RTI Real-Time Connect

Getting Started Guide

Version 5.0

Your systems. Working as one.

Technical Support

Real-Time Innovations, Inc.
232 E. Java Drive
Sunnyvale, CA 94089

Phone:	(408) 990-7444
Email:	support@rti.com
Website:	https://support.rti.com/

Available Documentation

The following documentation is available for *RTI® Real-Time Connect*:

- ❑ The *Release Notes*, **RTI_RTC_ReleaseNotes.pdf**. This document provides an overview of the current release's features and lists changes since the previous release, system requirements, supported architectures, and compatibility with other products.
- ❑ The *Getting Started Guide*, R**TI_RTC_GettingStarted.pdf**. This document provides installation instructions, a short 'Hello World' tutorial, and troubleshooting tips.
- ❑ The *User's Manual*, **RTI_RTC_UsersManual.pdf**. This document starts with an overview of *Real-Time Connect*'s basic concepts, terminology, and unique features. It then describes how to develop and implement applications that use *Real-Time Connect*.

Additional Resources

- ❑ The *ODBC API Reference* from Microsoft is available from http://msdn.microsoft.com/en-us/library/ms714562(VS.85).aspx.
- ❑ The documentation for the Oracle TimesTen In-Memory Database can be found in the **doc/** directory of the Oracle TimesTen installation.
- ❑ The documentation for Oracle databases can be found http://www.oracle.com/technology/documentation/index.html.
- ❑ The documentation for MySQL is available from http://dev.mysql.com/doc/#manual.

Contents

Chapter 1 Welcome to RTI Real-Time Connect

Welcome to *RTI® Real-Time Connect*—a high-performance solution for integrating applications and data across real-time and enterprise systems from RTI.

Real-Time Connect is the integration of two complementary technologies: data-centric publish-subscribe middleware and relational database management systems (RDBMS). This powerful integration allows your applications to uniformly access data from real-time/embedded and enterprise data sources via *RTI Connext*™ (formerly *RTI Data Distribution Service*), or via database interfaces. Since both these technologies are data-centric and complementary, they can be combined to enable a new class of applications. In particular, *Connext* can be used to produce a truly decentralized, distributed RDBMS, while RDBMS technology can be used to provide persistence for real-time data.

1.1 Intended Readers

This document is intended for system administrators and others who are responsible for performing installation and configuration tasks.

This document guides you through the process of installing *Real-Time Connect* and running three different scenarios:

- ❏ Storing Samples From Publishers
- ❏ Publishing Database Updates to Subscribers

- ❑ Database Table Replication. We only cover this scenario for *Real-Time Connect* to Oracle TimesTen. Database replication is also possible with MySQL and Oracle 11g.

The following platform configurations are covered. After reading this introduction, you can skip to the chapter that discusses the configuration that you will be using.

Note: *Real-Time Connect* to Oracle and *Real-Time Connect* to MySQL require that *Connext* is already installed.

To use Oracle TimesTen, read either:

- ❑ Chapter 2: Real-Time Connect to Oracle TimesTen on UNIX-based Systems
- ❑ Chapter 3: Real-Time Connect to Oracle TimesTen on Windows Systems
- ❑ Additionally, to use Oracle In-Memory Database Cache, read Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache

To use Oracle 10g, read either:

- ❑ Chapter 4: Real-Time Connect to Oracle 11g on UNIX-based Systems
- ❑ Chapter 5: Real-Time Connect to Oracle 11g on Windows Systems

To use MySQL, read either:

- ❑ Chapter 7: Real-Time Connect to MySQL on UNIX-based Systems
- ❑ Chapter 8: Real-Time Connect to MySQL on Windows Systems

Chapter 2 Real-Time Connect to Oracle TimesTen on UNIX-based Systems

This chapter provides instructions on how to install *Real-Time Connect* to Oracle TimesTen on the following platforms:

❑ Red Hat® Enterprise Linux® 3.0, 4.0 and 5.0 on Pentium®-class processors

2.1 Installation

Install *Real-Time Connect* to Oracle TimesTen from the distribution file:

1. Unpack the distribution file to a temporary location using GNU™'s version of tar, **gtar**. There are different distribution files depending on if you are running on a 32-bit or 64-bit version of the operating system.

   For example (your filename will be slightly different, depending on the version number/letter):

   - On Linux systems, use **tar**:

     ```
     > tar zxfv RTI_Real-Time_Connect_5.0.x.linux32bundle.tar.gz
     ```

   This will extract the compressed *Real-Time Connect* to Oracle TimesTen file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 11.2.1 distribution file.

2. If Oracle TimesTen 11.2.1 is not installed on your system, install it by unpacking the distribution and following the instructions in the *Oracle TimesTen In-Memory Database Installation Guide*, which can be found in **<*install-folder*>/doc/doc.zip**, where <*install-folder*> is **linux86** or **linux8664**.

   - On Linux systems, use **tar**:

     ```
     > tar zxfv timesten112140.linux86.tar.gz
     ```

     or

     ```
     > tar zxfv timesten112140.linux8664.tar.gz
     ```

   When installing Oracle TimesTen, you can accept all default options with the following exceptions:

   If you want to allow direct database access to users of any OS group (world accessibility), make sure you answer 'no' to the following two questions:

   - Restrict access to the TimesTen installation to the group 'xxxx'?
   - Do you want to restrict access to the TimesTen installation to a different group?

   If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB) (see Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache), be sure that during the installation process you configure the environment variable TNS_ADMIN with the path to the **tsnnames.ora** file.

   Real-Time Connect to Oracle TimesTen requires the installation of the Oracle TimesTen Data Manager.

   The rest of this chapter assumes that Oracle TimesTen is installed under **/opt/TimesTen/tt1121**.

3. Add the path to the Oracle TimesTen libraries to the beginning of **LD_LIBRARY_PATH**. For example:

   ```
   > setenv LD_LIBRARY_PATH /opt/TimesTen/tt1121/lib:$LD_LIBRARY_PATH
   ```

   or

   ```
   > export LD_LIBRARY_PATH=/opt/TimesTen/tt1121/lib:$LD_LIBRARY_PATH
   ```

4. Add the path to the Oracle TimesTen utilities to the beginning of **PATH**. For example:

   ```
   > setenv PATH /opt/TimesTen/tt1121/bin:$PATH
   ```

   or

   ```
   > export PATH=/opt/TimesTen/tt1121/bin:$PATH
   ```

5. Unpack the compressed *Real-Time Connect* to Oracle TimesTen installation file in a location of your choice:

   For example (your filename will be slightly different, depending on the version number/letter):

   - On Linux systems, use **tar**:

   ```
   > tar zpxfv RTI_Real-Time_Connect_5.0.x.linux32.tar.gz
   ```

   This process will result in four directories, **/bin**, **/docs**, **/lib** and **/resource**, which contain the *Real-Time Connect* to Oracle TimesTen executables, user documentation, libraries and resources, respectively.

6. Please read Chapter 9: License Installation.

2.2 Creating an Oracle TimesTen Database

Before you can use *Real-Time Connect* to Oracle TimesTen, you need to get a TimesTen user account to an existing data store (database) from your database administrator. This section explains how to create a TimesTen database when you act as your own database and system administrator.

❑ Log in as the Oracle TimesTen administrator (the user that installed TimesTen):

```
> su - root
```

❑ Create an ODBC INI file called **.odbc.ini** under the administrator's home directory. This file will contain the DSN (Data Source Name) that will provide access to the data store.

❑ Insert the following lines in the file:

```
[ODBC Data Source]
Example=TimesTen Driver

[Example]
DRIVER=/opt/TimesTen/tt1121/lib/libtten.so
DataStore=/tmp/Example
DatabaseCharacterSet=AL32UTF8
```

Note: Make sure that **DRIVER** points to the valid location of the Oracle TimesTen ODBC driver on your system. You may want to choose a different **DataStore** path than the one shown above.

Note: If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB) (see Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache), add the attribute **OracleNetServiceName** and be sure that the attribute **DatabaseCharacterSet** is set to the same character set that Oracle 11g uses. For example:

```
[Example]
DRIVER=/opt/TimesTen/tt1121/lib/libtten.so
DataStore=/tmp/Example
DatabaseCharacterSet=WE8ISO8859P1
OracleNetServiceName=orcl11g
```

You can determine the Oracle database character set by executing the following query on SQL*Plus as any user:

```
SQL> SELECT value FROM nls_database_parameters WHERE
   parameter='NLS_CHARACTERSET';
```

❑ Create the new data store by executing the following command:

```
> ttIsql -connStr "DSN=Example"
command> exit
```

2.3 Creating an Oracle TimesTen User Account

Starting with Oracle Timesten 11.2.1, database access control is mandatory.

If you are acting as your own database and system administrator, use the following steps to create an account for the user “Student”:

1. If Oracle TimesTen was not installed with world accessibility, the operating system user (for example, “oracle”) who is going to log in to the database as “Student” must be added to the Oracle Timesten user’s group. For example:

   ```
   > su - root
   > useradd -G <TimesTen users group> oracle
   ```

   For additional information on installation requirements for operating system group and file permissions, see the *Oracle TimesTen In-Memory Database Installation Guide*.

2. Log in as the Oracle TimesTen administrator (the user that installed TimesTen):

   ```
   > su - root
   ```

3. Connect to the Oracle TimesTen database where the user account must be created:

   ```
   > ttIsql[1] -connStr "DSN=Example"
   ```

4. Enter the following to create a user account with name “Student” and password “mypsswrd”:

   ```
   Command> create user Student identified by mypsswrd;
   Command> grant all privileges to Student;
   Command> exit
   ```

2.4 Creating a Data Source for Oracle TimesTen

Real-Time Connect to Oracle TimesTen uses the Oracle TimesTen ODBC driver to access data sources (datastores). Usually these are the same data sources to which your applications connect. The connection information for each data source is stored in the **.odbc.ini** file. The stored information describes each data source in detail, specifying the driver name, a description, and any additional information the driver needs to connect to the data source.

To create the **.odbc.ini** file, follow these steps:

1. Create a new text file named **.odbc.ini** in your home directory. Alternatively, you can use the **ODBCINI** environment variable to specify the file location.
2. Insert these lines in the file:

   ```
   [ODBC Data Source]
   Example=TimesTen Driver

   [Example]
   DRIVER=/opt/TimesTen/tt1121/lib/libtten.so
   DataStore=/tmp/Example
   UID=Student
   PWD=mypsswrd
   DatabaseCharacterSet=AL32UTF8
   ```

1. If *ttisql* is not on your path, it can be found in the **bin/** directory of your Oracle TimesTen installation.

Note: If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB) (see Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache), you must add the attributes **OracleNetServiceName** and **OraclePWD** to the DSN. In addition, the attribute **DatabaseCharacterSet** must be set to the same character set that Oracle 11g uses. For example:

```
[Example]
DRIVER=/opt/TimesTen/tt1121/lib/libtten.so
DataStore=/tmp/Example
UID=Student
PWD=mypsswrd
OraclePWD=mypsswrd
DatabaseCharacterSet=WE8ISO8859P1
OracleNetServiceName=orcl11g
```

3. Save your changes.

2.5 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from an XML file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*/../../resource/xml/RTI_REAL_TIME_CONNECT.xml.** You can specify a different file with the command line option, **-cfgFile**.

The default file, **RTI_REAL_TIME_CONNECT.xml**, does not contain any valid database configuration information yet. For this example, we will edit this file as follows:

1. Look for the tag **<timesten_connection>**. Replace the tags **<dsn>**, **<user_name>**, and **<password>** as follows:

```
<timesten_connection>
  <dsn>Example</dsn>
  <user_name>Student</user_name>
  <password>mypsswrd</password>
</timesten_connection>
```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "Example" DSN.

2.6 Starting the Real-Time Connect to Oracle TimesTen Daemon

Start *Real-Time Connect* to Oracle TimesTen by executing the following command from **bin/<*arch*>** in the installation directory (where <*arch*> is the directory containing the executables compiled for your host platform, such as i86Linux2.6gcc3.4.3, x64Linux2.6gcc3.4.5).

```
> ./rtirtc_timesten -noDaemon -cfgName default
```

By default, *Real-Time Connect* to Oracle TimesTen runs in the background as a daemon process. Specifying the "**-noDaemon**" option prevents that and the *Real-Time Connect* Daemon starts as a regular process. Messages are sent to standard output.

You should see output similar to the following, indicating that the process is running.

```
>./rtirtc_timesten -noDaemon -cfgName default
RTI Real-Time Connect to TimesTen, Release 5.0.x: startup succeeded
```

Real-Time Connect to Oracle TimesTen is now connected to the "**Example**" data source.

Note: Make sure that you have the Oracle TimesTen libraries in your **LD_LIBRARY_PATH**, see Step 3 of Section 2.1.

2.7 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in an Oracle TimesTen data store. The first step is to create an IDL type definition. By using *rtidds-gen's*[1] **-example** option, you will automatically create a Publisher of this IDL type.

1. Create a new text file called "**MyType.idl**" with the following contents:

```
struct MyType {
    short pkey; //@key
    char message[13];
};
```

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *Connext* documentation for more information about how to run *rtiddsgen*.

This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the "**pkey**" field.

