InterSystems SQL ゲートウェイの使用法

Version 2023.1
2024-01-02

InterSystems Corporation　1 Memorial Drive　Cambridge MA 02142　www.intersystems.com

InterSystems SQL ゲートウェイの使用法
InterSystems IRIS Data Platform　Version 2023.1　2024-01-02

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください：

InterSystems Worldwide Response Center (WRC)
Tel:　+1-617-621-0700
Tel:　+44 (0) 844 854 2917
Email:　support@InterSystems.com

目次

テーブル一覧

1

SQL ゲートウェイの概要

このドキュメントで取り上げる内容の詳細なリストは、"目次" を参照してください。

InterSystems SQL ゲートウェイを使用すると、InterSystems IRIS® データ・プラットフォームから外部データベースにアクセスできます。さまざまなウィザードを使用して外部ソースのテーブル、ビュー、またはストアド・プロシージャへのリンクを作成できるので、InterSystems IRIS オブジェクトにアクセスする場合と同じ方法でデータにアクセスできます。

- サードパーティのリレーショナル・データベースに保存されたデータへ、InterSystems IRIS アプリケーション内でオブジェクトや SQL クエリを使用してアクセスします。
- InterSystems IRIS の永続オブジェクトを外部リレーショナル・データベースに格納します。
- 対応する外部ストアド・プロシージャと同じアクションを実行するクラス・メソッドを作成します。
- InterSystems JDBC ドライバまたは InterSystems ODBC ドライバのいずれかを介して接続します。

詳細は、以下のトピックを参照してください。

- "SQL ゲートウェイを使用したデータベースへのアクセス" は、SQL ゲートウェイの概要を示し、外部ソースへリンクする方法を説明します。
- "JDBC 経由での SQL ゲートウェイへの接続" は、SQL ゲートウェイ用の JDBC 論理接続の定義を作成する方法を説明します。
- "ODBC 経由での SQL ゲートウェイへの接続" は、SQL ゲートウェイ用の ODBC 論理接続の定義を作成する方法を説明します。
- "プログラムによる SQL ゲートウェイの使用法" は、**%SQLGatewayConnection** クラスを使用して、ObjectScript から ODBC 関数を呼び出す方法を説明します。

関連ドキュメント

以下のドキュメントに関連資料が含まれています。

- InterSystems SQL の使用法 ー InterSystems IRIS データベース内に保存されているデータへの標準のリレーショナル・アクセスを提供する InterSystems SQL の使用法について説明しています。
- InterSystems ソフトウェアでの Java の使用法 ー InterSystems JDBC ドライバによって有効になるすべての InterSystems Java テクノロジの概要を示し、このドライバを使用して SQL 経由でデータ・ソースにアクセスする方法について説明します。
- InterSystems ODBC ドライバの使用法 ー InterSystems ODBC を介して外部アプリケーションから InterSystems IRIS に接続する方法、および InterSystems IRIS から外部 ODBC データ・ソースにアクセスする方法について説明します。

2

SQL ゲートウェイを使用したデータベースへのアクセス

InterSystems SQL ゲートウェイを使用すると、JDBC および ODBC 経由で InterSystems IRIS® Data Platform から外部データベースにアクセスできます。これらについて、以下のトピックで説明します。

- InterSystems SQL ゲートウェイのアーキテクチャ ― SQL ゲートウェイの内部構造および制約について説明します。
- 外部ソースへの SQL ゲートウェイ接続の作成 ― 論理接続の定義の概要を説明します。これは SQL ゲートウェイ・ウィザードでの外部データベースの特定に使用されます。
- リンク・テーブル・ウィザード：テーブルまたはビューへのリンク ― InterSystems IRIS オブジェクトへのアクセスと同じ方法でデータにアクセスできるように、外部ソースのテーブルまたはビューにリンクする手順について説明します。
- プロシージャへのリンク作成ウィザード：ストアド・プロシージャへのリンク ― 外部ソースのストアド・プロシージャにリンクする手順について説明します。
- SQL ゲートウェイ接続の制御 ― SQL ゲートウェイの接続の管理に使用されるメソッドを説明します。

論理接続定義の作成の詳細は、以下のセクションを参照してください。

- JDBC 経由での SQL ゲートウェイへの接続
- ODBC 経由での SQL ゲートウェイへの接続

2.1 InterSystems SQL ゲートウェイのアーキテクチャ

InterSystems SQL ゲートウェイは、内部的には以下のコンポーネントを使用しています。

- 接続マネージャは、InterSystems IRIS の論理接続定義のリストを管理します。各定義には InterSystems IRIS で使用する論理名、および ODBC または JDBC 準拠の特定の外部データベースに関する接続の詳細が含まれています。InterSystems SQL ゲートウェイは、接続の確立時にこれらの論理名を使用します（“外部ソースへの SQL ゲートウェイ接続の作成”を参照）。
- InterSystems SQL ゲートウェイ API は、サードパーティの RDBMS と通信するために InterSystems IRIS プログラムで使用される一連の関数です。これらの関数は、ODBC または JDBC 呼び出しを作成する共有ライブラリを使用して実装されます。
- 外部テーブル・クエリ・プロセッサは、外部テーブルを対象とするクエリを処理する InterSystems SQL クエリ・プロセッサの拡張機能です。

- SQL ディクショナリには、すべての定義済み SQL テーブルのリストが格納されます。任意のテーブルは、そのデータが外部の RDBMS に保存されたときに "外部" のものとしてマークされます。InterSystems SQL クエリ・プロセッサによって、SQL クエリ内で参照されたテーブルが外部のテーブルであることが検出されると、このプロセッサは、外部テーブル・クエリ・プロセッサを起動します。外部テーブル・クエリ・プロセッサでは、InterSystems IRIS 内に格納されているデータにアクセスする代わりに、InterSystems SQL ゲートウェイ API を呼び出すことによって、クエリ実行プランが生成されます。

2.1.1 InterSystems IRIS の外部テーブルの永続化

InterSystems IRIS のオブジェクト永続性はすべて、ストレージ・クラスによって実現されます ("クラスの定義と使用" の "ストレージ定義とストレージ・クラス" を参照)。ストレージ・クラスによって、データベース内の永続オブジェクトの保存および取得に必要なコードが生成されます。SQL ストレージ・クラス (**%Storage.SQL**) は、特別に生成された SQL クエリを使用してオブジェクトの永続性を実現します。

永続性を確保するために **%Storage.SQL** を使用するクラスでは、そのクラスの **CONNECTION** および **EXTERNALTABLENAME** クラス・パラメータに値を指定することによって "外部の" クラスであることを示します。クラス・コンパイラでは、そのクラスの SQL テーブル定義を作成し、オブジェクト永続性コードの SQL クエリを生成します。これらのクエリは、外部テーブル・クエリ・プロセッサによって自動的に正しい外部データベースを呼び出します。

2.1.2 SQL ゲートウェイ・クエリに関する制限事項

InterSystems SQL ゲートウェイを使用する場合、以下の制限事項に注意してください。

- 複数のデータ・ソースからデータを結合するクエリは、ODBC 接続の場合にのみサポートされます。JDBC を使用する場合、SQL クエリの **FROM** 節にあるすべてのテーブルは、同じデータ・ソースから派生したものである必要があります。
- 外部データベースを対象とした SQL クエリでは、以下の InterSystems SQL 拡張文法を使用できません。
  - "->" 演算子
  - %EXACT 関数、または EXACT に設定された照合フラグを伴う **%SYSTEM.Util** Collation() メソッド
  - Count (*) クエリ内に他の列を含むこと
  - 名前が **%** で始まる InterSystems IRIS 固有の演算子

2.2 外部ソースへの SQL ゲートウェイ接続の作成

InterSystems IRIS では、外部データ・ソースへの接続の論理名である、SQL ゲートウェイ接続定義のリストが管理されます。各接続定義は、論理名 (InterSystems IRIS 内で使用するため)、データ・ソースへの接続情報、および接続を確立するときに使用するユーザ名とパスワードで構成されます。これらの接続は、**%Library.sys_SQLConnection** テーブルに格納されます。このテーブルのデータをエクスポートして、同じバージョンの InterSystems IRIS の別のインスタンスにインポートできます。

SQL ゲートウェイ接続の構成要素は、以下のとおりです。

- 接続の論理名。この名前は、InterSystems SQL クエリなどで使用されます。
- データベースにアクセスするためのログイン資格情報 (オプション)。
- JDBC または ODBC ドライバを制御するための情報 (オプション)。
- ドライバ固有の接続の詳細 :

- JDBC 用 : JDBC クライアント・ドライバの完全クラス名、ドライバ・クラス・パス (JDBC ドライバの特定時に検索する JAR ファイルのリスト)、および JDBC 接続 URL
- ODBC 用 : 通常の方法で定義された DSN (データ・ソース名) (″InterSystems ODBC ドライバの使用法″ の ″Windows での ODBC データ・ソースの定義″ および ″UNIX® での ODBC データ・ソースの定義″ を参照)。

