ISQL 移行ガイド

Version 2024.1
2024-06-03

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

ISQL 移行ガイド
InterSystems IRIS Data Platform　Version 2024.1　2024-06-03

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

1

ISQL の使用法

InterSystems ISQL は多くの機能をサポートする Informix ISQL の実装です。

また、InterSystems ISQL には、Informix ISQL には含まれない専用の拡張機能がいくつかあります。

このドキュメントでは、Informix データベースからスキーマおよびストアド・プロシージャを迅速に移行する方法、および InterSystems IRIS® Data Platform で ISQL を実装する方法を説明します。

1.1 はじめに

既存の ISQL アプリケーションを InterSystems ISQL に移行するには、次の 3 つの操作を実行する必要があります。

- ISQL 向けの InterSystems IRIS の構成
- ISQL ソース・コードの移行
- ISQL データの移行

1.2 ISQL 構成の設定

InterSystems IRIS には、InterSystems SQL への ISQL の移行を支援するために、4 つの構成設定が用意されています。これらの構成設定は、InterSystems IRIS の管理ポータルを使用するか直接グローバル参照を使用して指定できます。

管理ポータルを使用して ISQL 構成を設定するには、以下の手順を実行します。

- 管理ポータルのメイン・ページから、**[システム管理]**、**[構成]**、**[SQL とオブジェクトの設定]**、**[ISQL 互換性]** の順に選択します。

  ここでは、**[区切り識別子をサポート]** (既定は [いいえ])、**[トレースコード生成]** (既定は [いいえ])、**[ストアドプロシージャ呼び出しの結果を Resultset で返す]** (既定は [はい]) をオンまたはオフに切り替えることができます。識別子が InterSystems SQL 予約語と競合する場合に、その識別子に追加する **[予約語プリフィックス]** を指定することもできます。これらの設定の目的については、以下で説明します。

  管理ポータルを使用して構成オプションを 1 つでも設定した場合、**[Informix SQL 設定]** 見出しの後ろにアスタリスクが表示されます。これは、変更が行われたが、まだ保存されていないことを示します。構成の変更を有効にするには、**[保存]** ボタンをクリックする必要があります。

- 管理ポータルのメイン・ページから、[システム管理]、[構成]、[SQL とオブジェクトの設定]、[SQL] の順に選択します。ここから、[既定のスキーマ] を設定できます。これは、未修飾の DDL エンティティすべての (パッケージにマップする) システム全体の既定のスキーマ名です。
- 管理ポータルのメイン・ページから、[システム管理]、[構成]、[SQL とオブジェクトの設定]、[ユーザー DDL マッピング] の順に選択します。このオプションを使用して、必要なユーザ定義のデータ型をマッピングできます。

直接グローバル参照を使用して ISQL 構成を設定する場合や、現在の ISQL 構成設定を確認する場合、以下のグローバルを使用できます。

- ^%SYS("xsql","informix","DELIMIDENT") = 0 | 1 は、区切り識別子の設定を制御します。1 の場合、Informix ソースは、リテラル文字列識別子の代わりに識別子区切り文字として使用される二重引用符で解析されます。
- ^%SYS("xsql","informix","RESERVEDWORDPREFIX")=prefix。これを設定した場合、予約語となっている列名はすべて、列名に prefix を追加することによって解決されます。
- ^%SYS("xsql","informix","TRACE") = 0 | 1。1 の場合、TRACE コードが生成されます。TRACE が 1 の間に変換されたプロシージャにはすべて、DEFERRED 構文ごとに %XSQL.Log の呼び出しが含まれます。%Prepare のための DEFERRED 構文と、結果のための DEFERRED 構文です。TRACE = 1 で変換されたプロシージャはすべて実行時に %XSQL.Log ファイルにメッセージを記録します。このログは、プロセスごとに一意です。
- ^%SYS("xsql","informix","FUNCTIONRETURNSSET") = 0 | 1。1 の場合、複数の値を返す関数は、結果セットとしてそれらの値を返します。

1.2.1 識別子の競合

InterSystems IRIS には、ISQL 識別子 (テーブル名や列名など) と InterSystems IRIS SQL 予約語との間の競合を解決するための方法として 2 つの選択肢が用意されています。