2. Now run the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example <arch> MyType.idl
   ```

   For example:

   ```
   > rtiddsgen -language C -example i86Linux2.6gcc4.1.1 MyType.idl
   ```

   This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

3. The generated example will have a makefile named:

   ```
   makefile_MyType_<arch>
   ```

   You may need to edit the makefile to specify the location of the compiler if it is not available on your path.

4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

   Insert the following lines immediately below this comment:

   ```
   instance->pkey = count;
   strcpy(instance->message, "Hello world!");
   ```

5. Save your changes and build the **MyType_publisher** and **MyType_subscriber** applications by executing:

   ```
   > gmake -f makefile_MyType_<arch>
   ```

6. Start the **MyType_publisher** application so that it starts publishing data samples.

   ```
   > objs/<arch>/MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the Oracle TimesTen data store yet. For this, you need to set up a subscription in *Real-Time Connect* to Oracle.

Subscriptions are set up in the "**RTIDDS_SUBSCRIPTIONS**" configuration table that *Real-Time Connect* to Oracle TimesTen created when it connected to the Oracle TimesTen data store. To insert entries into this table, you can use the Oracle TimesTen interactive SQL utility.

7. Start the TimesTen Interactive SQL utility from the command prompt:

```
> ttisql -connStr "DSN=Example"
```

8. You can see that the **RTIDDS_SUBSCRIPTIONS** configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon '**;**' at the end of the line*:

```
Command> select * from RTIDDS_SUBSCRIPTIONS;
0 rows found.
```

9. To store the samples from the **MyType_publisher** application in a table named **Example**, insert a corresponding entry into the **RTIDDS_SUBSCRIPTIONS** table:

```
Command> insert into RTIDDS_SUBSCRIPTIONS values
('Student', 'Example', 0, 'Example MyType', 'MyType');
1 row inserted.
```

This entry directs *Real-Time Connect* to Oracle TimesTen to create a user table named "**Example**" with owner "**Student**" and to start storing samples published with topic "**Example MyType**" and data type "**MyType**" in domain **0**.

10. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

```
Command> select * from Example;
```

The output will look something like this:

```
...
< 134, Hello world!\000, 0, 1 >
< 135, Hello world!\000, 0, 1 >
< 136, Hello world!\000, 0, 1 >
85 rows found.
```

The actual number of rows found depends on when exactly the *Real-Time Connect* Daemon started storing samples. If you execute the "**select**" statement repeatedly you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

2.8 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to Oracle TimesTen to automatically publish changes made to a table in a data source. For this, you will again use the Oracle TimesTen Interactive SQL utility to change a record in the "**Example**" table. We assume that you have followed the instructions in Section 2.6 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 through 4 in Section 2.7 to create the IDL file and generate and compile the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing "**Ctrl-c**" in the window where it is running.
2. Start the **MyType_subscriber** application from the command line:

    ```
    > objs/<arch>/MyType_subscriber
    MyType subscriber sleeping for 4 sec...
    MyType subscriber sleeping for 4 sec...
    ...
    ```

    You will see this message repeatedly, since nothing is being published.
3. For publishing changes to a data source, you need to configure the **RTIDDS_PUBLICATIONS** table. For publishing the changes to the **Example** table, execute the following SQL statement:

    ```
    Command> insert into RTIDDS_PUBLICATIONS values
    ('Student', 'Example', 0, 'Example MyType', 'MyType');
    ```

    This entry directs *Real-Time Connect* to Oracle TimesTen to start publishing changes to table **Example** of owner **Student** as the topic **Example MyType** with type **MyType** in domain **0**.
4. Now change one of the previously captured samples in the **Example** table, for instance the last one:

    ```
    Command> update Example set message = 'Hello again!' where
    pkey in (select max(pkey) from Example);
    ```

    You will see that the **MyType_subscriber** application reports the update, for example:

    ```
    ...
    MyType subscriber sleeping for 4 sec...
    pkey: 748
        message:
    ```

```
    message[ 0]: 'H'
    message[ 1]: 'e'
    message[ 2]: 'l'
    message[ 3]: 'l'
    message[ 4]: 'o'
    message[ 5]: ' '
    message[ 6]: 'a'
    message[ 7]: 'g'
    message[ 8]: 'a'
    message[ 9]: 'i'
    message[10]: 'n'
    message[11]: '!'
    message[12]: ' `
MyType subscriber sleeping for 4 sec...
...
```

5. You can also update all entries in the table:

   ```
   Command> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

2.9 Database Table Replication

This section explains how to use *Real-Time Connect* to replicate tables between two Oracle TimesTen DataStores.

Real-Time Connect uses lazy replication to send updates to other hosts in the network. Using lazy replication, a database update is sent to the subscribers after the transaction is committed.

1. Follow the instructions in Section 2.1 through Section 2.5 on a second host.
2. Set the tag **<enable_table_replication>** in **<general_options>** to true in the configuration files (**RTI_REAL_TIME_CONNECT.xml** in this example) on host1 and host2.
3. Start *Real-Time Connect* by running the following command from **bin\<architecture>** in the installation directory:

   ```
   host1> ./rtirtc_timesten -noDaemon -cfgName default
   host2> ./rtirtc_timesten -noDaemon -cfgName default
   ```

4. Start the TimesTen Interactive SQL utility from the command prompt:

```
host1> ttisql -connStr "DSN=Example"
host2> ttisql -connStr "DSN=Example"
```

5. Create a table named **ExampleRep** in host1 that will be replicated in host2:

```
Command> CREATE TABLE ExampleRep (
       pkey TT_SMALLINT NOT NULL,
       message TT_VARCHAR(13),
       RTIDDS_DOMAIN_ID TT_INTEGER,
       RTIRTC_REMOTE TT_INTEGER,
       PRIMARY KEY (pkey));
```

Note: RTIDDS_DOMAIN_ID and RTIRTC_REMOTE are meta-columns used by *Real-Time Connect* to control the replication process.

6. To replicate the **ExampleRep** table in host 1 and 2, set up a publication and a subscription in each one of the hosts:

   In host1:

```
Command> insert into RTIDDS_SUBSCRIPTIONS values
('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
Command> insert into RTIDDS_PUBLICATIONS values
('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
```

   In host2:

```
Command> insert into RTIDDS_SUBSCRIPTIONS values
('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
Command> insert into RTIDDS_PUBLICATIONS values
('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
```

7. Verify that the table **ExampleRep** has been created automatically in host 2:

```
Command> tables;
  STUDENT.EXAMPLEREP
  STUDENT.RTIDDS_PUBLICATIONS
  STUDENT.RTIDDS_SUBSCRIPTIONS
  STUDENT.RTIRTC_LOG
  STUDENT.RTIRTC_TBL_INFO
```

8. Insert a new row in host 1:

```
Command> insert into ExampleRep values (0, 'Hello World 0');
1 row inserted.
```

9. Verify that the row has been replicated in host 2:

```
Command> select * from ExampleRep;
```

```
< 0, Hello World 0, 0, 1 >
1 row found.
```

10. Modify the row in host 2:

```
Command> update ExampleRep set message = 'Hello World 1';
1 row updated.
```

11. Verify that the change has been propagated to host 1:

```
Command> select * from ExampleRep;
< 0, Hello World 1, 0, 123 >
1 row found.
```

12. Delete the row in host 1:

```
Command> delete from ExampleRep;
1 row deleted.
```

13. Verify that the change has been propagated to host 2:

```
Command> select * from ExampleRep;
0 rows found.
```

Chapter 3 Real-Time Connect to Oracle TimesTen on Windows Systems

This chapter provides instructions on how to install *Real-Time Connect* to Oracle TimesTen on Windows platforms.

This chapter assumes you are using Microsoft Visual Studio® 2005[1].

3.1 Installation

Install *Real-Time Connect* to Oracle TimesTen from the distribution file:

1. Unzip the distribution file **RTI_Real-Time_Connect_5.0.x.win32bundle.zip** for 32-bit Windows systems or **RTI_Real-Time_Connect_5.0.x.win64bundle.zip** for 64-bit Windows systems. (Your filename will be slightly different, depending on the version number.)

   This will extract the *Real-Time Connect* to Oracle TimesTen **rtirtc_setup.exe** file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 11.2.1 distribution file.

2. If Oracle TimesTen 11.2.1 is not installed on your system, install it by unzipping the Oracle TimesTen distribution file, and run **setup.exe** following the instructions in the *Oracle TimesTen In-Memory Database Installation Guide*.

1. You may use other supported Microsoft compilers such as Visual Studio 2010, Visual Studio 2008, Visual Studio .Net 2003 or Visual Studio 6.0, however, the instructions in this chapter were written assuming Visual Studio 2005.

- When installing Oracle TimesTen, you can accept all default options with the following exceptions:

  If you want to allow direct database access to users of any OS group, make sure to select the checkbox in the ‘File and Data Accessibility’ screen. If you do not select this option, permissions are restricted to users who are members of the Administrator group.

  If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB), be sure that during the installation process you configure the environment variable TNS_ADMIN with the path of the tsnnames.ora file.

- *Real-Time Connect* to Oracle TimesTen requires the installation of the Oracle TimesTen Data Manager.

3. Verify that Oracle TimesTen In-Memory database is properly installed on your system.

   a. Check that the **TimesTen 11.2.1** Start menu shortcut has been added to the **Windows Desktop, Start, Programs** menu.

   b. Open the ODBC Data Source Administrator.

      On the Windows Desktop, choose **Start**, **Settings, Control Panel**, **Administrative Tools, Data Sources (ODBC)**.

   c. Click **Drivers**.

      Check to see that the correct drivers are installed. At least you should see the TimesTen Data Manager driver. Click OK.

   d. Check to see if the TimesTen Data Manager 11.2.1 service has started

      On the Windows Desktop, choose **Start**, **Settings, Control Panel, Administrative Tools, Services** and check that the TimesTen Data Manager 11.2.1 service has the word “**Started**” in the Status field.

4. Install *Real-Time Connect* to Oracle TimesTen by running the “**rtirtc_setup.exe**” file. Follow the on-screen instructions to complete the installation.

5. Please read Chapter 9: License Installation.

3.2 Creating an Oracle TimesTen Database

Before you can use *Real-Time Connect* to Oracle TimesTen, you need to get a TimesTen user account to an existing data store (database) from your database administrator. This section explains how to create a TimesTen database when you act as your own database administrator.

1. Log in as the instance administrator (the user that installed TimesTen).
2. Create a DSN for the new TimesTen database as described in Section 3.2.1.
3. Create the new datastore by executing the following:

```
> ttIsql1 -connStr "DSN=ExampleAdmin"
Command> exit
```

3.2.1 Creating a Data Source

To add a data source:

1. Open the ODBC Data Source Administrator:
   - On Windows XP and Windows Server 2003 systems: Select **Start, Control Panel, Administrative Tools, Data Sources (ODBC)**.
   - On Windows Vista systems: Select **Start, Control Panel, System and Maintenance, Administrative Tools, Data Sources (ODBC)**.
   - On Windows 7 and Windows Server 2008 systems: Select **Start, Control Panel, System and Security, Administrative Tools, Data Sources (ODBC).**
2. Select the **User DSN** tab.
3. Click **Add...**; the **Create New Data Source** dialog appears.
4. Select **TimesTen Data Manager 11.2.1** from the list of drivers.
5. Click **Finish**; the **TimesTen ODBC Setup** dialog appears.
6. Fill in the fields in the dialog.
   a. Enter "**ExampleAdmin**" as the **Data Source Name** (DSN).
   b. Enter a **Data Store Path**; the path must exist, so you may want to create a new directory first.

1. If *ttisql* is not on your path, it can be found in the **bin** directory of your Oracle TimesTen installation.

The data-store path name uniquely identifies the physical data-store. It is the full *path* name of the data-store (e.g., **C:\data\AdminData**). Note that this is not a *file* name—the actual data-store file names have suffixes (e.g., **C:\data\AdminData.ds0**, **C:\data\AdminData.log0**).

   c. Be sure the **Temporary** check box is *not* selected.

   d. Set **Database Character Set** to AL32UTF8.

   If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB) (see Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache), make sure that the attribute **DatabaseCharacterSet** is set to the same character set used by Oracle 11g.

   You can determine the Oracle database character set by executing the following query on SQL*Plus (as any user):

```
SQL> SELECT value FROM nls_database_parameters WHERE
parameter='NLS_CHARACTERSET';
```

   e. All other fields can be left empty.

7. Select the **IMDB Cache** tab if you want to integrate with Oracle In-Memory Database Cache.

   a. Enter the Oracle Net service Name. For example, “orcl11g”.

8. Click **OK**.

3.3 Creating an Oracle TimesTen User Account

Starting with Oracle Timesten 11.2.1, database access control is mandatory.

If you are acting as your own database and system administrator, use the following steps to create an account for the user “Student”:

1. Log in as the instance administrator (the user that installed TimesTen).

2. Connect to the Oracle TimesTen database where the user account must be created:

```
> ttIsql -connStr "DSN=ExampleAdmin"
```

3. Enter the following to create a user account with the name “Student” and password “mypsswrd”:

```
Command> create user Student identified by mypsswrd;
Command> grant all privileges to Student;
Command> exit
```

3.4 Creating a Data Source for Oracle TimesTen

Real-Time Connect to Oracle TimesTen uses the Oracle TimesTen ODBC driver to access data sources (datastores). In this section you will create a data source (DSN) for *Real-Time Connect* to Oracle.

To add a data source:

1. Open the ODBC Data Source Administrator:
   - On Windows XP and Windows Server 2003 systems: Select **Start, Control Panel, Administrative Tools, Data Sources (ODBC)**.
   - On Windows Vista systems: Select **Start, Control Panel, System and Maintenance, Administrative Tools, Data Sources (ODBC)**.
   - On Windows 7 and Windows Server 2008 systems: Select **Start, Control Panel, System and Security, Administrative Tools, Data Sources (ODBC).**
2. Select the **User DSN** tab.

   In this document *Real-Time Connect* will be run from the command line. If you want to run the daemon as a Window service select **System DSN** instead of User DSN.
3. Click **Add**; the **Create New Data Source** dialog appears.
4. Select **TimesTen Data Manager 11.2.1** from the list of drivers.
5. Click **Finish**; the **TimesTen ODBC Setup** dialog appears.
6. Fill out the fields in the dialog.

   a. Enter “**Example**” as the **Data Source Name** (DSN).

   b. Enter a **Data Store Path**; the path must exist, so you may want to create a new directory first. Make sure that the value of this field is the same value that was used to create the DSN in Section 3.2.1.

The data-store path name uniquely identifies the physical data-store. It is the full *path* name of the data-store (e.g., **C:\data\AdminData**). Note that this is not a *file* name—the actual data-store file names have suffixes (e.g., **C:\data\AdminData.ds0**, **C:\data\AdminData.log0**).

c. Be sure the **Temporary** check box is *not* selected.

d. Set **Database Character Set** to AL32UTF8.

If you want to use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB) (see Chapter 6: Real-Time Connect to TimesTen with Oracle In-Memory Database Cache), be sure that the attribute **DatabaseCharacterSet** is set to the same character set that Oracle 11g uses.

You can determine the Oracle database character set by executing the following query on SQL*Plus as any user:

SQL> SELECT value FROM nls_database_parameters WHERE parameter='NLS_CHARACTERSET';

e. All other fields can be left empty.

7. Select the **General Connection** tab and enter "**Student**" as the **User ID**.

8. Select the **IMDB Cache** tab if you want to integrate with Oracle In-Memory Database Cache.

   a. Enter the Oracle Net service Name. For example "orcl11g".

   b. Enter the Oracle Password of the user 'Student' in the Oracle 11g database.

9. Click **OK**.

3.5 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from a file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*/../../resource/xml/RTI_REAL_TIME_CONNECT.xml**. You can specify a different file with the command line option, **-cfgFile**.

The default file, **RTI_REAL_TIME_CONNECT.xml**, does not contain any valid database configuration information yet. For this example, we will edit this file as follows:

1. Look for the tag **<timesten_connection>**. Replace the tags **<dsn>. <user_name>,** and **<password>** as follows:

```
 <timesten_connection>
    <dsn>Example</dsn>
    <user_name>Student</user_name>
    <password>mypsswrd</password>
</timesten_connection>
```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "**Example**" DSN.

3.6 Starting the Real-Time Connect to Oracle Daemon

You can start the *Real-Time Connect* Daemon as a Windows service (assuming that you allowed the installation program to do so, which it does by default). However, for the example in this chapter, we will start the daemon manually.

Start *Real-Time Connect* to Oracle TimesTen by executing the following command from **bin\i86Win32** (for 32-bit Windows systems) or **bin\x64Win64VS2010** (for 64-bit Windows systems) in the installation directory.

```
> rtirtc_timesten -noDaemon -cfgName default
```

By default, *Real-Time Connect* to Oracle TimesTen runs in the background as Windows service. Specifying the "**-noDaemon**" option prevents that, and starts up the *Real-Time Connect* Daemon as a regular process. Messages are sent to standard output.

You should see output similar to the following, indicating that the process is running.

```
> rtirtc_timesten -noDaemon -cfgName default
RTI Real-Time Connect to TimesTen, Release 5.0.x: startup succeeded
```

Real-Time Connect to Oracle TimesTen is now connected to the "**Example**" data source.

3.7 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in a Oracle TimesTen data store. The first step is to create an IDL type definition. By using *rtiddsgen's*[1] **-example** option, you will automatically create a Publisher of this IDL type.

1. Create a new text file called **MyType.idl** with the following contents:

```
struct MyType {
   short pkey; //@key
   char message[13];
};
```

This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the **pkey** field.

2. Now execute the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example i86Win32VS2005[1] MyType.idl
   ```

   This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

3. The generated example will also have a Visual Studio solution file, **MyType-vs2005.sln**. Start Microsoft Visual Studio 2005 and load this solution by clicking on this file.

4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

   Insert the following lines immediately below this comment:

   ```
   instance->pkey = count;
   strcpy(instance->message, "Hello world!");
   ```

5. Save your changes and build the **MyType_publisher** project in Visual Studio.

6. Start the **MyType_publisher** application so that it starts publishing data samples. From a command prompt,

   ```
   > objs\i86Win32VS2005\MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the Oracle TimesTen data store yet. For this, you need to set up a subscription in *Real-Time Connect* to Oracle TimesTen.

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *Connext* documentation for more information about how to run *rtiddsgen*.

1. If you are using a different supported compiler, you will need to use a different value here, such as i86Win32VS2010 for Visual Studio 2010.

Subscriptions are set up in the **RTIDDS_SUBSCRIPTIONS** configuration table that *Real-Time Connect* to Oracle TimesTen created when it connected to the Oracle TimesTen data store. For inserting entries into this table, you can use the Oracle TimesTen interactive SQL utility.

7. Start the TimesTen Interactive SQL utility from the command prompt:

```
> ttisql -connStr 'DSN=Example'
```

8. You can see that the **RTIDDS_SUBSCRIPTIONS** configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon '**;**' at the end of the line*:

```
Command> select * from RTIDDS_SUBSCRIPTIONS;
0 rows found.
```

9. To store the samples from the **MyType_publisher** application in a table named **Example**, you insert a corresponding entry into the **RTIDDS_SUBSCRIPTIONS** table:

```
Command> insert into RTIDDS_SUBSCRIPTIONS values
('Student', 'Example', 0, 'Example MyType', 'MyType');
1 row inserted.
```

This entry directs *Real-Time Connect* to Oracle TimesTen to create a user table named **Example** with owner **Student** and to start storing samples published with topic **Example MyType** and data type **MyType** in Domain **0**.

10. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

```
Command> select * from Example;
```

The output will look something like this:

```
...
< 134, Hello world!, 0, 1 >
< 135, Hello world!, 0, 1 >
< 136, Hello world!, 0, 1 >
85 rows found.
```

The actual number of rows found depends on when exactly the *Real-Time Connect* Daemon started storing samples. If you execute the **select** statement repeatedly you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

3.8 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to Oracle TimesTen to automatically publish changes made to a table in a data source. For this, you will again use the Oracle TimesTen Interactive SQL utility to change a record in the **Example** table. We assume that you have followed the instructions in Section 3.6 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 through 4 in Section 3.7 to create the IDL file and generate and compile the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing **Ctrl-c** in the window where it is running.
2. Build the **MyType_subscriber** project in Visual Studio and start the application from the command line:

```
> objs\i86Win32VS2005\MyType_subscriber
MyType subscriber sleeping for 4 sec...
MyType subscriber sleeping for 4 sec...
...
```

   You will get this message repeatedly, since nothing is being published.

3. For publishing changes to a data source, you need to configure the **RTIDDS_PUBLICATIONS** table. For publishing the changes to the **Example** table, execute the following SQL statement:

```
Command> insert into RTIDDS_PUBLICATIONS values
('Student', 'Example', 0, 'Example MyType', 'MyType');
```

   This entry directs *Real-Time Connect* to Oracle TimesTen to start publishing changes to table **Example** of owner **Student** as the topic **Example MyType** with type **MyType** in Domain **0**.

4. Now change one of the previously captured samples in the **Example** table, for instance the last one:

```
Command> update Example set message = 'Hello again!' where pkey
in (select max(pkey) from Example);
```

   You will see that the **MyType_subscriber** application reports the update, for example:

```
...
MyType subscriber sleeping for 4 sec...
pkey: 748
   message:
         message[ 0]: 'H'
         message[ 1]: 'e'
         message[ 2]: 'l'
         message[ 3]: 'l'
         message[ 4]: 'o'
         message[ 5]: ' '
         message[ 6]: 'a'
         message[ 7]: 'g'
         message[ 8]: 'a'
         message[ 9]: 'i'
         message[10]: 'n'
         message[11]: '!'
         message[12]: <0>
   MyType subscriber sleeping for 4 sec...
   ...
```

5. You can also update all entries in the table:

   ```
   Command> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

3.9 Database Table Replication

This section explains how to use *Real-Time Connect* to replicate tables between two Oracle TimesTen DataStores.

Real-Time Connect uses lazy replication to send updates to other hosts in the network. Using lazy replication, a database update is sent to the subscribers after the transaction is committed.

1. Follow the instructions in Section 3.1 through Section 3.5 on a second host.
2. Set the tag **<enable_table_replication>** in **<general_options>** to true in the configuration files on host1 and host2.
3. Start *Real-Time Connect* by running the following command from **bin\i86Win32VS2005** in the installation directory:

   ```
   host1> rtirtc_timesten -noDaemon -cfgName default
   host2> rtirtc_timesten -noDaemon -cfgName default
   ```

4. Start the TimesTen Interactive SQL utility from the command prompt:

```
host1> ttisql -connStr "DSN=Example"
host2> ttisql -connStr "DSN=Example"
```

5. Create a table named **ExampleRep** in host 1 that will be replicated in host2:

   ```
   Command> CREATE TABLE ExampleRep (
       pkey TT_SMALLINT NOT NULL,
       message TT_VARCHAR(13),
       RTIDDS_DOMAIN_ID TT_INTEGER,
       RTIRTC_REMOTE TT_INTEGER,
       PRIMARY KEY (pkey));
   ```

   Note: RTIDDS_DOMAIN_ID and RTIRTC_REMOTE are meta-columns used by *Real-Time Connect* to control the replication process.

6. To replicate the **ExampleRep** table in host 1 and 2, set up a publication and a subscription in each one of the hosts:

   On host1:

   ```
   Command> insert into RTIDDS_SUBSCRIPTIONS values
   ('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
   Command> insert into RTIDDS_PUBLICATIONS values
   ('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
   ```

   On host2:

   ```
   Command> insert into RTIDDS_SUBSCRIPTIONS values
   ('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
   Command> insert into RTIDDS_PUBLICATIONS values
   ('Student', 'ExampleRep', 0, 'Example MyTypeRep', 'MyTypeRep');
   ```

7. Verify that the table **ExampleRep** has been created automatically in host 2:

   ```
   Command> tables;
     STUDENT.EXAMPLEREP
     STUDENT.RTIDDS_PUBLICATIONS
     STUDENT.RTIDDS_SUBSCRIPTIONS
     STUDENT.RTIRTC_LOG
     STUDENT.RTIRTC_TBL_INFO
   ```

8. Insert a new row on host 1:

   ```
   Command> insert into ExampleRep values (0, 'Hello World 0');
   1 row inserted.
   ```

9. Verify that the row has been replicated on host 2:

   ```
   Command> select * from ExampleRep;
   < 0, Hello World 0, 0, 1 >
   1 row found.
   ```

10. Modify the row in host 2:

```
Command> update ExampleRep set message = 'Hello World 1';
1 row updated.
```

11. Verify that the change has been propagated to host 1:

```
Command> select * from ExampleRep;
< 0, Hello World 1, 0, 123 >
1 row found.
```

12. Delete the row on host 1:

```
Command> delete from ExampleRep;
1 row deleted.
```

13. Verify that the change has been propagated to host 2

```
Command> select * from ExampleRep;
0 rows found.
```

Chapter 4 Real-Time Connect to Oracle 11g on UNIX-based Systems

This chapter provides instructions on how to install *Real-Time Connect* to Oracle 11g on the following platforms:

❑ Red Hat® Enterprise Linux® 4.0 and 5.0 on Pentium®-class processors

4.1 Installation

First, verify that Oracle 11g Release 2 is installed and running on your system.

The installation of Oracle 11g is beyond the scope of this document. Please refer to Oracle documentation for the process to install and configure Oracle 11g.

Install *Real-Time Connect* to Oracle from the distribution file:

1. Unpack the distribution file to a temporary location using GNU™'s version of tar, gtar. There are different distribution files depending on if you are running on a 32-bit or 64-bit version of the operating system.

   - **On Linux systems** (your filename will be slightly different, depending on the version number/letter):

     ```
     > tar zxfv RTI_Real-Time_Connect_5.0.x.linux32bundle.tar.gz
     ```

     or

     ```
     > tar zxfv RTI_Real-Time_Connect_5.0.x.linux64bundle.tar.gz
     ```

This will extract the compressed *Real-Time Connect* to Oracle file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 11.2.1 distribution file. When using *Real-Time Connect* to interface with Oracle 11g directly, there is no need to have Oracle TimesTen installed.

2. Unpack the compressed *Real-Time Connect* to Oracle installation file in a location of your choice:

   - **On Linux systems** (your filename will be slightly different, depending on the version number/letter):

     ```
     > tar zpxfv RTI_Real-Time_Connect_5.0.x.linux32.tar.gz
     ```

     or

     ```
     > tar zpxfv RTI_Real-Time_Connect_5.0.x.linux64.tar.gz
     ```

   This gives you four directories, **bin/**, **docs/**, **lib/**, **resource/**, containing the *Real-Time Connect* to Oracle executables, user documentation, libraries, and resources, respectively.