  注釈　Microsoft SQL Server DNS 構成を使用して、リンク・テーブル・ウィザードで使用される接続を作成する場合は、[**地域設定を使用**] オプションを設定しないでください。このオプションは、データを処理するアプリケーションではなく、データを表示するアプリケーションのみを対象としています。

論理接続定義の作成の詳細は、以下の章を参照してください。

- JDBC 経由での SQL ゲートウェイへの接続
- ODBC 経由での SQL ゲートウェイへの接続

2.3 リンク・テーブル・ウィザード : テーブルまたはビューへのリンク

管理ポータルには、ODBC 準拠または JDBC 準拠のデータベースの外部テーブルへのリンクに使用できるウィザードがあります。外部テーブルにリンクすると、以下の操作を実行できます。

- サードパーティのリレーショナル・データベースに保存されたデータへ、InterSystems IRIS アプリケーション内でオブジェクトや SQL クエリを使用してアクセスします。
- InterSystems IRIS の永続オブジェクトを外部リレーショナル・データベースに格納します。

例えば、**Employee** テーブルが、外部リレーショナル・データベースに保存されているとします。JDBC または ODBC 経由で SQL クエリを実行することによって外部データベースと通信する **Employee** クラスを作成することによって、このテーブルをオブジェクトとして InterSystems IRIS 内で使用できます。

InterSystems IRIS アプリケーションから見ると、**Employee** クラスは他の永続クラスと同じように動作します。インスタンスを開き、変更して、それを保存できます。SQL クエリを **Employee** クラスに対して発行する場合、このクラスは自動的に外部データベースに送信されます。

InterSystems SQL ゲートウェイの使用は、アプリケーション論理とは独立した関係です。最小限の処理でアプリケーション論理が変更されることなく外部データベースと組み込みの InterSystems IRIS データベースの間で切り替わるようにアプリケーションを変更することができます。

InterSystems SQL ゲートウェイを使用して、オブジェクトに永続性を提供するクラスは、使用性においては、ネイティブの永続性を使用して、Java、SQL、Web アクセスなどの InterSystems IRIS のすべての機能を使用できるクラスと同等です。

2.3.1 リンク・テーブル・ウィザードの使用法

外部テーブルまたはビューにリンクする場合、そのテーブルまたはビューにリンクされる InterSystems IRIS 永続クラスを作成します。その新しいクラスは、SQL ゲートウェイを使用して外部ソースからデータを取得したり、格納したりします。InterSystems IRIS クラスと InterSystems IRIS 内の対応する SQL テーブルの両方に関する情報を指定できます。

注釈　このウィザードでは、制御対象のクラス名およびクラス・メンバ名を使用して ObjectScript コードが生成されます。このウィザードを使用する場合は、長さ制限も含め、ObjectScript 識別子のルールに従う必要があります (″クラスの定義と使用″ の ″名前付け規約″ のセクションを参照)。

- 外部データベースへの接続をまだ作成していない場合は、開始する前に作成してください（“外部ソースへの SQL ゲートウェイ接続の作成”を参照）。
- 管理ポータルで、**[システム・エクスプローラ]**、[SQL] の順に選択します。ページ上部の **[切り替え]** オプションを使ってネームスペースを選択します。利用可能なネームスペースのリストが表示されます。

  ページ上部の [　　] ドロップダウン・リストをクリックし、**[リンクテーブル]** を選択します。
- ウィザードの最初のページで、以下のように 1 つまたは複数のテーブルまたはビューを選択します。
  - [　　] ー データのコピー先の InterSystems IRIS ネームスペースを選択します。
  - [　　] ー テーブルまたはビューを含むスキーマ（クラス・パッケージ）名を指定します。ワイルドカードを使用して名前を指定し、複数のスキーマを返すことや、% を使用してすべてのスキーマを返すことができます。例えば、C% は、先頭文字が C であるネームスペース内のすべてのスキーマを返します。選択元となるスキーマの返されるリストが短くなることでロード速度が向上するため、このフィルタを使用することをお勧めします。複数の項目を選択できます。複数の項目を選択する場合、[　] をクリックすると、次の画面にパッケージ名の入力を求めるプロンプトが表示されます。クラスを格納するパッケージの名前を指定して、[　] をクリックします。
  - [　　] ー リンク先のテーブルまたはビューを指定します。ワイルドカードを使用して名前を指定し、複数のテーブルまたはビューを返すことや、% を使用してすべてのテーブル/ビューを返すことができます。
  - [　　] ー [　　]、[　　]、[　　]、または [　　] を選択します。既定は [　　] です。
  - [SQL　　] ー 使用する SQL ゲートウェイ接続を選択します。
- [　] をクリックします。
- 2 ページ目で、InterSystems IRIS のオブジェクト・プロパティとして使用できるようにするフィールドを指定します。以下のように変更します。
  - 1 つ以上のフィールドをハイライト表示して一重矢印をクリックすると、そのフィールドが 1 つのリストから別のリストに移動します。二重矢印をクリックすると、すべてのフィールドが（選択されているかどうかにかかわらず）1 つのリストから別のリストに移動します。
  - 選択したリストで、上下の矢印を使用して、InterSystems IRIS が所定のクラスに投影するテーブル内のフィールドの順序を変更できます。これは、クラス定義のプロパティの順序には影響しません。
- [　] をクリックします。
- 3 ページ目では、生成されたクラスのプロパティに関する情報を指定します。プロパティごとに、使用可能なすべてのオプションを指定できます。
  - [　　] ー プロパティを読み取り専用にする場合、このチェック・ボックスにチェックを付けます。これにより、プロパティの `ReadOnly` キーワードが制御されます。

    Tipヒント　`[select_all]` チェック・ボックスを使用すると、この列のすべてのチェック・ボックスにチェックを付けたり、外したりできます。
  - [　　] ー このフィールドのデータを含めるオブジェクト・プロパティの名前を指定します。
  - [　　(SQL　　)] ー このプロパティに使用する SQL フィールド名を指定します。これにより、プロパティの `SqlFieldName` キーワードが制御されます。
- [　] をクリックします。
- 最後のページで、以下を指定します。

- [　　] ― 提供されたリストから新規 InterSystems IRIS テーブルの主キーを選択します。提供された既定のキーのほかにも、[参照] ボタンをクリックして 1 つまたは複数の列を選択できます。複数の列を選択できますが、複数の列は、コンマで区切られた複合キーとして返されます。主キーを指定する必要があります。
- [　　] ― パッケージを含め、作成する InterSystems IRIS クラスの名前を指定します。既定のパッケージ名は `nullschema` です。
- [　　] ― InterSystems IRIS で作成する SQL テーブルの名前を指定します。これにより、クラスの `SqlTableName` キーワードが制御されます。

- [　] ボタンをクリックします。[バックグラウンド・ジョブ] ページに、バックグラウンド・タスクのページへのリンクが表示されます。
- [　　] をクリックします。または、バックグラウンド・タスクのページを表示する特定のリンクをクリックします。いずれの場合も、ウィザードによって、処理を実行するバックグラウンド・タスクが開始されます。

ウィザードにより、InterSystems IRIS データベース内に新しいクラス定義が保存され、そのクラス定義がコンパイルされます。データが存在する場合、外部データベースで即時に表示されます (新規に作成された InterSystems IRIS クラス/テーブルに対して SQL クエリを発行することでチェックできます)。これで InterSystems IRIS 内の他の永続クラスと同様に、新規のクラスを使用できます。

注釈　**リンク・テーブルの接続の切断**

設計上、リンク・テーブル・ウィザードによって生成されたコードは、それ自体が開いた接続を切断しないようになっています。これにより、接続を共有する SQL 文間の競合などの問題が回避されます。詳細は、“SQL ゲートウェイ接続の制御”を参照してください。

2.3.2 リンクされたテーブルを使用する場合の制限事項

通常どおり、接続先のデータベースに固有の制限事項 (構文上など) および要件を理解しておくことが重要です。いくつかの例を以下に示します。

- JDBC 接続では、ODBC とは異なり、異なる複数の INSERT...SELECT 文 (ローカル・テーブルとリンク・テーブルの両方、または 2 つのリンク・テーブルを含む) はサポートされません。
- JDBC とは異なり、ODBC 接続では %ROWID プロパティには入力しません。
- Informix：リンクされた Informix テーブルに基づくビューを InterSystems SQL 内で作成することはできません。これは、Informix では生成された SQL が有効でないためです。
- Sybase：クエリ処理の一環として、InterSystems SQL は外部結合の式を同等の正規化された形式に変換できます。リンクされたテーブルにアクセスする場合、この形式を SQL として再構築するために SQL92 標準の CROSS JOIN 構文が必要になることがあります。Sybase は SQL92 標準の CROSS JOIN をサポートしていないため、リンクされた Sybase テーブルに対して外部結合を使用する一部のクエリは実行に失敗します。
- Oracle：Oracle ソース・テーブルで使用される COUNT 集約に CAST を実行する必要があります。
- MySQL：INOUT と OUT でストアド・プロシージャにアクセスすると、結合したパラメータは正しく更新されません。代わりに、値は結果セットで返されます。

