Oracle® TimesTen In-Memory Database

Open Source Languages Support Guide

Release 22.1
F35401-01
November 2021

Oracle TimesTen In-Memory Database Open Source Languages Support Guide, Release 22.1

F35401-01

Contents

4 Support for Node.js

Preface

Oracle TimesTen In-Memory Database (TimesTen) is a relational database that is memory-optimized for fast response and throughput. The database resides entirely in memory at runtime and is persisted to the file system.

- Oracle TimesTen In-Memory Database in classic mode, or TimesTen Classic, refers to single-instance and replicated databases (as in previous releases).
- Oracle TimesTen In-Memory Database in grid mode, or TimesTen Scaleout, refers to a multiple-instance distributed database. TimesTen Scaleout is a grid of interconnected hosts running instances that work together to provide fast access, fault tolerance, and high availability for in-memory data.
- TimesTen alone refers to both classic and grid modes (such as in references to TimesTen utilities, releases, distributions, installations, actions taken by the database, and functionality within the database).
- TimesTen Cache is ideal for caching performance-critical subsets of an Oracle database into cache tables within a TimesTen database for improved response time in the application tier. Cache tables can be read-only or updatable. Applications read and update the cache tables using standard Structured Query Language (SQL) while data synchronization between the TimesTen database and the Oracle database is performed automatically.
- TimesTen Replication features, available with TimesTen Classic or TimesTen Cache, enable high availability.

TimesTen supports standard application interfaces JDBC, ODBC, and ODP.NET; Oracle interfaces PL/SQL, OCI, and Pro*C/C++; and the TimesTen TTClasses library for C++.

This document provides usage and reference information for TimesTen support of open source languages.

Audience

This guide is for application developers who access TimesTen through supported open source languages.

In addition to familiarity with the particular programming interface you use, you should be familiar with TimesTen, SQL (Structured Query Language), database operations, and ODBC.

Related documents

TimesTen documentation is available on the TimesTen documentation website.

Oracle Database documentation is also available on the Oracle documentation website. This may be especially useful for Oracle Database features that TimesTen supports but does not attempt to fully document, such as OCI and Pro*C/C++.

Conventions

TimesTen supports multiple platforms. Unless otherwise indicated, the information in this guide applies to all supported platforms. The term Windows applies to all supported Windows platforms. The term UNIX applies to all supported UNIX platforms. The term Linux is used separately. Refer to "Platforms and compilers" in *Oracle TimesTen In-Memory Database Release Notes* (`README.html`) in your installation directory for specific platform versions supported by TimesTen.

> **Note:**
>
> In TimesTen documentation, the terms "data store" and "database" are equivalent. Both terms refer to the TimesTen database.

This document uses the following text conventions:

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates code, commands, URLs, class names, function names, method names, attribute names, directory names, file names, text that appears on the screen, or text that you enter.
`italic monospace`	Italic monospace type indicates a placeholder or a variable in a code example for which you specify or use a particular value. For example: `LIBS = -L`*`timesten_home`*`/install/lib -ltten` Replace *`timesten_home`* with the path to the TimesTen instance home directory.
[ ]	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 alternative arguments.
. . .	An ellipsis (. . .) after an argument indicates that you may use more than one argument on a single command line.
% or $	The percent sign or dollar sign indicates the Linux or UNIX shell prompt, depending on the shell that is used.
#	The number (or pound) sign indicates the Linux or UNIX root prompt.

TimesTen documentation uses the following variables to identify path, file and user names.

Convention	Meaning
`installation_dir`	The path that represents the directory where TimesTen is installed.
`timesten_home`	The path that represents the home directory of a TimesTen instance.
`release` or *`rr`*	The first two parts in a release number, with or without the dot. The first two parts of a release number represent a major TimesTen release. For example, 221 or 22.1 represents TimesTen Release 22.1.

Convention	Meaning
DSN	The data source name (for the TimesTen database).

> **Note:**
>
> TimesTen release numbers are reflected in items such as TimesTen utility output, file names, and directory names, all of which are subject to change with every minor or patch release. The documentation cannot always be up to date. It seeks primarily to show the basic form of output, file names, directory names, and other code that may include release numbers. You can confirm the current release number by looking at *Oracle TimesTen In-Memory Database Release Notes* or executing the `ttVersion` utility.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

What's New

This section summarizes new features and functionality of TimesTen Release 22.1.