3. Add the path to the Oracle 11g libraries to the beginning of **LD_LIBRARY_PATH**. For example:

   ```
   > setenv LD_LIBRARY_PATH /opt/oracle/product/11.2.0/dbhome_1/
   lib:$LD_LIBRARY_PATH
   ```

   or

   ```
   > export LD_LIBRARY_PATH=/opt/oracle/product/11.2.0/dbhome_1/
   lib:$LD_LIBRARY_PATH
   ```

4. Set ORACLE_HOME. For example:

   ```
   > setenv ORACLE_HOME /opt/oracle/product/11.2.0/dbhome_1
   or
   > export ORACLE_HOME=/opt/oracle/product/11.2.0/dbhome_1
   ```

5. Please read Chapter 9: License Installation.

4.2 Configuring the Oracle Server

Real-Time Connect Daemon uses external procedures, EXTPROCs, to interface with the Oracle server. Or more accurately, *Real-Time Connect* provides external procedures in a library, **librtirtc_oracleq.so**, that the Oracle server must be able to load while running to communicate with the *Real-Time Connect* Daemon. Chapter 4 in the *Real-Time Connect*

User's Manual provides a detailed process to follow in order to install and configure the Oracle server to use **librtirtc_oracleq.so**.

This section provides a procedure with the least amount of configuration. Alternative ways to install the files required to support *Real-Time Connect*'s external procedures can be found in the accompanying *Real-Time Connect User's Manual.*

4.2.1 Install librtirtc_oracleq.so

Copy the appropriate version of **librtirtc_oracleq.so** into **$ORACLE_HOME/bin** on the server host. **$ORACLE_HOME** is the installation directory of the Oracle 11g database. This library is distributed with *Real-Time Connect* to Oracle and can be found in **lib/<*arch*>** in the installation directory. The correct version of the library to use depends on the platform on which the Oracle server is running. For example, <*arch*> might be i86Linux2.6gcc4.1.1.

You can also configure the Oracle server to find the shared library in directories outside of the Oracle installation tree. Please see Chapter 4 in the *Real-Time Connect User's Manual*.

4.2.2 Install libnddsc.so and libnddscore.so

This section only applies to Linux systems.

librtirtc_oracleq.so uses *Connext* and thus the shared libraries **libnddsc.so** and **libnddscore.so** must also be installed and be accessible at runtime by the Oracle server.

Add the directory containing the appropriate *Connext* libraries to the environment variable, **LD_LIBRARY_PATH**, for the user who starts the Oracle server. You may need to restart the Oracle server after this variable has been changed.

Other ways of installing **libnddsc.so** and **libnddscore.so** on the Oracle server can be found in Chapter 4 in the *Real-Time Connect User's Manual*.

4.2.3 Create an Oracle Account

Before you can use *Real-Time Connect* to Oracle, you need to obtain an Oracle account from your database administrator. If you are acting as your own database administrator, start **SQL*Plus** from the command prompt on the Oracle server and log in as the system manager:

```
> sqlplus system/<password>@<Oracle_Service_Name>
```

For example, to create a new Oracle account for “**Student**” on the Oracle database identified by the connection string “**orcl11g**” with system manager password “**mypsswrd**”, enter the following:[1]

```
> sqlplus system/mypsswrd@orcl
SQL> CREATE USER Student IDENTIFIED BY mypsswrd DEFAULT TABLESPACE
users;
SQL> GRANT connect, resource, create any trigger, create any
library, create any table TO Student;
SQL> COMMIT;
SQL> EXIT
```

The remaining sections in this chapter assume that an Oracle user named "**Student**" with the password "**mypsswrd**" has an account for the “**orcl11g**” database.

4.3 Creating a Data Source for Oracle

Real-Time Connect to Oracle usually connects to the same data source or data sources to which your applications connect. The connection information for each data source is stored in the “**.odbc.ini**” file. The stored information describes each data source in detail, specifying the driver name, a description, and any additional information the driver needs to connect to the data source.

To create the “**.odbc.ini**” file, follow these steps:

1. Create a new file named “**.odbc.ini**” in your home directory using your favorite text editor. Alternatively, you can use the **ODBCINI** environment variable to specify the file location.
2. Insert these lines in the file:

   ```
   [ODBC Data Source]
   Example=Oracle Driver

   [Example]
   DRIVER=/opt/oracle/product/11.2.0/dbhome_1/lib/libsqora.so.11.1
   Servername=ORCL11G
   ```

   NOTE Make sure that **DRIVER** points to the valid location of the Oracle driver on your system.

1. If *sqlplus* is not on your path, it can be found in the **bin/** directory of your Oracle installation.

3. Save your changes.

4.4 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from a file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*/../../resource/xml/RTI_REAL_TIME_CONNECT.xml**. You can specify a different file with the command line option, **-cfgFile**.

The default file, **RTI_REAL_TIME_CONNECT.xml**, does not contain any actual database configuration information yet. For this example we will edit this file as follows:

1. Look for the tag **<oracle_connection>**. Replace the tags **<dsn>, <user_name>,** and **<password>** as follows:

```
 <oracle_connection>
   <dsn>Example</dsn>
   <user_name>Student</user_name>
   <password>mypsswrd</password>
</oracle_connection>
```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "Example" DSN.

4.5 Starting the Real-Time Connect to Oracle Daemon

Start *Real-Time Connect* to Oracle by running the following command from **bin/<*arch*>/** in the installation directory. **<*arch*>** is the directory containing the executables compiled for your host platform (such as i86Linux2.6gcc3.4.3 or x64Linux2.6gcc3.4.5).

```
> ./rtirtc_oracle -noDaemon -cfgName default
```

By default, *Real-Time Connect* to Oracle runs in the background as a daemon process. Specifying the "**-noDaemon**" option prevents that, and starts up the *Real-Time Connect* Daemon as a regular process. Messages are sent to standard output.

You should see the following output, indicating that the process is running.

```
>./rtirtc_oracle -noDaemon -cfgName default
```

```
RTI Real-Time Connect to Oracle, Release 5.0.x: startup succeeded
```

Real-Time Connect to Oracle is now connected to the "**Example**" data source.

Note: Make sure that you have the Oracle 11g libraries on your **LD_LIBRARY_PATH**, see Step 3 of Section 4.1.

4.6 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in an Oracle 11g database. The first step is to create an IDL type definition. By using the "**-example**" option of *rtiddsgen*[1] you will automatically create a Publisher of this IDL type.

1. Create a new text file called "**MyType.idl**" with the following contents:

   ```
   struct MyType {
       short pkey; //@key
       char message[13];
   };
   ```

   This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the "**pkey**" field.

2. Now execute the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example <arch> MyType.idl
   ```

   For example, **<arch>** can be:

   ```
   > rtiddsgen -language C -example i86Linux2.6gcc4.1.1 MyType.idl
   ```

   This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

3. The generated example will also have a makefile named:

   ```
   makefile_MyType_<arch>
   ```

   You may need to edit the makefile to specify the location of the compiler if it is not available on your path.

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *Connext* documentation for more information about how to run *rtiddsgen*.

4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

   Insert the following lines immediately below this comment:

   ```
   instance->pkey = count;
   strcpy(instance->message, "Hello world!");
   ```

5. Save your changes and build the **MyType_publisher** and **MyType_subscriber** applications by executing:

   ```
   > gmake -f makefile_MyType_<arch>
   ```

6. Start the **MyType_publisher** application so that it starts publishing data samples.

   ```
   > objs/<arch>/MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the Oracle 11g database yet. For this you need to set up a subscription in *Real-Time Connect* to Oracle.

   Subscriptions are set up in the "**RTIDDS_SUBSCRIPTIONS**" configuration table that *Real-Time Connect* to Oracle created when it connected to the Oracle database. To insert entries into this table, you can use the Oracle's **SQL*Plus** utility again.

7. Start the **SQL*Plus** utility from the command prompt:[1]

   ```
   > sqlplus student/mypsswrd@orcl11g
   SQL> set autocommit on;[2]
   ```

8. You can see that the "**RTIDDS_SUBSCRIPTIONS**" configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon* '**;**' *at the end of the line*:

   ```
   SQL> select * from RTIDDS_SUBSCRIPTIONS;
   0 rows found.
   ```

1. If *sqlplus* is not on your path, it can be found in the **bin/** directory of your Oracle installation.
2. By default, *sqlplus* is not in autocommit mode. Use this statement to turn on autocommit.

9. To store the samples from the **MyType_publisher** application in a table named "**Example**", you insert a corresponding entry into the "**RTIDDS_SUBSCRIPTIONS**" table:

```
SQL> insert into RTIDDS_SUBSCRIPTIONS (table_owner, table_name,
domain_id, topic_name, type_name) values
('Student', 'Example', 0, 'Example MyType', 'MyType');
1 row inserted.
```

   This entry directs *Real-Time Connect* to Oracle to create a user table named "**Example**" with owner "**Student**" and to start storing samples published with topic "**Example MyType**" and data type "**MyType**" in Domain **0**.

10. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

```
SQL> select * from Example;
```

   The output will look something like this:

```
      PKEY MESSAGE       RTIDDS_DOMAIN_ID RTIRTC_REMOTE
---------- ------------- ---------------- -------------
        90 Hello world!                44             1
        91 Hello world!                44             1
        92 Hello world!                44             1
...
```

   The actual number of rows found depends on exactly when the *Real-Time Connect* Daemon started storing samples. If you execute the "**select**" statement repeatedly, you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

4.7 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to Oracle to automatically publish changes made to a table in a data source. For this, you will again use the Oracle's **SQL*Plus** utility to change a record in the "**Example**" table.

We assume that you have followed the instructions in Section 4.5 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 to 4 in Section 4.6 to create the IDL file and generate and compile the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing "**Ctrl-c**" in the window where it is running.
2. Start the **MyType_subscriber** application from the command line:

   ```
   > objs/<arch>/MyType_subscriber
   MyType subscriber sleeping for 4 sec...
   MyType subscriber sleeping for 4 sec...
   ...
   ```

   You will get this message repeatedly, since nothing is being published.
3. For publishing changes to a data source, you need to configure the "**RTIDDS_PUBLICATIONS**" table. For publishing the changes to the "**Example**" table, execute the following SQL statement:

   ```
   SQL> insert into RTIDDS_PUBLICATIONS (table_owner, table_name,
   domain_id, topic_name, type_name) values
   ('Student', 'Example', 0, 'Example MyType', 'MyType');
   ```

   This entry directs *Real-Time Connect* to Oracle to start publishing changes to table "**Example**" of owner "**Student**" as the topic "**Example MyType**" with type "**MyType**" in the Domain **0**.
4. Now change one of the previously captured samples in the "**Example**" table, such as the last one:

   ```
   SQL> update Example set message = 'Hello again!' where pkey in
   (select max(pkey) from Example);
   ```

   You will see that the **MyType_subscriber** application reports the update, for example:

   ```
   ...
   MyType subscriber sleeping for 4 sec...
   pkey: 748
      message:
         message[ 0]: 'H'
         message[ 1]: 'e'
         message[ 2]: 'l'
         message[ 3]: 'l'
         message[ 4]: 'o'
         message[ 5]: ' '
         message[ 6]: 'a'
         message[ 7]: 'g'
         message[ 8]: 'a'
         message[ 9]: 'i'
   ```

```
      message[10]: 'n'
      message[11]: '!'
      message[12]: <0>
MyType subscriber sleeping for 4 sec...
...
```

5. You can also update all entries in the table:

   ```
   SQL> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

Chapter 5 Real-Time Connect to Oracle 11g on Windows Systems

This chapter provides instructions on how to install *Real-Time Connect* to Oracle 11g on Windows platforms.

This chapter assumes you are using Microsoft Visual Studio® 2005[1].

5.1 Installation

Before installing *Real-Time Connect* to Oracle, verify that Oracle 11g Release 2 is installed and running on your system. The installation of Oracle 11g is beyond the scope of this document. Please refer to Oracle documentation for the process to install and configure Oracle 11g.

Install *Real-Time Connect* to Oracle from the distribution file:

1. Unzip the distribution file **RTI_Real-Time_Connect_5.0.x.win32bundle.zip** for 32-bit Windows systems or **RTI_Real-Time_Connect_5.0.x.win64bundle.zip** for 64-bit Windows systems. (Your filename will be slightly different, depending on the version number.)

1. You may use other supported Microsoft compilers such as Visual Studio 2010, Visual Studio 2008, Visual Studio .NET 2003, or Visual Studio 6.0, however, the instructions in this chapter were written assuming Visual Studio 2005.

This will extract the *Real-Time Connect* to Oracle **rtirtc_setup.exe** file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 11.2.1 distribution file. When using *Real-Time Connect* to interface with Oracle 11g directly, you do not need to have Oracle TimesTen installed.

2. Install *Real-Time Connect* to Oracle by running the "**rtirtc_setup.exe**" file. Follow the on-screen instructions to complete the installation.
3. Please read Chapter 9: License Installation.

5.2 Configuring the Oracle Server

Real-Time Connect Daemon uses external procedures, EXTPROCs, to interface with the Oracle server. Or more accurately, *Real-Time Connect* provides external procedures in a library, **rtirtc_oracleq.dll**, that the Oracle server must be able to load while running to communicate with the *Real-Time Connect* Daemon. Chapter 4 in the *Real-Time Connect User's Manual* provides a detailed process to follow in order to install and configure the Oracle server to use **rtirtc_oracleq.dll**.

This section provides a procedure with the least amount of configuration. Alternative ways to install the files required to support *Real-Time Connect*'s external procedures can be found in the *Real-Time Connect User's Manual.*

5.2.1 Install rtirtc_oracleq.dll

Make sure that the environment variable **ORACLE_HOME** exists and is set correctly to the directory containing the installation of Oracle.

Copy the appropriate version of **rtirtc_oracleq.dll** into **%ORACLE_HOME%\bin** on the server host. **%ORACLE_HOME%** is the installation directory of the Oracle 11g database. This library is distributed with *Real-Time Connect* to Oracle and can be found in **lib\<*architecture*>** in the installation directory.

You can also configure the Oracle server to find the shared library in directories outside of the Oracle installation tree. Please consult Chapter 4 in the *Real-Time Connect User's Manual*.

5.2.2 Install nddsc.dll and nddscore.dll

Because **rtirtc_oracleq.dll** uses *Connext*, the shared libraries **nddsc.dll** and **nddscore.dll** must also be installed and accessible at run time by the Oracle server.

Important! Make sure your **Path** system environment variable[1] contains the path to the *Connext* libraries (**<NDDSHOME>\lib\<*arch*>,** where <*arch*> depends on your architecture, such as i86Win32VS2008). If **<NDDSHOME>\lib\<*arch*>** is not already in your **Path**, add it now. You will need to restart your computer after you modify **Path**. If you do not want to restart the computer, you can copy the libraries into a directory that is already in the system **Path**, such as **c:\Windows\System32**.

5.2.3 Create an Oracle Account

Before you can use *Real-Time Connect* to Oracle, you need to obtain an Oracle account from your database administrator. If you are acting as your own database administrator, start **SQL*Plus** from the command prompt on the Oracle server and log in as the system manager:[2]