リンクされたテーブルを使用する前に、そのテーブルに対して生成されたクエリ・キャッシュを検証して、使用しているデータベースで構文が有効であることを確認することをお勧めします。リンクされた特定のテーブルに対するクエリ・キャッシュを確認するには、以下の操作を実行します。

- 管理ポータルで、**[システム・エクスプローラ]**、[SQL] の順に移動します。
- 対象のネームスペースをクリックします。
- プルダウン・リストからスキーマを選択します。

- 対象のテーブルが含まれるパッケージの [ ] をクリックします。このパッケージに対するクエリ・キャッシュのテーブルが表示されます。[ ] 列には、完全なクエリが表示されます。
- オプションで、クエリのリンクをクリックして、その他の詳細を表示します。

2.4 プロシージャへのリンク作成ウィザード：ストアド・プロシージャへのリンク

管理ポータルには、ODBC 準拠または JDBC 準拠のデータベースで定義されたストアド・プロシージャへのリンクに使用できるウィザードがあります。プロシージャにリンクすると、メソッドおよびそのメソッドを含むクラスが生成されます。ストアド・プロシージャにリンクする場合、そのストアド・プロシージャが実行するものと同じ操作を実行するクラス・メソッドを作成します。このメソッドは、`SqlProc` キーワードでマークします。このクラス・メソッドは、新しいクラス内に生成されます。クラス名やパッケージ名などの情報を指定できます。このメソッドは、可変個数の引数を受け取ることはできません。既定のパラメータは許可されますが、ストアド・プロシージャのシグニチャは固定されています。

注釈 **リンク・プロシージャの接続の切断**

設計上、リンク・プロシージャ・ウィザードによって生成されたコードは、それ自体が開いた接続を切断しないようになっています。これにより、接続を共有する SQL 文間の競合などの問題が回避されます。詳細は、“SQL ゲートウェイ接続の制御”を参照してください。

- 外部データベースへの接続をまだ作成していない場合は、開始する前に作成してください（“外部ソースへの SQL ゲートウェイ接続の作成”を参照）。
- 管理ポータルで、**[システム・エクスプローラ]**、[SQL] の順に選択します。ページ上部の **[切り替え]** オプションを使ってネームスペースを選択します。利用可能なネームスペースのリストが表示されます。

  ページ上部の [ ] ドロップダウン・リストをクリックし、**[プロシージャのリンク]** を選択します。
- ウィザードの最初のページで、以下のように 1 つまたは複数のプロシージャを選択します。
  - [ ] – データのコピー先の InterSystems IRIS ネームスペースを選択します。
  - [ ] – プロシージャを含むスキーマ（クラス・パッケージ）名を指定します。ワイルドカードを使用して名前を指定し、複数のスキーマを返すことや、% を使用してすべてのスキーマを返すことができます。例えば、C% は、先頭文字が C であるネームスペース内のすべてのスキーマを返します。選択元となるスキーマの返されるリストが短くなることでロード速度が向上するため、このフィルタを使用することをお勧めします。
  - [ ] – リンク先となるプロシージャを指定します。ワイルドカードを使用して名前を指定し、複数のプロシージャを返すことや、% を使用してすべてのプロシージャを返すことができます。複数のプロシージャを選択できます。複数の項目を選択する場合、[ ] をクリックすると、次の画面にパッケージ名の入力を求めるプロンプトが表示されます。クラスを格納するパッケージの名前を指定して、[ ] をクリックします。
  - [SQL] – 使用する SQL ゲートウェイ接続を選択します。
- [ ] をクリックします。
- 2 ページ目で、InterSystems IRIS で生成するクラスに関する詳細を指定します。
  - [ ] – クラスを含めるパッケージの名前を指定します。
  - [ ] – 生成するクラスの名前を指定します。
  - [ ] – プロシージャの名前、具体的には、メソッドの `SqlName` キーワードを制御するプロシージャの名前を指定します。

- [ ] ー 生成するメソッドの名前を指定します。
- [ ] ー オプションで、メソッドの説明を入力します。これは、クラス定義のコメントとして使用され、クラス・リファレンスに表示されます。

- [ ] ボタンをクリックします。[バックグラウンド・ジョブ] ページに、バックグラウンド・タスクのページへのリンクが表示されます。
- [ ] をクリックします。または、バックグラウンド・タスクのページを表示する特定のリンクをクリックします。いずれの場合も、ウィザードによって、処理を実行するバックグラウンド・タスクが開始されます。

ウィザードにより、InterSystems IRIS データベース内に新しいクラス定義が保存され、そのクラス定義がコンパイルされます。

注釈　このウィザードでは、制御対象のクラス名およびクラス・メンバ名を使用して ObjectScript コードが生成されます。このウィザードを使用する場合は、長さ制限も含め、ObjectScript 識別子のルールに従う必要があります (″クラスの定義と使用″ の ″名前付け規約″ のセクションを参照)。

2.5 SQL ゲートウェイ接続の制御

場合によっては、外部テーブルまたはストアド・プロシージャをリンクするコードによって作成された接続の管理が必要になります (“リンク・テーブル・ウィザード” および “リンク・プロシージャ・ウィザード” を参照)。SQL ゲートウェイ接続は、%SYSTEM.SQLGateway クラスによって管理できます。このクラスには、以下のようなメソッドがあります。

- DropAll() ー 開いているすべての接続を中断し、SQL ゲートウェイ・ライブラリをアンロードします。
- DropConnection() ー 指定された JDBC または ODBC 接続を切断します。
- TestConnection() ー 前に定義した SQL ゲートウェイ接続をテストし (“外部ソースへの SQL ゲートウェイ接続の作成” を参照)、診断の出力を現在のデバイスに書き込みます。
- 接続の確立およびトランザクションの制御のためのさまざまなメソッド。全詳細は、%SYSTEM.SQLGateway クラス・ドキュメントを参照してください。

これらのメソッドは、特別な $SYSTEM オブジェクトを使用して呼び出すことができます。例えば、以下のコマンドは、前に定義した "MyConnectionName" という名前の SQL ゲートウェイ接続を切断します。

```
do $system.SQLGateway.DropConnection("MyConnectionName")
```

SQL ゲートウェイ接続名の大文字と小文字は区別されます。

3

JDBC 経由での SQL ゲートウェイへの接続

この章では、SQL ゲートウェイ用の JDBC 論理接続の定義の作成方法を説明します。InterSystems JDBC ドライバの詳細は、"InterSystems ソフトウェアでの Java の使用法" を参照してください。

InterSystems IRIS では、外部データ・ソースへの接続の論理名である、SQL ゲートウェイ接続定義のリストが管理されます。各接続定義は、論理名 (InterSystems IRIS 内で使用するため)、データ・ソースへの接続情報、および接続を確立するときに使用するユーザ名とパスワードで構成されます。これらの接続は、**%Library.sys_SQLConnection** テーブルに格納されます。このテーブルのデータをエクスポートして、同じバージョンの InterSystems IRIS の別のインスタンスにインポートできます。

注釈 **SQL ゲートウェイのログおよびその他の JDBC 設定の制御**

JDBC 経由での接続時の問題を監視するために、SQL ゲートウェイ接続に対して JDBC ログを有効化できます ("SQL ゲートウェイのログ" を参照)。同じダイアログで、**JAVAHOME** およびその他の JDBC 設定も指定できます。

3.1 管理ポータルでの論理接続の定義

JDBC 準拠のデータ・ソースに SQL ゲートウェイ接続を定義するには、以下の手順を実行します。

1. 管理ポータルで、**System Administration** > **Configuration** > **Connectivity** > **SQL Gateway Connections** ページに移動します。
2. **[新規接続作成]** をクリックします。
3. **[SQL ゲートウェイ接続]** ページで、以下のフィールドの値を入力または選択します。
   - **[タイプ]** で **[JDBC]** を選択します。
   - **[接続名]** – InterSystems IRIS 内で使用する、接続の識別子名を指定します。
   - **[ユーザ]** – 必要に応じて、接続の確立時に既定値として機能するアカウントの名前を指定します。
   - **[パスワード]** – 既定のアカウントに関連付けられたパスワードを指定します。
   - **[ドライバ名]** – JDBC クライアント・ドライバの完全クラス名。
   - **[URL]** – 使用している JDBC クライアント・ドライバに必要な形式によるデータ・ソースの接続 URL。
   - **[クラスパス]** – ロードする追加の JAR ファイルのコンマ区切りのリストを指定します。
   - **[プロパティ]** – ベンダ固有の接続プロパティを指定する文字列 (オプション)。これを指定した場合、この文字列の形式は以下のようになります。

property= value; property= value;...