予約語の接頭語が構成されている場合、InterSystems SQL 予約語と競合する識別子名には、指定されている接頭語が自動的に割り当てられます。この接頭語付けは、SQL 予約語リストに競合するあらゆる識別子に対して実行されます。このリストには、(ISQL 識別子の解決を目的として) 予約語を追加でき、それにはグローバル変数 ^%SYS("xsql","informix",word) を使用します。ここで、word には、予約する語を大文字で指定します。予約語リストに追加できるこの機能は、異なる予約語リストを使用するデータベース・システム間の整合性の維持に役立ちます。

予約語の接頭語は構成されておらず、区切り識別子のサポートが構成されている場合、コンバータは SQL 予約語リスト (および追加の予約語) に競合する識別子を引用符で囲むことによって区切ります。区切り識別子は、予約語を含め、あらゆる値を取ることができます。

1.2.2 TRACE

XSQL DEFERRED 構文は、実行時に動的に作成されます。結果の実装は、後続の実行のパフォーマンス向上のためにキャッシュされます。キャッシュされた実装は、いずれかの参照オブジェクトが変更されるたびに削除されます。Informix ISQL コンバータでは、DEFERRED 構文テキスト、引数値、および発生エラーを記録するコンパイル済みプロシージャ向けに TRACE 機能がサポートされます。

1.2.2.1 トレースのサポートの構成

Informix のトレース機能は、以下の 2 つの方法のいずれかを使用して有効にできます。

- InterSystems IRIS の管理ポータルに移動します。[システム管理]、[構成]、[SQL とオブジェクトの設定]、[ISQL互換性設定] の順に選択して、[トレースコード生成] 設定をオンにします。
- ObjectScript マクロ $$$TraceOn を呼び出します。

1.2.2.2 ソース・コード内の TRACE 文

トレース構成が有効になっている ISQL プロシージャを InterSystems IRIS が変換する際に、トレースの基本的なサポートが生成されます。プロシージャ・コード内の TRACE 文によって、トレースの動作が決まります。TRACE ON 文によって、トレース機能が有効になり、生成されたすべてのトレースがログに記録されます。このトレース機能は、既定で有効になっています。TRACE OFF 文によって、トレース機能が無効になり、これ以降のメッセージはログに記録されません。トレース機能が有効の場合、TRACE expression によって、expression の値が現在のログに記録されます。

1.2.2.3 TRACE ロギング

トレースからの出力は、IRIS.DAT と同じディレクトリにあるテキスト・ファイルに送信されます。これは、%New() メソッドを使用し、%xsqlLog にファイル名を指定して **%XSQL.Log** クラス・オブジェクトをインスタンス化することにより、オーバーライドできます。例 : SET %xsqlLog = ##class(%XSQL.Log).%New("c:¥mylogs¥mytrace.log",1)。%New() に対する 2 番目の引数は、initialize フラグです。このフラグが TRUE (1) で、ログ・ファイルが既に存在する場合、そのログ・ファイルがクリアされます。initialize が FALSE (0) の場合、トレース情報が既存のログ・ファイルに追加され、"log restarted..." のメッセージがログに書き込まれます。既定で、ログ・ファイル名は現在のジョブ番号で修飾されます。

TRACE が有効な場合、結果セットが現在のコンテキスト・オブジェクトに追加されるたびに、メッセージがログに記録されます。結果セットは、コンテキスト・オブジェクトから取得され、最初の 10 行がログにコピーされます。

TRACE が有効な場合、変換された Informix プロシージャ内の埋め込み SQL および遅延 SQL コードの実行すべてに対してロギングが実行されます。SQL 文のトレースには、文テキスト、引数とその値、SQLCODE 値、%ROWCOUNT、および実行後の %msg (SQLCODE < 0 の場合)、さらにその SQL 文の実行の経過時間が含まれます。CURSOR ベースの SELECT 文の場合、トレースは、カーソルのクエリ、OPEN、および CLOSE について記録されます。

