

Oracle Database Easy Connect Plus

Configurable Database Connection Syntax for applications using Oracle Database

TECHNICAL BRIEF / AUGUST 8, 2024

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without the prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remain at the sole discretion of Oracle.

Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

TABLE OF CONTENTS

Easy Connect Plus Overview	4
Easy Connect Background	4
Easy Connect Plus Syntax	4
Protocol	5
Hosts	5
Ports	5
Service Name	6
Server Type	6
Instance Name	6
Parameters	6
Parameters	6
Security Parameters	6
Proxy Parameters	7
Description Parameters	7
DRCP Parameters	9
Implicit Connection Pooling Parameters	10
Examples	10
Conclusion	15

EASY CONNECT PLUS OVERVIEW

Oracle Database 19c introduced a significantly enhanced connection string syntax called Easy Connect Plus. For Oracle Database 21c, you can add DRCP connection parameters to the Easy Connect Plus syntax.

Easy Connect Plus makes it easier to use features such as TLS connections, wallets, load balancing, connection timeouts, and network buffer size tuning. It reduces the need for external configuration files such as *sqlnet.ora* and *tnsnames.ora*. For example, to access Oracle Cloud Database services via mutual TLS (mTLS) for secure communication, you can include the wallet location in the Easy Connect Plus string and do away with these configuration files.

Easy Connect Plus can be used in [JDBC](#)¹, C-based Oracle Call Interface (OCI), Oracle C++ Call Interface (OCCI), and .NET applications. It is also usable in applications and drivers built on OCI, such as the Oracle Database drivers for Python, Node.js, PHP, Ruby, and Go.

EASY CONNECT BACKGROUND

Easy Connect is a syntax currently used by Oracle programs and applications when connecting to an Oracle Database service. At its simplest, an Easy Connect URL string includes a hostname where the Oracle Database is running, an optional port, and the database service name:

```
host[:port] [/ [service_name]]
```

For example, when Oracle Database is running on a host with a name, say, *mydbhost.example.com* with the default Pluggable Database service "orclpdb1", then a typical connection in SQL*Plus would be:

```
SQL> connect scott/pw@mydbhost.example.com/orclpdb1
```

Or in Python:

```
connection = oracledb.connect(user='scott',  
password='password', dsn='mydbhost.example.com/orclpdb1')
```

EASY CONNECT PLUS SYNTAX

Easy Connect Plus syntax is a superset of Easy Connect that supports more connection options. The syntax is available in all the Oracle Database drivers (*JDBC*, *ODP.NET*², *OCI*, *C++ based OCCI*, *python-oracledb*, *node-oracledb*, etc.) that use Oracle Client 19c (and later versions) and connect to Oracle Database 11.2 or later.

In addition to the normal Easy Connect features listed in the earlier section, Easy Connect Plus provides the following capabilities:

- It allows **TCPS connections** and security-related properties in the URL. For example, `tcps://salesserver1:1521/sales.us.example.com?ssl_server_cert_dn="cn=sales,cn=OracleContext,dc=us,dc=example,dc=com"`

¹ Java Database Connectivity (JDBC) is Java's standard database access API developed by Oracle

² Oracle Data Provider for .NET – ODP.NET

- It allows **connection properties** in the URL, which can enable High Availability (HA) features for the application. For example,
`tcps://salesserver1:1521/sales.us.example.com?connect_timeout=60&transport_connect_timeout=30&retry_count=3`
- It allows multiple hostnames and multiple port numbers in the URL, making it compatible with **Oracle RAC**³. E.g.,
`tcps://salesserver1:1521,salesserver2,salesserver3:1522/sales.us.example.com`

The Oracle Database Easy Connect Plus syntax is:

```
[[protocol:]]/[host1{,host12}[:port1]{,host2:port2}{,host3{,host32}[:port3]}[/[service_name][:server_type]/instance_name]][?parameter_name1=value1{&parameter_name2= value2}]
```

Older Clients (such as 19c and 21c) do not support address lists (separated by a semi-colon) with Easy Connect Plus. So, the Oracle Database Easy Connect Plus syntax with Oracle 19c and 21c clients is:

```
[[protocol:]]/[host1{,host2}[:port1]{,host2:port2}[/[service_name][:server_type]/instance_name]][?parameter_name1=value1{&parameter_name2= value2}]
```

Protocol

The protocol specifies the transport protocol to be used while connecting to the database host. From 19c, the supported values are *TCP* and *TCPS*. The default is *TCP*.

