CONNX

Setting up Oracle Heterogeneous Services on a Linux machine to connect to a non-Oracle Database using CONNX

Overview

CONNX can be used within Oracle Heterogeneous Services to access a non-Oracle database. The following steps should enable you to use CONNX within Oracle Heterogeneous Services to be able access any of the databases that CONNX can connect to.

Prerequisites

1. The CONNX client must be installed on the Linux machine where you have Oracle installed
2. Create a CDD in CONNX to access the database you wish to connect to. Move that CDD to the Linux system.
3. Create a DSN in the Linux environment using the CONNX driver to connect to the CDD you created in step 2. Please refer to the Quick Ref guides for the UNIXODBC and Data Direct ODBC driver managers.
4. Test and verify you can connect to database using isql and the DSN you created in step 3.

Setting up Heterogeneous Services

1. Locate the $Oracle_home/hs/admin/initsodbc.ora file.
2. Save a copy of this file renaming it per the following template:
   init**DSNNAME**.ora (i.e. **initCONNXDSN.ora**)
   DSNNAME is the DSN name set up in step 3 of the prerequisites.
3. Open the initCONNXDSN.ora file.
   a. Locate the line "HS_FDS_TRACE_LEVEL = ". Change to OFF
   b. Locate the line "HS_FDS_CONNECT_INFO = ". Change to the data source name. (CONNXDSN)
   c. Add the line “HS_FDS_SHAREABLE_NAME = /usr/local/lib/libodbc.so”
      The path should be to the proper location of the libodbc.so file on your system.
   d. Add the line ”set ODBCINI=/usr/local/etc/odbc.ini”
      The path will be to the location of the odbc.ini file that your ODBC driver manager is using.
4. Configuring the Listener
   The Listener accessing the Oracle database must be configured to point to the Heterogeneous Services entry created in step 3. Create an entry in the SID _LIST similar to the one below. The bolded part is the actual entry. Note the following:

SID_NAME is the DSN name create in **prerequisite** step 3.
ORACLE_HOME is the actual Oracle home file path.
PROGRAM tells Oracle to use heterogeneous services. This entry may be either **"hsodbc" (for Oracle 11 and earlier)** or **"dg4odbc" (for Oracle 12 and later)** depending on the version of Oracle you are using. Please check with your Oracle administrator for the correct syntax.

Sample listener syntax:

```
SID_LIST_LISTENER2 =
        (SID_LIST =
                (SID_DESC =
                (SID_NAME=CONNXDSN) -- Enter the DSN on this line
                (ORACLE_HOME = /home/app/product/12.1.0/dbhome_1) -- Enter your Oracle
home on this line
                (PROGRAM = dg4odbc)
                )
        )

LISTENER2 =
(DESCRIPTION_LIST =
        (DESCRIPTION =
                 (ADDRESS = (PROTOCOL = TCP)(HOST = <Machine where Oracle
                resides>)(PORT = 1522))
  )
 )
```

5. Configuring Tnsnames.ora
   Oracle needs to know where to look for the remote database when it is called. This requires an entry in the Tnsnames.ora file; the following example can be followed:

CONNXDSN= -- This name can be customized
(DESCRIPTION=
(Address=(PROTOCOL=TCP) (HOST= -- Server where the Oracle re
sides (PORT=**1522**))) -- Enter the port for the second listener - from
step 4.
(CONNECT_DATA=(SID=**CONNXDSN**)) - Enter the DSN name
(**HS=OK**) -- Enter this value. It tells Oracle to use heterogeneous
services
)

Sample: CONNXDSN =
(DESCRIPTION=
(ADDRESS=(PROTOCOL=tcp)(HOST=localhost)(PORT=1522))
(CONNECT_DATA=(SID=CONNXDSN))
(HS=OK)

6. Reloading the listener

At this point the infrastructure is in place. You will need to reload the Listener.ora settings you defined in Step 4.
Execute one of the following from the command line:

lsnrctl reload listener2
lsnrctl start listener2

After completing this step, you need to check whether the listener and Tnsnames.ora file are configured correctly. Issue a Tnsping command from the command prompt. Make sure you get a successful response. The following exemplifies the command.

Tnsping CONNXDSN

7. Create a database link

The database link contains the Tnsnames reference (CONNXDSN) along with the remote data user id and password (CDD username and password). When the SQL statement containing the link is executed, Oracle passes the query through the link to the underlying database in the CDD.
There are a variety of database link options. This example is one that contains the remote database userid and password.

Note: it's important that the username and password must be in double quotes.

create database link **<link_name>** connect to **"<user_name>"** identified by **"<password>"** using **'<tnsname>';**

create database link **CONNXDSN** connect to **"connxuser"** identified by **"connxpassword"** using **'CONNXDSN'**;

You have now completed the configuration. You can test the configuration by executing a select statement against the non-Oracle database.

Select * from <table name defined in the CDD>@CONNXDSN;

If you need to have data outside the Oracle database, you will find this a great tool for making it transparent to the Oracle user.