Oracle TimesTen In-Memory Database Troubleshooting Procedures Guide

Release 7.0

B31688-03

ORACLE
TIMESTEN

September 2007

Printed in the United States of America

Contents

Index

About this Guide

This guide describes how to troubleshoot some of the problems users encounter when using the Oracle TimesTen In-Memory Database.

To work with this guide, you should understand how database systems work and have some knowledge of SQL (Structured Query Language).

TimesTen documentation

TimesTen documentation is available on the product distribution media and on the Oracle Technology Network: http://www.oracle.com/technology/documentation/timesten_doc.html.

Including this guide, the TimesTen documentation set consists of these documents:

Book Titles	Description
Oracle TimesTen In-Memory Database Installation Guide	Contains information needed to install and configure TimesTen on all supported platforms.
Oracle TimesTen In-Memory Database Introduction	Describes all the available features in the Oracle TimesTen In-Memory Database.
Oracle TimesTen In-Memory Database Operations Guide	Provides information on configuring TimesTen and using the ttIsql utility to manage a data store. This guide also provides a basic tutorial for TimesTen.
Oracle TimesTen In-Memory Database C Developer's and Reference Guide and the *Oracle TimesTen In-Memory Database Java Developer's and Reference Guide*	Provide information on how to use the full set of available features in TimesTen to develop and implement applications that use TimesTen.
Oracle TimesTen In-Memory Database API Reference Guide	Describes all TimesTen utilities, procedures, APIs and provides a reference to other features of TimesTen.
Oracle TimesTen In-Memory Database SQL Reference Guide	Contains a complete reference to all TimesTen SQL statements, expressions and functions, including TimesTen SQL extensions.

Oracle TimesTen In-Memory Database Error Messages and SNMP Traps	Contains a complete reference to the TimesTen error messages and information on using SNMP Traps with TimesTen.
Oracle TimesTen In-Memory Database TTClasses Guide	Describes how to use the TTClasses C++ API to use the features available in TimesTen to develop and implement applications.
TimesTen to TimesTen Replication Guide	Provides information to help you understand how TimesTen Replication works and step-by-step instructions and examples that show how to perform the most commonly needed tasks. This guide is for application developers who use and administer TimesTen and for system administrators who configure and manage TimesTen Replication.
TimesTen Cache Connect to Oracle Guide	Describes how to use Cache Connect to cache Oracle data in TimesTen data stores. This guide is for developers who use and administer TimesTen for caching Oracle data.
Oracle TimesTen In-Memory Database Troubleshooting Procedures Guide	Provides information and solutions for handling problems that may arise while developing applications that work with TimesTen, or while configuring or managing TimesTen.

Background reading

For a Java reference, see:

- Horstmann, Cay and Gary Cornell. *Core Java(TM) 2, Volume I--Fundamentals (7th Edition) (Core Java 2)*. Prentice Hall PTR; 7 edition (August 17, 2004).

A list of books about ODBC and SQL is in the Microsoft ODBC manual included in your developer's kit. Your developer's kit includes the appropriate ODBC manual for your platform:

- *Microsoft ODBC 3.0 Programmer's Reference and SDK Guide* provides all relevant information on ODBC for Windows developers.
- *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*, included online in PDF format, provides information on ODBC for UNIX developers.

For a conceptual overview and programming how-to of ODBC, see:

- Kyle Geiger. *Inside ODBC*. Redmond, WA: Microsoft Press. 1995.

For a review of SQL, see:

- Melton, Jim and Simon, Alan R. *Understanding the New SQL: A Complete Guide*. San Francisco, CA: Morgan Kaufmann Publishers. 1993.
- Groff, James R. / Weinberg, Paul N. *SQL: The Complete Reference, Second Edition.* McGraw-Hill Osborne Media. 2002.

For information about Unicode, see:

- The Unicode Consortium, *The Unicode Standard, Version 5.0*, Addison-Wesley Professional, 2006.
- The Unicode Consortium Home Page at http://www.unicode.org

Conventions used in this guide

TimesTen supports multiple platforms. Unless otherwise indicated, the information in this guide applies to all supported platforms. The term Windows refers to Windows 2000, Windows XP and Windows Server 2003. The term UNIX refers to Solaris, Linux, HP-UX, Tru64 and AIX.

TimesTen documentation uses these typographical conventions:

If you see...	It means...
`code font`	Code examples, filenames, and pathnames. For example, the `.odbc.ini.` or `ttconnect.ini` file.
`italic code font`	A variable in a code example that you must replace. For example: `Driver=`*`install_dir`*`/lib/libtten.sl` Replace *`install_dir`* with the path of your TimesTen installation directory.

TimesTen documentation uses these conventions in command line examples and descriptions:

If you see...	It means...
`fixed width italics`	Variable; must be replaced with an appropriate value. In some cases, such as for parameter values in built-in procedures, you may need to single quote (' ') the value.
`[ ]`	Square brackets indicate that an item in a command line is optional.
`{ }`	Curly braces indicated that you must choose one of the items separated by a vertical bar (|) in a command line.
`|`	A vertical bar (or pipe) separates arguments that you may use more than one argument on a single command line.
`...`	An ellipsis (. . .) after an argument indicates that you may use more than one argument on a single command line.
`%`	The percent sign indicates the UNIX shell prompt.
`#`	The number (or pound) sign indicates the UNIX root prompt.

TimesTen documentation uses these variables to identify path, file and user names:

If you see...	It means...
install_dir	The path that represents the directory where the current release of TimesTen is installed.
TTinstance	The instance name for your specific installation of TimesTen. Each installation of TimesTen must be identified at install time with a unique alphanumeric instance name. This name appears in the install path. The instance name “giraffe” is used in examples in this guide.
bits or *bb*	Two digits, either 32 or 64, that represent either the 32-bit or 64-bit operating system.
release or *rr*	Two digits that represent the first two digits of the current TimesTen release number, with or without a dot. For example, 51 or 7.0 represents TimesTen Release 7.0.
jdk_version	Two digits that represent the version number of the major JDK release. Specifically, 14 represent JDK 1.4; 5 represents JDK 5.
timesten	A sample name for the TimesTen instance administrator. You can use any legal user name as the TimesTen administrator. On Windows, the TimesTen instance administrator must be a member of the Administrators group. Each TimesTen instance can have a unique instance administrator name.
DSN	The data source name.

Technical Support

For information about obtaining technical support for TimesTen products, go to the following Web address:

http://www.oracle.com/support/contact.html

1

Tools for Troubleshooting TimesTen

This chapter describes how to use the TimesTen utilities and other tools that are used to diagnose problems with the TimesTen data store. This chapter includes the following topics:

Using the ttIsql utility

The **ttIsql** utility allows you to interactively execute SQL statements and report status information on your data store.

All TimesTen SQL operations can be executed from a **ttIsql** `Command>` prompt.

Example 1.1 To start the **ttIsql** utility for the demo data store, enter:

```
% ttIsql demo
```

You should see output similar to the following:

```
Copyright (c) 1996-2007, Oracle.  All rights reserved.
Type ? or "help" for help, type "exit" to quit ttIsql.
All commands must end with a semicolon character.

connect "DSN=demo";
Connection successful:
DSN=demo;UID=ttuser;DataStore=c:\temp\demo;
DatabaseCharacterSet=US7ASCII;ConnectionCharacterSet=US7ASCII;
DRIVER=C:\WINDOWS\system32\ttdv70.dll;Authenticate=0;PermSize=20;
TypeMode=0;
(Default setting AutoCommit=1)
Command>
```

You can then execute SQL statements or **ttIsql** commands at the `Command>` prompt.

"Using the ttIsql Utility" in *Oracle TimesTen In-Memory Database Operations Guide* describes how to use the most common **ttIsql** commands. The following **ttIsql** commands are commonly used when troubleshooting:

- **monitor** formats the contents of the SYS.MONITOR table.

  See "Displaying data store information" in *Oracle TimesTen In-Memory Database Operations Guide*.

- **dssize** prints data store size information.

  See "Displaying data store information" in *Oracle TimesTen In-Memory Database Operations Guide*.

- **showplan** prints the optimizer execution plans for selects/updates/deletes in this transaction.

  See "Viewing and changing query optimizer plans" in *Oracle TimesTen In-Memory Database Operations Guide*.

- **isolation** sets or displays the isolation level.

  See "Working with transactions" in *Oracle TimesTen In-Memory Database Operations Guide*.

- **timing** prints query timing.

See "Timing ODBC function calls" in *Oracle TimesTen In-Memory Database Operations Guide*.

- **optprofile** prints the current optimizer flag settings and join order.

  See "Viewing and changing query optimizer plans" in *Oracle TimesTen In-Memory Database Operations Guide*.

For the full list of **ttIsql** features, see the lists of options and commands under the description of the **ttIsql** utility in *Oracle TimesTen In-Memory Database API Reference Guide*.

Using the ttStatus utility

Use the **ttStatus** utility to check the status of the TimesTen daemon and the state of all TimesTen connections.

Example 1.2 In this example, the output from **ttStatus** indicates that no TimesTen daemon is running. If the daemon has stopped unexpectedly, see "No response from TimesTen daemon or subdaemon" on page 39 for troubleshooting information.

On Windows:

```
C:\>ttStatus
ttStatus: Could not connect to the TimesTen service.
If the TimesTen service is not running, please start it by
running "ttDaemonAdmin -start".
```

On UNIX platforms:

```
$ ttStatus
ttStatus: Could not connect to the TimesTen daemon.
If the TimesTen daemon is not running, please start it
by running "ttDaemonAdmin -start".
```

Example 1.3 In this example, the output from **ttStatus** indicates that the TimesTen daemon is running. It recognizes one data store, named *demo*.

The first line indicates that the TimesTen daemon is running as process 884 on port 17000 for the TimesTen instance MYINSTANCE. The second line indicates the TimesTen server daemon is running as process 2308 on port 17002.

There are currently seven connections to the data store: one user and six subdaemon connections. You may see up to 2047 connections.

The restart policies for the cache agent and the replication agent in the data store are set to `manual`.

Note: This example was produced on Windows. The results are the same on UNIX platforms except for the formats of the data store path and the shared memory key.

```
C:\>ttStatus

TimesTen status report as of Thu Jan 25 15:45:11 2007

Daemon pid 884 port 17000 instance MYINSTANCE
TimesTen server pid 2308 started on port 17002
TimesTen webserver pid 2188 started on port 17004
----------------------------------------------------------------------
Data store c:\temp\demo
There are 7 connections to the data store
Data store is in shared mode
```

```
Shared Memory KEY Global\DBI45b94095.1.SHM.4 HANDLE 0x278

Type            PID     Context     Connection Name              ConnID

Process         4616    0x00d08820  demo                            1
Subdaemon       2136    0x00526768  Worker                       2042
Subdaemon       2136    0x0072e750  Flusher                      2043
Subdaemon       2136    0x007348b8  Checkpoint                   2044
Subdaemon       2136    0x067b0068  Aging                        2045
Subdaemon       2136    0x067c0040  Monitor                      2047
Subdaemon       2136    0x068404c8  HistGC                       2046
Replication policy : Manual
Cache agent policy : Manual
------------------------------------------------------------------------
End of report
```

Example 1.4 In this example, the output from **ttStatus** indicates that the TimesTen daemon is running. It recognizes three data stores: *demo*, *subscriber1ds*, and *masterds*. The *subscriber1ds* and *masterds* data stores are replicated data stores. In this example, the output from **ttStatus** indicates that the replication agents for the replicated data stores have been started. Bidirectional replication has been configured between *masterds* and *subscriber1ds*. Each replication agent has five connections to the data store.

```
C:\>ttStatus
TimesTen status report as of Thu Jan 25 16:23:33 2007
Daemon pid 5088 port 17000 instance MYINSTANCE
TimesTen server pid 4344 started on port 17002
TimesTen webserver pid 4216 started on port 17004
------------------------------------------------------------------------
Data store c:\temp\subscriber1ds
There are 12 connections to the data store
Data store is in shared mode
Shared Memory KEY Global\DBI45b9471c.2.SHM.2 HANDLE 0x280
Type            PID     Context     Connection Name              ConnID
Process         1244    0x00d08fb0  subscriber1ds                   1
Replication     4548    0x00aed2f8  LOGFORCE                        4
Replication     4548    0x00b03470  TRANSMITTER                     5
Replication     4548    0x00b725a8  RECEIVER                        6
Replication     4548    0x00b82808  REPHOLD                         2
Replication     4548    0x00b98980  REPLISTENER                     3
Subdaemon       2752    0x00526768  Worker                       2042
Subdaemon       2752    0x0072a758  Flusher                      2043
Subdaemon       2752    0x007308c0  Checkpoint                   2044
Subdaemon       2752    0x00736a28  HistGC                       2046
Subdaemon       2752    0x067f02f8  Aging                        2045
Subdaemon       2752    0x068364a0  Monitor                      2047
Replication policy  : Manual
```

```
Replication agent is running.
Cache agent policy : Manual
------------------------------------------------------------------------
Data store c:\temp\masterds
There are 12 connections to the data store
Data store is in shared mode
Shared Memory KEY Global\DBI45b945d0.0.SHM.6 HANDLE 0x2bc
Type            PID     Context     Connection Name              ConnID
Process         5880    0x00d09008  masterds                          1
Replication     3728    0x00aed570  LOGFORCE                          4
Replication     3728    0x00b036e8  TRANSMITTER                       5
Replication     3728    0x00b168b8  REPHOLD                           3
Replication     3728    0x00b1ca20  REPLISTENER                       2
Replication     3728    0x00b22b88  RECEIVER                          6
Subdaemon       3220    0x00526768  Worker                         2042
Subdaemon       3220    0x0072e768  Flusher                        2043
Subdaemon       3220    0x007348d0  Checkpoint                     2044
Subdaemon       3220    0x067b0068  Aging                          2045
Subdaemon       3220    0x067c0040  Monitor                        2047
Subdaemon       3220    0x068404c8  HistGC                         2046
Replication policy  : Manual
Replication agent is running.
Cache agent policy : Manual
------------------------------------------------------------------------
Data store c:\temp\demo
There are no connections to the data store
Replication policy : Manual
Cache agent policy : Manual
------------------------------------------------------------------------
End of report
```

Example 1.5 This example shows the cache agent running on *rep1* data store. There is one cache group in the data store. The cache agent has five connections to the data store.

```
C:\>ttStatus
TimesTen status report as of Mon Mar 19 10:47:46 2007

Daemon pid 1012 port 17000 instance MYINSTANCE
No TimesTen server running
TimesTen webserver pid 1708 started on port 17004

------------------------------------------------------------
Data store c:\data\rep1
There are 12 connections to the data store
Data store is in shared mode
Shared Memory KEY Global\DBI45ef98ac.1.SHM.56 HANDLE 0x260
Type          PID     Context       Connection Name        ConnID
Cache Agent   3380    0x00bbddf0    Handler                2
```

```
Cache Agent   3380    0x00c3f318    Aging                 3
Cache Agent   3380    0x07380398    Timer                 4
Cache Agent   3380    0x073cfa18    ttora70               6
Cache Agent   3380    0x073ff010    ttora70               7
Process       2084    0x00c48ee8    rep1                  1
Subdaemon     1632    0x006bc430    Worker                2042
Subdaemon     1632    0x06630458    Flusher               2045
Subdaemon     1632    0x0664f978    Checkpoint            2044
Subdaemon     1632    0x0665ee60    HistGC                2043
Subdaemon     1632    0x066de720    Aging                 2046
Subdaemon     1632    0x0670dc78    Monitor               2047
Replication policy  : Manual
Cache agent policy  : Manual
TimesTen's Cache agent is running for this data store
------------------------------------------------------------------------
End of report
```

Example 1.6 This example shows a connection to an old instance of a data store. This can occur when a data store is invalidated, but some users have not disconnected from the invalidated copy of the data store still in memory. After all users disconnect, the memory can be freed.

```
C:\>ttStatus

TimesTen status report as of Thu Jan 25 16:44:49 2007
Daemon pid 5088 port 17000 instance MYINSTANCE
TimesTen server pid 4344 started on port 17002
TimesTen webserver pid 4216 started on port 17004
------------------------------------------------------------------------
Data store c:\temp\sample
There are no connections to the data store
Obsolete or not yet active connection(s):
Process  4696 context 0xd08800 name sample connid 1, obsolete
connection, shmKey 'Global\DBI45b94c6f.3.SHM.4'
Replication policy : Manual
Cache agent policy : Manual
------------------------------------------------------------------------
End of report
```

Using the ttCapture utility

The **ttCapture** utility captures information about the configuration and state of your TimesTen system into a file that provides Technical Support with a snapshot of your system at the time you encountered a problem. When you experience a problem with a TimesTen data store, run the **ttCapture** utility for the data store as soon as possible, either when you are encountering the problem or immediately afterward.

The **ttCapture** utility generates a file named `ttcapture.out.`*`number`*. By default, the file is written to the directory from which you invoke the **ttCapture** utility. Use the **ttCapture** `-dest` option to direct the output file to be written to another directory.

Note: Always double-quote directory and file names in case there are spaces in the names.

On Windows platforms, **ttCapture** also produces a file named `syssum.`*`number`*`.nfo` that contains detailed information about your system hardware and configuration.

Example 1.7 On a Windows platform, to capture information related to the data store, *MyDataStore*:

```
% ttCapture MyDataStore
Capturing to file
c:\timesten\tt70\bin\ttcapture.out.20040701.3692
Capturing data store information...
Capturing installation information...
Capturing system information...
Creating msinfo dump in c:\timesten\tt70\bin\syssum.3692.nfo
Finished capture
```

When you contact Technical Support, upload the `ttcapture.out.`*`number`* generated file to the Service Request. Windows users should also upload the `syssum` file. This can expedite the investigation.

Using the logs generated by the TimesTen daemon

TimesTen uses a *TimesTen daemon* to manage access to the data stores. As the daemon operates, it generates error, warning and informational messages. These messages may be useful for TimesTen system administration and for debugging applications.

By default, informational messages are stored in:

- A user error log that contains information you may need to see. Generally, these messages contain information about actions you may need to take.
- A support log containing everything in the user error log plus information for use by Technical Support.

See "Modifying informational messages" in *Oracle TimesTen In-Memory Database Operations Guide* for information about configuring the logs, including their location and size.

Using the ttTraceMon utility

Use the **ttTraceMon** utility to log various trace information on a number of TimesTen components. Each TimesTen component can be traced at different levels of detail. You can list all of the traceable TimesTen components and their current tracing level by specifying **ttTraceMon** with the `show` subcommand. The full list of options for **ttTraceMon** is described in *Oracle TimesTen In-Memory Database API Reference Guide*.

TimesTen tracing severely impacts application performance and consumes a great deal of disk space if trace output is directed to a file. Use the **ttTraceMon** utility only when diagnosing problems. When you are finished, reset tracing to the default values.

Example 1.8 This example shows that the levels for most tracing components are set to level 0 (off) for the *demo* data store. ERR tracing is level 1 by default. See "ERR tracing" on page 22.

```
% ttTraceMon -e show demo
LATCH         ... 0
LOCK          ... 0
LOG           ... 0
LOGF          ... 0
TRACE         ... 0
API           ... 0
HEAP          ... 0
SM            ... 0
XACT          ... 0
EE            ... 0
CG            ... 0
SQL           ... 0
TEST          ... 0
FLOW          ... 0
PT            ... 0
ERR           ... 1
REPL          ... 0
OPT           ... 0
CKPT          ... 0
XA            ... 0
ORACON        ... 0
AGING         ... 0
AUTOREFRESH   ... 0
```

The output for most TimesTen components is of interest only to Technical Support. However, the output for the *SQL*, *API*, *LOCK*, *ERR*, *AGING* and *AUTOREFRESH* components may be useful to you when you are troubleshooting application problems.

The rest of this section includes the following topics:

Starting a trace and reading the trace buffer

Start a new trace by specifying **ttTraceMon** *datastore*. For example, to start a trace on the *demo* data store, enter:

```
%  ttTraceMon demo
Trace monitor; empty line to exit
Trace >
```

At the Trace prompt, specify the type of trace and its level. For example, to start tracing the SQL component at level 3, enter:

```
Trace > level sql 3
```

At this point you can run your application and the TimesTen trace information is written to a trace buffer. There are two ways to read the contents of the trace buffer:

- From the Trace prompt, use the `outfile` command to direct the trace buffer data to a file. (You must do this before running your application.) When writing tracing information to a file, new trace information is concatenated to the existing contents of the file.
- From the Trace prompt, use the `dump` command to display the trace buffer data to your screen.

Note: The contents of the trace buffer accumulate with each new trace. To clear the trace buffer, use the `flush` command from a **ttTraceMon** prompt. Clear the buffered trace records for a specific component by specifying the component with the `flush` command.

Each record from the trace buffer has the following format:

timestamp sequence component level connection processid operation

The fields in the records are defined as follows:

- *timestamp* is the time at which the operation was executed.
- *sequence* is the incremental number that identifies the trace line.
- *component* is the TimesTen component being traced (such as SQL, API, LOCK, or ERR).

- *level* is the trace level associated with the trace line. The range of trace levels differs by component, but for all components, the lowest trace level generates the least verbose output and highest trace level generates the most verbose output. For example, if you are tracing SQL at level 4, your output includes lines for levels 2 (prepare), 3 (execute), and 4 (open, close, fetch).

Note: Trace levels for some components are not a continuous range of numbers. If you enter a number that does not correspond to a supported level for a component, tracing occurs at the highest supported level that is less than the number you entered. For example, if tracing levels for a component are 1, 2, 3, 4, and 6 and you enter 5, tracing events for level 1, 2, 3, and 4 are generated.

- *connection* is the internal connection ID identifying the connection that generated the trace. This number corresponds to the ConnID shown in **ttStatus** output. The connection ID is also used as the first element of the transaction ID.
- *processid* is the operating system process ID for the process that generated the trace.
- *operation* is the operation that occurred (such as SQL statement, API operation, or error message).

For example, a line from the trace buffer after a SQL trace at level 3 might look like this:

```
10:39:50.231 5281 SQL    2L   1C  3914P Preparing: select cust_num
from customer
```

SQL tracing

Using **ttTraceMon** with the SQL component provides information about the SQL being prepared or executed by the TimesTen engine. Table 1.1 describes the levels for SQL tracing.

Table 1.1 SQL tracing levels

SQL Tracing Level	Output
2	SQL commands being prepared.
3	+ SQL commands being executed

SQL Tracing Level	Output
4	+ The effect of command pooling (prepares not being done because the prepared command already exists in the pool), the need for reprepares (for example, because an index was created), and commands being destroyed. At this level, **ttTraceMon** also shows when a query command is being opened, fetched, and closed.
5	+ Some internal data, such as command numbers, which are not generally useful for customer-level debugging.

Note: TimesTen recommends tracing SQL at level 3 or 4. SQL tracing does not show any information about the optimizer. Optimizer tracing is managed by a separate component (OPT) at level 4 only, and is not designed for customer use.