トレース・ログにメッセージを書き込むには、ObjectScript マクロ $$$TraceMessage(text) を呼び出します。ここで、text は引用符で囲まれた任意の文字列です。

1.2.3 従来の関数呼び出しモード

[ストアドプロシージャ呼び出しの結果を Resultset で返す] 構成設定は、複数の値を返す関数がそれらの値をどのように返すのかを制御します。既定の “[はい]” では、値が結果セットとして返されます。この設定では、ISQL の従来の関数呼び出しモードがサポートされるため、いずれの返り値も、1 つの返り値と出力パラメータのいずれかまたは両方ではなく、単一行の結果セットとして返されます。

1.3 ソース・コードの移行

最初のアプリケーション移行は簡単です。

1. DDL/DML スクリプト・ファイルの移行 : 以下のメソッドによって、指定されたディレクトリ内のすべての DDL/DML スクリプト・ファイルがインポートされます。指定されたディレクトリ内の拡張子が .sql のファイルがすべてインポートされます。

   ```
   $SYSTEM.SQL.Schema.ImportDDLDir(DDLMode, directory, logfile, EOSDelimiter)
   ```

   DDLMode : スクリプト・ファイルのインポート元のベンダ。このパラメータは必須です。唯一許可される値が `Informix` です。

   directory : インポート先のディレクトリのフル・パス名。このパラメータは必須です。

   logfile : エラーの報告先のログ・ファイルのフル・パス名または数字の 1。このパラメータはオプションです。既定値は、読み込まれるディレクトリ内の DDLImportDir.log です。このパラメータ値が 1 の場合、読み込まれるファイルごとに個別のログ・ファイルが生成されます。各ログ・ファイルの名前は、インポートされるファイルの名前と同じですが、拡張子が .sql ではなく .log になります。

EOSDelimiter ： 文末区切り文字。このパラメータはオプションです。%DDLMode の値に基づいて適切な値が既定値となります。

詳細は、"InterSystemsクラスリファレンス" を参照してください。

2. また、$SYSTEM.SQL.Schema.LoadInformix() メソッドを呼び出してスキーマをインポートすることもできます。このメソッドでは、CREATE TABLE、ALTER TABLE、CREATE INDEX、CREATE VIEW、SET OPTION、および GRANT の各文がサポートされます。詳細は、"InterSystemsクラスリファレンス" を参照してください。

   ISQL ソースに CREATE PROC 文が含まれる場合は、CREATE PROC ソースを含むクラス・メソッドが作成されます。InterSystems IRIS では、このクラス・メソッドが既存のクラスに配置されるか、スキーマおよびプロシージャ名に基づく新しいクラスに配置されます。プロシージャが既に存在する場合は、既存のバージョンが新しいバージョンに置き換えられます。スキーマおよびプロシージャから生成されたクラス名に一致するクラスが既に存在する場合は、既存のクラス名が使用されます (以前に ISQL ユーティリティによって生成されている場合)。一致するクラスが存在しない場合は、スキーマおよびプロシージャ名に基づく一意のクラス名が生成されます。プロシージャが正常に作成されると、結果のクラスがコンパイルされます。

   ロギングが要求される場合は、ソース文が、それを含むクラス名、クラス・メソッド、および生成された仮引数と共に記録されます。プロセスで発生したエラーもログに記録されます。CREATE PROC の処理中にエラーが検出された場合、InterSystems IRIS では、そのプロシージャに対して生成された新しいクラスが削除されます。

3. エラーのログ・ファイルの検査 : エラー番号で検索します。エラーと正しく実行されたインポートの合計数がログの最後に表示されます。ほとんどの場合、エラーは、このドキュメントの情報を使用して回避または対処できます。

   InterSystems IRIS には、Informix エラー・コードおよびメッセージ、および対応する InterSystems SQLCODE エラー・コードおよびメッセージを相互参照するための %XSQL.System.CacheMessageXRef クラスが用意されています。

4. コンパイル ： DDL をインポートすると、テーブルおよびビュー定義のコンパイルが自動的に実行されます。その他の ISQL ソース・コードをコンパイルするには、以下のようにコマンドを使用します。