Hosts

The host list is a comma-separated list of hostnames or IP addresses used to connect to a database.

When multiple hosts are used, load balancing is turned on automatically.

The host list can also be separated by semi-colons, if there are multiple address lists in the Easy Connect string.

Ports

Port numbers indicate the ports that the database services are listening on. The default is 1521.

When using multiple hosts, if the same port is used on each host, then you can use:

```
host1,host2:port
```

Otherwise, you can indicate different ports that should be used:

```
host1:port1,host2:port2
```

³ Oracle Real Application Clusters

Service Name

The service name is the [service](#) the database is known by. It is a name comprised of the database name and domain name entered during installation or database creation.

Service names can be found by running "*lsnrctl services*" on the database host.

Note that database "System IDentifiers "(SIDs) cannot be used in Easy Connect or Easy Connect Plus syntax.

Server Type

The server type specifies what kind of server is used on the database host to handle the connection. It can be one of **DEDICATED**, **SHARED**, or **POOLED**. The default is to leave it unspecified and let the listener decide.

The common, traditional server type is [dedicated](#).

[Shared servers](#) ensure that the server processes are shared, but the session state is not shared. Session state is in the SGA.

A [pooled server](#) is used by [Database Resident Connection Pooling](#) (DRCP). The server process and session state are both shared. Session state is in the PGA.

Shared servers and DRCP are used when the database host does not have enough memory to handle a large number of dedicated connections.

Instance Name

If you need to connect to a specific instance that the service represents, you can use an instance name.

Parameters

Parameters are name-value pairs that control the behavior of connections. The syntax uses '?' to indicate the start of parameters and a '&' delimiter between each parameter. Leading and trailing white spaces are ignored within parameter values. If whitespace is required as part of the value, it should be placed within double quotes. Parameters are described in the next section.

PARAMETERS

Easy Connect Plus allows *Security*, *Proxy*, *Description* and *DRCP* parameters to be specified.

In all drivers except JDBC Thin, any parameters not noted here are passed through as *Description* parameters, allowing further (or future) options to be used when connecting. JDBC Thin has a [whitelist](#) of allowed *Description* parameters.

Security Parameters

Security parameters are used to configure a TLS connection to a database. When using these parameters, the protocol must be set to **TCPS**.

SSL SERVER CERT DN

Specifies the distinguished name (DN) of the database server.

When **SSL_SERVER_DN_MATCH** is **ON**, this DN is matched with the DN from the server certificate.

SSL_SERVER_DN_MATCH

Specifies if server certification validation through distinguished name (DN) matching should be enforced on the client side. The value can be **ON** or **OFF**. The default value is **ON** (from Oracle Client 19.2).

This parameter is used in conjunction with **SSL_SERVER_CERT_DN**.

If the **SSL_SERVER_CERT_DN** parameter is not specified, then the database server hostname (as specified by the **HOST** parameter in the original connect string) is checked against the Common Name (CN) and Server Alternate Name (SAN) fields from the server certificate by default. The connection will not succeed if the check fails.

WALLET_LOCATION

Sets the directory containing the Oracle wallets. Wallets are certificates, keys, and trustpoints⁴ processed by SSL.

Only filesystem wallets are supported with this method.

Proxy Parameters

Proxy parameters configure traffic to be routed through a proxy server.

Successful connection depends on specific proxy configurations. The performance of data transfers depends on proxy capacity. Oracle does not recommend using this feature in production environments where performance is critical.

HTTPS_PROXY

Sets an HTTP proxy hostname or IP address for tunneling TLS client connections. Non-TLS connections cannot be tunneled.