接続プロパティの詳細は、"InterSystems JDBC 接続プロパティ" を参照してください。

例えば、一般的な接続は以下の値を使用します。

設定	値
[タイプ]	JDBC
[接続名]	ConnectionJDBC1
[ユーザ]	JDBCUser
[パスワード]	JDBCPassword
[ドライバ名]	oracle.jdbc.driver.OracleDriver
[URL]	jdbc:oracle:thin:@//oraserver:1521/SID
[クラスパス]	/fill/path/to/ojdbc14.jar
[プロパティ]	oracle.jdbc.V8Compatibility=true; includeSynonyms=false;restrictGetTables=true

その他のオプションについては、"実装固有の JDBC 接続オプション" を参照してください。

4. オプションで、値が有効かどうかをテストします。テストするには、[テスト接続] ボタンをクリックします。画面に、入力した値を、有効な接続で使用できるかどうかを示すメッセージが表示されます。
5. 名前付き接続を作成するには、[保存] をクリックします。
6. [閉じる] をクリックします。

3.2 ネームスペース間の接続の作成

InterSystems IRIS は、JDBC ドライバを備えており、JDBC データ・ソースとして使用できます。つまり、InterSystems IRIS インスタンスは、それ自体または別の InterSystems IRIS インスタンスに JDBC および SQL ゲートウェイ経由で接続できます。具体的には、この接続は、1 つの InterSystems IRIS のネームスペースから別の InterSystems IRIS のネームスペースへの接続です。この方法で接続する場合は、他の外部データベースに接続するときと同じように、使用するデータベース・ドライバに関する接続詳細が必要です。このセクションでは基本的な情報を示します。

3.2.1 SQL ゲートウェイの JDBC データ・ソースとしての使用

別の異なるインスタンス (IrisDB-2) を JDBC データ・ソースとして使用するように InterSystems IRIS インスタンス (IrisDB-1) を構成するには、以下の操作を実行します。

1. IrisDB-1 内で、SQL ゲートウェイを使用して、使用する IrisDB-2 のネームスペースへの JDBC 接続を作成します。
   - [タイプ] で [JDBC] を選択します。
   - [接続名] – IrisDB-1 内で使用する、接続の識別子名を指定します。
   - [ユーザ] – 必要に応じて IrisDB-2 へのアクセスに必要なユーザ名を指定します。
   - [パスワード] – このユーザのパスワードを指定します。
   - [ドライバ名] – com.intersystems.jdbc.IRISDriver を使用します。

- [URL] ー データ・ソースの接続 URL を以下の形式で指定します。

  ```
  jdbc:IRIS://IP_address:port/namespace
  ```

  ここで、IP_address:port は、IrisDB-2 が実行されている IP アドレスおよび TCP ポートです。namespace は、接続先のネームスペースです (“JDBC 接続 URL の定義”を参照してください)。

例えば、一般的な接続は以下の値を使用します。

設定	値
[タイプ]	JDBC
[接続名]	ConnectUser
[ユーザ]	_SYSTEM
[パスワード]	SYS
[ドライバ名]	com.intersystems.jdbc.IRISDriver
[URL]	jdbc:IRIS://127.0.0.1:1972/User

- [クラスパス] ー これは空白のままにしておきます。
- [プロパティ] ー InterSystems JDBC ドライバでサポートされている接続プロパティを指定する文字列 (オプション) です。これを指定した場合、この文字列の形式は以下のようになります。

  property= value; property= value;...

その他のオプションについては、このセクションの後半の“実装固有の JDBC 接続オプション”を参照してください。

2. [保存] をクリックします。
3. [閉じる] をクリックします。

3.3 実装固有の JDBC 接続オプション

SQL ゲートウェイ接続を定義する前に、外部データベースとデータベース・ドライバの要件を理解しておく必要があります。これらの要件が接続の定義方法に影響を及ぼすためです。

デフォルトで区切り識別子を使用しない

[デフォルトで区切り識別子を使用しない] オプションは、生成されたルーチン内の識別子の形式を制御します。

SQL 区切り識別子をサポートしないデータベースを使用する場合は、このチェック・ボックスにチェックを付けます。これには現在、以下のデータベースが含まれます。

- Sybase
- Informix
- MS SQL Server

他のデータベースを使用する場合は、チェック・ボックスのチェックを外します。すべての SQL 識別子が区切られます。

COALESCE 使用

[COALESCE 使用] オプションは、クエリにパラメータ (?) が含まれる場合のクエリの処理方法を制御します。これは、クエリ・パラメータが NULL の場合にのみ適用されます。

- [COALESCE 使用] を選択しない場合に、クエリ・パラメータが NULL のとき、クエリは対応する値が NULL となっているレコードのみを返します。例えば、以下の形式のクエリを考えてみます。

  ```
  SELECT ID, Name from LinkedTables.Table WHERE Name %STARTSWITH ?
  ```

  指定されたパラメータが NULL のとき、クエリは、値が NULL の名前を持つ行のみを返します。

- [COALESCE 使用] を選択した場合、クエリは、COALESCE 関数呼び出し内の各パラメータを折り返します。これにより、NULL 値の処理方法が制御されます。

  ここで、クエリ・パラメータが NULL のとき、クエリは基本的にはパラメータをワイルドカードとして扱います。前の例では、指定されたパラメータが NULL のとき、このクエリはすべての行を返します。

このオプションを選択するかどうかは、ユーザの判断と COALESCE 関数が外部データベースでサポートされているかどうかによります。

外部データベースで COALESCE 関数がサポートされているかどうかを確認するには、そのデータベースのドキュメントを参照してください。

複合 Row ID 内で変換

[複合 Row ID 内で変換] オプションでは、複合 ID を形成するときの文字以外の値の処理方法を制御します。データベースでサポートされているオプションを選択します。

- [非文字値を変換しない] – このオプションでは、変換は実行されません。このオプションは、データベースで文字以外の値の文字値への連結がサポートされる場合にのみ適切です。
- [CAST を使用する] – このオプションでは、CAST を使用して、文字以外の値が文字値に変換されます。
- [{fn convert ...} を使用]– このオプションでは、`{fn convert ...}` を使用して、文字以外の値が文字値に変換されます。

すべての場合に、ID (または変換後の ID) の間に || を使用して ID が連結されます。

外部データベースでサポートされるオプションを確認するには、外部データベースのドキュメントを参照してください。

3.4 SQL ゲートウェイのログ

JDBC 接続の使用時に、SQL ゲートウェイのログを生成できます。このログを有効にするには、以下の操作を実行します。

- 管理ポータルを開いて、System Administration > Configuration > External Language Servers に移動します。
- %JDBC_Server リンクを選択して [編集] ダイアログを表示します (注意 : %Java_Server リンクを選択しないでください。これは、まったく別のリンクです)。
- 詳細設定がまだ表示されていない場合は、[編集] ダイアログの下部で Advanced Settings [表示] リンクを選択します。

- SQL ゲートウェイとデータベース間の相互作用を記録する [**ログファイル**] の名前を指定します (**jdbcSqlGateway.log** など)。完全修飾パスを指定しない場合、ログ・ファイルは JDBC SQL ゲートウェイが最初に起動されたときからの現在のディレクトリに存在します (多くの場合、**/mgr** または **/mgr/namespace** ディレクトリ)。
- ダイアログ上部にある [保存] ボタンを選択します。
- 外部サーバのメイン・ページで、%JDBC_Server 接続をシャットダウンしてから再起動して、新しい設定を有効にします。

[編集] ダイアログで [**Javaホームディレクトリ**] フィールドを設定して、SQL ゲートウェイで使用する Java バージョン (**JAVAHOME**) を指定することもできます。これらの設定の詳細は、"構成パラメータ・ファイル・リファレンス" の "%JDBC Server" を参照してください。

注釈　ログは、トラブルシューティングを実行する必要がある場合のみ有効にします。ログを有効にするとパフォーマンスが大幅に低下するため、通常の操作時は有効にしないでください。

4

ODBC 経由での SQL ゲートウェイへの接続

この章では、SQL ゲートウェイ用の ODBC 論理接続の定義の作成方法、およびデータ移行ウィザードの使用方法を説明します。InterSystems ODBC の使用方法の詳細は、"InterSystems ODBC ドライバの使用法" を参照してください。

以下の項目について説明します。

- 外部ソースへの接続の作成 – SQL ゲートウェイ用の ODBC 論理接続の定義の作成方法を説明します。
- [データ移行ウィザード] の使用法 – 外部の ODBC ソースからデータを移行し、適切な InterSystems IRIS クラス定義を作成してデータを格納する方法について説明します。

4.1 外部ソースへの接続の作成

InterSystems IRIS では、外部データ・ソースへの接続の論理名である、SQL ゲートウェイ接続定義のリストが管理されます。各接続定義は、論理名 (InterSystems IRIS 内で使用するため)、データ・ソースへの接続情報、および接続を確立するときに使用するユーザ名とパスワードで構成されます。これらの接続は、%Library.sys_SQLConnection テーブルに格納されます。このテーブルのデータをエクスポートして、同じバージョンの InterSystems IRIS の別のインスタンスにインポートできます。