   ObjectScript

   ```
   DO $SYSTEM.OBJ.CompileAll("-l")
   ```

   小文字の “L” 修飾子フラグによって、コンパイルの際にロックを適用しないように設定します。フラグ修飾子の完全なリストを確認するには、`DO $SYSTEM.OBJ.ShowFlags()` を呼び出します。

ISQL メソッドをコンパイルすると ObjectScript コードが生成されます。ネイティブの ISQL はシステム・レベルでサポートされません。見慣れた元のストアド・プロシージャの形式を生かす場合は、ISQL のメソッドを維持することが最善です。

1.4 Informix ストアド・プロシージャの移行

InterSystems IRIS には、Informix SPL ルーチンをクラス・メソッドに変換するコンバータが用意されています。この変換で得られるクラス・メソッドには、一時テーブル、DEFERRED 構文の解析 (構文は実行時に動的に作成され、クラスのコンパイル時にメタデータの検証は行われません)、および例外処理など、Informix SPL 動作と同じ動作が実現される ObjectScript コードがあります。InterSystems IRIS コンバータは、SPL で宣言された仮引数で仮仕様が構成されているクラス・メソッドを生成します。複数の値を返す関数には、宣言された返りタイプに対応する仮引数が追加されます。反復子関数は、宣言された返りタイプに列が対応している結果セットを返します。

元の Informix SPL ソースは、生成されたクラスのクラス説明で保持されます。コンバータは、ソース内のコメントを、変換された出力内に維持します。

必要に応じ、Informix SPL コンバータは、すべての変換されたルーチン、発生したエラー、および生成された各クラス・メソッドの名前をログに記録できます。このログは、%XSQL.Log で保持されます。変換中に発生したすべてのエラーの要約もログの末尾に記録されます。

ObjectScript バージョンの SPL ルーチンでは、%XSQL DEFERRED 構文、プライベート一時テーブル、およびエラー・メッセージ相互参照が使用されます。

1.4.1 一時テーブル

Informix 一時テーブル (CREATE TEMP TABLE) は、InterSystems IRIS グローバル・プライベート一時テーブルとして実装されます。これらの一時テーブルは、現在のプロセスからのみアクセスでき、明示的に削除されるかプロセスが終了するまで定義されます。これらの一時テーブルは、InterSystems SQL コマンド CREATE GLOBAL PRIVATE TEMPORARY TABLE を使用して作成されます。

GLOBAL PRIVATE TEMPORARY テーブルは、テーブル・オブジェクトが %sqlcontext ではなく %processcontext で保持される点を除いて、GLOBAL TEMPORARY テーブルと同じです。%processcontext は、プロセスの存続中は新しく開始されることも強制終了されることもない ProcedureContext インスタンスです。このため、%processcontext によって参照される一時テーブル・オブジェクトは、削除されるかプロセスが終了するまでスコープ内にあります。一時テーブルを削除しても、そのテーブルが物理的には削除されない場合があります。アクティブな結果セットがその一時テーブルを参照する場合、その一時テーブルは、それらの結果セットのオブジェクトがクローズ (破棄) されるまでアクティブなままです。アクティブな結果セットに起因してアクティブなままである一時テーブルを作成しようとすると、エラーが生じます。

1.4.2 プロシージャの呼び出しと反復子関数

Informix では以下の 4 つのタイプの SPL ルーチンがサポートされます。

- 何の値も返さないプロシージャ (SPL)。DO コマンドを使用して InterSystems IRIS から呼び出されます。
- 単一値を返す標準関数 (SPL RETURN)。SET retval=function() を使用して InterSystems IRIS から呼び出されます。
- 複数の値を返す関数 (SPL RETURN value1[,value2[,...]])。Informix SPL ルーチンでは、出力形式引数はサポートされていません。InterSystems IRIS では、これは複数の出力仮引数を取るクラス・メソッドに変換されます。この関数は、DO コマンドを使用して呼び出されます。出力値はプロシージャ・コンテキスト・オブジェクトから取得され、宣言された INTO 変数に割り当てられます。
- それぞれに 1 つ以上の値が含まれる行を複数返す反復子関数 (SPL RETURN...WITH RESUME)。InterSystems IRIS では、複数の出力仮引数があり返り値のないクラス・メソッドとしてこの機能は実装されます。この関数は、DO コマンドを使用して呼び出されます。この関数は、結果セット・クラスを使用して、まず結果セットに連続する行を挿入してから、反復子を結果セットの最初の行にリセットして複数行を取得できるようにします。行は、プロシージャ・コンテキスト・オブジェクトで返される最初の結果セットから取得されます。結果セット内の列は、宣言された返りタイプに対応します。