Example 1.9

In this example, we execute **ttTraceMon** to do a SQL trace at level 4 on the *demo* data store. We direct the output from the SQL trace to the *SQLtrace.txt* file. We then flush the buffer so that the trace does not report past SQL statements.

```
% ttTraceMon demo
Trace monitor; empty line to exit
Trace > outfile SQLtrace.txt
Trace > level sql 4
Trace > flush
```

At this point, we execute an application that includes the following SQL statement:

```
SELECT * FROM departments WHERE department_id = 10;
```

The trace information is written to the *SQLtrace.txt* file:

```
12:19:36.582     269 SQL      2L    3C  29570P Preparing: select *
from departments where department_id = 10
12:19:36.583     270 SQL      4L    3C  29570P sbSqlCmdCompile
()(E): (Found already compiled version: refCount:01, bucket:28)
cmdType:100, cmdNum:1000146.
12:19:36.583     271 SQL      4L    3C  29570P Opening: select *
from departments where department_id = 10;
12:19:36.583     272 SQL      4L    3C  29570P Fetching: select *
from departments where department_id = 10;
12:19:36.583     273 SQL      4L    3C  29570P Closing: select *
from departments where department_id = 10;
```

```
5 records dumped
```

When the application has completed, we turn off SQL tracing and exit **ttTraceMon**.

```
Trace > level sql 0
Trace > {press ENTER – blank line}
```

API tracing

API traces are generated for database operations such as connecting to a data store, changing a connection attribute, and committing a transaction. Table 1.2 describes the levels for API tracing.

Table 1.2 API tracing levels

API Tracing Level	Output
1	All rollback attempts by the subdaemon. This occurs if an application exits abruptly and the subdaemon recovers its connection.
2	+ Some low-on-space conditions.
3	+ Create, connect, disconnect, checkpoint, backup, and compact operations for the data store, as well as commit and rollback for each connection, and a few other operations.
4	+ Most other operations conducted at TimesTen's internal API level. It does not show numerous operations on the storage manager and indexes that are done internally.

Note: TimesTen recommends tracing at level 3.

Example 1.10 In this example, we execute **ttTraceMon** to do a API trace at level 3 on the *demo* data store. The output from the API trace is written to the *APItrace.txt* file. Before saving the API trace to the buffer, we use the `flush` command to empty the buffer.

```
% ttTraceMon demo
Trace monitor; empty line to exit
Trace> outfile APItrace.txt
Trace> level api 3
Trace > flush
```

At this point, we execute the application. When the application has completed, we turn off API tracing and exit **ttTraceMon**:

```
Trace > level api 0
Trace > {press ENTER – blank line}
```

The contents of *APItrace.txt* are similar to the sample output shown below. The output shows connection to the data store, setting the connection character set, setting the isolation level, and committing a transaction.

```
11:54:26.796    1016 API      3L    2C   4848P sb_dbConnect()(X)
11:54:26.796    1017 API      3L    2C   4848P sb_dbConnCharsetSet()(E)
11:54:26.796    1018 API      3L    2C   4848P sb_dbConnSetIsoLevel()(E)
11:54:39.795    1019 API      3L    2C   4848P sb_dbConnSetIsoLevel()(E)
11:54:45.253    1020 API      3L    2C   4848P sb_xactCommitQ()(E)
```

LOCK tracing

Use the LOCK component to trace the locking behavior of your application to detect trouble with deadlocks or lock waits. LOCK tracing generates a great deal of volume, so it is important to choose the appropriate level of tracing. Level 3, for example, begins to generate a large number of traces, as traces are written for fairly common events. In addition, the traces themselves may be somewhat hard to read in places. If you cannot discern enough information for your purposes, contact Technical Support for more information.

Table 1.3 describes the LOCK tracing levels.

Table 1.3 LOCK tracing levels

LOCK Tracing Level	Output
1	Deadlock cycles as they are discovered.
2	+ Failures to grant locks for any reason.
3	+ Lock waits (which may or may not be granted).
4	+ All lock grants/releases, some routine calls, and the mechanism of the deadlock detector.
6	+ Each step in cycle traversal.

Example 1.11 In this example, we execute **ttTraceMon** to do a LOCK trace at level 4 on the *myDSN* data store.

We make two connections to *myDSN*. For the first connection, we set autocommit on. We create table *test* and insert three rows. We create a materialized view using table *test*.

We turn on tracing at level 4 and use the `flush` command to empty the buffer.

```
% ttTraceMon myDSN
Trace monitor; empty line to exit
Trace> level lock 4
Trace> flush
```

For the second connection to *myDSN*, we set autocommit off. We insert a row into table test. Because autocommit is off, the row is not inserted into the table until we commit. A lock is held until we commit or roll back the transaction.

If we use the `dump` command to display the contents of the trace buffer, we see that there are no records in the trace buffer:

```
Trace> dump
0 records dumped
```

From the first connection, we try to drop the materialized view. We cannot drop the view because there is a transaction that has not been committed or rolled back:

```
Command> drop materialized view v;
 6003: Lock request denied because of time-out
Details: Tran 3.71 (pid 24524) wants Sn lock on table
TTUSER.TEST. But tran 1.42 (pid 24263) has it in IXn (request was
IXn). Holder SQL (insert into test values (100);)
The command failed.
```

The trace buffer contains the following information:

```
Trace> dump
20:09:04.789  174759 LOCK     3L    3C  24524P ENQ: xcb:00003 <Tbl 0x9b894,0x0>
0+Sn=>SL activity 0 Sn cnt=0; Holder xcb:00001 IXn
20:09:04.789  174760 LOCK     3L    3C  24524P Connection 3 scheduled for sleep
20:09:04.789  174761 LOCK     3L    3C  24524P Connection 3 sleeping
20:09:14.871  174762 LOCK     3L 2047C  24237P Connection 3 timed out
20:09:14.871  174763 LOCK     3L 2047C  24237P Connection 3 woken up
20:09:14.871  174764 LOCK     3L    3C  24524P Connection 3 awake
20:09:14.871  174765 LOCK     2L    3C  24524P ENQ: xcb:00003 <Tbl 0x9b894,0x0>
0+Sn=>TM activity 0 Sn cnt=1; Holder xcb:00001 IXn
7 records dumped
```

When finished with the trace, we set LOCK tracing back to its default setting (0) and exit **ttTraceMon**:

```
Trace > level lock 0
Trace > {press ENTER – blank line}
```

ERR tracing

It may be useful to trace the ERR component. For example, an ERR trace at level 4 shows all of the error messages that are pushed in the TimesTen direct driver (not all errors are output to the user because they are handled internally). ERR

tracing at level 1 is the default. No output is written for ERR tracing at level 2 and 3.

Table 1.4 describes ERR tracing levels.

Table 1.4 ERR tracing levels

ERR Tracing Level	Output
1 (set by default)	Fatal errors
4	+ All other error messages, many of which are handled internally by TimesTen.

Example 1.12

In this example, we execute **ttTraceMon** to do a ERR trace at level 4 on *myDSN* data store.

First we create a table:

```
Command> create table test (id tt_integer);
```

Next we turn on tracing at level 4. Rather than direct the trace output to a file as in the previous examples, we read it directly from the trace buffer. Before saving the ERR trace to the buffer, we use the `flush` command to empty the buffer.

```
% ttTraceMon myDSN
Trace monitor; empty line to exit
Trace> level err 4
Trace> flush
```

Now we execute a SQL script with three errors in it. The errors are:

- Creating a table with the same name as an existing table
- Using incorrect syntax to insert values into the table
- Inserting CHAR data into a TT_INTEGER column

```
Command> create table test (id tt_integer);
 2207: Table TEST already exists
The command failed.
Command> insert into test values 'abcd');
 1001: Syntax error in SQL statement before or at: "'abcd'",
character position:
 25
insert into test values 'abcd');
                        ^^^^^^
The command failed.
Command> insert into test values ('abcd');
 2609: Incompatible types found in expression
The command failed.
```

The trace information is written to the trace buffer. We display it by using the dump command.

```
Trace> dump
19:28:40.465  174227 ERR      4L    1C  24263P TT2207: Table TEST already exists
 -- file "eeDDL.c", lineno 2930, procedure "sbEeCrDtblEval()"
19:28:51.399  174228 ERR      4L    1C  24263P TT1001: Syntax error in SQL
statement before or at: "'abcd'", character position: 25
insert into test values 'abcd');
                        ^^^^^^
 -- file "ptSqlY.y", lineno 6273, procedure "reserved_word_or_syntax_error"
19:29:00.725  174229 ERR      4L    1C  24263P TT2609: Incompatible types found
in expression -- file "saCanon.c", lineno 12618, procedure "sbPtAdjustType()"
3 records dumped
```

We set ERR tracing back to its default setting (1) and exit **ttTraceMon**:

```
Trace > level err 1
Trace > {press ENTER – blank line}
```

AGING tracing

Use the **ttTraceMon** utility to obtain the following information:

- When aging starts and ends
- How many rows have been deleted by the aging subdaemon

See "Implementing aging in your tables" in *Oracle TimesTen In-Memory Database Operations Guide*.

Table 1.5 describes the AGING tracing levels.

Table 1.5 AGING tracing levels

Level	Description
1	Displays messages about the following events: • The aging subdaemon starts least recently used (LRU) or time-based aging. • The aging subdaemon repeats LRU aging because the LRU threshold was not met. • The aging subdaemon ends LRU or time-based aging.
2	+Displays messages about the following events *for each table*: • Aging has started. • Aging has ended. The message includes the reason for ending and the total number of rows deleted.

3	+Detailed report on how many rows were deleted during each aging cycle.
4	+Message every time the aging subdaemon wakes up.

Example 1.13 In this example, we execute **ttTraceMon** to do an AGING trace on *myDSN* data store. The data store contains TTUSER.MYTAB table, which has a time-based aging policy. The table is described as follows:

```
Command> describe TTUSER.MYTAB;

Table TTUSER.MYTAB:
  Columns:
   *ID                               TT_INTEGER NOT NULL
    TS                               TIMESTAMP (6) NOT NULL
  Aging use TS lifetime 3 minutes cycle 1 minute on

1 table found.
(primary key columns are indicated with *)
```

The table contains the following rows before the aging cycle begins:

```
Command> select * from TTUSER.MYTAB;
< 1, 2007-03-21 12:54:06.000000 >
< 3, 2010-03-17 08:00:00.000000 >
< 4, 2007-03-21 12:59:40.000000 >
< 5, 2007-03-21 13:00:10.000000 >
< 6, 2007-03-21 13:01:22.000000 >
5 rows found.
```

We execute **ttTraceMon** to do an AGING trace at level 3. Rather than direct the trace output to a file, we read it directly from the trace buffer. Before saving the AGING trace to the buffer, we use the `flush` command to empty the buffer.

```
% ttTraceMon myDSN
Trace monitor; empty line to exit
Trace> level aging 3
Trace> flush
```

Display the trace information in the buffer by using the `dump` command.

```
Trace> dump
13:16:56.802    1247 AGING    1L 2045C  17373P Entering
sbAgingTB(): curTime=78
13:16:56.803    1248 AGING    2L 2045C  17373P Entering
sbAgingOneTable(): curTime=78, ltblid= 637140
13:16:56.804    1249 AGING    3L 2045C  17373P curTime=78, 4
deleted, 1 remaining, tbl = TTUSER.MYTAB
13:16:56.804    1250 AGING    2L 2045C  17373P Exiting
sbAgingOneTable(): curTime=78, reason = 'no more rows', 4
deleted, 1 remaining, tbl = TTUSER.MYTAB
```

```
13:16:56.804    1251 AGING    1L 2045C  17373P Exiting
sbAgingTB(): curTime=78
5 records dumped
```

We set AGING tracing back to its default setting (0) and exit **ttTraceMon**:

```
Trace > level aging 0
Trace > {press ENTER – blank line}
```

AUTOREFRESH tracing

Use the **ttTraceMon** utility to obtain information about the progress of autorefresh operations for cache groups with the AUTOREFRESH cache group attribute.

See "AUTOREFRESH cache group attribute" in *TimesTen Cache Connect to Oracle Guide*.

Table 1.6 describes AUTOREFRESH tracing levels.

Table 1.6 AUTOREFRESH tracing levels

Level	Description
1	Autorefresh summary for the interval: • The time that autorefresh started • Number of autorefreshed rows for the interval • Number of autorefreshed root table rows for interval • Total number of autorefreshed rows since the cache agent started • Total number of autorefreshed rows in the root table since the cache agent started • The time that autorefresh ended **Note:** Times and information about root table rows are reported for full autorefresh.
2	+Autorefresh summary at the cache group level: • The time that autorefresh started for each cache group • Number of autorefreshed rows for each cache group • Number of autorefreshed root table rows for each cache group • Total number of autorefreshed rows for each cache group since the cache agent started • Total number of autorefreshed rows in the root table for each cache group since the cache agent started • The time that autorefresh ended for each cache group **Note:** Times and information about root table rows are reported for full autorefresh.

Level	Description
3	+Autorefresh summary at the table level: • The time that autorefresh started • Number of autorefreshed rows • Total number of autorefreshed rows since the cache agent started • The time that autorefresh ended
4	+Autorefresh details for each table: • The time that autorefresh started for each table • The autorefresh query • Query execute time in milliseconds on the Oracle database • Query fetch time in milliseconds on the Oracle database • Query apply time in milliseconds on TimesTen • Query execute time in milliseconds on the Oracle database for child tables • Query fetch time in milliseconds on the Oracle database for child tables • Query apply time in milliseconds on TimesTen for child tables • The time that autorefresh ended for each table • The autorefresh bookmark (`logseq`) to which autorefresh was completed

Example 1.14 In this example, we use the **ttTraceMon** utility to trace autorefresh operations on the *cgDSN* data store. When we set the trace level to 1, we see information that is similar to the output of the **ttCacheAutorefreshStatsGet** built-in procedure.

```
% tttracemon cgDSN
Trace monitor; empty line to exit
Trace> level autorefresh 1
Trace> dump

08:56:57.445 19398 AUTOREFRESH 1L 5C 32246P Autorefresh number 1415 started for
interval 60000
08:56:57.883 19419 AUTOREFRESH 1L 5C 32246P Duration For Interval 60000ms: 420
08:56:57.883 19420 AUTOREFRESH 1L 5C 32246P Num Rows For Interval 60000ms: 0
08:56:57.883 19421 AUTOREFRESH 1L 5C 32246P Num Root Rows For Interval 60000ms:
0
08:56:57.883 19422 AUTOREFRESH 1L 5C 32246P Cumulative Rows for Interval
60000ms: 11587
```

```
08:56:57.883 19423 AUTOREFRESH 1L 5C 32246P Cumulative Root Rows for Interval
60000ms: 1697
08:56:57.883 19424 AUTOREFRESH 1L 5C 32246P Autorefresh number 1415 ended for
interval 60000ms successfully.
7 records dumped
```

Tracing at level 4 produces additional information about autorefresh operation 1415. Information about autorefresh is provided for the *testuser.readcache* cache group, the *testuser.readtab* root table and the autorefresh interval.

```
Trace> level autorefresh 4
Trace> dump
```

```
08:56:57.445 19398 AUTOREFRESH 1L 5C 32246P Autorefresh number 1415 started for
interval 60000
08:56:57.471 19399 AUTOREFRESH 2L 5C 32246P Autorefresh started for cachegroup
TESTUSER.READCACHE
08:56:57.471 19400 AUTOREFRESH 3L 5C 32246P Incremental autorefresh started for
table TESTUSER.READTAB
08:56:57.471 19401 AUTOREFRESH 4L 5C 32246P Autorefresh Query: SELECT
L."COL_10", X."COL_20", X.ft$NotDelete, Z.rowid FROM (SELECT DISTINCT "COL_10"
FROM "TESTUSER"."TT_03_454854_L" WHERE logseq >:logseq AND ft_cacheGroup <>
100000000000*1844259679+-299200618) L,(SELECT "TESTUSER"."READTAB"."COL_10",
"TESTUSER"."READTAB"."COL_20", 1 AS ft$NotDelete  FROM "TESTUSER"."READTAB",
"TESTUSER"."T1" WHERE "TESTUSER"."READTAB"."COL_10" = "TESTUSER"."T1"."COL_10")
X, "TESTUSER"."READTAB" Z WHERE L ."COL_10" = X."COL_10" (+) AND X."COL_10" =
Z."COL_10" (+), logseq: 7
08:56:57.870 19402 AUTOREFRESH 3L 5C 32246P Duration for table TESTUSER.READTAB:
70
08:56:57.870 19403 AUTOREFRESH 3L 5C 32246P Num Rows for table TESTUSER.READTAB:
1
08:56:57.870 19404 AUTOREFRESH 3L 5C 32246P Cumulative rows for table
TESTUSER.READTAB: 1559
08:56:57.870 19405 AUTOREFRESH 4L 5C 32246P Autorefresh Query Execute duration
for table TESTUSER.READTAB: 60
08:56:57.870 19406 AUTOREFRESH 4L 5C 32246P Autorefresh Query Fetch duration for
table TESTUSER.READTAB: 0
08:56:57.870 19407 AUTOREFRESH 4L 5C 32246P Autorefresh Query Apply duration for
table TESTUSER.READTAB: 0
08:56:57.870 19408 AUTOREFRESH 4L 5C 32246P Max logseq applied for table
TESTUSER.READTAB: 8
08:56:57.870 19409 AUTOREFRESH 4L 5C 32246P Autorefresh Query Execute duration
for 7 child(ren) table(s): 32
08:56:57.870 19410 AUTOREFRESH 4L 5C 32246P Autorefresh Query Fetch duration for
7 child(ren) table(s): 0
08:56:57.870 19411 AUTOREFRESH 4L 5C 32246P Autorefresh Query Apply duration for
7 child(ren) table(s): 0
08:56:57.870 19412 AUTOREFRESH 3L 5C  32246P Incremental autorefresh ended for
table TESTUSER.READTAB
```

```
08:56:57.872 19413 AUTOREFRESH 2L 5C 32246P Duration For Cache Group
TESTUSER.READCACHE: 1020
08:56:57.872 19414 AUTOREFRESH 2L 5C 32246P Num Rows For Cache Group
TESTUSER.READCACHE: 1
08:56:57.872 19415 AUTOREFRESH 2L 5C 32246P Num Root Rows For Cache Group
TESTUSER.READCACHE: 0
08:56:57.872 19416 AUTOREFRESH 2L 5C 32246P Cumulative Rows for Cache Group
TESTUSER.READCACHE: 11776
08:56:57.872 19417 AUTOREFRESH 2L 5C 32246P Cumulative Root Rows for Cache Group
TESTUSER.READCACHE: 1697
08:56:57.872 19418 AUTOREFRESH 2L 5C 32246P Autorefresh ended for cache group
TESTUSER.READCACHE
08:56:57.883 19419 AUTOREFRESH 1L 5C 32246P Duration For Interval 60000ms: 420
08:56:57.883 19420 AUTOREFRESH 1L 5C 32246P Num Rows For Interval 60000ms: 0
08:56:57.883 19421 AUTOREFRESH 1L 5C 32246P Num Root Rows For Interval 60000ms:
0
08:56:57.883 19422 AUTOREFRESH 1L 5C 32246P Cumulative Rows for Interval
60000ms: 11587
08:56:57.883 19423 AUTOREFRESH 1L 5C 32246P Cumulative Root Rows for Interval
60000ms: 1697
08:56:57.883 19424 AUTOREFRESH 1L 5C 32246P Autorefresh number 1415 ended for
interval 60000ms successfully.
27 records dumped
```

We set AUTOREFRESH tracing back to its default setting (0) and exit **ttTraceMon**:

```
Trace > level autorefresh 0
Trace > {press ENTER – blank line}
```

Using the ttXactAdmin utility

The **ttXactAdmin** utility displays ownership, status, log and lock information for each outstanding transaction. You can also use it to show all current connections to a data store. **ttXactAdmin** is useful for troubleshooting problems with replication, XLA, and asynchronous writethrough cache groups.

Example 1.15

Use **ttXactAdmin** to diagnose a lock timeout. Consider two connections that are trying to update the same row. The following transaction by Connection 1 is in progress:

```
UPDATE table1 SET c1 = 2 WHERE c1 = 1;
```

Connection 2 attempts to make the following update:

```
UPDATE table1 SET c1 = 3 WHERE c1 = 1;
```

Connection 2 receives the following error:

```
6003: Lock request denied because of time-out
  Details: Tran 2.3 (pid 2880) wants Un lock on rowid 0x00156bbc,
table TTUSER.TABLE1.
  But tran 1.21 (pid 2564) has it in Xn (request was Xn). Holder
SQL (update table1 set c1 = 2 where c1 = 1;)
  The command failed.
```

The details of the error indicate that transaction 1.21 has a lock on row 0x00156bbc, the row that transaction 2.3 wants to update. **ttXactAdmin** displays this information in output that pertains to actions in the entire data store:

```
$ ttXactAdmin myDSN
2007-03-23 11:26:01.643
c:\datastore\myDSN
TimesTen Release 7.0.2.0.0

Outstanding locks

PID   Context   TransID  TransStatus  Resource   ResourceID   Mode Name

Program File Name: ttIsql

2564  0xeeb9a8  1.21     Active       Database   0x01312d00   IX
                                      Row        0x00156bbc   Xn   TTUSER.TABLE1
                                      Table      1910868      IXn  TTUSER.TABLE1

Program File Name: ttIsql

2880  0xeeb9a8  2.3      Active        Database   0x01312d00  IX
                                       Table      1910868     IXn  TTUSER.TABLE1
                                       Command    19972120    S

Awaiting locks
```

```
PID  Context   TransID Resource ResourceID  RMode HolderTransID HMode Name
2880 0xeeb9a8  2.3     Row      0x00156bbc  Un    1.21          Xn TTUSER.TABLE1

2 outstanding transactions found
```

See "ttXactAdmin" in *Oracle TimesTen In-Memory Database API Reference Guide*.

Using ODBC tracing

On Windows, use the ODBC trace facility to verify the sequence and content of your commands. The ODBC trace facility works only if you have linked your application with the ODBC Driver Manager. Enable tracing by double-clicking **ODBC** in the Control Panel. This opens the ODBC Data Source Administrator. Choose the **Tracing** tab.

On UNIX platforms, ODBC tracing is available only when using a driver manager. To turn on tracing, set the **Trace** and **TraceFile** attributes.

Using SNMP traps to detect events

Network management software uses SNMP (Simple Network Management Protocol) to query or control the state of network devices such as routers and switches. These devices can generate alerts called *traps* to inform the network management systems of problems.

TimesTen sends SNMP traps for particular critical events to help facilitate user recovery mechanisms. These events are also recorded in the support log. Exposing them through SNMP traps allows network management software to take immediate action.

How to configure TimesTen to generate SNMP traps as well as how to receive the traps is described in "Diagnostics through SNMP Traps" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

To understand how network software might be used to detect SNMP traps, use the `snmptrapd` program provided in your TimesTen directory: `/install_dir/demo/snmp`. This demo listens on a designated port for SNMP trap messages and either prints the traps to stdout or logs them to syslogd. See the `/install_dir/demo/snmp/README.txt` file for details.