```
> sqlplus system/<password>@<Oracle_Service_Name>
```

For example, to create a new Oracle account for "**Student**" on the Oracle database identified by the connection string "**orcl**" with system manager password "**mypsswrd**", enter the following:

```
> sqlplus system/mypsswrd@orcl
SQL> CREATE USER Student IDENTIFIED BY mypsswrd DEFAULT TABLESPACE
users;
SQL> GRANT connect, resource, create any trigger, create any
library, create any table TO Student;
SQL> COMMIT;
SQL> EXIT
```

The remaining sections in this chapter assume that an Oracle user named "**Student**" with the password "**mypsswrd**" has an account for the "**orcl**" database.

1. To view and/or edit the **Path** system environment variable: from the **Start** button/menu, select **Settings, Control Panel, System, Advanced tab,** then select the **Environment Variables** button. View or edit the **Path** in the System Variables. (The exact steps for accessing the environment variables may vary, depending on your version of the Windows operating system.)

2. If *sqlplus* is not on your path, it can be found in the **bin** directory of your Oracle installation.

5.3 Creating a Data Source for Oracle

Real-Time Connect to Oracle 11g usually connects to the same data source or data sources to which your applications connect. The connection information for each data source is stored in the Windows registry. The stored information describes each data source in detail, specifying the driver name, a description, and any additional information the driver needs to connect to the data source.

To add a data source, follow these steps:

1. Open the ODBC Data Source Administrator:
   - On Windows XP and Windows Server 2003 systems: Select **Start, Control Panel, Administrative Tools, Data Sources (ODBC)**.
   - On Windows Vista systems: Select **Start, Control Panel, System and Maintenance, Administrative Tools, Data Sources (ODBC)**.
   - On Windows 7 and Windows Server 2008 systems: Select **Start, Control Panel, System and Security, Administrative Tools, Data Sources (ODBC).**
2. Select the **User DSN** tab.
3. Click the **Add** button; the **Create New Data Source** dialog appears.
4. Select the **Oracle 11g** driver from the list of drivers.
5. Click the **Finish** button; the **Oracle ODBC Driver Configuration** dialog appears.
6. Fill out the fields in the dialog.
   a. Enter “**Example**” as the **Data Source Name** (DSN).
   b. Select the **TNS Service Name**, and enter "**Student**" as the User ID.
   c. All other fields can be left empty.
7. Click the **OK** button.

5.4 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from a file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*/../../resource/xml/RTI_REAL_TIME_CONNECT.xml**. You can specify a different configuration file using the command line option, **-cfgFile**.

The default file **RTI_REAL_TIME_CONNECT.xml** does not contain any valid database configuration information yet. For this example we will edit this file as follows:

1. Look for the tag **<oracle_connection>**. Replace the tags **<dsn>, <user_name>,**and **<password>** as follows:

    ```
    <oracle_connection>
           <dsn>Example</dsn>
           <user_name>Student</user_name>
           <password>mypsswrd</password>
    </oracle_connection>
    ```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "Example" DSN.

5.5 Starting the Real-Time Connect to Oracle Daemon

You can start the *Real-Time Connect* Daemon as a Windows service (assuming that you allowed the installation program to do so (the default case)). However, for the example in this chapter, we will start the daemon manually.

Start *Real-Time Connect* by executing the following command from **bin\i86Win32** (for 32-bit Windows systems) or **bin\x64Win64VS2010** (for 64-bit Windows systems) in the installation directory.

❏ For Windows XP Professional and Windows 2000 systems:

```
rtirtc_oracle -noDaemon -cfgName default
```

❏ For Windows 2003, Windows Server 2008, Windows Vista, and Windows 7[1] systems:

```
rtirtc_oracle -noDaemon -cfgName default -dbtransport 1
```

By default, *Real-Time Connect* to Oracle runs in the background as Windows service. Specifying the **-noDaemon** option prevents that, and the *Real-Time Connect* Daemon is started as a regular process. Messages are sent to standard output.

You should see output similar to the following, indicating that the process is running.

```
> rtirtc_oracle -noDaemon -cfgName default
RTI Real-Time Connect to Oracle, Release 5.0.x: startup succeeded
```

Real-Time Connect to Oracle is now connected to the "**Example**" data source.

5.6 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in an Oracle 11g database. The first step is to create an IDL type definition. By using the "**-example**" option of *rtiddsgen*[1] you will automatically create a Publisher of this IDL type.

1. Create a new file called "**MyType.idl**" with the following contents:

   ```
   struct MyType {
       short pkey; //@key
       char message[13];
   };
   ```

   This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the "**pkey**" field.

2. Now execute the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example i86Win32VS2005[2] MyType.idl
   ```

   This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

1. On Windows 2003, Windows Vista and Windows 7 systems: If you run the *Real-Time Connect* Daemon with the **-noDaemon** option and the Oracle server runs as a Windows service, *Real-Time Connect* will not be able to communicate using shared memory. To use shared memory, run both *Real-Time Connect* and the Oracle server as services, or run both of them from the command line.

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *RTI Core Libraries and Utilities User's Manual* (Chapter 3) for more information about how to run *rtiddsgen*.

2. If you are using a different supported compiler, you will need to use a different value here, such as i86Win32VS2008 for Visual Studio 2008. See the *RTI Core Libraries and Utilities Release Notes*.

3. The generated example will also have a Visual Studio Solution file called **MyType-vs2005.sln**. Start Microsoft Visual Studio 2005 and load this solution by clicking on this file.
4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

   Insert the following lines immediately below this comment:

   ```
   instance->pkey = count;
   strcpy(instance->message, "Hello world!");
   ```

5. Save your changes and build the **MyType_publisher** project in Visual Studio.
6. Start the **MyType_publisher** application so that it starts publishing data samples. From a command prompt,

   ```
   > objs\i86Win32VS2005\MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the Oracle database yet. You still need to set up a subscription in *Real-Time Connect* to Oracle.

   Subscriptions are set up in the "**RTIDDS_SUBSCRIPTIONS**" configuration table that *Real-Time Connect* to Oracle created when it connected to the Oracle database. For inserting entries into this table you can use the Oracle's **SQL*Plus** utility again.
7. Start the **SQL*Plus** utility from the command prompt:[1]

   ```
   > sqlplus student/mypsswrd@orcl
   SQL> set autocommit on;[2]
   ```

8. You can see that the "**RTIDDS_SUBSCRIPTIONS**" configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon '**;**' at the end of the line*:

   ```
   SQL> select * from RTIDDS_SUBSCRIPTIONS;
   ```

1. If *sqlplus* is not on your path, it can be found in the **bin\** directory of your Oracle installation.
2. By default, *sqlplus* is not in autocommit mode. Use this statement to turn on autocommit.

```
0 rows found.
```

9. To store the samples from the **MyType_publisher** application in a table named "**Example**", you insert a corresponding entry into the "**RTIDDS_SUBSCRIPTIONS**" table:

   ```
   SQL> insert into RTIDDS_SUBSCRIPTIONS (table_owner, table_name,
   domain_id, topic_name, type_name) values
   ('Student', 'Example', 0, 'Example MyType', 'MyType');
   1 row inserted.
   ```

   This entry directs *Real-Time Connect* to Oracle to create a user table named "**Example**" with owner "**Student**" and to start storing samples published with topic "**Example MyType**" and data type "**MyType**" in Domain **0**.

10. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

    ```
    SQL> select * from Example;
    ```

    The output will look something like this:

    ```
    PKEY MESSAGE       RTIDDS_DOMAIN_ID RTIRTC_REMOTE
    ---------- ------------- ---------------- -------------
            90 Hello world!                44             1
            91 Hello world!                44             1
            92 Hello world!                44             1
    ...
    ```

    The actual number of rows found depends on when exactly the *Real-Time Connect* Daemon started storing samples. If you execute the "**select**" statement repeatedly you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

5.7 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to Oracle to automatically publish changes made to a table in a data source. For this, you will again use the Oracle's **SQL*Plus** utility to change a record in the "**Example**" table. We assume that you have followed the instructions in Section 5.5 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 to 4 in Section 5.6 in creating the IDL file and generating and compiling the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing "**Ctrl-c**" in the window where it is running.
2. Build the **MyType_subscriber** project in Visual Studio and start the application from the command line:

   ```
   > objs\i86Win32VS2005\MyType_subscriber
   MyType subscriber sleeping for 4 sec...
   MyType subscriber sleeping for 4 sec...
   ...
   ```

   You will get this message repeatedly, since nothing is being published.
3. For publishing changes to a data source, you need to configure the "**RTIDDS_PUBLICATIONS**" table. For publishing the changes to the "**Example**" table, execute the following SQL statement:

   ```
   SQL> insert into RTIDDS_PUBLICATIONS (table_owner, table_name,
   domain_id, topic_name, type_name) values
   ('Student', 'Example', 0, 'Example MyType', 'MyType');
   ```

   This entry directs *Real-Time Connect* to Oracle to start publishing changes to table "**Example**" of owner "**Student**" as the topic "**Example MyType**" with type "**MyType**" in Domain **0**.
4. Now change one of the previously captured samples in the "**Example**" table, for instance the last one:

   ```
   SQL> update Example set message = 'Hello again!' where pkey in
   (select max(pkey) from Example);
   ```

   You will see that the **MyType_subscriber** application reports the update, for example:

   ```
   ...
   MyType subscriber sleeping for 4 sec...
   pkey: 748
      message:
         message[ 0]: 'H'
         message[ 1]: 'e'
         message[ 2]: 'l'
         message[ 3]: 'l'
         message[ 4]: 'o'
         message[ 5]: ' '
         message[ 6]: 'a'
   ```

```
          message[ 7]: 'g'
          message[ 8]: 'a'
          message[ 9]: 'i'
          message[10]: 'n'
          message[11]: '!'
          message[12]: <0>
    MyType subscriber sleeping for 4 sec...
    ...
```

5. You can also update all entries in the table:

   ```
   SQL> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

Chapter 6 Real-Time Connect to TimesTen with Oracle In-Memory Database Cache

This chapter describes how to install and run *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache (IMDB).

For detailed information about Oracle In-Memory Database Cache, see the *Oracle In-Memory Database Cache User's Guide.*

This chapter assumes that you are familiar with IMDB and that you have already installed *Real-Time Connect* to Oracle TimesTen and run through the exercises in Chapters 2 or 3. If you have not yet done this, please work through Chapter 2 or 3 first.

6.1 Installation

Oracle In-Memory Database Cache requires the installation of Oracle 11g and Oracle TimesTen 11.2.1.

The installation of Oracle 11g is beyond the scope of this document. Please refer to Oracle documentation for the process to install and configure Oracle 11g.

For instructions on how to install Oracle TimesTen11.2.1, refer to Section 2.1 (UNIX) and Section 3.1 (Windows) in this document.

If you did not configure the environment variable TNS_ADMIN during the Oracle TimesTen installation process, you can do it now using the **ttmodinstall** utility with the

-tns_admin option. See the Oracle TimesTen In-Memory Database reference for more details.

```
> ttmodinstall -tns_admin /opt/oracle/product/11.2.0/dbhome_1/
network/admin
```

The option **-tns_admin** is used to specify the location of the **tnsnames.ora** file in your Oracle 11g installation.

6.2 Creating Oracle Accounts

Before you can use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache, you need to create three Oracle users:

- ❑ A 'Student' user that owns the Oracle tables that will be cached in TimesTen. If you have not created the user yet, Section 4.2 (UNIX) and Section 5.2 (Windows) explain how to do it.
- ❑ A 'timesten' user that owns the Oracle tables used to store information about IMDB.
- ❑ A cache administration user 'cacheuser' that owns and maintains Oracle objects that store information used to managed IMDB.