New features in Release 22.1.1.1.0

- In this release, TimesTen supports the following:
  - ODPI-C 4.2
  - Python through cx_oracle 8.2.1
  - Node.js through node-oracledb 5.2.0

1
Getting Started

This document discusses TimesTen In-Memory Database support for open source languages. This support applies to both TimesTen Classic and TimesTen Scaleout. In the current release, the languages supported are Python and Node.js.

This section discusses basic concepts and initial considerations in using these open source languages supported by TimesTen.

The following topics are covered here:

- ODPI-C and language-specific packages
- Platform support for open source languages
- Requirements for open source languages
- Restrictions for open source languages

> **Note:**
>
> Sample programs for the open source languages are available on GitHub. For an overview of what is available, refer to README.md at:
>
> https://github.com/oracle/oracle-timesten-samples/tree/master/
>
> There are additional README.md files with more information for each language, referenced later in this document.

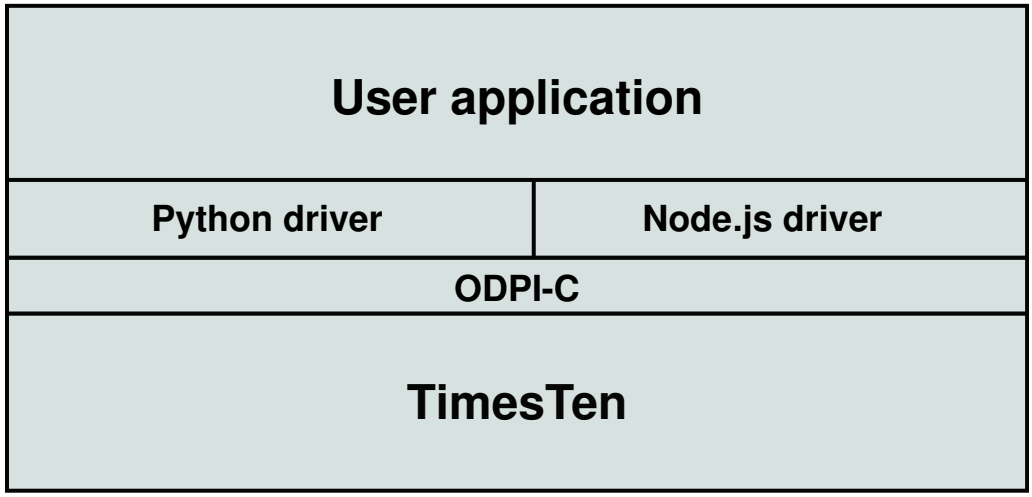
ODPI-C and language-specific packages

TimesTen support of open source languages is through the Oracle Database Programming Interface for C (ODPI-C). ODPI-C is an open source library from Oracle Corporation designed to simplify access to Oracle databases from a variety of programming languages. You can refer to `https://oracle.github.io/odpi/` for information.

For each language, there is an open source package, or driver, available through GitHub and maintained by Oracle. Terminology for these packages varies from language to language: for example, *extension module* for Python or *add-on* for Node.js. See the section for each supported language below for information about the corresponding package. ODPI-C is included when you download the driver for each language.

ODPI-C is a layer on top of the Oracle Call Interface (referred to in this document as OCI, but not to be confused with Oracle Cloud Infrastructure). Figure 1-1 shows the architecture.

Figure 1-1 TimesTen architecture for open source languages



Platform support for open source languages

Platform support for open source languages is of course limited to the platforms, or a subset of the platforms, supported by TimesTen, and may vary between languages.

TimesTen supports client/server access from Linux, macOS, or Windows, or direct access on Linux, for both Python and Node.js.

For the latest information about platform support for each language, refer to the applicable `README.md` file at:

https://github.com/oracle/oracle-timesten-samples/tree/master/languages/python

Or:

https://github.com/oracle/oracle-timesten-samples/tree/master/languages/nodejs

For more information about TimesTen platform support, including specific versions supported, refer to "Platforms and configurations" in *Oracle TimesTen In-Memory Database Release Notes*.