Monitoring the TimesTen system tables

Each TimesTen data store contains a group of system tables that store metadata about the current state of the data store. The system tables are described in "System and Replication Tables" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

Note: You can execute SELECT statements on a system table, but you cannot execute a statement such as INSERT, UPDATE or DELETE on these tables.

Of particular interest when troubleshooting is the SYS.MONITOR table, which contains statistics about certain events that have occurred since the first connection to the data store. For example, the SYS.MONITOR table contains information about the number of connections to the data store; the number of checkpoints taken; the size of the data store; and the amount of memory currently in use. Check the contents of the SYS.MONITOR table by executing SELECT statements on the columns or by using the **ttIsql monitor** command. For an example of how to use the **ttIsql monitor** command, see Example 5.17 in "Using the ttIsql Utility" in *Oracle TimesTen In-Memory Database Operations Guide*.

The SYS.PLAN table is useful for troubleshooting performance problems. See "Reading the PLAN table" in *Oracle TimesTen In-Memory Database Operations Guide* for details. Check the contents of the SYS.PLAN table by executing SELECT statements on the columns or by using the **ttIsql showplan** command, as described in "Viewing and changing query optimizer plans" in *Oracle TimesTen In-Memory Database Operations Guide*.

Using the query optimizer

The query optimizer is an important tool for performance tuning.

For details about using the query optimizer, see:

- "The TimesTen Query Optimizer" in *Oracle TimesTen In-Memory Database Operations Guide*
- "Viewing and changing query optimizer plans" in *Oracle TimesTen In-Memory Database Operations Guide*

If you find that a given query runs more slowly than expected, confirm that the query optimizer has the latest statistics for the tables in your query, as described in “Update query optimizer statistics” on page 61. If, after updating your statistics, your query still runs too slowly, it is possible that the TimesTen optimizer is not choosing the optimal query plan to answer that query. Under these circumstances, you can adjust how the optimizer generates a plan by using the **ttOpt*** procedures described in "Modifying plan generation" in *Oracle TimesTen In-Memory Database Operations Guide*.

2

Troubleshooting TimesTen Applications and Data Stores

This chapter helps you diagnose and remedy some of the problems encountered while using a TimesTen data store.

If you are still having problems with your data store after following the troubleshooting recommendations in this chapter, please contact Technical Support.

This chapter includes the following topics:

Unable to start or stop TimesTen daemon

This section describes what to check if you are unable to start or stop the TimesTen main daemon.

Possible cause	What to do
Incorrect privilege	Unless TimesTen was installed for non-root access, you need to have root or ADMIN privileges to start or stop the TimesTen daemon. Ensure that you are using the **ttDaemonAdmin** utility to start the daemon. The output from **ttDaemonAdmin** shows whether you have the correct privilege.
Another process is using the TimesTen daemon port.	Use the **ttVersion** utility to verify what port number the TimesTen daemon is expected to use. Use an OS command like `netstat` to check whether another process is listening on the port. If there is a conflict, either change the port number used by the other process or use **ttmodinstall** to change the port used by TimesTen.
TimesTen daemon is already running.	Ensure that you are using the **ttDaemonAdmin** utility to start the daemon. The output from **ttDaemonAdmin** shows whether the daemon is already running.
Other problems	Inspect the user error log produced by the daemon. See "Using the logs generated by the TimesTen daemon" on page 15.

No response from TimesTen daemon or subdaemon

This section describes what to do if one or more of the TimesTen processes appears to be unavailable:

- Check the TimesTen user error log
- Extract a stack trace from the core file

Check the TimesTen user error log

If you receive an error that indicates the TimesTen subdaemon has stopped, inspect the user error log, as described in "Using the logs generated by the TimesTen daemon" on page 15.

If the TimesTen daemon crashes, it cannot send anything to the user error log, but the subdaemons send a 'main daemon vanished' message to the log before exiting:

```
09:24:13 Err : 4375 ------------------: Main daemon has vanished
```

Restart the daemon. The next connection to each data store causes TimesTen to recover from the checkpoint and log files. See "Working with the Oracle TimesTen Data Manager Daemon" in *Oracle TimesTen In-Memory Database Operations Guide*.

Extract a stack trace from the core file

If you experience a crash by one of the TimesTen processes on a UNIX system and have exhausted all of the diagnostic options, check to see if TimesTen has generated a core file. Use the **ttVersion** utility to find the core file. Look for a line in the output that shows a path for the daemon home directory:

```
TimesTen Release 7.0.2.0.0 (32 bit Linux/x86) (ttuser:40732)
2007-04-04T17:53:04Z
   Instance admin: ttuser
   Instance home directory:
/node1/ttuser/ttcur/TTBuild/linux86_dbg/install
   Daemon home directory:
/node1/ttuser/ttcur/TTBuild/linux86_dbg/install/info
```

After locating the core file, attach to the debugger on the system and extract the stack trace from the core file and send the trace results to Technical Support.

On Windows systems you can obtain diagnostic information for a service failure by enabling the 'allow service to interact with desktop' option in the properties dialog for the TimesTen data manager in the **Service** menu. If a fatal fault occurs in the TimesTen data manager service, a pop-up asks if you would like to start the debugger. Contact Technical Support and provide the stack trace.

Application unable to connect to data store in direct mode

This section describes what to check if your application is unable to connect to a data store in direct mode.

Possible cause	See...
Mismatch between the release of TimesTen and data store	“Upgrading your data store” on page 40
Access Control is enabled on the TimesTen data store and user does not have access.	“Access Control privilege to access to data store” on page 41
Incorrect file permissions	"Check file system permissions to access data store"
TimesTen daemon or Data Manager service not running	“Check that the TimesTen daemon is running” on page 41
Incompatible connection attributes or incorrect path name for data store set in the DSN	“Check DSN definition” on page 41
No available shared memory segment or maximum size of shared memory segment too small	“Check size and availability of shared memory segments” on page 42
Not enough swap space	“Check available swap space (virtual memory)” on page 42
Inadequate number of file descriptors	“Increase the number of available file descriptors” on page 43
Other possible causes	“Using the logs generated by the TimesTen daemon” on page 15.

Upgrading your data store

A data store is only guaranteed to be accessible by the same minor release of TimesTen that was used to create the data store. When you upgrade the TimesTen software and you would like to use the new release to access a data store that was previously created, create a data store with the new release. Then use the **ttMigrate** utility to copy the tables, indexes, and table data from the old data store to the new one.

See "Data Store Upgrades" in *Oracle TimesTen In-Memory Database Installation Guide* for details.

Access Control privilege to access to data store

If Access Control is enabled on the data store, you need 'CREATE DATASTORE' privilege to access it. If you do not have access, the administrator must use the GRANT statement to grant you 'CREATE DATASTORE' privilege.

Check file system permissions to access data store

A "permission denied" error is generated if you attempt to connect to a data store and you do not have the proper permissions to access the checkpoint or log files or the directory where those files reside. Check the file system permissions on the files located in the directory specified in the **DataStore** in your DSN.

If the **GroupRestrict** attribute is set for the data store, confirm that you are listed in the specified group.

Check that the TimesTen daemon is running

If the TimesTen daemon or Data Manager service is not running, an attempt to connect to a data store generates TimesTen error 799 (Unable to connect to daemon; check daemon status).

Use the **ttStatus** utility as described in "Check the TimesTen user error log" on page 39 to check the status of the TimesTen daemon.

Check DSN definition

In your DSN description:

- Check DSN attributes
- Check path name to data store and log directories

Check DSN attributes

Certain connection options or DSN attribute settings combinations are not compatible. For example, if **Logging** is disabled, the default row-level lock setting (**LockLevel**=0) cannot be used. In cases where incompatible settings are used, an error is returned to the application when it attempts to connect to a data store.

Check path name to data store and log directories

Confirm that you have specified the correct path names in the **DataStore** and **LogDir** attributes in your DSN. Also confirm that the path names are absolute path names, rather than relative. Otherwise, the path name will be relative to the directory where the application was started.

On Windows, be careful to distinguish between User and System DSNs in the ODBC Data Source Administrator. Do not create user DSNs because they are visible only to the user who defines them. System DSNs are visible to all users.

In particular, if you run a TimesTen application as a Windows service, it runs as the user “SYSTEM” by default and does not see any User DSNs. Make sure that you are not using a mapped drive in the data store path name.

Check size and availability of shared memory segments

An error is generated if you attempt to connect to or create a shared data store whose size is larger than the maximum size of shared memory segments configured on your system. Also, an error is generated if the system cannot allocate any more shared memory segments.

On UNIX systems, use commands similar to the following:

- **ipcs -ma** to check if you have other shared memory segments using up memory, such as Oracle instances or other instances of TimesTen.
- **ipcrm** to remove a message queue, semaphore set or shared memory segment identifier after a faulty TimesTen shutdown.
- **ps -eafl** to see how much memory is being used by running processes.
- **ulimit -a** to see if there are any limits on the maximum amount of memory one process can address, maximum file size, and the maximum number of open files.

If a shared memory segment is available but is too small to hold your data store, use the **ttSize** utility to estimate the amount of memory required for your tables and then check the values of the **PermSize** and **TempSize** attributes to verify the amount of memory established for your data store. "Changing data store size" in *Oracle TimesTen In-Memory Database Operations Guide* describes guidelines for setting the size of your permanent and temporary data partitions. If the amount of memory established for your data store is too large, reset **PermSize** and **TempSize** to smaller values. See “Check the amount of memory allocated to the data store” on page 59 for more information. Another option is to increase the maximum size of the shared memory segment, as described below.

If a data store becomes invalidated because of a system or application failure, a subsequent connection recovers the data store. If recovery fails because you have run out of data store space, then reconnect to the data store with a larger **PermSize** and **TempSize** value than the ones that are currently in effect. If recovery fails because you do not have enough shared memory, then you should increase the maximum size of the shared memory segments for the system.

For more information on how to configure shared memory for TimesTen, see "Installation prerequisites" in *Oracle TimesTen In-Memory Database Installation Guide*.

Check available swap space (virtual memory)

There must be enough swap space to back up shared memory.

On UNIX systems, use the **swap** command to check and add virtual memory to your system.

On Windows systems, check and reset the size of your virtual memory from the **Advanced** tab in your **Computer Management Properties** dialog window.

Increase the number of available file descriptors

Each process connected to a TimesTen data store keeps at least one operating system file descriptor open. Additional file descriptors may be opened for each connection if disk logging is enabled, checkpoints are issued, and transactions are committed or rolled back. If you receive an error that all file descriptors are in use when attempting to connect to a data store, then increase the allowable number of file descriptors. See your operating system documentation for limits on file descriptors and information about changing the number of file descriptors.

Troubleshooting client/server problems

This section includes the following topics:

Also consider the topics described in "Application unable to connect to data store in direct mode" on page 40.

Cannot connect to the TimesTen Server

You have not correctly identified the system where the TimesTen Server is running.

On a Windows client machine, select the TimesTen Server in the TimesTen Data Source Setup dialog that is displayed as part of the ODBC Data Source Administrator. To verify the TimesTen Server:

1. On the Windows Desktop, choose **Start > Settings > Control Panel**.
2. Double click the **ODBC** icon. This opens the ODBC Data Source Administrator.
3. Click the **System DSN** tab. This displays the System Data Sources list.
4. Select the TimesTen Client data source. This opens the TimesTen Client DSN Setup dialog.
5. Click **Servers**. This opens the TimesTen Logical Server List.
6. Select the TimesTen Server from the list. This opens the TimesTen Logical Server Name Setup dialog.
7. Verify that the values for the **Network Address** and **Port Number** are correct. If necessary, change the values.

Note: If you typed the hostname or network address directly into the Server Name field of the TimesTen Client DSN Setup, the Client tries to connect to the TimesTen Server using the default port.

If the Network Address and Port Number values are correct, the TimesTen Server may not be running. See "Starting and stopping the Oracle TimesTen Data Manager Service on Windows" in *Oracle TimesTen In-Memory Database Operations Guide* for information about starting the server manually. See "Testing connections" in *Oracle TimesTen In-Memory Database Operations Guide* for more information about identifying this problem.

On UNIX, specify the TimesTen Server with the **TTC_Server** connection attribute in the ODBC.INI file on the client machine. If the value specified for **TTC_Server** is an actual hostname or IP address, the client tries to connect to the TimesTen Server using the default port. In TimesTen, the default port is associated with the TimesTen release number. If the value specified for **TTC_Server** is a logical ServerName, this logical ServerName must be defined in the TTCONNECT.INI file. The TTCONNECT.INI entry for this ServerName needs to correctly define the hostname/IP address and port number on which the TimesTen Server is listening.

If the **Network Address** and **Port Number** values are correct, the TimesTen Server may not be running or did not start. See "Starting and stopping the daemon on UNIX" in *Oracle TimesTen In-Memory Database Operations Guide* for information about starting the server manually. See "Testing connections" in *Oracle TimesTen In-Memory Database Operations Guide* for more information about identifying this problem.

TimesTen Server failed

Check the server's log file. Server log messages are stored in the files specified by the `-userlog` and `-supportlog` options in the `ttendaemon.options` file. See "Creating and configuring Client DSNs on UNIX"" and "Managing TimesTen daemon options" in *Oracle TimesTen In-Memory Database Operations Guide*.

The maximum number of concurrent IPC connections to the Server of a particular TimesTen instance is 24,999. However, TimesTen has a limit of 2043 connections (direct or client/server) to a single DSN.

Client/server users can change the file descriptor limit to support a large number of connections. For an example, see "Installation prerequisites" in *Oracle TimesTen In-Memory Database Installation Guide*.

Cannot find TimesTen Server DSN

On UNIX, verify that the Server DSN is defined in the SYS.ODBC.INI file on the machine running the TimesTen Server.

On Windows, verify that the Server DSN is defined as a System DSN in the ODBC Data Source Administrator on the machine running the TimesTen Server. See "Creating and configuring a logical server name" in *Oracle TimesTen In-Memory Database Operations Guide*.

TimesTen Server failed to load DRIVER

This error only occurs on UNIX platforms. Open the SYS.ODBC.INI file on the machine running the TimesTen Server and locate the Server DSN you are trying to connect. Verify that the dynamic library specified in the DRIVER attribute for the Server DSN exists and is executable.

Application times out when accessing TimesTen Server

The default TimeOut interval is 60 seconds.

To increase this interval on UNIX, change the value of the **TTC_Timeout** attribute in the ODBC.INI file.

To set the timeout interval on Windows, see the instructions in "Setting the timeout interval and authentication" in *Oracle TimesTen In-Memory Database Operations Guide*.

TimesTen Client loses connection with TimesTen Server

Check to see if the error was due to the Client timing out. Check the TimesTen Server's log to see why the Server may have severed connection with the Client. Use ping to determine if your network is up or try using `telnet` to connect to the TimesTen Server port number.

Failed to attach to shared memory segment for IPC

While using shared memory segment (SHM) as IPC, the application may see the following error message from the TimesTen Client ODBC Driver if the application reaches the system-defined per-process file-descriptor-limit.

```
SQLState    = S1000,
Native Error = 0,
Message     = [TimesTen][TimesTen 6.0 CLIENT]Failed to
    attach to shared memory segment for IPC. System error: 24
```

This may happen during a connect operation to the Client DSN when the `shmat` system call fails because the application has more open file descriptors than the system-defined per-process file descriptor limit. To correct this problem, you must increase your system-defined per-process file descriptor limit. For more information about file descriptor limits, see "System Limits" in *Oracle TimesTen In-Memory Database SQL Reference Guide*.

Increasing the maximum Server connections on Windows XP

On Windows XP, by default, there can be approximately 47 child server processes. You can increase the number of connections by setting the **MaxConnsPerServer** connection attribute in the `ttendaemon.options` file or in

the DSN. This increases the number of connections to 47 times the **MaxConnsPerServer** value.

Thread stack overflow when using multiple client connections

On Solaris, you may receive messages in the user error log about thread stack overflow. On other platforms, you may receive messages about a segmentation fault that mention a possible thread stack overflow.

If these messages occur, increase the server stack size by one of the following methods:

- Specify the `-ServerStackSize` option in the `ttendaemon.options` file. The `ttendaemon.options` file applies to all DSNs in the TimesTen instance.
- Specify the **ServerStackSize** connection attribute for a specific DSN. This takes precedence over the value in the `ttendaemon.options` file.

Increasing the server stack size decreases the number of concurrent connections that can be made before running out of swap space.

See "Working with the TimesTen Client and Server" in *Oracle TimesTen In-Memory Database Operations Guide*.

Out of space when DSN specifies new data store

You may receive “out of space” messages if you change a DSN to specify a new data store while there are existing connections to the original data store in a system with multiple client connections. This can happen on 32-bit platforms if either data store is close to 2 GB.

Close all connections to the original data store. This causes a new server process to be created for connections to the data store that is now specified in the DSN. Use the **ttStatus** utility to list the connections for the old data store. Alternatively, you can restart the server by using the **ttDaemonAdmin** utility with the `-restartServer` option, which resets all client connections on all DSNs in the instance.

Application connects or disconnects are slow

This section describes what to check if you encounter slow connects and disconnects to a data store.

Possible cause	See...
Data store is being recovered.	"Check if data store is being recovered" on page 48
ODBC tracing is enabled.	"Check ODBC tracing" on page 48
Other possible causes	"API tracing" on page 20

Check if data store is being recovered

A slow connect may indicate that a TimesTen data store is being recovered. This happens only for a first connect.

Check ODBC tracing

On Windows platforms, if ODBC tracing is enabled, it can slow connect and disconnect speeds. Double-click **ODBC** in the Control Panel to open the ODBC Data Source Administrator. Select the **Tracing** tab and confirm tracing is disabled. See "Using ODBC tracing" on page 33.

Application becomes disconnected unexpectedly

If an application becomes disconnected from a TimesTen data store, one of the following events occurs:

- If there was no outstanding transaction, the connection is cleanly removed by the TimesTen daemon. Other existing connections continue processing as if no problem had occurred.
- If there was an outstanding transaction but the application was not in the middle of executing code in the TimesTen library, the transaction is rolled back and the connection is cleanly removed by the TimesTen daemon. Other existing connections continue processing as if no problem had occurred.

This section describes what to check if your application unexpectedly disconnects from the data store.

Possible cause	See...
Internal application error.	"Check for ODBC or JDBC errors" on page 49
Failure of a concurrent application thread.	"Check for ODBC or JDBC errors" on page 49 "Check the user error log" on page 50
If using a client/server connection, the client may have disconnected from the application.	"Troubleshooting client/server problems" on page 44
An error in the TimesTen library	Contact Technical Support.

Check for ODBC or JDBC errors

Check for the following types of errors:

- ODBC errors returned by the **SQLError** function
- JDBC errors returned by the **SQLException** class

The application may have encountered a problem that caused it to exit prematurely, which in turn may have caused other connections to be forced to disconnect. Call **SQLError** after each ODBC call to identify error or warning conditions when they first happen. Examples of **SQLError** usage can be found in the demo programs and in "Retrieving errors and warnings" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

In more extreme cases, it may be helpful to use **ttTraceMon** to generate a level 4 ERR trace for the application and review all of the errors messages that are pushed in the TimesTen direct driver. See "ERR tracing" on page 22 for details.

Check the user error log

If a TimesTen application disconnects without returning an ODBC error or any other warning, look through the user error log. See "Using the logs generated by the TimesTen daemon" on page 15.

Application is slow

For details on how to maximize the performance of your application and TimesTen data store, see:

- "Data Store Performance Tuning" in *Oracle TimesTen In-Memory Database Operations Guide*
- "Application Tuning" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide*
- "Application Tuning" in *Oracle TimesTen In-Memory Database Java Developer's and Reference Guide*

This section describes some of the issues that impair performance.

Possible cause	See...
Using client/server mode	"Consider connection mode" on page 52
Outdated database statistics	"Update statistics for your tables" on page 52
Committing transactions too frequently	• "Turn off autocommit mode" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide* • "Turn off autocommit mode" in *Oracle TimesTen In-Memory Database Java Developer's and Reference Guide*
DurableCommits attribute enabled	"Use durable commits appropriately" in *Oracle TimesTen In-Memory Database Operations Guide*
Not preparing SQL statements used more than once	• "Prepare statements in advance" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide* • "Prepare statements in advance" in *Oracle TimesTen In-Memory Database Java Developer's and Reference Guide*
Wrong kind of index, too many indexes, wrong size for hash index	• "Select hash or T-tree indexes appropriately" in *Oracle TimesTen In-Memory Database Operations Guide* • "Size hash indexes appropriately" in *Oracle TimesTen In-Memory Database Operations Guide*
Inefficient use of locks	"Verify lock and isolation levels" on page 53

Possible cause	See...
Improperly configured materialized view	"Improving performance of materialized views" in *Oracle TimesTen In-Memory Database Operations Guide*
If replication is used, configuration of replication scheme or network environment may be impacting application.	"Poor replication or XLA performance" on page 132
If Cache Connect is used, Cache Connect configuration or environment may be impacting application.	"Poor autorefresh performance" on page 101
Too many table partitions	"Check partition counts for the tables" on page 54
Tracing is unnecessarily enabled for one or more TimesTen components.	"Check trace settings" on page 53

Consider connection mode

Client/server connections are slower than direct connections to TimesTen data stores. Driver manager connections can also moderately impact performance. The performance overhead imposed by client/server connections can be significant because of the network latencies involved in all communication with the data store.

If your application must run on a different machine from the one hosting the data store, see "Client/Server tuning" in *Oracle TimesTen In-Memory Database Operations Guide*.

Update statistics for your tables

The TimesTen query optimizer in general is very good at choosing the most efficient query plan. However, it needs additional information about the tables involved in complex queries in order to choose the best plan. By knowing the number of rows and data distributions of column values for a table, the optimizer has a much better chance of choosing an efficient query plan to access that table.

Before preparing queries that will access a TimesTen table, use the **ttOptUpdateStats** procedure to update the statistics for that table. When updating the statistics for a table, you will get the best results if you update statistics on your tables *after* loading them with data, but *before* preparing your queries. For example, if you update statistics on a table before populating it with data, then your queries are optimized with the assumption that the tables contain no rows (or very few). If you later populate your tables with millions of rows and

then execute the queries, the plans that worked well for the situation where your tables contained few rows may now be very slow.

For more information about updating statistics, see "The TimesTen Query Optimizer" in *Oracle TimesTen In-Memory Database Operations Guide*.

Verify lock and isolation levels

The manner in which multiple applications concurrently access the data store can have a major impact on performance.

An application can acquire locks on the entire data store, individual tables, and individual rows. Additionally, applications can set an isolation level that determines whether they hold read and update locks until their transactions commit or roll back.

Check the SYS.MONITOR table or use the **ttXactAdmin** utility to detect whether an application is spending time waiting for locks. See "Check for deadlocks and timeouts" on page 55 and "Using the ttXactAdmin utility" on page 31.