To create the '**timesten**' and '**cacheuser**' users, enter the following. We assume that the database is identified by the service name '**orcl11g**' and the sys password is '**mypsswrd**'.

```
> cd <Oracle TimesTen install dir>/oraclescripts
> sqlplus[1] sys/mypsswrd@orcl11g as sysdba
SQL> CREATE TABLESPACE cachetblsp DATAFILE 'datfttuser.dbf' SIZE
100M;
SQL> @initCacheGlobalSchema "cachetblsp"
SQL> CREATE USER cacheuser IDENTIFIED BY oracle DEFAULT
TABLESPACE cachetblsp QUOTA UNLIMITED ON cachetblsp;
SQL> @grantCacheAdminPrivileges "cacheuser"
SQL> exit
```

1. If *sqlplus* is not on your path, it can be found in the **bin/** directory of ORACLE_HOME.

6.3 Creating an Oracle TimesTen Database

Before you can use *Real-Time Connect* to Oracle TimesTen with Oracle In-Memory Database Cache, you need to create or gain access to a data store (database) with the same character set as the Oracle database. Section 2.2 (UNIX) and Section 3.2 (Windows) explain how to create a data store with Oracle TimesTen when you act as your own database administrator.

6.4 Creating Oracle TimesTen Accounts

In addition to the creation of Oracle users, integration with IMDB requires the creation of two Oracle TimesTen users:

❑ A 'Student' user that owns the cached tables in Oracle TimesTen. If you have not created the user yet, Section 2.3 (UNIX) and Section 3.2.1 (Windows) explain how to do it.

❑ A cache manager user 'cacheuser' that performs the cache group operations.

To create and configure the '**cacheuser**' user with password '**timesten'** log in as the Oracle TimesTen administrator (the user that installed TimesTen) and follow these steps:

1. Connect to the Oracle TimesTen database created in Section 6.3:

```
> ttIsql -connStr "DSN=Example"
```

2. Enter the following to create the user account:

```
Command> create user cacheuser identified by timesten;
Command> GRANT CREATE SESSION, CACHE_MANAGER, CREATE ANY TABLE
TO cacheuser;
Command> call ttCacheUidPwdSet('cacheuser','oracle');
Command> exit
```

The call to the **ttCacheUidPwdSet** built-in procedure sets the Oracle cache administration user name and password.

3. As the new cache manager user, create a cache grid:

```
> ttIsql "DSN=Example;UID=cacheuser;PWD=timesten;OraclePWD=ora-
cle"
```

```
Command> call ttGridCreate('myGrid');
Command> call ttGridNameSet('myGrid');
Command> exit
```

6.5 Creating a Data Source for Oracle In-Memory Database Cache

Real-Time Connect to Oracle TimesTen with Oracle In-Memory Database Cache will use the TimesTen ODBC driver to access data sources (data stores). If you have not done it yet, Section 2.4 (UNIX) and Section 3.4 (Windows) explain how to create a data source for Oracle TimesTen that also works with IMDB.

6.6 Starting Real-Time Connect to Oracle TimesTen

Before you start *Real-Time Connect* to Oracle TimesTen, verify that your configuration file, **<RTIRTCHOME>/resource/xml/RTI_REAL_TIME_CONNECT.xml**, has the following contents:

```
<timesten_connection>
  <dsn>Example</dsn>
  <user_name>Student</user_name>
</timesten_connection>
```

Next, start *Real-Time Connect* to Oracle TimesTen from the command prompt.

```
>./rtirtc_timesten -noDaemon -cfgName default
RTI Real-Time Connect to Oracle, Release 5.0.x: startup succeeded
```

Real-Time Connect to Oracle is now connected to the “**Example**” data source.

6.7 Creating an Oracle Table

Connect to the "**Student**" Oracle account and create a table that will be cached by Oracle TimesTen:

```
> sqlplus student/mypsswrd@orcl
SQL> set autocommit on;[1]
SQL> drop table Example;[2]
SQL> create table Example (pkey NUMBER(5), message CHAR(13), pri-
mary key(pkey));
```

Grant SELECT, INSERT, UPDATE and DELETE privileges on the Example table to the cache administration user:

```
SQL> GRANT SELECT ON Example TO cacheuser;
SQL> GRANT INSERT ON Example TO cacheuser;
SQL> GRANT UPDATE ON Example TO cacheuser;
SQL> GRANT DELETE ON Example TO cacheuser;
SQL> quit;
```

6.8 Creating a Cache Group and Start the Replication Agent

Use the *ttIsql* utility to connect to the "**Example**" data store. At the command prompt, call the **ttCacheStart** procedure to start the cache agent for the data store:

```
> ttIsql -connStr "DSN=Example"
Command> call ttCacheStart();
```

Next, use the CREATE CACHE GROUP statement to create an asynchronous writethrough cache group named **ExampleCache**, to cache the contents of the Example Oracle table on TimesTen:

```
Command> CREATE ASYNCHRONOUS WRITETHROUGH CACHE GROUP ExampleCache
FROM EXAMPLE (pkey TT_SMALLINT, message CHAR(13), primary
key(pkey));
```

1. By default, *sqlplus* is not in autocommit mode. Use this statement to turn on autocommit.

2. You only need to execute this step if the table "Example" already exists because you worked through the exercises in Chapters 4 or 5.

The cache group specification must include a TimesTen definition of the table(s) to be cached in Oracle TimesTen. The mapping between Oracle types and TimesTen types in the FROM clause is described in Chapter 5 of the *Real-Time Connect User's Manual*.

Start the TimesTen replication agent.

```
Command> call ttRepStart();
Command> quit;
```

6.9 Storing Samples from a Publisher into the TimesTen Table

Using *ttisql*, insert the following entry into the "**RTIDDS_SUBSCRIPTIONS"** table in the TimesTen data store to capture the samples from the **MyType_publisher** application. This is the same application that you created and compiled in either Section 2.7 or Section 3.7.

```
> ttIsql -connStr "DSN=Example"
Command> insert into RTIDDS_SUBSCRIPTIONS values
('Student', 'Example', 0, 'Example MyType', 'MyType');
Command> quit;
```

Start the **MyType_publisher** application from the command prompt.

```
> objs/<arch>/MyType_publisher
Writing MyType, count 0
Writing MyType, count 1
Writing MyType, count 2
```

The published samples are now being stored in the TimesTen "**Example**" table. These changes are automatically being synchronized to the Oracle "**Example**" table by Cache Connect to Oracle as configured by the cache group "**ExampleCache**."

Check that this is happening by examining the contents of the Oracle table using **SQL*Plus**:

```
> sqlplus student/mypsswrd@orcl
SQL> set autocommit on;
SQL> select * from Example;
```

The output will look something like this:

```
PKEY MESSAGE       RTIDDS_DOMAIN_ID RTIRTC_REMOTE
---------- ------------- ---------------- -------------
        90 Hello world!                44             1
        91 Hello world!                44            1
        92 Hello world!                44            1
```

The actual number of rows found depends on when the *Real-Time Connect* Daemon started storing samples. If you execute the “**select**” statement repeatedly, you will see the number of rows grow as new values are published by **MyType_publisher** every 4 seconds and the changes are stored into TimesTen by the *Real-Time Connect* Daemon, then replicated to Oracle with Oracle In-Memory Database Cache.

This example shows how *Real-Time Connect* can be used in a system where real-time applications produce data at high rates that need to be stored in a persistent database. In this solution, *Real-Time Connect* to Oracle uses the Oracle TimesTen In-Memory database as a high-performance data cache and Oracle In-Memory Database Cache to record the data permanently in an Oracle database.

6.10 Shutting Down

When you are finished with this example, you should shut down Oracle with Oracle In-Memory Database Cache (or else it will continue to run in the background).

1. Stop the replication agent:

```
> ttisql -connStr "DSN=Example”
Command> call ttRepStop();
```

2. Drop the cache group:

```
Command> DROP CACHE GROUP ExampleCache;
```

3. Stop the cache agent:

```
Command> call ttCacheStop();
Command> quit;
```

Chapter 7 Real-Time Connect to MySQL on UNIX-based Systems

This chapter provides instructions on how to install *Real-Time Connect* to MySQL on the following platforms:

❑ Red Hat® Enterprise Linux® 4.0 and 5.0 on Pentium®-class processors

7.1 Installation

First, verify that MySQL 5.1 is installed and running on your system.

The installation of MySQL is beyond the scope of this document. Please refer to the MySQL Reference Manual for the process to install and configure MySQL.

Real-Time Connect requires the installation of the *MySQL ODBC 5.1.6* (or higher) driver. The driver is not bundled with the MySQL server and must be installed separately.

The ODBC connector can be downloaded from http://dev.mysql.com/downloads/connector/odbc/5.1.html.

The installation guide can be found in http://dev.mysql.com/doc/refman/5.1/en/connector-odbc-installation.html.

The MySQL ODBC driver requires an ODBC driver manager. We recommend using *UnixODBC 2.2.12* (or higher), a complete, free/open ODBC solution for UNIX-based systems. The driver manager can be downloaded from http://www.unixodbc.org.

UnixODBC Note: *Real-Time Connect* links to the UnixODBC library **libodbc.so.1**. In release 2.3.1, UnixODBC changed the library version from 1 to 2. If after installing Unix-

ODBC, *Real-Time Connect* cannot find **libodbc.so**, create a symlink to **libodbc.so.1** from **libodbc.so.2**:

```
> ln -s libodbc.so.2 libodbc.so.1
```

To install Real-Time Connect from the distribution file:

1. Unpack the distribution file to a temporary location using GNU™'s version of tar, gtar. There are different distribution files depending on if you are running on a 32-bit or 64-bit version of the operating system.

   - **On Linux systems** (your filename will be slightly different, depending on the version number/letter):

     ```
     > tar zxfv RTI_Real-Time_Connect_5.0.x.linux32bundle.tar.gz
     ```

     or

     ```
     > tar zxfv RTI_Real-Time_Connect_5.0.x.linux64bundle.tar.gz
     ```

   This will extract the compressed *Real-Time Connect* file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 6.0.8 distribution file. When using *Real-Time Connect* to interface with MySQL only, there is no need to have Oracle TimesTen installed.

2. Unpack the compressed *Real-Time Connect* installation file in a location of your choice:

   - **On Linux systems** (your filename will be slightly different, depending on the version number/letter):

     ```
     > tar zpxfv RTI_Real-Time_Connect_5.0.x.linux32.tar.gz
     ```

     or

     ```
     > tar zpxfv RTI_Real-Time_Connect_5.0.x.linux64.tar.gz
     ```

   This process will result in four directories, **/bin**, **/docs**, **/lib** and **/resource**, which contain the *Real-Time Connect* to Oracle executables, user documentation, libraries and resources, respectively.

3. Add the path to the UnixODBC driver manager to the beginning of **LD_LIBRARY_PATH**. For example:

   ```
   > setenv LD_LIBRARY_PATH /usr/lib:$LD_LIBRARY_PATH
   ```

   or

   ```
   > export LD_LIBRARY_PATH=/usr/lib:$LD_LIBRARY_PATH
   ```

   Replace **/usr/lib** with the location of the UnixODBC driver manager in your system.

4. Please read Chapter 9: License Installation.

7.2 Configuring the MySQL Server

Real-Time Connect Daemon uses user defined functions, UDFs, to interface with the MySQL server. Or more accurately, *Real-Time Connect* provides user defined functions in a library, **librtirtc_mysqlq.so** (under lib/<arch> directory), that the MySQL server must be able to load while running to communicate with the *Real-Time Connect* Daemon.

This section provides instructions to install **librtirtc_mysqlq.so**.

7.2.1 Install librtirtc_mysqlq.so

Copy the appropriate version of **librtirtc_mysqlq.so** into the MySQL server's plugin directory (the directory named by the **plugin_dir** system variable).

To check the current location of the plugin directory, login to MySQL and execute the following statement:

```
> mysql -uroot (you can log in as any user)
mysql> show variables like 'plugin_dir'
+---------------+--------------------------------------+
| Variable_name | Value                                |
+---------------+--------------------------------------+
| plugin_dir    | /opt/mysql/product/5.1.44/lib/plugin |
+---------------+--------------------------------------+
1 row in set (0.14 sec)
```

The plugin directory can be changed by setting the value of **plugin_dir** when the MySQL server is started. For example, you can set its value in the **my.cnf** configuration file:

```
[mysqld]
plugin_dir=/path/to/plugin/directory
```

For additional information about the plugin directory see the following link:
http://dev.mysql.com/doc/refman/5.1/en/install-plugin.html

7.2.2 Install libnddsc.so and libnddscore.so

This section only applies to Linux systems.

librtirtc_mysqlq.so uses *Connext* and thus the shared libraries **libnddsc.so** and **libnddscore.so** must also be installed and be accessible at runtime by the MySQL server.

Add the directory containing the appropriate *Connext* libraries to the environment variable, **LD_LIBRARY_PATH**, for the user who starts the MySQL server. You may need to restart the MySQL server after this variable has been changed.

7.3 Creating a MySQL Account

Before you can use *Real-Time Connect*, you need to obtain a MySQL user account from your database administrator. If you are acting as your own database administrator, start **mysql** from the command prompt to connect to the MySQL server as the MySQL **root** user:

```
> mysql -uroot
```

If you have assigned a password to the **root** account, you will also need to provide a **-p** option.

For example, to create a new MySQL account with a user name of “**Student**” and a password of “**mypsswrd**”, enter the following:

```
> mysql -uroot
mysql> GRANT ALL PRIVILEGES ON *.* TO ‘Student’@’localhost’ IDENTI-
FIED BY ‘mypsswrd’ WITH GRANT OPTION;
```

The remaining sections in this chapter assume that a MySQL user named "**Student**" with the password "**mypsswrd**" has an account on the local host.

