphere MQ Integrator Broker、WebSphere MQ Integrator、および WebSphere Business Integration Message Broker) の詳細については、以下のサイトを参照してください。

  http://www.ibm.com/software/integration/mqfamily/library/manualsa/

- WebSphere Application Server の詳細については、以下を参照してください。

  http://www.ibm.com/software/webservers/appserv/library.html

  上記のサイトには資料のダウンロード、インストール、および表示に関する簡単な説明が記載されています。

注: 本書の発行後に公開されたテクニカル・サポートの技術情報や速報に、本書の対象製品に関する重要な情報が記載されている場合があります。これらの情報は、WebSphere Business Integration Support Web サイト (http://www.ibm.com/software/integration/websphere/support/) にあります。関心のあるコンポーネント・エリアを選択し、「Technotes」セクションと「Flashes」セクションを参照してください。

表記上の規則

本書では、以下のような規則を使用しています。

`Courier フォント`	コマンド名、ファイル名、入力情報、システムが画面に出力した情報など、記述されたとおりの値を示します。
太字	初めて出現した新規の用語を示します。
イタリック、***イタリック***	変数名または相互参照を示します。
青のアウトライン	オンラインで表示したときにのみ見られる青のアウトラインは、相互参照用のハイパーリンクです。アウトラインの内側をクリックすることにより、参照先オブジェクトにジャンプできます。
ProductDir	IBM WebSphere Business Integration Adapters 製品のインストール先ディレクトリーを表します。デフォルトのディレクトリーは WebSphereAdapters です。
{ }	構文行では、中括弧によって囲まれた複数のオプションから、1 つのオプションだけを選択する必要があります。
|	構文の記述行の場合、パイプで区切られた部分は、選択対象のオプションです。1 つのオプションだけを選択する必要があります。
[ ]	構文の記述行の場合、この大括弧 [] で囲まれた部分は、オプションのパラメーターです。
...	構文行では、省略符号は直前のパラメーターの繰り返しを示します。例えば、`option[,...]` は、複数のオプションをコンマで区切って入力できることを示します。
< >	1 つの名前の個々の要素を互いに区別するために、不等号括弧によって個々の要素が囲まれます。例えば、*<server_name><connector_name>*`tmp.log` のように使用します。

本リリースの新機能

リリース 2.5.x での新機能

2004 年 6 月更新。アダプターのバージョン 2.5.x に対応した本書のリリースでは、次の新規情報または訂正情報が追加されました。

- 新規のコネクター固有の構成プロパティーには、QueryTimeOut、ReturnDummyBOForSP、SelectiveCommitForPoll、および (UseDefaultsForRetrieve に代わって) UseDefaultsWhenPolling があります。既存のプロパティー DriverSupportForLong が文書化されました。
- コネクターは、IBM DB2 のシーケンスを使用する固有 ID の生成をサポートします。
- アダプターは、型 DATE の属性のアプリケーション固有情報に対する新規パラメーターをサポートします。
- コネクターは、DeltaUpdate 動詞の BeforeDeltaUpdateSP、AfterDeltaUpdateSP、DeltaUpdateSP のストアード・プロシージャー属性名を認識します。
- スキーマのツリーに、生成するビジネス・オブジェクト定義に関連付けるデータベース・オブジェクトを特定するための同義語/ニックネームと呼ばれる追加ノードが含まれています。
- スキーマのツリーに、オブジェクトに関連付けられたスキーマを持たないデータベースからオブジェクトを検索するためのスキーマ名 ALL SCHEMAS が含まれています。
- バージョン 2.5.0 以降の Adapter for i2 Active Data Warehouse は、Solaris 7 プラットフォームではサポートされなくなりました。

リリース 2.4.x の新機能

2004 年 2 月更新。アダプターのバージョン 2.4.x に対応した本書のリリースでは、次の新規情報または訂正情報が追加されました。

- アダプターは以下のプラットフォーム上で稼働します。
  - Windows 2000
  - HP-UX11i
  - AIX 5.x
  - Solaris 7 および 8
- 第 2 章の『コネクターの構成』に、トラステッド認証を使用する場合はコネクター固有の構成プロパティー ApplicationPassword と ApplicationUserName が不要であることが追記されました。
- 第 3 章の『ビジネス・オブジェクトの動詞の処理』で、DeltaUpdate 操作に関する説明が追加され、Delete 操作に関する説明が変更されました。
- 第 3 章の『単純属性のアプリケーション固有情報』に、名前値パラメーター [PH=truelfalse] の記述が追加されました。

- 第 3 章の『ビジネス・オブジェクトのアプリケーション固有の情報』に、CLOB データ型の定義に関する説明が追加されました。
- 第 3 章の上記と同じセクションの『ビジネス・オブジェクトの固有 ID の生成』に、IBM DB2 に関する情報が追加されました。
- 第 4 章の『ノードの展開と表およびビューの選択、ビューおよびストアード・プロシージャー』に、ストアード・プロシージャーに関する情報が追加されました。また、『追加情報の入力』に、ストアード・プロシージャーの属性に関する詳細情報が追加されました。
- 『付録 C』に、ビジネス・オブジェクトのサンプルが追加されました。

2003 年 12 月更新。

- バージョン 2.4.0 以降の Adapter for i2 Active Data Warehouse は Microsoft Windows NT ではサポートされなくなりました。
- アダプターのインストール情報は、本書から移動しました。この情報の新たな入手先については、第 2 章を参照してください。

リリース 2.3.x の新機能

2003 年 7 月更新。アダプターのバージョン 2.3.x に対応した本書のリリースでは、次の新規情報または訂正情報が追加されました。

- アダプターは、WebSphere Application Server を統合ブローカーとして使用できるようになりました。詳細については、9 ページの『互換性』を参照してください。
- アダプターは以下のプラットフォーム上で稼働します。
  - HP-UX11i
  - AIX 5.x
  - Solaris 7 および 8
- Oracle のストアード・プロシージャーからの結果セットの戻りをサポートするようになりました。
- CLOB データ型がサポートされるようになりました。
- コピー属性に対する親の親 (祖父母) からのアクセスがサポートされるようになりました。コピー属性に対して親からのアクセスが可能になりました。この結果、ビジネス・オブジェクト階層の下方向に属性を伝播できます。
- eventid は数値データ型でなければならないという制限が取り除かれました。

リリース 2.2.x の新機能

2003 年 3 月更新。「CrossWorlds」という名前は、現在ではシステム全体を表したり、コンポーネント名やツール名を修飾するためには使用されなくなりました。コンポーネント名およびツール名自体は、以前とほとんど変わりません。例えば、「CrossWorlds System Manager」は現在では「System Manager」となり、「CrossWorlds InterChange Server」は「WebSphere InterChange Server」となっています。

コネクターのバージョン 2.2.x では、次の新規情報または訂正情報が本書に追加されました。

- 以下に対するサポートが追加されました。
  - ビジネス・オブジェクトの最上位にあるラッパー・オブジェクト
  - LIKE 演算子
  - 16 進/バイナリー・データ
  - RetrieveUpdate 動詞のストアード・プロシージャー
  - RetrieveByContent 用の動詞に関するアプリケーション固有情報
  - RetrieveByContent の WHERE 文節の長さが 0 の場合の、WHERE 文節内の動詞に関するアプリケーション固有情報
- ConnectorID プロパティーが int から String に変更されたため、より記述的な名前を使用できるようになりました。
- カスタム JDBC ドライバーによって使用されるネイティブ・ライブラリーを指すため、DRIVERLIB 変数が追加されました。
- オブジェクト処理中のデータベース接続の喪失を検査するための機能が追加されました。
- イベントの検索およびアーカイブ時に、Schema Name プロパティーが使用されるようになりました。

リリース 2.1.x の新機能

IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) には、国際化されたコネクターが同梱されています。

リリース 2.0.x の新機能

コネクターは国際化されています。詳細については、7 ページの『ロケール依存データの処理』および 97 ページの『付録 A. コネクターの標準構成プロパティー』を参照してください。

リリース 1.9.x の新機能

IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) には、i2 Active Data Warehouse 用のコネクターが含まれます。このアダプターは、InterChange Server (ICS) と WebSphere MQ Integrator の 2 つの統合ブローカーをサポートします。統合ブローカーとは、異種のアプリケーション・セット間の統合を実行するアプリケーションです。統合ブローカーは、データ・ルーティングなどのサービスを提供します。

IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse には、以下のコンポーネントが含まれます。

- i2ADW に固有のアプリケーション・コンポーネント
- i2ADWODA
- サンプル・ビジネス・オブジェクト (格納場所: ¥connectors¥i2adw¥Samples¥directory)
- IBM WebSphere Adapter Framework。コンポーネントは以下のとおりです。
  - コネクター・フレームワーク

– 開発ツール (Business Object Designer と Connector Configurator を含む)
– API (ODK、JCDK、および CDK を含む)

本書では、ICS および WebSphere MQ Integrator の 2 つの統合ブローカーとアダプターを併用するための情報を提供します。

注: コネクターは国際化に対応していないため、ISO Latin-1 データのみが処理されることが確実である場合を除いて、コネクターと InterChange Server バージョン 4.1.1 を併用しないでください。

リリース 1.8.x の新機能

コネクターのビジネス・オブジェクト定義を生成する i2ADWODA の機能が拡張され、関連するドキュメンテーションが改訂されました。

リリース 1.7.x の新機能

製品のインストール用に、WebLogic JDBC ドライバーの代わりに MS SQL Server 用の IBM ブランド JDBC ドライバーが提供されます。Oracle thin ドライバーの提供は継続されます。

リリース 1.6.x の新機能

コネクター・バージョン 1.6.x 用の本書のリリースには、以下の新規情報または訂正情報が記載されています。

- コネクターのビジネス・オブジェクトを作成するために、Object Discovery Agent ユーティリティーが開発されました。以前のコネクター・リリースで提供された i2ADWBORGEN の代わりに、このユーティリティーを使用してください。
- `CheckForEventTableInInit` プロパティーの説明が 18 ページの『コネクター固有のプロパティー』に追加されました。
- `CloseDBConnection` プロパティーの説明が 18 ページの『コネクター固有のプロパティー』に追加されました。
- `SPBeforePollCall` プロパティーの説明が 18 ページの『コネクター固有のプロパティー』に追加されました。
- ResultSet を戻すストアード・プロシージャーのサポートが追加されました。 57 ページの『ビジネス・オブジェクトの Retrieve 操作』および 58 ページの『ビジネス・オブジェクトの RetrieveByContent 操作』を参照してください。
- 固定長ストリング属性のサポートが追加されました。 65 ページの表 11 の説明を参照してください。
- カスタム BO ハンドラーのサポートが追加されました。 17 ページの『カスタム・ビジネス・オブジェクト・ハンドラー・クラスの使用可能化』を参照してください。

リリース 1.5.x の新機能

コネクター・バージョン 1.5.0 用の本書のリリースには、以下の新規情報または訂正情報が記載されています。

- ArchiveTableName プロパティーの説明が更新されました。詳細については、 21 ページの『ArchiveTableName』を参照してください。
- AutoCommit プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- DateFormat プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- EventKeyDel プロパティーが名前値のペアに対応するように更新されました。詳細については、 24 ページの『EventKeyDel』を参照してください。
- EventQueryType プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- PingQuery プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- PreserveUIDSeq プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- SchemaName プロパティーが 18 ページの『コネクター固有のプロパティー』に追加されました。
- RetrieveByContent 動詞のストアード・プロシージャーがサポートされました。詳細については、 58 ページの『ビジネス・オブジェクトの RetrieveByContent 操作』を参照してください。
- イベント処理用の (非キー値に基づいた) ビジネス・オブジェクトの検索がサポートされました。詳細については、 5 ページの『イベント処理用ビジネス・オブジェクトの検索』を参照してください。

第 1 章 コネクターの概要

コネクターは、**コネクター・フレームワーク**、および**アプリケーション固有のコンポーネント**の 2 つの部分から構成されます。コネクター・フレームワークは、すべてのコネクターに共通のコードを備え、統合ブローカーとアプリケーション固有のコンポーネント間の仲介役として機能します。アプリケーション固有のコンポーネントには、特定のアプリケーションまたはテクノロジー (この場合は i2 Active Data Warehouse) に合わせて調整されたコードが含まれています。コネクター・フレームワークによって、統合ブローカーとアプリケーション固有のコンポーネント間に以下のサービスが提供されています。

- ビジネス・オブジェクトの送受信
- 始動メッセージおよび管理メッセージのやり取りの管理

この章では、IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) のコネクター・コンポーネントについて説明します。以下のセクションがあります。

- 『コネクター・コンポーネント』
- 2 ページの『コネクターの動作方法』

本書には、コネクター・フレームワーク、およびアプリケーション固有のコンポーネントの両方の情報が含まれているので注意してください。本書ではこれらを両方ともコネクターと呼んでいます。統合ブローカーとコネクターの関係の詳細については、「*IBM WebSphere InterChange Server システム管理ガイド*」または「*WebSphere MQ Integrator Broker 用インプリメンテーション・ガイド*」を参照してください。

コネクター・コンポーネント

コネクターにより、統合ブローカーは、JDBC 2.0 以降の仕様に準拠した Oracle データベース用の JDBC ドライバーを使用して、i2ADW アプリケーションでのビジネス・オブジェクトのやり取りを可能にしています。このセクションでは、コネクターのアーキテクチャー、およびさまざまな JDBC ドライバーの使用について、高いレベルで説明しています。

コネクターがデータベースへの接続に使用するドライバーの指定については、16 ページの『マルチ・ドライバー・サポートの使用可能化』を参照してください。

コネクターは、JDBC 接続機構を使用して、アプリケーション・データベースに接続します。コネクター固有の構成パラメーターの 1 つ (22 ページの『DatabaseURL』) を使用して、コネクターの接続先となるデータベース・サーバーの名前を指定できます。構成パラメーターについては、18 ページの『コネクターの構成』を参照してください。

コネクターは、始動時にデータベースでの接続プールを確立します。コネクターは、このプールからの接続を使用して、データベースでのあらゆるトランザクションを処理します。コネクターの停止時には、プール内の接続はすべて終了します。

コネクター・アーキテクチャー

図 1 に、IBM WebSphere Business Integration Adapter システム内のコネクター・コンポーネントおよびそれらの関係を示します。

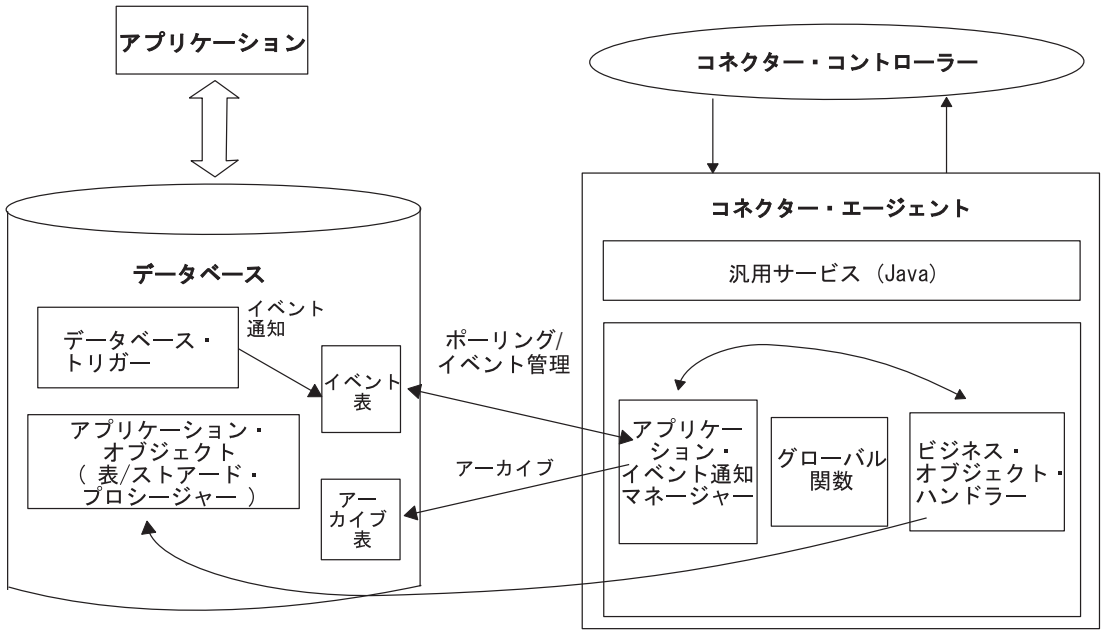


図 1. InterChange Server 環境におけるビジネス・オブジェクト要求アーキテクチャー

コネクターの動作方法

このセクションでは、メタデータによってどのようにコネクターの柔軟性が高まるのかを説明し、ビジネス・オブジェクト処理とイベント通知について概要を説明します。

コネクターおよびメタデータ

コネクターはメタデータ主導型です。**メタデータ**は、WebSphere Business Integration Adapter ビジネス・オブジェクト内に格納されているアプリケーション固有のデータであり、コネクターとアプリケーションの相互作用を支援します。メタデータ主導型コネクターは、サポートする各ビジネス・オブジェクトを処理する際に、コネクター内にハードコーディングされた命令ではなく、ビジネス・オブジェクト定義にエンコードされたメタデータに基づいて処理を行います。

ビジネス・オブジェクトのメタデータには、ビジネス・オブジェクトの構造、属性プロパティーの設定、およびアプリケーション固有情報の内容が含まれています。コネクターはメタデータ主導型であるため、コネクター・コードを変更する必要なしに、新規または変更されたビジネス・オブジェクトを処理することができます。

コネクターは、SQL ステートメントまたはストアード・プロシージャーを実行して、データベース/アプリケーション内のデータを検索または変更します。動的 SQL ステートメントまたはストアード・プロシージャーをビルドするために、コネクタ

ーはアプリケーション固有のメタデータを使用します。これらの SQL ステートメント、およびストアード・プロシージャーは、コネクターが処理中のビジネス・オブジェクトおよび動詞に必要とされる検索または変更を、データベース/アプリケーションに対して実行します。アプリケーション固有の情報の使用については、 37 ページの『第 3 章 コネクターのビジネス・オブジェクトについて』を参照してください。

ビジネス・オブジェクトの処理

このセクションでは、ビジネス・オブジェクトの要求およびアプリケーション・イベントがコネクターによって処理される仕組みについて概要を示します。詳細については、 42 ページの『ビジネス・オブジェクト動詞の処理』を参照してください。

ビジネス・オブジェクト要求の処理

コネクターは、統合ブローカーからアプリケーション操作実行の要求を受け取ると、階層ビジネス・オブジェクトを再帰的に処理します。つまり、コネクターは、個々のビジネス・オブジェクトをすべて処理し終えるまで、子ビジネス・オブジェクトごとに同じステップを実行します。子ビジネス・オブジェクトおよびトップレベル・ビジネス・オブジェクトがコネクターによって処理される順序は、その子ビジネス・オブジェクトの包含状態 (所有権の有無、または単一/複数カーディナリティー) に応じて異なります。

注: **階層**ビジネス・オブジェクトという用語は、完全なビジネス・オブジェクトを示します。これにはあらゆるレベルでこのビジネス・オブジェクトが含む子ビジネス・オブジェクトすべてが含まれます。**個々の**ビジネス・オブジェクトという用語は、互いに包含関係にあると考えられるビジネス・オブジェクトのいずれの子ビジネス・オブジェクトからも独立した、単一のビジネス・オブジェクトを指します。**トップレベル**・ビジネス・オブジェクトという用語は、個々のビジネス・オブジェクトで、階層の最上位にあり、それ自身が親ビジネス・オブジェクトを持たないものを示します。

ビジネス・オブジェクトの検索: 統合ブローカーがデータベースから階層ビジネス・オブジェクトを検索するようコネクターに要求すると、コネクターは、そのビジネス・オブジェクトの現行データベース表現と完全に一致するビジネス・オブジェクトを戻そうとします。つまり、統合ブローカーに戻された個々のビジネス・オブジェクトの単純な属性はすべて、データベース内にある対応するフィールドの値と一致します。また、戻されたビジネス・オブジェクトに包含される各配列内の個々のビジネス・オブジェクトの数は、その配列用のデータベース内の子の数と一致します。

コネクターはそのような検索を実行する場合、統合ブローカーから受け取ったトップレベル・ビジネス・オブジェクト内の基本キーの値を使用して、データベース内の対応するデータ全体を再帰的に降りて行きます。

内容によるビジネス・オブジェクトの検索: 統合ブローカーがトップレベル・ビジネス・オブジェクト内にある非キー属性の値をベースに階層ビジネス・オブジェクトを検索するようコネクターに要求すると、コネクターはすべての非ヌル属性の値をデータ検索基準として使用します。

ビジネス・オブジェクトの作成: 統合ブローカーが、データベース内に階層ビジネス・オブジェクトを作成するようコネクターに要求すると、コネクターによって以下のステップが実行されます。

1. 所有権ありで包含されている単一カーディナリティーの子ビジネス・オブジェクトをそれぞれ、データベースの中に再帰的に作成する。
2. 所有権なしで包含されている単一カーディナリティーの子ビジネス・オブジェクトをそれぞれ処理する。
3. データベース内にトップレベル・ビジネス・オブジェクトを作成する。
4. 親/子関係を子の中に格納する、単一カーディナリティーの子ビジネス・オブジェクトをそれぞれ作成する。
5. 複数カーディナリティーの子ビジネス・オブジェクトをそれぞれ作成する。

ビジネス・オブジェクトの変更: 統合ブローカーが、データベース内にある階層ビジネス・オブジェクトの更新をコネクターに要求すると、コネクターによって以下のステップが実行されます。

1. ソース・ビジネス・オブジェクトの基本キーの値を使用して、データベースから対応するエンティティーを検索する。
2. トップレベル・ビジネス・オブジェクトの単一カーディナリティーの子をすべて再帰的に更新する。
3. 親の中に関係を格納する単一カーディナリティーの子ビジネス・オブジェクトに対しては、親の中の外部キーの値を、対応する単一カーディナリティーの子ビジネス・オブジェクトの中の基本キーの値に設定する。
4. 検索されたビジネス・オブジェクトの単純属性すべてを更新する。ただし、ソース・ビジネス・オブジェクトの対応する属性の値が `CxIgnore` である場合を除く。
5. 親/子関係が子の中に格納されているそれぞれの子 (複数カーディナリティーと単一カーディナリティーの両方) の中の外部キーをすべて、対応する親ビジネス・オブジェクトの基本キーの値に設定する。
6. 検索されたビジネス・オブジェクトの配列をすべて処理する。

ビジネス・オブジェクトの削除: 統合ブローカーが、データベースから階層ビジネス・オブジェクトを削除するようコネクターに要求すると、コネクターによって以下のステップが実行されます。

1. 単一カーディナリティーの子を削除する。
2. 複数カーディナリティーの子を削除する。
3. トップレベル・ビジネス・オブジェクトを削除する。

アプリケーション・イベントの処理

コネクターは、アプリケーションで生成された Create、Update、および Delete イベントを下記の方法で処理します。

Create 通知: コネクターは、イベント表内に Create イベントを検出すると、イベントによって指定されたタイプのビジネス・オブジェクトを作成し、そのビジネス・オブジェクトのキー値 (イベント表内に指定されたキーを使用) を設定して、そ

のビジネス・オブジェクトをデータベースから検索します。コネクターは、ビジネス・オブジェクトを検索した後、動詞 Create を指定してそのビジネス・オブジェクトを統合ブローカーに送信します。

Update 通知: コネクターは、イベント表内に Update イベントを検出すると、イベントによって指定されたタイプのビジネス・オブジェクトを作成し、そのビジネス・オブジェクトのキー値 (イベント表内に指定されたキーを使用) を設定して、そのビジネス・オブジェクトをデータベースから検索します。コネクターは、ビジネス・オブジェクトを検索した後、動詞 Update を指定してそのビジネス・オブジェクトを統合ブローカーに送信します。

Delete 通知: コネクターは、イベント表内に Delete イベントを検出すると、イベントによって指定されたタイプのビジネス・オブジェクトを作成し、そのビジネス・オブジェクトのキー値 (イベント表内に指定されたキーを使用) を設定して、動詞 Delete を指定してそのビジネス・オブジェクトを統合ブローカーに送信します。キー値以外の値はすべて、CxIgnore に設定されます。ユーザーのサイトにある非キー・フィールドのいずれかが有効である場合は、必要に応じてそのフィールドの値を変更してください。

コネクターは、そのアプリケーションによって起動される、論理的および物理的 Delete 操作を処理します。物理的な削除の場合には、SmartFiltering 機構によってビジネス・オブジェクトの未処理イベント (例えば Create や Update) がすべて除去された後で、イベント表に Delete イベントが挿入されます。論理的な削除の場合、コネクターはビジネス・オブジェクトの他のイベントを除去せずに、イベント表内に Delete イベントを挿入します。

オブジェクト・イベント ID の設定: オブジェクト・イベント ID は、アプリケーション・ブローカーからのロギング重複イベントを避けるために使用される固有 ID です。例えば、イベントが進行中で、さらに統合ブローカーに送信されると、アダプターに障害が発生します。アダプターを再始動すると、イベントが再処理および再送されます。次に、統合ブローカーはイベント ID を比較し、重複イベントがある場合はそれを破棄します。これは、各イベント ID が固有であるためです。

イベント処理用ビジネス・オブジェクトの検索: イベント処理のためのビジネス・オブジェクト検索は、2 とおりの方法で実行することができます。第 1 の方法は、ビジネス・オブジェクトのキー属性に基づく検索です。第 2 の方法は、キー属性および非キー属性の両方に基づく検索です。この場合、ビジネス・オブジェクトで `RetrieveByContent` 動詞がサポートされ、オブジェクト・キーに `name_value` ペアが使用されていなければなりません。

注: オブジェクト・キーに `name_value` ペアが使用されていない場合、オブジェクト・キー・フィールド内のキーの順序は、ビジネス・オブジェクト内のキーと同じ順序でなければなりません。

イベント通知

コネクターのイベント検出機構には、イベント表、アーカイブ表、ストアード・プロシージャー、およびデータベース・トリガーが使用されています。イベント管理プロセスは、アーカイブ表へのイベントの挿入を完了するまで、イベント表からイベントを削除しません。これは、イベント処理に関連して、障害が発生する可能性がある点がいくつかあるためです。

データベース・トリガーは、データベース内に関心のあるイベントが発生するたびに、イベント表に取り込みます。コネクターは、一定間隔 (変更可能) でこの表に対してポーリングを実行し、イベントを検索して処理します。処理は、まず優先順位に従って実行され、次に順次実行されます。コネクターがこのイベント処理を完了すると、イベントの状況が更新されます。

注: インストール手順の一部として、データベースにトリガーを追加する必要があります。

コネクターがイベントの状況の更新後にそのイベントをアーカイブ表にアーカイブするかどうかは、そのイベントの ArchiveProcessed プロパティーの設定値に応じて判別されます。ArchiveProcessed プロパティーに関する詳細については、 18 ページの『コネクターの構成』を参照してください。

表 1 に、ArchiveProcessed プロパティーの設定に応じたアーカイブの振る舞いを示します。

表 1. アーカイブ時の振る舞い

アーカイブ処理設定	イベント表から削除される理由	コネクターの振る舞い
`true` または値なし	処理成功	アーカイブして、状況「Sent to InterChange」を出力する
	処理失敗	アーカイブして、状況「Error」を出力する
	ビジネス・オブジェクトのサブスクリプションなし	アーカイブして、状況「Unsubscribed」を出力する (サブスクリプションの具体的情報については、統合ブローカーのインプリメンテーション・ガイドを参照)
`false`	処理が成功した	アーカイブせずに、イベント表から削除する
	処理が失敗した	状況を `Error` にしてイベント表に残す
	ビジネス・オブジェクトに対するサブスクリプションがない	イベント表に保持して、状況「Unsubscribed」を出力する (サブスクリプションの具体的情報については、統合ブローカーのインプリメンテーション・ガイドを参照)

SmartFiltering は、データベース・トリガー内部において、統合ブローカーおよびコネクターによって実行される処理の量を最小限に抑える機構です。例えば、コネクターによる前回のイベント・ポーリングの後で、Contract ビジネス・オブジェクトがあるアプリケーションによって 15 回更新されている場合、SmartFiltering は、これらの変更を単一の Update イベントとして保管します。

データベース接続不能の処理

データベース接続が失われる理由は数多くあります。データベース接続が失われると、コネクターは終了します。JDBC の仕様では、接続の喪失を検出する機構は定められていません。コネクターには、この検出処理のため、 26 ページの『PingQuery』 プロパティーが用意されています。サービスの呼び出し要求中に障害が発生した場合、コネクターはこの `PingQuery` を実行して、データベース接続の

切断が障害の原因ではなかったことを確認します。PingQuery が失敗した場合に AutoCommit プロパティーが false に設定されていると、コネクターはデータベースへの新規の接続を作成しようとします。データベースへの新規接続の作成に成功した場合、コネクターは処理を続行します。失敗した場合、コネクターは APPRESPONSETIMEOUT を戻します。この結果、コネクターは終了します。

データベースへのアクセス中に障害が発生した場合は、トランザクションのタイプには関係なく 26 ページの『PingQuery』が実行されます。以下に例を示します。

- イベント表およびアーカイブ表にアクセスしているとき
- イベントに関連するビジネス・オブジェクトを検索しているとき
- ビジネス・オブジェクトに関連するレコードを作成または更新しているとき

ロケール依存データの処理

コネクターは、2 バイト文字セットをサポートして、指定された言語でメッセージ・テキストを送れるように国際化されています。ある文字コード・セットを使用する場所から別の文字コード・セットを使用する場所へデータを転送する場合、コネクターは、そのデータの意味を保持するように文字変換を実行します。

Java 仮想マシン (JVM) 内での Java ランタイム環境は、Unicode 文字コード・セットでデータを表します。Unicode には、最もよく知られた文字セット (単一バイトとマルチバイトの両方) の文字エンコードが含まれています。IBM WebSphere Business Integration システムのほとんどのコンポーネントは Java で書かれています。そのため、WebSphere Business Integration システム・コンポーネント間でデータを転送するときは、ほとんどの場合文字変換は必要ありません。

エラー・メッセージや通知メッセージを個々の国や地域に合った適切な言語で記録するには、個々の環境に合わせて Locale 標準構成プロパティーを構成する必要があります。これらのプロパティーの詳細については、97 ページの『付録 A. コネクターの標準構成プロパティー』を参照してください。

第 2 章 コネクターのインストールと構成

この章では、IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) のインストールおよび構成の方法について説明します。本章の内容は、次のとおりです。

互換性

アダプターが使用するアダプター・フレームワークは、アダプターと通信する統合ブローカーのバージョンとの互換性を備えている必要があります。Adapter for i2 Active Data Warehouse バージョン 2.5.x は、以下のバージョンのアダプター・フレームワークおよび統合ブローカーでサポートされています。

- **アダプター・フレームワーク:**
  WebSphere Business Integration Adapter Framework バージョン 2.1、2.2、2.3.x、および 2.4
- **統合ブローカー:**
  - WebSphere InterChange Server、バージョン 4.1.1、4.2、4.2.1、4.2.2
  - WebSphere MQ Integrator、バージョン 2.1.0
  - WebSphere MQ Integrator Broker、バージョン 2.1.0
  - WebSphere Business Integration Message Broker、バージョン 5.0
  - WebSphere Application Server Enterprise、バージョン 5.0.2 (WebSphere Studio Application Developer Integration Edition、バージョン 5.0.1 と併用)

例外については、「*リリース情報*」を参照してください。

注: 統合ブローカーのインストール手順およびその前提条件については、次の資料を参照してください。

WebSphere InterChange Server (ICS) については、「システム・インストール・ガイド (UNIX 版)」または「システム・インストールガイド (Windows 版)」を参照してください。Message Brokers (WebSphere MQ Integrator Broker、WebSphere MQ Integrator、および WebSphere Business Integration Message Broker) の場合は、

「WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド」およびそれぞれの Message Brokers のインストールに関する資料を参照してください。一部の資料は次の Web サイトにあります。
http://www.ibm.com/software/integration/mqfamily/library/manualsa/

WebSphere Application Server については「アダプター実装ガイド (WebSphere Application Server)」および次の場所にある資料を参照してください。
http://www.ibm.com/software/webservers/appserv/library.html

アダプターのプラットフォーム

このアダプターは以下のプラットフォームでサポートされます。

- Windows 2000
- AIX 5.1、5.2
- Solaris 8
- HP-UX 11i

前提条件

コネクターを使用するには、事前に以下の作業を行う必要があります。

- Adapter Development Kit は、アダプターの機能を拡張または変更する予定の場合にのみインストールしてください。

  コネクターを統合ブローカーとは別のマシンで実行する場合、その統合ブローカーと互換するバージョンの Adapter Development Kit をインストールしてください。
- 使用する JDBC ドライバーのインストール
- 必要なベンダー固有ソフトウェアがすべて (JDBC ドライバーに必要とされるものを含めて) インストールされていることの検証

  例えば、Oracle データベース用の JDBC Type 2 ドライバーを使用している場合は、OracleOCI ライブラリーをインストールする必要があります。
- アプリケーションにユーザー・アカウントが存在していることを確認します。

  コネクターは、JDBC アプリケーション内のデータを処理する場合、JDBC 仕様準拠の Oracle 用ドライバーを使用します。コネクターが直接対話するデータベース内のデータを処理するためには、アプリケーションで有効なユーザー・アカウントおよびパスワードへのアクセスが可能でなければなりません。使用するユーザー・アカウントには、アプリケーションのデータベースのデータを検索、挿入、更新、および削除する権限が付与されていなければなりません。このようなアカウントが存在しない場合は、作成する必要があります。
- 接続されたデータベースの文字コード・セットの検証

  Java 仮想マシン (JVM) 内での Java ランタイム環境は、Unicode 文字コード・セットでデータを表します。Unicode には、最もよく知られた文字セット (単一バイトとマルチバイトの両方) の文字エンコードが含まれています。コネクターは、Java で作成されているため、Unicode を認識します。

アダプターのインストール

WebSphere Business Integration Adapter 製品のインストールについては、「*WebSphere Business Integration Adapters インストール・ガイド*」を参照してください。この資料は、次の Web サイトの WebSphere Business Integration Adapters Infocenter にあります。

http://www.ibm.com/websphere/integration/wbiadapters/infocenter

i2 Active Data Warehouse コネクターのインストール済みファイル構造

以下のサブセクションでは、UNIX または Windows システム上での i2 Active Data Warehouse コネクターのインストール済みファイル構造について説明します。

注: 特に指示がない限り、この章の残りのセクションは、UNIX および Windows の両方のコネクターのインストールに適用されます。

インストール済みファイル構造 (UNIX システムの場合)

表 2 に、コネクターが使用する UNIX ファイル構造を示します。

表 2. i2 Active Data Warehouse コネクターのインストール済み UNIX ファイル構造

$*ProductDir* のサブディレクトリー	**説明**
`connectors/i2ADW`	コネクターの CWi2ADW.jar および start_i2ADW.sh ファイルが格納されています。start_i2ADW.sh ファイルは、コネクターのシステム始動スクリプトです。このスクリプトは、汎用コネクター・マネージャー・スクリプトから呼び出されます。「Connector Configurator からインストール (Install from Connector Configurator)」(WebSphere MQ Integrator Broker を統合ブローカーとして使用する場合)、または System Manager の「コネクター・コンフィグレーション (Connector Configuration)」画面 (ICS を統合ブローカーとして使用する場合) をクリックすると、インストーラーがこのコネクター管理スクリプト用にカスタマイズされたラッパーを作成します。 ICS でコネクターが動作している場合、コネクターを始動および停止するには、このカスタマイズされたラッパーを使用してください。WebSphere MQ Integrator Broker でコネクターを使用する場合、このカスタマイズされたラッパーは、コネクターの始動のみに使用します。コネクターを停止するには、`mqsiremotestopadapter` コマンドを使用します。

表 2. *i2 Active Data Warehouse* コネクターのインストール済み *UNIX* ファイル構造 *(続き)*

$*ProductDir* のサブディレクトリー	**説明**
connectors/i2ADW/dependencies	イベント表、アーカイブ表、および固有 ID 表を作成する SQL スクリプトが含まれています。
connectors/messages	i2ADWConnector.txt ファイルと、i2ADWConnector_*ll_TT*.txt ファイル (言語に固有なメッセージ・ファイル (*ll*) と国/地域に固有なメッセージ・ファイル (*TT*)) が含まれます。
repository/i2ADW	CN_i2ADW.txt ファイルが含まれています。
/lib	WBIA.jar ファイルが含まれています。
/bin	CWConnEnv.sh ファイルが含まれています。

コネクターをインストールし終えたら、コネクター構成ツールを使用して、コネクターの始動に必要なカスタマイズされたコネクター・ラッパー (connector_manager_i2ADW) を生成する必要があります。詳細については、「システム・インストール・ガイド *(UNIX 版)*」または「*IBM WebSphere Business Integration Adapters WebSphere MQ Integrator Broker 用インプリメンテーション・ガイド*」を参照してください。

コネクター・コンポーネントのインストールの詳細については、ご使用の統合ブローカーに応じて、以下のいずれかのガイドを参照してください。

- 「システム・インストール・ガイド *(UNIX 版)*」 (ICS を統合ブローカーとして使用している場合)
- 「*IBM WebSphere Business Integration Adapters WebSphere MQ Integrator Broker 用インプリメンテーション・ガイド*」(WebSphere MQ Integrator Broker を統合ブローカーとして使用している場合)

インストール済みファイル構造 (Windows システムの場合)

次の表に、コネクターが使用する Windows ファイル構造を示します。

表 3. *コネクター用としてインストールされた Windows ファイル構造*

%*ProductDir*% のサブディレクトリー	**説明**
connectors¥i2ADW	コネクターの CWi2ADW.jar および start_i2ADW.bat ファイルが格納されています。
connectors¥i2ADW¥dependencies	イベント表、アーカイブ表、および固有 ID 表を作成する SQL スクリプトが含まれています。
connectors¥messages	i2ADWConnector.txt ファイルと、i2ADWConnector_*ll_TT*.txt ファイル (言語に固有なメッセージ・ファイル (*ll*) と国/地域に固有なメッセージ・ファイル (*TT*)) が含まれます。
repository¥i2ADW¥	CN_i2ADW.txt ファイルが含まれています。
¥lib	WBIA.jar ファイルが含まれています。

表 3. コネクター用としてインストールされた Windows ファイル構造 (続き)

%ProductDir% のサブディレクトリー	説明
¥bin	CWConnEnv.bat ファイルが含まれています。

インストーラーによって、IBM WebSphere Business Integration Adapter メニューにコネクター・ファイルのアイコンが追加されます。コネクターをすばやく始動するには、このファイルへのショートカットをデスクトップに作成してください。

コネクター・コンポーネントのインストールの詳細については、ご使用の統合ブローカーに応じて、以下のいずれかのガイドを参照してください。

- 「*システム・インストール・ガイド (Windows 版)*」(ICS を統合ブローカーとして使用している場合)
- 「*IBM WebSphere Business Integration Adapters WebSphere MQ Integrator Broker 用インプリメンテーション・ガイド*」(WebSphere MQ Integrator Broker を統合ブローカーとして使用している場合)

コネクター用のアプリケーションの使用可能化

コネクターによるイベント・デリバリーの処理を可能にするには、事前にデータベースでイベント通知機構をセットアップしておく必要があります。このセットアップを行うには、以下のタスクを完了する必要があります。

- データベース内にイベント表およびアーカイブ表を作成します。
- ユーザーのサイトで稼働しているビジネス・プロセスに必要なビジネス・オブジェクトがサポートされるよう、アプリケーションの表上にデータベース・トリガーをインストールします。ユーザー固有のデータベース・トリガーが作成されていることを前提としています。
- オプションで、WebSphere Business Integration Adapter カウンター表をインストールします。このステップは、ビジネス・オブジェクトの作成時にコネクターに固有の ID を生成させる必要のある場合にのみ実行してください。固有の ID の生成についての詳細は、「UID=CW.uidcolumnname[=UseIfMissing]」パラメーターを参照してください。

以降のセクションでは、イベント表とアーカイブ表の作成および構成について説明します。

イベントおよびアーカイブ表

イベント表は、コネクターがピックアップ用のイベントをキューに入れる場合に使用します。 ArchiveProcessed プロパティーが true または値なしに設定されている場合、コネクターはイベント表内の状況を更新してから、アーカイブ表を使用してイベントを保管します。

コネクターは、イベントごとにビジネス・オブジェクトの名前、動詞、およびキーをイベント表から取得します。コネクターは、この情報を使用して、アプリケーションからエンティティー全体を検索します。イベントが初めてログに記録された後でエンティティーが変更された場合、コネクターは初期イベントおよび以降のすべての変更内容を取得します。つまり、コネクターがエンティティーをイベント表か

ら取得する前に、そのエンティティーが作成され更新されていた場合、1 回の検索で両方のデータ変更内容が取得されることになります。

コネクターによって処理された各イベントの結果としては、以下の 3 種類が考えられます。

- イベントが正常に処理された
- イベントが正常に処理されなかった
- イベントがサブスクライブされなかった (特定の統合ブローカーへのサブスクリプションについては、ブローカーのインプリメンテーション・ガイドを参照)

イベントがコネクターによってピックアップされた後もイベント表から削除されない場合は、そのイベントがイベント表の不必要なスペースを占めてしまうことになります。しかし、そのイベントが削除された場合は、未処理のイベントがすべて失われるため、ユーザーがイベント処理が監査不能になります。これらの理由から、アーカイブ表も作成して ArchiveProcessed プロパティーを true に設定しておくことをお勧めします。イベントはイベント表から削除されるたびに、コネクターによってアーカイブ表に挿入されます。

注: アプリケーション・データベースへのアクセスの問題が原因で、イベント表からのイベントの削除またはアーカイブ表へのイベントの挿入に失敗した場合、コネクターは APPRESPONSETIMEOUT を戻します。