Requirements for open source languages

Using open source languages through ODPI-C for access to TimesTen databases requires use of the special OCI client library that is included with the Oracle Instant Client shipped with TimesTen software. Because you must use the Instant Client that is supplied with TimesTen, your library path must be set so that the TimesTen Instant Client directory (*tt_installation_dir*`/ttoracle_home/instantclient`) precedes the path to any Oracle Database libraries. The path is set appropriately when you use the TimesTen `ttenv` script (`ttenv.sh` or `ttenv.csh`), described in "Install TimesTen".

Restrictions for open source languages

Because ODPI-C is a layer on top of OCI, TimesTen restrictions for OCI also apply to ODPI-C. These restrictions are documented in "TimesTen restrictions and limitations" in *Oracle TimesTen In-Memory Database C Developer's Guide*.

In addition, these features are not supported:

- Application continuity
- Continuous query notification
- Call timeouts
- Session tagging
- Database management, including startup and shutdown

2
TimesTen Setup

This section discusses the following actions to set up TimesTen.

- Install TimesTen
- Set the TimesTen environment
- Define and manage the TimesTen database
- Configure connections for TimesTen

Install TimesTen

If you do not already have TimesTen, you can download it from TimesTen downloads.

> **Note:**
>
> Full support for open source languages requires a TimesTen release of 18.1.4.1.0 or higher.

There is a single ZIP file distribution for both TimesTen Classic and TimesTen Scaleout, but setup instructions differ from there:

- For prerequisites, environment setup, and installation for TimesTen Classic, refer to "Overview of the Installation Process in TimesTen Classic" and the applicable installation chapter in *Oracle TimesTen In-Memory Database Installation, Migration, and Upgrade Guide*.

  The installation chapters also describe how to create a TimesTen instance.
- For prerequisites, environment setup, and installation for TimesTen Scaleout, refer to "Prerequisites and Installation of TimesTen Scaleout" in *Oracle TimesTen In-Memory Database Scaleout User's Guide*.

  "Setting Up a Grid" in that document includes information about creating the initial management instance.

Set the TimesTen environment

After you have installed TimesTen and created a TimesTen instance, use the appropriate `ttenv` script (`ttenv.sh` or `ttenv.csh`) under *`timesten_home`*`/bin` to finish setting up the environment.

Define and manage the TimesTen database

For defining and managing a database for TimesTen Classic, refer to "Managing TimesTen Databases" in *Oracle TimesTen In-Memory Database Operations Guide*.

For defining and managing a TimesTen grid and database for TimesTen Scaleout, refer to "Setting Up the Membership Service", "Setting Up a Grid", and "Managing a Database" in *Oracle TimesTen In-Memory Database Scaleout User's Guide*.

Configure connections for TimesTen

Because TimesTen support of open source languages goes through OCI, you can connect to TimesTen from Python or Node.js using either the `tnsnames` or the easy connect naming method. For details on these methods, see "Connecting to a TimesTen database from OCI" in *Oracle TimesTen In-Memory Database C Developer's Guide*.

You may have a `tnsnames.ora` entry such as the following, for example, for connecting to database `sampledb`. Then you would reference the TNS name, `sampledbconn`, for the connection string in your code.

```
sampledbconn =(DESCRIPTION=(CONNECT_DATA = (SERVICE_NAME = sampledb)(SERVER =
timesten_direct)))
```

Or, to avoid using `tnsnames.ora`, you can use an easy connect string, such as `myhost/sampledb:timesten_direct`, directly in your code for the connection string.

Use `timesten_direct` for a direct connection to TimesTen or `timesten_client` for a client/server connection.

See "Configure connections to TimesTen in Python" and "Configure connections to TimesTen in Node.js" for examples using TNS names and easy connect strings in your code.

You will need a TimesTen database user in order to connect to a TimesTen database. For information about creating users in TimesTen, see "CREATE USER" in *Oracle TimesTen In-Memory Database SQL Reference*.

The following example creates a TimesTen internal user `appuser` with password `appuser` and grants `CREATE SESSION` and `CREATE TABLE` privileges so `appuser` can create a user session in connecting to the database and then create a database table.

```
CREATE USER appuser IDENTIFIED BY "appuser";

GRANT CREATE SESSION TO appuser;
GRANT CREATE TABLE TO appuser;
```

3
Support for Python

This section covers topics about Python and TimesTen support:

- About Python
- Setup for Python
- Additional information for Python

About Python

Python is an interpreted, high-level, general purpose language that includes support for object-oriented programming and a robust standard library. Simple syntax rules make Python code easy to read and support.