HTTPS_PROXY_PORT

Sets an HTTP proxy host port for tunneling TLS client connections.

Description Parameters

All other parameters will be description parameters. These include:

CONNECT_TIMEOUT

Sets the timeout duration in seconds for an application to establish an Oracle Net connection. There is no timeout by default.

⁴ For drivers using Oracle Client's 19.14 and 21.5 version libraries, built-in trustpoints support is available. With this feature, an SSO wallet is not required as long as the server certificate is signed by a Root CA which is present in the system trust store (the most commonly used Root CAs are present). This is also supported in the Managed and Core ODP.NET drivers (version 19.13 and 21.4 onwards). All versions of JDBC Thin driver use the default "truststore" mechanism from the JDK.

EXPIRE_TIME

Sets a time interval in minutes to send probes to verify that connections are active. Setting a value greater than 0 ensures that connections are not left open indefinitely due to an abnormal database server termination. If **EXPIRE_TIME** is used to prevent firewalls from terminating idle connections, the value should be less than half of the firewall timeout period.

FAILOVER

Enables or disables connect-time failover for multiple hosts. The value can be **ON** or **OFF**. The default is **ON**.

LOAD_BALANCE

Enables or disables Oracle client load balancing for multiple hosts. The value can be **ON** or **OFF**. The default is **ON**.

RECV_BUF_SIZE

Sets the TCP/TCPs socket receive buffer size in bytes. This parameter should be tuned to accommodate the Bandwidth-Delay Product (BDP) of the connection. The database server configuration file *sqlnet.ora* should also set this parameter.

BDP refers to the amount of data on the "wire" (data network) at any given point in time. Default Operating System buffers do not hold enough data to fill the wire. A starting recommendation is to set the receive buffer size to accommodate the BDP. For example, on a 40 Mbps network with 25 millisecond round-trip time between endpoints, the BDP is $(40 * 1000 / 8) \text{ KB/second} * (0.025) \text{ seconds}$, which is 125 KB. The **RECV_BUF_SIZE** parameter could be set to 125000.

Large buffer sizes help applications queue more data to the operating system and utilize the available bandwidth on the "wire" efficiently.

SEND_BUF_SIZE

Sets the TCP/TCPs socket send buffer size in bytes. This parameter should be tuned to accommodate the bandwidth-delay product (BDP) of the connection. The database server configuration file *sqlnet.ora* should also set this parameter.

A starting recommendation is to set the send buffer size to accommodate the BDP described by **RECV_BUF_SIZE** earlier. See **RECV_BUF_SIZE** for an example.

SDU

Sets the Oracle Net Session Data Unit (SDU) packet size in bytes. Larger **SDU** values give better network throughput due to fewer system calls and lower CPU usage at the expense of memory. The database server configuration should also set this parameter.

For bulk data transfers, set this to 64 K.

SOURCE_ROUTE

Enables network routing through multiple hosts. The value can be **ON** or **OFF**. The default is **OFF**.

RETRY_COUNT

Sets the number of times the list of hosts is traversed when attempting to connect to Oracle Database.

Connection is attempted in turn to each host in the host list, until a connection is successful or the whole host list has been traversed the specified number of times. If only one host is used, then the connection to this host is attempted the specified number of times.

RETRY_DELAY

Sets the delay in seconds between retries of host list traversal. The default is 0. This parameter works in conjunction with the **RETRY_COUNT** parameter.

TRANSPORT_CONNECT_TIMEOUT

Sets the transport connect timeout duration in seconds for a client to establish an Oracle Net connection to an Oracle database. This parameter is a subset of **CONNECT_TIMEOUT** and only applies to TCP and TCPS connection establishment.

Other parameters can also be used for all drivers except for the JDBC Thin driver. They are treated as parameters in the **DESCRIPTION** section of an equivalent *tnsnames.ora* connection descriptor entry. They are validated during connection.

