

Oracle WebLogic Server 10.3 and Oracle Real Application Clusters (RAC)

An Oracle White Paper
January 2009

Oracle WebLogic Server 10.3 and Oracle Real Application Clusters (RAC)

1.0	Introduction.....	3
2.0	Oracle Real Application Clusters.....	3
2.1	New Support in Oracle RAC 11g.....	4
3.0	Oracle WebLogic Server and RAC	4
3.1	Oracle WebLogic Server Multi Data Sources.....	4
3.2	Oracle WebLogic Server Multi Data Sources and Oracle RAC.....	5
3.2.1	Planned outages.....	5
3.2.2	Unplanned outages	5
3.2.3	High Availability and Failover	6
3.2.4	Load Balancing	6
4.0	Oracle WebLogic Server and Oracle RAC Configurations	6
4.1	Configuring database listeners for Oracle RAC Instances.....	6
4.2	Configuring Multi data sources with or without global transactions	7
4.2.1	Multi data sources with global transactions.....	7
4.2.2	Multi data sources without global transactions.....	8
4.3	Configuring Connect-Time Failover	9
4.4	Configuring Fast Connection Failover (FCF)	10
4.4.1	Oracle RAC and Fast Connection Failover	10
4.4.2	Configuring Fast Connection Failover	11
5.0	Sample Configuration and Application	12
5.1	Configuring Oracle RAC and Defining a Database Service.....	12
5.2	Configuring Multi Data Source in Oracle WebLogic Server	13
5.3	Testing the Configurations	15
6.0	Summary	16
7.0	References.....	16

Oracle WebLogic Server 10.3 and Oracle Real Application Clusters (RAC)

1.0 INTRODUCTION

High availability and scalability has always been the focus in different systems enabling the customer scenarios with cost efficient solutions. There are various known issues in a heterogeneous complex environment involving JEE middle tier and the backend database. For example, the application requests may get blocked for a long period when a database node dies. There's no easy way to tell whether to obtain a fresh new connection after an application received the SQLException. The middle tier applications are unaware of new or restarted database nodes or it executes the work on a slow, hung or dead database node. Often it has to rely on waiting for TCP/IP time-outs.

Oracle WebLogic Server 10.3 provides strong support for the Real Application Clusters (RAC) features in Oracle Database 11g, minimizing database access time while allowing transparent access to rich pooling management functions that maximizes both connection performance and availability.

The combination of Oracle WebLogic Server JDBC multi data sources and Oracle RAC provides a high-end mission-critical environment offering high scalability and availability features. Load-balancing and failover can be handled by either Oracle WebLogic Server JDBC multi data source or Oracle RAC.

In this article, we start with a brief introduction to Oracle RAC and an overview of the Oracle RAC features supported in Oracle WebLogic Server 10.3. We then focus on details of Oracle RAC support configuration options in Oracle WebLogic Server JDBC multi data sources, with working samples. All the configuration steps and application samples are included in a companion "How-To" that is linked from this article.

2.0 ORACLE REAL APPLICATION CLUSTERS

Oracle RAC enables you to cluster Oracle databases. Single-instance Oracle databases have an one-to-one relationship between the Oracle database and the instance. Oracle RAC environments have a one-to-many relationship between the database and the instance.

Figure1 below shows how Oracle RAC is the Oracle Database option that provides a single system image for multiple servers to access one Oracle database. In Oracle RAC, each Oracle instance usually runs on a separate server.

Oracle WebLogic Server 10.3 provides strong support for Oracle Real Application Clusters (RAC) features in Oracle Database 11g, minimizing database access time while allowing transparent access to rich pooling management functions that maximizes both connection performance and availability.