Many platforms come with Python installed. Enter "python" at the command prompt to confirm. If it is installed, its response will include the version number.

You can find information about Python and download it from `https://www.python.org`

For information about Python versions supported by TimesTen, refer to the `README.md` file at `https://github.com/oracle/oracle-timesten-samples/tree/master/languages/python`

The Python extension module for access to Oracle or TimesTen databases from Python applications is `cx_Oracle`. This module conforms to the Python database API 2.0 specification. There is general information about the module at `https://github.com/oracle/python-cx_Oracle`.

For information about versions of the Python extension module supported by TimesTen, refer to the `README.md` file.

Setup for Python

This section covers these setup actions to use Python with TimesTen.

- Install the Python extension module
- Configure connections to TimesTen in Python

Install the Python extension module

You must have internet access to install the Python extension module. You may have to set the `http_proxy` and `https_proxy` environment variables to indicate your proxy server(s) in order to access the internet outside of a firewall.

There are several approaches for downloading the Python extension module. Choose the approach that is most suitable for your setup and environment.

- Download source files from GitHub, using `git clone`. This requires `git` to be installed on your system. Refer to the instructions at https://cx-oracle.readthedocs.io/en/latest/user_guide/installation.html#install-using-github

  > **Tip:**
  >
  > – Because the setup process requires access to the system `lib` directory, administrative privilege is required.
  >
  > – In addition to Python itself, your system must have the Python header files, which requires that the corresponding `python[3]-devel` development package is installed.

- Download prebuilt packages using the Python Pip package manager. Refer to the instructions at https://cx-oracle.readthedocs.io/en/latest/user_guide/installation.html#quick-start-cx-oracle-installation as well as to platform-specific sections of that document, as appropriate.

  This approach provides prebuilt binaries but downloads and compiles source code as necessary.

  > **Tip:**
  >
  > In the event that source code must be compiled, you must have package `python[3]-devel` installed.

- Download prebuilt binaries through a platform-specific package manager such as RPM.

  You can use the RPM package manager through a utility such as yum.

  The `cx_Oracle` RPM packages are available from the Oracle Linux yum server (http://yum.oracle.com/). This is discussed at https://cx-oracle.readthedocs.io/en/latest/user_guide/installation.html#installing-cx-oracle-rpms-on-oracle-linux

  > **Tip:**
  >
  > Using this approach requires Oracle Instant Client for successful installation. If you do not yet have Instant Client on your system, it will be provided. But for runtime operations, the Instant Client version that comes with TimesTen is required, as discussed in "Requirements for open source languages".

Information for all scenarios is available at https://cx-oracle.readthedocs.io/en/latest/user_guide/installation.html

> **Note:**
>
> In all three cases, ODPI-C is included. For the `git clone` approach, any dependencies, including ODPI-C, are automatically cloned and built.

Configure connections to TimesTen in Python

See "Configure connections for TimesTen" for information about using the Oracle `tnsnames.ora` or easy connect mechanisms to set up connections to your database.

This example uses a TNS name for a Python connection string:

```
connection = cx_Oracle.connect("appuser", "appuser", "sampledbconn")
```

This example uses an easy connect string that specifies a direct connection to TimesTen:

```
connection = cx_Oracle.connect("appuser", "appuser", "myhost/sampledb:timesten_direct")
```

Additional information for Python

This section covers these topics:

- Type mappings for Python applications
- Failure modes for Python
- Python sample: Connect to TimesTen and execute SQL
- TimesTen Python samples to download

Type mappings for Python applications

Table 3-1 documents mappings between `cx_Oracle` types (as available through the language extension module) and TimesTen SQL types.

> **Note:**
>
> Additional TimesTen SQL types can be mapped to these `cx_Oracle` types as a result of TimesTen implicit data type conversions. See "Data type conversion" in *Oracle TimesTen In-Memory Database SQL Reference*.