DRCP Parameters

For applications running Oracle Client version 19c or later that connect to Oracle Database 21c, you can configure [Database Resident Connection Pooling](#) (DRCP) settings in the Easy Connect Plus Syntax in your applications through these parameters:

POOL_CONNECTION_CLASS

Explicitly names the connection class for a DRCP connection to an Oracle Database. A connection class is a logical name for the type of connection required by the application. The connection class setting lets different applications (which connect to the Oracle Database as the same database user) identify their DRCP pooled sessions using a logical name that corresponds to the application. Thus, sessions belonging to a particular connection class are not shared outside the connection class.

The maximum allowed length for the **POOL_CONNECTION_CLASS** value is 1024 characters (including the NULL terminating character), and it cannot include an asterisk (*).

POOL_PURITY

Specifies if an application needs a new session that is not filled with any earlier session state or must reuse a previous session from a DRCP connection to an Oracle Database. The values can be **NEW** or **SELF**. Using **POOL_PURITY=SELF** gives the best reuse and performance.

It is recommended to set the **POOL_CONNECTION_CLASS** in the Easy Connect string to partition the connections effectively and prevent session information leakage. You can set **POOL_PURITY=SELF** to get optimum performance and best resource (processes and sessions) utilization, especially when multiple instances of the application access Oracle Database through DRCP and can share session information.

If the application uses **POOL_PURITY=SELF** in the Easy Connect string, there is no guarantee that the application will honor drop session calls for the DRCP connection.

The DRCP parameters can be configured for both Container Databases (CDBs) and Pluggable Databases (PDBs) in Oracle Database. They can be used for all client drivers that use Oracle Client version 19c or later and Managed ODP.NET Thin driver. They are not supported by the JDBC Thin driver.

Implicit Connection Pooling Parameters

From Oracle Database 23ai onwards, Implicit Connection Pooling capability has been introduced with both Oracle Database's DRCP and Oracle Connection Manager - Traffic Director Mode (CMAN-TDM)'s Proxy Resident Connection Pooling (PRCP) features. To enable Implicit Connection Pooling, the **POOL_BOUNDARY** parameter needs to be set in the Connect String.

POOL_BOUNDARY

Enables [Implicit Connection Pooling](#) (an Oracle Database 23ai feature) with DRCP or PRCP. The values can be **STATEMENT** or **TRANSACTION**. This setting specifies the boundary condition to release an application session back to the DRCP or PRCP pool.

EXAMPLES

This section includes various Easy Connect Plus syntax examples. It depicts how each Easy Connect connection string can replace the connect descriptor that might be used in *tnsnames.ora* and other configuration files.

Example 1 - Protocol

`tcp://salesserver1:1521/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com)) )
```

Example 2 – Multiple hosts

`salesserver1,salesserver2:1782/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
  (LOAD_BALANCE=ON)
  (ADDRESS= (PROTOCOL=tcp) (HOST=salesserver1) (PORT=1782))
  (ADDRESS= (PROTOCOL=tcp) (HOST=salesserver2) (PORT=1782))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com)) )
```

Example 3 – Multiple hosts and ports

`salesserver1:1521,salesserver2,salesserver3:1522/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
```

```
(LOAD_BALANCE=ON)
(ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
(ADDRESS=(PROTOCOL=tcp) (HOST=salesserver2) (PORT=1522))
(ADDRESS=(PROTOCOL=tcp) (HOST=salesserver3) (PORT=1522))
(CONNECT_DATA=(SERVICE_NAME=sales.us.example.com))
```

Example 4 – Multiple address lists

`salesserver1:1521;salesserver2,salesserver3:1522/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS_LIST=
    (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521)))
  (ADDRESS_LIST=
    (LOAD_BALANCE=ON)
    (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver2) (PORT=1522))
    (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver3) (PORT=1522)))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com)))
```

This example will not work with older Oracle Clients (21c or earlier).