イベントおよびアーカイブ処理の構成

イベントおよびアーカイブの処理を構成するには、構成プロパティーを使用して、以下の情報を指定する必要があります。

- イベント表の名前 (EventTableName)。ビジネス・オブジェクトの要求処理のためにのみコネクターを使用する場合は、このプロパティーの値を指定する必要はありません。
- 間隔 (頻度) (PollFrequency)。
- ポーリング間隔ごとのイベントの数 (PollQuantity)。
- アーカイブ表の名前 (ArchiveTableName)。
- アンサブスクライブされた未処理のイベントをコネクターがアーカイブするかどうか (ArchiveProcessed)。ご使用の統合ブローカー固有のサブスクリプションについては、ブローカーのインプリメンテーション・ガイドを参照してください。
- コネクターの固有 ID。この ID は、複数のコネクターが同じ表をポーリングするときに重要になります (ConnectorID)。

また、EventOrderBy プロパティーの値を指定して、イベントの処理順序を指定することも可能です。これらおよびその他の構成プロパティーに関する詳細については、97 ページの『付録 A. コネクターの標準構成プロパティー』および 19 ページの表 6 を参照してください。

注: イベント表およびアーカイブ表の作成はオプショナルです。しかし、EventTableName の値を指定してもイベントをポーリングするコネクターを使用しなかったり、およびイベント表を作成しなかったりすると、コネクターはタイムアウトになります。そうしたタイムアウトを回避するには、EventTableName の値をヌル (string 型) のままにしておいてください。

デフォルトでは、イベント・キュー表の名前は、xworlds_events になり、アーカイブ・キュー表の名前は、xworlds_ archive_events になります。

コネクターを要求処理用にのみ使用する場合は、コネクターを始動するときや EventTableName の値をヌル (string 型) に設定するときに、-fno オプションを使用してください。

使用するドライバーが Java クラス DatabaseMetaData をサポートしていない場合に、コネクターによるイベント表およびアーカイブ表の存在チェックを回避したいときは、CheckForEventTableInInit の値を false に設定して使用不可にしてください。デフォルトでは、true になります。この値は false に設定しないことを推奨します。

注: ユーザーのサイトで、アーカイブ表にイベントがアーカイブされない場合は、ArchiveProcessed の値を false に設定してください。

イベントおよびアーカイブ表をインストールするための SQL スクリプト

Oracle データベース用のイベント表、アーカイブ表、および固有 ID 表をインストールするスクリプトを以下に示します。

- event_table_oracle.sql
- event_package_oracle.sql
- archive_table_oracle.sql
- uid_table_oracle.sql

これらのファイルは、以下のディレクトリーにあります。

UNIX:

```
connectors/JDBC/dependencies/
```

Windows:

```
connectors¥JDBC¥dependencies¥
```

注: これらのスクリプトは、コネクターに必要とされる表の作成に役立つテンプレートとしてのみ提供されています。他のデータベースの場合、これらをガイドラインとして使用してスクリプトを作成してください。表列の順序およびデータ型は、非常に重要です。正しい順序および型については、『イベントおよびアーカイブ表のスキーマ』を参照してください。

コネクターを実装する DBA または担当者は、特定のインストール要件および照会最適化の要件が満たされるように、これらのスクリプトを変更してください。これらのスクリプトでは、例えば表上に索引が作成されません。照会最適化プログラムを使用してパフォーマンスの改善を図るための索引を作成するのは、コネクターを実装するユーザーの責任です。

イベントおよびアーカイブ表のスキーマ

16 ページの表 4 に、イベント表およびアーカイブ表の列を示します。

表 4. イベントおよびアーカイブ表のスキーマ

名前	説明	型	制約
event_id	イベントの内部 ID	INTEGER	基本キー
connector_id	イベントの宛先となるコネクターの固有 ID。この値は、複数のコネクターが同じ表をポーリングするときに重要になります。	VARCHAR	
object_key	ビジネス・オブジェクトの基本キー。キーは、name_value ペアまたはコロンなどの構成可能な区切り文字で区切られた一連のキー (例: 1000065:10056:2333) として表すことができます。詳細については、EventKeyDel プロパティーを参照してください。	VARCHAR	非ヌル
object_name	ビジネス・オブジェクトの名前	VARCHAR	非ヌル
object_verb	イベントに関連した動詞	VARCHAR	非ヌル
event_priority	コネクターがイベントを優先順位に基づいて取得する場合に使用する、イベントの優先順位 (0 は最高、n は最低)。コネクターは、この値を優先順位を上下させる目的には使用しません。	INTEGER	非ヌル
event_time	イベントが発生した日時	DATETIME	デフォルトの現在日付/時刻 (アーカイブ表の場合、実際のイベント時間)
archive_time	イベントをアーカイブした日時 (アーカイブ表にのみ適用)	DATETIME	アーカイブ日付/時刻
event_status	-2 (統合ブローカーへのイベント送信中のエラー) -1 (イベント処理中のエラー) 0 (ポーリング可能な状態) 1 (統合ブローカーへの送信) 2 (ビジネス・オブジェクトにサブスクリプションがない) 3 (進行中)。この状況はイベント表にのみ使用されます。アーカイブ表には使用されません。	INTEGER	非ヌル
event_comment	イベント・ストリングまたはエラー・ストリングの説明	VARCHAR	

マルチ・ドライバー・サポートの使用可能化

ドライバーを指定するには、以下の手順を実行します。

1. マシン上にドライバーをインストールします。
2. コネクターが実行時に必要とする動的ライブラリーをすべて、製品ディレクトリーの下の connectors¥i2ADW ディレクトリーに保管します。
3. コネクターの始動ファイルを、関係のあるすべてのクラス・パス名 (必要な場合は、ライセンス情報を含む) が JDBCDRIVERPATH 変数に格納されるように編集します。

   UNIX の場合、始動ファイルは以下のようになります。

   ```
   $ProductDir/connectors/i2ADW/start_i2ADW.sh
   ```

   Windows の場合、始動ファイルは以下のようになります。

   ```
   %ProductDir%¥connectors¥i2ADW¥start_i2ADW.bat
   ```

4. JDBCDriverClass 構成プロパティーの値を指定します。

注: サポートされるすべての機能に関して、コネクターは JDBC 2.0 以降の仕様に準拠するいずれのドライバーでも動作可能です。特定の機能がドライバーでサポートされていない場合、コネクターは正しく動作しません。例えば、ドライバーが i2ADWODA によって使用されるすべてのメソッド呼び出しをサポートしない場合、i2ADWODA ログにドライバーがサポートしないプロセスが示されます。この場合、別のドライバーを使用する必要があります。

カスタム・ビジネス・オブジェクト・ハンドラー・クラスの使用可能化

コネクターは、カスタム・ビジネス・オブジェクトのハンドラー・クラス CustomBOH をサポートしています。このクラスは、JDBCBOhandlerInterface インターフェースを実装します。このインターフェースの構文は以下のとおりです。

```
public interface JDBCBOhandlerInterface{
   public int doVerbForCustom(CWConnectorBusObj busObj) throws
     VerbProcessingFailedException, ConnectionFailureException;
}
```

doVerbForCustom メソッドを実装するときは、2 つの例外がスローされても、キャッチされないことを確認してください。各例外をスローする前に、例外の状況およびメッセージの設定も行ってください。

- VerbProcessingFailedException: 動詞によって指定された操作が失敗した場合にスローされます。
- ConnectionFailureException: コネクターがアプリケーションとの間の接続を確立できない場合にスローされます。

このビジネス・オブジェクト・ハンドラーがコネクターによってサポートされるようにするには、以下のようにします。

- 動詞アプリケーション固有の情報に CustomBOH クラス名を指定します。

  コネクターは、カスタム・ビジネス・オブジェクトのハンドラー・クラスの名前を、動詞アプリケーション固有の情報から取得します。以下の構文を使用してください。

  CustomBOH=*customBOhandlerClassName*

  例えば、動詞アプリケーション固有の情報が以下のように指定されているとします。

  CustomBOH=JDBCBOhandlerForOverrideSQL

  この場合、JDBCBOhandlerForOverrideSQL は、カスタム・ビジネス・オブジェクトのハンドラー・クラスの名前です。
- CustomBOH は、com.crossworlds.connectors.JDBC に属します。

コネクターは、「CustomBOH=」を動詞アプリケーション固有の情報の中に検出し、com.crossworlds.connectors.JDBC パッケージ内にクラスを検出した場合、カスタム・ビジネス・オブジェクトのハンドラーを実行します。コネクターは、CustomBOH が検出されない場合、クラスが見つからないというエラーをスローします。

コネクターの構成

コネクターの標準構成プロパティー、およびコネクター固有の構成プロパティーを実行する際には、事前に設定しておく必要があります。コネクターの構成プロパティーを設定するには、以下のツールのいずれかを使用します。

- Connector Configurator (ICS が統合ブローカーである場合) -- このツールには、System Manager からアクセスする。
- Connector Configurator (WebSphere MQ Integrator Broker が統合ブローカーである場合) -- このツールには、IBM WebSphere Business Integration Adapter プログラム・フォルダーからアクセスする。 Connector Configurator の詳細については、117 ページの『付録 B. Connector Configurator』を参照してください。

標準コネクター・プロパティー

標準構成プロパティーには、コネクターが使用する情報が用意されています。これらのプロパティーの詳細については、97 ページの『付録 A. コネクターの標準構成プロパティー』を参照してください。

重要: i2 ADW 用コネクターは ICS と WebSphere MQ Integrator Broker の両方の統合ブローカーをサポートするため、このコネクターには、両方のブローカーに関する構成プロパティーが関係します。

さらに、IBM WebSphere Business Integration Adapter for i2 Active Data Warehouse に固有な構成情報については、表 5 を参照してください。この表に示されている情報は、付録に収録されている情報を補足するものです。

表 5. このコネクターに固有のプロパティー情報

プロパティー	注
CharacterEncoding	CharacterEncoding プロパティーはこのコネクターによって使用されません。
Locale	このコネクターは国際化されているため、Locale プロパティーの値は変更可能です。 **注:** WebSphere MQ Integrator Broker をブローカーとして使用している場合は、アダプター、ブローカー、およびすべてのアプリケーションで同一のロケールを使用する必要があります。

コネクター固有のプロパティー

コネクター固有の構成プロパティーには、コネクターが実行時に必要とする情報が用意されています。コネクター固有の構成プロパティーは、エージェントを再コーディングまたは再ビルドせずに、コネクター内部の静的情報またはロジックを変更する手段にもなっています。

19 ページの表 6 に、コネクターに対するコネクター固有の構成プロパティーを示します。プロパティーの説明については、以下の各セクションを参照してください。

表 6. コネクター固有のプロパティー

名前	指定可能な値	デフォルト値	必須
ApplicationPassword	*コネクターのユーザー・アカウント用パスワード*		はい*
ApplicationUserName	*コネクターのユーザー・アカウントの名前*		はい*
ArchiveProcessed	`true` または `false`	true	いいえ
ArchiveTableName	*アーカイブ・キュー表の名前*	`xworlds_archive_events`	Archive Processed が `true` の場合は、はい
AutoCommit	`true` または `false`	`false`	いいえ
CheckforEventTableInInit	`true` または `false`	`true`	いいえ
ChildUpdatePhyDelete	`true` または `false`	`false`	いいえ
CloseDBConnection	`true` または `false`	`false`	いいえ
ConnectorID	*コネクターの固有 ID*	ヌル	いいえ
DatabaseURL	*データベース・サーバーの名前*		はい
DateFormat	*時間パターン・ストリング*	`MM/dd/yyyy HH:mm:ss`	いいえ
DriverConnectionProperties	*追加の JDBC ドライバー接続プロパティー*		いいえ
DriverSupportForLong	`true` または `false`	`false`	いいえ
EventKeyDel	*イベント表のオブジェクト・キー欄の区切り文字、または文字*	セミコロン (`;`)	いいえ
EventOrderBy	`none` または *`ColumnName`* [, *`ColumnName,...`*]		いいえ
EventQueryType	`Fixed` または `Dynamic`	`Fixed`	いいえ
EventTableName	*イベント・キュー表の名前*	`xworlds_events`	ポーリングが必須の場合ははい、ポーリングが必須でない場合はヌル (string 型)
JDBCDriverClass	ドライバーのクラス名		はい
MaximumDatabaseConnections	*同時データベース接続の数*	`5`	はい
PingQuery	`SELECT 1 FROM<`*`tablename`*`>;`		いいえ
PollQuantity	`1` から `500` の値	`1`	いいえ
PreserveUIDSeq	`true` または `false`	`true`	いいえ
QueryTimeOut	秒単位の整数値		いいえ
RDBMS.initsession	すべてのデータベース・セッションを初期化する *SQL* ステートメント		いいえ
RDBMSVendor	`Oracle`	`Oracle`	はい
ReplaceAllStr	`true` または `false`	`false`	いいえ
ReplaceStrList	*単一の文字、文字区切り文字、および文字の置換ストリングで構成されるセット。また、そのような複数のセットと、それらを区切る終了区切り文字。*	Q,DSQ 注: コネクター構成ツールでは、これらの文字は、単一引用符、それに続くコンマ、その後続の 2 つの単一引用符で表されます。	いいえ
RetryCountAndInterval	*カウント、間隔 (秒数)*	3,20	いいえ
ReturnDummyBOForSP	`true` または `false`	`false`	いいえ

表 6. コネクター固有のプロパティー (続き)

名前	指定可能な値	デフォルト値	必須
SchemaName	WebSphere Business Integration Adapter イベントが常駐するスキーマ		いいえ
SelectiveCommitForPoll	true または false	false	いいえ
SPBeforePollCall	ポーリング呼び出しのたびに実行されるストアード・プロシージャーの名前		いいえ
StrDelimiter	ReplaceStrList プロパティーに使用される文字区切り文字および終了区切り文字	,:	いいえ
TimingStats	0、1、2	0	いいえ
UniqueIDTableName	*ID の生成に使用される表の名前*	xworlds_uid	いいえ
UseDefaults	true または false	false	はい
UseDefaultsWhenPolling	true または false	true	いいえ
	true または false	false	いいえ

* トラステッド認証を使用する場合、ApplicationPassword と ApplicationUserName は不要です。

ApplicationPassword

コネクターのユーザー・アカウント用パスワード。

デフォルト値はありません。

ApplicationUserName

コネクターのユーザー・アカウントの名前。

デフォルト値はありません。

ArchiveProcessed

現在サブスクリプションが存在しないイベントを、コネクターにアーカイブさせるかどうかを指定します。

このプロパティーを true に設定すると、イベントがイベント表から削除される前にアーカイブ表に挿入されます。

このプロパティーを false に設定すると、コネクターがアーカイブ処理を実行しなくなります。この場合、ArchiveTableName プロパティーの値は検査されません。ArchiveProcessed を false に設定すると、コネクターは以下のような動作をします。

- イベントが正常に処理された場合、そのイベントはコネクターによってイベント表から削除されるため、アーカイブされません。
- コネクターは、イベントのビジネス・オブジェクトをサブスクライブしない場合、そのイベントをイベント表に残したままにして、イベント状況を「Unsubscribed」に変更します。ご使用の統合ブローカー固有のサブスクリプションについては、ブローカーのインプリメンテーション・ガイドを参照してください。

- ビジネス・オブジェクトの処理中に問題が発生した場合、イベントをイベント表に残し、イベントの状況を Error にします。

このプロパティーが false に設定されていて、ポーリング量が少ない場合、コネクターはイベント表をポーリングしている外観を呈しますが、繰り返し同じイベントをピックアップしているだけです。

このプロパティーが値を持たない場合、コネクターはその値を true と見なします。ArchiveTableName プロパティーも値を持たない場合、コネクターはアーカイブ表の名前を xworlds_archive_events と見なします。

デフォルト値は true です。

ArchiveTableName

アーカイブ・キュー表の名前。

ArchiveProcessed プロパティーが false に設定されている場合は、このプロパティーの値を設定する必要はありません。

デフォルトの名前は xworlds_archive_events です。

AutoCommit

このプロパティーによって、AutoCommit 設定が構成可能になります。true に設定されている場合、トランザクションがすべて自動的にコミットされます。一部のデータベース (Sybase など) は、AutoCommit を true に設定する必要があります。false に設定すると、Sybase 上のストアード・プロシージャーが失敗します。

データベース接続が失われた場合、AutoCommit が false に設定されていれば、コネクターは新規の接続を作成して完全処理を再始動しようとします。新規の接続が無効な場合、または AutoCommit が true に設定されている場合は、コネクターは APPRESPONSETIMEOUT を戻します。この結果、コネクターは終了します。

デフォルト値は false です。

CheckforEventTableInInit

このコネクター・プロパティーを false に設定すると、コネクターは初期化中に、イベント表およびアーカイブ表の存在チェックを実行しません。ご使用の JDBC ドライバーが JDBC クラス DatabaseMetaData をサポートしている場合は、このプロパティーを常に true に設定することをお勧めします。

プロパティーが false に設定されている場合、コネクターは EventTable および ArchiveTable の存在チェックを実行しませんが、イベント表およびアーカイブ表はコネクターの初期化処理中に使用されるため、常に存在していなければなりません。イベント表およびアーカイブ表がコネクターの初期化処理中に使用されるのを防ぐには、プロパティー EventTableName をヌルに設定します。

デフォルト値は true です。

ChildUpdatePhyDelete

更新操作中に、子ビジネス・オブジェクトによって表されるデータが、入力方向のビジネス・オブジェクトから脱落してしまい、データベース内には存在している場合に、コネクターにどう処理させるかを指定します。

このプロパティーを true に設定すると、コネクターはデータベースからデータ・レコードを物理的に削除します。

このプロパティーを false に設定すると、コネクターは状況列を適切な値に設定して、データベースからデータ・レコードを論理的に削除します。コネクターは、ビジネス・オブジェクト・レベルのアプリケーション固有の情報に指定された StatusColumnValue (SCN) パラメーターから、状況列の名前およびその値を取得します。詳細については、64 ページの『ビジネス・オブジェクト・レベルのアプリケーション固有情報』を参照してください。

デフォルト値は `false` です。

CloseDBConnection

このプロパティーによって、データベース接続終了が構成可能になります。`true` に設定されている場合、サービス呼び出し要求およびポーリング呼び出しのたびに、データベース接続が終了します。このプロパティーを `true` に設定すると、パフォーマンスが低下するため、お勧めできません。

デフォルト値は `false` です。

ConnectorID

コネクターの固有 ID。この ID は、コネクターの特定インスタンスに対するイベントを検索するときに役立ちます。

デフォルト値はヌルです。

DatabaseURL

コネクターの接続先となるデータベース・サーバーの名前。

- Oracle thin ドライバーを使用する場合、推奨される URL は以下のとおりです。

  `jdbc:oracle:thin:@`*`MachineName`*`:`*`PortNumber`*`:`*`DBname`*

- WebSphere ブランドの Oracle ドライバーを使用する場合、推奨される URL は以下のとおりです。

  `jdbc:ibm-crossworlds:oracle://`*`MachineName`*`:1521;SID=`*`DBname`*

コネクターが処理を正常に実行するためには、ユーザーがこの値を指定しておく必要があります。

DateFormat

コネクターが受け取って戻すことが予期される日付形式を指定します。このプロパティーは、23 ページの表 7 に記載されている構文に基づくフォーマットをすべてサポートしています。

表 7 では、時刻パターン・ストリングを使用して時刻 DateFormat 構文を定義します。このパターンでは、ASCII 文字がすべて、パターン文字として予約されています。

表 7. 時刻形式構文

記号	意味	表示	例
G	元号指定子	(テキスト)	AD
y	年	(数値)	1996
M	当年の月	(テキスト & 数値)	July & 07
d	当月の日付	(数値)	10
h	午前/午後の時刻 (1 から 12)	(数値)	12
H	1 日の時刻 (0 から 23)	(数値)	0
m	時間あたりの分	(数値)	30
s	分あたりの秒数	(数値)	55
S	ミリ秒	(数値)	978
E	曜日	(テキスト)	Tuesday
D	通年の日数	(数値)	189
F	当月の第何週目の曜日	(数値)	2 (2nd Wed in July)
w	通年の第何週	(数値)	27
W	当月の第何週	(数値)	2
a	am/pm マーカー	(テキスト)	PM
k	1 日の時刻 (1 から 24)	(数値)	24
K	午前/午後の時刻 (0 から 11)	(数値)	0
z	時間帯	(テキスト)	Pacific Standard Time
'	テキストのエスケープ	(区切り文字)	
''	単一引用符	(リテラル)	‘

表 8. US ロケールを使用した例

形式パターン	結果
"yyyy.MM.dd G 'at' hh:mm:ss z"	1996.07.10 AD at 15:08:56 PDT
"EEE, MMM d, ''yy"	Wed, July 10, ‘96
"h:mm a"	12:08 PM
"hh 'o''clock' a, zzzz"	12 o’clock PM, Pacific Daylight Time
"K:mm a, z"	0:00 PM, PST
"yyyy.MMMMM.dd GGG hh:mm aaa"	1996.July.10 AD 12:08 PM

DriverConnectionProperties

JDBC ドライバーには、ユーザー名とパスワード以外にも、追加のプロパティーまたは情報が必要になる場合があります。DriverConnectionProperties コネクター・プロパティーは、JDBC ドライバーに必要な、名前と値のペアとしての追加のプロパティーを取ります。プロパティーは以下のように指定してください。

```
property1=value1[;property2=value2...]
```

プロパティーは、セミコロンで区切られた名前と値のペアとして指定する必要があります。プロパティーとプロパティー値は、等号で (余分なスペースを入れずに) 区切られます。

例えば、ライセンス情報およびポート番号を必要とする JDBC ドライバーを想定します。ライセンス情報に予期されるプロパティー名を MyLicense とし、その値を ab23jk5 とします。ポート番号に予期されるプロパティー名を PortNumber とし、その値を 1200 とします。DriverConnectionProperties は、MyLicense=ab23jk5;PortNumber=1200 という値に設定する必要があります。

DriverSupportForLong

整数パラメーターが PreparedStatements でどのように処理されるかを指定します。このプロパティーが true の場合、

```
setLong
```

を使用して整数パラメーターが設定されます。このプロパティーが false の場合、

```
setInt
```

を使用してパラメーターが設定されます。

デフォルト値は true です。

EventKeyDel

イベント表の object_key 列に複数の属性値が含まれる場合は、区切り文字を指定してください。

トリガー・アプリケーションで、作成、更新、または削除されたビジネス・オブジェクトを検索するには、2 とおりの方法があります。

- 1 つは、object_key 列に、ビジネス・オブジェクトのキーとなっている属性の値を取り込む方法です。EventKeyDel 構成プロパティーを、キー・フィールドの一部にはなっていない単一の文字に設定します。**例えば、**区切り文字を「**;**」と指定した場合、object_key は、*xxx*;*123* のようになります。
- もう 1 つは、object_key 列に、ビジネス・オブジェクトのいずれかの属性の値を取り込む方法です。これらの値は name_value のペアとして表されます。最初の区切り文字は name_value 用、2 番目はキー用です。例えば、区切り文字を「**=;**」と指定した場合、object_key は、*CustomerName=xxx*;*CustomerId=123*; のようになります。

  例えば、区切り文字を「**=:**」と指定した場合、object_key は、*CustomerName=xxx*:*CustomerId=123*: のようになります。

注: キー値を定義する順序は、ビジネス・オブジェクト内のキー属性と同じ順序に従わなければなりません。

重要: Date 属性データを使用する場合、コロン (:) を区切り文字として使用しないでください。コロンが属性データに組み込まれてしまう可能性があるためです。

デフォルト値はセミコロン (;) です。これは、name_value ペアではなく、キーがベースとなっています。

EventOrderBy
イベントの順序付けをオフにするか、またはデフォルトの順序とは異なるイベント処理順序を指定します。

デフォルトでは、ポーリングのたびにコネクターは PollQuantity プロパティーに指定されたイベントの番号のみをプルし、イベント表の event_time 列および event_priority 列内の値でイベント処理を順序付けます。

コネクターにイベントを順序付けさせないようにするには、このプロパティーの値を none に設定してください。

コネクターにイベント表内にある複数の列による順序付けを実行させるには、それらの列の名前を指定してください。列名はコンマ (,) で区切ってください。このプロパティーに値を指定すると、デフォルト値が上書きされます。

このプロパティーにはデフォルト値が存在しません。

EventQueryType
EventQueryType プロパティーは、イベント表からイベントを検索する照会をコネクターに動的に生成させるか、組み込み照会をコネクターに使用させるかを指定する場合に使用します。コネクターは、動的に生成される照会に関してはイベントの構造をイベント表内の列に一致させます。表列内のデータの順序は、非常に重要です。正しい順序については、15 ページの『イベントおよびアーカイブ表のスキーマ』を参照してください。

EventQueryType の値が Fixed (string 型) の場合は、デフォルトの照会が実行されます。Dynamic (string 型) に設定されている場合は、25 ページの『EventTableName』プロパティーに指定されている表から列名を取得して、新規の照会が作成されます。

イベント表列名は変更できます。ただし、列の順序とデータ型は、イベント表の作成のセクションで指定したものと同じでなければなりません。デフォルトの照会または動的に生成された照会には、25 ページの『EventOrderBy』が追加されます。

EventQueryType プロパティーを追加しない場合、またはこのプロパティーに値が含まれていない場合は、デフォルトとして Fixed が使用されます。

デフォルト値は Fixed (string 型) です。

EventTableName
イベント・キュー表の名前。コネクターのポーリング機構によって使用されます。

デフォルトの名前は xworlds_events です。

コネクターによるポーリングをオフにするには、この名前をヌル (string 型) に設定してください。この設定により、イベント表およびアーカイブ表の存在に関する検証が行われなくなります。

ユーザー定義イベント表の場合は、event_id が INTEGER、BIGINT、NUMERIC、VARCHAR のいずれかの JDBC タイプにマップされるようにしてください。

JDBCDriverClass

ドライバーのクラス名を指定します。特定の JDBC ドライバーを使用するには、この構成プロパティーにドライバーのクラス名を指定してください。例えば、Oracle thin ドライバーを指定するには、このプロパティーの値を `oracle.jdbc.driver.OracleDriver` に設定します。

詳細については、 16 ページの『マルチ・ドライバー・サポートの使用可能化』を参照してください。

デフォルト値は提供されていません。

MaximumDatabaseConnections

同時データベース接続の最大許容数を指定します。この数に 1 を加算した合計が、実行時に開いているデータベース接続の数となります。

26 ページの『PreserveUIDSeq』プロパティーが false に設定されている場合は、この数に 2 を加算した合計が、実行時に開いているデータベース接続の数になります。

デフォルト値は 5 です。

PingQuery

コネクターがデータベース接続をチェックするときに使用する SQL ステートメントまたはストアード・プロシージャーを指定します。

次に示すのは、ping 照会として使用される SQL ステートメントの一例です。

```
SELECT 1 FROM <tablename>
```

次に示すのは、Oracle データベースで ping 照会として使用されるストアード・プロシージャー・コールの一例 (sampleSP) です。

```
call sampleSP( )
```

ストアード・プロシージャー・コールに出力パラメーターを指定することはできません。データベースによって入力パラメーターが必要とされる場合、入力値は、ping 照会の一部として指定する必要があります。その例を以下に示します。

```
call checkproc(2)
```

デフォルト値はありません。詳細については、 6 ページの『データベース接続不能の処理』および 94 ページの『アプリケーションへの接続不可』を参照してください。

PollQuantity

コネクターがポーリング間隔で検索する、データベース表内の行の数。許容値は 1 から 500 です。

デフォルトは、1 です。

PreserveUIDSeq

着信した固有 ID のシーケンスを固有 ID 表に保存するかどうかを指定します。

true に設定されている場合、宛先アプリケーションでビジネス・オブジェクトが正常に処理されるまで、固有 ID はコミットされません。トランザクションがコミットされるまでは、固有 ID 表にアクセスしようとする他のプロセスはすべて待機しなければなりません。

false に設定されている場合、固有 ID はビジネス・オブジェクトからの要求があったときにコミットされます。ビジネス・オブジェクト処理および固有 ID 処理には、それぞれ独自のトランザクション・ブロック (コネクターに対して内部的) があります。固有 ID 表に関連するトランザクションがそれ自身の接続を持つ場合のみ、この固有 ID のコミットが可能になります。

注: このプロパティーがコネクターの構成に追加されていない場合でも、そのデフォルトの動作は、プロパティーが追加されていて true に設定されている場合と同様になります。また、21 ページの『AutoCommit』 が true に設定されている場合は、コネクターは PreserveUIDSeq が false に設定されている場合と同様に振る舞います。

26 ページの『PreserveUIDSeq』プロパティーが false に設定されている場合は、この数に 2 を加算した合計が、実行時に開いているデータベース接続の数になります。

デフォルト値は true です。

QueryTimeOut

このプロパティーの値は秒単位の整数値であり、指定された秒数にすべての検索の QueryTimeOut を設定します。値を指定しない場合は、照会にタイムアウトを設定しないことが暗黙指定されます。照会の処理が、指定された秒数より長引く場合は、データベースにより、キャプチャーされる SQL 例外が生成されます。関連メッセージが、ログ・ファイルに記録されます。

デフォルト値は提供されていません。

RDBMS.initsession

データベースでのすべてのセッションを初期化する SQL ステートメント。コネクターは照会を取得して始動時に実行します。この照会を実行しても戻り値は得られません。プロパティー名は必要ですが、値は必要ではありません。

デフォルト値はありません。

RDBMSVendor

コネクターの特別な処理に使用される RDBMS を指定します。コネクターが処理を正常に実行するためには、その値が必要になります。

デフォルトは Oracle です。

ReplaceAllStr

28 ページの『ReplaceStrList』プロパティー内に識別される各文字のすべてのインスタンスを、そのプロパティー内に指定された置換ストリングでコネクターに置換させるかどうかを指定します。コネクターは、各属性の AppSpecificInfo プロパティーの ESC=[true | false] パラメーターに値が含まれていない場合にのみ、

ReplaceAllStr を評価します。つまり、ESC パラメーターが指定されている場合、そのパラメーター値は、ReplaceAllStr プロパティーの値セットよりも優先されます。ReplaceAllStr の値をコネクターに使用させるには、ESC パラメーターが指定されていないことを確認してください。

ReplaceAllStr のデフォルト値は `false` です。

注: ESC パラメーター、ReplaceAllStr プロパティー、および ReplaceStrList プロパティーは、データベースのエスケープ文字機能をサポートしています (単一引用符のエスケープなど)。 JDBC ドライバーによって提供される Prepared Statements でも同じ機能が利用できるため、今後リリースされるコネクターでは、こうしたプロパティーのサポートが廃止される予定です。コネクターは現在、JDBC の Prepared ステートメントの使用をサポートしています。

ReplaceStrList

置換対象の個々の文字、文字区切り文字、および置換ストリングでそれぞれ構成される、1 つまたは複数の置換セットを指定します。属性の AppSpecificInfo プロパティーの `ESC=[true | false]` パラメーターの値、またはコネクターの ReplaceAllStr プロパティーの値が指定されている場合にのみ、コネクターは属性値に対してこの置換を実行します。

注: ESC パラメーター、ReplaceAllStr プロパティー、および ReplaceStrList プロパティーは、データベースのエスケープ文字機能をサポートしています (単一引用符のエスケープなど)。 JDBC ドライバーによって提供される Prepared Statements でも同じ機能が利用できるため、今後リリースされるコネクターでは、こうしたプロパティーのサポートが廃止される予定です。コネクターは現在、JDBC の Prepared ステートメントの使用をサポートしています。

この属性の構文は以下のとおりです。

single_char1,substitution_str1[*:single_char2,substitution_str2*[*:...*]]

ここで、

single_char	置換の対象となる文字。
substitution_str	コネクターが文字の置き換えに使用する置換ストリング。
,	文字区切り文字。置換対象の文字とそれに取って代わるストリングを区切ります。デフォルトでは、文字区切り文字は コンマ (,) になります。この区切り文字を構成するには、`StrDelimiter` プロパティー内の最初の区切り文字を設定します。
:	終了区切り文字。置換対象の文字、文字区切り文字、および置換ストリングでそれぞれ構成される置換セットを区切ります。デフォルトでは、終了区切り文字はコロン (:) になります。この区切り文字を構成するには、StrDelimiter プロパティー内の 2 番目の区切り文字を設定します。

例えば、1 つのパーセント記号 (%) を 2 つのパーセント記号 (%%) で置き換え、脱字記号 (^) を円記号および脱字記号 (¥^) で置き換えたいと仮定します。デフォルトでは、StrDelimiter は文字区切り文字としてコンマ (,) を、終了区切り文字としてコロン (:) を指定します。デフォルトの区切り文字を保持する場合は、ReplaceStrList の値として以下のストリングを使用してください。

```
%,%%:^,¥^
```

注: コネクター構成ツールの制約のため、単一引用符の入力はできません。その理由から、単一引用句を文字 Q で表し、2 つの単一引用句を文字 DSQ で表す必要があります。上の例では、単一引用符 (') を 2 つの単一引用符 ('') で置き換えたい場合は、Q,DSQ:%,%%:^,¥^ という表記を使用してください。

RetryCountAndInterval

コネクターが更新操作の実行中にデータをロックできないときに使用する試行回数、および間隔 (秒) を指定します。

コネクターは更新を実行する前に、更新に関連する列をロックして、現在のデータを検索しようとします。コネクターは、行をロックできない場合、この構成プロパティーに指定されたカウントおよび間隔に対するロックを再び取得しようとします。コネクターは、ここに指定された値の範囲内でロックを取得できない場合、最終的にタイムアウトになります。

回数、間隔 (秒) というフォーマットで値を指定します。例えば、値が 3,20 の場合、試行回数として 3 回、間隔として 20 秒が指定されていることになります。

デフォルトは 3,20 です。

ReturnDummyBOForSP

このプロパティーは、結果のセットが空の場合でも出力パラメーターを戻すために使用されます。

RetrieveSP の場合、結果のセットが戻されます。結果のセットが空の場合は、ビジネス・オブジェクトが生成されず、プロシージャー呼び出しの戻す出力パラメーターもリトリーブできません。ただし、ReturnDummyBOForSP が true の場合は、対応する属性に読み込まれた出力パラメーターと入出力パラメーターの値を持つダミーのビジネス・オブジェクトが戻されます。

デフォルト値は false です。

SelectiveCommitForPoll

データベースへのコミットをいつ行うかを指定します。true の場合、イベントの処理終了後、コミットは 1 回のみ行われます。false の場合は、データベースのコミットがイベント処理の各段階で行われるという標準的な振る舞いとなります。

デフォルト値は false です。

SchemaName

このプロパティーは、特定のスキーマのみを対象にその内部にあるイベント表、およびアーカイブ表の検索を制限します。このプロパティーを追加しない場合、または空のままにした場合、ユーザーからのアクセスが可能なすべてのスキーマが検索

されます。この SchemaName は、イベント表およびアーカイブ表にアクセスするための照会を作成するときにも使用されます。

Oracle データベースでは、スキーマ名がサポートされています。

デフォルト値は提供されていません。

SPBeforePollCall

このプロパティーは、ポーリング呼び出しのたびに実行されるストアード・プロシージャーの名前を指定します。 SPBeforePollCall プロパティーが値 (ストアード・プロシージャーの名前) を持つ場合、ポーリング呼び出しが開始されるたびに、コネクターはそのストアード・プロシージャーに対して呼び出しを行い、コネクターのプロパティー (ConnectorID および PollQuantity) の値を渡します。プロシージャーによって PollQuantity 行数が更新され、「connector-id」列が ConnectorID に設定されます。ここで、status=0、connector-id はヌルです。これにより、コネクターの負荷平準化が可能になります。

注: ポーリング呼び出しが完了前に失敗した (データベースがダウンした、または接続が失われた) 場合、connector-id は設定されたままになります。この結果、ポーリング中にレコードがスキップされることがあります。この理由から、イベント表内にある status=0 のすべてのレコードに対して、connector-id を定期的にヌルにリセットすることをお勧めします。

StrDelimiter

ReplaceStrList プロパティー内に使用する文字区切り文字、および終了区切り文字を指定します。

- 文字区切り文字を使用することで、置換対象の文字とそれに取って代わるストリングが区切られます。文字区切り文字は、このプロパティーの値の先頭 (左側) 位置を占め、デフォルトとしてコンマ (,) が使用されます。
- 終了区切り文字を使用することで、置換対象の文字、文字区切り文字、および置換ストリングでそれぞれ構成される置換セットが区切られます。終了区切り文字は、このプロパティーの値の 2 番目の (右側) 位置を占め、デフォルトとしてコロン (:) が使用されます。

これらの区切り文字の両方または一方には、ユーザー固有の値を指定できます。その指定を行う場合は、それらの値の間にスペースなどの文字を指定しないでください。

デフォルト値は、コンマおよびその直後にコロンを続けた値 (,:) です。

TimingStats

コネクターが各動詞の操作タイミングを調整して、問題を探せるようにします。以下の設定が可能です。

0 (タイミング統計が存在しない)

1 (階層ビジネス・オブジェクト全体に対する動詞操作の際に、その入り口点および出口点でタイミングを表示する)

2 (階層ビジネス・オブジェクト内の個々のビジネス・オブジェクトに対する動詞操作のたびに、その入り口点および出口点でタイミングを表示する)

タイミング・メッセージは、トレース・メッセージではなくログ・メッセージです。トレース・レベルに依存せずにオン/オフの切り替えができます。

デフォルト値は 0 です。

UniqueIDTableName

固有 ID の生成に使用される最新の値を含んだ表を指定します。デフォルトでは、この表の列は 1 つです (id)。この表をカスタマイズすることにより、UID (固有 ID) の生成を必要とする属性ごとに列を 1 つずつ追加することができます。

デフォルト値は `xworlds_uid` です。

UseDefaults

UseDefaults が true に設定されている場合、または UseDefaults が設定されていない場合、コネクターは、ビジネス・オブジェクトの各属性に有効な値またはデフォルト値が指定されているかどうかを検査します。値が指定されていない場合はその後で Create が実行されますが、値が指定されていた場合 Create は失敗します。

UseDefaults が false に設定されている場合、コネクターは、ビジネス・オブジェクトの各属性に有効な値が指定されているかどうかのみを検査します。有効な値が指定されていない場合、Create 操作は失敗します。

デフォルト値は `false` です。

UseDefaultsForCreatingChildBOs

UseDefaultsForCreatingChildBOs が設定されていないか、または `true` に設定されている場合、コネクターは、ビジネス・オブジェクトの各属性に有効な値またはデフォルト値が指定されているかどうかを検査します。値が指定されていない場合はその後で Create が実行されますが、値が指定されていた場合 Create は失敗します。

UseDefaultsForCreatingChildBOs が `false` に設定されている場合、コネクターは、ビジネス・オブジェクトの各属性に有効な値が指定されているかどうかのみを検査します。有効な値が指定されていない場合、Create 操作は失敗します。

UseDefaultsWhenPolling

UseDefaultsWhenPolling が true に設定されている場合、ビジネス・オブジェクトは、デフォルト値が設定された後、データベースから検索され、サーバーにディスパッチされます。

UseDefaultsWhenPolling が false に設定されている場合、ビジネス・オブジェクトは、デフォルト値が設定されないで、データベースから検索され、サーバーにディスパッチされます。

デフォルト値は true です。

注: このコネクター固有の構成プロパティーは、UseDefaultsForRetrieve に代わります。

複数のコネクター・インスタンスの作成

コネクターの複数のインスタンスを作成する作業は、いろいろな意味で、カスタム・コネクターの作成と同じです。以下に示すステップを実行することによって、コネクターの複数のインスタンスを作成して実行するように、ご使用のシステムを設定することができます。次のようにする必要があります。

- コネクター・インスタンス用に新規ディレクトリーを作成します。
- 必要なビジネス・オブジェクト定義が設定されていることを確認します。
- 新規コネクター定義ファイルを作成します。
- 新規始動スクリプトを作成します。

新規ディレクトリーの作成

それぞれのコネクター・インスタンスごとにコネクター・ディレクトリーを作成する必要があります。このコネクター・ディレクトリーには、次の名前を付けなければなりません。

```
ProductDir¥connectors¥connectorInstance
```

ここで connectorInstance は、コネクター・インスタンスを一意的に示します。

コネクターに、コネクター固有のメタオブジェクトがある場合、コネクター・インスタンス用のメタオブジェクトを作成する必要があります。メタオブジェクトをファイルとして保管する場合は、次のディレクトリーを作成して、ファイルをそこに格納します。

```
ProductDir¥repository¥connectorInstance
```

ビジネス・オブジェクト定義の作成

各コネクター・インスタンスのビジネス・オブジェクト定義がプロジェクト内にまだ存在しない場合は、それらを作成する必要があります。

1. 初期コネクターに関連付けられているビジネス・オブジェクト定義を変更する必要がある場合は、適切なファイルをコピーし、Business Object Designer を使用してそれらのファイルをインポートします。初期コネクターの任意のファイルをコピーできます。変更を加えた場合は、名前を変更してください。
2. 初期コネクターのファイルは、次のディレクトリーに入っていなければなりません。

   ```
   ProductDir¥repository¥initialConnectorInstance
   ```

   作成した追加ファイルは、ProductDir¥repository の適切な connectorInstance サブディレクトリー内に存在している必要があります。

コネクター定義の作成

Connector Configurator 内で、コネクター・インスタンスの構成ファイル (コネクター定義) を作成します。これを行うには、以下のステップを実行します。

1. 初期コネクターの構成ファイル (コネクター定義) をコピーし、名前変更します。
2. 各コネクター・インスタンスが、サポートされるビジネス・オブジェクト (および関連メタオブジェクト) を正しくリストしていることを確認します。