If lock contention is high, you may be able to improve the overall performance of your system by implementing the following:

- Set the **LockLevel** configuration attribute or use the **ttLockLevel** procedure to place locks on rows, rather than on the entire data store. (Row locking is the default.)
- Use the **ttOptSetFlag** procedure to prevent the query optimizer from placing locks on tables. (Table locks are sometimes the default, particularly for updates that affect many rows.)
- Use read-committed isolation level (**Isolation**=1, the default) for those applications do not require serializable access to the transaction data.

If you see a lot of lock contention, but the above settings are all set to minimize contention, then the contention may be related to the application itself. For example, concurrent threads may be repeatedly accessing the same row. The **ttXactAdmin** utility can sometimes help you detect this sort of contention. Tracing can also be useful in this situation.

For more information about locks and isolation levels, see "Concurrency control" in *Oracle TimesTen In-Memory Database Operations Guide*.

Check trace settings

Use **ttTraceMon -e show** as described in "Using the ttTraceMon utility" on page 16 to confirm tracing is off on all TimesTen components. ERR should be set to 1; all other components should be set to 0. Trace levels are preserved when a data store is reloaded.

On Windows platforms, confirm that ODBC tracing is disabled. Double-click **ODBC** in the Control Panel to open the ODBC Data Source Administrator. Select the **Tracing** tab and confirm tracing is disabled. See "Using ODBC tracing" on page 33.

Check partition counts for the tables

When a table is created, it has one partition. When you use ALTER TABLE ADD COLUMN to add new columns, a new partition is added to the table. Adding multiple columns with a single ALTER TABLE ADD COLUMN statement only adds one partition.

There is a limit of 255 partitions per table. Exceeding this number generates an 8204 error. However, be aware that there is an extra read for each new partition added to a table that slightly degrades performance for a query on the added columns of that table.

The partition value for each table is tracked in the SYS16 column of the system table, SYS.TABLES. Obtain the partition counts for tables by using the following query:

```
SELECT tblname, sys16 FROM SYS.TABLES;
```

If you discover that a table has too many partitions, either recreate the table or save and restore the table by using **ttMigrate** create (`-c -noRepUpgrade`) followed by **ttRestore** (`-r -noRepUpgrade`). ALTER TABLE DROP COLUMN does not remove partitions from a table.

Application unresponsive, appears hung

This section describes what to check if your application is unresponsive and appears to be hung.

Possible cause	See...
All causes	“Check logs and gather trace information” on page 55
Internal application error	“Check for ODBC errors” on page 55
Inconsistent connection attributes set in DSN	“Consider connection mode” on page 52
Excessive lock contention	“Check for deadlocks and timeouts” on page 55

Check logs and gather trace information

If your application hangs, check the transaction log by using the **ttXactAdmin** utility. See “Using the ttXactAdmin utility” on page 31.

Also check the user error log for errors, as described in “Using the logs generated by the TimesTen daemon” on page 15.

You can also generate a trace log to detect the activities on various TimesTen components as described in “Using the ttTraceMon utility” on page 16.

Check for ODBC errors

Check the ODBC errors returned by the **SQLError** function in all applications to determine whether one of them has encountered a problem that caused it to hang. Call **SQLError** after each ODBC call to identify error or warning conditions when they first happen. Examples of **SQLError** usage can be found in the demo programs and in "Retrieving errors and warnings" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

If the problem is repeatable, use **ttTraceMon** to generate a SQL trace to determine where the application is hanging. See “SQL tracing” on page 18 for details. In more extreme cases, it may be helpful to generate a level 4 ERR trace for the application and review all of the errors messages that are pushed in the TimesTen direct driver. See “ERR tracing” on page 22 for details.

Check for deadlocks and timeouts

If there is no connect problem, a deadlock or timeout may be the problem. The SYS.MONITOR table records information about deadlocks and timeouts. See “Monitoring the TimesTen system tables” on page 35 for information on how view the contents of this table. You can also use the **ttXactAdmin** utility to

detect the types of locks currently held by uncommitted transactions and the resources on which they are being held.

If a deadlock occurs, the TimesTen subdaemon negotiates the problem by having an application involved in the deadlock generate TimesTen error 6002 (Lock request denied because of deadlock). The error message contains the SQL that the lock holder is running, which can help you diagnose the cause of the deadlock. If your application encounters this error, it should roll back the transaction and then reissue the statements for that transaction. Deadlocks can be caused if your application issues statements in a particular order that results in a circular wait, and can sometimes be prevented by changing the order in which the statements are issued.

An application encounters TimesTen error 6003 (Lock request denied because of timeout) if it is unable to acquire a lock within the time period defined by the lock timeout interval set by the **LockWait** attribute in the DSN or by the **ttLockWait** procedure in your application. Upon encountering a timeout error, your application can reissue the statement. Keeping transactions short reduces the possibility of lock timeout errors.

System tables are a common source of lock contention. Reduce contention on the system tables by executing prepared statements, rather than executing the same statements directly each time.

In multithreaded applications, a thread that issues requests on different connection handles to the same data store may encounter lock conflict with itself. TimesTen resolves these conflicts with lock timeouts.

Application unable to find previously created objects

This section describes what to check if your application is unable to locate previously created tables, indexes, sequences or views in the data store.

Possible cause	See...
No owner or incorrect owner specified	"Specify object owner" on page 57
Access Control is enabled on the TimesTen data store and user does not have SELECT privileges to tables.	"Check Access Control privilege to access tables" on page 57
Data store is temporary.	"Check Temporary DSN attribute" on page 57
Overwrite attribute is enabled.	"Check Overwrite DSN attribute" on page 58
Path name specified in DSN is relative.	"Check path name to data store" on page 58

Specify object owner

Tables, indexes and sequences can be created either with a single name, such as `PARTS`, or with a qualified name incorporating an owner and table name, such as `STAN.PARTS`. When accessing a table or index, if no owner is specified, TimesTen first assumes that the owner is the login ID of the user (the value of **UID**). If TimesTen cannot find the table or index under the user's login ID, it then assumes that the owner is user SYS.

If applications need to connect to a data store as different users and share objects, explicitly specify the owners of the objects when they are created and referenced.

Check Access Control privilege to access tables

If Access Control is enabled on the data store, use the **ttUserPrivileges** procedure to check that you have 'SELECT' privilege for the tables. If you do not have 'SELECT' privilege for the tables, the administrator must use the GRANT statement to grant you the privilege.

Check Temporary DSN attribute

Temporary data stores (DSN attribute: **Temporary**=1) persist until all connections to the data store have been removed. When attempting to access a table in a temporary data store and the table does not exist, it is possible that the data store in which the table resided in has been dropped.

Check Overwrite DSN attribute

If the **Overwrite** and **AutoCreate** DSN attributes are enabled and the data store already exists, TimesTen drops that data store and creates a new one. Any tables that were created in the old data store are dropped.

Check path name to data store

To ensure that you are always accessing the same data store when connecting to a particular DSN, use an absolute data store path name instead of a relative one. For example, if the *demo* data store is in the *datastore* directory, specify:

```
DataStore=/datastore/demo
```

rather than:

```
DataStore=demo
```

In the latter case, the data store path name is relative to the directory where the application was started. If you are unable to find a table and you are using a relative data store path name, it is possible that the data store in which the table resides in does exist but the data store (checkpoint and log) files are in a different directory than the one that you are accessing.

See "Specify the data store path name" in *Oracle TimesTen In-Memory Database Operations Guide*.

Running out of a resource

This section describes what to check if TimesTen runs out of resources such as memory space, disk space, file descriptors, and semaphores.

Symptom	See...
Memory consumption seems high.	"Operating system tools and shared memory" on page 59
Running out of memory space	• "Check the amount of memory allocated to the data store" on page 59 • "Update query optimizer statistics" on page 61 • "Check memory used by queries" on page 61 • "Check available swap space (virtual memory)" on page 62
Running out of disk space	"Check log file use of disk space" on page 62
Running out of log space	"Check log file use of disk space" on page 62
Running out of file descriptors	"Increase the number of available file descriptors" on page 43
Running out of semaphores	"Check the semaphore limit" on page 64
Running out of CPU	Obtain a stack trace and contact Technical Support.

Operating system tools and shared memory

Operating system tools such as `top`, `vmstat`, and `sar` provide statistics about processes and memory usage. The output from these tools can be misleading as an indicator of TimesTen memory consumption because they report shared memory usage for each process but do not report total shared memory usage. Adding together various memory statistics for TimesTen processes overestimates the amount of memory used by TimesTen because shared memory is by definition shared.

Check the amount of memory allocated to the data store

TimesTen uses both permanent and temporary data partitions. The amount of memory allocated for these partitions is set by the **PermSize** and **TempSize** attributes in the DSN definition for the data store.

When the TimesTen data store fills up, it is important to determine whether it is the permanent or the temporary segment that is filling up. Use the **ttIsql dssize** command to list allocated, in-use, and high water mark sizes for the permanent

and temporary data partitions. The **dssize** command selects the following values from SYS.MONITOR:

- PERM_ALLOCATED_SIZE
- PERM_IN_USE_SIZE
- PERM_IN_USE_HIGH_WATER
- TEMP_ALLOCATED_SIZE
- TEMP_IN_USE_SIZE
- TEMP_IN_USE_HIGH_WATER

The permanent segment consists of table and index data, while the temporary segment consists of internal structures, such as locks, sorting areas, and compiled commands.

Keeping transactions short and making sure there is enough temporary space in the data store prevents locks from occupying all of the remaining temporary space. You can also use table locks if transactions are acquiring tens of thousands of row locks.

For tips on how to estimate the size of your data store, see "Size your data store correctly" in *Oracle TimesTen In-Memory Database Operations Guide*.

Permanent segment filling up

Consider whether you can drop any indexes. You may want to look at query plans to see which indexes are actually used. See "Viewing and changing query optimizer plans" in *Oracle TimesTen In-Memory Database Operations Guide*. You can also use the **ttRedundantIndexCheck** procedure to discover redundant indexes. The procedure returns suggestions about which indexes to drop.

Use the **ttSize** utility to estimate the amount of memory used by each table in the data store. If the amount of data you need to store is too big, you may need to reset the **PermSize** attribute for the data store to increase the size of the permanent segment. Alternatively, you may need to partition your data into several different data stores if, for example, you cannot shrink the temporary segment or create a bigger data store because of limits on the memory segment size.

Sometimes when the permanent segment fills up, copying the data out of the data store, deleting all the data, and copying it back in frees up space. This can be done more efficiently by using the **ttMigrate** utility with the `-noRepUpgrade` option to migrate the data out, destroy and re-create the data store, and migrate the data back in. This operation is described in "Reducing data store size" in *Oracle TimesTen In-Memory Database Installation Guide*.

Finally, you may have to configure the operating system to allow a larger amount of shared memory to be allocated to a process. You may also have to allocate

more swap space for virtual memory. See “Check available swap space (virtual memory)” on page 62.

Temporary segment filling up

Some commands may be allocating too much space because of out-of-date statistics. See “Update query optimizer statistics” on page 61.

If updating the statistics does not reduce temporary segment memory usage, disconnect all connections and then reconnect them. Verify that all connections have been disconnected by using the **ttStatus** utility. That frees up all temporary space, but you must reprepare commands.

Diagnose memory usage by queries. See “Check memory used by queries” on page 61.

If the problem is chronic, monitor the data store to try to identify the source of the problem. Use the **ttWarnOnLowMemory** procedure to enable warnings in the user log that indicate that the data store is filling up.

Update query optimizer statistics

If the data store seems to have enough free space but runs out of data store space when executing a query, make sure you have updated the optimizer statistics with the **ttOptUpdateStats** or **ttOptEstimateStats** procedure. To execute some queries, TimesTen needs to allocate temporary space. The amount of temporary space required is estimated from statistics about the tables used by the query. Without correct statistics, the temporary space required may be underestimated.

See “Using the query optimizer” on page 36.

Check memory used by queries

You can check the memory that a query uses by observing the high water mark for temporary memory usage. The high water mark represents the largest amount of in-use temporary space used since the high water mark was initialized or reset.

Complete the following tasks:

1. Use the **ttIsql dssize** command to check TEMP_IN_USE_SIZE and TEMP_IN_USE_HIGH_WATER. (Alternatively, you can query the SYS.MONITOR table for these values.)
2. Call the **ttMonitorHighWaterReset** procedure to reset the TEMP_IN_USE_HIGH_WATER to the current value for TEMP_IN_USE_SIZE.
3. Execute a query.
4. Use **dssize** to check TEMP_IN_USE_HIGH_WATER for peak memory usage for the query.

Check available swap space (virtual memory)

If you receive an error indicating that you have run out of swap space, you may need to increase the amount of available swap space (also referred to as “virtual memory”).

On UNIX systems, use the **swap** command to check and reset the amount of virtual memory currently established for your system.

On Windows systems, check and reset the size of your virtual memory by choosing **Control Panel** > **System** > **Advanced**.

Check log file use of disk space

TimesTen saves a copy of the data store in one of two checkpoint files, which are stored in the directory specified by the **DataStore** attribute. Each checkpoint file can grow on disk to be equivalent to the size of the data store in shared memory. For each permanent data store, you must have enough disk space for the two checkpoint files and for log files (assuming logging to disk is enabled).

Log files accumulate in the directory specified by the **LogDir** attribute and are only deleted when checkpoints are performed. If the **LogDir** attribute is not specified in the DSN, log files accumulate in the directory specified by the **DataStore** attribute. The maximum size of your log files is set by the **LogFileSize** attribute.

When a disk fills up with TimesTen data, it is most often due to a build-up of log files. Log files are used for numerous purposes in TimesTen, including checkpointing, backups, and replication. It is important to determine which operation is putting a “hold” on the log files, so that appropriate action can be taken to allow the log files to be purged. This can be done by using the **ttLogHolds** built-in procedure. There are six types of log holds. They are discussed in detail below.

- **Checkpoint** — If a TimesTen application crashes and the data store needs to be recovered, the checkpoint files and log files are used to recover the data. The “most recent” log files are used -- those written since the checkpoint was done. Log files accumulate during the interval between checkpoints. Your application should periodically call the **ttCkpt** or **ttCkptBlocking** procedure to checkpoint the data and free up the space on the disk. If checkpoints are done very infrequently, a large number of log files may accumulate, particularly if many changes are made to the data store during that interval. See "Checkpoints" in *Oracle TimesTen In-Memory Database Operations Guide*.
- **Replication** — TimesTen replication transmits changes to one data store to one or more other data stores. It does this by reading the log and sending any relevant changes. If replication is paused, the log files build up. To prevent log build-up, avoid pausing replication for too long. Delete subscriptions entirely, and reset replication where appropriate. See "Setting the replication state of

subscribers" in *TimesTen to TimesTen Replication Guide* for more information on pausing and restarting or resetting replication.

- **Backup** — TimesTen supports an incremental backup facility that uses log files to augment a backup with changes made since the last backup. Log files accumulate during the interval between incremental backups. To avoid a large log build-up, do incremental backups at relatively frequent intervals. If desired, disable incremental backups and do full backups instead. See "Copying, migrating, backing up and restoring a data store" in *Oracle TimesTen In-Memory Database Operations Guide*.
- **XLA** — TimesTen's persistent XLA facility reports changes to the data store by using log files. Log files are kept until the corresponding transactions have been acknowledged using the **ttXlaAcknowledge** function. Call **ttXlaAcknowledge** frequently enough to prevent log files building up. See "Retrieving update records from the transaction log" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide*.
- **XA** — TimesTen's XA support uses log files to resolve distributed transactions. If these transactions are not resolved in a timely manner, log files build up. See "Distributed Transaction Processing XA" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide*.
- **Long-Running Transactions** — TimesTen uses the transaction log to roll back transactions. A log hold is placed for the duration of a transaction. Transactions that are active for a long time result in log file building up if the transaction has written at least one log record. (That is, it is not a read-only transaction.) Commit write transactions with reasonable frequency to avoid significant log file build-up. See "Size transactions appropriately" in *Oracle TimesTen In-Memory Database C Developer's and Reference Guide* for more information on transaction length.

The following attributes are related to disk use:

- The **LogPurge** attribute indicates whether log files that no longer have a hold on them are purged (removed from the disk) or simply archived (renamed). If the **LogPurge** attribute is set to the default value of 0, TimesTen renames log files that it no longer needs by appending the string `.arch` to the name. Once renamed, you must delete the log files manually when they are no longer needed. If log files are not purged, they continue to accumulate space, even when no longer needed by TimesTen.
- The **Preallocate** attribute indicates whether disk space should be reserved for checkpoint files at connect time. This is useful for big data stores, to ensure that the disk always has room for the checkpoint files as data is added to the data store.

Check the semaphore limit

When creating multiple client/server connections to a TimesTen data store configured to allow shared memory segment as IPC, you may encounter errors that indicate TimesTen was unable to create a semaphore.

Semaphore limits are platform-dependent. See your operating system documentation and "Increase number of semaphores" in *Oracle TimesTen In-Memory Database Installation Guide*.

Duplicate results from a SELECT statement

Using read committed isolation level can lead to duplicates in a result set. A SELECT statement selects more or fewer rows than the total number of rows in the table if some rows are added or removed and committed in the range in which the SELECT scan is occurring. This may happen when an UPDATE, INSERT or DELETE statement adds or deletes a value from an index and the SELECT scan is using this index. This can also happen when an INSERT or DELETE adds or deletes rows from the table and the SELECT operation is using an all-table scan.

Index values are ordered. An UPDATE of an index value may delete the old value and insert the new value into a different place. In other words it moves a row from one position in the index to another position. If an index scan sees the same row in both positions, it returns the row twice. This does not happen with a serial scan because table pages are unordered and rows do not need to be moved around for an UPDATE. Hence once a scan passes a row, it will not see that same row again.

The only general way to avoid this problem is for the SELECT statement to use serializable isolation. This prevents a concurrent INSERT, DELETE or UPDATE operation. There is no reliable way to avoid this problem with INSERT or DELETE by forcing the use of an index because these operations affect all indexes. With UPDATE, this problem can be avoided by forcing the SELECT statement to use an index that is not being updated.

For more information about serializable isolation, see "Concurrency control" in *Oracle TimesTen In-Memory Database Operations Guide*.

3

Troubleshooting Installation, Upgrades and Downgrades

This chapter includes the following topics:

Installing 32-bit TimesTen on 64-bit Windows

The default ODBC Data Source Administrator on 64-bit Windows does not show TimesTen 32-bit drivers and DSNs. If Windows is installed in the default location (`C:\WINDOWS`), use `C:\WINDOWS\SysWOW64\odbcad32.exe` for the ODBC Data Source Administrator when you are installing 32-bit TimesTen on a 64-bit Windows machine.

Downgrading a data store with Oracle data types to TimesTen 6.0

In rare situations, after upgrading a data store from TimesTen 6.0, you may find that you need to downgrade a TimesTen 7.0 data store back to TimesTen 6.0 after the data types are already converted to Oracle types. However, the ttMigrate utility for TimesTen 6.0 does not understand Oracle data types, and this can lead to problems when downgrading data stores from TimesTen 7.0. To avoid any pitfalls in the downgrade process, you should convert the Oracle data types back to TimesTen types using TimesTen 7.0 first, and only then downgrade the data store to TimesTen 6.0, using the following steps:

1. Create a migration file using TimesTen 7.0 **ttMigrate**.

   ```
   ttMigrate -c datastore datastore.migrate
   ```

2. Destroy the data store using TimesTen 7.0 **ttDestroy**.

   ```
   ttDestroy datastore
   ```

3. Convert the data types to TimesTen types using TimesTen 7.0 **ttMigrate**.

   ```
   ttMigrate -r -noRepUpgrade -convertTypesToTT datastore
      datastore.migrate
   ```

4. Create a new migration file using TimesTen 7.0 **ttMigrate**.

   ```
   ttMigrate -c datastore datastore.migrate
   ```

5. Destroy the data store using TimesTen 7.0 **ttDestroy**.

   ```
   ttDestroy datastore
   ```

6. In another terminal, with the environment set correctly for TimesTen 6.0, restore the data store as a TimesTen 6.0 data store using TimesTen 6.0 **ttMigrate**.

   ```
   ttMigrate -r datastore datastore.migrate
   ```

Note: Before restoring the data store with TimesTen 6.0 **ttMigrate**, you must modify the DSN attributes appropriately for using with TimesTen 6.0.

4

Troubleshooting Cache Connect to Oracle

This chapter describes how to troubleshoot some of the problems you may encounter when using Cache Connect to Oracle. It includes the following topics:

If you are having problems with an AWT cache group, see also Chapter 5, “Troubleshooting AWT Cache Groups.”

Unable to create a cache group

This section describes some of the problems you might encounter when executing the CREATE CACHE GROUP statement.

Possible cause	What to do
User does not have the correct Oracle privileges to create the cache group type.	See "Check Oracle privileges" on page 79.
Access Control is enabled on the TimesTen data store and user has insufficient access to data store.	If Access Control is enabled on the data store, you must have DDL privileges to create a cache group.
Access Control is enabled on the TimesTen data store and the internal/external user does not match the Oracle user.	If Access Control is enabled on the data store, the TimesTen user name must be the same as the Oracle user name.
Cannot connect to Oracle	See: • "Unable to resolve Oracle Service Name" on page 75 • "Unable to resolve connect identifier" on page 76 • "Unable to validate Oracle username and password" on page 78 • "Incompatible Oracle server and client versions" on page 77 Check whether Oracle needs to be restarted. Check the network status.
Cache administration user ID or password not set (when trying to create AWT or autorefresh cache groups)	See "Set the cache administration user ID and password" on page 80.
Unsupported data type mapping	See "Unsupported data type mapping" on page 83.
Different nullability setting in Oracle	See "NULL constraint does not match Oracle" on page 84.
Failure to specify primary key in root table	The root table of a cache group must have a primary key. See "Defining cache group tables" in *TimesTen Cache Connect to Oracle Guide*.

Unable to start or stop the cache agent

Possible cause	What to do
Cache agent already running	See "Check status of the cache agent" on page 73.
Unable to locate Oracle libraries	• See "Check library path environment variable" on page 78. • Check the permissions on the libraries.
ORACLE_HOME is invalid.	See "Check ORACLE_HOME environment variable" on page 74.
Wrong version of Oracle client libraries	32-bit TimesTen must use 32-bit Oracle client libraries. 64-bit timesTen must use 64-bit Oracle client libraries.
Access Control is enabled on the TimesTen data store.	If Access Control is enabled on the data store, you must have ADMIN privileges to start or stop the cache agent.
Wrong **OracleID**	Ensure that the **OracleID** set in your DSN definition matches the Oracle Service Name for the Oracle instance that contains the tables to cache in TimesTen.

Check status of the cache agent

Check the status of the cache agent by using the **ttStatus** utility as described in "Using the ttStatus utility" on page 10 to check the status of the cache agent.