このセクションで説明する項目は以下のとおりです。

- 管理ポータルでの論理接続の定義
- SQL ゲートウェイの ODBC データ・ソースとしての使用
- 実装固有の ODBC 接続オプション

注釈　DSN を作成するための OS 固有の手順は、"InterSystems ODBC ドライバ の使用法" の以下のセクションを参照してください。

- "Windows での ODBC データ・ソースの定義"
- "UNIX® での ODBC データ・ソースの定義"

4.1.1 管理ポータルでの論理接続の定義

ODBC 準拠のデータ・ソースに接続を定義するには、以下の手順を実行します。

1. 外部データベースの ODBC データ・ソース名 (DSN) を定義します (手順は、データベースのドキュメントで説明されています)。

2. 管理ポータルで、System Administration > Configuration > Connectivity > SQL Gateway Connections ページに移動します。
3. **[新規接続作成]** をクリックします。
4. **[ゲートウェイ接続]** ページで、以下のフィールドの値を入力または選択します。
   - **[接続の種類]** で **[ODBC]** を選択します。
   - **[接続名]** – InterSystems IRIS 内で使用する、接続の識別子名を指定します。
   - **[DSN を選択してください]** – 前に作成した DSN を選択します。ODBC SQL ゲートウェイでは DSN を使用しない接続はサポートされていないため、DSN を使用する必要があります。
   - **[ユーザ]** – 必要に応じて、接続の確立時に既定値として機能するアカウントの名前を指定します。
   - **[パスワード]** – 既定のアカウントに関連付けられたパスワードを指定します。

   例えば、一般的な接続は以下の値を使用します。

設定	値
[タイプ]	ODBC
[接続名]	ConnectionODBC1
[DSN を選択してください]	MyAccessPlayground
[ユーザ]	DBOwner
[パスワード]	DBPassword

   その他のオプションについては、このセクションの後半の“実装固有の ODBC 接続オプション”を参照してください。
5. オプションで、値が有効かどうかをテストします。テストするには、**[テスト接続]** ボタンをクリックします。画面に、前の手順で入力した値を、有効な接続で使用できるかどうかを示すメッセージが表示されます。
6. 名前付き接続を作成するには、**[保存]** をクリックします。
7. **[閉じる]** をクリックします。

4.1.2 SQL ゲートウェイの ODBC データ・ソースとしての使用

InterSystems IRIS は、ODBC ドライバを備えており、ODBC データ・ソースとして使用できます。つまり、InterSystems IRIS インスタンスは、それ自体または別の InterSystems IRIS インスタンスに ODBC および SQL ゲートウェイ経由で接続できます。具体的には、この接続は、1 つの InterSystems IRIS のネームスペースから別の InterSystems IRIS のネームスペースへの接続です。この方法で接続する場合は、他の外部データベースに接続するときと同じように、使用するデータベース・ドライバに関する接続詳細が必要です。このセクションでは基本的な情報を示します。

別の InterSystems IRIS インスタンス (InterSystems IRIS_B) を ODBC データ・ソースとして使用するように InterSystems IRIS インスタンス (InterSystems IRIS_A) を構成するには、以下の操作を実行します。

1. InterSystems IRIS_A を実行しているマシン上で、使用する InterSystems IRIS_B のネームスペースを表す DSN を作成します

   Tipヒント InterSystems IRIS をインストールするとインストーラによって自動的に DSN が作成されるため、InterSystems IRIS_B をこのマシンにインストールしたときに、既に適切な DSN が使用可能になっている場合があります。

2. InterSystems IRIS_A 内で、SQL ゲートウェイを使用して、DSN を使用する ODBC 接続を作成します。以下の詳細を指定します。
   - **[タイプ]** で **[ODBC]** を選択します。
   - **[接続名]** ー InterSystems IRIS_A 内で使用する、接続の識別子を指定します。
   - **[DSN を選択してください]** ー InterSystems IRIS_B で前に作成した DSN を選択します。

   例えば、一般的な接続は以下の値を使用します。

設定	値
[タイプ]	ODBC
[接続名]	TestConnection
[DSN を選択してください]	TestConnection

   Tipヒン **[ユーザ]** と **[パスワード]** の情報は DSN の一部なので、これらを指定する必要はありません。
   ト
3. **[保存]** をクリックします。
4. **[閉じる]** をクリックします。

4.1.3 実装固有の ODBC 接続オプション

SQL ゲートウェイ接続を定義する前に、外部データベースとデータベース・ドライバの要件を理解しておく必要があります。これらの要件が接続の定義方法に影響を及ぼすためです。以下のオプションは、すべてのドライバ実装に適用されるわけではありません。

レガシーな外部結合

[レガシー outer join (Sybase) を有効に] オプションは、接続でレガシーな外部結合の使用を有効にするかどうかを制御します。レガシーな外部結合では、SQL-92 規格より前の SQL 構文が使用されます。外部データベースがこのような結合をサポートしているかどうかを確認するには、そのデータベースのドキュメントを参照してください。

長いデータ長が必要

[長いデータ長が必要] オプションは、接続でデータを結合する方法を制御します。このオプションの値は、データベース・ドライバの SQL_NEED_LONG_DATA_LEN 設定と一致している必要があります。この設定の値を確認するには、ODBC SQLGetInfo 関数を使用します。SQL_NEED_LONG_DATA_LEN が Y の場合は **[長いデータ長が必要]** オプションを選択し、それ以外の場合は選択を解除します。

Unicode ストリームのサポート

[Unicode ストリームのサポート] オプションは、ストリームで接続が Unicode データをサポートするかどうかを制御します。これらは、LONGVARCHAR または LONGVARBINARY タイプのフィールドです。

- Sybase の場合はこのチェック・ボックスのチェックを外します。Sybase データベースを使用している場合は、SQL ゲートウェイ経由でアクセスするすべてのフィールドに UTF-8 データのみが含まれるようにする必要があります。
- その他のデータベースの場合は、このチェック・ボックスにチェックを付けます。

デフォルトで区切り識別子を使用しない

[デフォルトで区切り識別子を使用しない] オプションは、生成されたルーチン内の識別子の形式を制御します。

SQL 区切り識別子をサポートしないデータベースを使用する場合は、このチェック・ボックスにチェックを付けます。これには現在、以下のデータベースが含まれます。

- Sybase
- Informix
- MS SQL Server

他のデータベースを使用する場合は、チェック・ボックスのチェックを外します。すべての SQL 識別子が区切られます。

COALESCE 使用

[COALESCE 使用] オプションは、クエリにパラメータ (?) が含まれる場合のクエリの処理方法を制御します。これは、クエリ・パラメータが NULL の場合にのみ適用されます。

- [COALESCE 使用] を選択しない場合に、クエリ・パラメータが NULL のとき、クエリは対応する値が NULL となっているレコードのみを返します。例えば、以下の形式のクエリを考えてみます。

  ```
  SELECT ID, Name from LinkedTables.Table WHERE Name %STARTSWITH ?
  ```

  指定されたパラメータが NULL のとき、クエリは、値が NULL の名前を持つ行のみを返します。

- [COALESCE 使用] を選択した場合、クエリは、COALESCE 関数呼び出し内の各パラメータを折り返します。これにより、NULL 値の処理方法が制御されます。

  ここで、クエリ・パラメータが NULL のとき、クエリは基本的にはパラメータをワイルドカードとして扱います。前の例では、指定されたパラメータが NULL のとき、このクエリはすべての行を返します。これには、通常の ODBC クライアントの動作との一貫性があります。

このオプションを選択するかどうかは、ユーザの判断と COALESCE 関数が外部データベースでサポートされているかどうかによります。

外部データベースで COALESCE 関数がサポートされているかどうかを確認するには、そのデータベースのドキュメントを参照してください。

複合 Row ID 内で変換

[複合 Row ID 内で変換] オプションでは、複合 ID を形成するときの文字以外の値の処理方法を制御します。データベースでサポートされているオプションを選択します。

- [非文字値を変換しない] ― このオプションでは、変換は実行されません。このオプションは、データベースで文字以外の値の文字値への連結がサポートされる場合にのみ適切です。
- [CAST を使用する] ― このオプションでは、CAST を使用して、文字以外の値が文字値に変換されます。
- [{fn convert ...} を使用] ― このオプションでは、`{fn convert ...}` を使用して、文字以外の値が文字値に変換されます。

すべての場合に、ID (または変換後の ID) の間に || を使用して ID が連結されます。

外部データベースでサポートされるオプションを確認するには、外部データベースのドキュメントを参照してください。

4.2 [データ移行ウィザード] の使用法

管理ポータルには、外部テーブルまたは外部ビューからのデータの移行に使用できるウィザードがあります。

外部ソースのテーブルまたはビューからデータを移行する場合、そのテーブルまたはビューのデータを格納する永続クラスが生成され、データがコピーされます。このウィザードには、クラスの名前が移行元のテーブルまたはビューの名前と同じであり、同様にプロパティ名もテーブルやビューのものと同じであるという前提があります。クラスが生成されると、そのクラスには、外部データ・ソースとの接続はありません。