3. 必要に応じて、コネクター・プロパティーをカスタマイズします。

始動スクリプトの作成

始動スクリプトは以下のように作成します。

1. 初期コネクターの始動スクリプトをコピーし、コネクター・ディレクトリーの名前を含む名前を付けます。

   dirname

2. この始動スクリプトを、32 ページの『新規ディレクトリーの作成』で作成したコネクター・ディレクトリーに格納します。
3. 始動スクリプトのショートカットを作成します (Windows のみ)。
4. 初期コネクターのショートカット・テキストをコピーし、新規コネクター・インスタンスの名前に一致するように (コマンド行で) 初期コネクターの名前を変更します。

これで、ご使用の統合サーバー上でコネクターの両方のインスタンスを同時に実行することができます。

カスタム・コネクター作成の詳細については、「*コネクター開発ガイド (C++ 用)*」または「*コネクター開発ガイド (Java 用)*」を参照してください。

コネクターの開始

コネクターは、**コネクター始動スクリプト**を使用して明示的に始動する必要があります。始動スクリプトは、次に示すようなコネクターのランタイム・ディレクトリーに存在していなければなりません。

ProductDir¥connectors¥*connName*

ここで、*connName* はコネクターを示します。始動スクリプトの名前は、表 9 に示すように、オペレーティング・システム・プラットフォームによって異なります。

表 9. コネクターの始動スクリプト

オペレーティング・システム	始動スクリプト
UNIX ベースのシステム	connector_manager_*connName*
Windows	start_*connName*.bat

コネクター始動スクリプトは、以下に示すいずれかの方法で起動することができます。

- Windows システムで**「スタート」**メニューから。

  「プログラム」>「IBM WebSphere Business Integration Adapters」>「アダプター」>「コネクター」 を選択します。デフォルトでは、プログラム名は「IBM WebSphere Business Integration Adapters」となっています。ただし、これはカスタマイズすることができます。あるいは、ご使用のコネクターへのデスクトップ・ショートカットを作成することもできます。

- コマンド行から。
  - Windows システム:

    start_*connName connName brokerName* [-c*configFile*]

– UNIX ベースのシステム:

```
connector_manager_connName -start
```

ここで、*connName* はコネクターの名前であり、*brokerName* は以下のようにご使用の統合ブローカーを表します。

– WebSphere InterChange Server の場合は、*brokerName* に ICS インスタンスの名前を指定します。

– WebSphere Message Brokers (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、または WebSphere Business Integration Message Broker) または WebSphere Application Server の場合は、*brokerName* にブローカーを示すストリングを指定します。

注: Windows システム上の WebSphere Message Broker または WebSphere Application Server の場合は、-c オプションに続いてコネクター構成ファイルの名前を指定しなければなりません。ICS の場合は、-c はオプションです。

- Adapter Monitor から (WebSphere Business Integration Adapters 製品のみ)。Adapter Monitor は System Manager 始動時に起動されます。

  このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。
- System Monitor から (WebSphere InterChange Server 製品のみ)。

  このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。
- Windows システムでは、Windows サービスとして始動するようにコネクターを構成することができます。この場合、Windows システムがブートしたとき (自動サービスの場合)、または Windows サービス・ウィンドウを通じてサービスを始動したとき (手動サービスの場合) に、コネクターが始動します。

コマンド行の始動オプションなどのコネクターの始動方法の詳細については、以下の資料のいずれかを参照してください。

- WebSphere InterChange Server については、「*システム管理ガイド*」を参照してください。
- WebSphere Message Brokers については、「*WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド*」を参照してください。
- WebSphere Application Server については、「*アダプター実装ガイド (WebSphere Application Server)*」を参照してください。

コネクターの停止

コネクターを停止する方法は、以下に示すように、コネクターが始動された方法によって異なります。

- コマンド行からコネクターを始動した場合は、コネクター始動スクリプトを用いて、以下の操作を実行します。

- Windows システムでは、始動スクリプトを起動すると、そのコネクター用の別個の「コンソール」ウィンドウが作成されます。このウィンドウで、「Q」と入力して Enter キーを押すと、コネクターが停止します。
- UNIX ベースのシステムでは、コネクターはバックグラウンドで実行されるため、別ウィンドウはありません。代わりに、次のコマンドを実行してコネクターを停止します。

  ```
  connector_manager_connName -stop
  ```

  ここで、*connName* はコネクターの名前です。

- Adapter Monitor から (WebSphere Business Integration Adapters 製品のみ)。Adapter Monitor は System Manager 始動時に起動されます。

  このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- System Monitor から (WebSphere InterChange Server 製品のみ)

  このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- Windows システムでは、Windows サービスとして始動するようにコネクターを構成することができます。この場合、Windows システムのシャットダウン時に、コネクターは停止します。

第 3 章 コネクターのビジネス・オブジェクトについて

この章では、WebSphere Business Integration Adapter for JDBC がビジネス・オブジェクトを処理する方法および、コネクターがデータを検索または変更する場合の前提事項について解説します。本章の内容は、次のとおりです。

- 『ビジネス・オブジェクトおよび属性の命名規則』
- 『ビジネス・オブジェクトの構造』
- 42 ページの『ビジネス・オブジェクト動詞の処理』
- 60 ページの『ビジネス・オブジェクトの属性プロパティー』
- 62 ページの『ビジネス・オブジェクトのアプリケーション固有の情報』

この情報は、既存のビジネス・オブジェクトを変更する場合のためのガイドとして、または新規ビジネス・オブジェクトのインプリメントに関する提案事項として役立てることができます。データベース表からの WebSphere Business Integration Adapter ビジネス・オブジェクト定義ファイルの作成を自動化するユーティリティーについては、75 ページの『第 4 章 i2ADWODA を使用したビジネス・オブジェクト定義の生成』を参照してください。

コネクターには、サポートするビジネス・オブジェクトの構造、親のビジネス・オブジェクトと子のビジネス・オブジェクトの間の関係、アプリケーション固有情報のフォーマット、およびビジネス・オブジェクトのデータベース表記について前提事項があります。そのため、コネクターで処理されるビジネス・オブジェクトを作成または変更する場合は、その変更がコネクターの従うべきルールに適合している必要があります。適合していないと、コネクターは新規の、あるいは変更されたビジネス・オブジェクトを正しく処理できません。

ビジネス・オブジェクトおよび属性の命名規則

コネクターが使用するビジネス・オブジェクトの名前に使用できる文字は、英数字および下線文字のみです。ビジネス・オブジェクトの属性名に使用できる文字も、英数字および下線文字に限られます。

ビジネス・オブジェクトの構造

多くの場合、コネクターはすべての個別ビジネス・オブジェクトが 1 つのデータベース表またはビューによって表され、オブジェクト内部のそれぞれの**単純属性** (つまり、`String` または `Integer` または `Date` などの単一値を表す属性) はその表またはビュー内の列によって表されると想定します。つまり、同一ビジネス・オブジェクト内の属性は、異なるデータベース表に保管することはできません。ただし、次のような状態はありえます。

- データベース表には、対応する個々のビジネス・オブジェクトが持つ単純属性よりも多くの列を持つものもあります (つまりデータベース内の列には、ビジネス・オブジェクト内に存在していないものもあります)。設計に際しては、ビジネス・オブジェクト処理に必要な列のみを含めるようにしてください。

- 個々のビジネス・オブジェクトの中には、対応するデータベース表が持つ列よりも多くの単純属性を持つものもあります (つまりビジネス・オブジェクト内の属性には、データベース内に存在していないものもあります)。データベース内に存在していない属性は、アプリケーション固有の情報を持たない、またはデフォルト値が設定されている、あるいはストアード・プロシージャーを指定する、のいずれかになります。
- 個々のビジネス・オブジェクトでは、複数のデータベース表をまたがるビューを表すことができます。コネクターはそのようなビジネス・オブジェクトを、アプリケーションで起動される作成、検索、更新、削除の各イベントを処理する際に使用できます。ただし、内蔵ブローカー要求を処理する際には、コネクターかそのようなビジネス・オブジェクトを使用できるのは検索要求のみです。
- 個別ビジネス・オブジェクトは、関連のないビジネス・オブジェクトのコンテナーとして使用されるラッパー・オブジェクトを表すことができます。ラッパー・オブジェクトはデータベース表やビューによって表されません。ラッパー・オブジェクトは他のオブジェクトの子として使用することはできません。

注: ビジネス・オブジェクトがストアード・プロシージャーに基づいている場合は、各単純属性に (特別な SP 属性以外は) アプリケーション固有の情報があってもなくても構いません。詳細については、52 ページの『ストアード・プロシージャー』を参照してください。

WebSphere Business Integration Adapter にはフラットなビジネス・オブジェクトと階層のあるビジネス・オブジェクトがあります。**フラットな**ビジネス・オブジェクトの属性はすべて、単純で、単一の値を表します。

階層ビジネス・オブジェクトは、子ビジネス・オブジェクト、子ビジネス・オブジェクト配列、あるいは、その両方の組み合わせを表す属性を持ちます。各子ビジネス・オブジェクトは子ビジネス・オブジェクト、子ビジネス・オブジェクト配列などを含むことが可能です。**単一カーディナリティーの関係**は、親ビジネス・オブジェクトの属性が単一の子ビジネス・オブジェクトを表す場合に現れます。この場合、属性は子ビジネス・オブジェクトと同一タイプになります。

複数カーディナリティーの関係は、親ビジネス・オブジェクトの属性が子ビジネス・オブジェクトの配列を表す場合に現れます。この場合、属性は子ビジネス・オブジェクトと同一タイプの配列になります。

注: **階層**ビジネス・オブジェクトという用語は、完全なビジネス・オブジェクトを示します。これにはあらゆるレベルでこのビジネス・オブジェクトが含む子ビジネス・オブジェクトすべてが含まれます。**個々の**ビジネス・オブジェクトという用語は、互いに包含関係にあると考えられるビジネス・オブジェクトのいずれの子ビジネス・オブジェクトからも独立した、単一のビジネス・オブジェクトを指します。**トップレベル**・ビジネス・オブジェクトという用語は、個々のビジネス・オブジェクトで、階層の最上位にあり、それ自身が親ビジネス・オブジェクトを持たないものを示します。

コネクターはビジネス・オブジェクト間の下記の関係をサポートします。

- 39 ページの『単一カーディナリティー関係』
- 39 ページの『単一カーディナリティー関係および所有権のないデータ』
- 40 ページの『複数カーディナリティー関係』

- 41 ページの『関係を子に格納する単一カーディナリティー関係』

カーディナリティーの各タイプにおいて、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のアプリケーション固有情報で記述されます。このアプリケーション固有情報の詳細は、 65 ページの表 11 の FK=[fk_object_name.]fk_attribute_name を参照してください。

注: コネクター用のビジネス・オブジェクトを作成する前に、i2 ドキュメンテーションを参照して推奨されるローダー・シーケンスを確認してください。

単一カーディナリティー関係

通常、単一カーディナリティーの子ビジネス・オブジェクトを含むビジネス・オブジェクトは、関係を表す属性を少なくとも 2 つ持っています。一方の属性は子のタイプと同一です。もう一方の属性は、単純属性で、子の基本キーを親の外部キーとして持っています。親は子が持つ基本キー属性と同数の外部キー属性を持ちます。

関係を確立する外部キーは親に保管されるため、各親は指定されたタイプの単一カーディナリティーの子は 1 つしか持つことができません。

図 2 に典型的な単一カーディナリティー関係を示します。この例では、fk1 は単純属性で、子の基本キーを含み、child[1] は子ビジネス・オブジェクトを表す属性です。

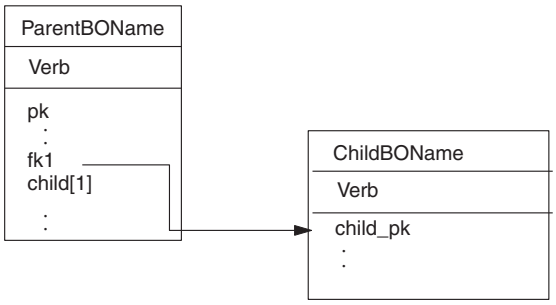


図 2. 典型的な単一カーディナリティー関係

単一カーディナリティー関係および所有権のないデータ

通常、各親ビジネス・オブジェクトは、それが含む子ビジネス・オブジェクト内のデータを所有します。例えば、各 Customer ビジネス・オブジェクトが単一の Address ビジネス・オブジェクトを含む場合、新規カスタマーが作成されたときに、新規の行がカスタマー表とアドレス表に挿入されます。新規アドレスは新規カスタマーに固有です。同様に、カスタマー表からカスタマーを削除するときに、カスタマー・アドレスがアドレス表から削除されます。

ただし、複数の階層ビジネス・オブジェクトが同一のデータを含み、そのいずれもが所有していない場合もあります。例えば、Address ビジネス・オブジェクトが単一カーディナリティーの StateProvince 参照表を表す StateProvince[1] 属性を持つとします。参照表が更新されることは大変少なく、アドレス・データとは独立して

保守されるため、アドレス・データの作成や変更は参照表中のデータに影響しません。コネクターは、既存の都道府県名を検出するか、失敗するかのいずれかです。参照表の値を追加したり、変更したりすることはありません。

複数のビジネス・オブジェクトが同一の単一カーディナリティーの子ビジネス・オブジェクトを含む場合、各親ビジネス・オブジェクトの外部キー属性の関係を NO_OWNERSHIP 指定する必要があります。ビジネス・プロセスからコネクターに階層ビジネス・オブジェクトが、作成、削除、更新のいずれかの要求と共に送信されるとき、コネクターは所有権なしに含まれている単一カーディナリティーの子を無視します。コネクターはこれらのビジネス・オブジェクトについては検索のみを実施します。コネクターがこのような単一カーディナリティーのビジネス・オブジェクトの検索に失敗すると、エラーを返して処理を停止します。

所有権なしの関係を指定する方法については、71 ページの『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属性』を参照してください。外部キー関係の指定については、67 ページの『属性の外部キーの指定』を参照してください。

非正規化データおよび所有権のないデータ

所有権なし包含には、静的参照表を簡単に使用できるようにするほかに、もう 1 つ、正規化データと非正規化データの同期をとる機能もあります。

正規化データから非正規化データへの同期化: 関係を NO_OWNERSHIP と指定すれば、正規化アプリケーションから非正規化アプリケーションへと同期をとったときに、データを作成または変更できます。例えば、正規化されたソース・アプリケーションが、A と B という 2 つの表にデータを保管するとします。さらに正規化されていない宛先アプリケーションは、すべてのデータを単一の表に保管する、つまり表 A の各エンティティーは冗長的に表 B のデータ保管すると想定します。

この例では、ソース・アプリケーションから宛先アプリケーションに表 B のデータを同期させるため、表 B のデータが変更したときには必ず、表 A イベントを起動する必要があります。さらに、表 B のデータは冗長的に表 A に保管されるので、表 A の行ごとに表 B で変更になったデータを含むビジネス・オブジェクトを送信することが必要です。

非正規化データから正規化データへの同期化: 正規化されていないソース・アプリケーションから正規化された宛先アプリケーションにデータを同期させる場合、コネクターは正規化されたアプリケーション内に含まれている所有権のないデータについて、作成、削除、および更新のいずれも行いません。

正規化アプリケーションにデータを同期させる場合、コネクターは所有権なく含まれている単一カーディナリティーの子をすべて無視します。そのような子データを作成、削除、または修正する場合は、手動でデータを処理してください。

複数カーディナリティー関係

通常、子ビジネス・オブジェクト配列を含むビジネス・オブジェクトは、関係を表す属性を 1 つのみ持っています。属性タイプは子ビジネス・オブジェクトと同一タイプの配列になります。1 つ以上の子を持つ親もあるため、関係を確立する外部キーはそれぞれの子に保管されます。

そのため、子は少なくとも 1 つ、親の基本キーを外部キーとして含む単純属性を持つことになります。子は親が持つ基本キー属性と同数の外部キー属性を持ちます。

関係を確立する外部キーは子に保管されるため、各親は子を複数持つこともまったく持たないこともできます。

図 3 に複数カーディナリティー関係を示します。この例では、parentId は単純属性で、親の基本キーを含み、child[n] は子ビジネス・オブジェクト配列を表す属性です。

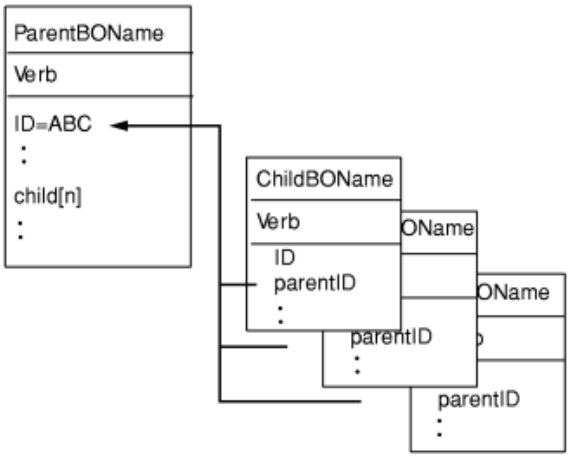


図 3. 複数カーディナリティー・ビジネス・オブジェクト関係

関係を子に格納する単一カーディナリティー関係

アプリケーションによっては、関係を親ではなく子の中に保管するように、単一の子エンティティーを保管します。つまり、子は親の基本キーと同一の値の外部キーを含みます。

図 4 に、特別なタイプの単一カーディナリティー関係を示します。

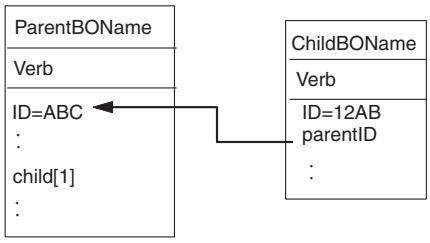


図 4. 関係を子に格納している単一のカーディナリティー関係

子データが親から独立して存在するのではなく、親を通してのみアクセスできる場合、アプリケーションはこのタイプの単一カーディナリティーの関係を使用します。このような子データは、複数の親に所有されることはなく、親とその基本キー値は、子とその外部キー値が作成されるより以前に存在することが必要です。

このようなアプリケーションに対応するため、コネクターは、単一カーディナリティーを持つ子を含む階層ビジネス・オブジェクトもサポートします。ただし、関係は親ではなく子に保管されます。

このような特殊な方法で、親ビジネス・オブジェクトの中に単一カーディナリティーの子が入るように指定するには、その子を含む属性のアプリケーション固有の情報を指定する際に、CONTAINMENT パラメーターを指定しないでください。詳細については、71 ページの『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属性』を参照してください。

ラッパー・オブジェクト

ラッパー・オブジェクトは、どのデータベース表またはビューにも対応しないトップレベル・ビジネス・オブジェクトです。ラッパー・オブジェクトは、true の値を持つトップレベル・ビジネス・オブジェクト・プロパティー WRAPPER によって示されます。ラッパー・オブジェクトは関連のない子のコンテナーとして使用されるダミーの親です。ラッパー・オブジェクトの処理中、コネクターはトップレベル・ビジネス・オブジェクトを無視し、子のみを処理します。ラッパー・オブジェクトには N のカーディナリティーを持つエンティティーまたは N-1 のカーディナリティーを持つエンティティー、あるいはその両方を含めることができます。

N のカーディナリティーを持つエンティティーは、最低でも 1 つの固有属性が基本キーとしてマークされ、最低でも 1 つの属性が外部キーとしてマークされている必要があります。この外部キーは、次に基本キーとしてラッパー・オブジェクトに追加されます。エンティティーの外部キーは、ここで追加されたラッパー・オブジェクトの基本キーを参照します。

N-1 のカーディナリティーを持つエンティティーの場合、基本キーは基本キーとしてマークされると同時に、ラッパーの基本キーを参照する外部キー (N-1 のエンティティーの基本キーと同じ) としてマークされる必要があります。

ビジネス・オブジェクト動詞の処理

このセクションでは、ビジネス・オブジェクトの動詞処理の以下の性質について解説します。

- 43 ページの『動詞の判別』では、コネクターがそれぞれのソース・ビジネス・オブジェクトごとに使用する動詞を決定する方法を説明します。
- 43 ページの『変更後イメージと差分』では、用語を定義し、コネクターが変更後イメージを扱う方法を説明します。
- 44 ページの『動詞の処理』では、ビジネス・オブジェクトを作成、検索、更新、および削除を行う際に、コネクターが実行するステップを説明します。
- 52 ページの『ストアード・プロシージャー』では、コネクターがストアード・プロシージャーを使用する方法を説明します。
- 60 ページの『トランザクション・コミットとロールバック』では、コネクターがトランザクション・ブロックを使用する方法を簡単に説明します。

動詞の判別

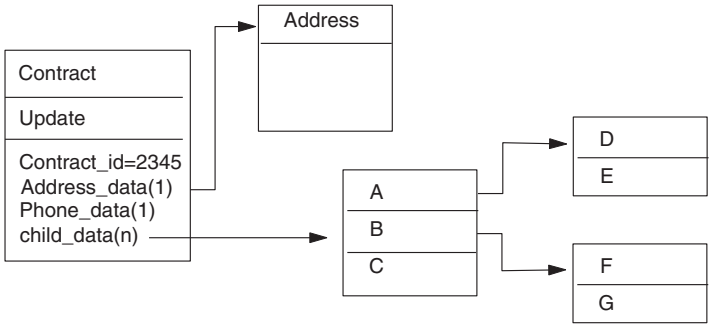
トップレベル・ビジネス・オブジェクトと、その子ビジネス・オブジェクトは、それぞれが自分自身の動詞を含むことができます。したがって、ビジネス・プロセスは、親と子のビジネス・オブジェクトに異なる動詞を持つビジネス・オブジェクトを、コネクターに渡すことができます。このとき、コネクターはトップレベルの親ビジネス・オブジェクトの動詞を使用して、ビジネス・オブジェクト全体の処理方法を決定します。詳細については、44 ページの『動詞の処理』を参照してください。

変更後イメージと差分

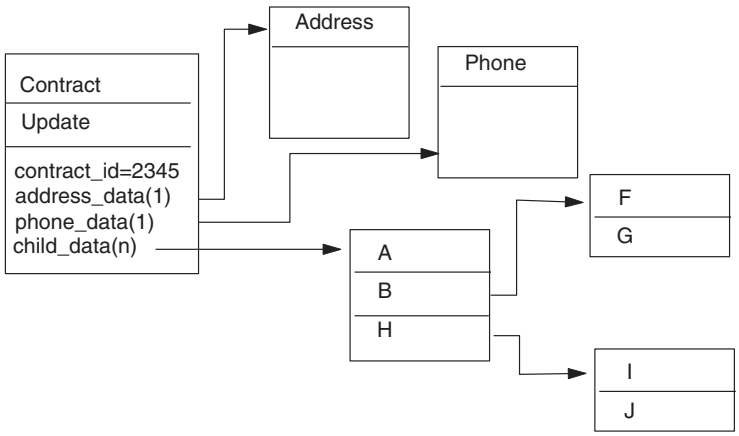
変更後イメージとは、すべての変更が行われた後のビジネス・オブジェクトの状態のことです。デルタとは、キー値と変更の対象となるデータのみを含む、更新操作で使用するビジネス・オブジェクトのことです。コネクターは、変更後イメージのみをサポートするため、統合ブローカーから更新のためのビジネス・オブジェクトを受け取ると、そのビジネス・オブジェクトは更新後のデータのあるべき状態を表すものと想定します。

そのため、ビジネス・プロセスが、動詞 Update を持つビジネス・オブジェクトをコネクターに送信すると、コネクターはデータベース中の該当ビジネス・オブジェクトの現在値を変更して、ソース・ビジネス・オブジェクトと厳密に一致させます。こうするには、コネクターは単純属性値を変更し、子ビジネス・オブジェクトを追加、または除去します。

例えば、データベース内の Contract 2345 の現在の状態が次のとおりである場合を想定します。



さらに、ビジネス・プロセスが、次に示すビジネス・オブジェクトをコネクターに受け渡すと想定します。



更新を処理するために、コネクターはデータベースに次の変更を設定します。

- トップレベルの Contract ビジネス・オブジェクト、および Address ビジネス・オブジェクト中の単純属性を更新します。
- Phone ビジネス・オブジェクトを作成します。
- 子ビジネス・オブジェクト A、B、F、および G の単純属性を更新します。
- 子ビジネス・オブジェクト C、D、および E を削除します。
- 子ビジネス・オブジェクト H、I、および J を作成します。

コネクターは、ビジネス・プロセスから受信する各ビジネス・オブジェクトが変更後イメージを表していると想定しているので、コネクターに送信する更新用ビジネス・オブジェクトがそれぞれ、既存の有効な子ビジネス・オブジェクトを確実に含むことが大切です。たとえ子ビジネス・オブジェクトの単純属性が 1 つも変わらなかったとしても、子ビジネス・オブジェクトはソース・ビジネス・オブジェクトにインクルードする必要があります。

更新操作中に、コネクターが欠落した子ビジネス・オブジェクトを削除しないように予防する方法があります。子もしくは子の配列を表す属性のアプリケーション固有の情報を使用して、ソース・ビジネス・オブジェクトにインクルードされなかった子ビジネス・オブジェクトを保持するよう、コネクターに指示します。このためには、KEEP_RELATIONSHIP を true に設定してください。詳細については、67 ページの『属性の外部キーの指定』を参照してください。

動詞の処理

このセクションでは、コネクターがビジネス・プロセスから受け取るビジネス・オブジェクトを作成、検索、更新および削除を行う際の、ステップの概略を説明します。コネクターは、階層ビジネス・オブジェクトを再帰的に処理します。つまり、すべてのビジネス・オブジェクトを処理するまで、それぞれの子ビジネス・オブジェクトに同じステップを実行します。

注: ラッパーであるトップレベル・ビジネス・オブジェクトは、 create、 retrieve、 update、および delete 動詞をサポートします。ラッパー・オブジェクトの処理

で唯一異なる点は、ラッパー・オブジェクトが処理されずにラッパー・オブジェクトに含まれるオブジェクトのみが処理されるということです。

ビジネス・オブジェクトの比較

以下に概説する処理のさまざまな時点で、コネクターは、2 つのビジネス・オブジェクトを比較し、同一かどうか確認します。例えば更新操作中に、コネクターは特定のビジネス・オブジェクトがあるビジネス・オブジェクト配列中に存在するかどうかを判断します。チェックするため、コネクターはビジネス・オブジェクトを配列中の各ビジネス・オブジェクトと比較します。2 つのビジネス・オブジェクトが同一であるということは、次の 2 つの条件が満たされるということです。

- 比較するビジネス・オブジェクトのタイプが同一であること。例えば、Customer ビジネス・オブジェクトは、たとえ属性すべてが厳密に等しくても、Contact ビジネス・オブジェクトと同一になることは決してありません。
- 2 つのビジネス・オブジェクト内の対応するキー属性が同一の値を含むこと。両方のビジネス・オブジェクトにおいて、キー属性が `CxIgnore` であれば、コネクターは同一であると判断します。ただし、一方のビジネス・オブジェクトのキー属性が `CxIgnore` で、もう一方がそれとは異なる場合、それらのビジネス・オブジェクトは同一ではありません。

Create 操作

ビジネス・オブジェクトを作成する際には、コネクターは正常終了であれば (ビジネス・オブジェクトに変更が加えられたかどうかに関係なく) `VALCHANGE` を、操作が失敗すれば `FAIL` の状況を戻します。

コネクターは階層ビジネス・オブジェクトを作成するときには次のステップを実行します。

1. 所有権付きで含まれている単一カーディナリティーの子ビジネス・オブジェクトそれぞれを、再帰的にデータベースに挿入します。つまり、コネクターは子、および子とその子が含むすべての子ビジネス・オブジェクトを作成します。

   そのビジネス・オブジェクト定義で、属性が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ように指定されており、さらにその属性が空であれば、コネクターはその属性を無視します。ただし、そのビジネス・オブジェクト定義で、その属性が子を表すために必須であるが存在しない場合、コネクターはエラーを戻し、処理を停止します。
2. 所有権なく含まれている単一カーディナリティーの子ビジネス・オブジェクトをそれぞれ、次のように処理します。
   a. ビジネス・プロセスによって渡されたキー値を使用して、データベース中で再帰的に子の検索を試みます。
   b. 検索が正常に終了せず、データベース中にその子が現在存在しないことが示された場合は、コネクターはエラーを戻し、処理を停止します。検索が正常終了した場合、コネクターは再帰的に子ビジネス・オブジェクトを更新します。

   注: 子ビジネス・オブジェクトが既にアプリケーション・データベースに存在するときに、この方法が正確に動作するためには、開発者が作成操作において、子ビジネス・オブジェクトの基本キー属性が正しく相互参照されるよう

に注意する必要があります。子ビジネス・オブジェクトがアプリケーション・データベースにまだ存在しない場合は、基本キー属性を CxBlank にセットしてください。

3. 次のようにデータベースにトップレベル・ビジネス・オブジェクトを挿入します。
   a. 単一カーディナリティーで表される対応する子ビジネス・オブジェクトの基本キー値に対して、それぞれ外部キー値をセットします。子ビジネス・オブジェクト内の値は、子の作成時にデータベース・シーケンスやデータベース・カウンター、あるいはデータベース自身によってセットされることもあるので、コネクターがデータベースに親を挿入する前に、このステップで、親の外部キー値が正しいことを保証します。
   b. 属性ごとに新規の固有 ID を生成します。これはデータベースによって自動的にセットされます。データベース・シーケンス名、またはデータベース・カウンター名は属性のアプリケーション固有情報に保管されます。属性が、関連するデータベース・シーケンスまたはデータベース・カウンターを持つ場合、コネクターによって生成される値はビジネス・プロセスから渡されたいずれの値も上書きします。データベース・シーケンスまたはカウンターの指定については、 65 ページの『単純属性のアプリケーション固有情報』の UID=uid_name[=UseIfMissing] を参照してください。
   c. 属性のアプリケーション固有情報である CA (CopyAttribute) パラメーターで指定されたように、属性の値を別の属性の値にコピーします。CA パラメーターの使用については、 65 ページの『単純属性のアプリケーション固有情報』の CA=set_attr_name を参照してください。
   d. データベースにトップレベル・ビジネス・オブジェクトを挿入します。

      注: ラッパーであるトップレベル・ビジネス・オブジェクトは、データベース内に挿入されません。
4. 親/子関係を子に保管する単一カーディナリティーの子ビジネス・オブジェクトをそれぞれ次のように処理します。
   a. 子の外部キー値を、対応する親の基本キー属性中の値を参照するようにセットします。親の基本キー値は親の作成中に生成されることもあるため、コネクターがデータベースに子を挿入する前に、この処理でそれぞれの子の外部キー値が正しいことを保証します。
   b. 子をデータベースに挿入します。
5. 複数カーディナリティーの子ビジネス・オブジェクト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、対応する親の基本キー属性中の値を参照するようにセットします。親の基本キー値は親の作成中に生成されることもあるため、コネクターがデータベースに子を挿入する前に、この処理でそれぞれの子の外部キー値が正しいことを保証します。
   b. 複数カーディナリティーの子ビジネス・オブジェクトをデータベースに挿入します。

Retrieve 操作

コネクターは階層ビジネス・オブジェクトを検索するときには次のステップを実行します。

1. ビジネス・プロセスから受け取った子ビジネス・オブジェクトすべてを、トップレベル・ビジネス・オブジェクトから除去します。
2. データベース中のトップレベル・ビジネス・オブジェクトを検索します。
   - 検索で 1 行戻ると、コネクターは処理を継続します。
   - 検索で戻る行がなく、トップレベル・ビジネス・オブジェクトがデータベース中にないことが示された場合、コネクターは BO_DOES_NOT_EXIST を戻します。
   - 検索で複数行が戻された場合、コネクターは FAIL を戻します。

     注: ビジネス・オブジェクトは、プレースホルダー属性のような、いずれのデータベース列にも対応しない属性を持つこともできます。検索の間、コネクターはトップレベル・ビジネス・オブジェクトのそのような属性を変更しません。ビジネス・プロセスから受け取った値のままです。子ビジネス・オブジェクトでは、検索の間に、コネクターがそのような属性をデフォルト値にセットします。

     注: ラッパーであるトップレベル・ビジネス・オブジェクトには、ラッパー・オブジェクトの直下にあるオブジェクトのすべての属性値が含まれている必要があります。この値はキーおよびプレースホルダー属性などのオブジェクトの検索に必要となります。ラッパー・オブジェクトにはすべてのキーおよびプレースホルダー属性が取り込まれる必要があります。ラッパーの 1 レベル下のオブジェクトで外部キーとして使用されるラッパー・オブジェクトの単純属性は、ラッパー・オブジェクトのキーとしてマークされる必要があります。
3. 複数カーディナリティーの子ビジネス・オブジェクトすべてを再帰的に検索します。

   注: コネクターはビジネス・オブジェクトの配列を取り込むときに、固有であることを強制しません。固有性を保証するのはデータベースの責任です。データベースが重複した子ビジネス・オブジェクトを戻した場合、コネクターはビジネス・プロセスに重複した子を戻します。
4. 子ビジネス・オブジェクトが所有権付きで含まれているか、所有権なしで含まれているかにかかわらず、単一カーディナリティーの子を再帰的に検索します。

   注: 単一カーディナリティーの子はすべて、ビジネス・オブジェクト内のオカレンスに基づいて、親ビジネス・オブジェクトが処理される前に、処理されます。子オブジェクトの所有権や非所有権は処理シーケンスを決定しませんが、処理タイプは決定します。

RetrieveByContent 操作

コネクターは、属性に基づいた検索をトップレベル・ビジネス・オブジェクトでのみ実行するため、RetrieveByContent 動詞はトップレベル・ビジネス・オブジェクトにのみ適用できます。

トップレベル・ビジネス・オブジェクトが RetrieveByContent 動詞を使用する場合、非ヌルの属性すべて (非キー属性を含む) が検索基準として使用されます。

複数行が戻された場合、コネクターは最初の行を結果行として使用し、MULTIPLE_HITS を戻します。

注: RetrieveByContent 動詞はラッパーであるトップレベル・ビジネス・オブジェクトには適用されません。

Update 操作

ビジネス・オブジェクトを更新する際には、コネクターは正常終了であれば (ビジネス・オブジェクトに変更が加えられたかどうかに関係なく) VALCHANGE を、操作が失敗すれば FAIL の状況を戻します。Oracle データベースで作業しているときは、コネクターはデータ保全性を確実にするため、検索中、データをロックします。

コネクターは階層ビジネス・オブジェクトを更新するときには次のステップを実行します。

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のデータの現在の状態を正確に表現します。
   - 検索が失敗し、トップレベル・ビジネス・オブジェクトがデータベース中にないことが示された場合、コネクターは BO_DOES_NOT_EXIST を戻し、更新は失敗します。

     注: ラッパーであるトップレベル・ビジネス・オブジェクトはデータベース内に存在する必要はありません。ただし、ラッパー・オブジェクトの直下にあるオブジェクトのすべての属性値が含まれている必要があります。この値はキーおよびプレースホルダー属性などのオブジェクトの検索に必要となります。ラッパー・オブジェクトにはすべてのキーおよびプレースホルダー属性が取り込まれる必要があります。ラッパーの 1 レベル下のオブジェクトで外部キーとして使用されるラッパー・オブジェクトの単純属性は、ラッパー・オブジェクトのキーとしてマークされる必要があります。
   - 検索が正常終了すると、コネク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、データベース中の変更が必要な子ビジネス・オブジェクトを決定します。ただし、ソース・ビジネス・オブジェクトと検索されたビジネス・オブジェクトの単純属性の比較は行いません。コネクターは、非キー単純属性すべての値を更新します。

     トップレベル・ビジネス・オブジェクト内の単純属性すべてがキーを表す場合、コネクターはトップレベル・ビジネス・オブジェクトの更新照会を生成できません。この場合、コネクターは警告をログに記録して、ステップ 2 に進みます。
2. トップレベル・ビジネス・オブジェクトの単一カーディナリティーの子をすべて再帰的に更新します。

   ビジネス・オブジェクト定義で、属性が子ビジネス・オブジェクトを表すことが必須になっている場合、子はソース・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する必要があります。存在しない場合には、更新が失敗し、コネクターはエラーを戻します。

コネクターは以下のいずれかの方法で、所有権付きで含まれている単一カーディナリティーの子を扱います。

- 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、コネクターは、データベース内の既存の子を更新するのではなく、既存の子を削除して新規の子を作成します。
- 子がソース・ビジネス・オブジェクトにはあるが、検索されたビジネス・オブジェクトにはない場合、コネクターはデータベース内で、子を再帰的に作成します。
- 子が検索されたビジネス・オブジェクトにはあるが、ソース・ビジネス・オブジェクトにはない場合、コネクターはデータベース内で、子を再帰的に削除します。削除タイプが物理的であるか、論理的であるかは 22 ページの『ChildUpdatePhyDelete』プロパティーの値に依存します。

所有権なしに含まれている単一カーディナリティーの子について、コネクターはソース・ビジネス・オブジェクトに存在する子すべてをデータベースから検索しようと試みます。子が正常に検索できた場合、コネクターは子ビジネス・オブジェクトを取り込みますが、更新はしません。所有権なしに含まれる単一カーディナリティーの子は、コネクターによって変更されることはありません。

3. 親に関係を保管する単一カーディナリティーの子ビジネス・オブジェクトについては、親の外部キー値をそれぞれ、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セットしてください。このステップは、前のステップで単一カーディナリティーの子がデータベースに追加され、新規の固有 ID が生成されている可能性があるために必要です。
4. 検索されたビジネス・オブジェクトの単純属性すべてを更新します。ただし、ソース・ビジネス・オブジェクトの対応する属性の値が `CxIgnore` である場合を除きます。

   更新されるビジネス・オブジェクトは固有である必要があるため、コネクターは結果として処理される行が 1 行のみであるかを検査します。複数行が戻されるとエラーを戻します。
5. 親/子関係を子に保管する (複数カーディナリティー、単一カーディナリティーとも) 子それぞれの外部キー値すべてを、対応する親ビジネス・オブジェクトの基本キー値にセットします。(統合ブローカーが ICS の場合、この値は通常データ・マッピングで相互参照されます。) このステップは、コネクターが子を更新する前に、子に関係を保管する、新規の子の外部キー値が正しいことを保証するために大切です。
6. 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それぞれを次のように処理します。
   - 子がソース・ビジネス・オブジェクト配列と検索されたビジネス・オブジェクト配列の両方にあれば、コネクターはデータベース内で、子を再帰的に更新します。
   - 子がソース・ビジネス・オブジェクト配列にはあるが、検索されたビジネス・オブジェクト配列にはない場合、コネクターはデータベース内で、子を再帰的に作成します。
   - 子が検索されたビジネス・オブジェクト配列にはあるが、ソース・ビジネス・オブジェクト配列にはない場合、コネクターはデータベース内で、子を再帰的

に削除します。ただし、親の、子を表す属性のアプリケーション固有情報で KEEP_RELATIONSHIP が true にセットされている場合を除きます。この場合、コネクターはデータベースから子を削除しません。詳細については、 67 ページの『属性の外部キーの指定』を参照してください。削除タイプが物理的であるか、論理的であるかは 22 ページの『ChildUpdatePhyDelete』プロパティーの値に依存します。

注: ビジネス・プロセスは、複数カーディナリティーと共にソース・ビジネス・オブジェクト内に含まれるビジネス・オブジェクトが固有であること (つまり、配列が同一のビジネス・オブジェクトに対して、複数コピーを持たないこと) を保証する必要があります。コネクターがソース配列内で重複しているビジネス・オブジェクトを受け取った場合、そのビジネス・オブジェクトを 2 回処理し、予測不能な結果を招くことがあります。

DeltaUpdate 操作

DeltaUpdate 動詞の処理は、Update 動詞の処理と以下の点で異なります。

1. Update 動詞が処理される際には更新の前に検索が実行されますが、DeltaUpdate が処理される際には実行されません。
2. 着信ビジネス・オブジェクトとデータベース内のビジネス・オブジェクトの比較が行われません。
3. どの子も、各子オブジェクトに設定されている動詞に基づいて処理されます。子に動詞が設定されていない場合、コネクターはエラーを戻します。

コネクターは、ビジネス・オブジェクトの差分更新時に、2 つの状況のいずれかを戻します。戻される状況は、操作が正常に終了した場合 (操作によってビジネス・オブジェクトの変更が生じたかどうかを問いません) は VALCHANGE、操作が失敗した場合は FAIL です。

コネクターでは、階層ビジネス・オブジェクトの差分更新時に、以下のステップを実行します。

1. 親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処理します。ビジネス・オブジェクト仕様で IsRequired とマークされている子は、インバウンド・オブジェクトに必ず存在していなければなりません。存在しない場合、差分更新は失敗し、コネクターはエラーを戻します。
2. 親に含まれる外部キー値のうち、単一カーディナリティーの子の属性を参照するものすべてを、それぞれ対応する子の値に設定します。この処理が必要なのは、これ以前のステップで単一カーディナリティーの子がデータベースに追加され、新しいシーケンス値が生成されている可能性があるためです。
3. 現在処理中のオブジェクトを、SQL UPDATE ステートメントまたはストアード・プロシージャーを使用して更新します。個々のビジネス・オブジェクトのすべての単純属性が更新されます。ただし、インバウンド・ビジネス・オブジェクトで IsIgnore に設定されている属性を除きます。コネクターでは、インバウンド・オブジェクトと現在のオブジェクトを属性レベルで比較して、UPDATE ステートメントに追加する必要がある属性を決定することはありません。つまり、属性はすべて更新されます。更新されるオブジェクトは一意である必要があるため、コネクターは、結果として 1 行のみが処理されることを確認します。複数の行が処理される場合、エラーが戻されます。

