

# Using the Caché SQL Gateway

Version 5.1  
15 June 2006

*Using the Caché SQL Gateway*

Caché Version 5.1 15 June 2006

Copyright © 2006 InterSystems Corporation.

All rights reserved.

This book was assembled and formatted in Adobe Page Description Format (PDF) using tools and information from the following sources: Sun Microsystems, RenderX, Inc., Adobe Systems, and the World Wide Web Consortium at [www.w3c.org](http://www.w3c.org). The primary document development tools were special-purpose XML-processing applications built by InterSystems using Caché and Java.



The Caché product and its logos are trademarks of InterSystems Corporation.



The Ensemble product and its logos are trademarks of InterSystems Corporation.



The InterSystems name and logo are trademarks of InterSystems Corporation.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Memorial Drive, Cambridge, MA 02142, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

Caché, InterSystems Caché, Caché SQL, Caché ObjectScript, Caché Object, Ensemble, InterSystems Ensemble, Ensemble Object, and Ensemble Production are trademarks of InterSystems Corporation. All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

For Support questions about any InterSystems products, contact:

**InterSystems Worldwide Customer Support**

Tel: +1 617 621-0700

Fax: +1 617 374-9391

Email: [support@InterSystems.com](mailto:support@InterSystems.com)

# Table of Contents

<b>1 Introduction</b> .....	<b>1</b>
<b>2 Architecture</b> .....	<b>3</b>
2.1 The Connection Manager .....	3
2.2 The SQL Gateway API .....	3
2.3 The External Table Query Processor .....	4
2.4 SQL Storage Class .....	4
<b>3 Configuration</b> .....	<b>7</b>
3.1 Creating an SQL Gateway Connection .....	7
<b>4 Using the SQL Gateway</b> .....	<b>9</b>
4.1 Using the Link Table Wizard .....	9
4.2 Debugging Any Gateway Problems .....	10
<b>5 Restrictions</b> .....	<b>13</b>

# List of Figures

Using the SQL Gateway to View External Data .....	1
---	---

# 1

## Introduction

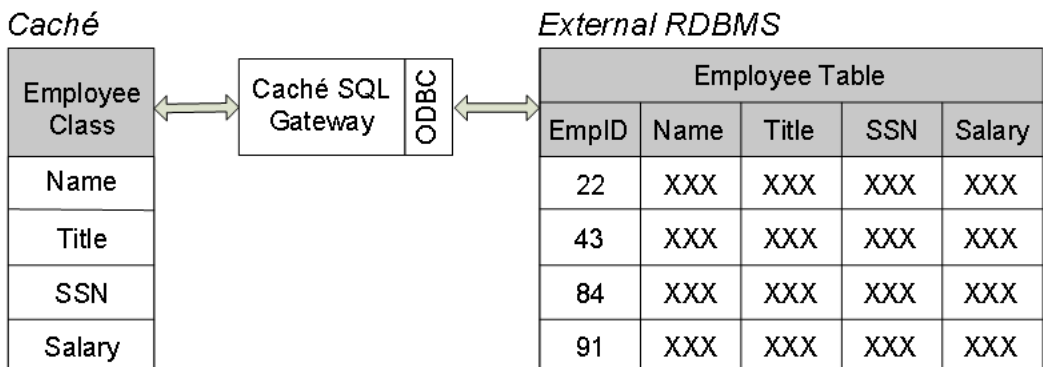
The Caché SQL Gateway, in conjunction with Caché Objects, provides object access to data stored in a relational database management system (RDBMS) outside of (or “external” to) Caché.

Using the SQL Gateway, applications can:

- Access data stored in third-party relational databases within Caché applications using objects and/or SQL queries.
- Store persistent Caché objects in external relational databases.

For example, suppose you have an “Employee” table stored within an external relational database. You can use this table within Caché as an object by creating (or letting the Caché Link Table Wizard create for you) an Employee class that communicates (by executing SQL queries via ODBC) with the external database. This is illustrated below:

### *Using the SQL Gateway to View External Data*



From the perspective of a Caché application, the Employee class behaves in much the same way as any other persistent class: You can open instances, modify, and save them. If you issue SQL queries against the Employee class, they are automatically dispatched to the external database.

The use of the SQL Gateway is independent of application logic; an application can be modified to switch between external databases and the built-in Caché database with minimal effort and no change to application logic.