- 外部データベースへの SQL ゲートウェイ接続をまだ作成していない場合は、開始する前に作成してください (“外部ソースへの SQL ゲートウェイ接続の作成” を参照)。
- 管理ポータルで、**[システム・エクスプローラ]**、[SQL] の順に選択します。ページ上部の **[切り替え]** オプションを使ってネームスペースを選択します。利用可能なネームスペースのリストが表示されます。

  ページ上部の [ ] ドロップダウン・リストをクリックし、**[データ移行]** を選択します。
- ウィザードの最初のページで、以下のようにテーブルまたはビューを選択します。
  - [ ] ー データのコピー先の InterSystems IRIS ネームスペースを選択します。
  - [ ] ー テーブルまたはビューを含むスキーマ (クラス・パッケージ) 名を指定します。ワイルドカードを使用して名前を指定し、複数のスキーマを返すことや、% を使用してすべてのスキーマを返すことができます。例えば、C% は、先頭文字が C であるネームスペース内のすべてのスキーマを返します。選択元となるスキーマの返されるリストが短くなることでロード速度が向上するため、このフィルタを使用することをお勧めします。
  - [ ] ー テーブルまたはビューの名前を指定します。ワイルドカードを使用して名前を指定し、複数のテーブルまたはビューを返すことや、% を使用してすべてのテーブル/ビューを返すことができます。
  - [ ] ー [ ]、[ ]、[ ]、または [ ] を選択します。既定は [ ] です。
  - [SQL] ー 使用する SQL ゲートウェイ接続を選択します。
- [ ] をクリックします。
- 次のページでは、必要に応じて、クラスごとに以下の情報を指定することができます。
  - [ ] ー クラスを含むパッケージを指定します。長さ制限も含め、ObjectScript 識別子のルールに従う必要があります (”クラスの定義と使用” の ”名前付け規約” のセクションを参照)。

    Tipヒント: すべてのクラスのパッケージ名を変更するには、この列の一番上に値を入力し、[ ] をクリックします。
  - [ ] ー 外部ソースのテーブル定義に基づいてこのクラスを生成する場合は、このチェック・ボックスにチェックを付けます。クラスが既に生成されている場合は、このチェック・ボックスのチェックを外すことができます。
  - [ ] ー 外部ソースからこのクラスのデータをコピーする場合は、このチェック・ボックスにチェックを付けます。データをコピーするときに、ウィザードによって、InterSystems IRIS クラスの既存データが上書きされます。
- [ ] をクリックします。ウィザードに以下のオプション設定が表示されます。
  - [ ] ー チェックを付けると、INSERT コマンドの restriction パラメータで %NOCHECK が指定されて、データがインポートされます。
  - [ ] ー チェックを付けると、データ移行 (システム規模ではない移行) の実行プロセスについてジャーナリングが無効化されます。これにより、移行を高速化できますが、システム障害で移行が

中断された場合、移行されたデータが不確定の状態で残される可能性があります。ジャーナリングは、実行が成功したかどうかにかかわらず、実行終了時に再有効化されます。

- [ ] ー チェックを付けると、データの挿入後にインデックスが構築されます。ウィザードはクラスの %SortBegin() メソッドをテーブルへのデータ挿入の前に呼び出します。これにより、インデックス・エントリが並べ替えのための一時的な場所に書き込まれます。すべての行が挿入された後でウィザードが %SortEnd() を呼び出すと、インデックス・エントリは実際のインデックス位置に書き込まれます。テーブルにユニーク・インデックスが定義されており、一意の制約に対する違反を移行時にすべて検出したい場合は、[インデックスを延期] を使用しないでください。[インデックスを延期] を使用すると、一意の制約に対する違反は検出されません。
- [ ] ー チェックを付けると、INSERT コマンドの restriction パラメータで %NOTRIGGER が指定されて、データがインポートされます。
- [ ] ー チェックを付けると、既存のデータは新しいデータにマージされずに削除されます。

- [ ] ボタンをクリックします。新しいウィンドウが開き、[バックグラウンドジョブ] ページに、バックグラウンド・タスクのページへのリンクが表示されます。直ちにインポートを開始する場合は [ ] をクリックし、バックグラウンド・タスクのページを表示する場合は特定のリンクをクリックします。いずれの場合も、ウィザードによって、インポートはバックグラウンド・タスクとして開始されます。
- [データ移行ウィザード] ウィンドウで [ ] をクリックして、管理ポータルのホーム・ページに戻ります。

注釈　%SQL.Migration.Import クラスには、データ移行ウィザードのラッパが含まれます。詳細は、クラス・ライブラリのドキュメントを参照してください。

4.2.1 Microsoft Access と外部キーの制約

データ移行ウィザードを Microsoft Access で使用する場合、Access のテーブルに定義されている外部キー制約のコピーが試行されます。これを実行するために、ウィザードでは、Access の MSysRelationships テーブルをクエリします。既定では、このテーブルは非表示であり、読み取りアクセス権がありません。ウィザードが MSysRelationships にアクセスできないと、ウィザードは、外部キー制約なしでデータ・テーブル定義を InterSystems SQL に移行します。

このユーティリティによって外部キー制約をテーブル定義と共に移行するには、以下のように Microsoft Access で MSysRelationships に対する読み取りアクセスを付与するよう設定する必要があります。

- Microsoft Access で、そのシステム・オブジェクトが表示されることを確認します。
- [ ] [ ] をクリックして、[ ] タブで設定を選択します。
- [ ] [ ] [/] をクリックします。テーブル名の隣の [ ] チェック・ボックスにチェックを付けます。

5
プログラムによる SQL ゲートウェイの使用法

注釈　このセクションでは、読者が ODBC API 呼び出しの使用経験が豊富であると想定しており、ODBC 関数の詳しい使用方法は説明しません。問題が発生した場合、InterSystems IRIS と ODBC の両方のログを有効化して SQL ゲートウェイを監視できます (″InterSystems ODBC ドライバ の使用法″ の “ログと環境変数” の章を参照してください)。

標準 SQL ゲートウェイ・ウィザードで提供されないオプションが必要な場合、%Library.SQLGatewayConnection クラスを使用して ObjectScript から ODBC 関数を呼び出すことができます。ダイナミック・クエリ (結果セットを取得する) を実行することも、低レベルの ODBC プログラミングを実行することもできます。この章で説明する項目は以下のとおりです。

- FetchSamples の例 ー 接続を開き、クエリを実行し、結果セットにアクセスする簡単なプログラムを紹介します。
- 外部データ・セットの作成および使用 ー %SQL.Statement メソッドを使用してクエリを実行しデータ・セットにアクセスする方法について説明します。
- ODBC 関数の直接呼出し ー %SQL.Statement を使用せずに ODBC クエリ関数を直接呼び出す方法について説明します。
- %SQLGatewayConnection のクイック・リファレンス ー サポートされるメソッドおよびプロパティの詳細を示します。

この章の残りの部分では、%Library.SQLGatewayConnection を、その略名である %SQLGatewayConnection で参照します。

5.1 FetchSamples の例

以下に、接続を開いてクエリの準備と実行を行い結果のデータ・セットにアクセスする簡単な例を示します。Connect()、Disconnect()、ConnectionHandle、および sqlcode の詳細は、“%SQLGatewayConnection のクイック・リファレンス” のエントリを参照してください。サポートされている ODBC 関数およびそれらの関数を呼び出す %SQLGatewayConnection メソッドのリストは、このクイック・リファレンスの “サポートされる ODBC 関数呼び出し” のセクションを参照してください。

ClassMethod FetchSamples

Class Member

```
ClassMethod FetchSamples()
{
   #include %occInclude
   //Create new SQL Gateway connection object
   set gc=##class(%SQLGatewayConnection).%New()
   if gc=$$$NULLOREF quit $$$ERROR($$$GeneralError,"Cannot create %SQLGatewayConnection.")

   //Make connection to target DSN
```

```
    set pDSN="Cache Samples"
    set usr="_system"
    set pwd="SYS"
    set sc=gc.Connect(pDSN,usr,pwd,0)
    if $$$ISERR(sc) quit sc
    if gc.ConnectionHandle="" quit $$$ERROR($$$GeneralError,"Connection failed")

    set sc=gc.AllocateStatement(.hstmt)
    if $$$ISERR(sc) quit sc

    //Prepare statement for execution
    set pQuery= "select * from Sample.Person"
    set sc=gc.Prepare(hstmt,pQuery)
    if $$$ISERR(sc) quit sc
    //Execute statement
    set sc=gc.Execute(hstmt)
    if $$$ISERR(sc) quit sc
    //Get list of columns returned by query
    set sc=gc.DescribeColumns(hstmt, .columnlist)
    if $$$ISERR(sc) quit sc

    //display column headers delimited by ":"
    set numcols=$listlength(columnlist)-1  //get number of columns
    for colnum=2:1:numcols+1 {
       Write $listget($listget(columnlist,colnum),1),":"
    }
    write !

    //Return first 200 rows
    set sc=gc.Fetch(hstmt)
    if $$$ISERR(sc) quit sc
    set rownum=1
    while((gc.sqlcode'=100) && (rownum<=200)) {
       for ii=1:1:numcols {
          set sc=gc.GetData(hstmt, ii, 1, .val)
          write " "_val
          if $$$ISERR(sc) break
       }
       set rownum=rownum+1
       write !
       set sc=gc.Fetch(hstmt)
       if $$$ISERR(sc) break
    }

    //Close cursor and then disconnect
    set sc=gc.CloseCursor(hstmt)
    if $$$ISERR(sc) quit sc

    set sc=gc.Disconnect()
    quit sc
}
```