4. 現在のオブジェクトの子のうち、カーディナリティーが N のものすべてで、親の属性を参照する外部キー値のすべてを、それぞれ対応する親の値に設定します。通常、これらの値はデータ・マッピング時に相互参照されます。ただし、これはカーディナリティーが N のコンテナーに含まれる新しい子には該当しないことがあります。ここでの処理により、カーディナリティーが N の子のすべてで、これらの子が更新される前に外部キー値を確実に正しい値にすることができます。
5. 現在のオブジェクトの、カーディナリティーが N のコンテナーをすべて更新します。

子オブジェクトが処理されるときには、それぞれの子の動詞が取得されて適切な操作が実行されます。DeltaUpdate が処理される際に許可される子の動詞は、Create、Delete、および DeltaUpdate です。

- 子で Create 動詞が検出された場合、その子が所有関係にある子であれば、データベースにその子が作成されます。所有関係にない子に関しては、検索により、データベースに存在するかどうかが確認されます。
- 子で Delete 動詞が検出された場合、その子は削除されます。
- 子で DeltaUpdate 動詞が検出された場合、データベースでその子が更新されます。

Delete 操作

ビジネス・オブジェクトを削除する際には、コネクターは正常終了であれば SUCCESS を、操作が失敗すれば FAIL の状況を戻します。アダプターは、まず親ビジネス・オブジェクトを検索します。次に、親から見て所有関係にある単一カーディナリティーの子のすべてを再帰的に削除してから、親ビジネス・オブジェクト自体を削除します。最後に、カーディナリティーが N の子をすべて削除します。所有関係にない単一カーディナリティーの子は削除されません。操作対象のビジネス・オブジェクトが存在しない場合、コネクターは FAIL を戻します。

コネクターはオブジェクトのアプリケーション固有情報にある状況列名 (SCN) 値によって、論理的な削除も物理的な削除もサポートします。SCN 値が定義済みであれば、コネクターは論理的な削除を実行します。SCN 値が定義されていなければ、コネクターは物理的な削除を実行します。

物理削除: コネクターは階層ビジネス・オブジェクトを物理削除するときには次のステップを実行します。

1. 所有権付きで含まれている単一カーディナリティーの子ビジネス・オブジェクトすべてを再帰的に削除します。
2. トップレベル・ビジネス・オブジェクトを削除します。
3. 複数カーディナリティーの子ビジネス・オブジェクトすべてを再帰的に削除します。

注: ラッパーであるトップレベル・ビジネス・オブジェクトは対応するデータベース表を持たないため、データベースから削除されません。ラッパーの単純属性値はすべて無視されます。

論理削除: ビジネス・オブジェクトを論理的に削除するときは、コネクターは以下のステップを実行します。

1. UPDATE を発行して、ビジネス・オブジェクトの状況属性をビジネス・オブジェクトのアプリケーション固有情報によって指定された値にセットします。コネクターは、データベース行が結果として、1 行のみ更新されるように保証し、そうならない場合はエラーを戻します。
2. 所有権付きで含まれている単一カーディナリティーの子すべてと、複数カーディナリティーの子すべてを再帰的に論理削除します。コネクターは所有権なしで含まれている単一カーディナリティーの子は削除しません。

SQL ステートメント

コネクターは、選択、更新、検索、削除の各操作に、単純 SQL ステートメントを使用できます。SQL ステートメントの列名は属性の AppSpecificInfo プロパティーから派生します。照会は、ビューに POST される場合を除き、それぞれ 1 つの表のみを対象としています。

ストアード・プロシージャー

ストアード・プロシージャーとは、SQL ステートメントのグループで、論理単位を形成して特定のタスクを実行します。ストアード・プロシージャーは、命令あるいは照会のセットをコネクター用にカプセル化し、データベース・サーバー内のオブジェクトに対して実行します。

コネクターは以下の場合に、ストアード・プロシージャーを呼び出します。

- ビジネス・オブジェクトを処理する前に事前操作処理を実行する。
- ビジネス・オブジェクトを処理した後に操作事後処理を実行する。
- 単純な INSERT、RETRIEVE、UPDATE、または DELETE ステートメントを使用する代わりに、ビジネス・オブジェクトに対して命令セットを実行する。

階層ビジネス・オブジェクトを処理するとき、コネクターはストアード・プロシージャーを使用して、トップレベル・ビジネス・オブジェクトや、その子ビジネス・オブジェクトのいずれを処理することもできます。ただし、ビジネス・オブジェクトやビジネス・オブジェクト配列はそれぞれ自分自身のストアード・プロシージャーを持っている必要があります。

ストアード・プロシージャーの指定

このセクションでは、コネクターがビジネス・オブジェクトにストアード・プロシージャーを使用するようにするために、実行する必要があるステップについて解説します。以下のセクションがあります。

- 『ビジネス・オブジェクトへの属性の追加』
- 53 ページの『ストアード・プロシージャーの構文』
- 54 ページの『ストアード・プロシージャーの例』
- 54 ページの『ストアード・プロシージャーの指定』

ビジネス・オブジェクトへの属性の追加: コネクターが処理するストアード・プロシージャーのタイプごとに、ビジネス・オブジェクトに特別な種類の属性を追加する必要があります。これらの属性はストアード・プロシージャーのタイプと、それを定義するアプリケーション固有情報のみを表します。これらの属性は標準的な単純属性に、使用可能なアプリケーション固有情報パラメーターを使用しません。

使用されるストアード・プロシージャーのタイプに応じて属性に名前を付けてください。例えば、コネクターが AfterUpdate ストアード・プロシージャーと、BeforeRetrieve ストアード・プロシージャーを使用できるようにするには、AfterUpdateSP 属性と、BeforeRetrieveSP 属性を追加してください。

コネクターは以下のビジネス・オブジェクト属性名を認識します。

```
BeforeCreateSP
AfterCreateSP
CreateSP
BeforeUpdateSP
AfterUpdateSP
UpdateSP
BeforeDeleteSP
AfterDeleteSP
DeleteSP
BeforeRetrieveSP
AfterRetrieveSP
RetrieveSP
BeforeRetrieveByContentSP
AfterRetrieveByContentSP
RetrieveByContentSP
BeforeRetrieveUpdateSP
AfterRetrieveUpdateSP
RetrieveUpdateSP
BeforeDeltaUpdateSP
AfterDeltaUpdateSP
DeltaUpdateSP
```

注: コネクターに実行させるストアード・プロシージャーについてのみ、属性を作成してください。ビジネス・オブジェクトがコネクターに送信される前に、アプリケーション固有情報、またはマッピング (ICS が統合ブローカーとして使用されている場合のみ) を使用して、これらの属性の値を指定します。これらの値に変更が加えられた場合に、それ以後のビジネス・オブジェクトに対するストアード・プロシージャーの呼び出しのためにコネクターにその変更を認識させるには、コネクターを再始動する必要があります。

ストアード・プロシージャーの構文: ストアード・プロシージャーを指定するための構文は、以下のとおりです。

```
SPN=StoredProcedureName;RS=true|false[;IP=Attribute_Name1[:Attribute_Name2[:...]]]
[;OP=Attribute_Name1|RS[:Attribute_Name2|RS[:...]]]
[;IO=Attribute_Name1[:Attribute_Name2[:...]]]
```

ここで、以下のように説明されます。

StoredProcedureName
: ストアード・プロシージャーの名前。

RS
: ストアード・プロシージャーが結果セットを戻す場合は、true、戻さない場合は、false。デフォルトでは、false。この値が true である場合、属性のアプリケーション固有情報に含まれる ColumnName プロパティーは、結果セットの該当するカラムを指します。RS が出力パラメーター・リストの一部である場合は、その特定のパラメーターが結果セットを戻します。1 つの結果セット OUT パラメーターのみがサポートされます。複数の結果セットが OUT パラメーターとして戻された場合は、最初の結果セットのみが戻され、その他の結果セットはすべて無

視されます。現在、この機能は Oracle 8i 以上、および Oracle JDBC ドライバーを使用するストアード・プロシージャーについてのみサポートされます。データベース内のストアード・プロシージャーの場合、対応するパラメーターは REFCURSOR タイプを戻します。

IP　入力パラメーター。コネクターがストアード・プロシージャーの処理時に入力値として使用する値が格納されているビジネス・オブジェクト属性のリスト。

OP　出力パラメーター。コネクターが、ストアード・プロシージャーを実行後に、値を戻す対象のビジネス・オブジェクト属性のリスト。結果セットの記述は RS を参照してください。

IO　入出力パラメーター。コネクターが入力値として使用し、コネクターがストアード・プロシージャーの実行後に戻り値を格納するビジネス・オブジェクト属性のリスト。

注: StoredProcedureName、RS、およびパラメーターの順序は重要です。パラメーターの中での順序は異なっても構いません。つまり、ストアード・プロシージャーのパラメーターがタイプ別にまとめて並べられていても、タイプによる区別なく並べられていても、コネクターの動作に違いは生じません。

複数の同じタイプのパラメーターがまとめて並べられている場合は、その値をコロンで区切ります。パラメーター名をそれぞれの値の前に繰り返す必要はありません。タイプの異なるパラメーターの間は、セミコロンで区切ります。パラメーターの値を指定するときには、等号 (=) の前後どちらにも、空白を入れません。

ストアード・プロシージャーの例: 以下の例では、CustomerInsert および VendorInsert というストアード・プロシージャーを示します。これらのストアード・プロシージャーは、2 つの入力属性から値を取得して、4 つの出力属性に値を戻します。これらの例では、ストアード・プロシージャーの構造が異なっています。

- 同じタイプのパラメーターがまとめて並べられているものは、次のとおりです (IP、IP、OP、OP、OP、OP、IO)。

```
SPN=CustomerInsert;RS=false;IP=LastName:FirstName;OP=CustomerName:
CustomerID:ErrorStatus:ErrorMessage;IO=VendorID
```

- 同じタイプのパラメーターがまとめて並べられていないものは、次のとおりです (IP、OP、OP、OP、IP、IO、OP)。

```
SPN=VendorInsert;RS=false;IP=LastName;OP=CustomerName:CustomerID:
ErrorStatus;IP=FirstName;IO=VendorID;OP=ErrorMessage
```

コネクターは JDBC ドライバーがサポートする単純データ型のみをサポートします。

ストアード・プロシージャーの指定: ストアード・プロシージャー名とパラメーター値を指定するには、2 つの方法があります。

- 属性の AppSpecificInfo プロパティー

  ストアード・プロシージャーを指定するテキストの長さが 4 KB 文字以下であれば、属性の AppSpecificInfo プロパティーに値を指定できます。このプロパティーを使用すると、コネクターがビジネス・オブジェクトをポーリングしたか (つまり、ビジネス・オブジェクトがアプリケーション・イベントを表しているか)、あるいはビジネス・オブジェクトをビジネス・オブジェクト要求として受け取ったかのいずれでも関係なく、ストアード・プロシージャーを指定できます。

  次の例では、アプリケーション固有情報を使用したストアード・プロシージャーの指定を示します。このケースでは、MaxLength プロパティーに指定した値は、ストアード・プロシージャーに影響しません。

```
   [Attribute]
   Name = BeforeCreateSP
   Type = String
   MaxLength = 15
   IsKey = false
   IsRequired = false
   AppSpecificInfo =SPN=ContactInsert;IP=LastName:FirstName;OP=CustomerName:
CustomerID:ErrorStatus:ErrorMessage

   [End]
```

- 属性値 (ICS が統合ブローカーとして使用されている場合のみ)

  ストアード・プロシージャーを指定するテキストの長さが 4 KB 文字より多い場合は、ストアード・プロシージャーの指定にマッピングを使用する必要があります。ストアード・プロシージャーの指定にマッピングが使用できるのは、ビジネス・オブジェクトがビジネス・オブジェクト要求を表す場合のみです。つまり、コネクターがイベントをポーリングしている場合は、ストアード・プロシージャーの指定に属性の値を使用できません。

  ストアード・プロシージャーのテキストが 4 KB 文字よりも長くて、指定にマッピングを使用する場合は必ず、MaxLength プロパティーの値をテキスト全体に合わせて拡張してください。

注: 作成、更新、または削除操作を処理するストアード・プロシージャーが、子ビジネス・オブジェクトの配列が含まれる階層ビジネス・オブジェクトに対して実行されると、コネクターは各子ビジネス・オブジェクトを個別に処理します。例えば、コネクターは、BeforeCreate ストアード・プロシージャーを実行する場合、子ビジネス・オブジェクトの配列をまとめて処理せずに、その配列に含まれるメンバーをそれぞれ処理します。コネクターが BeforeRetrieve ストアード・プロシージャーを処理するときには、単一のビジネス・オブジェクトに対して作動します。AfterRetrieve ストアード・プロシージャーを処理する場合には、検索によって戻されたビジネス・オブジェクトのすべてを操作します。

ストアード・プロシージャーまたは単純な SQL ステートメントを使用したビジネス・オブジェクトの処理

以下のセクションでは、コネクターがストアード・プロシージャーを処理する方法について説明します。

- 56 ページの『ビジネス・オブジェクトの Create 操作』

- 『ビジネス・オブジェクトの Update 操作』
- 57 ページの『ビジネス・オブジェクトの Delete 操作』
- 57 ページの『ビジネス・オブジェクトの Retrieve 操作』
- 58 ページの『ビジネス・オブジェクトの RetrieveByContent 操作』
- 59 ページの『ビジネス・オブジェクトの Update 用 Retrieve 操作』

ビジネス・オブジェクトの Create 操作: Create ストアード・プロシージャーは、通常、コネクターが最上位ビジネス・オブジェクトの単純属性を設定するために使用する値を戻します。コネクターは、作成ストアード・プロシージャー (BeforeCreate、Create、AfterCreate) を処理するとき、以下のステップを実行します。

1. ビジネス・オブジェクトが BeforeCreateSP 属性を含むかどうかをチェックします。持っていれば、BeforeCreate ストアード・プロシージャーを呼び出します。
2. ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。
3. 単一カーディナリティーの子ビジネス・オブジェクトを作成します。
4. 最上位ビジネス・オブジェクトの外部キー値のそれぞれを、単一カーディナリティーの子オブジェクトのそれぞれの基本キー値に設定します。
5. ビジネス・オブジェクトが CreateSP 属性を持っているか検査します。持っていれば、Create ストアード・プロシージャーを呼び出して、トップレベル・ビジネス・オブジェクトを作成します。持っていなければ、INSERT ステートメントをビルドし、実行して、トップレベル・ビジネス・オブジェクトを作成します。
6. Create ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。
7. 複数カーディナリティーの子それぞれの外部キー値を、その親の基本キー属性値にセットします。
8. 複数カーディナリティーの子ビジネス・オブジェクトを作成します。
9. ビジネス・オブジェクトが AfterCreateSP 属性を持っているか検査します。含まれている場合、AfterCreate ストアード・プロシージャーを呼び出します。
10. ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。

コネクターはステップ 10 で戻された値を使用して、ステップ 3 またはステップ 5 で作成したビジネス・オブジェクトの値を変更できます。

ビジネス・オブジェクトの Update 操作: Update ストアード・プロシージャーは、通常、コネクターが最上位ビジネス・オブジェクトの単純属性を設定するために使用する値を戻します。コネクターは、更新ストアード・プロシージャー (BeforeUpdate、Update、AfterUpdate) を処理するとき、以下のステップを実行します。

1. ビジネス・オブジェクトが BeforeUpdateSP 属性を持っているか検査します。持っていれば、BeforeUpdate ストアード・プロシージャーを呼び出します。
2. BeforeUpdate ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。

3. 単一カーディナリティーの子ビジネス・オブジェクトを更新します。
4. 最上位ビジネス・オブジェクトの外部キー値のそれぞれを、単一カーディナリティーの子オブジェクトのそれぞれの基本キー値に設定します。
5. ビジネス・オブジェクトが UpdateSP 属性を持っているか検査します。含まれている場合、Update ストアード・プロシージャーを呼び出して、最上位ビジネス・オブジェクトを更新します。含まれていない場合は、UPDATE ステートメントを作成して実行することにより、最上位ビジネス・オブジェクトを更新します。
6. Update ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。
7. 複数カーディナリティーの子の外部キー値を、対応する親の基本キー属性中の値を参照するようにセットします。
8. 複数カーディナリティーの子ビジネス・オブジェクトを更新します。
9. ビジネス・オブジェクトが AfterUpdateSP 属性を持っているか検査します。持っていれば、AfterUpdate ストアード・プロシージャーを呼び出します。
10. ストアード・プロシージャーが出力パラメーターを通じて値を戻した場合、その値をビジネス・オブジェクトの単純属性にセットします。

ビジネス・オブジェクトの Delete 操作: 削除ストアード・プロシージャーはコネクターに値を戻しません。コネクターは、削除ストアード・プロシージャー (BeforeDelete、Delete、AfterDelete) を処理するとき、以下のステップを実行します。

1. ビジネス・オブジェクトが BeforeDeleteSP 属性を持っているか検査します。持っていれば、BeforeDelete ストアード・プロシージャーを呼び出します。
2. 単一カーディナリティーの子ビジネス・オブジェクトを削除します。
3. 複数カーディナリティーの子ビジネス・オブジェクトを削除します。
4. ビジネス・オブジェクトが DeleteSP 属性を持っているか検査します。持っていれば、Delete ストアード・プロシージャーを呼び出して、トップレベル・ビジネス・オブジェクトを作成します。持っていなければ、DELETE ステートメントをビルドして実行します。
5. ビジネス・オブジェクトが AfterDeleteSP 属性を持っているか検査します。持っていれば、AfterDelete ストアード・プロシージャーを呼び出します。

ビジネス・オブジェクトの Retrieve 操作: 単純な検索操作としては、トップレベル・ビジネス・オブジェクト、単一カーディナリティーの子、および複数のカーディナリティーの子に使用できるストアード・プロシージャーがあります。ストアード・プロシージャーの順序は、次のとおりです。

- BeforeRetrieve
- Retrieve
- AfterRetrieve

コネクターは、単一カーディナリティーの子ビジネス・オブジェクトや複数カーディナリティーの子ビジネス・オブジェクトの検索の際に、一時オブジェクトを作成します。コネクターは、BeforeRetrieve ストアード・プロシージャーを一時ビジネ

ス・オブジェクトに適用します。また、このコンテナー用に検索された子オブジェクトのそれぞれには、AfterRetrieve ストアード・プロシージャーが適用されます。

コネクターは、ビジネス・オブジェクト・メタデータまたはビジネス・オブジェクト上のストアード・プロシージャーから動的に生成された Retrieve 照会を実行した後で、AfterRetrieve ストアード・プロシージャーを実行します。

JDBC の仕様によると、StoredProcedure 呼び出しには、次の 3 つのタイプがあります。

- {call <spName>(?,?,?)}
- {call <spName>}
- {?= call <spName>(?,?,?)}

コネクターでは、最初の 2 つのタイプがサポートされています。StoredProcedure から戻される ResultSet を処理します。

ストアード・プロシージャーの構文に RS=true と指定されている場合は、ストアード・プロシージャーから戻された結果セットが処理されます。RS=false の場合は、結果セットは処理されません。デフォルトでは、RS の値は false です。結果セットの値の処理が終了されてから、ストアード・プロシージャーの出力変数が処理されます。RS=true と指定されている場合、複数カーディナリティーの子では、関連するストアード・プロシージャーの出力変数を指定できません。

注: 結果セットの処理のサポートは、Retrieve 動詞操作および RetrieveSP に対してのみ提供されています。

Retrieve ストアード・プロシージャー (RetrieveSP) から戻された結果セットの処理: Retrieve ストアード・プロシージャーから戻された結果セットに対し、ResultSetMetaData が取得されます。結果セット内のすべての列の値が取得され、ビジネス・オブジェクト内の対応する属性に格納されます。属性のアプリケーション固有情報の ColumnName プロパティーには、属性を列と突き合わせる ResultSet 列名が含まれている必要があります。

単一カーディナリティーのオブジェクトに関しては、対応する結果セットは 1 行のみで構成されています。複数の行が結果セット内に含まれて戻された場合、エラーが報告されます。

複数カーディナリティーの子に関しては、結果セットを介して複数の行が戻される場合があります。戻された行ごとに新しいオブジェクトが作成され、コンテナーに追加されます。このコンテナーは、その後親オブジェクトの必須属性索引に追加されます。

ビジネス・オブジェクトの RetrieveByContent 操作: 単純な RetrieveByContent 操作では、ストアード・プロシージャーはトップレベル・ビジネス・オブジェクト、およびその単一カーディナリティーの子にのみ使用できます。つまり、結果セットや複数の行を戻すためには使用できません。ストアード・プロシージャーの順序は、次のとおりです。

- BeforeRetrieveByContent
- RetrieveByContent

- AfterRetrieveByContent

コネクターは、単一カーディナリティーの子ビジネス・オブジェクトや複数カーディナリティーの子ビジネス・オブジェクトの検索の際に、一時オブジェクトを作成します。複数カーディナリティーのビジネス・オブジェクトに関しては、BeforeRetrieveByContent ストアード・プロシージャーが一時ビジネス・オブジェクトに適用されます。また、このコンテナー用に検索された子オブジェクトのそれぞれには、AfterRetrieveByContent ストアード・プロシージャーが適用されます。

コネクターは、ビジネス・オブジェクト・メタデータまたはビジネス・オブジェクト上のストアード・プロシージャーから動的に生成された RetrieveByContent 照会を実行した後で、AfterRetrieveByContent ストアード・プロシージャーを実行します。このとき、階層ビジネス・オブジェクトの検索でもそのビジネス・オブジェクトの子ビジネス・オブジェクトが検索されるにもかかわらず、コネクターは、配列内のすべてのビジネス・オブジェクトに対して AfterRetrieveByContent ストアード・プロシージャーを実行します。

ビジネス・オブジェクトの Update 用 Retrieve 操作: 以下のストアード・プロシージャーが最上位ビジネス・オブジェクトに対して呼び出され、単純な Retrieve と同じ方法で、子ビジネス・オブジェクトのすべてを検索します。

ストアード・プロシージャーの順序は、次のとおりです。

- BeforeRetrieveUpdate
- RetrieveUpdate
- AfterRetrieveUpdate

これらのストアード・プロシージャーは、BeforeRetrieve および AfterRetrieve と同じ操作を実行します。異なる名前が付いているのは、コネクターに BeforeRetrieve 操作と BeforeRetrieveUpdate 操作の両方、また AfterRetrieve 操作と AfterRetrieveUpdate 操作の両方を実行させるように、それぞれに個別の属性を作成できるようにするためです。

コネクターは、単一カーディナリティーの子ビジネス・オブジェクトや複数カーディナリティーの子ビジネス・オブジェクトの検索の際に、一時オブジェクトを作成します。複数カーディナリティーのビジネス・オブジェクトに関しては、BeforeRetrieveUpdate ストアード・プロシージャーが一時ビジネス・オブジェクトに適用されます。また、このコンテナー用に検索された子オブジェクトのそれぞれには、AfterRetrieveUpdate ストアード・プロシージャーが適用されます。

コネクターは、ビジネス・オブジェクト・メタデータまたはビジネス・オブジェクト上のストアード・プロシージャーから動的に生成された RETRIEVE 照会を実行した後で、AfterRetrieveUpdate ストアード・プロシージャーを実行します。このとき、階層ビジネス・オブジェクトの検索でもそのビジネス・オブジェクトの子ビジネス・オブジェクトが検索されるにもかかわらず、コネクターは、配列内のすべてのビジネス・オブジェクトに対して AfterRetrieveUpdate ストアード・プロシージャーを実行します。

トランザクション・コミットとロールバック

コネクターは、処理すべきビジネス・オブジェクトを受信すると、必ずトランザクション・ブロックを開始します。コネクターがそのビジネス・オブジェクトを処理するときに実行する SQL ステートメントのすべてが、そのトランザクション・ブロック内にカプセル化されます。コネクターは、そのビジネス・オブジェクトの処理に成功した場合には、処理の終了後、そのトランザクション・ブロックをコミットします。エラーが発生した場合は、トランザクションをロールバックします。

ビジネス・オブジェクトの属性プロパティー

ビジネス・オブジェクト・アーキテクチャーで、属性に適用するさまざまなプロパティーを定義します。このセクションでは、コネクターがこれらのプロパティーのいくつかを解釈する方法と、ビジネス・オブジェクトを変更する際にそのプロパティーをセットする方法を解説します。

Name プロパティー

どのビジネス・オブジェクト属性にも、固有の名前が含まれていなければなりません。

Type プロパティー

どのビジネス・オブジェクト属性にも、Integer や String などの型か、子ビジネス・オブジェクトのタイプが含まれていなければなりません。コネクターでは、Date、Long Text、または String 型の属性を検出すると、その値を引用符で囲み、文字データとして取り扱います。

Cardinality プロパティー

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) を表すビジネス・オブジェクト属性では、いずれもこのプロパティーに値 1 または n が設定されています。子ビジネス・オブジェクトを表す属性はすべて、ContainedObjectVersion プロパティー (子のバージョン番号を指定) と Relationship プロパティー (値の Containment を指定) を持ちます。

Max length プロパティー

String 型の属性では、このプロパティーにより、その属性の値に許可される最大長が指定されます。

Key プロパティー

どのビジネス・オブジェクトでも、1 つ以上の単純属性がキーに指定されなければなりません。属性をキーとして定義するには、このプロパティーを Yes に設定します。ビジネス・オブジェクト属性が、タイプ String の場合、データベース中のデータ・タイプをタイプ char ではなく、タイプ Varchar にすることをお勧めします。

注: コネクターでは、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) を表す属性をキー属性に指定することについては、サポートしていません。

単純属性のキー・プロパティーを true に設定すると、コネクターは、ビジネス・オブジェクトの処理中に生成する SELECT、UPDATE、RETRIEVE、および DELETE の各 SQL ステートメントの WHERE 文節にその属性を追加します。

親/子関係が子 (複数カーディナリティーであるか、単一カーディナリティーであるかは問いません) に定義されている場合、そうした子の属性に対して Key プロパティーが true に設定されると、コネクターは、親の基本キーを SELECT ステートメントの WHERE 文節に使用し、設定された Key プロパティーは使用しません。子の外部キー属性をセットするために値が使用されるビジネス・オブジェクト属性の、名前を指定する方法は、65 ページの『属性レベルのアプリケーション固有情報』を参照してください。

Required プロパティー

Required プロパティーでは、属性が値を含む必要があるかどうかを指定します。

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属性に対してこのプロパティーが指定されていると、コネクターはこの属性があることにより、親のビジネス・オブジェクトに子ビジネス・オブジェクトが組み込まれていることを必要とします。

コネクターでは、Create 要求を伴うビジネス・オブジェクトを受信しても、以下の条件の両方が true である場合には、作成操作を失敗させます。

- 受信したビジネス・オブジェクトの必須属性に、有効な値またはデフォルト値が含まれていない場合。
- アプリケーション固有情報で、コネクターに固有の ID を生成するように指定していない。

コネクターが検索要求を持つビジネス・オブジェクトを受け取り、ビジネス・オブジェクトに要求された属性の有効な値またはデフォルト値が存在しないと、コネクターは検索操作を失敗させます。

コネクターは、子ビジネス・オブジェクトの配列を含む属性に関しては、このプロパティーを使用しません。

注: キー属性がシーケンス、またはカウンターを使用する場合、あるいはデータベースから取り込まれる場合 (UID=uid_name[=UseIfMissing]) は、Required にしないでください。

Foreign key プロパティー

コネクターでは、このプロパティーを使用して、属性が外部キーであるかどうかを判別します。

AppSpecificInfo

このプロパティーに関する詳細については、65 ページの『属性レベルのアプリケーション固有情報』を参照してください。

Default value プロパティー

このプロパティーは、データベース表の値が格納されない単純属性に値を設定するために、コネクターで使用される値 (デフォルト値) を指定します。コネクターは、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) を表す属性に関しては、このプロパティーを評価しません。

このプロパティーは、UseDefaults 構成プロパティーが true の場合にのみ使用されます。詳細については、 19 ページの表 6 を参照してください。

特殊な属性値

ビジネス・オブジェクトの単純属性は特殊値 CxIgnore を持つことがあります。コネクターはビジネス・プロセスからビジネス・オブジェクトを受け取ると、CxIgnore という値を持つ属性すべてを無視します。まるでコネクターにはそれらの属性が見えないかのように処理されます。

コネクターがデータベースからデータを検索して、SELECT ステートメントが属性にヌル値を戻したときは、コネクターはデフォルトでは、その属性の値に CxIgnore をセットします。属性のアプリケーション固有情報の UNVL パラメーターに値が指定されている場合、コネクターはヌルを表すのにその値を使用します。

コネクターは、ビジネス・オブジェクトすべてに少なくとも 1 つの基本キー属性を必要とするため、開発者は、コネクターに渡されるビジネス・オブジェクトが少なくとも 1 つの基本キーを持つように注意する必要があります。この基本キーは CxIgnore 以外であることが必要です。この要件の例外は、コネクター がカウンターまたはシーケンスを使用して、あるいはデータベースによって基本キーを生成するビジネス・オブジェクトの場合のみです。

コネクターは、データベースへのデータの挿入時に、値が指定されていないビジネス・オブジェクト属性があると、その属性の UseNullValue プロパティーに指定されている値を使用します。UseNullValue に関する詳細については、 65 ページの表 11 のUNVL=value を参照してください。

ビジネス・オブジェクトのアプリケーション固有の情報

ビジネス・オブジェクト定義内のアプリケーション固有情報は、コネクターに対し、ビジネス・オブジェクトの処理方法に関するアプリケーション依存の指示を与えるものです。コネクターでは、ビジネス・オブジェクトの属性または動詞、あるいはビジネス・オブジェクト自体から取得したアプリケーション固有情報を解析して、作成、更新、検索、および削除操作のための照会を生成します。

コネクターは、ビジネス・オブジェクトのアプリケーション固有情報の一部については、キャッシュに保管し、その情報をすべての動詞の照会をビルドするために使用します。

アプリケーション固有のビジネス・オブジェクトを拡張、または変更する場合は、ビジネス・オブジェクト定義のアプリケーション固有情報が、コネクターの予期する構文に必ず合致するよう確認してください。

注: InterChange Server の最適パフォーマンスを得るには、アプリケーション固有ビジネス・オブジェクトは、1 MB 未満にし、いかなる場合も、5 MB を超えないようにします。InterChange Server が実行されている Java 仮想マシンに対する制限のため、ビジネス・オブジェクトを大きくすることは、パフォーマンス上の問題の原因になります。

このセクションでは、コネクターがサポートするビジネス・オブジェクトのための、オブジェクト・レベル、属性、および動詞に関するアプリケーション固有情報について、その形式に関する情報を提供します。

表 10 に、ビジネス・オブジェクトのアプリケーション固有情報で使用可能な機能の概要を示します。

表 10. サポートされているビジネス・オブジェクトのアプリケーション固有情報の概説

アプリケーション固有情報の有効範囲	機能
ビジネス・オブジェクト全体	以下のものを指定します。 • 対応するデータベース表の名前。 • コネクターが論理 (ソフト) 削除実行のために WHERE 文節内に使用する値が含まれる列を定義します。 • トップレベル・ビジネス・オブジェクトがラッパーであることを指定します。
単純属性	以下のものを指定します。 • 属性に対応するデータベース列名。 • 現在のビジネス・オブジェクトの属性と親 (または子) ビジネス・オブジェクトの間の外部キー関係。 • 固有の ID 値の自動生成。 • コネクターが現在の属性の値を設定するために必要とする値が含まれる、同一ビジネス・オブジェクト内の別の属性の名前。 • 検索のソート時に現在の属性を使用するかどうか。 • 現在の属性値がヌルのときに使用する値。 • ストリング置換時の動作。 • ストリングの比較時に LIKE 演算子または = 演算子のどちらを使用するか。 • LIKE 演算子の使用時に、ワイルドカード位置として使用する値。
子または子ビジネス・オブジェクト配列を含む属性	単一カーディナリティーの子が親に所有されているかどうかを指定します。更新操作において、子データがソース・ビジネス・オブジェクトにない場合、コネクターがその子データを削除するかどうかを指定します。
ビジネス・オブジェクト動詞	動詞 Retrieve に対してのみ使用されます。テキストを使用して、検索時に WHERE 文節に組み込む属性を指定します。演算子や属性値を指定することもできます。

以下のセクションでは、この機能をより詳細に解説します。

ビジネス・オブジェクト・レベルのアプリケーション固有情報

ビジネス・オブジェクト・レベルのアプリケーション固有情報により、次のことが可能です。

- 対応するデータベース表の名前を指定する。
- 物理削除または論理削除の実行に必要な情報を提供する。
- 最上位ビジネス・オブジェクトがラッパー・オブジェクトであることを指定する。

ビジネス・オブジェクト・レベルでは、アプリケーション固有情報の形式は、コロン (:) またはセミコロン (;) によって区切られた複数のパラメーターで構成されています。

```
TN=TableName; SCN=StatusColumnName:StatusValue; WRAPPER=true|false
```

ここで、TableName は、データベース表を識別します。StatusColumnName は、論理的な削除を実行するために使用されるデータベース列の名前です。StatusValue は、ビジネス・オブジェクトが非アクティブであるか、または削除済みであるかを示す値です。また、*true|false* はトップレベル・ビジネス・オブジェクトがラッパー・オブジェクトであるかどうかを示します。

例えば、Customer ビジネス・オブジェクトで、そのアプリケーション固有情報に、以下の値が指定されているとします。

```
TN=CUSTOMER; SCN=CUSTSTATUS:DELETED
```

また、コネクターで、カスタマー削除要求を受信したとします。上記のような値が指定されている場合、コネクターは次の SQL ステートメントを発行します。

```
UPDATE CUSTOMER SET CUSTSTATUS = 'DELETED' WHERE CUSTOMER_ID = 2345
```

SCN パラメーターが含まれていないか、または値が指定されていない場合、コネクターはデータベースからビジネス・オブジェクトを物理的に削除します。つまり、Delete 動詞を伴うビジネス・オブジェクトで、アプリケーション固有情報に SCN パラメーターが含まれている場合、コネクターは論理削除を実行します。Delete 動詞を持つビジネス・オブジェクトが SCN パラメーターをアプリケーション固有情報内に含まない場合、コネクターは物理削除を実行します。

SCN プロパティーの値は、更新操作と削除操作の両方で使用できます。

- 更新を実行するとき、コネクターは `ChildUpdatePhyDelete` プロパティーの値を使用して、欠落している子データを物理的に削除するか、論理的に削除するかを判断します。子のデータを論理的に削除する場合には、SCN パラメーターの値を使用して、状況列の名前と状況値を示すテキストを取得します。詳細については、48 ページの『Update 操作』を参照してください。
- コネクターは、削除の実行時には、SCN パラメーターの値に基づいて、ビジネス・オブジェクト全体を物理的に削除するか、論理的に削除するかを決定します。SCN パラメーターに値が含まれている場合は、論理削除を実行します。SCN パラメーターに値が含まれていない場合は、物理削除を実行します。詳細については、51 ページの『Delete 操作』を参照してください。

ビジネス・オブジェクト・レベルでは、アプリケーション固有情報はラッパーの指定に使用される場合があります。

WRAPPER=*true*|*false*

wrapper パラメーターが true に設定されている場合、トップレベル・ビジネス・オブジェクトはラッパー・オブジェクトです。ラッパー・オブジェクトはデータベース表やビューによって表されません。ラッパーは関連のないビジネス・オブジェクトのコンテナーとして使用されます。コネクターはトップレベル・オブジェクトを無視し、子のみを処理します。ラッパー・オブジェクトには N のカーディナリティーを持つエンティティーまたは N-1 のカーディナリティーを持つエンティティー、あるいはその両方を含めることができます。

属性レベルのアプリケーション固有情報

属性のアプリケーション固有情報は、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。さらに、子を表す属性のアプリケーション固有情報は、親/子関係が子に保管されるか、親に保管されるかによって異なります。子または子ビジネス・オブジェクト配列を表す属性のアプリケーション固有情報については、67 ページの『属性の外部キーの指定』を参照してください。

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、名前と値のペアを表す多数のパラメーターで構成されています。どのパラメーターにも、パラメーター名とその値が含まれます。各パラメーター・セットはコロンで次のものと区切られます。

属性のアプリケーション固有情報の形式は、次のとおりです。大括弧 ([ ]) で囲まれた部分は、オプションのパラメーターです。縦線 (|) はオプション・セットのメンバー同士を区切ります。コロンは、区切り文字として予約されています。

```
CN=col_name:[FK=[fk_object_name.]fk_attribute_name]:
[UID=[AUTO|uid_name|schema_name.uid_name
[=UseIfMissing]|CW.uidcolumnname [=UseIfMissing]]]:
[PH=true|false]:[CA=set_attr_name|..set_attr_name]:[OB=[ASC|DESC]]:[UNVL=value]:
[ESC=true|false]:[FIXEDCHAR=true|false]:
[BYTEARRAY=true|false]:[USE_LIKE=true|false:
[WILDCARD_POSITION=non-negative number|NONE|BEGIN|END|BOTH]]:
[CLOB=true]
[TS=true|false]]
```

コネクターで処理する単純属性に必須のパラメーターは、列名だけです。例えば、列名だけを指定するには、次の形式を使用します。

```
CN=customer_id
```

表 11 に、それぞれの名前と値のペアを表すパラメーターを示します。

表 11. 属性アプリケーション固有情報内の名前と値のペアを表すパラメーター

パラメーター	説明
CN=*col_name*	この属性のデータベース列名。
FK=[*fk_object_name*.] *fk_attribute_name*	このプロパティーの値は、親/子関係が親ビジネス・オブジェクトまたは子に格納されているかどうかによって異なります。属性が外部キーでない場合は、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。詳細については、67 ページの『属性の外部キーの指定』を参照してください。

表 11. 属性アプリケーション固有情報内の名前と値のペアを表すパラメーター (続き)

パラメーター	説明
UID=AUTO	コネクターでは、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。属性で固有 ID の生成が必要とされていない場合には、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。ビジネス・オブジェクトの処理中に固有の ID を保存する方法の詳細は、PreserveUIDSeq プロパティー記述を参照してください。詳細については、 70 ページの『ビジネス・オブジェクトの固有 ID の生成』を参照してください。
UID=*uid_name*|schema_name.uid_name [=UseIfMissing]	
UID=CW.*uidcolumnname* [=UseIfMissing]	
PH=true|false	PH=true の場合、対応する単純属性はプレースホルダー属性です。単純属性は、アプリケーション固有の情報 (ASI) がブランクまたはヌルの場合も、プレースホルダーとなります。
CA=*set_attr_name*|..set_attr_name	コネクターは、現在のビジネス・オブジェクトそれぞれの、別の属性の名前に *set_attr_name* がセットされている場合、Create 操作中にデータベースにビジネス・オブジェクトを追加する前に、指定された属性の値を使用してこの属性の値をセットします。set_attr_name の値は子ビジネス・オブジェクトの属性を参照できませんが、set_attr_name の前にピリオドが 2 つある場合は親ビジネス・オブジェクトの属性を参照できます。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、コネクターは、別の属性の属性値をコピー (CA) せずに、現在の属性の値を使用します。
OB=[ASC|DESC]	このパラメーターに値が指定されている場合、このパラメーターが指定されている属性が子ビジネス・オブジェクト内に存在するものであれば、コネクターでは、検索照会の ORDER BY 文節に、その属性の値を使用します。コネクターは、子ビジネス・オブジェクトを昇順または降順で検索することができます。 • 昇順での検索を指定するには ASC を使用します。 • 降順の検索を指定するには DESC を使用します。 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、コネクターは、検索順序を指定するときに、このパラメーターが指定されている属性を使用しません。
UNVL=*value*	値が null の属性を含むビジネス・オブジェクトが検索された場合に、null 表現用としてコネクターに使用させる値を指定します。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、コネクターは、その属性の値として CxIgnore を挿入します。
ESC=[true|false]	コネクターが、ReplaceAllStr プロパティーで特定された各文字のすべてのインスタンスを、ReplaceStrList プロパティーで指定された置換ストリングに置き換えるかどうかを決定します。このパラメーターが値を含んでいなければ、コネクターは ReplaceStrList プロパティーの値を使用して決定します。 **注:** ESC パラメーター、ReplaceAllStr プロパティー、および ReplaceStrList プロパティーは、データベースのエスケープ文字機能をサポートしています (単一引用符のエスケープなど)。 JDBC ドライバーによって提供される Prepared Statements でも同じ機能が利用できるため、今後リリースされるコネクターでは、こうしたプロパティーのサポートが廃止される予定です。コネクターは現在、JDBC の Prepared ステートメントの使用をサポートしています。
FIXEDCHAR=true|false	表内の列が (VARCHAR 型ではなく) CHAR 型である場合に、このパラメーターが指定されている属性を固定長とするかどうかを指定します。例えば、ある特定の属性が CHAR 型の列にリンクされている場合は、その属性のアプリケーション固有の情報では FIXEDCHAR=true が指定されるため、コネクターはその属性の値を**固定長**と見なします。データベースで指定されている、その属性の MaxLength プロパティーが、必ず CHAR の長さになるようにしてください。デフォルトでは、FIXEDCHAR=false です。

表 11. 属性アプリケーション固有情報内の名前と値のペアを表すパラメーター (続き)