返り変数に定義されているサイズより返り値が長い場合、InterSystems IRIS は (Informix と同様に) 返り値を自動的に切り捨てます。

スカラ関数は結果セットを返すことができません。InterSystems IRIS では、単一の返り値が必要だが実際には複数行を返す関数が呼び出されると、SQLCODE -432 を発行します。これは、Informix エラー -686 に相当します。

反復子関数では、フローに 1 つの大きな違いがあります。Informix の文 FOREACH EXECUTE は、結果セットを返すプロシージャを呼び出します。この結果セットは、呼び出されたプロシージャ・コンテキストのスコープ内でのみ有効な一時オブジェクトを参照できます。元の Informix では、FOREACH EXECUTE は、その前の RETURN WITH RESUME 文の時点で実行を再開します。変換された ObjectScript メソッドは、反復子関数を実行して完了させ、FOREACH ループによって反復されることになる結果セットで結果を返します。実際の列値は、呼び出されたプロシージャ・コンテキストなしで、結果セットから直接取得されます。EXIT FOREACH は、カーソルをクローズして、最後に取得された行のロックを解除します。

1.4.3 仮引数

InterSystems IRIS では、仮引数に対して、DEFAULTFORMAL および DEFAULTFORMALTYPE の 2 つの設定がサポートされています。DEFAULTFORMAL は、ダミー仮引数の名前を指定するオプション設定です。DEFAULTFORMALTYPE は、DEFAULTFORMAL 引数のタイプを指定するオプション設定です。

DEFAULTFORMAL を設定すると、その名前を持つ仮引数が、いずれの仮引数も宣言しないプロシージャまたは関数に対して生成されます。DEFAULTFORMALTYPE を定義した場合、これは生成された仮引数に使用される Informix タイプの宣言になります。DEFAULTFORMALTYPE を定義しない場合、既定の INTEGER に設定されます。

1.5 データの移行

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。[**ウィザード**] ドロップダウン・リストから [**データ移行**] を選択します。

1.5.1 Informix ROWID

プロシージャ変換ユーティリティは、ROWID の ISQL 列参照を %ID に変換します。Informix では、ROWID 列は、WITH ROWIDS 節を使用して作成された断片化されているテーブルや断片化されていないテーブルの非表示列です。ROWID 列は行ごとに固有ですが、必ずしも連続する必要はありません。InterSystems SQL では、ROWID 列は %ID に相当します。

1.5.2 NULL および空文字列の処理

Informix では、文字列データ型 CHAR および VARCHAR では、空文字列 (″″) は単一の空白 (″ ″) とまったく同じように処理されます。数値、時刻、および日付のデータ型すべてにおいて、空文字列 (″″) は NULL として処理されます。

文字列を連結する場合、NULL を string に連結すると、NULL になります。空文字列を string に連結しても、string には何も影響がありません。

1.5.3 末尾の空白文字列

ObjectScript で末尾の空白文字列は自動的に切り捨てられます。これは、CHAR と VARCHAR の両方のデータ型に当てはまります。これは、Informix の標準の動作とは異なるため、データの移行時に重要な考慮事項となる可能性があります。

1.5.4 DATETIME データ型

InterSystems IRIS では、Informix DATETIME データ型を %Library.InformixTimestamp データ型クラスにマップすることにより、このデータ型をサポートしています。Informix には、完全な日付や時刻は必要ありません。Informix では、修飾子を使用して、日付と時刻のコンポーネントの範囲が指定されます。InterSystems IRIS では、最大および最小の修飾子を指定するための構文がサポートされます。YEAR TO DAY 修飾子は、DATE データ型としてマップされます。MINUTE TO SECOND 修飾子は、TIME データ型としてマップされます。その他すべての修飾子は %Library.InformixTimeStamp としてマップされます。例えば、YEAR TO SECOND および HOUR TO SECOND などがあります。