5.2 外部データ・セットの作成および使用

外部データベースをクエリするデータ・セットを作成および使用するには、以下の操作を実行します。

1. %New() メソッドを使用して、%SQLGatewayConnection のインスタンスを作成します。
2. 作成したインスタンスの Connect() メソッドを呼び出し、ODBC データ・ソース名を指定する引数、および必要に応じて、ソースへのログインに必要なユーザ名とパスワードを渡します。

   Connect() メソッドには、以下のシグニチャがあります。

   ```
   method Connect(dsn, usr, pwd, timeout) as %Status
   ```

   ここで dsn は、データ・ソースの DSN です。usr は、データ・ソースにログインできるユーザです。pwd は対応するパスワードです。timeout は、接続を待機する長さを指定します。
3. %New() メソッドを使用して %ResultSet のインスタンスを作成し、文字列引数 "%DynamicQueryGW:SQLGW" を指定します。

   注釈　これは、通常のダイナミック・クエリ（"%DynamicQuery:SQL"）で使用する引数とは若干異なります。

4. 結果セットの Prepare() メソッドを呼び出します。最初の引数は SQL クエリを構成する文字列、2 番目の引数は省略、3 番目の引数は %SQLGatewayConnection のインスタンスである必要があります。
5. オプションとして、クエリで必要とする順序で引数を指定することにより、結果セットの Execute() メソッドを呼び出します。このメソッドはステータスを返すので、それを確認する必要があります。

結果セットを使用する場合は、通常、一度に 1 行ずつ検証します。%ResultSet のメソッドを使用して、指定した列の値などの情報を取得します。以下の例で示しているように、通常、Next() を使用してすべての行で反復を行います。

例

Class Member

```
ClassMethod SelectAndWrite() as %Status
{
    Set conn=##class(%SQLGatewayConnection).%New()
    Set sc=conn.Connect("AccessPlayground","","")
    If $$$ISERR(sc) do $System.Status.DisplayError(sc) quit

    Set res=##class(%ResultSet).%New("%DynamicQueryGW:SQLGW")
    Set sc=res.Prepare("SELECT * FROM PEOPLE",,conn)
    If $$$ISERR(sc) do $System.Status.DisplayError(sc) quit

    Set sc=res.Execute()
    If $$$ISERR(sc) do $System.Status.DisplayError(sc) quit

    While res.Next()
    { Write !,res.GetData(1)," ",res.GetData(2)," ",res.GetData(3)
    }
    Set sc=conn.Disconnect()
    Quit sc
}
```

%ResultSet の詳細は、"InterSystems SQL の使用法" の "ダイナミック SQL の使用" の章を参照してください。%ResultSet のクラス・ドキュメントも参照してください。

5.3 ODBC 関数の直接呼出し

%SQL.Statement で十分に制御できない場合は、%SQLGatewayConnection クラスを使用して ODBC に直接アクセスすることができます。このクラスは、ODBC 関数に対応する一連のメソッド ("サポートされる ODBC 関数呼び出し" を参照) およびその他のユーティリティ関数を提供します。ODBC 準拠のデータベースに接続して、これを使用できます。その後、低レベルの ODBC プログラミングを実行できます。全手順は、以下のとおりです。

1. %New() メソッドを使用して、%SQLGatewayConnection のインスタンスを作成します。
2. 作成したインスタンスの Connect() メソッドを呼び出し、ODBC データ・ソース名を指定する引数、および必要に応じて、ソースへのログインに必要なユーザ名とパスワードを渡します。
3. AllocateStatement() メソッドを呼び出して、文ハンドルを (参照により) 受け取ります。
4. この文ハンドルを引数として使用して、SQL ゲートウェイ・インスタンスの他のメソッドを呼び出します。これらのメソッドのほとんどが、ODBC 関数を呼び出します。

以下の簡単な例は、この手順を示しています。これは、前のセクションの例に似ていますが、%SQL.Statement メソッドを使用するのではなく、%SQLGatewayConnection バージョンの Prepare() および Execute() を使用して、ODBC クエリ関数 SQLPrepare() および SQLExecute() を直接呼び出します。

%SQLGatewayConnection メソッドを使用したクエリの実行

Class Member

```
ClassMethod ExecuteQuery(mTable As %String)
{
   set mDSN="DSNtest"
   set mUsrName="SYSDBA"
   set mUsrPwd="masterkey"

   // Create an instance and connect
   set gateway=##class(%SQLGatewayConnection).%New()
   set status=gateway.Connect(mDSN,mUsrName,mUsrPwd)
   if $$$ISERR(status) do $System.Status.DisplayError(status) quit $$$ERROR()
   set hstmt=""

   // Allocate a statement
  set status=gateway.AllocateStatement(.hstmt)
   if $$$ISERR(status) do $System.Status.DisplayError(status) quit $$$ERROR()

   // Use %SQLGatewayConnection to call ODBC query functions directly
   set status=gateway.Prepare(hstmt,"SELECT * FROM "_mTable)
   if $$$ISERR(status) do $System.Status.DisplayError(status) quit $$$ERROR()
   set status=gateway.Execute(hstmt)

   if $$$ISERR(status) do $System.Status.DisplayError(status) quit $$$ERROR()
   quit gateway.Disconnect()
}
```

注釈 NULL 値および空の文字列

この章で説明するメソッドを使用する場合、InterSystems IRIS と SQL との間には以下の重要な相違点があることに留意してください。

- SQL では、"" は空の文字列を表します。
- InterSystems IRIS では、"" は NULL です。
- InterSystems IRIS では、$char(0) が空の文字列です。

5.4 %SQLGatewayConnection のクイック・リファレンス

- %SQLGatewayConnection API の概要
- %SQLGatewayConnection のメソッドおよびプロパティ
- サポートされる ODBC 関数呼び出し

5.4.1 SQLGatewayConnection API の概要

%SQLGatewayConnection クラスには、外部データ・ソースへの接続の管理、ステータス情報の確認、および ODBC 共有ライブラリに関する情報の取得に使用できるプロパティとメソッドが用意されています。このリファレンスで取り上げるメソッドとプロパティは、以下に用途別にリストされています (ここにリストされていないメソッドは、“サポートされる ODBC 関数呼び出し” を参照してください)。

接続の管理

%SQLGatewayConnection クラスには、外部データ・ソースへの接続の管理に使用できるプロパティとメソッドが用意されています。

- DSN ― (%String プロパティ) 接続先の ODBC 準拠データ・ソースのデータ・ソース名。
- User ― (%String プロパティ) データ・ソースにログインするためのユーザ名。

- Password ー (%String プロパティ) 関連付けられているパスワード。
- ConnectionHandle ー (%Binary プロパティ) 現在の接続では、ODBC 準拠のデータ・ソースが処理されます。
- Connect() ー DSN への接続を確立します。
- GetConnection() ー DSN、ユーザ名、およびパスワードを指定する構成設定を使用して接続を確立します。
- SetConnectOption() ー ODBC 関数 SQLSetConnectAttr を呼び出します。
- Disconnect() ー 接続を切断します。

スタータスおよびクエリ用のメソッド

%SQLGatewayConnection のメソッドのほとんどはステータスを返すので、それを確認する必要があります。以下のプロパティとメソッドを使用してステータス情報を確認することもできます。

- sqlcode ー (%Integer プロパティ) 前回の呼び出し (ある場合) による SQL コードの戻り値が含まれます。
- GatewayStatus ー (%Integer プロパティ) 前回の呼び出しのステータスを示します。
- GetLastSQLCode() ー この呼び出しが SQL コードを返さない場合は、前回の呼び出しの SQL コードを返します。
- GatewayStatusGet() ー 前回の呼び出しのエラー・コードを返します。

以下のメソッドは、結果セットから行を取得します。

- FetchRows() ー 所定の接続ハンドルに指定されている行数を (参照により) 返します。
- GetOneRow() ー 所定の接続ハンドルの次の行を (参照により) 返します。

以下のメソッドは、結合されたクエリ・パラメータの値を取得および設定します。

- GetParameter() ー 指定したパラメータの現在の値を (参照により) 返します。
- SetParameter() ー 以前に結合したパラメータの値を設定します。

共有ライブラリの使用

%SQLGatewayConnection クラスには、ODBC SQL ゲートウェイで使用する共有ライブラリに関する情報を取得するために呼び出すことが可能なプロパティとメソッドが用意されています。