7.4 Creating a Data Source for MySQL

Real-Time Connect uses the MySQL ODBC driver to access data sources. Usually these are the same data sources to which your applications connect. The connection information for each data source is stored in the “**.odbc.ini**” file. The stored information describes each data source in detail, specifying the driver name, a description, and any additional information the driver needs to connect to the data source.

To create the “**.odbc.ini**” file, follow these steps:

1. Create a new file named "**.odbc.ini**" in your home directory using your favorite text editor. Alternatively, you can use the **ODBCINI** environment variable to specify the file location.
2. Insert these lines in the file:

   ```
   [ODBC Data Source]
   Example=MySQL Driver

   [Example]
   DRIVER=/usr/lib/libmyodbc5.so
   Database=test
   ```

   Notes:

   - Make sure that **DRIVER** points to the valid location of the MySQL ODBC driver on your system.
   - When connecting to a MySQL server located on the local system, the mysql client connects through a local file called a socket instead of connecting to the localhost loopback address 127.0.0.1. For the mysql client, the default location of this socket file is **/tmp/mysql.sock**. However, many MySQL installations place this socket file somewhere else, such as **/var/lib/mysql/mysql.sock**. You may see this error message when you start the daemon:

     ```
     [unixODBC][MySQL][ODBC 3.51 Driver]Can't connect to
     local MySQL server through socket '/tmp/mysql.sock'
     ```

     To correct this error, specify the right socket file by adding the SOCKET attribute to the DSN entry. For example:

     ```
     [Example]
     DRIVER=/usr/lib/libmyodbc3.so
     SOCKET=/var/lib/mysql/mysql.sock
     Database=test
     ```

3. Save your changes.

The "Example" data source uses the "test" database which is usually available as a workspace for users to try things out.

7.5 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from a file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*/../../resource/xml/RTI_REAL_TIME_CONNECT.xml**. You can specify a different file with the command-line option, **-cfgFile**.

The default file, **RTI_REAL_TIME_CONNECT.xml**, does not contain any actual valid information yet. For this example we will edit this file as follows:

1. Look for the tag **<mysql_connection>**. Replace the tags **<dsn>, <user_name>,** and **<password>** as follows:

```
 <mysql_connection>
   <dsn>Example</dsn>
   <user_name>Student</user_name>
   <password>mypsswrd</password>
</mysql_connection>
```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "Example" DSN.

7.6 Running the MySQL Server in ANSI_QUOTES Mode

Real-Time Connect requires the MySQL server to be configured in ANSI_QUOTES mode. Under that configuration, the MySQL server treats '"' as an identifier quote character and not as a string quote character.

To verify if the MySQL server is already configured in ANSI_QUOTES mode, run the following SQL statement from the **mysql** command prompt:

```
mysql> SELECT @@global.sql_mode;
```

If the string 'ANSI_QUOTES' is not part of the result, the MySQL server needs to be configured in ANSI_QUOTES mode by executing the following SQL statement:

```
mysql> SET GLOBAL sql_mode = 'ANSI_QUOTES'
```

7.7 Starting the Real-Time Connect Daemon

Start *Real-Time Connect* by executing the following command from **bin/<*arch*>/** in the installation directory. **<*arch*>** is the directory containing the executables compiled for your host platform. For example, i86Linux2.6gcc3.4.3 or x64Linux2.6gcc3.4.5.

```
> ./rtirtc_mysql -noDaemon -cfgName default
```

By default, *Real-Time Connect* runs in the background as a daemon process. Specifying the "**-noDaemon**" option prevents that, and starts up the *Real-Time Connect* Daemon as a regular process. Messages are sent to standard output.

You should see the following output, indicating that the process is running.

```
>./rtirtc_mysql -noDaemon -cfgName default
RTI Real-Time Connect to MySQL, Release 5.0.x: startup succeeded
```

Real-Time Connect is now connected to the "**Example**" data source.

Note: Make sure that you have the UnixODBC driver manager library, **libodbc.so**, on your **LD_LIBRARY_PATH (**see Step 3 in Section 7.1).

7.8 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in MySQL database. The first step is to create an IDL type definition. By using the "**-example**" option of *rtiddsgen*[1] you will automatically create a Publisher of this IDL type.

1. Create a new text file called "**MyType.idl**" with the following contents:

   ```
   struct MyType {
       short pkey; //@key
       char message[13];
   };
   ```

   This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the "**pkey**" field.

2. Now execute the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example <arch> MyType.idl
   ```

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *Connext* documentation for more information about how to run *rtiddsgen*.

For example:

```
 > rtiddsgen -language C -example i86Linux2.6gcc4.1.1 MyType.idl
```

This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

3. The generated example will also have a makefile named:

   ```
   makefile_MyType_<arch>
   ```

   You may need to edit the makefile to specify the location of the compiler if it is not available on your path.

4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

   Insert the following lines immediately below this comment:

   ```
   instance->pkey = count;
   strcpy(instance->message, "Hello world!");
   ```

5. Save your changes and build the **MyType_publisher** and **MyType_subscriber** applications by executing:

   ```
   > gmake -f makefile_MyType_<arch>
   ```

6. Start the **MyType_publisher** application so that it starts publishing data samples.

   ```
   > objs/<arch>/MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the MySQL database yet. For this you need to set up a subscription in *Real-Time Connect*.

   Subscriptions are set up in the “**RTIDDS_SUBSCRIPTIONS**” configuration table that *Real-Time Connect* created when it connected to the MySQL database.

   Start **mysql** from the command prompt:

   ```
   > mysql -uStudent -pmypsswrd test
   mysql>
   ```

7. You can see that the "**RTIDDS_SUBSCRIPTIONS**" configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon '**;**' at the end of the line*:

```
mysql> select * from RTIDDS_SUBSCRIPTIONS;
Empty set (0.01 sec)
```

8. To store the samples from the **MyType_publisher** application in a table named "**Example**", insert a corresponding entry into the "**RTIDDS_SUBSCRIPTIONS**" table:

```
mysql> insert into RTIDDS_SUBSCRIPTIONS (table_owner,
 table_name, domain_id, topic_name, type_name) values
('test', 'Example', 0, 'Example MyType', 'MyType');
1 row inserted.
```

This entry directs *Real-Time Connect* to create a user table named "**Example**" and to start storing samples published with topic "**Example MyType**" and data type "**MyType**" in Domain **0**.

Note: The **table_owner** in MySQL is the database.

9. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

```
mysql> select * from Example;
```

The output will look something like this:

```
+------+--------------+------------------+---------------+------------+
| pkey | message      | RTIDDS_DOMAIN_ID | RTIRTC_REMOTE | RTIRTC_SCN |
+------+--------------+------------------+---------------+------------+
...
|  134 | Hello world! |                0 |             1 |          0 |
|  135 | Hello world! |                0 |             1 |          0 |
|  136 | Hello world! |                0 |             1 |          0 |
+------+--------------+------------------+---------------+------------+
85 rows in set (0.00 sec)
```

The actual number of rows found depends on exactly when the *Real-Time Connect* Daemon started storing samples. If you execute the "**select**" statement repeatedly, you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

7.9 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to automatically publish changes made to a table in a data source.

We assume that you have followed the instructions in Section 7.7 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 to 4 in Section 7.8 to create the IDL file and generate and compile the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing "**Ctrl-c**" in the window where it is running.
2. Start the **MyType_subscriber** application from the command line:

   ```
   > objs/<arch>/MyType_subscriber
   MyType subscriber sleeping for 4 sec...
   MyType subscriber sleeping for 4 sec...
   ...
   ```

   You will get this message repeatedly, since nothing is being published.
3. For publishing changes to a data source, you need to configure the "**RTIDDS_PUBLICATIONS**" table. For publishing the changes to the "**Example**" table, execute the following SQL statement:

   ```
   mysql> insert into RTIDDS_PUBLICATIONS (table_owner, table_name,
    domain_id, topic_name, type_name) values
   ('test', 'Example', 0, 'Example MyType', 'MyType');
   ```

   This entry directs *Real-Time Connect* to start publishing changes to table "**Example**" as the topic "**Example MyType**" with type "**MyType**" in Domain **0**.
4. Now change one of the previously captured samples in the "**Example**" table, such as the last one:

   ```
   mysql> update Example set message = 'Hello again!' where pkey =
    (select * from (select max(pkey) from Example) as _max);
   ```

   You will see that the **MyType_subscriber** application reports the update, for example:

   ```
   MyType subscriber sleeping for 4 sec...
   pkey: 748
      message:
         message[ 0]: 'H'
         message[ 1]: 'e'
         message[ 2]: 'l'
   ```

```
      message[ 3]: 'l'
      message[ 4]: 'o'
      message[ 5]: ' '
      message[ 6]: 'a'
      message[ 7]: 'g'
      message[ 8]: 'a'
      message[ 9]: 'i'
      message[10]: 'n'
      message[11]: '!'
      message[12]: <0>
MyType subscriber sleeping for 4 sec...
...
```

5. You can also update all entries in the table:

   ```
   mysql> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

Chapter 8 Real-Time Connect to MySQL on Windows Systems

This chapter provides instructions on how to install *Real-Time Connect* to MySQL on Windows platforms.

This chapter assumes you are using Microsoft Visual Studio 2005[1].

8.1 Installation

Before installing *Real-Time Connect*, verify that MySQL 5.1.44 or higher is installed and running on your system. The installation of MySQL is beyond the scope of this document. Please refer to the MySQL Reference Manual for the process to install and configure MySQL.

Real-Time Connect requires the installation of the *MySQL ODBC* 5.1.6 driver or higher. The driver is not bundled with the MySQL server and must be installed separately.

The ODBC connector can be downloaded from:
http://dev.mysql.com/downloads/connector/odbc/5.1.html.

The installation guide can be found here:
http://dev.mysql.com/doc/refman/5.1/en/connector-odbc-installation.html.

1. You may use other supported Microsoft compilers such as Visual Studio 2010, Visual Studio 2008, or Visual Studio .Net 2003, however, the instructions in this chapter were written assuming Visual Studio 2005.

Install *Real-Time Connect* from the distribution file:

1. Unzip the distribution file **RTI_Real-Time_Connect_5.0.x.win32bundle.zip** for 32-bit Windows systems or **RTI_Real-Time_Connect_5.0.x.win64bundle.zip** for 64-bit Windows systems. (Your filename will be slightly different, depending on the version number/letter.)

   This will extract the *Real-Time Connect* **rtirtc_setup.exe** file, the *Getting Started Guide* (this file), and the Oracle TimesTen In-Memory Database Release 11.2.1 distribution file. When using *Real-Time Connect* to interface with MySQL only, you do not need to have Oracle TimesTen installed.
2. Install *Real-Time Connect* by running the **rtirtc_setup.exe** file. Follow the on-screen instructions to complete the installation.
3. Please read Chapter 9: License Installation.

8.2 Configuring the MySQL Server

Real-Time Connect Daemon uses user-defined functions, UDFs, to interface with the MySQL server. Or more accurately, *Real-Time Connect* provides UDFs in a library, **rtirtc_mysqlq.dll** (in the **lib\\<*architecture*>** directory), that the MySQL server must be able to load while running to communicate with the *Real-Time Connect* Daemon.

This section provides instructions on how to install **rtirtc_mysqlq.dll**.

8.2.1 Installing rtirtc_mysqlq.dll

Copy **rtirtc_mysqlq.dll** into the MySQL server's plugin directory (the directory named by the **plugin_dir** system variable). To check the current location of the plugin directory, login to MySQL and execute the following statement:

```
> mysql -uroot (you can log in as any user)
mysql> show variables like 'plugin_dir'
+---------------+--------------------------------------------------+
| Variable_name | Value                                            |
+---------------+--------------------------------------------------+
| plugin_dir    | C:\Program Files\MySQL\MySQL Server 5.1\lib/plugin
+---------------+--------------------------------------------------+
1 row in set (0.03 sec)
```

The plugin directory can be changed by setting the value of **plugin_dir** when the MySQL server is started. For example, you can set its value in the **my.cnf** configuration file:

```
[mysqld]
plugin_dir=/path/to/plugin/directory
```

For additional information about the plugin directory, see the following link: http://dev.mysql.com/doc/refman/5.1/en/install-plugin.html.

8.2.2 Installing nddsc.dll and nddscore.dll

Because **rtirtc_mysqlq.dll** uses *Connext*, the dynamic libraries **nddsc.dll** and **nddscore.dll** must also be installed and accessible at runtime by the MySQL server.

Make sure your **Path** system environment variable[1] contains the path to **nddsc.dll** and **nddscore.dll** (in **<NDDSHOME>\lib\<*architecture*>**).

If **<NDDSHOME>\lib\<*architecture*>** is not already in you **Path**, you will need to restart your computer after you modify the **Path**. If you do not want to restart your computer, you can copy the libraries into a directory that is already in the system **Path**, such as **c:\Windows\System32**.

8.3 Creating a MySQL Account

Before you can use *Real-Time Connect*, you need to obtain a MySQL user account from your database administrator. If you are acting as your own database administrator, start **mysql** from the command prompt to connect to the MySQL server as the MySQL **root** user:

```
> mysql -uroot
```

If you have assigned a password to the **root** account, you will also need to provide a **-p** option.

For example, to create a new MySQL account with a user name of "**Student**" and a password of "**mypsswrd**", enter the following:

1. To view and/or edit the **Path** system environment variable, from the **Start** button/menu, select **Settings, Control Panel, System, Advanced tab,** then select the **Environment Variable** button. View or edit the **Path** in the System Variables. (The exact steps for accessing environment variables may vary, depending on your version of the Windows operating system).

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'Student'@'localhost'
       IDENTIFIED BY 'mypsswrd' WITH GRANT OPTION;
```

The remaining sections in this chapter assume that a MySQL user named "**Student**" with the password "**mypsswrd**" has an account on the local host.