InterSystems IRIS では、DATETIME 算術がサポートされます。InterSystems IRIS では、DATE、TIME、TIMESTAMP、および INTERVAL に対して加算と減算の演算がサポートされます。

1.5.5 SERIAL データ型

InterSystems IRIS では、Informix SERIAL データ型を %Library.Counter データ型にマップすることにより、これをサポートしています。特定のフィールドがデータ型 %Counter として定義されている場合、InterSystems IRIS では SQL INSERT の実行時に、そのフィールドに値が挿入されない、0 (ゼロ) が挿入される、または数値以外の値が挿入されるときに、カウンタ値が 1 ずつ増加され、結果のカウンタ値がこのフィールドに割り当てられます。正の整数値をこのフィールドに明示的に挿入する場合は、その値を挿入します。また、必要に応じて、カウンタ・ノードの値が指定フィールド値を下回らないように、カウンタ・ノードの値を進めます。

%Counter 型で定義されているフィールドに UNIQUE 制約が自動的に作成されることはありません。値が必ず UNIQUE になるようにする場合は、そのプロパティに独自の UNIQUE インデックスを作成する必要があります。

%Counter フィールドとして定義されているフィールドは、そのフィールドの古い値が 0 または NULL でない限り更新されない場合があります。データが含まれる既存テーブルに %Counter フィールドを追加してから戻り、既存の行で %Counter フィールドにデータを入力する場合、これが当てはまる場合があります。このとき、%Counter フィールドを更新しようとすると、SQLCODE -105 エラーと %msg テキストが生成されます。

InterSystems SQL %Counter データ型には、Informix SERIAL データ型と同様の機能が用意されています。Informix では、テーブルごとに 1 つの SERIAL フィールドという制限がありますが、InterSystems IRIS には、テーブルごとに定義する %Counter フィールドの数に制限はありません。

1.6 ISQL 言語の実装

1.6.1 ISQL のコメントと注釈

ISQL では、以下のコメント・タイプがサポートされています。

- `-- comment` (2 つのハイフン) 単一行コメントの接頭語。
- `{ comment }` 複数行のコメント。
- `/* comment */` 複数行のコメント。

複数行のコメントを入れ子にすることができます。

ISQL では、以下の単一行の注釈形式がサポートされています。

- `--!! annotation`
- `--## annotation`

SELECT 文に埋め込まれている注釈は、InterSystems SQL への変換後に FROM 節に追加されます。以下はその例です。

```
FOREACH
    SELECT a,b,c
    INTO v_a, v_b, v_c
    FROM
       --!!%NOFLATTEN %NOTOPOPT
       my_table as f
    WHERE f.a IN ( "a") AND f.b = 88 AND f.c ="ZZ"
    ORDER BY a, c DESC, b DESC
```

注釈は、SELECT 文内でのみ使用できます。

1.6.2 MATCHES 演算子

ISQL では、パターン・マッチング用に MATCHES 演算子がサポートされています。この演算子では、0 以上の文字を表すために使用できる * ワイルドカード文字がサポートされています。

1.6.3 DEFINE var LIKE コマンド

DEFINE var LIKE column 文では、特定の列のデータ型に基づいて、特定のローカル変数のデータ型が定義されます。このコマンドは、変換時に列参照を解決できる場合に使用できます。

1.6.4 LET コマンド

LET コマンドは、変数に値を割り当てるために使用します。以下のように、複数の変数への割り当てに使用できます。

```
LET a,b = c,d;
```

1.6.5 サポートしている関数

Informix コンパイラでは、以下の Informix 関数がサポートされます。

- DATE
- CURRENT は内部ストレージ形式 ($HOROLOG 値の日付部分) で現在の日付を返します。
- LOWER
- REPLACE
- SUBSTR
- SUBSTRING
- TRIM (部分的にサポート)
- UPPER