If the cache agent is not running, start it as described in "Starting and stopping the cache agent" in *TimesTen Cache Connect to Oracle Guide*. If attempts to start the cache agent fail, then investigate the possible causes and reboot the machine before attempting to start the cache agent.

Check the Oracle Database version

Check that the correct version of the Oracle Database is installed on the machine.

If you are on an AIX machine, check that:

- The link *install_dir*`/lib/libttor.so` points to:
  - *install_dir*`/lib/libttor_9i.so` for Oracle9i
  - *install_dir*`/lib/libttor_10g.so` for Oracle Database 10g
- The link *install_dir*`/bin/timestenorad` points to:
  - *install_dir*`/bin/timestenorad_9i` for Oracle9i
  - *install_dir*`/bin/timestenorad_10g` for Oracle Database 10g

Check ORACLE_HOME environment variable

On UNIX or Linux platforms, check that the ORACLE_HOME environment variable is set correctly for the shell from which you are starting the cache agent and the TimesTen daemon. Use the **ttmodinstall** utility if you need to change the setting for ORACLE_HOME.

See "ORACLE_HOME environment variable" in *Oracle TimesTen In-Memory Database Installation Guide*.

Unable to resolve Oracle Service Name

If you receive error ORA-12514 indicating "could not resolve service name":

- Use the Oracle TNSPING utility to verify that the service can be reached.
- Ensure that the **OracleID** set in your DSN definition matches the Oracle Service Name for the Oracle instance that contains the tables to cache in TimesTen.
- Ensure that there is a service name defined. If it is a Windows Oracle client, use Oracle Net Configuration Assistant to configure a service name. In **Oracle Net Configuration Assistant**, navigate to **Oracle Net Configuration** -> **Local** -> **Service Naming**, select your Oracle server and confirm that there is a service name or a SID that identifies the Oracle server. If you add or modify a service name, you may need to reboot.

  Check the cache administration user name and password on Oracle with SQL*Plus to make sure this service name works. For example:

  ```
  %sqlplus scott/tiger@OracleHost
  ```

  scott is the cache administration user name, *tiger* is the cache administration user password, and *OracleHost* is the **OracleID** specified in your DSN definition.

Note: Your cache administration user may be different from your regular Oracle user. See "Create Oracle users and set privileges" in *TimesTen Cache Connect to Oracle Guide*.

- Ensure that there is only one copy of `tnsnames.ora` on your TimesTen machine. Also check the permission on `tnsnames.ora`.
- If you are running TimesTen on a UNIX system, check that the ORACLE_HOME environment variable points to the correct Oracle installation directory. For example:

  ```
  ORACLE_HOME=/products/oracle10g
  ```

- Check the Oracle client and server versions. See "Incompatible Oracle server and client versions" on page 77.

Unable to resolve connect identifier

You may receive ORA-12154 “TNS:could not resolve the connect identifier specified” when you try to connect to a a data store.

This can occur when you are trying to use Cache Connect and Oracle on the same machine and the TNS_ADMIN environment variable does not point to the proper `tnsnames.ora` file for Oracle. For example, you may have several instances of the Oracle Database running on a laptop.

In a production environment, you typically have TimesTen and Oracle running on different machines. In this case, do not reset the TNS_ADMIN environment variable to point to a `tnsnames.ora` file on the machine where TimesTen is running. The Oracle client uses the TNS_ADMIN setting to resolve the connection, but the TimesTen main daemon, the cache agent, the Web server, and the replication agent are unaware of the TNS_ADMIN setting. Cache Connect cannot operate properly when the Oracle client and TimesTen use different `tnsnames.ora` files.

On Windows, set the TNS_ADMIN environment variable as follows:

1. Right-click My Computer and choose Properties.
2. On the Advanced tab, choose Environment Variables.
3. Add or edit TNS_ADMIN as a system environment variable so that it points to the directory that contains the `tnsnames.ora` file that you wish to use. You can include other `tnsnames.ora` files with the INAME command inside the `tnsnames.ora` file.

Incompatible Oracle server and client versions

If you receive connection timeout errors such as ORA-12170 or ORA-12535, or if you receive ORA-03134 (server version not supported), verify that you are using an Oracle client and Oracle server whose versions are compatible.

Metalink Documentation Note 207303.1, “Client/Server/Interoperability Support Between Different Oracle Versions”, lists the client/server combinations supported by Oracle.

See “Cache Connect to Oracle” in *Oracle TimesTen In-Memory Database Installation Guide* for information about Oracle clients and servers supported for use with TimesTen. Also check Oracle and TimesTen release notes for known problems with client/server versions.

Unable to validate Oracle username and password

Possible cause	See...
The library environment variable is not set correctly	"Check library path environment variable" on page 78.
Oracle processes not running	"Check status of TNS listener and Oracle server" on page 79.
User does not have the correct Oracle privileges	"Check Oracle privileges" on page 79.
Incorrectly configured DSN	"Check DSN definition" on page 79.
Problems with cache administration user ID or password	"Set the cache administration user ID and password" on page 80.
Inconsistent user and system environments	"Check user and system environment" on page 80.
Dynamic libraries not loading	"Verify the loaded dynamic libraries" on page 80.

Check library path environment variable

Check the library path environment variable:

On this platform...	Check this variable...
UNIX except HP-UX	LD_LIBRARY_PATH On 64-bit platforms, LD_LIBRARY_PATH64 takes precedence over LD_LIBRARY_PATH. Make sure that the library path is specified in LD_LIBRARY_PATH64.
HP-UX	SHLIB_PATH
Windows	PATH

The library path environment variable must include the following information:

TimesTen and Platform Bit Combination	Setting
64-bit TimesTen or 32-bit TimesTen on 32-bit platform	$ORACLE_HOME/LIB and $ORACLE_HOME/NETWORK/LIB
32-bit TimesTen on 64-bit platform	$ORACLE_HOME/LIB32 and $ORACLE_HOME/NETWORK/LIB32

Check status of TNS listener and Oracle server

Try to connect to the Oracle database by using SQL*Plus or use Oracle Enterprise Manager to verify the status.

Check Oracle privileges

From an Oracle SQL*Plus command prompt, list the current Oracle privileges granted to you by entering:

```
SELECT * FROM SESSION_ROLES;
SELECT * FROM SESSION_PRIVS;
```

Compare the privileges listed against the required privileges for the various Cache Connect to Oracle operations that are specified in "Create Oracle users and set privileges" in *TimesTen Cache Connect to Oracle Guide*. Contact your Oracle Administrator if you require additional privileges.

Check DSN definition

- Confirm you have correctly set the DSN attributes as described in "Defining DSNs for cached tables" in *TimesTen Cache Connect to Oracle Guide*.
- Confirm that the DSN definition for Cache Connect to Oracle is a *system* DSN.
- Confirm that the DSN for Cache Connect to Oracle is defined only once.
- Confirm Oracle user name and password. Use SQLPlus and connect to Oracle using the same **OracleID** and **OraclePWD** used in your DSN definition to confirm they are correct.

Reboot TimesTen machine

If the Oracle client was installed and the machine has not been restarted, then the TimesTen daemon is still running under the "old" environment before the Oracle client install. Reboot your machine so the TimesTen can start under the "new" environment.

Set the cache administration user ID and password

From a ttIsql session, connect to the data store and enter the following:

```
Command> call ttCacheUidPwdSet('scott','tiger');
```

If it returns an error, then check the Oracle ID, the cache administration user ID and cache administration password. Also check whether the Oracle instance is running.

Check user and system environment

Test to see if the problem is due to differences in user and system environment. This procedure requires two session windows (Command Prompt windows in Windows or shell windows in UNIX).

1. Stop the TimesTen daemon.
2. In one session window, start the Timesten daemon as a regular user.

   On Windows:

   ```
   % install_dir/srv/ttsrv70.exe -d -verbose
   ```

   On UNIX:

   ```
   % install_dir/srv/timestend -d verbose
   ```

   Some messages will flash by, and then it goes into a wait state.
3. In another session window, try to restart the cache agent.
4. If Step 3. succeeds, then use Ctrl-C on Windows or the `kill` command on UNIX to stop the TimesTen daemon you started for the other session in Step 2.
5. Compare the user environment and system environment. For example, do both user and system see the same copy of `oci.dll`? Are there any differences in the pathname to the `oci.dll` library between the user and system environments?
6. If you detect differences, make the necessary modifications.
7. Reboot the system and restart the TimesTen daemon.

Verify the loaded dynamic libraries

If you are running on a Windows system with Virtual C++ installed, verify the loaded dynamic libraries. *This works only if you can start the cache agent without autorefresh*:

1. Make sure TimesTen is started.
2. Start the cache agent without autorefresh.

   ```
   Command> call ttCacheStart;
   Command> create cache group cg1 from t1(c1 int not null primary
   key);
   ```

3. Open the Windows **Task Manager**, find process `ttora70.exe` and highlight it. Right-click on it and select Debug. This brings you into Virtual C++ and you should see the loaded dll in the debug window, as described in "Unable to resolve Oracle Service Name" on page 75.
4. Load the cache group to force an cache connection from the cache agent:

   ```
   Command> load cache group cg1 commit every 100 rows;
   ```

5. Compare the loaded dll in your debug window with the partial list shown in Example 4.1.

Example 4.1 This partial list was created with the Oracle 10.2.0.1.0 client.

```
Loaded symbols for 'C:\tt70\bin\ttora70.exe'
Loaded 'C:\WINDOWS\system32\ntdll.dll', no matching symbolic
information found.
Loaded 'C:\WINDOWS\system32\kernel32.dll', no matching symbolic
information found.
Loaded symbols for 'C:\tt70\bin\tten70d.dll'
Loaded symbols for 'C:\tt70\bin\ttco70d.dll'
Loaded 'C:\WINDOWS\system32\wsock32.dll', no matching symbolic
information found.
Loaded 'C:\WINDOWS\system32\ws2_32.dll', no matching symbolic
information found.
Loaded 'C:\WINDOWS\system32\msvcrt.dll', no matching symbolic
information found.
Loaded 'C:\WINDOWS\system32\ws2help.dll', no matching symbolic
information found.
Loaded 'C:\WINDOWS\system32\advapi32.dll', no matching symbolic
information found.
...
```

OCI initialization failed

Error 5105, "OCI initialization failed," may occur when an operation requires contact with the Oracle database. For example, the error can occur in the following situations:

- Starting the cache agent
- Setting the cache administration user ID or password
- Entering a SQL statement in TimesTen when autocommit=0 and **PassThrough**=3

Error 5105 contains additional information about its cause:

- OCI is unable to find an Oracle library. See "Check library path environment variable" on page 78 and check the permissions on the library specified in the error message.
- ORACLE_HOME is invalid. See "Check ORACLE_HOME environment variable" on page 74.

Unsupported data type mapping

When you try to create a cache group, you may receive the following error:

```
5115: Unsupported type mapping for column name
```

For example, table *tab* on Oracle can be described as follows:

```
COL1      NUMBER(38) NOT NULL
COL2      NUMBER(38)
```

Try to create the cache group as follows:

```
CREATE CACHE GROUP cg FROM tab(col1 CHAR(10) NOT NULL PRIMARY KEY);
```

Error 5119 is displayed and the cache group is not created because the statement attempts to map a column of NUMBER data type to a column of CHAR data type.

See "Data type mappings for Cache Connect to Oracle" in *TimesTen Cache Connect to Oracle Guide*.

NULL constraint does not match Oracle

When you try to create a cache group, you may receive the following warning:

```
Warning 5119: Column name has different nullability setting in Oracle
```

For example, table *tab* on Oracle can be described as follows:

```
COL1      NUMBER(38) NOT NULL
COL2      NUMBER(38)
```

Try to create the cache group as follows:

```
CREATE CACHE GROUP cg
     FROM tab(col1 INTEGER NOT NULL PRIMARY KEY,
              col2 INTEGER NOT NULL);
```

Warning 5119 is displayed because *col2* on Oracle does not have a NULL constraint, but *col2* in the cache group is defined as NOT NULL.

Loading or refreshing fails

If the LOAD CACHE GROUP or REFRESH CACHE GROUP statement fails when you specify COMMIT EVERY *n* ROWS and *n* is greater than 0, the contents of the target cache group could be in an inconsistent state. Some cache instances may be partially loaded.

Unload the cache group and then load it again. In some situations, it may be easier to drop and re-create the cache group.

Monitoring autorefresh cache groups

This section includes the following topics:

- Using the ttCacheAutorefreshStatsGet procedure
- Displaying information from the change log tables
- Understanding messages about autorefresh in the support log
- Diagnosing autorefresh failure
- Diagnosing autorefresh performance problems
- Using SNMP traps for alerts about autorefresh problems

Using the ttCacheAutorefreshStatsGet procedure

The **ttCacheAutorefreshStatsGet** procedure returns information about the last ten autorefresh operations on a specified cache group.

ttCacheAutorefreshStatsGet returns information only when the cache agent is running and the autorefresh state is ON or PAUSED. All of the return fields are set to 0 when the cache agent is restarted or the autorefresh state is changed to OFF.

Example 4.2 *testcache* is a READONLY cache group with one table and an incremental autorefresh interval of 10 seconds.

```
Command> call ttcacheautorefreshstatsget('user1','testcache');

< 1164260, 2007-07-23 15:43:52.000000, 850280, 44, 0, 75464, 528255, 75464, 310,
110, 6800, 1890912, 12439795, 1890912, 160020, InProgress >
< 1164260, 2007-07-23 15:43:33.000000, 831700, 43, 13550, 108544, 759808,
108544, 1030, 230, 12290, 1815448, 11911540, 1815448, 160020, Complete >
< 1164260, 2007-07-23 15:43:12.000000, 810230, 42, 17040, 115712, 809984,
115712, 610, 330, 16090, 1706904, 11151732, 1706904, 146470, Complete >
< 1164260, 2007-07-23 15:42:52.000000, 790190, 41, 14300, 94208, 659456,
94208,560, 320, 13410, 1591192, 10341748, 1591192, 129430, Complete >
< 1164260, 2007-07-23 15:42:32.000000, 770180, 40, 12080, 99328, 695296,
99328,450, 290, 11340, 1496984, 9682292, 1496984, 115130, Complete >
< 1164260, 2007-07-23 15:42:12.000000, 750130, 39, 10380, 86016, 598368,
86016,430, 230, 9720, 1397656, 8986996, 1397656, 103050, Complete >
< 1164260, 2007-07-23 15:41:52.000000, 730130, 38, 13530, 112640, 700768,
112640, 530, 220, 12780, 1311640, 8388628, 1311640, 92670, Complete >
< 1164260, 2007-07-23 15:41:32.000000, 710120, 37, 9370, 56320, 326810, 56320,
310, 160, 8900, 1199000, 7687860, 1199000, 79140, Complete >
< 1164260, 2007-07-23 15:41:22.000000, 700120, 36, 2120, 10240, 50330, 10240,
50, 200, 1870, 1142680, 7361050, 1142680, 69770, Complete >
< 1164260, 2007-07-23 15:41:12.000000, 690110, 35, 0, 0, 0, 0, 0, 0, 0, 1132440,
7310720, 1132440, 67650, Complete >
10 rows found.
```

Table 4.1 describes the results from the first row of output.

Table 4.1 ttCacheAutorefreshStatsGet results from last autorefresh operation

Result	Field name	Description
`1164260`	*cgId*	Cache group ID
`2007-07-23 15:43:52.000000`	*startTimestamp*	Timestamp when autorefresh started for this interval
`850280`	*cacheAgentUpTime*	The start time for the autorefresh interval, expressed in number of milliseconds since the cache agent started
`44`	*autorefNumber*	Autorefresh number
`0`	*autorefDuration*	The number of milliseconds spent in this autorefresh operation. It is zero because the operations is in progress.
`75464`	*autorefNumRows*	The number of rows autorefreshed in this autorefresh operation. This would include all rows in the root table and child tables if the cache group had child tables. **Note:** This information is not provided for full autorefresh.
`528255`	*numOracleBytes*	The number of bytes transferred from Oracle in this autorefresh operation. **Note:** This information is not provided for full autorefresh.
`75464`	*autorefNumRootTblRows*	The number of root table rows autorefreshed in this autorefresh operation.
`310`	*autorefQueryExecDuration*	The duration in milliseconds for the autorefresh query to execute on Oracle. **Note:** This information is not provided for full autorefresh.

Result	Field name	Description
110	*autorefQueryFetchDuration*	The duration in milliseconds for the autorefresh query to fetch rows from Oracle. **Note:** This information is not provided for full autorefresh.
6800	*autorefTtApplyDuration*	The duration in milliseconds for TimesTen to apply the updated rows to the cache group. **Note:** This information is not provided for full autorefresh.
1890912	*totalNumRows*	The total number of rows autorefreshed since the cache agent started. **Note:** This information is not provided for full autorefresh.
12439795	*totalNumOracleBytes*	The total number of bytes transferred from Oracle since the cache agent started. **Note:** This information is not provided for full autorefresh. **Note:** This information is not provided for full autorefresh. **Note:** This information is not provided for full autorefresh.
1890912	*totalNumRootTblRows*	The total number of root table rows autorefreshed since the cache agent started.

Result	Field name	Description
160020	*totalDuration*	The total autorefresh duration in milliseconds since the cache agent started.
InProgress	*autorefreshStatus*	Status. The status can also be **Complete** or **Failed**.

Note that the total number of autorefreshed rows (1890912) is the same as the total number of autorefreshed root table rows in this example because there are no child tables.

The number of autorefreshed rows in TimesTen does not necessarily reflect the number of rows updated on Oracle. The Oracle updates may be applied in TimesTen more than once, or multiple Oracle updates on the same row may be applied as one update in TimesTen.

Displaying information from the change log tables

TimesTen provides a SQL script that gathers information from the change log tables that exist on the Oracle database for autorefresh cache groups. See "Managing Oracle objects for READONLY, AUTOREFRESH, and AWT cache groups" in *TimesTen Cache Connect to Oracle Guide* for more information about change log tables.

The script displays the following information for each cached table:

```
**************************
 * Host name: my-pc
 * Timesten datastore name: c:\data\tt70
 * Cache table name: USER1.TESTCACHE
 * Change log table name: tt_03_55555_L
 * number of rows in change log table: 100000
 * Maximum logseq on the change log table: 38
 * Timesten has autorefreshed updates up to logseq: 38
 * Number of updates waiting to be autorefreshed: 0
 * Number of updates that has not been marked with a valid
logseq: 0
**************************
```

The log sequence number (`logseq`) acts as a marker for the autorefresh operation.

Run the script as the cache administration user on the Oracle database using SQL*Plus. If you run the script as a different user, it reports that the change log tables do not exist.

The script is in the following location:

install_dir/bin/autorefreshChangeLogInfo.sql

Understanding messages about autorefresh in the support log

The support log contains messages that show the progress of autorefresh. For example, *testcache* is a READONLY cache group with an autorefresh interval of 10 seconds (10,000 milliseconds).

The support log shows when autorefresh starts:

```
15:43:33.96 Info: ORA:  5264: ora-5264-5676-refresh03918: Starting
autorefresh number 43 for interval 10000ms
```

The message includes the following information:

- Timestamp (`15:43:33.96`)
- Cache agent process ID (`5264`)
- Thread ID (`5676`)

The thread ID is important because autorefresh numbers are unique only for a specific interval. Always check both the thread ID and the autorefresh number when you are tracking a specific autorefresh operation.

The support log also contains a longer message that reports information similar to the **ttCacheAutorefreshStatsGet** procedure. 108544 rows were updated in this autorefresh interval, and 1815448 rows have been updated since the cache agent was started. Note that the total number of rows and the total number of root table rows are the same in this message because there is only one table in the cache group. *Number* refers to the autorefresh number. All times are expressed in milliseconds.

```
15:43:51.81 Info: ORA: 5264: ora-5264-5676-refresh04387: Cache agent
refreshed cache group USER1.TESTCACHE: Number - 43, Duration - 13550,
NumRows - 108544, NumRootTblRows - 108544, NumOracleBytes - 759808,
queryExecDuration - 230, queryFetchDuration - 1030, ttApplyDuration
- 12290, totalNumRows - 1815448, totalNumRootTblRows - 1815448,
totalNumOracleBytes - 11911540, totalDuration - 160020
```

Additional messages show that the autorefresh operation completes successfully:

```
15:43:51.81 Info: ORA:  5264: ora-5264-5676-refresh04449:
Autorefresh number 43 finished for interval 10000ms successfully
15:43:51.81 Info: ORA:  5264: ora-5264-5676-fresher01619:
Autorefresh number 43 succeeded for interval 10000 milliseconds
```

Inspect the timestamps to determine whether autorefresh is progressing as expected.

See "Managing TimesTen daemon options" in *Oracle TimesTen In-Memory Database Operations Guide* for information about setting the support log location.

Diagnosing autorefresh failure

If **ttCacheAutorefreshStatsGet** shows that the status of an autorefresh operation is **Failed**, check the support log for messages related to the autorefresh operation with number the number shown in the **ttCacheAutorefreshStatsGet** output. Look for errors that occurred after the autorefresh operation started.

Example 4.3 This row of output from **ttCacheAutorefreshStatsGet** shows a failed autorefresh operation.

```
< 1164260, 2007-08-01 14:56:36.000000, 959350, 9, 0, 0, 0, 0, 0,
0, 0, 1, 7, 1, 50, Failed >
```

The autorefresh number is 9.

The support log shows the start message for autorefresh number 9:

```
14:56:36.10 Info: ORA:  5988: ora-5988-4724-refresh03926: Starting
autorefresh number 9 for interval 15000ms
```

The thread ID for autorefresh number 9 is 4724. Look for error messages with this thread ID.

The following messages appear in the support log:

```
14:56:36.10 Info: ORA:  5988: ora-5988-4724-refresh03953:
Autorefresh thread for interval 15000ms is connected to instance inst1
on host host1. Server handle 231976252
14:56:36.12 Err : ORA:  5988: ora-5988-4724-refresh07567:
TimesTen error code:5901, msg The Oracle refresh log table,
"USER2"."TT_03_81799_L", for base table, USER2.READTAB2,
cannot be found.
14:56:36.12 Info: ORA:  5988: ora-5988-4724-refresh05559:
Autorefresh rolled back.
14:56:36.12 Info: ORA:  5988: ora-5988-4724-refresh04458:
Autorefresh number 9 finished for interval 15000ms with error.
14:56:36.12 Err : ORA:  5988: ora-5988-4724-fresher01606:
Autorefresh number 9 failed for cache groups with interval 15000
ms after 10 retries.
```

The error message for thread ID 4724 shows that the change log table, TT_03_81799_L, is missing. The introduction to “Autorefresh not refreshing cache at the specified interval” on page 93 has a table entry that describes what to do in this situation.

Diagnosing autorefresh performance problems

You can use the **ttTraceMon** utility to diagnose autorefresh performance problems. See “AUTOREFRESH tracing” on page 26.

TimesTen tracing severely impacts application performance and consumes a great deal of disk space if trace output is directed to a file. When you are finished, reset tracing to the default values.