For the latest information about data type support in the current release, refer to the `README.md` file at `https://github.com/oracle/oracle-timesten-samples/tree/master/languages/python`

Table 3-1 Type mappings for cx_Oracle

cx_Oracle type	TimesTen type
`cx_Oracle.BINARY`	`BINARY`, `VARBINARY`

Table 3-1 (Cont.) Type mappings for cx_Oracle

cx_Oracle type	TimesTen type
cx_Oracle.BLOB	BLOB
cx_Oracle.CLOB	CLOB
cx_oracle.CURSOR	REF CURSOR
cx_Oracle.DATETIME	DATE
cx_Oracle.FIXED_CHAR	CHAR
cx_Oracle.FIXED_NCHAR	NCHAR (In TimesTen, this type is UTF-16 only.)
cx_Oracle.NATIVE_FLOAT	BINARY_DOUBLE, BINARY_FLOAT
cx_Oracle.NCHAR	NVARCHAR2 (In TimesTen, this type is UTF-16 only.)
cx_Oracle.NCLOB	NCLOB
cx_Oracle.NUMBER	NUMBER, TT_BIGINT, TT_INTEGER, TT_SMALLINT, TT_TINYINT
cx_Oracle.ROWID	ROWID
cx_Oracle.STRING	VARCHAR2
cx_Oracle.TIMESTAMP	TIMESTAMP, TT_TIMESTAMP

Failure modes for Python

Be aware of the following:

- Error messages may differ slightly from those produced by Oracle under the same conditions.
- For the Python method `Cursor.executemany()`, TimesTen ignores the `batcherrors` option and therefore supports only the default `batcherrors=false` setting. An error is always returned if any row in the batch encounters an error. (By contrast, Oracle Database supports `batcherrors=true`, in which case success is always returned if any row in the batch is successful.) Also, against a TimesTen database, if multiple errors are encountered in a batch, TimesTen makes no attempt to order the errors as they would be ordered if encountered against an Oracle database.

Python sample: Connect to TimesTen and execute SQL

This sample program connects to a TimesTen database, creates a table named `employees`, inserts three rows into the table, selects and displays the rows, drops the table, and disconnects from the database.

```
##
## simple.py
##
from __future__ import print_function
import cx_Oracle
import AccessControl

def run():
```

```
  try:
    credentials = AccessControl.getCredentials("simple.py")
    connection = cx_Oracle.connect(credentials.user, credentials.password,
                 credentials.connstr)

    cursor = connection.cursor()
    cursor.execute("""
      CREATE TABLE employees(first_name VARCHAR2(20), last_name VARCHAR2(20))""")
    print("Table has been created")
    values = [["ROBERT", "ROBERTSON"], ["ANDY", "ANDREWS"], ["MICHAEL",
              "MICHAELSON"]]
    cursor.executemany("INSERT INTO employees VALUES (:1, :2)", values)
    print("Inserted ", len(values), "employees into the table")
    cursor.execute("""
      SELECT first_name, last_name FROM employees""")
    for fname, lname in cursor:
      print("Selected employee:", fname, lname)
    cursor.execute("DROP TABLE employees")
    print("Table has been dropped")
    cursor.close()
    connection.close()
    print("Connection has been released")

  except Exception as e:
    # Something went wrong
    print("An error occurred", str(e))

run()
```

Here are the results:

```
% python3 simple.py -u username -p password
Table has been created
Inserted  3 employees into the table
Selected employee: ROBERT ROBERTSON
Selected employee: ANDY ANDREWS
Selected employee: MICHAEL MICHAELSON
Table has been dropped
Connection has been released
```

TimesTen Python samples to download

TimesTen sample programs for Python through `cx_Oracle` are available at https://github.com/oracle/oracle-timesten-samples/tree/master/languages/python

Refer to `README.md` at that location for information.

4
Support for Node.js

This section covers topics about Node.js and TimesTen support:

- About Node.js
- Setup for Node.js
- Additional information for Node.js

About Node.js

Node.js is an open source Javascript runtime environment you can use to build scalable network applications and tools, such as web servers. It includes "modules" to reduce the complexity of writing server applications by handling basic functions such as file system I/O, networking, data streams, and cryptography.

You can find information about Node.js and download it from `https://nodejs.org`.

For information about Node.js versions supported by TimesTen, refer to the `README.md` file at `https://github.com/oracle/oracle-timesten-samples/tree/master/languages/nodejs`

The Node.js installation includes the package manager `npm`. Use this to install the Node.js add-on, as described below in the section about that add-on.