Classes using the SQL Gateway to provide object persistence are identical in usage to those using native persistence and can make full use of Caché features including Java, ActiveX, SQL, and Web access.

# 2

## Architecture

The SQL Gateway consists of the following components.

- The Connection Manager
- The SQL Gateway API
- The External Table Query Processor
- SQL Storage Class

These are described in the following sections.

### 2.1 The Connection Manager

The Connection Manager is a list of logical connection definitions maintained by Caché. Each connection definition associates an external database with a logical name used by Caché. The SQL Gateway uses these logical names to determine how to establish a connection with a specific external database.

For more details, see [SQL Gateway Connections](#).

### 2.2 The SQL Gateway API

The SQL Gateway API is a set of functions used by a Caché program to communicate with a third-party RDBMS. These functions are implemented by means of a shared library, also

known as a Dynamically Linked Library (DLL). This shared library is responsible for establishing an ODBC connection with an external database, making the appropriate calls to the external ODBC interface, and performing any data conversions that may be required. When connecting to an external database using the SQL Gateway API, a Caché program has access to that portion of the ODBC API that allows it to execute those queries supported by the gateway.

## 2.3 The External Table Query Processor

The External Table Query Processor is an extension to the Caché SQL Query Processor that handles queries targeted at external tables.

Caché maintains a list of all defined SQL tables within what is called the SQL Dictionary. A given table is marked as “external” when its data is stored in a third-party RDBMS. When the Caché SQL Query Processor detects that the table (or tables) referenced within an SQL query are external, it invokes the External Table Query Processor which generates a query execution plan by means of SQL Gateway API calls instead of accessing data stored within Caché.

It is the integration within the internal workings of the Caché SQL Engine that provides the SQL Gateway with its transparency; since any object or SQL-based request automatically accesses the correct storage location, applications can be written independently of the actual data storage used.

## 2.4 SQL Storage Class

All object persistence in Caché is provided by means of a “storage class.” A storage class generates the code needed to save and retrieve a persistent object within a database. The SQL Storage class (%Caché.SQLStorage) is a special purpose storage class that provides object persistence by means of specially-generated SQL queries. Typically, this SQL provides storage within the Caché database in cases where an application needs finer control over storage structures than is provided by the default, Caché Storage storage class.

A class using the SQL Storage Class for persistence indicates that it is an “external” class by providing a value for its *CONNECTION* and *EXTERNALTABLENAME* class parameters. When such a class is compiled, the Class Compiler does the following:

- It creates an SQL table definition for the class. This table is marked as “external” and is associated with the SQL Gateway connection specified by the class' *CONNECTION* class parameter. The name of the table in the external database is specified by the *EXTERNALTABLENAME* class parameter.
- It generates object persistence code for the class using SQL queries. These queries automatically make calls to the correct external database by means of the External Table Query Processor.



# 3

## Configuration

The SQL Gateway is automatically installed as part of Caché. The SQL Gateway connects to a specific external database via a defined connection. To define such a connection, the tasks are:

1. Define an ODBC data source for the external database (refer to the external database's documentation for more information on how to do this).
2. Define a logical connection to this data source.

### 3.1 Creating an SQL Gateway Connection

Caché maintains a list of SQL Gateway connection definitions. Each connection definition consists of a logical name (used within the application to refer to this connection); an ODBC data source name (DSN); and a user name and password to use when establishing the connection.

Every external table within Caché is associated with a particular logical connection name. Caché automatically establishes a connection with an external database the first time a process refers to its logical connection.

To define a new SQL Gateway connection (or edit an existing one), perform the following steps:

1. In the System Management Portal, go to the **SQL Gateway Connections** page ([Home] > [Configuration] > [SQL Gateway Connections]). (Specifically, from the Portal Home page,

select **Configuration** from the **System Administration** column; from the **Configuration** page that appears, select **SQL Gateway Connections** from the **Connectivity** column.)