パラメーター	説明
BYTEARRAY=true|false	BYTEARRAY=true の場合、コネクターはデータベースに対するバイナリー・データの読み取りおよび書き込みを実行し、そのデータをストリングとして ICS または WebSphere Integrator Broker に送信します。BYTEARRAY=false がデフォルトです。詳細については、72 ページの『バイナリー・データを使用した作業』を参照してください。
USE_LIKE=true|false	コネクターがストリングを比較する時に = **演算子**または LIKE **演算子**のどちらを使用するかを指定します。USE_LIKE が true に設定されている場合、ワイルドカード照会を実行するには WILDCARD_POSITION を設定します。USE_LIKE が false に設定されている場合は、= **演算子**が使用されます。
WILDCARD_POSITION=non-negative number|NONE|BEGIN|END|BOTH	USE_LIKE が true の場合、ワイルドカードの位置を指定するために WILDCARD_POSITION が使用されます。この値は負以外の任意の数値、NONE、BEGIN、END、または BOTH に設定できます。例えば、BEGIN を使用すると、ワイルドカード文字がストリングの先頭に置かれます (%string)。END を使用すると、ワイルドカード文字がストリングの末尾に置かれます (string%)。BOTH を使用すると、ワイルドカード文字がストリングの先頭と末尾の両方に置かれます (%string%)。
CLOB=true	String 属性タイプにのみ適用可能。この属性に対応するデータベース列が CLOB データ型であることを指定します。 **注:** CLOB データ型については、以下のように定義されています。 • CLOB に対応する属性は String 型に設定されており、長さを示す値は CLOB の長さを規定するために使用されています。 • CLOB に対応する属性では、ASI=CN=xyz; CLOB=true と指定されています。 • その他のタイプの属性の ASI で CLOB を使用すると、エラーが発生します。 • CLOB=false と指定すると、エラーが発生します。 通常の String 型の属性は CLOB 対応の属性とほぼ同じですが、ASI に CLOB が使用されていません。CLOB データ型を使用する場合、4 KB 以上のサイズのデータを挿入または更新することができます。ただし、このデータ型を使用できるのは Oracle に限られており、また、Oracle でこのデータ型を使用するためには CLOB をサポートするシン・ドライバーが必要です。それ以外のドライバーを使用すると、エラーが発生する可能性があります。
TS=true|false	型 DATE の属性の場合、その属性のアプリケーション固有情報内で TS=false を指定すると、属性は、DATE 型として処理されます。TS=true の場合、属性は TIMESTAMP 型として処理されます。TS のデフォルト値は true です。

注: ビジネス・オブジェクトのどの属性にも、コネクターに照会を作成または実行させるアプリケーション固有情報が含まれない場合、コネクターは警告を記録して、動作を継続します。例外を throw することや、失敗を戻すことはありません。

属性の外部キーの指定: このプロパティーの値は、親/子関係が親ビジネス・オブジェクトに保管されるか、子ビジネス・オブジェクトに保管されるかによって異なります。

- 親に保管される場合: 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含まれるように値を設定します。

- 子に保管される場合: 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。

fk_object_name の値が子ビジネス・オブジェクトのタイプと一致せず、*fk_attribute_name* の値が親または子 (該当する場合) の属性の名前と一致しない場合、コネクターはこの属性を外部キーとして処理できません。ビジネス・オブジェクト名、および属性名は大文字小文字を区別します。

例えば、Customer ビジネス・オブジェクトが Address 子ビジネス・オブジェクトを表す `Addr[1]` 属性と、子ビジネス・オブジェクトの基本キーを外部キーとして保管する `AID` 属性を持つとします。この場合、親の外部キー属性のアプリケーション固有情報には、子ビジネス・オブジェクトのタイプ (Address) と基本キー属性の名前 (ID) が含まれていなければなりません。この例では、`AID` 属性のアプリケーション固有の情報に `FK=Address.ID` が含まれます。

外部キー属性の命名: 複数の親ビジネス・オブジェクトに、同一の子ビジネス・オブジェクトを含めることができます。このとき、子ビジネス・オブジェクトは、単一カーディナリティーの関係にあっても、複数カーディナリティーの関係にあってもかまいません、また、親/子関係は、親に保管しても、子に保管してもかまいません。ただし、親/子関係が保管される親ビジネス・オブジェクトは、いずれも同じ名前の属性を使用して、子の基本キーを格納しなければなりません。さらに、親/子関係が保管される子ビジネス・オブジェクトは、いずれも同じ名前の属性を使用して、親の基本キーを格納しなければなりません。図 5 にこれらの関係を示します。

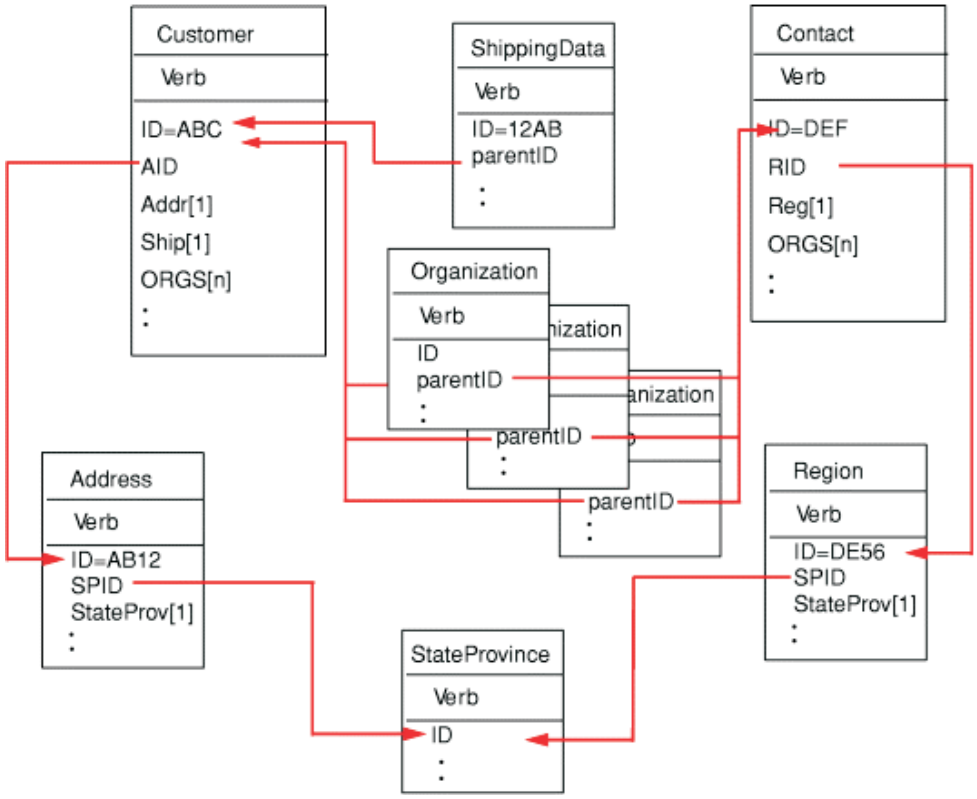


図 5. ビジネス・オブジェクト間の関係の例

図 5 に以下の関係を示します。

- Customer (ID=ABC) および Contact (ID=DEF) の ORGS[n] 属性は、Organization ビジネス・オブジェクトの配列を表しています。Organization ビジネス・オブジェクトの配列に含まれる各ビジネス・オブジェクトの外部キー値は、Customer ビジネス・オブジェクトおよび Contact ビジネス・オブジェクトに含まれる ID 属性の基本キー値に対応しています。この場合、配列内の各ビジネス・オブジェクトは、複数の親に含まれています。

  ORGS 属性のアプリケーション固有の情報は以下のようになっています。

  ```
  KEEP_RELATIONSHIP=true
  ```

  KEEP_RELATIONSHIP パラメーターに関する詳細については、71 ページの『子を表す属性のアプリケーション固有情報』を参照してください。

  Organizations の配列中の子それぞれの parentID 属性にあるアプリケーション固有情報は、現在の属性に対応するデータベース中の列の名前を含み、親の基本キー属性の名前を持つことによって現在の属性の外部キーを指定します。例えば、次のとおりです。

  ```
  CN=ORG_ID:FK=ID
  ```

  注: 親/子関係を子に保管する手法で同一の子を複数のビジネス・オブジェクトに含める場合、すべての親ビジネス・オブジェクトにおいて、子の外部キーが格納される属性の名前が同一である必要があります。子ビジネス・オブジェクトのアプリケーション固有情報の外部キー・パラメーター (FK) には、この属性の名前のみを指定します。親ビジネス・オブジェクトのタイプは指定しません。コネクターでは、どの子についても、その直接の親が所有者であると見なされます。

- Customer の Addr[1] 属性は、所有権付きの Address ビジネス・オブジェクトを表します。Customer の AID 属性では、Address ビジネス・オブジェクトの基本キーが、親の外部キーに指定されています。この場合、親の外部キー属性には、子ビジネス・オブジェクトの基本キー属性の名前だけでなく、子ビジネス・オブジェクトのタイプも含まれていなければなりません。単一カーディナリティーの子である Address を含む親は 1 つだけです。

  Addr 属性のアプリケーション固有の情報は、次のとおりです。

  ```
  CONTAINMENT=OWNERSHIP
  ```

  AID 属性にあるアプリケーション固有情報は、現在の属性に対応するデータベース中の列の名前を含み、子ビジネス・オブジェクトのタイプと子の基本キー属性の名前を持つことによって現在の属性の外部キーを指定します。例えば、次のとおりです。

  ```
  CN=FK_AD:FK=Address.ID
  ```

  子ビジネス・オブジェクトの基本キー属性のアプリケーション固有情報は、次のとおりです。

  ```
  CN=pk
  ```

- Address ビジネス・オブジェクトと Region ビジネス・オブジェクトの StateProv[1] 属性は、所有権なしの StateProvince ビジネス・オブジェクトを表します。Address ビジネス・オブジェクトおよび Region ビジネス・オブジェクトの SPID 属性には、子ビジネス・オブジェクトのタイプ (StateProvince) と、この

子ビジネス・オブジェクトの基本キー属性 (親の外部キー) の名前が含まれています。このようにして、複数の親に、同じ単一カーディナリティーの子 (StateProvince) が含まれています。

SPID 属性のアプリケーション固有の情報は、次のとおりです。

```
CONTAINMENT=NOOWNERSHIP
```

CONTAINMENT パラメーターに関する詳細については、71 ページの『子を表す属性のアプリケーション固有情報』を参照してください。

Address ビジネス・オブジェクトに含まれる Address SPID 属性のアプリケーション固有情報には、この属性に対応するデータベース列の名前が含まれています。また、この属性の外部キーが、子ビジネス・オブジェクトのタイプと子ビジネス・オブジェクトの基本キー属性名を使用して指定されています。これらは、次のような形式で指定されています。

```
CN=FK_SP:FK=StateProvince.ID
```

子ビジネス・オブジェクトの基本キー属性のアプリケーション固有情報は、次のとおりです。

```
CN=SP_ID
```

注: 親/子関係を親に保管する手法で同一の子を複数のビジネス・オブジェクトに含める場合、すべての子ビジネス・オブジェクトにおいて、親の外部キーが格納される属性の名前が同一である必要があります。

- Customer の Ship[1] 属性は、カスタマーの配送情報を含む ShippingData ビジネス・オブジェクトを表します。Customer の ID 属性は、この出荷データの外部キーとして機能します。この場合、ShippingData はその親から独立して存在できず、親が作成された後でなければ作成されないものであるため、親/子関係は子に保管されます。

  この子の parentID 属性のアプリケーション固有情報には、この属性に対応するデータベース列の名前が含まれています。また、この属性の外部キーが、親の基本キー属性名を使用して指定されています。

  ```
  CN=SD_ID:FK=ID
  ```

ビジネス・オブジェクトの固有 ID の生成: コネクターでは、UID パラメーターを参照して、ビジネス・オブジェクトの固有 ID を生成します。コネクターは、シーケンス (DB2 と Oracle の処理の場合) またはカウンター (表構造) を使用して固有 ID を生成した後、INSERT ステートメントを発行します。

IBM DB2 および Microsoft SQL Server では、INSERT ステートメントで ID を渡す必要はありません。その代わりに、作成時に ID を生成します。ビジネス・オブジェクトの作成が正常に完了すると、コネクターでこの値を検索および使用することができます。

コネクターは、シーケンスまたはカウンターを使用して ID 値を生成し、その後で INSERT ステートメントを発行します。

- UID = AUTO の場合、データベースで ID が生成され、コネクターでは生成された ID を検索しなければなりません。この設定は、IBM DB2 および Microsoft SQL Server データベースに対してのみ使用できます。

- UID =*uid_name* の場合、*uid_name* の値は、Oracle シーケンスの名前を提供します。コネクターはこの名前を使用して、属性に固有の ID を生成します。コネクターは、シーケンス値を取り出した後、キー属性を取り込み、INSERT ステートメントを発行します。この構文は、現在は DB2 および Oracle データベース用にのみ使用されています。
- UID =*uid_name*=UseIfMissing で、属性の値が CxIgnore でない場合、コネクターは固有の ID を生成するのではなく、属性値を使用します。=UseIfMissing パラメーターにはブランクを入れることはできず、大文字と小文字は区別されません。このオプションは、DB2 および Oracle データベースに対してのみ使用できます。
- UID=CW.*uidcolumnname* の場合、コネクターは WebSphere Business Integration Adapter のカウンター表を使用して、属性に固有の ID を生成します。表は、名前を構成可能で、ID という名前の単一の列を持つ状態で作成されます。この表をカスタマイズすることにより、UID (固有 ID) の生成を必要とする属性ごとに列を 1 つずつ追加することができます。uidcolumnname パラメーターを使用して、コネクターが固有の ID を生成するときに使用する列の名前を指定します。コネクターでは、UID の生成を必要とする列に関しては数値データ型のみがサポートされていることに注意してください。

  表の名前の構成方法については、UniqueIDTableName を参照してください。この表をインストールするスクリプトは次のとおりです。

  ¥connectors¥i2ADW¥dependencies¥uid_table_oracle.sql

  ¥connectors¥i2ADW¥dependencies¥uid_table_mssqlserver.sql

  ¥connectors¥i2ADW¥dependencies¥uid_table_db2.sql
- UID=CW.*uidcolumnname*=UseIfMissing であり、属性の値が CxIgnore でない場合は、コネクターは、固有 ID を生成せずに属性の値を使用します。=UseIfMissing パラメーターにはブランクを入れることはできず、大文字と小文字は区別されません。

処理中に固有 ID シーケンスを保持する方法については、26 ページの『PreserveUIDSeq』プロパティーを参照してください。

子を表す属性のアプリケーション固有情報

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属性では、その子が親に所有されるか、または複数の親の間で共用されるかを指定することができます。

単一カーディナリティーの子、または子ビジネス・オブジェクト配列を表す属性は、親および子のサブセットを更新するときのコネクターの動作を指定できます。

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属性: 単一カーディナリティーの子を表す属性の、アプリケーション固有情報の形式は、次のとおりです。

```
CONTAINMENT= [OWNERSHIP|NO_OWNERSHIP]
```

親ビジネス・オブジェクトが子ビジネス・オブジェクトを所有する単一カーディナリティーの関係を表すには、CONTAINMENT を OWNERSHIP に設定します。親ビジネス・オブジェクトが子ビジネス・オブジェクトを共用する単一カーディナリティー

の関係を表すには、CONTAINMENT を NO_OWNERSHIP に設定します。関係が親ではなく子に保管される単一カーディナリティーの関係を表す場合は、CONTAINMENT パラメーターを含めないでください。

詳細については、 39 ページの『単一カーディナリティー関係および所有権のないデータ』および 41 ページの『関係を子に格納する単一カーディナリティー関係』を参照してください。

親のキーを保管する子を表す属性: 親/子関係を子に保管するビジネス・オブジェクトの配列に対する更新操作に関しては、その子を表す属性でのみ使用できる、KEEP_RELATIONSHIPが用意されています。これを true に設定すると、既存の子のデータがソース・ビジネス・オブジェクト内に表されていない場合に、コネクターがそのデータを削除するのを防ぐことができます。

例えば、ある既存の契約が、既存サイトである New York に関連付けられているとします。また、コネクターで Contract ビジネス・オブジェクトの更新要求を受信し、その要求には San Francisco をサイトとして関連付けるための子ビジネス・オブジェクトが 1 つだけ含まれているとします。サイト・データを表す属性の KEEP_RELATIONSHIP が true である場合、コネクターは、既存の契約を更新してその契約に San Francisco との関連付けを追加しますが、その契約の New York との関連付けは削除しません。

しかし、KEEP_RELATIONSHIP が false である場合には、コネクターは、既存の子のデータのうち、ソース・ビジネス・オブジェクトに含まれないものすべてを削除します。この場合、更新対象の契約は、San Francisco のみに関連付けられることになります。

このアプリケーション固有情報の形式は、次のとおりです。

```
KEEP_RELATIONSHIP=[true|false]
```

このアプリケーション固有情報の検査の際には、大文字小文字は区別されません。

バイナリー・データを使用した作業: BYTEARRAY=true の場合、コネクターはデータベースに対するバイナリー・データの読み取りおよび書き込みを実行します。WebSphere Business Integration システムの現行バージョンではバイナリー・データがサポートされないため、バイナリー・データは String に変換されてから統合ブローカーに送信されます。このストリングの形式は、1 バイトにつき 2 文字を使用した 16 進数になります。例えば、データベース内のバイナリー・データが 3 バイトで (10 進数の) 値が (1, 65, 255) の場合、ストリングは「0141ff」となります。

動詞のアプリケーション固有情報形式

コネクターでは、Retrieve および RetrieveByContent 動詞の場合にのみ、動詞に関するアプリケーション固有情報を使用します。テキストで記述されるこの情報を使用して、検索時に WHERE 文節に組み込まれる属性を指定することができます。演算子や属性値を指定することもできます。

Retrieve および RetrieveByContent 動詞用のアプリケーション固有情報の構文を以下に示します。

```
[condition_variable conditional_operator @ [...]:[..]attribute_name [, ...]]
```

ここで、以下のように説明されます。

condition_variable	データベースの列名。
conditonal_operator	データベースでサポートされる演算子。例えば =、>、OR、AND、および IN (*value1*,*value2*)。
@	getAttrValue(*attribute_name*) によって取得した値で置換される変数。置換は定位置です。つまり、コネクターは最初の @ を区切り文字 : の後の最初の *attribute_name* 変数で置換します。
..	attribute_name 変数に指定されている属性は、直接の親にあたるビジネス・オブジェクトに属すると見なされます。この値が欠落している場合は、現在のビジネス・オブジェクトに属すると見なされます。
attribute_name	コネクターが @ を置換する値を持つ属性の名前。

このプロパティーの構文規則を学習するため、Item ビジネス・オブジェクトの item_id attribute 値が XY45、Color 属性の値が RED であると想定します。さらに、Retrieve 動詞の AppSpecificInfo プロパティーを次のように指定するとします。

```
Color='RED'
```

前述のアプリケーション固有情報の値により、コネクターは検索のために次の WHERE 文節をビルドします。

```
where item_id=XY45 and Color = 'RED'
```

より複雑な事例として、Customer ビジネス・オブジェクトの customer_id 属性値が 1234 で、creation_date が 01/01/90 である場合を想定します。また、このビジネス・オブジェクトの親の数量属性値は 20 であるとします。

さらに、Retrieve 動詞の AppSpecificInfo プロパティーを次のように指定するとします。

```
creation_date > @ OR quantity = @ AND customer_status
IN ('GOLD', 'PLATINUM') : creation_date, ..quantity
```

前述のアプリケーション固有情報の値により、コネクターは検索のために次の WHERE 文節をビルドします。

```
where customer_id=1234 and creation_date > '01/01/90' OR
quantity = 20 AND customer_status IN ('GOLD', 'PLATINUM')
```

コネクターは現在のビジネス・オブジェクトの creation_date 属性から、日付値 ('01/01/90') を取得します。数量値 (20) を親ビジネス・オブジェクトの数量属性から (アプリケーション固有情報の ..quantity に従って) 取得します。

コネクターは、Retrieve 動詞用のアプリケーション固有情報の解析を完了すると、ビジネス・オブジェクトの基本キーまたは外部キーに基づいて構成した RETRIEVE ステートメントの WHERE 文節に、解析によって得られたテキストを追加します。コネクターは、先行する AND を WHERE 文節に追加します。アプリケーション固有情報の値は、有効な SQL 構文である必要があります。RetrieveByContent の場合、ア

プリケーション固有情報は、値が取り込まれたビジネス・オブジェクトの属性に基づいて構成した RETRIEVE ステートメントの WHERE 文節に追加されます。

また、WHERE 文節では、実際の属性に代えて、親ビジネス・オブジェクト内のプレースホルダー属性を参照することもできます。このプレースホルダー属性には、アプリケーション固有情報は含まれません。属性が ASI について以下のいずれかの条件を満たしている場合は、属性をプレースホルダーにすることができます。

1. ASI=null or '' を持つ単純属性
2. ASI=PH=TRUE を持つ単純属性

例: Order ビジネス・オブジェクトに複数カーディナリティーの品目用ビジネス・オブジェクトが含まれています。このうち、特定の品目のみを検索する必要があります。この検索は、Order ビジネス・オブジェクト内のプレースホルダー属性を使用して処理することができます。子オブジェクトはすべて除去されるので、このプレースホルダーは親オブジェクトに含まれていなければなりません。プレースホルダー属性は、実行時に統合ブローカーによって、コンマ (,) で区切られた特定の line item のリストから取り込まれます。

この例では、子にあたる品目ビジネス・オブジェクトに対する Retrieve 動詞の WHERE 文節に、次の情報を追加します。

```
line_item_id in(0):..placeholder
```

ここで、line_item_id は子ビジネス・オブジェクトの ID であり、placeholder は親のプレースホルダー属性です。placeholder が値 12,13,14 を含む場合は、照会で WHERE 文節から以下のものが選択されます。

```
line_item_id in(12,13,14)
```

ここで、SELECT:..FROM:..WHERE*x* in (1,2,3) は標準のデータベース SQL 構文です。

RetrieveByContent 動詞で、WHERE 文節の長さが 0 の場合、コネクターは RETRIEVE ステートメントの WHERE 文節内のアプリケーション固有情報を使用します。この機能を使用すると、ユーザーは属性値が取り込まれていないビジネス・オブジェクトを送信し、RetrieveByContent に動詞に関するアプリケーション固有情報を指定できます。また、コネクターは動詞に関するアプリケーション固有情報のみに指定された情報に基づいて WHERE 文節を作成できます。

第 4 章 i2ADWODA を使用したビジネス・オブジェクト定義の生成

この章では、i2ADWODA について説明します。i2ADWODA は、WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) 用のビジネス・オブジェクト定義を生成する、オブジェクト・ディスカバリー・エージェント (ODA) です。コネクターは表またはビューをベースとしたオブジェクトを処理の対象とするため、i2ADWODA はデータベース表およびビューを使用して、そのデータ・ソース固有のビジネス・オブジェクトの要件を明確にします。

注: データベース概念および JDBC ドライバー (構成を目的とする場合) を熟知することで、i2ADWODA の動作の仕方の理解に役立ちます。

この章には、以下のセクションが含まれています。

インストールと使用法

このセクションでは、以下について説明します。

i2ADWODA のインストール

i2ADWODA をインストールするには、製品のインストーラーを使用します。「システム・インストール・ガイド *(UNIX 版)*」または「システム・インストール・ガイド *(Windows 版)*」の指示に従ってください。インストールが完了すると、コネクターがインストールされているシステムのディレクトリーに、以下のファイルがインストールされます。

- `ODA¥i2ADWODA.jar`
- `ODA¥i2ADWODAAgent.txt`
- `ODA¥messages¥i2ADWODAAgent_ll_TT.txt` (言語 (*ll*) および国または地域 (*TT*) に固有なメッセージ・ファイル)
- `ODA¥i2ADW¥start_i2ADWODA.bat` (Windows のみ)
- `ODA/i2ADW/start_i2ADWODA.sh` (UNIX のみ)

- bin¥CWODAEnv.bat (Windows のみ)
- bin/CWODAEnv.sh (UNIX のみ)

注: 特に断りのない限り、円記号 (¥) をディレクトリー・パスの規則として使用します。UNIX のインストールでは、円記号の代わりにスラッシュ (/) を使用します。製品のパス名はすべて、システム上の製品のインストール先ディレクトリーからの相対パスとなっています。

i2ADWODA を使用する前に

i2ADWODA を実行するには、事前に以下の作業を行う必要があります。

- 適切な JDBC ドライバーをインストールする。i2ADWODA は、JDBC 2.0 以上をサポートする JDBC ドライバーを使用することで、どのようなデータベースへの接続も可能になります。
- i2ADWODA は、対応するデータベース表および列の名前からビジネス・オブジェクトの名前と属性名を生成し、かつ、ビジネス・オブジェクト名と属性名は ISO Latin-1 で指定されている必要があるため、それぞれのデータベース・コンポーネントに Latin-1 の名前が付けられているかどうかを確認してください。Latin-1 の名前が付けられていない場合には、次のいずれかの方法で名前を付けることができます。
  - Business Object Designer において手動でビジネス・オブジェクト定義を作成する。
  - i2ADWODA によって生成された定義を編集して、すべてのビジネス・オブジェクト名および属性名を Latin-1 で指定する。
- シェルまたはバッチ・ファイルを編集用に開いて、表 12 に記載されている値を構成します。

表 12. シェルおよびバッチ・ファイルの構成変数

変数	説明	例
AGENTNAME	ODA の名前	**UNIX の場合:** AGENTNAME=i2ADWODA **Windows の場合:** set AGENTNAME=i2ADWODA
AGENT	ODA の JAR ファイルの名前	**UNIX の場合:** AGENT=$*CROSSWORLDS*/ODA/i2ADW/i2ADWODA.jar **Windows の場合:** set AGENT=%*CROSSWORLDS*%¥ODA¥i2ADW¥i2ADWODA.jar
DRIVERPATH	JDBC ドライバー・ライブラリーのパス。i2ADWODA は指定されたデータベースへの接続を確立するためのドライバー・クラスを使用します。	**UNIX の場合:** DRIVERPATH=$CROSSWORLDS/lib/xwutil.jar:$CROSSWORLDS/lib/xwbase.jar:$CROSSWORLDS/lib/xwsqlserver.jar:$CROSSWORLDS/lib/spy/lib/spy.jar **Windows の場合:** set DRIVERPATH=%CROSSWORLDS%¥lib¥xwutil.jar;%CROSSWORLDS%¥lib¥xwbase.jar;%CROSSWORLDS%¥lib¥xwsqlserver.jar;%CROSSWORLDS%¥lib¥spy¥lib¥spy.jar

表 12. シェルおよびバッチ・ファイルの構成変数 (続き)

変数	説明	例
DRIVERLIB	JDBC ドライバーによって使用されるネイティブ・ライブラリーのパス	**UNIX の場合:** DRIVERLIB=$CROSSWORLDS/bin/db2jdbc.dll **Windows の場合:** DRIVERLIB=%CROSSWORLDS%¥bin¥db2jdbc.dll

JDBC ドライバーをインストールし、シェルまたはバッチ・ファイル内に構成値の設定が終了したら、以下の作業を行ってビジネス・オブジェクトを生成する必要があります。

1. ODA を起動する。
2. Business Object Designer を起動する。
3. Business Object Designer で、6 段階のプロセスに従って ODA を構成して実行する。

これらのステップについては、以降のセクションで詳しく説明します。

i2ADWODA の起動

i2ADWODA を起動するには、ご使用のオペレーティング・システムに応じた始動スクリプトを使用します。

UNIX:

```
start_i2ADWODA.sh
```

Windows:

```
start_i2ADWODA.bat
```

Business Object Designer を使用して、i2ADWODA を構成し実行してください。Business Object Designer は、各スクリプトまたはバッチ・ファイルの AGENTNAME 変数に指定された名前で、各 ODA を位置指定します。このコネクターのデフォルトの ODA の名前は、i2ADWODA です。

複数の i2ADWODA インスタンスの実行

ODA の複数インスタンスを実行するときには、その ODA の名前を変更することをお勧めします。i2ADWODA の固有名のインスタンスをさらに作成するには、以下のようにします。

- 各インスタンス用に別個のスクリプトまたはバッチ・ファイルを作成します。
- 各スクリプトまたはバッチ・ファイルの AGENTNAME 変数に固有の名前を指定します。

いくつかのマシン上で ODA インスタンスを実行する場合は、各名前の前にホスト・マシン名を付けるようお勧めします。ODA を Object Activation Daemon に登録した場合、ORB ファインダー (osfind) を使用して、ネットワーク上の既存の CORBA オブジェクト名を見つけてください。

エラーおよびトレース・メッセージ・ファイルの処理

エラー・メッセージ・ファイルおよびトレース・メッセージ・ファイル (デフォルトは i2ADWODAAgent.txt) は、製品ディレクトリーの下の ¥ODA¥messages¥ にあります。これらのファイルには、以下の命名規則が使用されます。

*AgentName*Agent.txt

ODA スクリプトまたはバッチ・ファイルの複数インスタンスを作成して、表示された ODA にそれぞれ固有の名前を付けると、各 ODA インスタンス用のメッセージ・ファイルが得られます。代わりに、いくつかの異なる名前の ODA に同じメッセージ・ファイルを使用させることも可能です。有効なメッセージ・ファイルを指定するには、以下の 2 とおりの方法があります。

- ODA の名前を変更して、その ODA 用のメッセージ・ファイルを作成しない場合は、Business Object Designer で、ODA 構成の一部としてメッセージ・ファイル名を変更する必要があります。 Business Object Designer ではメッセージ・ファイルに名前が与えられますが、そのファイルは事実上作成されません。ODA 構成の一部として表示されたファイルが存在していない場合は、既存ファイルが参照されるように値を変更してください。
- 特定の ODA 用に既存メッセージ・ファイルをコピーして、必要に応じて変更できます。 Business Object Designer では、ユーザーが命名規則に従って各ファイルに名前を付けることを想定しています。例えば、AGENTNAME 変数に i2ADWODA1 が指定されている場合、ツールは、関連したメッセージ・ファイルの名前が i2ADWODA1Agent.txt であると見なします。そのため、Business Object Designer が ODA 構成の一部として確認のためのファイル名を提示する場合、そのファイル名は ODA の名前をベースにしたものになります。デフォルトのメッセージ・ファイルが正しく命名されていることを検証して、必要な場合はその名前を訂正してください。

重要: メッセージ・ファイルの名前が正しく指定されていない場合は、ODA を構成したときに、ODA がメッセージを使用せずに実行されてしまう原因になります。メッセージ・ファイル名の指定の詳細については、80 ページの『初期設定プロパティーの構成』を参照してください。

構成処理中に、以下を指定してください。

- i2ADWODA がエラーおよびトレース情報を書き込むファイル
- 0 から 5 を範囲とするトレース・レベル

表 13 にこれらの値を示します。

表 13. トレース・レベル

トレース・レベル	説明
0	エラーをすべてログに記録します。
1	メソッドの開始メッセージおよび既存のメッセージをすべてトレースします。
2	ODA のプロパティーおよびプロパティー値をトレースします。
3	すべてのビジネス・オブジェクトの名前をトレースします。
4	作成されたすべてのスレッドの詳細をトレースします。

表 13. トレース・レベル (続き)

トレース・レベル	説明
5	• ODA のすべてのプロパティーに対する初期値を示します。 • i2ADWODA によって作成された各スレッドの詳細な状況をトレースします。 • ビジネス・オブジェクト定義のダンプをトレースします。

これらの値をどこで構成するかについては、80 ページの『初期設定プロパティーの構成』を参照してください。

Business Object Designer での i2ADWODA の使用

このセクションでは、Business Object Designer で i2ADWODA を使用して、ビジネス・オブジェクト定義を生成する方法について説明します。 Business Object Designer の起動の詳細については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

ODA を起動させた後、Business Object Designer を起動し、構成して実行する必要があります。 Business Object Designer で、ODA を使用しビジネス・オブジェクト定義を生成するには、6 つのステップがあります。Business Object Designer には、これらのステップを手順を追って実行するためのウィザードが用意されています。

ODA の始動後、以下の手順を実行してウィザードを開始してください。

1. Business Object Designer を開きます。
2. 「ファイル」メニューから「ODA で新規作成を選択... (select the New Using ODA...)」サブメニューを選択します。

   Business Object Designer のウィザードに、「エージェントの選択」という名前の最初のウィンドウが表示されます。

ODA を選択、構成、実行するには、以下のステップに従ってください。

1. 『ODA の選択』
2. 80 ページの『初期設定プロパティーの構成』
3. 81 ページの『ノードの展開と表、ビュー、ストアード・プロシージャー、および同義語/ニックネームの選択』
4. 82 ページの『データベース・オブジェクトの選択の確認』
5. 83 ページの『定義の生成』および 83 ページの『追加情報の入力』(オプション)
6. 85 ページの『定義の保管』

ODA の選択

Business Object Designer は 6 つのステップから成るウィザードを提供します。ウィザードを開いて、実行する ODA を選択します。

ODA を選択するには、以下のようにします。

1. 「エージェントの検索」ボタンをクリックします。「検索されたエージェント」フィールドに、登録済みの ODA または現在実行中の ODA が表示されます。

注: 希望する ODA が Business Object Designer で検索されない場合は、ODA のセットアップを検査してください。

2. 表示されたリストから、希望する ODA を選択します。

   Business Object Designer の「エージェント名」フィールドに、選択した ODA が表示されます。

初期設定プロパティーの構成

Business Object Designer で最初に i2ADWODA とやり取りするときに、一連の初期設定プロパティーの入力プロンプトが出されます (図 6 を参照)。これらのプロパティーを指定したファイル名でプロファイルに保管しておくと、i2ADWODA を使用するたびに同じプロパティーを再入力する必要はありません。ODA プロファイルの指定方法については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

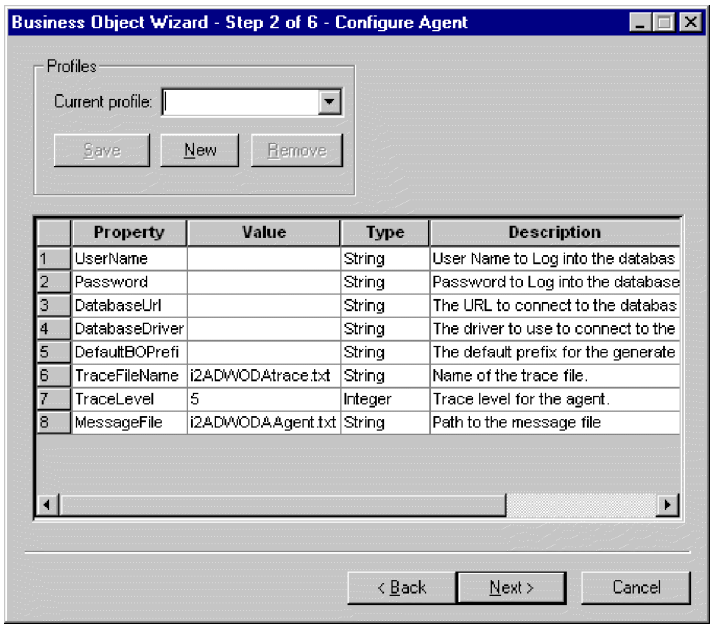


図 6. エージェント初期設定プロパティーの構成

i2ADWODA のプロパティーは、表 14 に記載されているように構成します。

表 14. i2ADWODA プロパティー

行番号	プロパティー名	プロパティー・タイプ	説明
1	UserName	String	データベースに接続する権限を持つユーザーの名前。
2	Password	String	データベースに接続する権限を持つユーザーのパスワード。
3	DatabaseUrl	String	データベースに接続するための URL。 例: jdbc:oracle:thin:@MACHINENAME:1521:SIDNAME
4	DatabaseDriver	String	接続の確立に使用されるドライバーの名前。 例: oracle.jdbc.driver.OracleDriver

表 14. i2ADWODA プロパティー (続き)

行番号	プロパティー名	プロパティー・タイプ	説明
5	DefaultBOPrefix	String	ビジネス・オブジェクト名が固有となるように、名前の先頭に付加されるテキスト。後で Business Object Designer でビジネス・オブジェクトのプロパティーの入力プロンプトが出されたときに、このテキストを必要に応じて変更できます。詳細については、83 ページの『追加情報の入力』を参照してください。
6	TraceFileName	String	i2ADWODA がトレース情報を書き込むファイル。ファイルが存在しない場合、i2ADWODA はそのファイルを ¥ODA¥i2ADW ディレクトリー内に作成します。ファイルがすでに存在している場合、i2ADWODA はその既存ファイルに情報を付加します。i2ADWODA は、命名規則に従ってファイルに名前を付けます。例えば、エージェントの名前が i2ADWODA の場合は、i2ADWODAAtrace.txt という名前のトレース・ファイルが生成されます。このプロパティーを使用して、このファイルとは異なる名前を指定します。
7	TraceLevel	Integer	i2ADWODA で使用可能なトレースのレベル。
8	MessageFile	String	エラーおよびメッセージ用のファイルの名前。 i2ADWODA は、命名規則に従ったファイル名を表示します。例えば、エージェントの名前が i2ADWODA の場合、メッセージ・ファイルのプロパティーの値は i2ADWODAAtrace.txt と表示されます。 **重要:** エラーおよびメッセージ・ファイルは、¥ODA¥messages ディレクトリー内になければなりません。 このプロパティーを使用して、既存のファイルの確認や指定をします。

重要: Business Object Designer に、存在しないファイルを表すデフォルト値が表示された場合は、メッセージ・ファイルの名前を訂正してください。その名前が正しくないと、このダイアログ・ボックスから先へ進むときに、Business Object Designer は ODA を起動したウィンドウにエラー・メッセージを表示します。このメッセージは、Business Object Designer にはポップアップされません。有効なメッセージ・ファイルを指定しないと、ODA がメッセージを使用せずに実行されてしまう原因になります。

ノードの展開と表、ビュー、ストアード・プロシージャー、および同義語/ニックネームの選択

i2ADWODA の初期設定プロパティーの構成がすべて終了すると、Business Object Designer は指定されたデータベースに接続して、データベース内にあるすべてのスキーマ名を持つツリーを表示します。これらのスキーマ名は、ツリー内のノードとして表示され、展開が可能です。クリックすると、各スキーマに含まれる表、ビュー、ストアード・プロシージャー、および同義語/ニックネームのすべてが表示されます。図 7 にこのダイアログ・ボックスでいくつかのスキーマを展開した様子を示します。

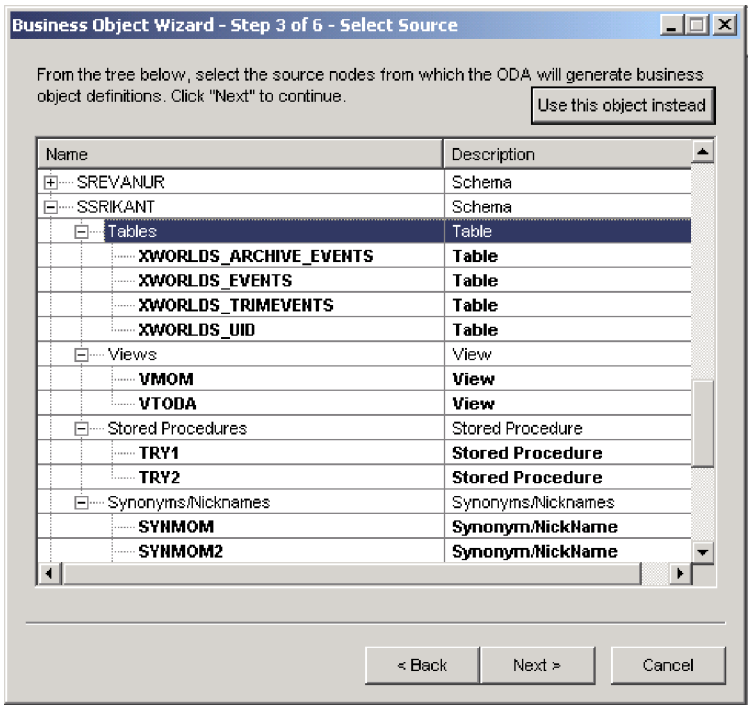


図 7. ノードを展開したスキーマのツリー

ビジネス・オブジェクト定義の生成に必要なデータが格納されているデータベース・オブジェクトをすべて特定するため、必要な表、ビュー、ストアード・プロシージャー、および同義語/ニックネームをすべて選択し、「次へ」をクリックします。戻されるオブジェクトをフィルター操作する方法については、「ビジネス・オブジェクト開発ガイド」を参照してください。

スキーマ名 ALL SCHEMAS は、スキーマが関連付けられていないオブジェクト (表やビューなど) があるデータベースのオブジェクトの検索を容易にするために使用されています。「ALL SCHEMAS」を展開すると、表、ビュー、ストアード・プロシージャー、および同義語/ニックネームを含むツリーが表示されます。これらのノードのいずれかを展開すると、そのノードに分類されるデータベース・オブジェクトが、所属するスキーマに関係なくすべて表示されます。

データベース・オブジェクトの選択の確認

生成するビジネス・オブジェクト定義に関連付けるすべてのデータベース・オブジェクトの特定を完了すると、Business Object Designer に、選択された表、ビュー、ストアード・プロシージャー、および同義語/ニックネームのみを示すダイアログ・ボックスが表示されます。図 8 にこのダイアログ・ボックスを示します。

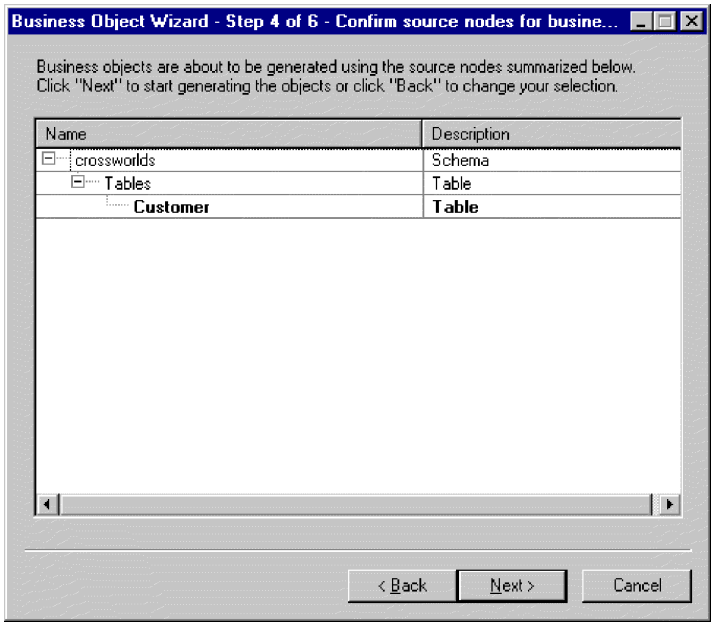


図 8. データベース・オブジェクトの選択の確認

このウィンドウには、以下のオプションがあります。

- 選択を確定するには、「次へ」をクリックします。
- 選択が正しくない場合は、「戻る」をクリックして前のウィンドウに戻り、必要な変更を行ってください。選択が正しい場合は、「次へ」をクリックします。