Example 5 – IPv6 address as the host

`[2606:b400:41c:8a19:f816:3eff:fe8f:e3ef]:1521/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp) (HOST=2606:b400:41c:8a19:f816:3eff:fe8f:e3ef)
    (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com)))
```

Note that a numeric IPv6 address has to be placed between enclosing square brackets.

Example 6 – TLS Connections

`tcps://salesserver1:1521/sales.us.example.com`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcps) (HOST=salesserver1) (PORT=1521))
  (SECURITY=(SSL_SERVER_DN_MATCH=ON))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com)))
```

Note that `SSL_SERVER_DN_MATCH=ON` gets automatically added to the full connect string and enforces server DN matching. The hostname used for connecting (*salesserver1* in this example) is matched against the CN⁵ and SAN⁶ fields in the server's SSL certificate. The client application gets connected to the database, only if the match succeeds.

⁵ CN - Common Name

⁶ SAN – Subject Alternative Name

Example 7 – TLS Connections (No Server DN matching)

`tcps://salesserver1:1521/sales.us.example.com?ssl_server_dn_match=off`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcps) (HOST=salesserver1) (PORT=1521))
  (SECURITY= (SSL_SERVER_DN_MATCH=OFF))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com)) )
```

The server DN matching is not enforced in this case.

Example 8 – Wallet location

`tcps://salesserver1:1521/sales.us.example.com?wallet_location="/tmp/oracle"`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcps) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com))
  (SECURITY= (WALLET_LOCATION=/tmp/oracle) (SSL_SERVER_DN_MATCH=ON)) )
```

Example 9 – Distinguished Name Matching

`tcps://salesserver1:1521/sales.us.example.com?ssl_server_cert_dn="cn=sales,cn=OracleContext,dc=us,dc=example,dc=com"`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcps) (HOST=salesserver1) (PORT=1521))
  (SECURITY= (SSL_SERVER_DN_MATCH=ON)
  (SSL_SERVER_CERT_DN=cn=sales,cn=OracleContext,dc=us,dc=example,dc=com))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com)) )
```

Example 10 – HTTPS Proxy

`tcps://salesserver1:1521/sales.us.example.com?https_proxy=www-proxy.mycompany.com&https_proxy_port=80`

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS= (PROTOCOL=tcps) (HOST=salesserver1) (PORT=1521)
  (https_proxy=www-proxy.mycompany.com) (https_proxy_port=80))
  (CONNECT_DATA= (SERVICE_NAME=sales.us.example.com)) )
```

Example 11 – Session Data Unit (SDU)

`salesserver1:1521/sales.us.example.com?sdu=16384`

replaces the following descriptor

```
(DESCRIPTION=
```

```
(SDU=16384)
(ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=sales.us.example.com))
```

Example 12 – Connect Timeouts

```
salesserver1:1521/sales.us.example.com?connect_timeout=60&
transport_connect_timeout=30&retry_count=3
```

replaces the following descriptor

```
(DESCRIPTION=
  (RETRY_COUNT=3) (CONNECT_TIMEOUT=60) (TRANSPORT_CONNECT_TIMEOUT=30)
  (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com))
```

Example 13 – DRCP Pool Connection Class

```
salesserver1:1521/sales.us.example.com:pooled?pool_connection_class=
value
```

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com)
  (SERVER=POOLED) (POOL_CONNECTION_CLASS=value))
```

Example 14 – DRCP Pool Purity

```
salesserver1:1521/sales.us.example.com:pooled?
pool_connection_class=value&pool_purity=new
```

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com) (SERVER=POOLED)
  (POOL_CONNECTION_CLASS=value) (POOL_PURITY=NEW))
```

Example 15 – Implicit Connection Pooling with DRCP (STATEMENT boundary)

```
salesserver1:1521/sales.us.example.com:pooled?
pool_boundary=statement
```

replaces the following descriptor

```
(DESCRIPTION=
  (ADDRESS=(PROTOCOL=tcp) (HOST=salesserver1) (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=sales.us.example.com) (SERVER=POOLED)
  (POOL_BOUNDARY=STATEMENT))
```

Example 16 – Easy Connect to an Oracle Cloud Autonomous Database (ADB) using mTLS

Applications can connect to Oracle Cloud ADBs through mutual TLS (mTLS) for secure communication. mTLS requires a database wallet on the client side, where the application is run. This wallet is a [.jso file](#) downloaded as part of a zip package from the ADB cloud console. A sample Easy Connect Plus syntax for connecting to an Oracle Cloud ADB via mTLS is as follows:

```
tcps://hostname.region.oraclecloud.com:1522/abc_dbsvcname_high.oraclecloud.com?wallet_location=/path/to/walletdir&retry_count=20&retry_delay=3
```

The wallet is stored in the `/path/to/walletdir` directory on the client side.