Using SNMP traps for alerts about autorefresh problems

Enable SNMP traps to alert you when autorefresh problems occur.The SNMP traps related to autorefresh include:

- **ttCacheAutoRefQueFullTrap**
- **ttCacheIncAutoRefFailedTrap**
- **ttCacheValidationErrorTrap**
- **ttCacheValidationWarnTrap**
- **ttCacheValidationAbortedTrap**

See "Diagnostics through SNMP Traps" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

Autorefresh not refreshing cache at the specified interval

The following table shows possible causes for autorefresh problems.

Possible cause	What to do
Cache agent not started with a cache administration user	Specify a cache administration user ID and password when starting the cache agent, as described in "Starting and stopping the cache agent" in *TimesTen Cache Connect to Oracle Guide*.
Object ID of the base table has changed.	See "Recover and reset autorefresh Oracle objects" on page 94.
Autorefresh trigger not enabled	See "Recover and reset autorefresh Oracle objects" on page 94.
Current log sequence number recorded in the TT_*version*_USER_COUNT table is less than to the maximum log sequence number in the autorefresh log table.	See "Recover and reset autorefresh Oracle objects" on page 94.
There is no row in the TT_*version*_USER_COUNT table with usercount > 0 for every active incrementally autorefresh table	See "Recover and reset autorefresh Oracle objects" on page 94.
Change log table is empty.	See "Recover and reset autorefresh Oracle objects" on page 94.
User count is less than 0 or any TT_*version*_USER_COUNT log sequence anomalies	See "Recover and reset autorefresh Oracle objects" on page 94.
Autorefresh log table, trigger, or sequence associated with a cached table does not exist or is not valid.	Check whether the cache agent was started with the correct cache administration user ID. If the cache administration user ID is correct, follow the procedure described in "Recover and reset autorefresh Oracle objects" on page 94. Check the user error log for messages about "fatal anomalies". This indicates corrupt or missing Oracle objects.

Possible cause	What to do
TT_*version*_USER_COUNT table is missing.	Check whether the cache agent was started with the correct cache administration user ID. If the cache administration user ID is correct, follow the procedure in "Recover and reset autorefresh Oracle objects" on page 94. Check the user error log for messages about "fatal anomalies". This indicates corrupt or missing Oracle objects.
If the current log sequence number in the TT_*version*_USER_COUNT table changes, is different from the bookmark and the associated cached table is not refreshed by the next committed autorefresh.	Restart the cache agent. If that does not work, follow the procedure in "Recover and reset autorefresh Oracle objects" on page 94.
Resource problem	Restart the cache agent.

Reset autorefresh state

Incremental autorefresh does not work if the TRUNCATE statement is used on an Oracle base table. If TRUNCATE is used on an Oracle base table, then you must reset autorefresh by using the ALTER CACHE GROUP statement to set the autorefresh state to OFF followed by another ALTER CACHE GROUP to reset the autorefresh state to ON.

Recover and reset autorefresh Oracle objects

If you know or suspect the Oracle objects used by autorefresh are the cause of the problem, use the following procedure to re-create the Oracle objects.

1. Use ALTER CACHE GROUP to reset the autorefresh state to OFF on all cache groups on all data stores that have the affected cached table:

   ```
   ALTER CACHE GROUP cache_group_name SET AUTOREFRESH STATE OFF;
   ```

2. Shut down all cache agents on all affected data stores.

3. Check if the user count is zero for each table in the cache group.

   On the Oracle database, execute the following statement:

   ```
   SELECT usercount FROM autorefresh_id.tt_version_user_count
       WHERE tablename ='owner.tablename';
   ```

   If the count is not zero, set the count to zero:

   ```
   UPDATE autorefresh_id.tt_version_user_count SET usercount = 0
       WHERE tablename ='owner.tablename';
   ```

4. Start one of the cache agents. The cache agent performs a clean up operation. It displays the following message to the support log after it has completed the cleanup:

   ```
   Cleanup of the Oracle objects completed
   ```

5. After the cache agent has completed the clean up, use ALTER CACHE GROUP to reset the autorefresh state back to ON:

   ```
   ALTER CACHE GROUP cache_group_name SET AUTOREFRESH STATE ON;
   ```

6. Start all other cache agents.

7. Use ALTER CACHE GROUP to reset the autorefresh state back to ON for all of the affected cache groups on all data stores.

Incremental autorefresh not progressing

If incremental autorefresh is not progressing, verify that:

- Autorefresh state is ON
- Cache agent is running

Inspect the support log for the conditions described in the following table:

Condition	What To Do
Oracle server connection errors or warnings	See "Troubleshooting client/server problems" on page 44 for information about resolving connection problems.
Lock timeout errors or warnings on TimesTen	This usually occurs because of an open DDL transaction on the cache group. Commit the DDL transaction so that autorefresh can get the necessary locks.
Insufficient permanent data partition errors on TimesTen	Increase **PermSize**.
Autorefresh Oracle object validations errors or warnings	See "Recover and reset autorefresh Oracle objects" on page 94.
Cache agent exits unexpectedly.	Contact Technical Support.
Core files in main daemon directory	Contact Technical Support.
Warnings about incremental autorefresh becoming full refresh	See "Incremental autorefresh becomes full autorefresh" on page 98.
Warnings that autorefresh has not finished for a long time	The autorefresh transaction can take a long time if many transactions have occurred since the last autorefresh. **Note**: Cache groups with the same autorefresh interval are autorefreshed in one transaction.

Validate autorefresh Oracle objects

The cache agent automatically verifies that Oracle objects exist and that they are valid so that autorefresh can progress. In normal operation, you should not see object validation errors or warnings in the user error log. If you see object

validation errors, contact Technical Support *unless* one of the following conditions has occurred:

- The TimesTen data store has been destroyed without using the DROP CACHE GROUP statement.
- A customer application inadvertently modifies the objects directly in the Oracle Database.
- A DDL operation occurs on the base table on the Oracle Database. This disables the trigger that controls autorefresh operations.

The cache group needs to be re-created if one of the preceding conditions has occurred.

Incremental autorefresh becomes full autorefresh

Incremental autorefresh can become full autorefresh if the cache administration user tablespace becomes full.

This section includes the following topics:

- Detecting when incremental autorefresh becomes full
- Understanding the cache administration user tablespace
- Diagnosing a full cache administration user tablespace

Detecting when incremental autorefresh becomes full

You can detect when incremental autorefresh becomes full refresh by several methods:

- Check for messages in the support log that indicate full autorefresh operations are occurring. For example:

  ```
  2007-08-08 08:06:51.35 Warn: ORA: 22119: ora-22119-0015-
  refresh05652: A full autorefresh will be performed for
  Incremental autorefresh table USER1.READTAB because change log
  table T_03_55555_L on Oracle has been truncated.
  ```

- Use the **ttCacheAutorefreshStatsGet** procedure.
  - If autorefresh is *InProgress* for longer than usual, full autorefresh may be occurring.
  - If a much larger number of rows (*autoRefNumRows*) was autorefreshed than usual, full autorefresh may have occurred.

  Check the support log for messages about full autorefresh.
- If SNMP traps are enabled, the **ttCacheRecoveryAutorefreshTrap** SNMP trap indicates a full autorefresh.

Understanding the cache administration user tablespace

TimesTen strongly recommends creating a separate tablespace for the cache administration user. This tablespace is used as the cache administration user's default tablespace. The tablespace contains autorefresh triggers for each Oracle table, change log tables for each Oracle table, and other objects that TimesTen needs for each cache administration user. If you do not specify a separate tablespace, then these objects are placed in the Oracle system tablespace.

Specify the tablespace when you create the cache administration user on Oracle. You can also specify the tablespace after user creation with the DEFAULT TABLESPACE clause of the Oracle ALTER USER statement.

Change log tables for each of the cached Oracle tables reside in the cache administration user tablespace. For each update on an Oracle table, one row (a

change log record) is inserted into the change log table for that Oracle table. The size of a change log record in bytes is as follows:

```
size of change log record = size of primary key on Oracle table + 250
```

The number of records in a change log table depends on the update rate on the Oracle table and on the autorefresh interval on TimesTen. Every 20 seconds, TimesTen removes change log records that have been applied to all data stores that cache the associated Oracle table.

When change logs are removed, a message similar to the following is displayed in the support log:

```
16:32:26.73 Info: ORA: 5652: ora-5652-4756-ogTblGC01036: Garbage
collector deleted 1 rows from TT_03_383270_L where logseq < 1
```

When the cache administration user tablespace gets full, the autorefresh trigger makes space for new change log records by deleting existing change log records. This can cause a full refresh for some tables on some TimesTen data stores.

Diagnosing a full cache administration user tablespace

Check for the following conditions if the cache administration user tablespace is full:

- Is the autorefresh state set to PAUSED? Change log records accumulate when the state is PAUSED.
- Has the cache group been created but not loaded? The default autorefresh state for cache group creation is PAUSED.
- Is a cache group being created or is a data store being duplicated? Both of these operations temporarily stop clean-up operations on the change log table.
- Are the cache agents on all TimesTen data stores running? If a cache agent is not running, change log records accumulate.
- Has a data store been abandoned without dropping autorefresh cache groups in the data store? Abandoned data stores result from scenarios such as the following:
  - The data store is destroyed by **ttDestroy** `-force`.
  - The application connected to the data store with the **Overwrite** connection attribute set to 1, but the cache groups that were in the old data store are not re-created.

  If the data store still exists, connect to the abandoned data store and drop the cache group.

Use the `autorefreshChangeLogInfo.sql` script to find out how large the change log tables are for each cached Oracle table. Use the output to verify that the data stores are still in use. See “Displaying information from the change log tables” on page 89.

If the data stores are still in use, verify that the cache agents are running.

Compare the autorefresh progress on TimesTen to the maximum log sequence number on the change log table. If TimesTen is behind, then call the **ttCacheAutorefreshStatsGet** procedure to see whether the autorefresh operations are successful. See "Using the ttCacheAutorefreshStatsGet procedure" on page 86.

If the status is *InProgress* longer than seems reasonable, see "Poor autorefresh performance" on page 101.

You may need to decrease the autorefresh interval or increase the size of the cache administration user tablespace.

Poor autorefresh performance

Poor autorefresh performance is usually the result of large autorefresh operations. Use the **ttCacheAutorefreshStatsGet** procedure to check the autorefresh duration and observe whether the status remains *InProgress* for a long time.

Factors that can cause large autorefresh operations include:

- Incremental autorefresh becomes full autorefresh
- Large autorefresh interval
- Large number of cache groups with the same interval
- High rate of changes to the Oracle tables
- The number of generations of child tables in a cache group
- The number of rows in the cached Oracle tables
- The size of the rows in the cached Oracle tables

Enable an AUTOREFRESH trace to diagnose autorefresh performance problems. See “AUTOREFRESH tracing” on page 26.

Problems with Cache Administrator

This section describes some of the possible problems you may encounter with Cache Administrator.

Possible cause	See...
Web server not running	“Check Web server” on page 102
Cache Administrator cannot access the columns list	“Check the type of DSN defined for your data store” on page 102
“Page Cannot Be Displayed” error when opening the Cache Administrator	“Check URL and Web server configuration” on page 102
The Cache Administrator does not allow login	“Check Cache Connect to Oracle attributes in the DSN” on page 103
Selected child tables do not appear in the cache group hierarchy and the error “child tables are not yet added” is displayed	“Define table hierarchy” on page 103

Check Web server

Use the **ttStatus** utility as described in “Using the ttStatus utility” on page 10 to check the status of the Web server process. For example, if the Web server is running, **ttStatus** reports the Web server as ‘started’, along with the process ID and port number:

```
TimesTen status report as of Tue Oct 05 13:45:31 2005

Daemon pid 556 port 16000 instance tt60
TimesTen server pid 1168 started on port 15102
TimesTen webserver pid 1108 started on port 15104
```

If the Web server is not running, use the **ttDaemonAdmin** utility to start it:

```
% ttDaemonAdmin -startwebserver
```

Check the type of DSN defined for your data store

When using the Cache Administrator to access a columns list from the Oracle database, you must define your DSN as a System DSN. The columns list cannot be accessed from the Cache Administrator if the DSN is a User DSN. See “Data source names” in *Oracle TimesTen In-Memory Database Operations Guide*.

Check URL and Web server configuration

Check the following:

- Has the Web server been configured correctly? See *Oracle TimesTen In-Memory Database Installation Guide*.
- Are you using the correct URL? It should be http://localhost:*port*/cache. *port* is the port number of the daemon's Web server.
- Are you using the correct host and port number? Use the **ttStatus** utility to check the port number and the **ttmodinstall** utility to change the port number, if necessary.
- Is Cache Administrator on your local machine? The Cache Administrator can only be accessed locally on the TimesTen host.

Check Cache Connect to Oracle attributes in the DSN

Check the following:

- Does the DSN exist?
- Is the DSN string syntax correct?
- Is the Oracle ID set in the DSN string?
- Is the Oracle ID valid? Through SQL*Plus, can you connect to the Oracle instance using the Oracle ID?
- Are the user name and password correct?

Make sure that the perl that is in use (derived from the $PATH environment variable) is the same as the installed perl. The location of the installed perl can be found in the PERL and PERLLIB parameters of the `webserver.config` file.

If you are running on Windows, make sure your PATH environment variable includes the path to your Windows 'system32' folder (for example: `C:\winnt\system32`)

Define table hierarchy

After selecting the root and **other tables** on the **Create a Cache Group Definition** page, the Cache Administrator does not display the other tables within the cache hierarchy until they have been added. For example, if there are three tables in the cache group: ROOT, TABLE1 and TABLE2, there are three possible hierarchies for the cache group:

- TABLE1 and TABLE2 are children of ROOT.
- TABLE1 is a child table of ROOT and TABLE2 is a child table of TABLE1.
- TABLE2 is a child table of ROOT and TABLE1 is a child table of TABLE2.

As the number of tables increases, so does the number of possible hierarchies. The Cache Administrator does not compute the various combinations. You must explicitly define the group hierarchy from the list of selected tables.

5

Troubleshooting AWT Cache Groups

Creating an asynchronous writethrough (AWT) cache group automatically creates a replication scheme that allows the data store to communicate with the Oracle database. You must start the replication agent after you create an AWT cache group and start the cache agent. See "Setting up an AWT cache group" in *TimesTen Cache Connect to Oracle Guide*.

Material in Chapter 6, "Troubleshooting Replication" is useful for troubleshooting AWT cache group problems. The useful replication topics are summarized in these sections:

- Unable to start or stop replication agent
- Replication does not work
- Using SNMP traps for notification of replication events

This chapter also contains the following sections:

- Poor AWT performance
- Permanent Oracle errors reported by TimesTen
- Transient Oracle errors reported by TimesTen

Unable to start or stop replication agent

This section describes what to check if you are unable to start or stop a replication agent.

Possible cause	What to do
Access Control is enabled and you do not have ADMIN privileges	If Access Control is enabled on the data store, you must have root or ADMIN privileges to use the **ttAdmin** utility or the **ttRepStart** or **ttRepStop** procedures to start or stop a replication agent.
TimesTen daemon not started	Check the state of the TimesTen daemon, as described in "Check the TimesTen user error log" on page 39. If necessary, start the TimesTen daemon as described in "Working with the Oracle TimesTen Data Manager Daemon" in *Oracle TimesTen In-Memory Database Operations Guide*.

Replication does not work

If you are unable to get replication working, the problem may be one or more of the following:

Possible cause	See...
TimesTen daemon or replication agents not running	"Check status of TimesTen daemon and replication agents" on page 122
Replication agents not communicating	"Check that replication agents are communicating" on page 124
Replication not in **Start** state	"Check replication state" on page 124

Using SNMP traps for notification of replication events

TimesTen can send SNMP traps for certain replication events to enable network management software to take immediate action. TimesTen can send the following SNMP traps:

- **ttRepAgentExitingTrap**
- **ttRepAgentDiedTrap**
- **ttRepAgentStartingTrap**

These traps are described in "Diagnostics through SNMP Traps" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

Poor AWT performance

This section addresses issues that may degrade AWT performance.

Possible cause	See...
Slow network	"Check network bandwidth" on page 132
Log buffer too small	"Check size of log buffer" on page 133
Frequent or inefficient disk writes	"Check durability settings" on page 133
Reading from log files on disk instead of the log buffer	"Check for reads from log files" on page 133

Permanent Oracle errors reported by TimesTen

Insert, update, or delete errors that occur while applying changes to Oracle are saved in an error file located in the data store directory with the following name:

DatastoreName.awterr

Errors reported to this file are *permanent* errors. TimesTen does not retry the transaction. The errors may be reported in the AWT error file long after the commit to TimesTen occurs.

The format of the messages in the AWT error file is similar to those generated for conflict and transaction errors in replication, as shown in Example 5.1. Oracle error messages are also reported in the support log and the user log.

Example 5.1 If a constraint violation occurs when a cache group update is propagated to Oracle, the message in the AWT error file is similar to the following:

```
Error occurred 14:48:55 on 03-22-2007
Datastore: c:\temp\cgDSN
Oracle Id: system1
Transmitting name: cgDSN
Error message:
  TT5210: Oracle unique constraint violation error in
OCIStmtExecute(): ORA-00001: unique constraint
(GUSER.SYS_C00357240) violated rc = -1 -- file "bdbTblH.c",
lineno 1205, procedure "ttBDbStmtForce()"
  TT5025: Commit failure in Oracle. Transaction must be rolled
back in TimesTen. -- file "bdbConnect.c", lineno 885, procedure
"ttBDbXact()"
```

```
Operation that caused the error:
Insert into table TESTUSER.T1 <9,1000>

Failed transaction:
Insert into table TESTUSER.T1 <9, 1000>
End of failed transaction
```

Example 5.2 If an object that TimesTen has placed on Oracle is dropped, the message in the AWT error file is similar to the following:

```
May 04 18:12:36 HOST1 TimesTen Replication 7.0[2136]:
[Err ] DEFAULT:meta.c(639):
TT16062: Failed to compile command:
select p.commit_timestamp, p.commit_seqnum, p.protocol from
owner1.TT_03_REPPEERS p where p.replication_name = :rname and
p.replication_owner = :rowner and p.tt_store_id   = :oid and
p.subscriber_id = :sid

May 04 18:12:36 HOST1 TimesTen Replication 7.0[2136]:
[Err ] DEFAULT:meta.c(639):
TT5221: TT5221: Oracle syntax error in OCIStmtExecute():
ORA-00942: table or view does not exist rc = -1 -- file
"bdbStmt.c", lineno 1041, procedure "getOraOutTypesNLengths()"
```

In this example, the `TT_03_REPPEERS` table does not exist. To recover from this error, perform the following tasks:

1. Stop the replication agent.
2. Drop and re-create the cache group.
3. Restart the replication agent.

Transient Oracle errors reported by TimesTen

The support log for data stores with AWT cache groups may contain Oracle errors if the replication agent encounters a problem on the Oracle database. If the replication agent encounters one of these errors, AWT rolls back the transaction and retries it. If the support log becomes full, the oldest messages are deleted and replaced by new messages.

The Oracle errors in the support log are considered *transient* because AWT retries the transaction.

Some transient errors indicate an underlying problem on the Oracle database must be solved before AWT operations can continue. For example:

```
ORA-01536: space quota exceeded for tablespace
ORA-01034: ORACLE not available
```

After the underlying problem has been fixed, AWT retries the operation.

For more information about the Oracle errors, see *Oracle Database Error Messages* for the Oracle release you are using.