2. On the **SQL Gateway Connections** page, click **Create New Connection**. This displays the **SQL Gateway Connection** page ([Home] > [Configuration] > [SQL Gateway Connections] > [SQL Gateway Connection]).
3. On the **SQL Gateway Connection** page, enter or choose values for the following fields:
  - **Connection Name** — An identifier for the connection being configured.
  - **Select an existing DSN** — The Data Source Name to accept connections
  - **User** — The name for the account to serve as the default for establishing connections
  - **Password** — The password associated with the default account
  - **Enable legacy outer join syntax (Sybase)** — Whether or not the connection supports this syntax
4. You can test that the values allow for establishing a connection by clicking the **Test Connection** button.
5. To create the named connection, click **Save**.

# 4

## Using the SQL Gateway

Using the SQL Gateway within an application is straightforward. The simplest way to use it is to start from a table already defined in a third-party RDBMS; in this case, simply define a connection to this database and use the Caché Link Table Wizard — available on the **SQL** page of the System Management Portal (**[Home]** > **[SQL]**) — to automatically generate a Caché persistent class that uses the external table for its data storage.

### 4.1 Using the Link Table Wizard

The Link Table Wizard reads a table definition from an external database and generates a new persistent Caché class. The new class stores and retrieves its data from the external table using the SQL Gateway.

After you have defined an external database connection (see [Configuration](#), you can use the Link Table Wizard as follows:

1. From the **SQL** page of the System Management Portal (**[Home]** > **[SQL]**), click **Link Table Wizard**. This displays the Link Table Wizard.
2. The first page of the wizard allows you to select the table to be made available through the Gateway connection. The fields on this page allow you to specify the available table:
  - Select a SQL Gateway connection — The SQL Gateway connection that is to serve as the source for the data. To create an SQL Gateway connection, see the section “[Creating an SQL Gateway Connection](#).”
  - Table Type — Which kinds of tables are to be made available.

- Schema — The schema (package) from which tables are to be made available.
  - Tables — The table itself to be made available.
3. The second page of the wizard lists available and selected fields (properties). Highlight one or more fields and click the single arrow to move it or them from one list to another; click the double arrow to move all fields (selected or not) from one list to another. In the selected list, the up and down arrows allow you to modify the order of the list.
  4. The third page of the wizard allows you to specify the behavior and appearance of the displayed information. The available options are:
    - Read only — Makes the displayed field non-writeable. (The `select_all` checkbox makes all the fields non-writeable.)
    - New Caption
    - New Column Name
  5. On the last page of the wizard, check that the primary key field(s), as well as the SQL table and Class name to be used for the new class is correct.
  6. Clicking Finish on the last page of the wizard displays a page that allows you to establish a test connection to the data source for the Gateway.

At this point, the Link Table Wizard stores a new class definition within the Caché database and compiles it. If data is present, it should be immediately visible in the external database (you can check by issuing SQL queries against the newly created Caché class/table). You can now use the new class as you would any other persistent class within Caché.

## 4.2 Debugging Any Gateway Problems

If you encounter any problems, you can monitor the Gateway as follows:

1. Set the value of the `^%SYSLOG` global to 3 by typing the following at the Caché terminal prompt:

```
Set ^%SYSLOG = 3
```

The `^%SYSLOG` global logs events in Caché for use in debugging.

2. Turn on logging for the DSN in use, if the DSN supports this. A DSN with logging functionality will have a check box or other switch on its screen. For instance, the Caché DSN has an ODBC Log check box.

To display the configuration screen for a DSN, select the DSN from the one of the lists (user, system, or file) on the ODBC Data Source Administrator screen, which you can display by selecting Data Sources (ODBC) from the Administrative Tools screen, which is available from the Windows Control Panel.

3. Turn on logging for the ODBC driver manager by selecting the Start Tracing Now button on the Tracing tab of the ODBC Data Source Administrator screen. Again, to display this screen, select Data Sources (ODBC) from the Administrative Tools screen, which is available from the Windows Control Panel.
4. Run your application, ensuring that you trigger the error condition.
5. Check all the logs for error messages or any other unusual activity. The cause of the error is often obvious.



# 5

## Restrictions

The current version of the Caché SQL Gateway has the following limitations and restrictions:

- All the tables listed in the **FROM** clause of an SQL query must come from the same data source. Queries joining data from heterogeneous data sources are not allowed.
- SQL queries targeted at external databases cannot use Caché SQL extensions. This includes the “->” operator, including other columns within a count (\*) query, and the various Caché-specific operators whose name starts with %.
- The external RDBMS must have an ODBC interface. This interface must correctly support the functionality required by the SQL Gateway.