Using Easy Connect Plus syntax in the application means that you do not need the `sqlnet.ora` and `tnsnames.ora` files of the Oracle Cloud ADB. The equivalent settings in the `sqlnet.ora` and `tnsnames.ora` configuration files are shown below:

sqlnet.ora

```
WALLET_LOCATION = (SOURCE = (METHOD = file) (METHOD_DATA =  
(DIRECTORY="/path/to/walletdirectory")))  
SSL_SERVER_DN_MATCH=yes
```

tnsnames.ora

```
(description=  
(retry_count=20) (retry_delay=3) (address=(protocol=tcps) (port=1522) (host=hostname.region.oraclecloud.com)) (connect_data=(service_name=dbsvcname_high.adb.oraclecloud.com)) (security=(ssl_server_cert_dn="CN=common_name.oraclecloud.com, OU=Oracle BMCS US, O=Oracle Corporation, L=Redwood City, ST=California, C=US")))
```

Example 17 – Easy Connect to an Oracle Cloud Autonomous Database (ADB) using 1-way TLS

Applications can connect to Oracle Cloud ADBs through TLS for secure communication using 1-way TLS. Latest versions of database clients can use the built-in trust points (as mentioned in [Note 4 – Page 7](#)) and no wallet download is needed. A sample Easy Connect Plus syntax for connecting to an Oracle Cloud ADB via 1-way TLS is as follows:

```
tcps://hostname.region.oraclecloud.com:1521/abc_dbsvcname_high.oraclecloud.com?retry_count=20&retry_delay=3
```

Using Easy Connect Plus syntax in the application means that you do not need the `sqlnet.ora` and `tnsnames.ora` files of the Oracle Cloud ADB.

The equivalent settings in `sqlnet.ora` and `tnsnames.ora` configuration files are shown below:

sqlnet.ora

```
SSL_SERVER_DN_MATCH=yes
```

tnsnames.ora

```
(description=  
(retry_count=20) (retry_delay=3) (address=(protocol=tcps) (port=1521) (host=hostname.region.oraclecloud.com)) (connect_data=(service_name=dbsvcname_high.adb.oraclecloud.com))
```

Example 18 – Quoting Easy Connect Strings

Quoting of parameters may be required in some tools and environments.

The following example invokes SQL*Plus in a Linux shell. This illustrates the placement of single and double quotes to escape interpretation of special characters by the shell:

```
$ sqlplus
scott@'tcps://salesserver1:1521/sales.us.example.com?ssl_server_cert
_dn="cn=sales,cn=Oracle Context
Server,dc=us,dc=example,dc=com"&sdu=8128&connect_timeout=60'
```

CONCLUSION

Oracle Database's Easy Connect Plus syntax makes deploying highly available and secure applications easier without needing external configuration files for database connection management.

ORACLE CORPORATION

Worldwide Headquarters

500 Oracle Parkway, Redwood Shores, CA 94065 USA

Worldwide Inquiries

TELE + 1.650.506.7000 + 1.800.ORACLE1

FAX + 1.650.506.7200

oracle.com

CONNECT WITH US

Call +1.800.ORACLE1 or visit [oracle.com](https://www.oracle.com). Outside North America, find your local office at [oracle.com/contact](https://www.oracle.com/contact).



blogs.oracle.com/oracle



facebook.com/oracle



twitter.com/oracle

Integrated Cloud Applications & Platform Services

Copyright © 2024, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0824

Technical Brief **Oracle Database Easy Connect Plus** Configurable Database Connection Syntax Oracle Database Easy Connect Plus Oracle Database Easy Connect Plus
August 2024



Oracle is committed to developing practices and products that help protect the environment

ORACLE®