定義の生成

データベース・オブジェクトの確認が終了すると、次のダイアログ・ボックスに、Business Object Designer が定義を生成していることが通知されます。

追加情報の入力

i2ADWODA で追加情報が必要とされた場合、Business Object Designer は、情報の入力プロンプトを出す「BO プロパティー」ウィンドウを表示します。

「BO プロパティー」ウィンドウで、以下の情報を入力または変更します。

- *プレフィックス:* ビジネス・オブジェクト名を固有の名前にするために、名前の前に付加されるテキスト。「エージェントの構成」ウィンドウ (図 6) の *DefaultBOPrefix* プロパティーに入力した値でよければここで値を変更する必要はありません。
- *動詞:* 「*値*」フィールドをクリックし、ポップアップ・メニューから 1 つ以上の動詞を選択します。これらの動詞は、ビジネス・オブジェクトによってサポートされています。
- *ストアード・プロシージャーの追加:*「値」フィールドで「はい 」または「いいえ」をクリックします。
  - 「はい」を選択して「OK」をクリックすると、i2ADWODA はストアード・プロシージャー属性をすべてリストしたウィンドウに表示します。ビジネス・オブジェクトに追加したいストアード・プロシージャー属性を選択します。

– ビジネス・オブジェクト定義にストアード・プロシージャー属性を追加しない場合は、「いいえ」を選択します。

デフォルトは「はい」です。

注: 「BO プロパティー」ダイアログ・ボックス内のフィールドが複数の値を持つ場合、ダイアログ・ボックスが最初に表示されたとき、フィールドは空になっています。フィールドをクリックすると、その値のドロップダウン・リストが表示されます。

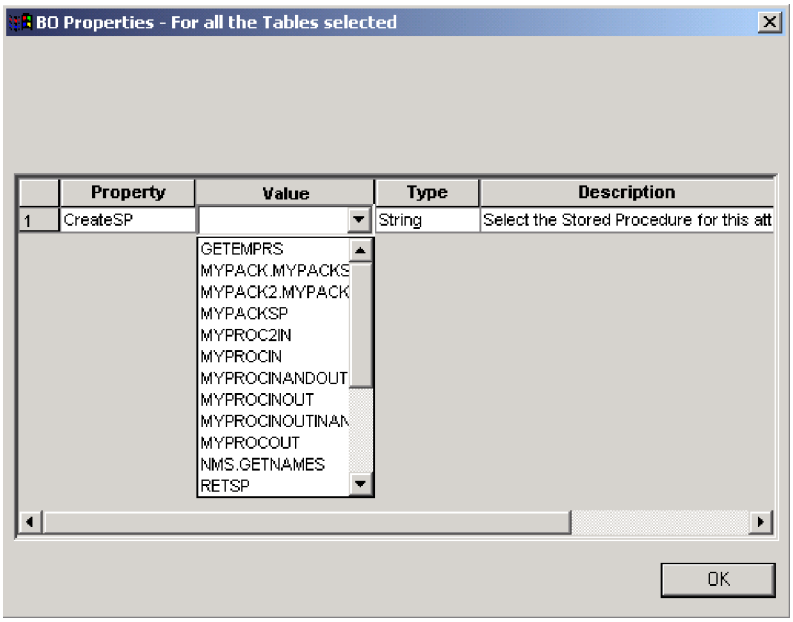


図 9. ストアード・プロシージャーとストアード・プロシージャー属性の関連付け

ビジネス・オブジェクトに追加されるストアード・プロシージャーの属性は、接続先データベースの特定のスキーマに含まれるストアード・プロシージャーのいずれかと関連付けることができます。ストアード・プロシージャーは、ドロップダウン・リストを使用して、ストアード・プロシージャーの属性ごとに選択することができます。ドロップダウン・リストには、接続先データベースの特定のスキーマに含まれるストアード・プロシージャーがすべて表示されます。ここで指定した情報に基づいて、各属性に必要な ASI 情報が生成されます。

オブジェクト・レベルの ASI (アプリケーション固有の情報) は、`TN=tableName` のようになります。

また、属性レベルでは、ASI は `CN=ColumnName` のようになります。

ビジネス・オブジェクトがストアード・プロシージャーから生成されているときに、SPForCreate などの i2 Active Data Warehouse Adapter ストアード・プロシージャー属性が、ビジネス・オブジェクトに関連付けられている場合は、ODA が、ストアード・プロシージャー属性に対して、そのスキーマ内のすべてのストアード・プロシージャー名のリストを提供し、必要なストアード・プロシージャーをビジネス・オブジェクトに関連付けることを可能にします。これにより、以下のような i2 ADW Adapter ストアード・プロシージャー属性に対する ASI が生成されます。

```
SPN=stored procedure Name; IN=a1:a2; OUT=b1:b2; IO=c1:c2
```

ここで、IN はその後に続くストアード・プロシージャー用のパラメーターが入力タイプであることを意味し、OUT はその後に続くパラメーターが出力タイプであることを意味し、IO はその後に続くパラメーターが入出力タイプであることを意味しています。ODA は、ASI を生成するとき、RS を (true または false に) 設定しません。したがって、この情報は手動で設定する必要があります。

ビジネス・オブジェクトに追加される動詞は標準の動詞です。これは、基本的には Retrieve、RetrieveByContent、Create、Update、および Delete です。

ストアード・プロシージャーの戻りパラメーターのタイプが ResultSet である場合、ODA は、結果セットを分析し、結果セットの列がビジネス・オブジェクトの属性になるようにビジネス・オブジェクトを作成します。ストアード・プロシージャーによって戻される列に対応する ASI には、`CN=StoredProcedureColumnName` が設定されます。ODA では、ドライバーから戻される JDBC メタデータ情報を基にキー属性を設定します。この情報が戻されない場合、ODA は、デフォルトではどの属性もキー属性としてマークしません。その他の属性 (長さや型など) については、いずれも、表から生成された属性に設定される場合と同様に設定されます。

定義の保管

「BO プロパティー」ダイアログ・ボックスで、必要な情報をすべて入力して「OK」をクリックした後、Business Object Designer はウィザード内に最後のダイアログ・ボックスを表示します。ここで、定義をサーバーまたはファイルに保管するか、Business Object Designer での編集用に定義を開くことができます。この詳細と、さらに変更する方法については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

生成された定義の内容

i2ADWODA で生成されたビジネス・オブジェクト定義には、以下の内容が含まれます。

- 指定されたデータベース表、ビュー、ストアード・プロシージャー、および同義語/ニックネームが指す対象の列に対応する属性 (1 列 につき 1 属性)
- 「BO プロパティー」ウィンドウで指定された動詞
- アプリケーション固有の情報
  - ビジネス・オブジェクトのレベル
  - 属性別
  - 動詞別

このセクションでは、以下の内容について説明します。

- 86 ページの『ビジネス・オブジェクト・レベルのプロパティーの生成』
- 86 ページの『属性プロパティー』
- 88 ページの『動詞』

ビジネス・オブジェクト・レベルのプロパティーの生成

i2ADWODA は、以下の情報をビジネス・オブジェクト・レベルで生成します。

- ビジネス・オブジェクトの名前
- バージョン (デフォルトで 1.0.0)
- アプリケーション固有の情報

ビジネス・オブジェクト・レベルのアプリケーション固有情報により、次のことが可能です。

- 対応するデータベース表の名前を指定する
- 物理的な削除、または論理的な削除を実行するために必要な情報を提供する

ビジネス・オブジェクト・レベルでは、アプリケーション固有の情報のフォーマットが、セミコロン (;) の区切り文字で区切られたパラメーターから構成されます。パラメーターの名前および値は、コロン (:) の区切り文字で区切られます。構文は以下のとおりです。

```
TN=TableName; SCN=StatusColumnName:StatusValue
```

では、TableName はデータベース表を示します。また、*StatusColumnName* は論理削除の実行に使用されるデータベース列の名前であり、*StatusValue* はビジネス・オブジェクトが非アクティブまたは削除済みであることを示す値です。

i2ADWODA がこのレベルで生成する AppSpecificInfo には、指定されたデータベース表、ビュー、ストアード・プロシージャー、または同義語/ニックネームの名前に対応する値が 1 つだけ含まれます。状況列の値の指定については、64 ページの『ビジネス・オブジェクト・レベルのアプリケーション固有情報』を参照してください。

属性プロパティー

このセクションでは、i2ADWODA が属性ごとに生成するプロパティーについて説明します。属性の詳細については、60 ページの『ビジネス・オブジェクトの属性プロパティー』を参照してください。

Name プロパティー

i2ADWODA は、データベース表またはビュー内にある列名から属性名の値を取得します。

Data Type プロパティー

属性のタイプの設定時に、i2ADWODA は表またはビューの列のデータ・タイプを、対応する IBM WebSphere Business Integration Adapter Business Object タイプに変換します。この変換は 2 つのステップで実行されます。まず、データベース内のデータ・タイプが JDBC タイプに変換されます。次に、JDBC タイプが IBM WebSphere Business Integration Adapter Business Object タイプに変換されます。最初の変換はご使用の JDBC ドライバーによって実行されます。ご使用の JDBC ドライバーの資料を参照してください。JDBC タイプへの個々のデータベース・タイプのマッピングの詳細については、JDBC 仕様 (2.0 以上) を参照してください。87 ページの表 15 は、JDBC タイプから対応する IBM WebSphere Business Integration Adapter Business Object タイプへの変換を示しています。

表 15. データ型の対応

JDBC 型	WebSphere Business Integration Adapter ビジネス・オブジェクト・タイプ
BIT	BOOLEAN
CHAR	STRING
VARCHAR	STRING
LONGVARCHAR	STRING
INTEGER	INTEGER
NUMERIC	INTEGER
SMALLINT	INTEGER
TINYINT	INTEGER
BIGINT	INTEGER
DATE	DATE
TIME	DATE
TIMESTAMP	DATE
DECIMAL	STRING
DOUBLE	DOUBLE
FLOAT	DOUBLE
REAL	FLOAT
BINARY	STRING, BYTEARRAY=TRUE を以下に追加 AppSpecificInfo
VARBINARY	STRING, BYTEARRAY=TRUE を以下に追加 AppSpecificInfo

注: 列のデータ型が、表 15 に示されているデータ型のいずれにも該当しない場合、i2ADWODA は列をスキップして、列を処理できないというメッセージを表示します。

Cardinality プロパティー

i2ADWODA は、すべての単純な属性のカーディナリティーを 1 に設定します。

MaxLength プロパティー

i2ADWODA は varchar、char、またはテキスト・データ型の所定の長さから、ストリングの長さを取得します。

IsKey プロパティー

列が表の基本キーである場合、i2ADWODA はその列をキー属性としてマークします。ただし、ビジネス・オブジェクトを生成するソース・ノードとして表ではなくビューが選択されている場合は、i2ADWODA はその列をキー属性としてマークしません。この場合、キー属性を手動で設定する必要があります。

IsForeignKey プロパティー

i2ADWODA では IsForeignKey プロパティーの設定は行われません。このプロパティーは、Business Object Designer で設定することができます。

IsRequired プロパティー

表またはビュー内で非ヌルとして指定されているフィールドに対しては、i2ADWODA は必須属性としてマークを付けます。ただし、i2ADWODA はそのキー・フィールドには必須のマークを付けません。キー・フィールドは、シーケンスが関連付けられていたり、ID 列になっていたりする可能性があるためです。

AppSpecificInfo プロパティー

i2ADWODA によって属性レベルで組み込まれる AppSpecificInfo プロパティーのパラメーターは、2 つあります。指定パラメーターの構文は、次のとおりです。

```
CN=ColumnName
```

ここで、ColumnName は、特定の属性に関連付けられたデータベース表またはビュー内にある列の名前です。

```
BYTEARRAY=true|false
```

i2ADWODA はバイナリー・データを含む列を認識し、AppSpecificInfo プロパティーが BYTEARRAY=true の String タイプの属性を作成します。

注: 追加の AppSpecificInfo パラメーターは、Business Object Designer で設定することができます。これらのパラメーターに関する詳細については、65 ページの『属性レベルのアプリケーション固有情報』を参照してください。

動詞

i2ADWODA は、「BO プロパティー」ウィンドウで指定された動詞を生成します。各動詞の AppSpecificInfo プロパティーを作成しますが、プロパティーへのデータの取り込みは行いません。詳細については、72 ページの『動詞のアプリケーション固有情報形式』を参照してください。

ビジネス・オブジェクト定義ファイルのサンプル

ビジネス・オブジェクト定義のサンプルを、以下に示します。

```
[BusinessObjectDefinition]
Name = CUSTOMER
Version = 1.0.0
AppSpecificInfo = TN=ra_customers;SCN=

              [Attribute]
              Name = customer_id
              Type = Integer
              Cardinality = 1
```

```
                MaxLength = 0
                IsKey = true
                IsForeignKey = false
                IsRequired = false
                AppSpecificInfo = CN=customer_id
                DefaultValue =
[End]

                *******Other attributes *********

                [Attribute]
                Name = ObjectEventId
                Type = String
                Cardinality = 1
                MaxLength = 0
                IsKey = false
                IsForeignKey = false
                IsRequired = false
                AppSpecificInfo =
                DefaultValue =
[End]

                [Verb]
                Name = Delete
                AppSpecificInfo =
[End]

                [Verb]
                Name = Update
                AppSpecificInfo =
[End]

                [Verb]
                Name = Create
                AppSpecificInfo =
[End]

                [Verb]
                Name = Retrieve
                AppSpecificInfo =
[End]
[End]
```

子ビジネス・オブジェクトを含む属性の挿入

Business Object Designer を使用して、単一カーディナリティー、または複数カーディナリティーの子ビジネス・オブジェクトを表す属性を挿入することができます。詳細については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

ビジネス・オブジェクト定義への情報の追加

データベース表およびビューには、ビジネス・オブジェクト定義に必要とされる情報の一部がない場合があります。そのため、この場合は i2ADWODA で作成されたビジネス・オブジェクト定義に情報を追加する必要があります。詳細については、37 ページの『第 3 章 コネクターのビジネス・オブジェクトについて』を参照してください。

ビジネス・オブジェクトの定義を調べる場合や情報を追加する場合は、Business Object Designer またはテキスト・エディターを使用できます。修正済みの定義を

WebSphere Business Integration Adapter リポジトリーに再ロードするには、Business Object Designer または repos_copy コマンド (ICS が統合ブローカーの場合) を使用する方法があります。

第 5 章 トラブルシューティングおよびエラー処理

この章では、WebSphere Business Integration Adapter for i2 Active Data Warehouse (ADW) を始動または実行するときに発生する可能性のある問題について説明します。以下のセクションがあります。

- 『始動時の問題』
- 『イベント処理』
- 『マッピング (ICS 統合ブローカーのみ)』
- 93 ページの『エラー処理とロギング』
- 94 ページの『アプリケーションへの接続不可』
- 95 ページの『resource busy エラー』

始動時の問題

コネクターの始動を試行しているときに問題が発生した場合は、統合ブローカーが稼働中であるかどうかを確認してください。

イベント処理

イベント表内にイベントが存在し、コネクターの実行中にこれらのイベントが処理されない場合は、以下の点を確認してください。

- 関係のあるビジネス・プロセスが実行中である。
- イベント表内のビジネス・オブジェクトの名前が、ビジネス・プロセス・ポートのために指定されたビジネス・オブジェクトの名前と一致している。

マッピング (ICS 統合ブローカーのみ)

このセクションでは、以下について説明します。

- 『マッピングの問題』
- 『日付型変換』

マッピングの問題

ビジネス・オブジェクトがマップされていない場合、またはマッピングが呼び出されていない場合は、マップが正しいディレクトリーにインストールされているかどうかを確認してください。

日付型変換

注: この日付変換手順は、コネクターのバージョン 1.5.0 より前のバージョンにのみ適用されます。

WebSphere Business Integration Adapter マップを使用して、データベース内に Date 形式で格納されたデータを、WebSphere Business Integration Adapter ビジネス・オブジェクトが使用する String 形式に変換します。

例えば、Oracle データベース内に格納されている以下の日付を変換する場合を想定します。

```
Sun Jan 01 00:00:00 CEST 1999
```

この日付を、WebSphere Business Integration Adapter i2ADW ビジネス・オブジェクト内で処理される以下のストリングに変換します。

```
Jan 01 1999 00:00:00
```

この変換を実行するには、WebSphere Business Integration Adapter マッピング内のデータ形式変更用に定義された DtpDate() コンストラクターおよび DtpSplitString() コンストラクターを使用します。これらのコンストラクターおよび構築されるオブジェクトのクラスの構文および説明については、「*マップ開発ガイド*」を参照してください。

WebSphere Business Integration Adapter マップを使用して Date 値を String に変換するには、以下のステップを実行します。

1. DtpSplitString() とスペース区切り文字を使用して、ストリングを 6 個に分割し、分割した各部分を DtpDate が使用できる順序に並べ換えます。上記の例の日付を変換するには、以下の構文を使用します。

   ```
   DtpSplitString OurSplitString = new DtpSplitString
      ("Sun Jan 01 00:00:00 CEST 1999"," ");
   ```

   上記のステートメントでは、OurSplitString は DtpSplitString 型のユーザー定義変数であり、スペースが区切り文字として指定されています。

2. DtpSplitString クラスの nextElement() メソッドを使用して、新規に作成した OurSplitString 変数をループ処理することにより、変数の 6 個の要素それぞれを String 型の要素から構成される配列に格納します。以下の例では、OurStringPieces を出力配列として指定します。

   ```
   String[] OurStringPieces = new String[6];
   for (i=0;i<=5;i=i+1){
      OurStringPieces[i]=OurSplitString.nextElement();
   }
   ```

   このループ処理により、以下の配列要素が生成されます。

   ```
   OurStringPieces[0] = Sun
   OurStringPieces[1] = Jan
   OurStringPieces[2] = 01
   OurStringPieces[3] = 00:00:00
   OurStringPieces[4] = CEST
   OurStringPieces[5] = 1999
   ```

3. DtpDate の入力のために必要なストリングの各部分を連結します。この変換では、"M D Y h:m:s" を DtpDate の入力形式として使用して、変換されたストリングが "Jan 01 1999 00:00:00" のように表示されるようにします。この例の String は、OurStringPieces 配列の要素 1、2、5、および 3 を使用します。

   ```
   OurConcatenatedString =
   OurStringPieces[1]+OurStringPieces[2]+OurStringPieces[5]+OurStringPieces[3];
   ```

4. 新規の連結されたストリングを DtpDate への入力として使用します。

```
DtpDate OurDtpDate = new DtpDate(OurConcatenatedString,"M D Y h:m:s");
```

Date 値を DtpDate 形式に変換した後で、マップ内の日付を処理できます。

エラー処理とロギング

コネクターは、ビジネス・オブジェクトと動詞の現在の処理が失敗する原因となる条件を検出すると、エラー・メッセージをログに記録します。コネクターは、このようなエラーが発生すると、ビジネス・プロセスから受信した情報に基づいて、失敗したビジネス・オブジェクトのテキスト表現も出力します。コネクターは、構成に応じてコネクター・ログ・ファイルまたは標準出力ストリームにテキストを書き込みます。このテキストは、エラーのソースを判別するために利用できます。

エラー・タイプ

表 16 では、コネクターが各トレース・レベルで出力するトレース・メッセージのタイプについて説明します。これらのメッセージは、Java コネクター実行ラッパーおよび WebSphere MQ メッセージ・インターフェースなどの WebSphere Business Integration Adapter アーキテクチャーによるトレース・メッセージ出力に追加されます。

表 16. コネクター・トレース・メッセージ

トレース・レベル	トレース・メッセージ
レベル 0	コネクター・バージョンを識別するメッセージ。このレベルではその他のトレースは実行されません。これはデフォルト値です。
レベル 1	• 状況メッセージ。 • 処理される各ビジネス・オブジェクトの識別 (キー) 情報を提供するメッセージ。 • pollForEvents メソッドが実行されるごとに配信されるメッセージ。
レベル 2	• ビジネス・オブジェクトの処理中にコネクターが検出または検索した配列や子ビジネス・オブジェクトなどの情報を含むビジネス・オブジェクト・ハンドラー・メッセージ。 • ビジネス・オブジェクトが gotApplEvent() または executeCollaboration() から統合ブローカーに追加されるごとにログに記録されるメッセージ。 • ビジネス・オブジェクトがビジネス・オブジェクト要求として受信されたことを示すメッセージ。
レベル 3	• コネクターがビジネス・オブジェクト内の外部キーを検出または設定した時点などの情報を含む外部キー処理メッセージ。 • ビジネス・オブジェクト処理についての情報を提供するメッセージ。例えば、これらのメッセージは、コネクターがビジネス・オブジェクト間の一致を検出したときや、子ビジネス・オブジェクトの配列内でビジネス・オブジェクトを検出したときに配信されます。

表 16. コネクター・トレース・メッセージ (続き)

トレース・レベル	トレース・メッセージ
レベル 4	• アプリケーション固有の情報メッセージ。例えば、ビジネス・オブジェクトのアプリケーション固有の情報フィールドを解析する関数によって戻された値を示すメッセージです。 • コネクターがある関数を開始または終了した時点を識別するメッセージ。このメッセージは、コネクターのプロセス・フローをトレースするために利用できます。 • すべてのスレッド固有のメッセージ。コネクターが複数のスレッドを作成する場合は、新規のスレッドを作成するごとに 1 つのメッセージが生成されます。
レベル 5	• コネクターの初期設定を示すメッセージ。例えば、統合ブローカーから検索された構成プロパティーの値を示すメッセージです。 • アプリケーション内で実行されたステートメントを含むメッセージ。このトレース・レベルのコネクター・ログ・ファイルには、宛先のアプリケーション内で実行されたすべてのステートメント、および置換されたすべての変数の値が含まれます。 • コネクターが処理を開始する前のビジネス・オブジェクトの表現 (コネクターがビジネス・プロセスからビジネス・オブジェクトを受信したときのビジネス・オブジェクトの状態を表示します) およびコネクターが処理を完了した後のビジネス・オブジェクトの表現 (コネクターがビジネス・プロセスにビジネス・オブジェクトを戻したときのビジネス・オブジェクトの状態を表示します) を示すメッセージ。 • ビジネス・オブジェクト・ダンプを示すメッセージ。 • コネクターが実行中に作成する各スレッドの状況を示すメッセージ。

エラー・メッセージ

コネクター・メッセージ・ファイル

コネクターが生成するすべてのエラー・メッセージは、JDBCConnector.txt または JDBCConnector_*ll_TT*.txt というメッセージ・ファイルに保管されます (*ll* の部分には言語、*TT* の部分には国または地域を示す文字が入ります)。各エラーでは、エラー番号に続いてエラー・メッセージが示されます。その例を以下に示します。

```
20017
コネクター・インフラストラクチャーのバージョンが一致しません。

20018
Connection from {1} to the Application is lost! Please enter 'q'
to stop the connector, then restart it after the problem is fixed.

20019
エラー: pollForEvent() の ev_id が NULL です。
```

アプリケーションへの接続不可

コネクターが接続の確立に失敗した場合、コネクターは FAIL を統合ブローカーに送信し、終了します。

AutoCommit を false に設定している場合に PingQuery が失敗すると、コネクターはデータベースへの新規の接続を作成しようとします。データベースへの新規接続の作成に成功した場合、コネクターは処理を続行します。失敗した場合、コネクターは APPRESPONSETIMEOUT を戻します。この結果、コネクターは終了します。

fetch out-of-sequence エラー

Oracle データベースのバージョン 8.0 および 8.1 を Sun Solaris または Oracle 8.1 とともに Windows 2000 で使用する場合は、AutoCommit プロパティーを false に設定する必要があります。false に設定しないと、ORA-01002 (フェッチ順序が無効です (fetch out-of-sequence)) というエラー・メッセージが表示されます。Oracle データベースの以前のバージョンでは、このエラーは発生しません。AutoCommit を false に設定すると、パフォーマンスが向上します。

resource busy エラー

注: このコネクターでこのエラーが発生するのは、Oracle データベース上で実行されているときだけです。

コネクターがアプリケーション内のデータを検索または変更しているときに、以下のようなエラーが発生することがあります。

```
[Time: 2001/05/29 16:30:07.356] [System: ConnectorAgent] [SS: SOVTConnector]
[Type: Trace] [Mesg: Select CLIENT,COUNTRY,STRT_CODE,CITY_CODE,
CITYP_CODE,STRTYPEAB,COMMU_CODE,REGIOGROUP,TAXJURCODE from ADRSTREET where
CLIENT='100' and COUNTRY='DE' and STRT_CODE='000001114136' FOR UPDATE NOWAIT]
[Time: 2001/05/29 16:30:07.526] [System: ConnectorAgent] [SS: SOVTConnector]
[Type: Trace ] [Mesg: :logMsg]
[Time: 2001/05/29 16:30:07.536] [System: ConnectorAgent] [SS: SOVTConnector]
[Type: Error ] [MsgID: 37002]
[Mesg: Execution of Retrieve statement failed : java.
sql.SQLException: ORA-00054: Versuch, mit NOWAIT eine bereits
belegte Ressourceanzufordern.]
```

このエラーは、現在ロックされているレコードの更新をコネクターが試行したときに発生します。レコードは他のプロセスによってロックされているか、またはコネクターがマルチスレッドである場合にはコネクター自体によってロックされている可能性があります。

更新処理の際にレコードはロックされていなければなりません。コネクターは、統合ブローカーの受信したオブジェクトの変更後イメージを探し出そうとし、その際にデータの保全のためにデータベース内のオブジェクト全体をロックします。

この問題を解決するには、コネクターがレコードに対するロックを取得するのを妨げているプロセスを停止するか、コネクターの RetryCountInterval 構成プロパティーを調整します。

付録 A. コネクターの標準構成プロパティー

この付録では、WebSphere Business Integration Adapter のコネクター・コンポーネントの標準構成プロパティーについて説明します。この付録の内容は、以下の統合ブローカーで実行されるコネクターを対象としています。

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (WebSphere Message Brokers (WMQI) と総称)
- WebSphere Application Server (WAS)

コネクターによっては、一部の標準プロパティーが使用されないことがあります。Connector Configurator から統合ブローカーを選択するときには、そのブローカーで実行されるアダプターについて構成する必要のある標準プロパティーのリストが表示されます。

コネクター固有のプロパティーの詳細については、該当するアダプターのユーザーズ・ガイドを参照してください。

注: 本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムを使用している場合は、円記号をスラッシュ (/) に置き換えてください。また、各オペレーティング・システムの規則に従ってください。

新規プロパティーと削除されたプロパティー

以下の標準プロパティーは、本リリースで追加されました。

新規プロパティー

- XMLNameSpaceFormat

削除されたプロパティー

- RestartCount

標準コネクター・プロパティーの構成

アダプター・コネクターには 2 つのタイプの構成プロパティーがあります。

- 標準構成プロパティー
- コネクター固有のプロパティー

このセクションでは、標準構成プロパティーについて説明します。コネクター固有の構成プロパティーについては、該当するアダプターのユーザーズ・ガイドを参照してください。

Connector Configurator の使用

Connector Configurator からコネクター・プロパティーを構成します。Connector Configurator には、System Manager からアクセスします。Connector Configurator の使用法の詳細については、本書の Connector Configurator に関する付録を参照してください。

注: Connector Configurator と System Manager は、Windows システム上でのみ動作します。コネクターを UNIX システム上で稼動している場合でも、これらのツールがインストールされた Windows マシンが必要です。UNIX 上で動作するコネクターのコネクター・プロパティーを設定する場合は、Windows マシン上で System Manager を起動し、UNIX の統合ブローカーに接続してから、コネクター用の Connector Configurator を開く必要があります。

プロパティー値の設定と更新

プロパティー・フィールドのデフォルトの長さは 255 文字です。

コネクターは、以下の順序に従ってプロパティーの値を決定します (最も番号の大きい項目が他の項目よりも優先されます)。

1. デフォルト
2. リポジトリー (WebSphere InterChange Server が統合ブローカーである場合のみ)
3. ローカル構成ファイル
4. コマンド行

コネクターは、始動時に構成値を取得します。実行時セッション中に 1 つ以上のコネクター・プロパティーの値を変更する場合は、プロパティーの**更新メソッド**によって、変更を有効にする方法が決定されます。標準コネクター・プロパティーには、以下の 4 種類の更新メソッドがあります。

- **動的**
  変更を System Manager に保管すると、変更が即時に有効になります。例えば WebSphere Message Broker で稼動している場合など、コネクターがスタンドアロン・モードで (System Manager から独立して) 稼動している場合は、構成ファイルでのみプロパティーを変更できます。この場合、動的更新は実行できません。
- **エージェント再始動 (ICS のみ)**
  アプリケーション固有のコンポーネントを停止して再始動しなければ、変更が有効になりません。
- **コンポーネント再始動**
  System Manager でコネクターを停止してから再始動しなければ、変更が有効になりません。アプリケーション固有コンポーネントまたは統合ブローカーを停止、再始動する必要はありません。
- **サーバー再始動**
  アプリケーション固有のコンポーネントおよび統合ブローカーを停止して再始動しなければ、変更が有効になりません。

特定のプロパティーの更新方法を確認するには、「Connector Configurator」ウィンドウ内の**「更新メソッド」**列を参照するか、次に示す 99 ページの表 17 の「更新メソッド」列を参照してください。

標準プロパティーの要約

表 17 は、標準コネクター構成プロパティーの早見表です。標準プロパティーの依存関係は RepositoryDirectory に基づいているため、コネクターによっては使用されないプロパティーがあり、使用する統合ブローカーによってプロパティーの設定が異なる可能性があります。

コネクターを実行する前に、これらのプロパティーの一部の値を設定する必要があります。各プロパティーの詳細については、次のセクションを参照してください。

注: 表 17 の「注」列にある「Repository Directory は REMOTE」という句は、ブローカーが InterChange Server であることを示します。ブローカーが WMQI または WAS の場合には、リポジトリー・ディレクトリーは LOCAL に設定されます。

表 17. 標準構成プロパティーの要約

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
AdminInQueue	有効な JMS キュー名	*CONNECTORNAME* /ADMININQUEUE	コンポーネント再始動	Delivery Transport は JMS
AdminOutQueue	有効な JMS キュー名	*CONNECTORNAME*/ADMINOUTQUEUE	コンポーネント再始動	Delivery Transport は JMS
AgentConnections	1 から 4	1	コンポーネント再始動	Delivery Transport は MQ および IDL: Repository Directory は <REMOTE> (ブローカーは ICS)
AgentTraceLevel	0 から 5	0	動的	
ApplicationName	アプリケーション名	コネクター・アプリケーション名として指定された値	コンポーネント再始動	
BrokerType	ICS、WMQI、WAS		コンポーネント再始動	
CharacterEncoding	ascii7、ascii8、SJIS、Cp949、GBK、Big5、Cp297、Cp273、Cp280、Cp284、Cp037、Cp437 **注:** これは、サポートされる値の一部です。	ascii7	コンポーネント再始動	
ConcurrentEventTriggeredFlows	1 から 32,767	1	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)
ContainerManagedEvents	値なしまたは JMS	値なし	コンポーネント再始動	Delivery Transport は JMS

表 17. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
ControllerStoreAndForwardMode	true または false	true	動的	Repository Directory は <REMOTE> (ブローカーは ICS)
ControllerTraceLevel	0 から 5	0	動的	Repository Directory は <REMOTE> (ブローカーは ICS)
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	コンポーネント再始動	JMS トランスポートのみ
DeliveryTransport	MQ、IDL、または JMS	JMS	コンポーネント再始動	Repository Directory がローカルの場合は、値は JMS のみ
DuplicateEventElimination	true または false	false	コンポーネント再始動	JMS トランスポートのみ: Container Managed Events は <NONE> でなければならない
FaultQueue		*CONNECTORNAME*/FAULTQUEUE	コンポーネント再始動	JMS トランスポートのみ
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory または CxCommon.Messaging.jms.SonicMQFactory または任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	コンポーネント再始動	JMS トランスポートのみ
jms.MessageBrokerName	FactoryClassName が IBM の場合は crossworlds.queue.manager を使用。FactoryClassName が Sonicの場合 localhost:2506 を使用。	crossworlds.queue.manager	コンポーネント再始動	JMS トランスポートのみ
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	JMS トランスポートのみ
jms.Password	任意の有効なパスワード		コンポーネント再始動	JMS トランスポートのみ
jms.UserName	任意の有効な名前		コンポーネント再始動	JMS トランスポートのみ
JvmMaxHeapSize	ヒープ・サイズ (メガバイト単位)	128m	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)

表 17. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
JvmMaxNativeStackSize	スタックのサイズ (キロバイト単位)	128k	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)
JvmMinHeapSize	ヒープ・サイズ (メガバイト単位)	1m	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)
ListenerConcurrency	1 から 100	1	コンポーネント再始動	Delivery Transport は MQ でなければならない
Locale	en_US、 ja_JP、 ko_KR、 zh_CN、 zh_TW、 fr_FR、 de_DE、 it_IT、 es_ES、 pt_BR **注:** これは、サポートされるロケールの一部です。	en_US	コンポーネント再始動	
LogAtInterchangeEnd	true または false	false	コンポーネント再始動	Repository Directory は <REMOTE> でなければならない (ブローカーは ICS)
MaxEventCapacity	1 から 2147483647	2147483647	動的	Repository Directory は <REMOTE> でなければならない (ブローカーは ICS)
MessageFileName	パスまたはファイル名	CONNECTORNAMEConnector.txt	コンポーネント再始動	
MonitorQueue	任意の有効なキュー名	*CONNECTORNAME*/MONITORQUEUE	コンポーネント再始動	JMS トランスポートのみ: DuplicateEvent Elimination は true でなければならない
OADAutoRestartAgent	true または false	false	動的	Repository Directory は <REMOTE> でなければならない (ブローカーは ICS)

表 17. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
OADMaxNumRetry	正数	1000	動的	Repository Directory は <REMOTE> でなければならない (ブローカーは ICS)
OADRetryTimeInterval	正数 (単位: 分)	10	動的	Repository Directory は <REMOTE> でなければならない (ブローカーは ICS)
PollEndTime	HH:MM	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒) no (ポーリングを使用不可にする) key (コネクターのコマンド・プロンプト・ウィンドウで文字 p が入力された場合にのみポーリングする)	10000	動的	
PollQuantity	1 から 500	1	エージェント再始動	JMS トランスポートのみ: Container Managed Events を指定
PollStartTime	HH:MM (HH は 0 から 23、MM は 0 から 59)	HH:MM	コンポーネント再始動	
RepositoryDirectory	メタデータ・リポジトリーの場所		エージェント再始動	ICS の場合は <REMOTE> に設定する。WebSphere MQ Message Brokers および WAS の場合: C:¥crossworlds¥repository に設定する
RequestQueue	有効な JMS キュー名	*CONNECTORNAME*/REQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
ResponseQueue	有効な JMS キュー名	*CONNECTORNAME*/RESPONSEQUEUE	コンポーネント再始動	Delivery Transport が JMS の場合: Repository Directory が <REMOTE> の場合のみ必要
RestartRetryCount	0 から 99	3	動的	

表 17. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
RestartRetryInterval	適切な正数 (単位: 分): 1 から 2147483547	1	動的	
RHF2MessageDomain	mrm、xml	mrm	コンポーネント再始動	Delivery Transport が JMS であり、かつ WireFormat が CwXML である
SourceQueue	有効な WebSphere MQ 名	*CONNECTORNAME*/SOURCEQUEUE	エージェント再始動	Delivery Transport が JMS であり、かつ Container Managed Events が指定されている場合のみ
SynchronousRequestQueue		*CONNECTORNAME*/SYNCHRONOUSREQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
SynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	Delivery Transport は JMS
SynchronousResponseQueue		*CONNECTORNAME*/SYNCHRONOUSRESPONSEQUEUE	コンポーネント再始動	Delivery Transport は JMS
WireFormat	CwXML、CwBO	CwXML	エージェント再始動	Repository Directory が <REMOTE> でない場合は CwXML。Repository Directory が <REMOTE> であれば CwBO
WsifSynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	WAS のみ
XMLNameSpaceFormat	short、long	short	エージェント再始動	WebSphere MQ Message Brokers および WAS のみ

標準構成プロパティー

このセクションでは、各標準コネクター構成プロパティーの定義を示します。

AdminInQueue

統合ブローカーからコネクターへ管理メッセージが送信されるときに使用されるキューです。

デフォルト値は CONNECTORNAME/ADMININQUEUE です。

AdminOutQueue

コネクターから統合ブローカーへ管理メッセージが送信されるときに使用されるキューです。

デフォルト値は CONNECTORNAME/ADMINOUTQUEUE です。

AgentConnections

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

AgentConnections プロパティーは、orb.init[] により開かれる ORB (オブジェクト・リクエスト・ブローカー) 接続の数を制御します。

このプロパティーのデフォルト値は 1 に設定されます。必要に応じてこの値を変更できます。

AgentTraceLevel

アプリケーション固有のコンポーネントのトレース・メッセージのレベルです。デフォルト値は 0 です。コネクターは、設定されたトレース・レベル以下の該当するトレース・メッセージをすべてデリバリーします。

ApplicationName

コネクターのアプリケーションを一意的に特定する名前です。この名前は、システム管理者が WebSphere Business Integration システム環境をモニターするために使用されます。コネクターを実行する前に、このプロパティーに値を指定する必要があります。

BrokerType

使用する統合ブローカー・タイプを指定します。オプションは ICS、WebSphere Message Brokers (WMQI、WMQIB または WBIMB) または WAS です。

CharacterEncoding

文字 (アルファベットの文字、数値表現、句読記号など) から数値へのマッピングに使用する文字コード・セットを指定します。

注: Java ベースのコネクターでは、このプロパティーは使用しません。C++ ベースのコネクターでは、現在、このプロパティーに ascii7 という値が使用されています。

デフォルトでは、ドロップダウン・リストには、サポートされる文字エンコードの一部のみが表示されます。ドロップダウン・リストに、サポートされる他の値を追加するには、製品ディレクトリーにある ¥Data¥Std¥stdConnProps.xml ファイルを手動で変更する必要があります。詳細については、本書の Connector Configurator に関する付録を参照してください。

ConcurrentEventTriggeredFlows

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

コネクターがイベントのデリバリー時に並行処理できるビジネス・オブジェクトの数を決定します。この属性の値を、並行してマップおよびデリバリーできるビジネス・オブジェクトの数に設定します。例えば、この属性の値を 5 に設定すると、5 個のビジネス・オブジェクトが並行して処理されます。デフォルト値は 1 です。

このプロパティーを 1 よりも大きい値に設定すると、ソース・アプリケーションのコネクターが、複数のイベント・ビジネス・オブジェクトを同時にマップして、複数のコラボレーション・インスタンスにそれらのビジネス・オブジェクトを同時にデリバリーすることができます。これにより、統合ブローカーへのビジネス・オブジェクトのデリバリーにかかる時間、特にビジネス・オブジェクトが複雑なマップを使用している場合のデリバリー時間が短縮されます。ビジネス・オブジェクトのコラボレーションに到達する速度を増大させると、システム全体のパフォーマンスを向上させることができます。

ソース・アプリケーションから宛先アプリケーションまでのフロー全体に並行処理を実装するには、次のようにする必要があります。

- Maximum number of concurrent events プロパティーの値を増加して、コラボレーションが複数のスレッドを使用できるように構成します。
- 宛先アプリケーションのアプリケーション固有コンポーネントが複数の要求を並行して実行できることを確認します。つまり、このコンポーネントがマルチスレッド化されているか、またはコネクター・エージェント並列処理を使用でき、複数プロセスに対応するよう構成されている必要があります。Parallel Process Degree 構成プロパティーに、1 より大きい値を設定します。

ConcurrentEventTriggeredFlows プロパティーは、順次に実行される単一スレッド処理であるコネクターのポーリングでは無効です。

ContainerManagedEvents

このプロパティーにより、JMS イベント・ストアを使用する JMS 対応コネクターが、保証付きイベント・デリバリーを提供できるようになります。保証付きイベント・デリバリーでは、イベントはソース・キューから除去され、単一 JMS トランザクションとして宛先キューに配置されます。

デフォルト値はありません。

ContainerManagedEvents を JMS に設定した場合には、保証付きイベント・デリバリーを使用できるように次のプロパティーも構成する必要があります。

- PollQuantity = 1 から 500
- SourceQueue = /SOURCEQUEUE

また、MimeType、DHClass (データ・ハンドラー・クラス)、および DataHandlerConfigMOName (オプションのメタオブジェクト名) プロパティーを設定したデータ・ハンドラーも構成する必要があります。これらのプロパティーの値を設定するには、Connector Configurator の**「データ・ハンドラー」**タブを使用します。

これらのプロパティーはアダプター固有ですが、**例**の値は次のようになります。

- MimeType = text¥xml

- DHClass = com.crossworlds.DataHandlers.text.xml
- DataHandlerConfigMOName = MO_DataHandler_Default

「データ・ハンドラー」タブのこれらの値のフィールドは、ContainerManagedEvents を JMS に設定した場合にのみ表示されます。

注: ContainerManagedEvents を JMS に設定した場合、コネクターはその pollForEvents() メソッドを*呼び出*さなくなる ため、そのメソッドの機能は使用できなく なります。

