Easysoft ODBC-Oracle Driver

Performance Tuning Guide

Introduction

The Easysoft ODBC-Oracle Driver is configured to be the fastest driver across a broad range of database requirements as installed out-of-the-box. As with any database software however, the performance of the driver can be greatly enhanced by tuning the driver configuration to match your specific requirements.

This document outlines a few simple steps you can take which in the right circumstances can give dramatic reductions in query response times.

For advanced optimization, Easysoft provide a range of services to help you to obtain maximum performance from your Oracle® investment. For further details contact sales@easysoft.com.

Note: Some of the performance optimization techniques described in this guide are reliant on the OCI client libraries. They are therefore applicable to the OCI version of the Oracle ODBC driver, but not the WP version, which does not use Oracle client software to access Oracle.

Contents

Version 3.3
Issued: 15 October 2020

Note: Some of the suggestions in this guide involve changing Easysoft ODBC-Oracle Driver attributes. For boolean attributes, set the attribute value to 0 (OFF) or 1 (ON) if you are changing the attribute in the odbc.ini file or a DSN-less connection. If you are changing the attribute in the Easysoft ODBC-Oracle Driver dialog box, uncheck (OFF) or check (ON) the attribute. For more information about changing Easysoft ODBC-Oracle Driver attribute values, see the Easysoft ODBC-Oracle Driver documentation.

Retrieving Multiple Rows

Suggestion: Increase OCI_ATTR_PREFETCH_ROWS

Explanation: OCI_ATTR_PREFETCH_ROWS specifies the number of rows to return from a single round trip "fetch" call made to the server. The client will store these rows and use them for subsequent SQLFetch calls. The default value for OCI_ATTR_PREFETCH_ROWS is 10. Increasing this value can reduce the number of round trip network calls to the server needed to return result sets from the server at the expense of greater memory use.

Retrieving Large Recordsets

Suggestion: Set OCI_ATTR_PREFETCH_MEMORY

Explanation: OCI_ATTR_PREFETCH_MEMORY has exactly the same effect as the OCI_ATTR_PREFECT_ROWS setting described above. It is only different in that you specify the amount of memory to use instead of the number of rows, so the number of rows returned in each round trip to the server is the amount of memory allocated by OCI_ATTR_PREFECT_ROWS divided by the size a row.

This value overrides OCI_ATTR_PREFETCH_ROWS.

By default, OCI_ATTR_PREFETCH_MEMORY is not set.

Oracle Synonyms

Suggestion: Set ENABLE_SYNONYMS to OFF

Explanation: If set to ON this parameter will return synonyms in all metadata calls. If you do not need to see synonyms, leave ENABLE_SYNONYMS set to its default value OFF.

Version 3.3
Issued: 15 October 2020

Metadata Calls

Suggestion: Set ENABLE_USER_CATALOG to ON

Explanation: Metadata calls will normally return results for all objects in the database. Many ODBC applications will never need this amount of catalog data. Setting ENABLE_USER_CATALOG to ON reduces the number of rows returned by SQLTables calls. It does this by retrieving metadata from the current Oracle user's catalog rather than the catalog for all users.

The default action for the Easysoft ODBC-Oracle Driver is to retrieve metadata for all users. If you want to restrict the metadata returned and you access Oracle by using a DSN-less connection or a data source configured in the odbc.ini file, you need to set the value of ENABLE_USER_CATALOG to ON. Note that for data sources configured with the Easysoft ODBC-Oracle Driver dialog box, ENABLE_USER_CATALOG is set to ON by default.

Large Binary Objects

Suggestion: Set NO_LOBS to ON

Explanation: If you are using version 8 of the Oracle client software and you do not need to insert or update large objects (BLOBs or CLOBS), set NO_LOBS to ON to increase Easysoft ODBC-Oracle Driver performance.

For the version 8 client, the default method used to insert or update LOBs requires the Easysoft ODBC-Oracle Driver to check for LOBs to decide if it needs to create them. This introduces a performance overhead because every insert, update or delete statement has to create a background query to determine what data types are in use. If you enable the NO_LOBS setting, the Easysoft ODBC-Oracle Driver uses an alternative method of inserting or updating LOBs—one that uses SQLPutData to insert the data in parts and does not require the driver to check for LOBs. The disadvantage of this method is that in Oracle releases earlier than version 9, there is a restriction on the amount of data that can be inserted in a part insert. This means that enabling NO_LOBS can cause LOB inserts or updates to fail for those earlier versions of the database.

The NO_LOBS setting is only applicable to the version 8 Oracle client.