The following Oracle errors are transient:

```
ORA-00018: maximum number of sessions exceeded
ORA-00019: maximum number of session licenses exceeded
ORA-00020: maximum number of processes (%s) exceeded
ORA-00025: failed to allocate %s
ORA-00028: your session has been killed
ORA-00038: Cannot create session: server group belongs to another
user
ORA-00051: timeout occurred while waiting for a resource
ORA-00052: maximum number of enqueue resources (%s) exceeded
ORA-00053: maximum number of enqueues exceeded
ORA-00054: resource busy and acquire with NOWAIT specified
ORA-00055: maximum number of DML locks exceeded
ORA-00057: maximum number of temporary table locks exceeded
ORA-00058: DB_BLOCK_SIZE must be %s to mount this database (not
%s)
ORA-00059: maximum number of DB_FILES exceeded
ORA-00060: deadlock detected while waiting for resource
ORA-00063: maximum number of LOG_FILES exceeded
ORA-00064: object is too large to allocate on this O/S (%s,%s)
ORA-00099: timed out while waiting for resource, potential PDML
deadlock
ORA-00104: deadlock detected; all public servers blocked waiting
for resources
ORA-00107: failed to connect to ORACLE listener process
ORA-00115: connection refused; dispatcher connection table is
full
ORA-00125: connection refused; invalid presentation
ORA-00126: connection refused; invalid duplicity
ORA-00284: recovery session still in progress
ORA-00370: potential deadlock during kcbchange operation
ORA-00371: not enough shared pool memory
ORA-00376: file %s cannot be read at this time
ORA-00379: no free buffers available in buffer pool %s for block
size %sK
ORA-00384: Insufficient memory to grow cache
ORA-00568: Maximum number of interrupt handlers exceeded
ORA-00579: osndnt: server received malformed connection request
ORA-00600: internal error code, arguments: [%s], [%s], [%s],
[%s], [%s], [%s], [%s], [%s]
ORA-00603: ORACLE server session terminated by fatal error
```

```
ORA-01000: maximum open cursors exceeded
ORA-01012: not logged on
ORA-01014: ORACLE shutdown in progress
ORA-01019: unable to allocate memory in the user side
ORA-01031: insufficient privileges
ORA-01033: ORACLE initialization or shutdown in progress
ORA-01034: ORACLE not available
ORA-01035: ORACLE only available to users with RESTRICTED SESSION
privilege
ORA-01037: maximum cursor memory exceeded
ORA-01046: cannot acquire space to extend context area
ORA-01073: fatal connection error: unrecognized call type
ORA-01089: immediate shutdown in progress - no operations are
permitted
ORA-01090: shutdown in progress - connection is not permitted
ORA-01092: ORACLE instance terminated. Disconnection forced
ORA-01094: ALTER DATABASE CLOSE in progress. Connections not
permitted
ORA-01109: database not open
ORA-01147: SYSTEM tablespace file %s is offline
ORA-01154: database busy. Open, close, mount, and dismount not
allowed now
ORA-01155: the database is being opened, closed, mounted or
dismounted
ORA-01219: database not open: queries allowed on fixed tables/
views only
ORA-01237: cannot extend datafile %s
ORA-01456: may not perform insert/delete/update operation inside
a READ ONLY transaction
ORA-01536: space quota exceeded for tablespace '%s'
ORA-01539: tablespace '%s' is not online
ORA-01542: tablespace '%s' is offline, cannot allocate space in
it
ORA-01562: failed to extend rollback segment number %s
ORA-01573: shutting down instance, no further change allowed
ORA-01628: max # extents (%s) reached for rollback segment %s
ORA-01629: max # extents (%s) reached saving undo for tablespace
%s
ORA-01630: max # extents (%s) reached in temp segment in
tablespace %s
ORA-01631: max # extents (%s) reached in table %s.%s
ORA-01632: max # extents (%s) reached in index %s.%s
ORA-01650: unable to extend rollback segment %s by %s in
tablespace %s
ORA-01651: unable to extend save undo segment by %s for
tablespace %s
ORA-01652: unable to extend temp segment by %s in tablespace %s
ORA-01653: unable to extend table %s.%s by %s in tablespace %s
ORA-01654: unable to extend index %s.%s by %s in tablespace %s
```

```
ORA-01655: unable to extend cluster %s.%s by %s in tablespace %s
ORA-01656: max # extents (%s) reached in cluster %s.%s
ORA-01658: unable to create INITIAL extent for segment in
tablespace %s
ORA-01659: unable to allocate MINEXTENTS beyond %s in tablespace
%s
ORA-01680: unable to extend LOB segment by %s in tablespace %s
ORA-01681: max # extents (%s) reached in LOB segment in
tablespace %s
ORA-01683: unable to extend index %s.%s partition %s by %s in
tablespace %s
ORA-01684: max # extents (%s) reached in table %s.%s partition %s
ORA-01685: max # extents (%s) reached in index %s.%s partition %s
ORA-01686: max # files (%s) reached for the tablespace %s
ORA-01688: unable to extend table %s.%s partition %s by %s in
tablespace %s
ORA-01691: unable to extend lob segment %s.%s by %s in tablespace
%s
ORA-01692: unable to extend lob segment %s.%s partition %s by %s
in tablespace %s
ORA-01693: max # extents (%s) reached in lob segment %s.%s
ORA-01694: max # extents (%s) reached in lob segment %s.%s
partition %s
ORA-03113: end-of-file on communication channel
ORA-03114: not connected to ORACLE
ORA-03134: Connections to this server version are no longer
supported.
ORA-03135: connection lost contact
ORA-03136: inbound connection timed out
ORA-03232: unable to allocate an extent of %s blocks from
tablespace %s
ORA-03233: unable to extend table %s.%s subpartition %s by %s in
tablespace %s
ORA-03234: unable to extend index %s.%s subpartition %s by %s in
tablespace %s
ORA-03235: max # extents (%s) reached in table %s.%s subpartition
%s
ORA-03236: max # extents (%s) reached in index %s.%s subpartition
%s
ORA-03237: Initial Extent of specified size cannot be allocated
ORA-03238: unable to extend LOB segment %s.%s subpartition %s by
%s in tablespace %s
ORA-03239: maxextents (%s) reached in LOB segment %s.%s
subpartition %s
ORA-04020: deadlock detected while trying to lock object
%s%s%s%s%s
ORA-06019: NETASY: invalid login (connect) string
ORA-06021: NETASY: connect failed
ORA-06030: NETDNT: connect failed, unrecognized node name
```

```
ORA-06031: NETDNT: connect failed, unrecognized object name
ORA-06032: NETDNT: connect failed, access control data rejected
ORA-06033: NETDNT: connect failed, partner rejected connection
ORA-06034: NETDNT: connect failed, partner exited unexpectedly
ORA-06035: NETDNT: connect failed, insufficient resources
ORA-06036: NETDNT: connect failed, no response from object
ORA-06037: NETDNT: connect failed, node unreachable
ORA-06039: NETDNT: connect failed
ORA-06040: NETDNT: invalid login (connect) string
ORA-06108: NETTCP: connect to host failed
ORA-06113: NETTCP: Too many connections
ORA-06114: NETTCP: SID lookup failure
ORA-06143: NETTCP: maximum connections exceeded
ORA-06315: IPA: Invalid connect string
ORA-06316: IPA: Invalid database SID
ORA-06317: IPA: Local maximum number of users exceeded
ORA-06318: IPA: Local maximum number of connections exceeded
ORA-06319: IPA: Remote maximum number of users exceeded
ORA-06320: IPA: Remote maximum number of connections exceeded
ORA-06404: NETCMN: invalid login (connect) string
ORA-06413: Connection not open.
ORA-10435: enable tracing of global enqueue service deadlock
detetction
ORA-10626: specify timeout for online index rebuild to wait for
DML
ORA-10906: Unable to extend segment after insert direct load
ORA-12150: TNS:unable to send data
ORA-12151: TNS:received bad packet type from network layer
ORA-12152: TNS:unable to send break message
ORA-12153: TNS:not connected
ORA-12154: TNS:could not resolve service name
ORA-12155: TNS:received bad datatype in NSWMARKER packet
ORA-12156: TNS:tried to reset line from incorrect state
ORA-12157: TNS:internal network communication error
ORA-12158: TNS:could not initialize parameter subsystem
ORA-12159: TNS:trace file not writeable
ORA-12160: TNS:internal error: Bad error number
ORA-12161: TNS:internal error: partial data received
ORA-12162: TNS:service name is incorrectly specified
ORA-12163: TNS:connect descriptor is too long
ORA-12166: TNS:Client can not connect to HO agent.
ORA-12168: TNS:Unable to contact Directory Server.
ORA-12169: TNS:Net service name given as connect identifier is
too long
ORA-12170: TNS:Connect timeout occurred
ORA-12171: TNS:could not resolve connect identifier: %s
ORA-12196: TNS:received an error from TNS
ORA-12197: TNS:keyword-value resolution error
ORA-12198: TNS:could not find path to destination
```

```
ORA-12200: TNS:could not allocate memory
ORA-12201: TNS:encountered too small a connection buffer
ORA-12202: TNS:internal navigation error
ORA-12203: TNS:unable to connect to destination
ORA-12204: TNS:received data refused from an application
ORA-12205: TNS:could not get failed addresses
ORA-12206: TNS:received a TNS error during navigation
ORA-12207: TNS:unable to perform navigation
ORA-12208: TNS:could not find the TNSNAV.ORA file
ORA-12209: TNS:encountered uninitialized global
ORA-12210: TNS:error in finding Navigator data
ORA-12211: TNS:needs PREFERRED_CMANAGERS entry in TNSNAV.ORA
ORA-12212: TNS:incomplete PREFERRED_CMANAGERS binding in
TNSNAV.ORA
ORA-12213: TNS:incomplete PREFERRED_CMANAGERS binding in
TNSNAV.ORA
ORA-12214: TNS:missing local communities entry in TNSNAV.ORA
ORA-12215: TNS:poorly formed PREFERRED_NAVIGATORS Addresses in
TNSNAV.ORA
ORA-12216: TNS:poorly formed PREFERRED_CMANAGERS addresses in
TNSNAV.ORA
ORA-12217: TNS:could not contact PREFERRED_CMANAGERS in
TNSNAV.ORA
ORA-12218: TNS:unacceptable network configuration data
ORA-12219: TNS:missing community name from address in
ADDRESS_LIST
ORA-12221: TNS:illegal ADDRESS parameters
ORA-12222: TNS:no such protocol adapter
ORA-12223: TNS:internal limit restriction exceeded
ORA-12224: TNS:no listener
ORA-12225: TNS:destination host unreachable
ORA-12226: TNS:operating system resource quota exceeded
ORA-12227: TNS:syntax error
ORA-12228: TNS:protocol adapter not loadable
ORA-12229: TNS:Interchange has no more free connections
ORA-12230: TNS:Severe Network error ocurred in making this
connection
ORA-12231: TNS:No connection possible to destination
ORA-12232: TNS:No path available to destination
ORA-12233: TNS:Failure to accept a connection
ORA-12235: TNS:Failure to redirect to destination
ORA-12236: TNS:protocol adapter not loaded
ORA-12316: syntax error in database link's connect string
ORA-12326: database %s is closing immediately;  no operations are
permitted
ORA-12329: database %s is closed;  no operations are permitted
ORA-12500: TNS:listener failed to start a dedicated server
process
ORA-12501: TNS:listener failed to spawn process
```

```
ORA-12502: TNS:listener received no CONNECT_DATA from client
ORA-12504: TNS:listener was not given the SID in CONNECT_DATA
ORA-12505: TNS:listener could not resolve SID given in connect
descriptor
ORA-12506: TNS:listener was not given the ALIAS in CONNECT_DATA
ORA-12507: TNS:listener could not resolve ALIAS given
ORA-12508: TNS:listener could not resolve the COMMAND given
ORA-12509: TNS:listener failed to redirect client to service
handler
ORA-12510: TNS:database temporarily lacks resources to handle the
request
ORA-12511: TNS:service handler found but it is not accepting
connections
ORA-12512: TNS:service handler found but it has not registered a
redirect address
ORA-12513: TNS:service handler found but it has registered for a
different protocol
ORA-12514: TNS:listener could not resolve SERVICE_NAME given in
connect descriptor
ORA-12515: TNS:listener could not find a handler for this
presentation
ORA-12516: TNS:listener could not find available handler with
matching protocol stack
ORA-12517: TNS:listener could not find service handler supporting
direct handoff
ORA-12518: TNS:listener could not hand off client connection
ORA-12519: TNS:no appropriate service handler found
ORA-12520: TNS:listener could not find available handler for
requested type of server
ORA-12521: TNS:listener could not resolve INSTANCE_NAME given in
connect descriptor
ORA-12522: TNS:listener could not find available instance with
given INSTANCE_ROLE
ORA-12523: TNS:listener could not find instance appropriate for
the client connection
ORA-12524: TNS:listener could not resolve HANDLER_NAME given in
connect descriptor
ORA-12525: TNS:listener has not received client's request in time
allowed
ORA-12526: TNS:listener: all appropriate instances are in
restricted mode
ORA-12527: TNS:listener: all instances are in restricted mode or
blocking new connections
ORA-12528: TNS:listener: all appropriate instances are blocking
new connections
ORA-12529: TNS:connect request rejected based on current
filtering rules
ORA-12531: TNS:cannot allocate memory
ORA-12532: TNS:invalid argument
```

```
ORA-12533: TNS:illegal ADDRESS parameters
ORA-12534: TNS:operation not supported
ORA-12535: TNS:operation timed out
ORA-12536: TNS:operation would block
ORA-12537: TNS:connection closed
ORA-12538: TNS:no such protocol adapter
ORA-12539: TNS:buffer over- or under-flow
ORA-12540: TNS:internal limit restriction exceeded
ORA-12541: TNS:no listener
ORA-12542: TNS:address already in use
ORA-12543: TNS:destination host unreachable
ORA-12544: TNS:contexts have different wait/test functions
ORA-12545: Connect failed because target host or object does not
exist
ORA-12546: TNS:permission denied
ORA-12547: TNS:lost contact
ORA-12549: TNS:operating system resource quota exceeded
ORA-12550: TNS:syntax error
ORA-12551: TNS:missing keyword
ORA-12552: TNS:operation was interrupted
ORA-12554: TNS:current operation is still in progress
ORA-12555: TNS:permission denied
ORA-12556: TNS:no caller
ORA-12557: TNS:protocol adapter not loadable
ORA-12558: TNS:protocol adapter not loaded
ORA-12560: TNS:protocol adapter error
ORA-12561: TNS:unknown error
ORA-12562: TNS:bad global handle
ORA-12564: TNS:connection refused
ORA-12566: TNS:protocol error
ORA-12569: TNS:packet checksum failure
ORA-12570: TNS:packet reader failure
ORA-12571: TNS:packet writer failure
ORA-12574: TNS:redirection denied
ORA-12582: TNS:invalid operation
ORA-12583: TNS:no reader
ORA-12585: TNS:data truncation
ORA-12589: TNS:connection not bequeathable
ORA-12590: TNS:no I/O buffer
ORA-12591: TNS:event signal failure
ORA-12592: TNS:bad packet
ORA-12593: TNS:no registered connection
ORA-12595: TNS:no confirmation
ORA-12596: TNS:internal inconsistency
ORA-12600: TNS: string open failed
ORA-12602: TNS: Connection Pooling limit reached
ORA-12606: TNS: Application timeout occurred
ORA-12607: TNS: Connect timeout occurred
ORA-12608: TNS: Send timeout occurred
```

```
ORA-12609: TNS: Receive timeout occurred
ORA-12612: TNS:connection is busy
ORA-12615: TNS:preempt error
ORA-12623: TNS:operation is illegal in this state
ORA-12624: TNS:connection is already registered
ORA-12636: Packet send failed
ORA-12637: Packet receive failed
ORA-12829: Deadlock - itls occupied by siblings at block %s of
file %s
ORA-12993: tablespace '%s' is offline, cannot drop column
ORA-14117: partition resides in offlined tablespace
ORA-14268: subpartition '%s' of the partition resides in offlined
tablespace
ORA-16000: database open for read-only access
ORA-16003: standby database is restricted to read-only access
ORA-16403: shutdown in progress - remote connection is not
permitted
ORA-16724: the intended state for resource has been set to
OFFLINE
ORA-16903: Unable to connect to database
ORA-16914: Missing connect string. Try \"help\"
ORA-18014: deadlock detected while waiting for resource %s
ORA-21521: exceeded maximum number of connections in OCI (object
mode only)
ORA-21522: attempted to use an invalid connection in OCI (object
mode only)
ORA-23317: a communication failure has occurred
ORA-24401: cannot open further connections
ORA-24418: Cannot open further sessions.
ORA-24778: cannot open connections
ORA-25400: must replay fetch
ORA-25401: can not continue fetches
ORA-25402: transaction must roll back
ORA-25403: could not reconnect
ORA-25405: transaction status unknown
ORA-25407: connection terminated
ORA-25408: can not safely replay call
ORA-25409: failover happened during the network operation,cannot
continue
ORA-25425: connection lost during rollback
ORA-29306: datafile %s is not online
ORA-30006: resource busy; acquire with WAIT timeout expired
ORA-30036: unable to extend segment by %s in undo tablespace '%s'
ORA-30040: Undo tablespace is offline
ORA-31443: deadlock detected while acquiring lock on %s
ORA-37013: (XSACQUIRE_DEADLOCK) Cannot wait to acquire object %j
since doing so would cause a deadlock.
ORA-44317: database open read-only
```

6

Troubleshooting Replication

This chapter describes how to troubleshoot some of the problems you may encounter when replicating data stores.

This chapter includes the following topics:

Unable to create a replication scheme

This section describes what to check if you are unable to use CREATE REPLICATION to create a replication scheme.

Possible cause	What to do
Access Control is enabled and you do not have DDL privileges	If Access Control is enabled on the data store, you must have DDL privileges to use the CREATE REPLICATION or DROP REPLICATION statements.
Incorrect data store name, host name, or element name.	• Check the CREATE REPLICATION statement for typographical errors. • See "Check host names" on page 126. • Use official host names instead of aliases. • The host name should match the value returned by the `hostname` command on your system and should be used consistently throughout the replication scheme.
The local host is not part of the replication scheme.	Create the replication scheme on a host that will be part of the replication scheme.
Replication tables defined in the CREATE REPLICATION statement do not exist.	The name, owner, and column definitions of the tables participating in the replication scheme must be identical on both the master and subscriber data stores. Use CREATE TABLE to create tables on the data store, or use the **ttRepAdmin** `-duplicate` utility or the **ttRepDuplicateEx** C function to duplicate the entire data store to be replicated.
Other problems	Review the procedures and requirements described in "Defining Replication Schemes" in *TimesTen to TimesTen Replication Guide*.

Unable to alter a replication scheme

This section describes what to check if you are unable to use ALTER REPLICATION to alter a replication scheme.

Possible cause	What to do
Access Control is enabled and you do not have DDL privileges	If Access Control is enabled on the data store, you must have DDL privileges to use the ALTER REPLICATION statement.
Replication agent in **Start** state	Most ALTER REPLICATION operations are supported only when the replication agent is stopped (**ttAdmin** `-repStop`). Stop the replication agents on both master and subscriber data stores, alter the replication scheme on both master and subscriber data stores, then restart both replication agents.
Incorrect data store name, host name, or element name	• Check ALTER REPLICATION statement for typographical errors. • See “Check host names” on page 126.
Replication table defined in the ALTER REPLICATION statement does not exist	Use CREATE TABLE to create a table on the data store.
Other problems	Review the procedures and requirements described in "Altering Replication" in *TimesTen to TimesTen Replication Guide*.

Unable to start or stop replication agent

This section describes what to check if you are unable to start or stop a replication agent.

Possible cause	What to do
Access Control is enabled and you do not have ADMIN privileges	If Access Control is enabled on the data store, you must have root or ADMIN privileges to use the **ttAdmin** utility or the **ttRepStart** or **ttRepStop** procedures to start or stop a replication agent.
TimesTen daemon not started	Check the state of the TimesTen daemon, as described in "Check the TimesTen user error log" on page 39. If necessary, start the TimesTen daemon as described in "Working with the Oracle TimesTen Data Manager Daemon" in *Oracle TimesTen In-Memory Database Operations Guide*.
Data store does not participate in a replication scheme.	If a data store does not participate in a replication scheme, attempts to start a replication agent for that data store will fail. Use CREATE REPLICATION to create a replication scheme for the data store.

Using SNMP traps for notification of replication events

TimesTen can send SNMP traps for certain replication events to enable network management software to take immediate action. TimesTen can send the following traps for replication events:

- **ttRepAgentExitingTrap**
- **ttRepAgentDiedTrap**
- **ttRepAgentStartingTrap**
- **ttRepCatchupStartTrap**
- **ttRepCatchupStopTrap**
- **ttRepReturnTransitionTrap**
- **ttRepSubscriberFailedTrap**
- **ttRepUpdateFailedTrap**

These traps are described in "Diagnostics through SNMP Traps" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

Replication does not work

If you are unable to get replication working between a master and subscriber data store, the problem may be one or more of the following:

Possible cause	See...
TimesTen daemon and/or replication agents not running	“Check status of TimesTen daemon and replication agents” on page 122
Master and subscriber agents not communicating	“Check that replication agents are communicating” on page 124
Replication not in **Start** state	“Check replication state” on page 124
Error in replication scheme	“Check replication scheme configuration” on page 125
Inconsistent owner names for replication scheme and tables	“Check owner names” on page 127
Inconsistent replication tables	“Check consistency between replicated tables” on page 129

Check status of TimesTen daemon and replication agents

Use the **ttStatus** utility to confirm the main TimesTen daemon is running and the replication agents are started for all of your master and subscriber data stores. The output from a simple replication scheme using a single master and subscriber data store should look like that shown in Example 6.1.

If the TimesTen daemon is running, but the replication agents are not, the output looks like that shown in Example 6.2. In this case, start the replication agents as described in "Starting and stopping the replication agents" in *TimesTen to TimesTen Replication Guide*.

If neither the TimesTen daemon or replication agents are running, the output looks like that shown in Example 6.3. In this case, confirm you have correctly installed TimesTen and the Data Manager service is started, as described in "TimesTen Installation" in *Oracle TimesTen In-Memory Database Installation Guide*.

Example 6.1

```
C:\>ttStatus
TimesTen status report as of Thu Jan 25 16:23:33 2007
Daemon pid 5088 port 17000 instance MYINSTANCE
TimesTen server pid 4344 started on port 17002
TimesTen webserver pid 4216 started on port 17004
```

```
------------------------------------------------------------------------
Data store c:\temp\subscriber1ds
There are 12 connections to the data store
Data store is in shared mode
Shared Memory KEY Global\DBI45b9471c.2.SHM.2 HANDLE 0x280
Type            PID     Context     Connection Name              ConnID
Process         1244    0x00d08fb0  subscriber1ds                     1
Replication     4548    0x00aed2f8  LOGFORCE                          4
Replication     4548    0x00b03470  TRANSMITTER                       5
Replication     4548    0x00b725a8  RECEIVER                          6
Replication     4548    0x00b82808  REPHOLD                           2
Replication     4548    0x00b98980  REPLISTENER                       3
Subdaemon       2752    0x00526768  Worker                         2042
Subdaemon       2752    0x0072a758  Flusher                        2043
Subdaemon       2752    0x007308c0  Checkpoint                     2044
Subdaemon       2752    0x00736a28  HistGC                         2046
Subdaemon       2752    0x067f02f8  Aging                          2045
Subdaemon       2752    0x068364a0  Monitor                        2047
Replication policy  : Manual
Replication agent is running.
Cache agent policy : Manual
------------------------------------------------------------------------
Data store c:\temp\masterds
There are 12 connections to the data store
Data store is in shared mode
Shared Memory KEY Global\DBI45b945d0.0.SHM.6 HANDLE 0x2bc
Type            PID     Context     Connection Name              ConnID
Process         5880    0x00d09008  masterds                          1
Replication     3728    0x00aed570  LOGFORCE                          4
Replication     3728    0x00b036e8  TRANSMITTER                       5
Replication     3728    0x00b168b8  REPHOLD                           3
Replication     3728    0x00b1ca20  REPLISTENER                       2
Replication     3728    0x00b22b88  RECEIVER                          6
Subdaemon       3220    0x00526768  Worker                         2042
Subdaemon       3220    0x0072e768  Flusher                        2043
Subdaemon       3220    0x007348d0  Checkpoint                     2044
Subdaemon       3220    0x067b0068  Aging                          2045
Subdaemon       3220    0x067c0040  Monitor                        2047
Subdaemon       3220    0x068404c8  HistGC                         2046
Replication policy  : Manual
Replication agent is running.
Cache agent policy : Manual
------------------------------------------------------------------------
Data store c:\temp\demo
There are no connections to the data store
Replication policy : Manual
Cache agent policy : Manual
------------------------------------------------------------------------
```

```
End of report
```

Example 6.2

```
> ttStatus
TimesTen status report as of Tue Oct 28 10:31:30 2006

Daemon pid 3396 port 15000 instance MYINSTANCE
TimesTen server pid 3436 started on port 15002
------------------------------------------------------------------------
Data store c:\temp\subscriberds
There are no connections to the data store
cache agent restart policy: manual
------------------------------------------------------------------------
Data store c:\temp\masterds
There are no connections to the data store
cache agent restart policy: manual
------------------------------------------------------------------------
End of report
```

Example 6.3

```
> ttStatus
ttStatus: Could not connect to TimesTen daemon: Connection
refused
```

Check that replication agents are communicating

Use **ttRepAdmin** `-receiver -list` to see that the replication agents are communicating with each other. If the *masterds* data store is replicating to *subscriberds*, the output should look similar to the following:

Example 6.4

```
> ttRepAdmin -receiver -list masterDSN
Peer name         Host name                Port    State  Proto
---------------  ------------------------ ------  ------- -----
SUBSCRIBERDS      MYHOST                   Auto   Start       10

Last Msg Sent Last Msg Recv Latency TPS     RecordsPS Logs
------------- ------------- ------- ------- --------- ----
0:01:12       -               19.41       5        52    2
```

Check replication state

Use the **ttReplicationStatus** procedure to check state of the subscriber data store with respect to its master. If the subscriber is in the **Stop**, **Pause**, or **Failed** state, use the **ttReplicationStatus** procedure to reset the subscriber state to **Start**, as described in "Setting the replication state of subscribers" in *TimesTen to TimesTen Replication Guide*.