このプロパティーは、DeliveryTransport プロパティーが値 JMS に設定されている場合にのみ表示されます。

ControllerStoreAndForwardMode

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

宛先側のアプリケーション固有のコンポーネントが使用不可であることをコネクター・コントローラーが検出した場合に、コネクター・コントローラーが実行する動作を設定します。

このプロパティーを true に設定した場合、イベントが ICS に到達したときに宛先側のアプリケーション固有のコンポーネントが使用不可であれば、コネクター・コントローラーはそのアプリケーション固有のコンポーネントへの要求をブロックします。アプリケーション固有のコンポーネントが作動可能になると、コネクター・コントローラーはアプリケーション固有のコンポーネントにその要求を転送します。

ただし、コネクター・コントローラーが宛先側のアプリケーション固有のコンポーネントにサービス呼び出し要求を**転送した後で**このコンポーネントが使用不可になった場合、コネクター・コントローラーはその要求を失敗させます。

このプロパティーを false に設定した場合、コネクター・コントローラーは、宛先側のアプリケーション固有のコンポーネントが使用不可であることを検出すると、ただちにすべてのサービス呼び出し要求を失敗させます。

デフォルト値は true です。

ControllerTraceLevel

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

コネクター・コントローラーのトレース・メッセージのレベルです。デフォルト値は 0 です。

DeliveryQueue

DeliveryTransport が JMS の場合のみ適用されます。

コネクターから統合ブローカーへビジネス・オブジェクトが送信されるときに使用されるキューです。

デフォルト値は CONNECTORNAME/DELIVERYQUEUE です。

DeliveryTransport

イベントのデリバリーのためのトランスポート機構を指定します。指定可能な値は、WebSphere MQ の MQ、CORBA IIOP の IDL、Java Messaging Service の JMS です。

- RepositoryDirectory がリモートの場合は、DeliveryTransport プロパティーの指定可能な値は MQ、IDL、または JMS であり、デフォルトは IDL になります。
- RepositoryDirectory がローカル・ディレクトリーの場合は、指定可能な値は JMS のみです。

DeliveryTransport プロパティーに指定されている値が、MQ または IDL である場合、コネクターは、CORBA IIOP を使用してサービス呼び出し要求と管理メッセージを送信します。

WebSphere MQ および IDL

イベントのデリバリー・トランスポートには、IDL ではなく WebSphere MQ を使用してください (1 種類の製品だけを使用する必要がある場合を除きます)。
WebSphere MQ が IDL よりも優れている点は以下のとおりです。

- 非同期 (ASYNC) 通信:
  WebSphere MQ を使用すると、アプリケーション固有のコンポーネントは、サーバーが利用不能である場合でも、イベントをポーリングして永続的に格納することができます。
- サーバー・サイド・パフォーマンス:
  WebSphere MQ を使用すると、サーバー・サイドのパフォーマンスが向上します。最適化モードでは、WebSphere MQ はイベントへのポインターのみをリポジトリー・データベースに格納するので、実際のイベントは WebSphere MQ キュー内に残ります。これにより、サイズが大きい可能性のあるイベントをリポジトリー・データベースに書き込む必要がありません。
- エージェント・サイド・パフォーマンス:
  WebSphere MQ を使用すると、アプリケーション固有のコンポーネント側のパフォーマンスが向上します。WebSphere MQ を使用すると、コネクターのポーリング・スレッドは、イベントを選出した後、コネクターのキューにそのイベントを入れ、次のイベントを選出します。この方法は IDL よりも高速で、IDL の場合、コネクターのポーリング・スレッドは、イベントを選出した後、ネットワーク経由でサーバー・プロセスにアクセスしてそのイベントをリポジトリー・データベースに永続的に格納してから、次のイベントを選出する必要があります。

JMS

Java Messaging Service (JMS) を使用しての、コネクターとクライアント・コネクター・フレームワークとの間の通信を可能にします。

JMS をデリバリー・トランスポートとして選択した場合は、
jms.MessageBrokerName、jms.FactoryClassName、jms.Password、jms.UserName などの追加の JMS プロパティーが Connector Configurator 内に表示されます。このうち最初の 2 つは、このトランスポートの必須プロパティーです。

重要: 以下の環境では、コネクターに JMS トランスポート機構を使用すると、メモリー制限が発生することもあります。

- AIX 5.0
- WebSphere MQ 5.3.0.1
- ICS が統合ブローカーの場合

この環境では、WebSphere MQ クライアント内でメモリーが使用されるため、(サーバー側の) コネクター・コントローラーと (クライアント側の) コネクターの両方を始動するのは困難な場合があります。ご使用のシステムのプロセス・ヒープ・サイズが 768M 未満である場合には、次のように設定することをお勧めします。

- CWSharedEnv.sh スクリプト内で LDR_CNTRL 環境変数を設定する。

  このスクリプトは、製品ディレクトリー配下の ¥bin ディレクトリーにあります。テキスト・エディターを使用して、CWSharedEnv.sh スクリプトの最初の行として次の行を追加します。

  ```
  export LDR_CNTRL=MAXDATA=0x30000000
  ```

  この行は、ヒープ・メモリーの使用量を最大 768 MB (3 セグメント * 256 MB) に制限します。プロセス・メモリーがこの制限値を超えると、ページ・スワッピングが発生し、システムのパフォーマンスに悪影響を与える場合があります。
- IPCCBaseAddress プロパティーの値を 11 または 12 に設定する。このプロパティーの詳細については、「*システム・インストール・ガイド (UNIX 版)*」を参照してください。

DuplicateEventElimination

このプロパティーを true に設定すると、JMS 対応コネクターによるデリバリー・キューへの重複イベントのデリバリーが防止されます。この機能を使用するには、コネクターに対し、アプリケーション固有のコード内でビジネス・オブジェクトの **ObjectEventId** 属性として一意のイベント ID が設定されている必要があります。これはコネクター開発時に設定されます。

このプロパティーは、false に設定することもできます。

注: DuplicateEventElimination を true に設定する際は、MonitorQueue プロパティーを構成して保証付きイベント・デリバリーを使用可能にする必要があります。

FaultQueue

コネクターでメッセージを処理中にエラーが発生すると、コネクターは、そのメッセージを状況表示および問題説明とともにこのプロパティーに指定されているキューに移動します。

デフォルト値は CONNECTORNAME/FAULTQUEUE です。

JvmMaxHeapSize

エージェントの最大ヒープ・サイズ (メガバイト単位)。このプロパティーは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 128M です。

JvmMaxNativeStackSize

エージェントの最大ネイティブ・スタック・サイズ (キロバイト単位)。このプロパティーは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 128K です。

JvmMinHeapSize

エージェントの最小ヒープ・サイズ (メガバイト単位)。このプロパティーは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 1M です。

jms.FactoryClassName

JMS プロバイダーのためにインスタンスを生成するクラス名を指定します。JMS をデリバリー・トランスポート機構 (DeliveryTransport) として選択する際は、このコネクター・プロパティーを必ず 設定してください。

デフォルト値は CxCommon.Messaging.jms.IBMMQSeriesFactory です。

jms.MessageBrokerName

JMS プロバイダーのために使用するブローカー名を指定します。JMS をデリバリー・トランスポート機構 (DeliveryTransport) として選択する際は、このコネクター・プロパティーを必ず 設定してください。

デフォルト値は crossworlds.queue.manager です。ローカル・メッセージ・ブローカーに接続する場合は、デフォルト値を使用します。

リモート・メッセージ・ブローカーに接続すると、このプロパティーは次の (必須) 値をとります。
QueueMgrName:<Channel>:<HostName>:<PortNumber>
各変数の意味は以下のとおりです。
QueueMgrName: キュー・マネージャー名です。
Channel: クライアントが使用するチャネルです。
HostName: キュー・マネージャーの配置先のマシン名です。
PortNumber: キュー・マネージャーが listen に使用するポートの番号です。

例えば、次のように指定します。
jms.MessageBrokerName = WBIMB.Queue.Manager:CHANNEL1:RemoteMachine:1456

jms.NumConcurrentRequests

コネクターに対して同時に送信することができる並行サービス呼び出し要求の数 (最大値) を指定します。この最大値に達した場合、新規のサービス呼び出し要求はブロックされ、既存のいずれかの要求が完了した後で処理されます。

デフォルト値は 10 です。

jms.Password

JMS プロバイダーのためのパスワードを指定します。このプロパティーの値はオプションです。

デフォルトはありません。

jms.UserName

JMS プロバイダーのためのユーザー名を指定します。このプロパティーの値はオプションです。

デフォルトはありません。

ListenerConcurrency

このプロパティーは、統合ブローカーとして ICS を使用する場合の MQ Listener でのマルチスレッド化をサポートしています。このプロパティーにより、データベースへの複数イベントの書き込み操作をバッチ処理できるので、システム・パフォーマンスが向上します。デフォルト値は 1 です。

このプロパティーは、MQ トランスポートを使用するコネクターにのみ適用されます。DeliveryTransport プロパティーには MQ を設定してください。

Locale

言語コード、国または地域、および、希望する場合には、関連した文字コード・セットを指定します。このプロパティーの値は、データの照合やソート順、日付と時刻の形式、通貨記号などの国/地域別情報を決定します。

ロケール名は、次の書式で指定します。

ll_TT.codeset

ここで、

ll	2 文字の言語コード (普通は小文字)
TT	2 文字の国または地域コード (普通は大文字)
codeset	関連文字コード・セットの名前。名前のこの部分は、通常、オプションです。

デフォルトでは、ドロップダウン・リストには、サポートされるロケールの一部のみが表示されます。ドロップダウン・リストに、サポートされる他の値を追加するには、製品ディレクトリーにある ¥Data¥Std¥stdConnProps.xml ファイルを手動で変更する必要があります。詳細については、本書の Connector Configurator に関する付録を参照してください。

デフォルト値は en_US です。コネクターがグローバル化に対応していない場合、このプロパティーの有効な値は en_US のみです。特定のコネクターがグローバル化に対応しているかどうかを判別するには、以下の Web サイトにあるコネクターのバージョン・リストを参照してください。

http://www.ibm.com/software/websphere/wbiadapters/infocenter、または
http://www.ibm.com/websphere/integration/wicserver/infocenter

LogAtInterchangeEnd

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

統合ブローカーのログ宛先にエラーを記録するかどうかを指定します。ブローカーのログ宛先にログを記録すると、電子メール通知もオンになります。これにより、エラーまたは致命的エラーが発生すると、InterchangeSystem.cfg ファイルに指定された MESSAGE_RECIPIENT に対する電子メール・メッセージが生成されます。

例えば、LogAtInterChangeEnd を true に設定した場合にコネクターからアプリケーションへの接続が失われると、指定されたメッセージ宛先に、電子メール・メッセージが送信されます。デフォルト値は false です。

MaxEventCapacity

コントローラー・バッファー内のイベントの最大数。このプロパティーはフロー制御が使用し、RepositoryDirectory プロパティーの値が <REMOTE> の場合にのみ適用されます。

値は 1 から 2147483647 の間の正整数です。デフォルト値は 2147483647 です。

MessageFileName

コネクター・メッセージ・ファイルの名前です。メッセージ・ファイルの標準位置は、製品ディレクトリーの ¥connectors¥messages です。メッセージ・ファイルが標準位置に格納されていない場合は、メッセージ・ファイル名を絶対パスで指定します。

コネクター・メッセージ・ファイルが存在しない場合は、コネクターは InterchangeSystem.txt をメッセージ・ファイルとして使用します。このファイルは、製品ディレクトリーに格納されています。

注: 特定のコネクターについて、コネクター独自のメッセージ・ファイルがあるかどうかを判別するには、該当するアダプターのユーザーズ・ガイドを参照してください。

MonitorQueue

コネクターが重複イベントをモニターするために使用する論理キューです。このプロパティーは、DeliveryTransport プロパティー値が JMS であり、かつ DuplicateEventElimination が TRUE に設定されている場合にのみ使用されます。

デフォルト値は CONNECTORNAME/MONITORQUEUE です。

OADAutoRestartAgent

RepositoryDirectory が <REMOTE> の場合のみ有効です。

コネクターが自動再始動およびリモート再始動機能を使用するかどうかを指定します。この機能では、MQ により起動される Object Activation Daemon (OAD) を使用して、異常シャットダウン後にコネクターを再始動したり、System Monitor からリモート・コネクターを始動したりします。

自動再始動機能およびリモート再始動機能を使用可能にするには、このプロパティーを true に設定する必要があります。MQ によりトリガーされる OAD 機能の構成方法については、「システム・インストール・ガイド *(Windows 版)*」または「システム・インストール・ガイド *(UNIX 版)*」を参照してください。

デフォルト値は false です。

OADMaxNumRetry

RepositoryDirectory が <REMOTE> の場合のみ有効です。

異常シャットダウンの後で MQ によりトリガーされる OAD がコネクターの再始動を自動的に試行する回数の最大数を指定します。このプロパティーを有効にするためには、OADAutoRestartAgent プロパティーを true に設定する必要があります。

デフォルト値は 1000 です。

OADRetryTimeInterval

RepositoryDirectory が <REMOTE> の場合のみ有効です。

MQ によりトリガーされる OAD の再試行時間間隔の分数を指定します。コネクター・エージェントがこの再試行時間間隔内に再始動しない場合は、コネクター・コントローラーはコネクター・エージェントを再び再始動するように OAD に要求します。OAD はこの再試行プロセスを OADMaxNumRetry プロパティーで指定された回数だけ繰り返します。このプロパティーを有効にするためには、OADAutoRestartAgent プロパティーを true に設定する必要があります。

デフォルト値は 10 です。

PollEndTime

イベント・キューのポーリングを停止する時刻です。形式は HH:MM です。ここで、*HH* は 0 から 23 時を表し、*MM* は 0 から 59 分を表します。

このプロパティーには必ず有効な値を指定してください。デフォルト値は HH:MM ですが、この値は必ず変更する必要があります。

PollFrequency

これは、前回のポーリングの終了から次のポーリングの開始までの間の間隔です。PollFrequency は、あるポーリング・アクションの終了から次のポーリング・アクションの開始までの時間をミリ秒単位で指定します。これはポーリング・アクション間の間隔ではありません。この論理を次に説明します。

- ポーリングし、PollQuantity の値により指定される数のオブジェクトを取得します。
- これらのオブジェクトを処理します。一部のアダプターでは、これは個別のスレッドで部分的に実行されます。これにより、次のポーリング・アクションまで処理が非同期に実行されます。
- PollFrequency で指定された間隔にわたって遅延します。
- このサイクルを繰り返します。

PollFrequency は以下の値のいずれかに設定します。

- ポーリング・アクション間のミリ秒数 (整数)。
- ワード key。コネクターは、コネクターのコマンド・プロンプト・ウィンドウで文字 p が入力されたときにのみポーリングを実行します。このワードは小文字で入力します。
- ワード no。コネクターはポーリングを実行しません。このワードは小文字で入力します。

デフォルト値は 10000 です。

重要: 一部のコネクターでは、このプロパティーの使用が制限されています。このようなコネクターが存在する場合には、アダプターのインストールと構成に関する章で制約事項が説明されています。

PollQuantity

コネクターがアプリケーションからポーリングする項目の数を指定します。アダプターにコネクター固有のポーリング数設定プロパティーがある場合、標準プロパティーの値は、このコネクター固有のプロパティーの設定値によりオーバーライドされます。

電子メール・メッセージもイベントと見なされます。コネクターは、電子メールに関するポーリングを受けたときには次のように動作します。

コネクターは、1 回目のポーリングを受けると、メッセージの本文を選出します。これは、本文が添付とも見なされるからです。本文の MIME タイプにはデータ・ハンドラーが指定されていないので、コネクターは本文を無視します。
コネクターは PO の最初の添付を処理します。この添付の MIME タイプには対応する DH があるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。
2 回目のポーリングを受けると、コネクターは PO の 2 番目の添付を処理します。この添付の MIME タイプには対応する DH があるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。
これが受け入れられると、PO の 3 番目の添付が届きます。

PollStartTime

イベント・キューのポーリングを開始する時刻です。形式は *HH:MM* です。ここで、*HH* は 0 から 23 時を表し、*MM* は 0 から 59 分を表します。

このプロパティーには必ず有効な値を指定してください。デフォルト値は HH:MM ですが、この値は必ず変更する必要があります。

RequestQueue

統合ブローカーが、ビジネス・オブジェクトをコネクターに送信するときに使用されるキューです。

デフォルト値は CONNECTOR/REQUESTQUEUE です。

RepositoryDirectory

コネクターが XML スキーマ文書を読み取るリポジトリーの場所です。この XML スキーマ文書には、ビジネス・オブジェクト定義のメタデータが含まれています。

統合ブローカーが ICS の場合はこの値を <REMOTE> に設定する必要があります。これは、コネクターが InterChange Server リポジトリーからこの情報を取得するためです。

統合ブローカーが WebSphere Message Broker または WAS の場合は、この値を <*local directory*> に設定する必要があります。

ResponseQueue

`DeliveryTransport` が JMS の場合のみ適用可能で、`RepositoryDirectory` が <REMOTE> の場合のみ必須です。

JMS 応答キューを指定します。JMS 応答キューは、応答メッセージをコネクター・フレームワークから統合ブローカーへデリバリーします。統合ブローカーが ICS の場合、サーバーは要求を送信し、JMS 応答キューの応答メッセージを待ちます。

RestartRetryCount

コネクターによるコネクター自体の再始動の試行回数を指定します。このプロパティーを並列コネクターに対して使用する場合、コネクターのマスター側のアプリケーション固有のコンポーネントがスレーブ側のアプリケーション固有のコンポーネントの再始動を試行する回数が指定されます。

デフォルト値は `3` です。

RestartRetryInterval

コネクターによるコネクター自体の再始動の試行間隔を分単位で指定します。このプロパティーを並列コネクターに対して使用する場合、コネクターのマスター側のアプリケーション固有のコンポーネントがスレーブ側のアプリケーション固有のコンポーネントの再始動を試行する間隔が指定されます。指定可能な値の範囲は 1 から 2147483647 です。

デフォルト値は `1` です。

RHF2MessageDomain

WebSphere Message Brokers および WAS でのみ使用されます。

このプロパティーにより、JMS ヘッダーのドメイン名フィールドの値を構成できます。JMS トランスポートを介してデータを WMQI に送信するときに、アダプター・フレームワークにより JMS ヘッダー情報、ドメイン名、および固定値 `mrm` が書き込まれます。この構成可能なドメイン名により、ユーザーは WMQI ブローカーによるメッセージ・データの処理方法を追跡できます。

サンプル・ヘッダーを以下に示します。

```
<mcd><Msd>mrm</Msd><Set>3</Set><Type>
Retek_POPhyDesc</Type><Fmt>CwXML</Fmt></mcd>
```

デフォルト値は mrm ですが、このプロパティーには xml も設定できます。このプロパティーは、DeliveryTransport が JMS に設定されており、かつ WireFormat が CwXML に設定されている場合にだけ表示されます。

SourceQueue

DeliveryTransport が JMS で、ContainerManagedEvents が指定されている場合のみ適用されます。

JMS イベント・ストアを使用する JMS 対応コネクターでの保証付きイベント・デリバリーをサポートするコネクター・フレームワークに、JMS ソース・キューを指定します。詳細については、105 ページの『ContainerManagedEvents』を参照してください。

デフォルト値は CONNECTOR/SOURCEQUEUE です。

SynchronousRequestQueue

DeliveryTransport が JMS の場合のみ適用されます。

同期応答を要求する要求メッセージを、コネクター・フレームワークからブローカーに配信します。このキューは、コネクターが同期実行を使用する場合にのみ必要です。同期実行の場合、コネクター・フレームワークは、SynchronousRequestQueue にメッセージを送信し、SynchronousResponseQueue でブローカーから戻される応答を待機します。コネクターに送信される応答メッセージには、元のメッセージの ID を指定する相関 ID が含まれています。

デフォルトは CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE です。

SynchronousResponseQueue

DeliveryTransport が JMS の場合のみ適用されます。

同期要求に対する応答として送信される応答メッセージを、ブローカーからコネクター・フレームワークに配信します。このキューは、コネクターが同期実行を使用する場合にのみ必要です。

デフォルトは CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE です。

SynchronousRequestTimeout

DeliveryTransport が JMS の場合のみ適用されます。

コネクターが同期要求への応答を待機する時間を分単位で指定します。コネクターは、指定された時間内に応答を受信できなかった場合、元の同期要求メッセージをエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

WireFormat

トランスポートのメッセージ・フォーマットです。

- RepositoryDirectory がローカル・ディレクトリーの場合は、設定は CwXML になります。
- RepositoryDirectory の値が <REMOTE> の場合には、設定値は CwBO です。

WsifSynchronousRequestTimeout

WAS 統合ブローカーでのみ使用されます。

コネクターが同期要求への応答を待機する時間を分単位で指定します。コネクターは、指定された時間内に応答を受信できなかった場合、元の同期要求メッセージをエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

XMLNameSpaceFormat

WebSphere Message Brokers および WAS 統合ブローカーでのみ使用されます。

ビジネス・オブジェクト定義の XML 形式でネーム・スペースを short と long のどちらにするかをユーザーが指定できるようにするための、強力なプロパティーです。

デフォルト値は short です。

付録 B. Connector Configurator

この付録では、Connector Configurator を使用してアダプターの構成プロパティー値を設定する方法について説明します。

Connector Configurator を使用して次の作業を行います。

- コネクターを構成するためのコネクター固有のプロパティー・テンプレートを作成する
- 構成ファイルを作成する
- 構成ファイル内のプロパティーを設定する

注:

本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムを使用している場合は、円記号をスラッシュ (/) に置き換えてください。また、各オペレーティング・システムの規則に従ってください。

この付録では、次のトピックについて説明します。

Connector Configurator の概要

Connector Configurator では、次の統合ブローカーで使用するアダプターのコネクター・コンポーネントを構成できます。

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (WebSphere Message Brokers (WMQI) と総称)
- WebSphere Application Server (WAS)

Connector Configurator を使用して次の作業を行います。

- コネクターを構成するための**コネクター固有のプロパティー・テンプレート**を作成する。
- **コネクター構成ファイル**を作成する。インストールするコネクターごとに構成ファイルを 1 つ作成する必要があります。
- 構成ファイル内のプロパティーを設定する。
  場合によっては、コネクター・テンプレートでプロパティーに対して設定されているデフォルト値を変更する必要があります。また、サポートされるビジネス・オブジェクト定義と、ICS の場合はコラボレーションとともに使用するマップを

指定し、必要に応じてメッセージング、ロギング、トレース、およびデータ・ハンドラー・パラメーターを指定する必要があります。

Connector Configurator の実行モードと使用する構成ファイルのタイプは、実行する統合ブローカーによって異なります。例えば、使用している統合ブローカーが WMQI の場合、Connector Configurator を System Manager から実行するのではなく、直接実行します (『スタンドアロン・モードでの Configurator の実行』を参照)。

コネクター構成プロパティーには、標準の構成プロパティー (すべてのコネクターがもつプロパティー) と、コネクター固有のプロパティー (特定のアプリケーションまたはテクノロジーのためにコネクターで必要なプロパティー) とが含まれます。

標準プロパティーはすべてのコネクターにより使用されるので、標準プロパティーを新規に定義する必要はありません。ファイルを作成すると、Connector Configurator により標準プロパティーがこの構成ファイルに挿入されます。ただし、Connector Configurator で各標準プロパティーの値を設定する必要があります。

標準プロパティーの範囲は、ブローカーと構成によって異なる可能性があります。特定のプロパティーに特定の値が設定されている場合にのみ使用できるプロパティーがあります。Connector Configurator の「標準のプロパティー」ウィンドウには、特定の構成で設定可能なプロパティーが表示されます。

ただし**コネクター固有プロパティー**の場合は、最初にプロパティーを定義し、その値を設定する必要があります。このため、特定のアダプターのコネクター固有プロパティーのテンプレートを作成します。システム内で既にテンプレートが作成されている場合には、作成されているテンプレートを使用します。システム内でまだテンプレートが作成されていない場合には、120 ページの『新規テンプレートの作成』のステップに従い、テンプレートを新規に作成します。

注: Connector Configurator は、Windows 環境内でのみ実行されます。UNIX 環境でコネクターを実行する場合には、Windows で Connector Configurator を使用して構成ファイルを変更し、このファイルを UNIX 環境へコピーします。

Connector Configurator の始動

以下の 2 種類のモードで Connector Configurator を開始および実行できます。

- スタンドアロン・モードで個別に実行
- System Manager から

スタンドアロン・モードでの Configurator の実行

どのブローカーを実行している場合にも、Connector Configurator を個別に実行し、コネクター構成ファイルを編集できます。

これを行うには、以下のステップを実行します。

- **「スタート」>「プログラム」**から、**「IBM WebSphere InterChange Server」>「IBM WebSphere Business Integration Tools」>「Connector Configurator」**をクリックします。
- **「ファイル」>「新規」>「コネクター構成」**を選択します。

- **「システム接続: Integration Broker」**の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、または WAS を選択します。

Connector Configurator を個別に実行して構成ファイルを生成してから、System Manager に接続してこの構成ファイルを System Manager プロジェクトに保存することもできます (125 ページの『構成ファイルの完成』を参照)。

System Manager からの Configurator の実行

System Manager から Connector Configurator を実行できます。

Connector Configurator を実行するには、以下のステップを実行します。

1. System Manager を開きます。
2. 「System Manager」ウィンドウで、**「統合コンポーネント・ライブラリー」**アイコンを展開し、**「コネクター」**を強調表示します。
3. System Manager メニュー・バーから、**「ツール」>「Connector Configurator」**をクリックします。「Connector Configurator」ウィンドウが開き、**「新規コネクター」**ダイアログ・ボックスが表示されます。
4. **「システム接続: Integration Broker」**の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、または WAS を選択します。

既存の構成ファイルを編集するには、以下のステップを実行します。

- 「System Manager」ウィンドウの「コネクター」フォルダーでいずれかの構成ファイルを選択し、右クリックします。Connector Configurator が開き、この構成ファイルの統合ブローカー・タイプおよびファイル名が上部に表示されます。
- Connector Configurator で**「ファイル」>「開く」**を選択します。プロジェクトまたはプロジェクトが保管されているディレクトリーからコネクター構成ファイルを選択します。
- 「標準のプロパティー」タブをクリックし、この構成ファイルに含まれているプロパティーを確認します。

コネクター固有のプロパティー・テンプレートの作成

コネクターの構成ファイルを作成するには、コネクター固有プロパティーのテンプレートとシステム提供の標準プロパティーが必要です。

コネクター固有プロパティーのテンプレートを新規に作成するか、または既存のコネクター定義をテンプレートとして使用します。

- テンプレートの新規作成については、120 ページの『新規テンプレートの作成』を参照してください。
- 既存のファイルを使用する場合には、既存のテンプレートを変更し、新しい名前でこのテンプレートを保管します。既存のテンプレートは ¥WebSphereAdapters¥bin¥Data¥App ディレクトリーにあります。

新規テンプレートの作成

このセクションでは、テンプレートでプロパティーを作成し、プロパティーの一般特性および値を定義し、プロパティー間の依存関係を指定する方法について説明します。次にそのテンプレートを保管し、新規コネクター構成ファイルを作成するためのベースとして使用します。

Connector Configurator でテンプレートを作成するには、以下のステップを実行します。

1. **「ファイル」>「新規」>「コネクター固有プロパティー・テンプレート」**をクリックします。
2. **「コネクター固有プロパティー・テンプレート」** ダイアログ・ボックスが表示されます。
   - **「新規テンプレート名を入力してください」**の下の**「名前」**フィールドに、新規テンプレートの名前を入力します。テンプレートから新規構成ファイルを作成するためのダイアログ・ボックスを開くと、この名前が再度表示されます。
   - テンプレートに含まれているコネクター固有のプロパティー定義を調べるには、**「テンプレート名」**表示でそのテンプレートの名前を選択します。そのテンプレートに含まれているプロパティー定義のリストが**「テンプレートのプレビュー」**表示に表示されます。
3. テンプレートを作成するときには、ご使用のコネクターに必要なプロパティー定義に類似したプロパティー定義が含まれている既存のテンプレートを使用できます。ご使用のコネクターで使用するコネクター固有のプロパティーが表示されるテンプレートが見つからない場合は、自分で作成する必要があります。
   - 既存のテンプレートを変更する場合には、**「変更する既存のテンプレートを選択してください: 検索テンプレート」**の下の**「テンプレート名」**テーブルのリストから、テンプレート名を選択します。
   - このテーブルには、現在使用可能なすべてのテンプレートの名前が表示されます。テンプレートを検索することもできます。

一般特性の指定

「次へ」をクリックしてテンプレートを選択すると、**「プロパティー: コネクター固有プロパティー・テンプレート」**ダイアログ・ボックスが表示されます。このダイアログ・ボックスには、定義済みプロパティーの「一般」特性のタブと「値」の制限のタブがあります。「一般」表示には以下のフィールドがあります。

- **一般:**
  プロパティー・タイプ
  更新されたメソッド
  説明
- **フラグ**
  標準フラグ
- **カスタム・フラグ**
  フラグ

プロパティーの一般特性の選択を終えたら、**「値」**タブをクリックします。

値の指定

「値」タブを使用すると、プロパティーの最大長、最大複数値、デフォルト値、または値の範囲を設定できます。編集可能な値も許可されます。これを行うには、以下のステップを実行します。

1. **「値」**タブをクリックします。「一般」のパネルに代わって「値」の表示パネルが表示されます。
2. **「プロパティーを編集」**表示でプロパティーの名前を選択します。
3. **「最大長」**および**「最大複数値」**のフィールドに値を入力します。

新規プロパティー値を作成するには、以下のステップを実行します。

1. **「プロパティーを編集」**リストでプロパティーを選択し、右マウス・ボタンでクリックします。
2. ダイアログ・ボックスから**「追加」**を選択します。
3. 新規プロパティー値の名前を入力し、「OK」をクリックします。右側の**「値」**パネルに値が表示されます。

「値」パネルには、3 つの列からなるテーブルが表示されます。

「値」の列には、**「プロパティー値」**ダイアログ・ボックスで入力した値と、以前に作成した値が表示されます。

「デフォルト値」の列では、値のいずれかをデフォルトとして指定することができます。

「値の範囲」の列には、**「プロパティー値」**ダイアログ・ボックスで入力した範囲が表示されます。

値が作成されて、グリッドに表示されると、そのテーブルの表示内から編集できるようになります。

テーブルにある既存の値の変更を行うには、その行の行番号をクリックして行全体を選択します。次に**「値」**フィールドを右マウス・ボタンでクリックし、**「値の編集 (Edit Value)」**をクリックします。

依存関係の設定

「一般」タブと**「値」**タブで変更を行ったら、**「次へ」**をクリックします。**「依存関係: コネクター固有プロパティー・テンプレート」**ダイアログ・ボックスが表示されます。

依存プロパティーは、別のプロパティーの値が特定の条件に合致する場合にのみ、テンプレートに組み込まれて、構成ファイルで使用されるプロパティーです。例えば、テンプレートに `PollQuantity` が表示されるのは、トランスポート機構が JMS であり、`DuplicateEventElimination` が `True` に設定されている場合のみです。プロパティーを依存プロパティーとして指定し、依存する条件を設定するには、以下のステップを実行します。

1. **「使用可能なプロパティー」**表示で、依存プロパティーとして指定するプロパティーを選択します。

2. **「プロパティーを選択」**フィールドで、ドロップダウン・メニューを使用して、条件値を持たせるプロパティーを選択します。
3. **「条件演算子」**フィールドで以下のいずれかを選択します。

   == (等しい)

   != (等しくない)

   > (より大)

   < (より小)

   >= (より大か等しい)

   <= (より小か等しい)
4. **「条件値」**フィールドで、依存プロパティーをテンプレートに組み込むために必要な値を入力します。
5. **「使用可能なプロパティー」**表示で依存プロパティーを強調表示させて矢印をクリックし、**「依存プロパティー」**表示に移動させます。
6. **「完了」**をクリックします。Connector Configurator により、XML 文書として入力した情報が、Connector Configurator がインストールされている `¥bin` ディレクトリーの `¥data¥app` の下に保管されます。

新規構成ファイルの作成

構成ファイルを新規に作成するには、構成ファイルの名前を指定し、統合ブローカーを選択する必要があります。

- 「System Manager」ウィンドウで**「コネクター」**フォルダーを右クリックし、**「新規コネクターの作成」**を選択します。Connector Configurator が開き、**「新規コネクター」**ダイアログ・ボックスが表示されます。
- スタンドアロン・モードの場合は、Connector Configurator で**「ファイル」>「新規」>「コネクター構成」**を選択します。「新規コネクター」ウィンドウで、新規コネクターの名前を入力します。

また、統合ブローカーも選択する必要があります。選択したブローカーによって、構成ファイルに記述されるプロパティーが決まります。ブローカーを選択するには、以下のステップを実行します。

- **「Integration Broker」**フィールドで、ICS 接続、WebSphere Message Brokers 接続、WAS 接続のいずれかを選択します。
- この章で後述する説明に従って**「新規コネクター」**ウィンドウの残りのフィールドに入力します。

コネクター固有のテンプレートからの構成ファイルの作成

コネクター固有のテンプレートを作成すると、テンプレートを使用して構成ファイルを作成できます。

1. **「ファイル」>「新規」>「コネクター構成」**をクリックします。
2. 以下のフィールドを含む**「新規コネクター」**ダイアログ・ボックス表示されます。

- **名前**

  コネクターの名前を入力します。名前では大文字と小文字が区別されます。入力する名前は、システムにインストールされているコネクターのファイル名に対応した一意の名前でなければなりません。

  重要: Connector Configurator では、入力された名前のスペルはチェックされません。名前が正しいことを確認してください。

- **システム接続**

  ICS 接続、WebSphere Message Brokers 接続、WAS のいずれかをクリックします。

- **「コネクター固有プロパティー・テンプレート」を選択します。**

  ご使用のコネクター用に設計したテンプレートの名前を入力します。**「テンプレート名」**表示に、使用可能なテンプレートが表示されます。「テンプレート名」表示で名前を選択すると、**「プロパティー・テンプレートのプレビュー」**表示に、そのテンプレートで定義されているコネクター固有のプロパティーが表示されます。

  使用するテンプレートを選択し、**「OK」**をクリックします。

3. 構成しているコネクターの構成画面が表示されます。タイトル・バーに統合ブローカーとコネクターの名前が表示されます。ここですべてのフィールドに値を入力して定義を完了するか、ファイルを保管して後でフィールドに値を入力するかを選択できます。
4. ファイルを保管するには、**「ファイル」>「保管」>「ファイルに」**をクリックするか、**「ファイル」>「保管」>「プロジェクトに」**をクリックします。プロジェクトに保管するには、System Manager が実行中でなければなりません。
   ファイルとして保管する場合は、**「ファイル・コネクターを保管」**ダイアログ・ボックスが表示されます。`*.cfg` をファイル・タイプとして選択し、「ファイル名」フィールド内に名前が正しいスペル (大文字と小文字の区別を含む) で表示されていることを確認してから、ファイルを保管するディレクトリーにナビゲートし、**「保管」**をクリックします。Connector Configurator のメッセージ・パネルの状況表示に、構成ファイルが正常に作成されたことが示されます。

   重要: ここで設定するディレクトリー・パスおよび名前は、コネクターの始動ファイルで指定するコネクター構成ファイルのパスおよび名前に一致している必要があります。
5. この章で後述する手順に従って、「Connector Configurator」ウィンドウの各タブにあるフィールドに値を入力し、コネクター定義を完了します。

既存ファイルの使用

使用可能な既存ファイルは、以下の 1 つまたは複数の形式になります。

- コネクター定義ファイル。
  コネクター定義ファイルは、特定のコネクターのプロパティーと、適用可能なデフォルト値がリストされたテキスト・ファイルです。コネクターの配布パッケー

ジの ¥repository ディレクトリー内には、このようなファイルが格納されていることがあります (通常、このファイルの拡張子は .txt です。例えば、XML コネクターの場合は CN_XML.txt です)。

- ICS リポジトリー・ファイル。
  コネクターの以前の ICS インプリメンテーションで使用した定義は、そのコネクターの構成で使用されたリポジトリー・ファイルで使用可能になります。そのようなファイルの拡張子は、通常 .in または .out です。
- コネクターの以前の構成ファイル。
  これらのファイルの拡張子は、通常 *.cfg です。

これらのいずれのファイル・ソースにも、コネクターのコネクター固有プロパティーのほとんど、あるいはすべてが含まれますが、この章内の後で説明するように、コネクター構成ファイルは、ファイルを開いて、プロパティーを設定しない限り完成しません。

既存ファイルを使用してコネクターを構成するには、Connector Configurator でそのファイルを開き、構成を修正し、そのファイルを再度保管する必要があります。

以下のステップを実行して、ディレクトリーから *.txt、*.cfg、または *.in ファイルを開きます。

1. Connector Configurator 内で、**「ファイル」>「開く」>「ファイルから」**をクリックします。
2. **「ファイル・コネクターを開く」**ダイアログ・ボックス内で、以下のいずれかのファイル・タイプを選択して、使用可能なファイルを調べます。
   - 構成 (*.cfg)
   - ICS リポジトリー (*.in、*.out)

     ICS 環境でのコネクターの構成にリポジトリー・ファイルが使用された場合には、このオプションを選択します。リポジトリー・ファイルに複数のコネクター定義が含まれている場合は、ファイルを開くとすべての定義が表示されます。
   - すべてのファイル (*.*)

     コネクターのアダプター・パッケージに *.txt ファイルが付属していた場合、または別の拡張子で定義ファイルが使用可能である場合は、このオプションを選択します。
3. ディレクトリー表示内で、適切なコネクター定義ファイルへ移動し、ファイルを選択し、**「開く」**をクリックします。

System Manager プロジェクトからコネクター構成を開くには、以下のステップを実行します。

1. System Manager を始動します。System Manager が開始されている場合にのみ、構成を System Manager から開いたり、System Manager に保管したりできます。
2. Connector Configurator を始動します。
3. **「ファイル」>「開く」>「プロジェクトから」**をクリックします。

構成ファイルの完成

構成ファイルを開くか、プロジェクトからコネクターを開くと、「Connector Configurator」ウィンドウに構成画面が表示されます。この画面には、現在の属性と値が表示されます。

構成画面のタイトルには、ファイル内で指定された統合ブローカーとコネクターの名前が表示されます。正しいブローカーが設定されていることを確認してください。正しいブローカーが設定されていない場合、コネクターを構成する前にブローカー値を変更してください。これを行うには、以下のステップを実行します。

1. **「標準のプロパティー」**タブで、BrokerType プロパティーの値フィールドを選択します。ドロップダウン・メニューで、値 ICS、WMQI、または WAS を選択します。
2. 選択したブローカーに関連付けられているプロパティーが「標準のプロパティー」タブに表示されます。ここでファイルを保管するか、または 128 ページの『サポートされるビジネス・オブジェクト定義の指定』の説明に従い残りの構成フィールドに値を入力することができます。
3. 構成が完了したら、**「ファイル」>「保管」>「プロジェクトに」**を選択するか、または**「ファイル」>「保管」>「ファイルに」**を選択します。

   ファイルに保管する場合は、*.cfg を拡張子として選択し、ファイルの正しい格納場所を選択して、**「保管」**をクリックします。

   複数のコネクター構成を開いている場合、構成をすべてファイルに保管するには**「すべてファイルに保管」**を選択し、コネクター構成をすべて System Manager プロジェクトに保管するには**「すべてプロジェクトに保管」**をクリックします。

   Connector Configurator では、ファイルを保管する前に、必須の標準プロパティーすべてに値が設定されているかどうかが確認されます。必須の標準プロパティーに値が設定されていない場合、Connector Configurator は、検証が失敗したというメッセージを表示します。構成ファイルを保管するには、そのプロパティーの値を指定する必要があります。

構成ファイル・プロパティーの設定

新規のコネクター構成ファイルを作成して名前を付けるとき、または既存のコネクター構成ファイルを開くときには、Connector Configurator によって構成画面が表示されます。構成画面には、必要な構成値のカテゴリーに対応する複数のタブがあります。

Connector Configurator では、すべてのブローカーで実行されているコネクターで、以下のカテゴリーのプロパティーに値が設定されている必要があります。

- 標準のプロパティー
- コネクター固有のプロパティー
- サポートされるビジネス・オブジェクト
- トレース/ログ・ファイルの値
- データ・ハンドラー (保証付きイベント・デリバリーで JMS メッセージングを使用するコネクターの場合に該当する)

注: JMS メッセージングを使用するコネクターの場合は、データをビジネス・オブジェクトに変換するデータ・ハンドラーの構成に関して追加のカテゴリーが表示される場合があります。

ICS で実行されているコネクターの場合、以下のプロパティーの値も設定されている必要があります。

- 関連付けられたマップ
- リソース
- メッセージング (該当する場合)

重要: Connector Configurator では、英語文字セットまたは英語以外の文字セットのいずれのプロパティー値も設定可能です。ただし、標準のプロパティーおよびコネクター固有プロパティー、およびサポートされるビジネス・オブジェクトの名前では、英語文字セットのみを使用する必要があります。

標準プロパティーとコネクター固有プロパティーの違いは、以下のとおりです。

- コネクターの標準プロパティーは、コネクターのアプリケーション固有のコンポーネントとブローカー・コンポーネントの両方によって共用されます。すべてのコネクターが同じ標準プロパティーのセットを使用します。これらのプロパティーの説明は、各アダプター・ガイドの付録 A にあります。変更できるのはこれらの値の一部のみです。
- アプリケーション固有のプロパティーは、コネクターのアプリケーション固有コンポーネント (アプリケーションと直接対話するコンポーネント) のみに適用されます。各コネクターには、そのコネクターのアプリケーションだけで使用されるアプリケーション固有のプロパティーがあります。これらのプロパティーには、デフォルト値が用意されているものもあれば、そうでないものもあります。また、一部のデフォルト値は変更することができます。各アダプター・ガイドのインストールおよび構成の章に、アプリケーション固有のプロパティーおよび推奨値が記述されています。