- DLLHandle ー (%Binary プロパティ) 現在使用されている共有ライブラリのハンドル。これは、接続時に設定されます。
- DLLName ー (%String プロパティ) 現在使用されている共有ライブラリの名前。これは、接続時に設定されます。
- GetGTWVersion() ー 共有ライブラリの現在のバージョンを返します。
- GetUV() ー 共有ライブラリが Unicode として構築されているかどうかを (参照により) 返します。このメソッドは常に $$$OK のステータスを返します。
- UnloadDLL() ー プロセス・メモリから共有ライブラリをアンロードします。

5.4.2 %SQLGatewayConnection のメソッドおよびプロパティ

以下に、選定されたメソッドおよびプロパティのアルファベット順のリストを示します。ここにリストされていないメソッドは、“サポートされる ODBC 関数呼び出し” を参照してください。

AllocateStatement()

ODBC 関数 SQLAllocHandle() を呼び出して、対応する構造を SQL ゲートウェイに作成します。

```
method AllocateStatement(ByRef hstmt) as %Status
```

Connect()

DSN への接続を確立します。

```
method Connect(dsn, usr, pwd, timeout) as %Status
```

ユーザ名とパスワードが両方とも空のとき、このメソッドは ODBC 関数 SQLDriverConnect() を呼び出します。その呼び出しが成功しないとき、またはユーザ名とパスワードを指定したとき、このメソッドは ODBC関数 SQLConnect() を呼び出します。

timeout パラメータが 0 以外の場合、まず、SQLSetConnectAttr() が呼び出されて、SQL_ATTR_LOGIN_TIMEOUT が設定されます。

ConnectionHandle プロパティ

ODBC 準拠データ・ソースに現在の接続ハンドルを提供する %Binary プロパティ。

Disconnect()

接続を切断します。

```
method Disconnect() as %Status
```

DLLHandle プロパティ

現在使用されている共有ライブラリのハンドルを提供する %Binary プロパティ。これは、接続時に設定されます。

DLLName プロパティ

現在使用されている共有ライブラリの名前を提供する %String プロパティ。これは、接続時に設定されます。

DSN プロパティ

接続先の ODBC 準拠データ・ソースのデータ・ソース名を提供する %String プロパティ。

FetchRows()

所定の接続ハンドルに指定されている行数を (参照により) 返します。

```
method FetchRows(hstmt, Output rlist As %List, nrows As %Integer) as %Status
```

ここでは、hstmt がその接続ハンドルです。これは、(参照により) AllocateStatement(). から返されます。また、rlist は、返される行リストです。これは、InterSystems IRIS の $list の 1 つです。リストの各アイテムには行が 1 つ含まれています。データがない場合 (SQL_CODE = 100)、フェッチは成功したと見なされますが、返されるリストは空です。

注意　基本的に、このメソッドはテストに使用すると便利です。文字フィールドを最大 120 文字に切り捨てるので、1 行に収まるフィールド数が多くなります。切り捨てられていないデータが必要な場合は、代わりに GetData() を使用します。

GatewayStatus プロパティ

前回の呼び出しのステータスを提供する %String プロパティ。ステータス値は、以下のいずれかです。

- 0 – 成功
- -1 – SQL エラー
- -1000 – 重大なエラー

GatewayStatusGet()

前回の呼び出しのエラー・コードを返します。

```
method GatewayStatusGet() as %Integer
```

この呼び出しはエラー・コードを初期化しないため、複数回呼び出すことができます。前述の GatewayStatus プロパティに関する注記を参照してください。

GetConnection()

DSN、ユーザ名、およびパスワードを指定する構成ファイル・エントリを使用して接続を確立します。

```
method GetConnection(conn, timeout) as %Status
```

GetGTWVersion()

共有ライブラリの現在のバージョンを返します。

```
method GetGTWVersion() as %Integer
```

GetLastSQLCode()

この呼び出しが SQL コードを返さない場合 (SQLGetData() を使用した場合など) は、前回の呼び出しの SQL コードを返します。

```
method GetLastSQLCode() as %Integer
```

GetOneRow()

所定の接続ハンドルの次の行を (参照により) 返します。

```
method GetOneRow(hstmt, ByRef row) as %Status
```

ここでは、hstmt がその接続ハンドルです。これは、(参照により) AllocateStatement(). から返されます。また、row は返される行で、InterSystems IRIS の **$list** の 1 つです。リストの各アイテムにはフィールドが 1 つ含まれています。データがない場合 (SQL_CODE = 100)、フェッチは成功したと見なされますが、返されるリストは空です。

注意　基本的に、このメソッドはテストに使用すると便利です。文字フィールドを最大 120 文字に切り捨てるので、1 行に収まるフィールド数が多くなります。切り捨てられていないデータが必要な場合は、代わりに GetData() を使用します。

GetParameter()

指定したパラメータの現在の値を (参照により) 返します。

```
method GetParameter(hstmt, pnbr, ByRef value) as %Status
```

ここで、hstmt は AllocateStatement() から (参照により) 返される接続ハンドルで、pnbr はパラメータの順序数です。

GetUV()

共有ライブラリが Unicode として構築されているかどうかを (参照により) 返します。

```
method GetUV(ByRef infoval) as %Status
```

このメソッドは常に $$$OK のステータスを返します。

Password プロパティ

関連付けられているパスワードを提供する %String プロパティ。

SetConnectOption()

ODBC 関数 SQLSetConnectAttr() を呼び出します。

```
method SetConnectOption(opt, val) as %Status
```

整数値のみがサポートされます。opt 引数の整数値は、sql.h ヘッダ・ファイルおよび sqlext.h ヘッダ・ファイルから取得することもできます。

SetParameter()

以前に結合したパラメータの値を設定します。

```
method SetParameter(hstmt, pvalue, pnbr) as %Status
```

ここで、hstmt は AllocateStatement() から (参照により) 返される接続ハンドルで、pvalue は使用する値で、pnbr はパラメータの順序数です。パラメータは、$list 形式で格納されます。割り当てられているバッファが不十分な場合は、新しいバッファが割り当てられます。

sqlcode プロパティ

前回の呼び出し (ある場合) によって返された SQL コードを提供する %Integer プロパティ。

UnloadDLL()

ODBC SQL ゲートウェイの共有ライブラリをプロセス・メモリからアンロードします。

```
method UnloadDLL() as %Status
```

User プロパティ

データ・ソースにログインするためのユーザ名を提供する %String プロパティ。

5.4.3 サポートされる ODBC 関数呼び出し

次の表に、対応する %SQLGatewayConnection メソッドで直接サポートされる ODBC 関数と、それらのメソッドのクラス・ドキュメントへのリンクを示します。メソッドを呼び出して ODBC 関数 SQLPrepare と SQLExecute を起動する例については、“ODBC 関数の直接呼出し”を参照してください。

この章は、これらのメソッドの詳細なリファレンスではありません。メソッドの引数、アクション、および戻り値の詳細は、InterSystems クラス・ライブラリのリファレンスで %SQLGatewayConnection を参照してください。

テーブル 5-1: %SQLGatewayConnection からの ODBC 関数の呼び出し

ODBC 関数	関数を呼び出す ObjectScript メソッド
SQLAllocHandle	AllocateStatement()

ODBC 関数	関数を呼び出す ObjectScript メソッド
SQLBindParameter	BindParameter()、BindParameters()
SQLCloseCursor	CloseCursor()
SQLColAttribute	DescribeColumns()
SQLColumnPrivileges	ColumnPrivileges()、ColumnPrivilegesW()
SQLColumns	Columns()、ColumnsW()
SQLDescribeCols	DescribeColumns()
SQLDescribeParam	DescribeParameters()
SQLDiagRec	GetErrorList()
SQLEndTran	Transact()
SQLExecute	Execute()
SQLFetch	Fetch()
SQLForeignKeys	ForeignKeys()、ForeignKeysW()
SQLFreeHandle	DropStatement()
SQLFreeStmt	UnbindParameters()
SQLGetData	GetData()、GetDataL()、GetDataLW()、GetDataW()
SQLGetInfo	GetInfo()
SQLGetTypeInfo	GetTypeInfo()
SQLMoreResults	MoreResults()
SQLNumParams	DescribeParameters()
SQLParamData	ParamData()
SQLPrepare	Prepare()、PrepareW()
SQLPrimaryKeys	PrimaryKeys()、PrimaryKeysW()
SQLProcedureColumns	DescribeProcedureColumns()、DescribeProcedureColumnsW()
SQLProcedures	DescribeProcedures()、DescribeProceduresW()
SQLPutData	PutData()、PutDataW()
SQLRowCount	RowCount()
SQLSetConnectAttr	SetConnectOption()
SQLSetStmtAttr	SetStmtOption()
SQLSpecialColumns	SpecialColumns()、SpecialColumnsW()
SQLStatistics	Statistics()、StatisticsW()
SQLTablePrivileges	TablePrivileges()、TablePrivilegesW()
SQLTables	Tables()、TablesW()