Version 3.3
Issued: 15 October 2020

Statement Parsing

Suggestion: Set NO_PARSE to ON

Explanation: Setting NO_PARSE to ON stops the Easysoft ODBC-Oracle Driver from preparsing SQL passed to SQLPrepare and SQLExecDirect to convert ODBC escape sequences and parameter markers (?). Enabling this option provides a small speed increase but does prevent your application from using ODBC escapes sequences and parameter markers. By default, the Easysoft ODBC-Oracle Driver preparses SQL statements passed to SQLPrepare and SQLExecDirect.

Statement Caching

Suggestion: Enable Oracle statement caching

Explanation: Oracle statement caching establishes and manages a cache of statements within a session. It improves performance by efficiently using prepared cursors on the Oracle server and eliminating repetitive statement parsing. To enable statement caching, use the STATEMENT_CACHING option to define the size of the required cache. The default value 0 disables statement caching. For more information about Oracle statement caching, see your Oracle documentation.

Connection Pooling

Suggestion: Use unixODBC connection pooling

Explanation: Connection pooling allows an application to reuse a connection that was previously created and since closed. Connection pooling can significantly speed up applications that repeatedly open and close ODBC connections. The connection pool is maintained by the unixODBC driver manager. Enable connection pooling in unixODBC when an application running in a single process will be repeatedly opening and closing connections to the same data source.

By default, connection pooling is disabled. To enable connection pooling, make the following changes to the odbcinst.ini file. Add `Pooling = Yes` to the [ODBC] section and then add a `CPTimeout` value to the driver section. For example:

```
[ODBC]
Pooling    = Yes

[ORACLE]
Description             = Easysoft ODBC-Oracle Driver
Driver          = /usr/local/easysoft/oracle/libesoracle.so
Setup           = /usr/local/easysoft/oracle/libesoraclesetup.so
FileUsage               = 1
UsageCount              = 2
CPTimeout  = 120
```

`CPTimeout` defines the time in seconds that unused connections remain in the pool before being dropped by the driver manager.

Once connection pooling is enabled, calls to SQLDisconnect do not actually result in a SQLDisconnect call in the Easysoft ODBC-Oracle Driver. This means that while the process is still running, the different connections stay open. This increases the total number of open connections at any one time and therefore has an impact on available database resources.

Note that `CPTimeout` is only checked on SQLDriverConnect, SQLConnect or SQLDisconnect calls.

For more information about the connection pooling mechanism, see the Easysoft ODBC-ODBC Bridge Performance White Paper, which is available from the Easysoft ODBC-ODBC Bridge section of the Easysoft web site.

Remove Driver Manager

Suggestion: Link your application to the ODBC driver

Explanation: You can gain some speed increases by linking you application directly to the Easysoft ODBC-Oracle Driver. You therefore eliminate any performance overhead introduced by the driver manager. However, by doing this you lose functionality provided by the driver manager. For example, the driver manager:

- Lets you choose a data source from a list and dynamically loads the correct ODBC driver for you.
- Allows your application to work with other ODBC drivers.
- Provides one central place for all your ODBC driver and data source configuration information.

odbc.ini File

Suggestion: Put regularly used data sources at the top of the file

Explanation: For each connection the driver has to serially read the odbc.ini file to find the connection attributes. Unused DSN definitions should therefore be removed and remaining definitions prioritized with the most frequently used at the beginning of the file.

ODBC Data Source

Suggestion: Use DSN-less connections

Explanation: Connection details are stored in odbc.ini and can therefore be easily changed without recompiling your application. The drawback is that the driver has first to find the odbc.ini file and then read the parameters. Using DSN-less connections compiles the connection details into the application, removing the need to read the odbc.ini file for every connection.

tnsnames.ora File

Suggestion: Set the database in the odbc.ini file to the Oracle connection string.

Explanation: Making the Database name equal to the database definition in tnsnames.ora will improve connection time, although it can make the odbc.ini file a little hard to read.

odbc.ini file :

```
[ORACLE]
Driver              = ORACLE
Database
=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)
(HOST=oracle)(PORT=1521)))(CONNECT_DATA=
(SERVER=DEDICATED)(SERVICE_NAME=nineone.oracle)))
User                = username
Password     = password
```

Note: The Database entry must all be on one line.

Top Five ODBC Programming Tips

1. Bind column data (see SQLBindCol) and where possible use arrays (see SQL_ATTR_ROW_ARRAY_SIZE).
2. Reuse prepared statements with parameters (see SQLPrepare and SQLBindParameter) and where possible use arrays (see SQL_ATTR_PARAMSET_SIZE).
3. Avoid using cursors other than forward-only unless you really need them.
4. Cache results of metadata calls like SQLDescribeCol as they can be expensive.
5. Reuse connections where possible.

Contact Information

For Oracle optimisation services, email sales@easysoft.com
or call +44 (0)1937 860000.

Easysoft Limited
Thorp Arch Grange
Thorp Arch
Wetherby
LS23 7BA

Version 3.3
Issued: 15 October 2020