The Node.js add-on for access to Oracle or TimesTen databases from Node.js applications is `node-oracledb`. There is general information about the add-on at `https://oracle.github.io/node-oracledb/`.

For information about versions of the Node.js add-on supported by TimesTen, refer to the `README.md` file.

Setup for Node.js

This section covers these setup actions to use Node.js with TimesTen.

- Install the Node.js add-on
- Configure connections to TimesTen in Node.js

Install the Node.js add-on

You must have internet access to install the Node.js ad-on. You may have to set the `http_proxy` and `https_proxy` environment variables to indicate your proxy server(s) in order to access the internet outside of a firewall.

There are several approaches for downloading the Node.js add-on. Choose the approach that is most suitable for your setup and environment.

- Download source files from GitHub, using `git clone` (requiring `git` to be installed on your system), then install the driver using the NPM package manager.

You can install `node-oracledb` by downloading source files from GitHub. Once you have downloaded and extracted Node.js, you can optionally use the NPM package manager, located in the Node.js `bin` directory, to install a prebuilt `node-oracledb` binary. (Add the Node.js `bin` directory to your path.)

Refer to the installation instructions at `https://oracle.github.io/node-oracledb/INSTALL.html#-3-node-oracledb-installation-instructions` under "Source code from GitHub".

- Download prebuilt packages using the NPM package manager.

  Refer to the installation instructions.

  This approach provides prebuilt binaries only. If prebuilt binaries are not available, you must compile the source code. In order to do this, you must have an Oracle Instant Client installed. The one provided with TimesTen will suffice for this.

- Download prebuilt binaries through a platform-specific package manager such as RPM.

  The `node-oracledb` RPM packages are available from the Oracle Linux yum server (`http://yum.oracle.com/`).

  Refer to the installation instructions.

> **Tip:**
>
> Using this approach requires Oracle Instant Client for successful installation. If you do not yet have Instant Client on your system, it will be provided. But for runtime operations, the Instant Client version that comes with TimesTen is required, as discussed in "Requirements for open source languages".

Information for all scenarios is available at `https://oracle.github.io/node-oracledb/INSTALL.html`.

> **Note:**
>
> In all three cases, ODPI-C is included. For the `git clone` approach, any dependencies, including ODPI-C, are automatically cloned and built.

Configure connections to TimesTen in Node.js

See "Configure connections for TimesTen" for information about using the Oracle `tnsnames.ora` or easy connect mechanisms to set up connections to your database.

This example uses a TNS name for a Node.js connection string:

```
// Get connection
function connect(cb) {
    oracledb.getConnection({
      user          :  "appuser",
      password      :  "appuser",
      connectString :  "sampledbconn"
    } ,
```

```
    cb);
}
```

This example uses an easy connect string that specifies a direct connection to TimesTen:

```
// Get connection
function connect(cb) {
    oracledb.getConnection({
      user          : "appuser",
      password      : "appuser",
      connectString : "myhost/sampledb:timesten_direct"
    } ,
    cb);
}
```

Additional information for Node.js

This section covers these topics:

- Type mappings for Node.js applications
- Failure modes for Node.js
- Node.js sample: Connect to TimesTen
- TimesTen Node.js samples to download

Type mappings for Node.js applications

Table 4-1 documents mappings between `node-oracledb` types as available through the language add-on and TimesTen SQL types.

> **Note:**
>
> Additional TimesTen SQL types can be mapped to these Oracle Database types and `node-oracledb` alias types as a result of TimesTen implicit data type conversions. See "Data type conversion" in *Oracle TimesTen In-Memory Database SQL Reference*.