Example 6.5 Use **ttReplicationStatus** to obtain the status of the *subscriberds* data store from its master data store, *masterDSN*, enter:

```
> ttIsql masterDSN

Command> CALL ttReplicationStatus ('subscriberds');
< SUBSCRIBERDS, MYHOST, 0, pause, 1, 10, REPSCHEME, REPL >
1 row found.
```

To reset state to **Start** call the **ttRepSubscriberStateSet** procedure:

```
Command> CALL ttRepSubscriberStateSet('REPSCHEME', 'REPL',
'SUBSCRIBERDS', 'MYHOST', 0)

Command> CALL ttReplicationStatus ('subscriberds');
< SUBSCRIBERDS, MYHOST, 0, start, 1, 152959, REPSCHEME, REPL >
1 row found.
```

Check replication scheme configuration

This section describes some procedures you can use to confirm the correct configuration of the various components in your replicated system. The basic procedure categories are:

- Check ttRepAdmin -showconfig
- Check the TTREP.TTSTORES table
- Check host names

Check ttRepAdmin -showconfig

Use **ttRepAdmin** `-showconfig` to confirm the configuration of your replication scheme.

What to look for:

- Are all of the subscriber agents started and reported to be in the **Start** state? If not, reset the agents to the **Start** state. See "Setting the replication state of subscribers" in *TimesTen to TimesTen Replication Guide*.
- Do the reported **Peer names** match the names given in the **DataStore** attributes in the DSN definitions for the replicated data stores? Replication does not work if you specified the names given for the **Data Source Name** attributes.
- Is there anything under **List of subscribers**? If not, confirm the data store names you specified in the DSN definition are consistent with those you specified in your replication scheme configuration file.
- Are the **Host names** correct? If in doubt, see “Check host names” on page 126.
- Are the correct table names displayed under **Table details**? If not, correct the table names in your replication scheme configuration file.

Example 6.6

```
> ttRepAdmin -showconfig masterDSN
Self host "MYHOST", port auto, name "MASTERDS", LSN 4/2970276,
timeout 120, threshold 0
List of subscribers
-----------------
Peer name         Host name                 Port    State  Proto
---------------  ------------------------ ------  ------- -----
SUBSCRIBERDS      MYHOST                    Auto   Start      10
Last Msg Sent Last Msg Recv Latency TPS     RecordsPS Logs
------------- ------------- ------- ------- --------- ----
0:01:12       -               19.41       5        52    2
List of tables and subscriptions
--------------------------------
Table details
-------------
Table : REPL.TAB
Master Name               Subscriber Name
-----------               -------------
MASTERDS                  SUBSCRIBERDS
```

Check the TTREP.TTSTORES table

Check the TTREP.TTSTORES table to confirm that replication associates the replication scheme with the local data store. Connect to the data store and enter:

```
SELECT * FROM ttrep.ttstores WHERE is_local_store <> 0x0;
```

Example 6.7

```
Command> select * from ttrep.ttstores where is_local_store <> 0x0;
< -5193371075573733683, MYHOST, MASTERDS, 01, 0, 0, 4, 0 >
1 row found.
```

There should be exactly one row returned. If more than one row is returned, contact Technical Support. If no rows are returned, then none of the hosts returned by the following statement is perceived to be a local system by TimesTen replication:

```
SELECT DISTINCT host_name FROM ttrep.ttstores;
```

It may also be that none of the data store names specified in your replication scheme match those specified in your DSN descriptions.

Check host names

Some hosts or IP addresses specified in a replication scheme cannot be resolved by the replication agent because:

- Host names or IP addresses specified in the replication scheme are wrong or misspelled.

- Host names or IP addresses cannot be resolved or found by DNS or in `/etc/hosts`.
- Entries in the `/etc/hosts` file are incorrectly ordered in appearance. This error is most common when multiple NICs are used. It is strongly recommended to have a network administrator or system administrator make changes to `/etc/hosts` files or DNS configuration.

See "Configuring host IP addresses" in *TimesTen to TimesTen Replication Guide* for details on how to configure DNS and `/etc/hosts` files for host machines used for replication.

To check if a host name in the replication scheme matches the host name of the local machine, write an application to perform these tasks:

1. Use a **gethostname** OS function call to determine the host name of the running host.
2. Call **gethostbyname** with the output from Step 1.
3. Call **gethostbyname** with the host name specified in the replication scheme.
4. Compare output of Step 2 and Step 3. If there is a match, then the running host is involved in replication. Otherwise, it is not involved in replication.

Check owner names

As described in "Table requirements and restrictions" and "Owner of the replication scheme and tables" in *TimesTen to TimesTen Replication Guide*, the owner names of your replication scheme and your replicated tables must be consistent across all participating data stores.

Checking replication owner

Check the owner name assigned to your replication scheme by calling the **ttIsql** `repschemes` command or by listing the contents of the TTREP.REPLICATIONS table.

Example 6.8 shows that the replication scheme name, REPSCHEME, has a consistent owner name (REPL) in the data stores on both SYSTEM1 and SYSTEM2. Example 6.9 shows the scheme name with inconsistent owner names. This can occur if you omit the owner name from the replication scheme definition and the system uses the Id of the replication scheme creator.

Example 6.8 On SYSTEM1:

```
> ttIsql masterDSN

Command> select * from ttrep.replications;
< REPSCHEME           , REPL                , C, 0, 0, -1 >
1 row found.
```

On SYSTEM2:

```
> ttIsql -connStr "dsn=subscriberDSN"
Command> select * from ttrep.replications;
< REPSCHEME            , REPL                 , C, 0, 0, -1 >
1 row found.
```

Example 6.9 On SYSTEM1:

```
> ttIsql masterDSN
Command> select * from ttrep.replications;
< REPSCHEME            , SYSTEM1              , C, 0, 0, -1 >
1 row found.
```

On SYSTEM2:

```
> ttIsql -connStr "dsn=subscriberDSN"
Command> select * from ttrep.replications;
< REPSCHEME            , SYSTEM2              , C, 0, 0, -1 >
1 row found.
```

Checking table owner

Check the owner names assigned to the tables in each data store by using the **ttIsql tables** command.

Example 6.10 This example shows that the TAB table has a consistent owner name (REPL) in the data stores on both SYSTEM1 and SYSTEM2.

Output for SYSTEM1	Output for SYSTEM2
SYS.CACHE_GROUP	SYS.CACHE_GROUP
SYS.COLUMNS	SYS.COLUMNS
SYS.COL_STATS	SYS.COL_STATS
SYS.INDEXES	SYS.INDEXES
SYS.MONITOR	SYS.MONITOR
SYS.PLAN	SYS.PLAN
SYS.TABLES	SYS.TABLES
SYS.TBL_STATS	SYS.TBL_STATS
SYS.TRANSACTION_LOG_API	SYS.TRANSACTION_LOG_API
REPL.TAB	**REPL.TAB**
TTREP.REPELEMENTS	TTREP.REPELEMENTS
TTREP.REPLICATIONS	TTREP.REPLICATIONS
TTREP.REPPEERS	TTREP.REPPEERS

Output for SYSTEM1	Output for SYSTEM2
TTREP.REPSTORES	TTREP.REPSTORES
TTREP.REPSUBSCRIPTIONS	TTREP.REPSUBSCRIPTIONS
TTREP.REPTABLES	TTREP.REPTABLES
TTREP.TTSTORES	TTREP.TTSTORES

Example 6.11 This example shows the TAB table with inconsistent owner names, which were automatically assigned for each host.

Output for SYSTEM1	Output for SYSTEM2
SYS.CACHE_GROUP	SYS.CACHE_GROUP
SYS.COLUMNS	SYS.COLUMNS
SYS.COL_STATS	SYS.COL_STATS
SYS.INDEXES	SYS.INDEXES
SYS.MONITOR	SYS.MONITOR
SYS.PLAN	SYS.PLAN
SYS.TABLES	SYS.TABLES
SYS.TBL_STATS	SYS.TBL_STATS
SYS.TRANSACTION_LOG_API	SYS.TRANSACTION_LOG_API
SYSTEM1.TAB	**SYSTEM2.TAB**
TTREP.REPELEMENTS	TTREP.REPELEMENTS
TTREP.REPLICATIONS	TTREP.REPLICATIONS
TTREP.REPPEERS	TTREP.REPPEERS
TTREP.REPSTORES	TTREP.REPSTORES
TTREP.REPSUBSCRIPTIONS	TTREP.REPSUBSCRIPTIONS
TTREP.REPTABLES	TTREP.REPTABLES
TTREP.TTSTORES	TTREP.TTSTORES

Check consistency between replicated tables

Replicated tables on both master and subscriber data stores must be exactly the same.

Example 6.12 This output from the user error log shows a mismatch on the number of columns for the subscriber table TTUSER.MYDSN.

```
11:37:58.00 Info: REP:  9430: REP1:transmitter.c(4936): TT16136: Sending
definition for table TTUSER.MYDSN (1 columns)
11:37:58.00 Info: REP:  9412: REP2:receiver.c(5928): TT16193: Adding definition
for table: TTUSER.MYDSN
11:37:58.00 Info: REP:  9412: REP2:meta.c(5580):TTUSER.MYDSN ptn 0: srcoff 0,
destoff 0, length 8
11:37:58.00 Info: REP:  9412: REP2:meta.c(5580):TTUSER.MYDSN ptn 1: srcoff 8,
destoff 12, length 12
11:37:58.00 Err : REP:  9412: REP2:receiver.c(6203): TT16198: Table definition
mismatch on number of columns for table TTUSER.MYDSN.  Local definition: 2;
transmitting peer: 1
11:37:58.00 Err : REP:  9412: REP2:receiver.c(6380): TT16204: Table TTUSER.MYDSN
marked invalid. Will not apply transactions received for it until a valid
definition is received
11:37:58.00 Err : REP:  9412: REP2:receiver.c(7200): TT16078: Table definition
for ID 637068 is invalid (Original failure 11:37:58
REP2:receiver.c(6203): TT16198: Table definition mismatch on number of columns
for table TTUSER.MYDSN.  Local definition: 2; transmitting peer: 1)
11:37:58.00 Err : REP:  9412: REP2:receiver.c(5002): TT16187: Transaction
1173958671/2; Error: transient 0, permanent 1
```

Replication unresponsive, appears hung

Possible cause	See...
Failed subscriber	“Check replication state” on page 131
Return-receipt timeout period too long	"Check return-receipt timeout setting"

Check replication state

Use the **ttReplicationStatus** procedure to check state of the subscriber data store with respect to its master. If the subscriber is in the **Failed** state, see "Managing data store failover and recovery" in *TimesTen to TimesTen Replication Guide* for information on how to deal with failed data stores.

Example 6.13 Use **ttReplicationStatus** to obtain the status of the *subscriberds* data store from its master data store, *masterDSN*, enter:

```
> ttIsql masterDSN

Command> CALL ttReplicationStatus ('subscriberds');
< SUBSCRIBERDS, MYHOST, 0, failed, 1, 10, REPSCHEME, REPL >
1 row found.
```

Check return-receipt timeout setting

Use the **ttRepSyncGet** procedure to check the return-receipt timeout setting. A value of -1 indicates the application is to wait until it receives an acknowledgement from the subscriber. Network latency or other issues might delay receipt of the subscriber acknowledgment. You either address these issues or use the **ttRepSyncGet** procedure to reset the return-receipt timeout period.

See "Checking the status of return service transactions" in *TimesTen to TimesTen Replication Guide* for more information.

Poor replication or XLA performance

Most of this section addresses issues that may impact replication performance. Some issues, such as log buffer too small and reading from the log files on disk, can impact the performance of both replication and XLA applications.

Possible cause	See...
Slow network	“Check network bandwidth” on page 132
Using RETURN RECEIPT	“Check use of return-receipt blocking” on page 132
Inefficient replication scheme	“Check replication configuration” on page 133
Log buffer too small	“Check size of log buffer” on page 133
Frequent or inefficient disk writes	“Check durability settings” on page 133
Reading from log files on disk rather than the log buffer	“Check for reads from log files” on page 133
High rate of conflicts	“Conflict reporting slows down replication” on page 139

Check network bandwidth

Replication agents typically communicate over some type of network connection. If replicating over a network slower than 10 MB per second (such as common with a 100 Base-T Ethernet typical in a LAN), you must be careful to match the transaction load to the available bandwidth of the network. See "Network bandwidth requirements" in *TimesTen to TimesTen Replication Guide* for details.

Check use of return-receipt blocking

Unless you need receipt confirmation for all your transactions, disable RETURN RECEIPT blocking. If you require receipt confirmation for some, but not all, transactions, then set RETURN RECEIPT BY REQUEST and call the **ttRepSyncSet** procedure to enable the return receipt service for specific transactions. See “RETURN RECEIPT BY REQUEST” under "Using a return service" in *TimesTen to TimesTen Replication Guide* for details.

Note: The performance degradation caused by return-receipt becomes less of an issue when multiple applications (or threads) are updating the data store. If you

must use return-receipt in a transaction, you can improve the performance of your application by using multiple threads to update the data store. Though each thread must block for receipt confirmation, the other threads are free to make updates.

Check replication configuration

In addition to return-receipt setting described above, a number of other factors related to the configuration of your replication scheme could impact replication performance. As described in "Performance and recovery trade-offs" in *TimesTen to TimesTen Replication Guide*, you often have to weigh the ability to efficiently failover and recover a data store against replication performance.

Topics that might be of interest include:

- "Efficiency and economy" in *TimesTen to TimesTen Replication Guide*
- "Direct replication or propagation" in *TimesTen to TimesTen Replication Guide*

Check size of log buffer

As described in "Setting attributes for disk-based logging" in *TimesTen to TimesTen Replication Guide*, setting your log buffer too small may impact replication performance. Try setting the **LogBuffSize** DSN attribute to a larger size.

Check durability settings

You can improve replication performance by setting TRANSMIT NONDURABLE on the replication ELEMENT to eliminate the flush-log-to-disk operation from the replication cycle. See "Setting transmit durability on data store elements" in *TimesTen to TimesTen Replication Guide* for details.

Enabling the DURABLE COMMIT option in your replication scheme also impacts performance. See "DURABLE COMMIT" in *TimesTen to TimesTen Replication Guide* for more information.

Check for reads from log files

In some situations a “log reader,” such as a master replication agent ‘transmitter’ thread or a ttXlaNextUpdate call in an XLA application, may not be able to keep up with the update rate of the applications writing to the data store. Normally, replication and XLA readers get update records from the log buffer in memory. When the readers fall behind the application update rate, log files can accumulate on the disk until the backlog can be cleared. This forces the readers to read transactions from the log files on disk, which is much slower. Should you detect

reads from the log files, you may want to respond by decreasing the rate of application updates to that sustainable by the log readers.

Applications can monitor whether log readers are obtaining update records from log files on disk rather than from the log buffer in memory by tracking the SYS.MONITOR table entry LOG_FS_READS. For example, you can check the value of LOG_FS_READS for the data store, MASTERDSN, with the following **ttIsql** command:

```
% ttIsql -v1 -e "select log_fs_reads from monitor; quit;" -connStr dsn=MASTERDSN
```

If the LOG_FS_READS counter is increasing, the log readers are falling behind or clearing out a backlog in the log files.

For more complete monitoring of replication progress, create a simple shell script like the following:

```
!/bin/sh
trap exit 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DSN=$1

while [ 1 ] ; do
   date
   ttRepAdmin -receiver -list -connStr dsn=$DSN
   echo -n "Log reads from disk: "
   ttIsql -v1 -e "select log_fs_reads from monitor; quit;" -connStr dsn=$DSN
   echo
   ttRepAdmin -bookmark -connStr dsn=$DSN
sleep 15
done
```

Example 6.14 For example, you name the above script *monitorLog* and your replication scheme replicates from the MASTERDSN data store to the SUBSCRIBER1DSN data store. You can then check the status of the transaction log by entering:

```
$ monitorLog masterdsn
```

This generates output similar to the following:

```
Mon Aug  2 10:44:40  2004
Peer name         Host name                 Port    State  Proto
---------------  ------------------------ ------  ------- -----
SUBSCRIBER1DSN    MYHOST                    Auto    Start   12

Last Msg Sent Last Msg Recv Latency TPS     RecordsPS Logs
------------- ------------- ------- ------- --------- ----
00:00:05      -               -1.00      -1        -1    1

Log reads from disk: < 0 >

Replication hold LSN ..... 10/2656136
```

```
Last written LSN .......... 10/4015824
Last LSN forced to disk ... 10/3970152
```

The output from the script displays an updated status every 15 seconds until you Control-C to exit.

Following the date in the output in Example 6.14 is the name of the subscriber, its host, and so on. Next is latency and rate information, as well as the number of log files being retained on behalf of this subscriber. The specific meaning of each value is described in "From the command line: ttRepAdmin -receiver -list" in *TimesTen to TimesTen Replication Guide*. The main interest here is the 'Last Msg Sent' and 'Logs' values. The 'Last Msg Sent' value indicates the elapsed time since the last message was sent by the master to the subscriber and 'Logs' indicates how many log files behind the replication log reader is from the current log insertion point used by the writers (`Last written LSN`).

Normally the 'Logs' value should be '1', as shown in Example 6.14. A steadily increasing 'Logs' value indicates latency is increasing and eventually log reads are satisfied from disk.

Note: If the **LogBuffSize** is larger than the **LogFileSize**, an increase in the 'Logs' value does not necessarily mean the log readers are reading from the log files. This is because the log manager does not allow more than one log file's worth of data to be outstanding before writing it to the file system. After the log manager writes the data, the data remains in the log buffer to be read directly by the log readers. So, when the **LogBuffSize** is larger than the **LogFileSize**, the 'Logs" value alone may not be the best measure of whether log readers are reading from memory or from disk.

The output from:

```
ttRepAdmin -bookmark -connStr dsn=$DSN
```

displays the number of the log files and the location of the bookmarks set by the log manager, as described in "From the command line: ttRepAdmin -bookmark" in *TimesTen to TimesTen Replication Guide*. The difference between the *Replication hold LSN* and the *last written LSN* indicates the number of records in the transaction log that have not yet been transmitted to the subscribers. A steady increase in the difference between these values is another indication that replication latency is increasing and log file reads are likely to occur.

Example 6.15 In this example, assume the **LogBuffSize** is 16MB than the **LogFileSize** is 8MB. The following output indicates the log reader is approximately 1.8 MB behind the capacity of the log buffer and must read from the log files, 14 and 15.

```
Peer name         Host name                Port    State   Proto
---------------  ----------------------- ------  ------- -----
SUBSCRIBER1DSN    MYHOST                   Auto    Start   12
```

```
Last Msg Sent Last Msg Recv Latency TPS     RecordsPS Logs
------------- ------------- ------- ------- --------- ----
00:00:03      -               -1.00      -1        -1    4

Log reads from disk: <20>

Replication hold LSN ..... 14/7007464
Last written LSN ......... 17/465336
Last LSN forced to disk ... 17/456152
```

Problems using ttRepAdmin

This section includes the following topics:

- Problems using ttRepAdmin -duplicate
- Returns ‘Must specify -scheme’ error

Problems using ttRepAdmin -duplicate

If you connected to your new subscriber DSN before running **ttRepAdmin** `-duplicate`, the data store has already been created. In this situation, `-duplicate` returns:

```
Error : Restore not done : The datastore already exists.
Unable to restore datastore locally
```

Confirm the existence of the data store by running **ttStatus** and checking to see if the data store is in the returned list. If the new subscriber data store exists, destroy it and try **ttRepAdmin** `-duplicate` again:

```
> ttDestroy /tmp/newstore
```

```
> ttRepAdmin -dsn newstoreDSN -duplicate -name newstore
-from masterds -host "server1"
```

If you have made an error entering the subscriber data store name or host name in the replication scheme, you may see something like the following:

```
Unable to swap datastore locally
No receiver NEWSTORE on SERVER2 found to swap with
```

Returns ‘Must specify -scheme’ error

If you have more than one scheme specified in your TTREP.REPLICATIONS table, some **ttRepAdmin** commands may return the error:

```
Must specify -scheme to identify which replication scheme to use
```

To check the names of the replication schemes used by your data store, use the **ttIsql** utility to connect, and enter:

```
Command> SELECT * from TTREP.REPLICATIONS;
```

Example 6.16 This example shows that two replications schemes, REPSCHEME1 and REPSCHEME2, are assigned to the data store associated with *subDSN*. In this case, it is necessary to use the **ttRepAdmin** `-scheme` option.

```
> ttIsql -connStr "dsn=subDSN"
```

```
Command> SELECT * from TTREP.REPLICATIONS;
< REPSCHEME1      , REPL                , C, 0, 0, -1 >
< REPSCHEME2      , REPL                , C, 0, 0, -1 >
2 rows found.
Command> exit
```

```
> ttRepAdmin -dsn subDSN -receiver -list -scheme REPSCHEME1
Peer name         Host name                 Port    State  Proto
---------------  ------------------------ ------  ------- -----
SUBSCRIBER1        MYHOST                   Auto   Start      10

Last Msg Sent Last Msg Recv Latency TPS     RecordsPS Logs
------------- ------------- ------- ------- --------- ----
0:01:12       -               19.41       5        52    2
```

Problems with conflict checking

This section includes the following topics:

- Column cannot be used for replication timestamp
- Timestamp does not exist
- Conflict reporting slows down replication

Column cannot be used for replication timestamp

When attempting to set CHECK CONFLICTS for an element in a CREATE REPLICATION statement, you may encounter an error similar to the following:

```
8004: Column REPL.TABS.TS cannot be used for replication timestamp
checking if in an index or added by ALTER TABLE; and must be binary(8)
with NULL values allowed.
```

In this situation, check:

- That the timestamp column in the specified table is a nullable column of type BINARY(8). In the above example, the TS column in the REPL.TAB table should have a type of BINARY(8).
- The timestamp column is defined in the original CREATE TABLE statement, rather than added later using ALTER TABLE.

Timestamp does not exist

You may receive an error similar to the following:

```
2208: Column TS does not exist in table.
```

In this situation, confirm that you have specified the correct name for the timestamp COLUMN in the CHECK CONFLICTS clause and that it exists in the specified table.

Also, make sure the timestamp column is not part of a primary key or index.

Conflict reporting slows down replication

If you have configured replication to check conflicts, TimesTen sends reports to the local host. You can also configure a report file. See "Diagnostics through SNMP Traps" in *Oracle TimesTen In-Memory Database Error Messages and SNMP Traps*.

If there is a large number of conflicts in a short period of time, subscriber performance can slow down because of the reporting requirements. You can use store attributes in the CREATE REPLICATION or ALTER REPLICATION statements to suspend and resume conflict reporting at specified rates of conflict:

- CONFLICT REPORTING SUSPEND AT *rate*
- CONFLICT REPORTING RESUME AT *rate*

Information about conflicts that occur while reporting is suspended cannot be retrieved.

See "Conflict reporting" in *TimesTen to TimesTen Replication Guide*.

Index

O

P

Q

R

S

T

U

V

X