8.4 Creating a Data Source for MySQL

Real-Time Connect uses the MySQL ODBC driver to access data sources. Usually these are the same data sources to which your applications connect. The connection information for each data source is stored in the Windows registry. The stored information describes each data source in detail, specifying the driver name, a description, and any additional information the driver needs to connect to the data source.

To add a data source, follow these steps:

1. Open the ODBC Data Source Administrator:
   - On Windows XP and Windows Server 2003 systems: Select **Start, Control Panel, Administrative Tools, Data Sources (ODBC)**.
   - On Windows Vista systems: Select **Start, Control Panel, System and Maintenance, Administrative Tools, Data Sources (ODBC)**.
   - On Windows 7 and Windows Server 2008 systems: Select **Start, Control Panel, System and Security, Administrative Tools, Data Sources (ODBC).**
2. Select the **User DSN** tab.
3. Click the **Add** button; the **Create New Data Source** dialog appears.
4. Select the **MySQL** driver from the list of drivers.
5. Click the **Finish** button; the MySQL ODBC Driver Configuration dialog appears.
6. Fill out the fields in the dialog.
   a. Enter “**Example**” as the **Data Source Name** (DSN).
   b. Enter "**Student**" as the **User** and “**mypsswrd**” as the **Password**.
   c. Select “**test**” as the **Database**.
   d. All other fields can be left empty.
7. Click the **OK** button.

8.5 Creating a Configuration File for Real-Time Connect

Real-Time Connect reads its configuration information from a file. By default, *Real-Time Connect* tries to load the configuration file, ***<Real-Time Connect executable location>*****/../../resource/xml/RTI_REAL_TIME_CONNECT.xml**. You can specify a different file using the command line option, **-cfgFile**.

The default file **RTI_REAL_TIME_CONNECT.xml** does not contain any actual configuration information yet. For this example we will edit this file as follows:

1. Look for the tag **<mysql_connection>**. Replace the tags **<dsn>**, **<user_name>,** and **<password>** as follows:

```
 <mysql_connection>
   <dsn>Example</dsn>
   <user_name>Student</user_name>
   <password>mypsswrd</password>
</mysql_connection>
```

2. Save the file.

This configuration file instructs *Real-Time Connect* to monitor the data source as specified by the "Example" DSN.

8.6 Running the MySQL Server in ANSI_QUOTES Mode

Real-Time Connect requires the MySQL server to be configured in ANSI_QUOTES mode. Under that configuration, the MySQL server treats '"' as an identifier quote character and not as a string quote character.

To verify if the MySQL server is already configured in ANSI_QUOTES mode, run the following SQL statement from the **mysql** command prompt:

```
mysql> SELECT @@global.sql_mode;
```

If the string 'ANSI_QUOTES' is not part of the result, the MySQL server needs to be configured in ANSI_QUOTES mode by executing the following SQL statement:

```
mysql> SET GLOBAL sql_mode = 'ANSI_QUOTES';
```

8.7 Starting the Real-Time Connect Daemon

You can start the *Real-Time Connect* Daemon as a Windows service (assuming that you allowed the installation program to do so which it does by default). However, for the example in this chapter, we will start the daemon manually.

Start *Real-Time Connect* by executing the following command from **bin\i86Win32** (for 32-bit Windows systems) or **bin\x64Win64VS2010** (for 64-bit Windows systems) in the installation directory.

❑ For Windows XP Professional and Windows 2000 systems:

```
rtirtc_mysql -noDaemon -cfgName default
```

❑ For Windows 2003, Windows Server 2008, Windows Vista, and Windows 7[1] systems:

```
rtirtc_mysql -noDaemon -cfgName default -dbtransport 1
```

If you see the following error message, verify that the *Real-Time Connect* library **rtirtc_mysqlq.dll** is in your **Path**:

```
[DDSQLDaemonCNAHelper_createCNAConnection,line
761:ERROR:4096:50002] [CNA:CNAOpe n] Error initializing MySQL
Server Queue: [MySQL][ODBC 5.1.6 Driver][mysqld-
5.1.44-community-nt]Can't open shared library
'rtirtc_mysqlq.dll' (errno: 2)
```

If you see the following error message, verify that the *Connext* libraries **nddsc.dll** and **nddscore.dll** are in your **Path** and restart the MySQL database server:

```
[DDSQLDaemonCNAHelper_createCNAConnection,line
761:ERROR:4096:50002] [CNA:CNAOpen] Error initializing MySQL
Server Queue: [MySQL][ODBC 5.1.6 Driver][mysqld- 5.1.44-commu-
nity-nt]FUNCTION test.MySqlNddsQueue_initialize does not exist
```

By default, *Real-Time Connect* runs in the background as Windows service. Specifying the **-noDaemon** option prevents that, and starts up the *Real-Time Connect* Daemon as a regular process. Messages are sent to standard output.

1. On Windows 2003, Windows Vista, and Windows 7 systems, the Real-Time Connect Daemon executed using the **-noDaemon** option cannot communicate using shared memory with the MySQL server running as a service.

You should see the following output, indicating that the process is running.

```
> rtirtc_mysql -noDaemon -cfgName default
RTI Real-Time Connect to MySQL, Release 5.0.x: startup succeeded
```

Real-Time Connect is now connected to the "**Example**" data source.

8.8 Storing Samples from Publishers

In this section, you will enable automatic capturing of data samples in a MySQL database. The first step is to create an IDL type definition. By using the **-example** option of *rtiddsgen*[1] you will automatically create a Publisher of this IDL type.

1. Create a new file called "**MyType.idl**" with the following contents:

   ```
   struct MyType {
       short pkey; //@key
       char message[13];
   };
   ```

   This IDL file specifies a data type that contains a message. Each instance is uniquely identified by the "**pkey**" field.

2. Now execute the following command to compile the IDL type:

   ```
   > rtiddsgen -language C -example i86Win32VS2005[2] MyType.idl
   ```

   This generates the **MyType**, **MyTypePlugin**, and **MyTypeSupport** files, as well as the **MyType_publisher** and **MyType_subscriber** example code.

3. The generated example will also have a Visual Studio Solution file called **MyType-vs2005.sln**. Start Microsoft Visual Studio 2005 and load this workspace by clicking on this file.

4. Edit **MyType_publisher.c**, and find the line containing the comment:

   ```
   /* Modify the data to be written here */
   ```

1. *rtiddsgen* is an IDL code generator distributed with *Connext*. Please refer to the *Connext* documentation for more information about how to run *rtiddsgen*.

2. If you are using a different supported compiler, you will need to use a different value here, such as i86Win32VS2008 for Visual Studio 2008.

Insert the following lines immediately below this comment:

```
instance->pkey = count;
strcpy(instance->message, "Hello world!");
```

5. Save your changes and build the **MyType_publisher** project in Visual Studio.
6. Start the **MyType_publisher** application so that it starts publishing data samples. From a command prompt, enter:

   ```
   > objs\i86Win32VS2005\MyType_publisher
   ```

   On the screen, you will see:

   ```
   Writing MyType, count 0
   Writing MyType, count 1
   Writing MyType, count 2
   ...
   ```

   The samples are not captured in the MySQL database yet. For this you need to set up a subscription in *Real-Time Connect*.

   Subscriptions are set up in the "**RTIDDS_SUBSCRIPTIONS**" configuration table that *Real-Time Connect* created when it connected to the MySQL database.
7. Start **mysql** from the command prompt:

   ```
   > mysql -uStudent -pmypsswrd test
   mysql>
   ```

8. You can see that the "**RTIDDS_SUBSCRIPTIONS**" configuration table is still empty at this point by executing the following SQL command—*don't forget to type a semicolon '**;**' at the end of the line*:

   ```
   mysql> select * from RTIDDS_SUBSCRIPTIONS;
   Empty set (0.01 sec)
   ```

9. To store the samples from the **MyType_publisher** application in a table named "**Example**", you insert a corresponding entry into the "**RTIDDS_SUBSCRIPTIONS**" table:

   ```
   mysql> insert into RTIDDS_SUBSCRIPTIONS (table_owner,
   table_name, domain_id, topic_name, type_name) values
   ('test', 'Example', 0, 'Example MyType', 'MyType');
   1 row inserted.
   ```

This entry directs *Real-Time Connect* to create a user table named “**Example**” and to start storing samples published with topic “**Example MyType**” and data type “**MyType**” in Domain **0**.

Note: The **table_owner** in MySQL is the database.

10. If the **MyType_publisher** application is still running, you can execute the following SQL statement to view the contents of the table—otherwise, restart **MyType_publisher** as described above before executing this statement.

```
mysql> select * from Example;
```

The output will look something like this:

```
+------+--------------+------------------+---------------+------------+
| pkey | message      | RTIDDS_DOMAIN_ID | RTIRTC_REMOTE | RTIRTC_SCN |
+------+--------------+------------------+---------------+------------+
...
|  134 | Hello world! |                0 |             1 |          0 |
|  135 | Hello world! |                0 |             1 |          0 |
|  136 | Hello world! |                0 |             1 |          0 |
+------+--------------+------------------+---------------+------------+
85 rows in set (0.00 sec)
```

The actual number of rows depends on when exactly the *Real-Time Connect* Daemon started storing samples. If you execute the “**select**” statement repeatedly you will see the number of rows grow. This is because the **MyType_publisher** application writes a new instance every 4 seconds.

8.9 Publishing Database Updates to Subscribers

In this section, you will enable *Real-Time Connect* to automatically publish changes made to a table in a data source.

We assume that you have followed the instructions in Section 8.7 to start a *Real-Time Connect* Daemon. In addition, we assume that you have followed the instructions in Steps 1 to 4 in Section 8.8 in creating the IDL file and generating and compiling the example code.

1. If you have not already done so, stop the **MyType_publisher** application by pressing “**Ctrl-c**” in the window where it is running.
2. Build the **MyType_subscriber** project in Visual Studio and start the application from the command line:

```
> objs\i86Win32VS2005\MyType_subscriber
```

```
MyType subscriber sleeping for 4 sec...
MyType subscriber sleeping for 4 sec...
...
```

You will get this message repeatedly, since nothing is being published.

3. For publishing changes to a data source, you need to configure the "**RTIDDS_PUBLICATIONS**" table. For publishing the changes to the "**Example**" table, execute the following SQL statement:

```
mysql> insert into RTIDDS_PUBLICATIONS (table_owner, table_name,
domain_id, topic_name, type_name) values
('test', 'Example', 0, 'Example MyType', 'MyType');
```

   This entry directs *Real-Time Connect* to start publishing changes to table "**Example**" as the topic "**Example MyType**" with type "**MyType**" in Domain **0**.

4. Now change one of the previously captured samples in the "**Example**" table, for instance the last one:

```
mysql> update Example set message = 'Hello again!' where pkey =
(select * from (select max(pkey) from Example) as _max);
```

   You will see that the **MyType_subscriber** application reports the update, for example:

```
...
MyType subscriber sleeping for 4 sec...
pkey: 748
   message:
      message[ 0]: 'H'
      message[ 1]: 'e'
      message[ 2]: 'l'
      message[ 3]: 'l'
      message[ 4]: 'o'
      message[ 5]: ' '
      message[ 6]: 'a'
      message[ 7]: 'g'
      message[ 8]: 'a'
      message[ 9]: 'i'
      message[10]: 'n'
      message[11]: '!'
      message[12]: <0>
MyType subscriber sleeping for 4 sec...
...
```

5. You can also update all entries in the table:

   ```
   mysql> update Example set message = 'Hello again!';
   ```

   Notice that the **MyType_subscriber** application receives all changes, possibly a large number.

Chapter 9 License Installation

If your *Real-Time Connect* to Oracle distribution requires a license file, you will receive one via email from RTI.

If you have *RTI Connext Messaging* and you want to use *RTI Launcher*[1] to start *Real-Time Connect* to Oracle:

By default, *RTI Launcher* looks for the license file **rti_license.dat** in the top-level directory where you installed *Connext Messaging*. If you choose to save the license file elsewhere, you can configure *RTI Launcher* to look in a different location by using its Configuration tab.

Otherwise:

Save the license file in any location of your choice. When *Real-Time Connect* to Oracle starts, it will look in these locations until it finds a valid license:

1. The file specified with the **-licenseFile** option when you start *Real-Time Connect* to Oracle from the command-line.
2. The file **rti_license.dat** in the ***Real-Time Connect*** executable directory.
3. The file specified in the environment variable RTI_LICENSE_FILE, which you may set to point to the full path of the license file, including the filename (for example, **C:\RTI\my_rti_license.dat**).
4. The file **rti_license.dat** in the current working directory.
5. The file **rti_license.dat** in the directory specified by the environment variable NDDSHOME.

If you have any questions about license installation, please contact **support@rti.com**.

1. *Launcher* is a convenient GUI-based tool that can start and configure all of your *Connext Messaging* components, including *Real-Time Connect* to Oracle.