For the latest information about data type support in the current release, refer to the `README.md` file at `https://github.com/oracle/oracle-timesten-samples/tree/master/languages/nodejs`

Table 4-1 Type mappings for node-oracledb

Node.js type	Oracle Database type	node-oracledb alias type	TimesTen type
Number	oracledb.DB_TYPE_BINARY_DOUBLE	n/a	BINARY_DOUBLE
Number	oracledb.DB_TYPE_BINARY_FLOAT	n/a	BINARY_FLOAT
Lob	oracledb.DB_TYPE_BLOB	oracledb.BLOB	BLOB
String	oracledb.DB_TYPE_CHAR	n/a	CHAR

Table 4-1 (Cont.) Type mappings for node-oracledb

Node.js type	Oracle Database type	node-oracledb alias type	TimesTen type
Lob	oracledb.DB_TYPE_CLOB	oracledb.CLOB	CLOB
ResultSet	oracledb.DB_TYPE_CURSOR	oracledb.CURSOR	REF CURSOR
Date	oracledb.DB_TYPE_DATE	oracledb.DATE	DATE
Number	oracledb.DB_TYPE_INTEGER	n/a	NUMBER, TT_BIGINT, TT_INTEGER, TT_SMALLINT, TT_TINYINT
String	oracledb.DB_TYPE_NCHAR	n/a	NCHAR (In TimesTen, this type is UTF-16 only.)
Lob	oracledb.DB_TYPE_NCLOB	oracledb.NCLOB	NCLOB
Number	oracledb.DB_TYPE_NUMBER	oracledb.NUMBER	NUMBER, TT_BIGINT, TT_INTEGER, TT_SMALLINT, TT_TINYINT
String	oracledb.DB_TYPE_NVARCHAR	n/a	NVARCHAR2 (In TimesTen, this type is UTF-16 only.)
Buffer	oracledb.DB_TYPE_RAW	oracledb.BUFFER	BINARY, VARBINARY
String	oracledb.DB_TYPE_ROWID	n/a	ROWID
Date	oracledb.DB_TYPE_TIMESTAMP	n/a	TIMESTAMP, TT_TIMESTAMP
String	oracledb.DB_TYPE_VARCHAR	oracledb.STRING	VARCHAR

Failure modes for Node.js

Be aware of the following:

- Error messages may differ slightly from those produced by Oracle under the same conditions.
- For the Node.js method `connection.executeMany()`, TimesTen ignores the `batcherrors` option and therefore supports only the default `batcherrors=false` setting. An error is always returned if any row in the batch encounters an error. (By contrast, Oracle Database supports `batcherrors=true`, in which case success is always returned if any row in the batch is successful.) Also, against a TimesTen database, if multiple errors are encountered in a batch, TimesTen makes no attempt to order the errors as they would be ordered if encountered against an Oracle database.

Node.js sample: Connect to TimesTen

This sample program connects to a TimesTen database, creates a table named `employees`, inserts three rows into the table, selects and displays the rows, drops the table, and disconnects from the database.

```
//
// simple.js
//
var oracledb      = require('oracledb');
var accessControl = require('./AccessControl');

async function run() {

  let connection;

  try {
    let credentials   = accessControl.getCredentials("simple.js");
    connection = await oracledb.getConnection({
      user: credentials['-u'], password: credentials['-p'], connectString:
            credentials['-c']
    });

    let result;

    await connection.execute("CREATE TABLE employees(first_name VARCHAR2(20),
          last_name VARCHAR2(20))");
    console.log("Table has been created");
    const values = [["ROBERT", "ROBERTSON"], ["ANDY", "ANDREWS"], ["MICHAEL",
                   "MICHAELSON"]];
    await connection.executeMany("INSERT INTO employees VALUES(:1, :2)", values);
    console.log("Inserted", values.length, "employees into the table");
    result = await connection.execute("SELECT first_name, last_name FROM
             employees");
    result.rows.forEach(function(row){
    console.log("Selected employee:", row[0], row[1]);
    });
    await connection.execute("DROP TABLE employees");
    console.log("Table has been dropped");
  }
  catch (err) {
    console.log(err);
  }
  finally {
    if (connection) {
      try {
        connection.close();
        console.log("Connection has been released");
      }
      catch (err) { console.log(err); }
    }
  }
}

run();
```

Here are the results:

```
% node simple.js -u username -p password
Table has been created
Inserted 3 employees into the table
Selected employee: ROBERT ROBERTSON
Selected employee: ANDY ANDREWS
Selected employee: MICHAEL MICHAELSON
Table has been dropped
Connection has been released
```

TimesTen Node.js samples to download

TimesTen sample programs for Node.js through `node-oracledb` are available under https://github.com/oracle/oracle-timesten-samples/tree/master/languages/nodejs

Refer to `README.md` at that location for information.