「標準プロパティー」と**「コネクター固有プロパティー」**のフィールドは、どのフィールドが構成可能であるかを示すために色分けされています。

- 背景がグレーのフィールドは、標準のプロパティーを表します。値を変更することはできますが、名前の変更およびプロパティーの除去はできません。
- 背景が白のフィールドは、アプリケーション固有のプロパティーを表します。これらのプロパティーは、アプリケーションまたはコネクターの特定のニーズによって異なります。値の変更も、これらのプロパティーの除去も可能です。
- 「値」フィールドは構成できます。
- プロパティーごとに**「更新メソッド」**フィールドが表示されます。これは、変更された値をアクティブにするためにコンポーネントまたはエージェントの再始動が必要かどうかを示します。この設定を構成することはできません。

標準コネクター・プロパティーの設定

標準のプロパティーの値を変更するには、以下の手順を実行します。

1. 値を設定するフィールド内でクリックします。

2. 値を入力するか、ドロップダウン・メニューが表示された場合にはメニューから値を選択します。
3. 標準のプロパティーの値をすべて入力後、以下のいずれかを実行することができます。
   - 変更内容を破棄し、元の値を保持したままで Connector Configurator を終了するには、**「ファイル」>「終了」**をクリックし (またはウィンドウを閉じ)、変更内容を保管するかどうかを確認するプロンプトが出されたら**「いいえ」**をクリックします。
   - Connector Configurator 内の他のカテゴリーの値を入力するには、そのカテゴリーのタブを選択します。**「標準のプロパティー」**(またはその他のカテゴリー) で入力した値は、次のカテゴリーに移動しても保持されます。ウィンドウを閉じると、すべてのカテゴリーで入力した値を一括して保管するかまたは破棄するかを確認するプロンプトが出されます。
   - 修正した値を保管するには、**「ファイル」>「終了」**をクリックし (またはウィンドウを閉じ)、変更内容を保管するかどうかを確認するプロンプトが出されたら**「はい」**をクリックします。「ファイル」メニューまたはツールバーから**「保管」>「ファイルに」**をクリックする方法もあります。

アプリケーション固有の構成プロパティーの設定

アプリケーション固有の構成プロパティーの場合、プロパティー名の追加または変更、値の構成、プロパティーの削除、およびプロパティーの暗号化が可能です。プロパティーのデフォルトの長さは 255 文字です。

1. グリッドの左上端の部分で右マウス・ボタンをクリックします。ポップアップ・メニュー・バーが表示されます。プロパティーを追加するときは**「追加」**をクリックします。子プロパティーを追加するには、親の行番号で右マウス・ボタンをクリックし、**「子を追加」**をクリックします。
2. プロパティーまたは子プロパティーの値を入力します。
3. プロパティーを暗号化するには、**「暗号化」**ボックスを選択します。
4. 126 ページの『標準コネクター・プロパティーの設定』の説明に従い、変更内容を保管するかまたは破棄するかを選択します。

各プロパティーごとに表示される「更新メソッド」は、変更された値をアクティブにするためにコンポーネントまたはエージェントの再始動が必要かどうかを示します。

重要: 事前設定のアプリケーション固有のコネクター・プロパティー名を変更すると、コネクターに障害が発生する可能性があります。コネクターをアプリケーションに接続したり正常に実行したりするために、特定のプロパティー名が必要である場合があります。

コネクター・プロパティーの暗号化

「コネクター固有プロパティー」ウィンドウの**「暗号化」**チェック・ボックスにチェックマークを付けると、アプリケーション固有のプロパティーを暗号化することができます。値の暗号化を解除するには、**「暗号化」**チェック・ボックスをクリックしてチェックマークを外し、**「検証」**ダイアログ・ボックスに正しい値を入力し、**「OK」**をクリックします。入力された値が正しい場合は、暗号化解除された値が表示されます。

各プロパティーとそのデフォルト値のリストおよび説明は、各コネクターのアダプター・ユーザーズ・ガイドにあります。

プロパティーに複数の値がある場合には、プロパティーの最初の値に**「暗号化」**チェック・ボックスが表示されます。**「暗号化」**を選択すると、そのプロパティーのすべての値が暗号化されます。プロパティーの複数の値を暗号化解除するには、そのプロパティーの最初の値の**「暗号化」**チェック・ボックスをクリックしてチェックマークを外してから、**「検証」**ダイアログ・ボックスで新規の値を入力します。入力値が一致すれば、すべての複数値が暗号化解除されます。

更新メソッド

付録 A『*コネクターの標準構成プロパティー*』の 98 ページの『プロパティー値の設定と更新』にある更新メソッドの説明を参照してください。

サポートされるビジネス・オブジェクト定義の指定

コネクターで使用するビジネス・オブジェクトを指定するには、Connector Configurator の**「サポートされているビジネス・オブジェクト」**タブを使用します。汎用ビジネス・オブジェクトと、アプリケーション固有のビジネス・オブジェクトの両方を指定する必要があり、またそれらのビジネス・オブジェクト間のマップの関連を指定することが必要です。

注: コネクターによっては、アプリケーションでイベント通知や (メタオブジェクトを使用した) 追加の構成を実行するために、特定のビジネス・オブジェクトをサポートされているものとして指定することが必要な場合もあります。詳細は、「*コネクター開発ガイド (C++ 用)*」または「*コネクター開発ガイド (Java 用)*」を参照してください。

ご使用のブローカーが ICS の場合

ビジネス・オブジェクト定義がコネクターでサポートされることを指定する場合や、既存のビジネス・オブジェクト定義のサポート設定を変更する場合は、**「サポートされているビジネス・オブジェクト」**タブをクリックし、以下のフィールドを使用してください。

ビジネス・オブジェクト名: ビジネス・オブジェクト定義がコネクターによってサポートされることを指定するには、System Manager を実行し、以下の手順を実行します。

1. **「ビジネス・オブジェクト名」**リストで空のフィールドをクリックします。System Manager プロジェクトに存在するすべてのビジネス・オブジェクト定義を示すドロップダウン・リストが表示されます。
2. 追加するビジネス・オブジェクトをクリックします。
3. ビジネス・オブジェクトの**「エージェント・サポート」**(以下で説明) を設定します。
4. 「Connector Configurator」ウィンドウの「ファイル」メニューで、**「プロジェクトに保管」**をクリックします。追加したビジネス・オブジェクト定義に指定されたサポートを含む、変更されたコネクター定義が、System Manager の ICL (Integration Component Library) プロジェクトに保管されます。

サポートされるリストからビジネス・オブジェクトを削除する場合は、以下の手順を実行します。

1. ビジネス・オブジェクト・フィールドを選択するため、そのビジネス・オブジェクトの左側の番号をクリックします。
2. 「Connector Configurator」ウィンドウの**「編集」**メニューから、**「行を削除」**をクリックします。リスト表示からビジネス・オブジェクトが除去されます。
3. **「ファイル」**メニューから、**「プロジェクトの保管」**をクリックします。

サポートされるリストからビジネス・オブジェクトを削除すると、コネクター定義が変更され、削除されたビジネス・オブジェクトはコネクターのこのインプリメンテーションで使用不可になります。コネクターのコードに影響したり、そのビジネス・オブジェクト定義そのものが System Manager から削除されることはありません。

エージェント・サポート: ビジネス・オブジェクトがエージェント・サポートを備えている場合、システムは、コネクター・エージェントを介してアプリケーションにデータを配布する際にそのビジネス・オブジェクトの使用を試みます。

一般に、コネクターのアプリケーション固有ビジネス・オブジェクトは、そのコネクターのエージェントによってサポートされますが、汎用ビジネス・オブジェクトはサポートされません。

ビジネス・オブジェクトがコネクター・エージェントによってサポートされるよう指定するには、**「エージェント・サポート」**ボックスにチェックマークを付けます。「Connector Configurator」ウィンドウでは「エージェント・サポート」の選択の妥当性は検査されません。

最大トランザクション・レベル: コネクターの最大トランザクション・レベルは、そのコネクターがサポートする最大のトランザクション・レベルです。

ほとんどのコネクターの場合、選択可能な項目は「**最大限の努力**」のみです。

トランザクション・レベルの変更を有効にするには、サーバーを再始動する必要があります。

ご使用のブローカーが WebSphere Message Broker の場合

スタンドアロン・モードで作業している (System Manager に接続していない) 場合、手動でビジネス・オブジェクト名を入力する必要があります。

System Manager を実行している場合、**「サポートされているビジネス・オブジェクト」**タブの**「ビジネス・オブジェクト名」**列の下にある空のボックスを選択できます。コンボ・ボックスが表示され、コネクターが属する統合コンポーネント・ライブラリー・プロジェクトから選択可能なビジネス・オブジェクトのリストが示されます。リストから必要なビジネス・オブジェクトを選択します。

「メッセージ・セット ID」は、WebSphere Business Integration Message Broker 5.0 のオプションのフィールドです。この ID が提供される場合、一意である必要はありません。ただし、WebSphere MQ Integrator および Integrator Broker 2.1 の場合は、一意の **ID** を提供する必要があります。

ご使用のブローカーが WAS の場合

使用するブローカー・タイプとして WebSphere Application Server を選択した場合、Connector Configurator にメッセージ・セット ID は必要ありません。**「サポートされているビジネス・オブジェクト」**タブには、サポートされるビジネス・オブジェクトの**「ビジネス・オブジェクト名」**列のみが表示されます。

スタンドアロン・モードで作業している (System Manager に接続していない) 場合、手動でビジネス・オブジェクト名を入力する必要があります。

System Manager を実行している場合、「サポートされているビジネス・オブジェクト」タブの「ビジネス・オブジェクト名」列の下にある空のボックスを選択できます。コンボ・ボックスが表示され、コネクターが属する統合コンポーネント・ライブラリー・プロジェクトから選択可能なビジネス・オブジェクトのリストが示されます。このリストから必要なビジネス・オブジェクトを選択します。

関連付けられているマップ (ICS のみ)

各コネクターは、現在 WebSphere InterChange Server でアクティブなビジネス・オブジェクト定義、およびそれらの関連付けられたマップのリストをサポートします。このリストは、**「関連付けられたマップ」**タブを選択すると表示されます。

ビジネス・オブジェクトのリストには、エージェントでサポートされるアプリケーション固有のビジネス・オブジェクトと、コントローラーがサブスクライブ・コラボレーションに送信する、対応する汎用オブジェクトが含まれます。マップの関連によって、アプリケーション固有のビジネス・オブジェクトを汎用ビジネス・オブジェクトに変換したり、汎用ビジネス・オブジェクトをアプリケーション固有のビジネス・オブジェクトに変換したりするときに、どのマップを使用するかが決定されます。

特定のソースおよび宛先ビジネス・オブジェクトについて一意的に定義されたマップを使用する場合、表示を開くと、マップは常にそれらの該当するビジネス・オブジェクトに関連付けられます。ユーザーがそれらを変更する必要はありません (変更できません)。

サポートされるビジネス・オブジェクトで使用可能なマップが複数ある場合は、そのビジネス・オブジェクトを、使用する必要のあるマップに明示的にバインドすることが必要になります。

「関連付けられたマップ」タブには以下のフィールドが表示されます。

- **ビジネス・オブジェクト名**

  これらは、**「サポートされているビジネス・オブジェクト」**タブで指定した、このコネクターでサポートされるビジネス・オブジェクトです。「サポートされているビジネス・オブジェクト」タブでビジネス・オブジェクトを追加指定した場合、その内容は、「Connector Configurator」ウィンドウの**「ファイル」**メニューから**「プロジェクトに保管」**を選択して、変更を保管した後に、このリストに反映されます。
- **関連付けられたマップ**

この表示には、コネクターの、サポートされるビジネス・オブジェクトでの使用のためにシステムにインストールされたすべてのマップが示されます。各マップのソース・ビジネス・オブジェクトは、**「ビジネス・オブジェクト名」**表示でマップ名の左側に表示されます。

- **明示的**

  場合によっては、関連付けられたマップを明示的にバインドすることが必要になります。

  明示的バインディングが必要なのは、特定のサポートされるビジネス・オブジェクトに複数のマップが存在する場合のみです。ICS は、ブート時、各コネクターでサポートされるそれぞれのビジネス・オブジェクトにマップを自動的にバインドしようとします。複数のマップでその入力データとして同一のビジネス・オブジェクトが使用されている場合、サーバーは、他のマップのスーパーセットである 1 つのマップを見つけて、バインドしようとします。

  他のマップのスーパーセットであるマップがないと、サーバーは、ビジネス・オブジェクトを単一のマップにバインドすることができないため、バインディングを明示的に設定することが必要になります。

  以下の手順を実行して、マップを明示的にバインドします。

  1. **「明示的 (Explicit)」**列で、バインドするマップのチェック・ボックスにチェックマークを付けます。
  2. ビジネス・オブジェクトに関連付けるマップを選択します。
  3. 「Connector Configurator」ウィンドウの**「ファイル」**メニューで、**「プロジェクトに保管」**をクリックします。
  4. プロジェクトを ICS に配置します。
  5. 変更を有効にするため、サーバーをリブートします。

リソース (ICS)

「リソース」タブでは、コネクター・エージェントが、コネクター・エージェント並列処理を使用して同時に複数のプロセスを処理するかどうか、またどの程度処理するかを決定する値を設定できます。

すべてのコネクターがこの機能をサポートしているわけではありません。複数のプロセスを使用するよりも複数のスレッドを使用する方が通常は効率的であるため、Java でマルチスレッドとして設計されたコネクター・エージェントを実行している場合、この機能を使用することはお勧めできません。

メッセージング (ICS)

メッセージング・プロパティーは、`DeliveryTransport` 標準プロパティーの値として MQ を設定し、ブローカー・タイプとして ICS を設定した場合にのみ、使用可能です。これらのプロパティーは、コネクターによるキューの使用方法に影響します。

トレース/ログ・ファイル値の設定

コネクター構成ファイルまたはコネクター定義ファイルを開くと、Connector Configurator は、そのファイルのログおよびトレースの値をデフォルト値として使用します。Connector Configurator 内でこれらの値を変更できます。

ログとトレースの値を変更するには、以下の手順を実行します。

1. **「トレース/ログ・ファイル」**タブをクリックします。
2. ログとトレースのどちらでも、以下のいずれかまたは両方へのメッセージの書き込みを選択できます。
   - コンソールに (STDOUT):
     ログ・メッセージまたはトレース・メッセージを STDOUT ディスプレイに書き込みます。

     注: STDOUT オプションは、Windows プラットフォームで実行しているコネクターの**「トレース/ログ・ファイル」**タブでのみ使用できます。
   - ファイルに:
     ログ・メッセージまたはトレース・メッセージを指定されたファイルに書き込みます。ファイルを指定するには、ディレクトリー・ボタン (省略符号) をクリックし、指定する格納場所に移動し、ファイル名を指定し、**「保管」**をクリックします。ログ・メッセージまたはトレース・メッセージは、指定した場所の指定したファイルに書き込まれます。

     注: ログ・ファイルとトレース・ファイルはどちらも単純なテキスト・ファイルです。任意のファイル拡張子を使用してこれらのファイル名を設定できます。ただし、トレース・ファイルの場合、拡張子として `.trc` ではなく `.trace` を使用することをお勧めします。これは、システム内に存在する可能性がある他のファイルとの混同を避けるためです。ログ・ファイルの場合、通常使用されるファイル拡張子は `.log` および `.txt` です。

データ・ハンドラー

データ・ハンドラー・セクションの構成が使用可能となるのは、DeliveryTransport の値に JMS を、また ContainerManagedEvents の値に JMS を指定した場合のみです。すべてのアダプターでデータ・ハンドラーを使用できるわけではありません。

これらのプロパティーに使用する値については、『付録 A. コネクターの標準構成プロパティー』にある ContainerManagedEvents の下の説明を参照してください。その他の詳細は、「*コネクター開発ガイド (C++ 用)*」または「*コネクター開発ガイド (Java 用)*」を参照してください。

構成ファイルの保管

コネクターの構成が完了したら、コネクター構成ファイルを保管します。Connector Configurator では、構成中に選択したブローカー・モードでファイルを保管します。Connector Configurator のタイトル・バーには現在のブローカー・モード (ICS、WMQI、または WAS) が常に表示されます。

ファイルは XML 文書として保管されます。XML 文書は次の 3 通りの方法で保管できます。

- System Manager から、統合コンポーネント・ライブラリーに *.con 拡張子付きファイルとして保管します。
- 指定したディレクトリーに保管します。
- スタンドアロン・モードで、ディレクトリー・フォルダーに *.cfg 拡張子付きファイルとして保管します。デフォルトでは、このファイルは ¥WebSphereAdapters¥bin¥Data¥App に保管されます。
- WebSphere Application Server プロジェクトをセットアップしている場合には、このファイルを WebSphere Application Server プロジェクトに保管することもできます。

System Manager でのプロジェクトの使用法、および配置の詳細については、以下のインプリメンテーション・ガイドを参照してください。

- ICS:「*WebSphere InterChange Server システム・インプリメンテーション・ガイド*」
- WebSphere Message Brokers: 「*WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド*」
- WAS: 「*アダプター実装ガイド (WebSphere Application Server)*」

構成ファイルの変更

既存の構成ファイルの統合ブローカー設定を変更できます。これにより、他のブローカーで使用する構成ファイルを新規に作成するときに、このファイルをテンプレートとして使用できます。

注: 統合ブローカーを切り替える場合には、ブローカー・モード・プロパティーと同様に他の構成プロパティーも変更する必要があります。

既存の構成ファイルでのブローカーの選択を変更するには、以下の手順を実行します (オプション)。

- Connector Configurator で既存の構成ファイルを開きます。
- **「標準のプロパティー」**タブを選択します。
- 「標準のプロパティー」タブの**「BrokerType」**フィールドで、ご使用のブローカーに合った値を選択します。
  現行値を変更すると、プロパティー画面の利用可能なタブおよびフィールド選択がただちに変更され、選択した新規ブローカーに適したタブとフィールドのみが表示されます。

構成の完了

コネクターの構成ファイルを作成し、そのファイルを変更した後で、コネクターの始動時にコネクターが構成ファイルの位置を特定できるかどうかを確認してください。

これを行うには、コネクターが使用する始動ファイルを開き、コネクター構成ファイルに使用されている格納場所とファイル名が、ファイルに対して指定した名前およびファイルを格納したディレクトリーまたはパスと正確に一致しているかどうかを検証します。

グローバル化環境における Connector Configurator の使用

Connector Configurator はグローバル化され、構成ファイルと統合ブローカー間の文字変換を処理できます。 Connector Configurator では、ネイティブなエンコード方式を使用しています。構成ファイルに書き込む場合は UTF-8 エンコード方式を使用します。

Connector Configurator は、以下の場所で英語以外の文字をサポートします。

- すべての値のフィールド
- ログ・ファイルおよびトレース・ファイル・パス (**「トレース/ログ・ファイル」**タブで指定)

CharacterEncoding および Locale 標準構成プロパティーのドロップ・リストに表示されるのは、サポートされる値の一部のみです。ドロップ・リストに、サポートされる他の値を追加するには、製品ディレクトリーの ¥Data¥Std¥stdConnProps.xml ファイルを手動で変更する必要があります。

例えば、Locale プロパティーの値のリストにロケール en_GB を追加するには、stdConnProps.xml ファイルを開き、以下に太文字で示した行を追加してください。

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
                <ValidType>String</ValidType>
            <ValidValues>
                                <Value>ja_JP</Value>
                                <Value>ko_KR</Value>
                                <Value>zh_CN</Value>
                                <Value>zh_TW</Value>
                                <Value>fr_FR</Value>
                                <Value>de_DE</Value>
                                <Value>it_IT</Value>
                                <Value>es_ES</Value>
                                <Value>pt_BR</Value>
                                <Value>en_US</Value>
                                <Value>en_GB</Value>

                    <DefaultValue>en_US</DefaultValue>
            </ValidValues>
    </Property>
```

付録 C. ビジネス・オブジェクトのサンプル

この付録では、以下の i2ADW Connector 用のビジネス・オブジェクトのサンプルを示します。

- BO_I2ADW_BOMMASTER
- BO_I2ADW_DEMANDHISTORY
- BO_I2ADW_ITEMMASTER
- BO_I2ADW_DEMANDFORECAST
- BO_I2ADW_FORECAST

BO_I2ADW_BOMMASTER

テストに使用するビジネス・オブジェクト

```
[BusinessObjectDefinition]
Name = BOM
Version = 1.0.0
AppSpecificInfo = TN=BOM;SCN=status:inactive

[Attribute]
Name = CTRL_ID
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=CTRL_ID:::::
DefaultValue =
[End]

[Attribute]
Name = FCLTY_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=FCLTY_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = BOM_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=BOM_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = EFF_START_DTTM
Type = Date
Cardinality = 1
```

```
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EFF_START_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = EFF_END_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EFF_END_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = ENGINE_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ENGINE_ID:::::
DefaultValue =
[End]

[Attribute]
Name = SRC_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRC_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SYNC_IND
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SYNC_IND:::::
DefaultValue =
[End]

[Attribute]
Name = ObjectEventId
Type = String
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo =
DefaultValue = null
[End]
```

```
[Verb]
Name = Create
AppSpecificInfo =
[End]

[Verb]
Name = Retrieve
AppSpecificInfo =
[End]

[Verb]
Name = Update
AppSpecificInfo =
[End]

[Verb]
Name = Delete
AppSpecificInfo =
[End]

[End]
```

BO_I2ADW_DEMANDHISTORY

テストに使用するビジネス・オブジェクト

```
[BusinessObjectDefinition]
Name = DEMAND_HISTORY
Version = 1.0.0
AppSpecificInfo = TN=DEMAND_H;SCN=status:inactive


        [Attribute]
        Name = HIST_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=HIST_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = CTRL_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=CTRL_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = PLAN_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=PLAN_ID:::::
        DefaultValue =
        [End]
```

```
[Attribute]
Name = DEMAND_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=DEMAND_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = CUST_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CUST_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = SELLER_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SELLER_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = CAT
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CAT:::::
DefaultValue =
[End]

[Attribute]
Name = DEMAND_TYPE
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=DEMAND_TYPE:::::
DefaultValue =
[End]

[Attribute]
Name = PRIORITY
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
```

```
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = PRIORITY_CRITERION
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=PRIORITY_CRITERION:::::
DefaultValue =
[End]

[Attribute]
Name = EXPIRED_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EXPIRED_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SHIP_COMPLETE
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SHIP_COMPLETE:::::
DefaultValue =
[End]

[Attribute]
Name = ON_TIME
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ON_TIME:::::
DefaultValue =
[End]

[Attribute]
Name = ONE_PRODUCT
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ONE_PRODUCT:::::
DefaultValue =
[End]

[Attribute]
```

```
Name = ERLNS_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ERLNS_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = LATNS_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=LATNS_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=TOL_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = MAKE_TO_STOCK
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MAKE_TO_STOCK:::::
DefaultValue =
[End]

[Attribute]
Name = MAX_SHIPMENTS
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MAX_SHIPMENTS:::::
DefaultValue =
[End]

[Attribute]
Name = CUST_ORD_NUM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
```

```
AppSpecificInfo = CN=CUST_ORD_NUM:::::
DefaultValue =
[End]

[Attribute]
Name = FCST_OR_ACTUAL
Type = String
Cardinality = 1
MaxLength = 18
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FCST_OR_ACTUAL:::::
DefaultValue =
[End]

[Attribute]
Name = ISSUED_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ISSUED_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = ENGINE_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ENGINE_ID:::::
DefaultValue =
[End]

[Attribute]
Name = WORKFLOW_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=WORKFLOW_ID:::::
DefaultValue =
[End]

[Attribute]
Name = ERLST_TOL_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ERLST_TOL_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = LATST_TOL_DTTM
Type = Date
```

```
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=LATST_TOL_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SRC_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRC_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SYNC_IND
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SYNC_IND:::::
DefaultValue =
[End]

[Attribute]
Name = ObjectEventId
Type = String
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo =
DefaultValue = null
[End]

[Verb]
Name = Create
AppSpecificInfo =
[End]

[Verb]
Name = Retrieve
AppSpecificInfo =
[End]

[Verb]
Name = Update
AppSpecificInfo =
[End]

[Verb]
Name = Delete
AppSpecificInfo =
[End]


[End]
```

BO_I2ADW_ITEMMASTER

テストに使用するビジネス・オブジェクト

```
[BusinessObjectDefinition]
Name = ITEM_MASTER
Version = 1.0.0
AppSpecificInfo = TN=ITEM_MASTER;SCN=status:inactive


        [Attribute]
        Name = CTRL_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=CTRL_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = ITEM_NAME
        Type = String
        Cardinality = 1
        MaxLength = 40
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=ITEM_NAME:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = DESCR
        Type = String
        Cardinality = 1
        MaxLength = 200
        IsKey = false
        IsForeignKey = false
        IsRequired  = false
        AppSpecificInfo = CN=DESCR:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = CAT
        Type = String
        Cardinality = 1
        MaxLength = 40
        IsKey = false
        IsForeignKey = false
        IsRequired  = false
        AppSpecificInfo = CN=CAT:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = SUB_CAT
        Type = String
        Cardinality = 1
        MaxLength = 40
        IsKey = false
        IsForeignKey = false
        IsRequired  = false
        AppSpecificInfo = CN=SUB_CAT:::::
```

```
DefaultValue =
[End]

[Attribute]
Name = CUBE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CUBE:::::
DefaultValue =
[End]

[Attribute]
Name = UNIT_PRICE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_PRICE:::::
DefaultValue =
[End]

[Attribute]
Name = UNIT_PRICE_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_PRICE_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = WGT
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=WGT:::::
DefaultValue =
[End]

[Attribute]
Name = UNIT_COST
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_COST:::::
DefaultValue =
[End]

[Attribute]
Name = UNIT_COST_UOM
Type = String
Cardinality = 1
```

```
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_COST_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = STOR_STACK_HEIGHT
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=STOR_STACK_HEIGHT:::::
DefaultValue =
[End]

[Attribute]
Name = UNITS_PER_PALLET
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNITS_PER_PALLET:::::
DefaultValue =
[End]

[Attribute]
Name = INV_VALUE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=INV_VALUE:::::
DefaultValue =
[End]

[Attribute]
Name = COMMODITY_CLASS
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=COMMODITY_CLASS:::::
DefaultValue =
[End]

[Attribute]
Name = VOL_CLASS
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=VOL_CLASS:::::
DefaultValue =
[End]
```

```
[Attribute]
Name = REBALANCE_FLAG
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=REBALANCE_FLAG:::::
DefaultValue =
[End]

[Attribute]
Name = ALLOC_FLAG
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ALLOC_FLAG:::::
DefaultValue =
[End]

[Attribute]
Name = QTY_PER_UNIT_SPACE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=QTY_PER_UNIT_SPACE:::::
DefaultValue =
[End]

[Attribute]
Name = STACK_GRP
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=STACK_GRP:::::
DefaultValue =
[End]

[Attribute]
Name = DEFICIT_PRIORITY
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=DEFICIT_PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = SRPLS_PRIORITY
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
```

```
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRPLS_PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = IS_PRIORITY
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=IS_PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = ITEM_LOAD_MIX_GRP
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ITEM_LOAD_MIX_GRP:::::
DefaultValue =
[End]

[Attribute]
Name = ARTIFICIAL
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ARTIFICIAL:::::
DefaultValue =
[End]

[Attribute]
Name = DMD_PART_NUM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=DMD_PART_NUM:::::
DefaultValue =
[End]

[Attribute]
Name = UNIT_SPACE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_SPACE:::::
DefaultValue =
[End]

[Attribute]
```

```
Name = UNIT_SPACE_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=UNIT_SPACE_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = MIN_RTNG_TIME
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MIN_RTNG_TIME:::::
DefaultValue =
[End]

[Attribute]
Name = MIN_RTNG_TIME_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MIN_RTNG_TIME_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = STOCK_PART
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=STOCK_PART:::::
DefaultValue =
[End]

[Attribute]
Name = SELLABLE
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SELLABLE:::::
DefaultValue =
[End]

[Attribute]
Name = ENGINE_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
```

```
AppSpecificInfo = CN=ENGINE_ID:::::
DefaultValue =
[End]

[Attribute]
Name = EXP_TIMING
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EXP_TIMING:::::
DefaultValue =
[End]

[Attribute]
Name = GL_CAT
Type = String
Cardinality = 1
MaxLength = 18
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=GL_CAT:::::
DefaultValue =
[End]

[Attribute]
Name = SFTY_STOCK_QTY_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SFTY_STOCK_QTY_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = U_CARRYING_CPT
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=U_CARRYING_CPT:::::
DefaultValue =
[End]

[Attribute]
Name = U_CARRYING_C_PER_U
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=U_CARRYING_C_PER_U:::::
DefaultValue =
[End]

[Attribute]
Name = VMI_PART
Type = String
```

```
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=VMI_PART:::::
DefaultValue =
[End]

[Attribute]
Name = DFT_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=DFT_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = ATO_DEXP_QTY_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_DEXP_QTY_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = ATO_QTY_TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_QTY_TOL_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = ATO_DEXP_TM_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_DEXP_TM_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = ATO_TM_TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_TM_TOL_UOM:::::
DefaultValue =
```

```
[End]

[Attribute]
Name = ATO_EXP_QTY_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_EXP_QTY_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = ATO_EXP_TM_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ATO_EXP_TM_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = MTRL_AV_SET_EPST
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MTRL_AV_SET_EPST:::::
DefaultValue =
[End]

[Attribute]
Name = MTRL_AV_SET_EPST_U
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MTRL_AV_SET_EPST_U:::::
DefaultValue =
[End]

[Attribute]
Name = IS_SUBCONTRACTED
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=IS_SUBCONTRACTED:::::
DefaultValue =
[End]

[Attribute]
Name = CRIT_FOR_ASSEM
Type = String
Cardinality = 1
MaxLength = 1
```

```
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CRIT_FOR_ASSEM:::::
DefaultValue =
[End]

[Attribute]
Name = FREIGHT_CLASS
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FREIGHT_CLASS:::::
DefaultValue =
[End]

[Attribute]
Name = SRC_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRC_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SYNC_IND
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SYNC_IND:::::
DefaultValue =
[End]

[Attribute]
Name = EDC_DEXP_QTY_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_DEXP_QTY_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = EDC_QTY_TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_QTY_TOL_UOM:::::
DefaultValue =
[End]
```

```
[Attribute]
Name = EDC_DEXP_TM_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_DEXP_TM_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = EDC_TM_TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_TM_TOL_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = EDC_EXP_QTY_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_EXP_QTY_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = EDC_EXP_TM_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EDC_EXP_TM_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = CONFIG_ID
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CONFIG_ID:::::
DefaultValue =
[End]

[Attribute]
Name = ITEM_MAKE_TYPE
Type = String
Cardinality = 1
MaxLength = 18
IsKey = false
IsForeignKey = false
```

```
        IsRequired  = false
        AppSpecificInfo = CN=ITEM_MAKE_TYPE:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = ObjectEventId
        Type = String
        Cardinality = 1
        MaxLength = 0
        IsKey = false
        IsForeignKey = false
        IsRequired  = false
        AppSpecificInfo =
        DefaultValue = null
        [End]

        [Verb]
        Name = Create
        AppSpecificInfo =
        [End]

        [Verb]
        Name = Retrieve
        AppSpecificInfo =
        [End]

        [Verb]
        Name = Update
        AppSpecificInfo =
        [End]

        [Verb]
        Name = Delete
        AppSpecificInfo =
        [End]


[End]
```

BO_I2ADW_DEMANDFORECAST

テストに使用するビジネス・オブジェクト

```
[BusinessObjectDefinition]
Name = DEMAND
Version = 1.0.0
AppSpecificInfo = TN=DEMAND;SCN=status:inactive

        [Attribute]
        Name = CTRL_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=CTRL_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = PLAN_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
```

```
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=PLAN_ID:::::
DefaultValue =
[End]

[Attribute]
Name = DEMAND_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=DEMAND_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = CUST_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CUST_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = SELLER_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SELLER_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = CAT
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CAT:::::
DefaultValue =
[End]

[Attribute]
Name = DEMAND_TYPE
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=DEMAND_TYPE:::::
DefaultValue =
[End]
```

```
[Attribute]
Name = PRIORITY
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = PRIORITY_CRITERION
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=PRIORITY_CRITERION:::::
DefaultValue =
[End]

[Attribute]
Name = EXPIRED_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=EXPIRED_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SHIP_COMPLETE
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SHIP_COMPLETE:::::
DefaultValue =
[End]

[Attribute]
Name = ON_TIME
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ON_TIME:::::
DefaultValue =
[End]

[Attribute]
Name = ONE_PRODUCT
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
```

```
IsRequired  = false
AppSpecificInfo = CN=ONE_PRODUCT:::::
DefaultValue =
[End]

[Attribute]
Name = ERLNS_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ERLNS_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = LATNS_TOL
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=LATNS_TOL:::::
DefaultValue =
[End]

[Attribute]
Name = TOL_UOM
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=TOL_UOM:::::
DefaultValue =
[End]

[Attribute]
Name = MAKE_TO_STOCK
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MAKE_TO_STOCK:::::
DefaultValue =
[End]

[Attribute]
Name = MAX_SHIPMENTS
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=MAX_SHIPMENTS:::::
DefaultValue =
[End]

[Attribute]
Name = CUST_ORD_NUM
```

```
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=CUST_ORD_NUM:::::
DefaultValue =
[End]

[Attribute]
Name = FCST_OR_ACTUAL
Type = String
Cardinality = 1
MaxLength = 18
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FCST_OR_ACTUAL:::::
DefaultValue =
[End]

[Attribute]
Name = ISSUED_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ISSUED_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = ENGINE_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ENGINE_ID:::::
DefaultValue =
[End]

[Attribute]
Name = WORKFLOW_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=WORKFLOW_ID:::::
DefaultValue =
[End]

[Attribute]
Name = ERLST_TOL_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ERLST_TOL_DTTM:::::
```

```
DefaultValue =
[End]

[Attribute]
Name = LATST_TOL_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=LATST_TOL_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SRC_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRC_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SYNC_IND
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SYNC_IND:::::
DefaultValue =
[End]

[Attribute]
Name = ObjectEventId
Type = String
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo =
DefaultValue = null
[End]

[Verb]
Name = Create
AppSpecificInfo =
[End]

[Verb]
Name = Retrieve
AppSpecificInfo =
[End]

[Verb]
Name = Update
AppSpecificInfo =
[End]

[Verb]
```

```
        Name = Delete
        AppSpecificInfo =
        [End]


[End]
```

BO_I2ADW_FORECAST

テストに使用するビジネス・オブジェクト

```
[BusinessObjectDefinition]
Name = FORECAST
Version = 1.0.0
AppSpecificInfo = TN=FORCST;SCN=status:inactive


        [Attribute]
        Name = CTRL_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=CTRL_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = PLAN_ID
        Type = Integer
        Cardinality = 1
        MaxLength = 0
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=PLAN_ID:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = PROD_NAME
        Type = String
        Cardinality = 1
        MaxLength = 40
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=PROD_NAME:::::
        DefaultValue =
        [End]

        [Attribute]
        Name = SELLER_NAME
        Type = String
        Cardinality = 1
        MaxLength = 40
        IsKey = true
        IsForeignKey = true
        IsRequired  = false
        AppSpecificInfo = CN=SELLER_NAME:::::
        DefaultValue =
        [End]

        [Attribute]
```

```
Name = FCST_DUE_DTTM_STRT
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = true
IsForeignKey = true
IsRequired  = false
AppSpecificInfo = CN=FCST_DUE_DTTM_STRT:::::
DefaultValue =
[End]

[Attribute]
Name = FORCST_NAME
Type = String
Cardinality = 1
MaxLength = 40
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FORCST_NAME:::::
DefaultValue =
[End]

[Attribute]
Name = FCST_DUE_DTTM_END
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FCST_DUE_DTTM_END:::::
DefaultValue =
[End]

[Attribute]
Name = FORCST_VALUE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FORCST_VALUE:::::
DefaultValue =
[End]

[Attribute]
Name = COMMITTED_VALUE
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=COMMITTED_VALUE:::::
DefaultValue =
[End]

[Attribute]
Name = COMMITTED_PCTG
Type = Float
Cardinality = 1
MaxLength = 19
IsKey = false
IsForeignKey = false
IsRequired  = false
```

```
AppSpecificInfo = CN=COMMITTED_PCTG:::::
DefaultValue =
[End]

[Attribute]
Name = PRIORITY
Type = Float
Cardinality = 1
MaxLength = 15
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=PRIORITY:::::
DefaultValue =
[End]

[Attribute]
Name = FULFILLMENT_PLCY
Type = String
Cardinality = 1
MaxLength = 200
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=FULFILLMENT_PLCY:::::
DefaultValue =
[End]

[Attribute]
Name = ENGINE_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ENGINE_ID:::::
DefaultValue =
[End]

[Attribute]
Name = COMMITTED_RANK
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=COMMITTED_RANK:::::
DefaultValue =
[End]

[Attribute]
Name = SURP_RANK
Type = Integer
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SURP_RANK:::::
DefaultValue =
[End]

[Attribute]
Name = ALLOC_RANK
Type = Integer
```

```
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=ALLOC_RANK:::::
DefaultValue =
[End]

[Attribute]
Name = WORKFLOW_ID
Type = Integer
Cardinality = 1
MaxLength = 28
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=WORKFLOW_ID:::::
DefaultValue =
[End]

[Attribute]
Name = SRC_DTTM
Type = Date
Cardinality = 1
MaxLength = 7
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SRC_DTTM:::::
DefaultValue =
[End]

[Attribute]
Name = SYNC_IND
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo = CN=SYNC_IND:::::
DefaultValue =
[End]

[Attribute]
Name = ObjectEventId
Type = String
Cardinality = 1
MaxLength = 0
IsKey = false
IsForeignKey = false
IsRequired  = false
AppSpecificInfo =
DefaultValue = null
[End]

[Verb]
Name = Create
AppSpecificInfo =
[End]

[Verb]
Name = Retrieve
AppSpecificInfo =
[End]
```

```
        [Verb]
        Name = Update
        AppSpecificInfo =
        [End]

        [Verb]
        Name = Delete
        AppSpecificInfo =
        [End]


[End]
```

付録 D. ヌル値およびブランク値のサポート

この付録では、ビジネス・オブジェクトのキー値がブランクまたはヌルの場合のさまざまな合格/不合格シナリオについて詳しく説明します。この付録では、ブランクまたはヌルのビジネス・オブジェクト値を持つために必要な機能的変更についても説明します。

成功と失敗のシナリオ

ビジネス・オブジェクトのキー値がデータベース内にブランク値またはヌル値を持っている場合は、「=」演算子タイプではなく「is null」タイプの where 文節を構築してください。

IBM では、ビジネス・オブジェクトに対して、ブランク値を持たないキー属性を 1 つ以上定義することをお勧めします。

以下は、1 つのキーを持ち、このキーがヌル値を持っている親オブジェクトのシナリオです。このような条件に当てはまるシナリオは不合格です。

表 18. Customer

属性	タイプ
cid	Integer (キー)
name	String
comments	String

以下は、2 つのキーを持ち、そのうちの 1 つがヌル値を持っている親オブジェクトのシナリオです。このような条件に当てはまるシナリオは合格です。

表 19. Customer

属性	タイプ
cid	Integer (キー)
name	String
comments	String

2 つ目の例では、cid=1000 かつ name がヌルに設定されているという条件で customer から cid、name、および comments を選択することにより、retrieve 照会を構築します。

以下は、コンテナー・オブジェクト内に 1 つの外部キー参照を持つ子オブジェクトを 1 つ持っている親オブジェクトのシナリオです。このような条件に当てはまるシナリオは不合格です。

表 20. Customer

属性	タイプ
cid	Integer (キー)

表 20. Customer (続き)

属性	タイプ
name	String (キー)
comments	String
Address	Address
Aid	Integer (キー) ASI:FK=cid
Acity	String
Azip	String

cid にヌル値が含まれている場合は、address から Aid、Acity、および Azip を選択することにより、retrieve 照会を構築します。Aid の値はヌルに設定してください。

以下は、コンテナー・オブジェクト内に 2 つの外部キー参照を持つ子オブジェクトを 1 つ持っている親オブジェクトのシナリオです。このような条件に当てはまるシナリオは合格です。

表 21. Customer

属性	タイプ
cid	Integer (キー)
name	String
comments	String
Address	Address
Aid	Integer (キー) ASI:FK=cid
Acity	String (キー) ASI:FK=name
Azip	String

name にヌル値が含まれている場合は、Aid=Cid と Acity がヌル値を持っているという条件で address から Aid、Acity、および Azip を選択することにより、Retrieve 照会を構築します。

機能性

コネクターは、キーにブランク値を見つけると、その値を属性の UseNull 値と比較します。この結果の値が true の場合、コネクターはヌル値を照会に追加します。これにより、以下の動詞の操作が影響を受けます。

- Retrieve
- RetrieveBy Content
- Update
- Delete

特記事項

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。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
Licensing

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 Burlingame Laboratory Director
IBM Burlingame Laboratory
577 Airport Blvd., Suite 800
Burlingame, CA 94010
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。

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。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

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。

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

以下は、IBM Corporation の商標です。

IBM
IBM ロゴ
AIX
CrossWorlds
DB2
DB2 Universal Database
Lotus
Lotus Domino
Lotus Notes
MQIntegrator
MQSeries
Tivoli
WebSphere

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

MMX、Pentium および ProShare は、Intel 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または登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

Adapter for i2 Active Data Warehouse 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
〒106-8711　東京都港区六本木3-2-12