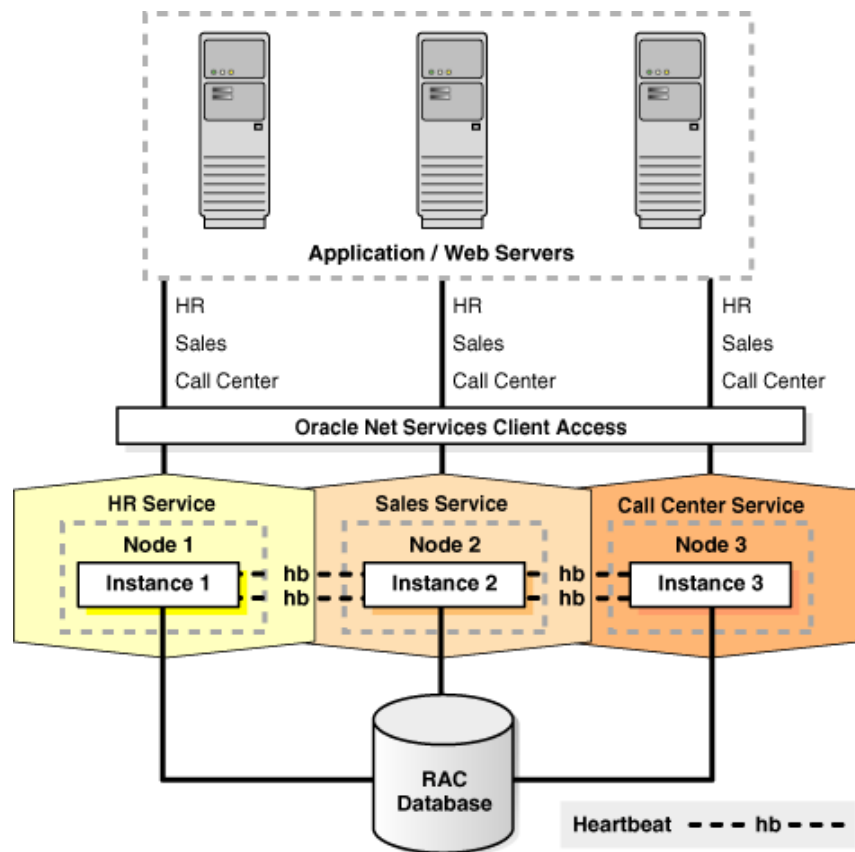


Figure 1

The new features in Oracle RAC 11g include enhanced Oracle RAC monitoring and diagnostics in Enterprise Manager, enhanced Oracle RAC Configuration Assistants, OCI runtime connection load balancing, support for parallel execution processes running on database instances and support for distributed transactions spanning Oracle RAC instances, etc.

2.1 New Support in Oracle RAC 11g

The new features in Oracle RAC 11g include enhanced Oracle RAC monitoring and diagnostics in Enterprise Manager, enhanced Oracle RAC Configuration Assistants, OCI runtime connection load balancing, support for parallel execution processes running on database instances and support for distributed transactions spanning Oracle RAC instances, etc.

In Oracle WebLogic Server 10.3, Oracle 11g RAC is fully certified to work together providing the high-availability solutions. The Oracle RAC services, such as failover, runtime connection load-balancing, etc, are available via Oracle WebLogic Server JDBC multi data source implementation.

3.0 ORACLE WEBLOGIC SERVER AND RAC

3.1 Oracle WebLogic Server Multi Data Sources

In JEE Application Servers, database interactions are typically handled by data source implementations. You configure and expose a connections to databases as

JDBC data sources. In Oracle WebLogic Server, you configure database connectivity by configuring JDBC data sources and multi data sources and then targeting or deploying the JDBC resources to servers or clusters in a WebLogic domain. Each data source that is configured contains a pool of database connections that are created when the data source instance is created, deployed or targeted, or at server startup.

A multi data source is an abstraction around a set of specified data sources that provides load balancing or failover processing across the set of data sources associated with the multi data source. Applications look up a multi data source through JNDI or in the local application context (`java:comp/env`) just like they do for a normal data source, and then request a database connection. The multi data source determines which data source to use to satisfy the request depending on the algorithm selected in the multi data source configuration: load balancing or failover.

- **Load Balancing:** Pools in WebLogic Server are accessed using a standard round-robin method. When moving to a new connection, WebLogic Server will select a connection based on the next connection pool in the specified order.
- **High Availability:** Connection pools are listed in the order that determines when connection pool switching occurs. WebLogic Server provisions database connections starting with the first connection pool in the list. If for some reason that connection pool fails, it uses the subsequent pool in the specified order.

3.2 Oracle WebLogic Server Multi Data Sources and Oracle RAC

Oracle WebLogic Server multi data sources together with Oracle RAC manage pooled connections for high availability and provide the below features.

3.2.1 Planned outages

Oracle WebLogic Server provides support for planned outages, where database maintenance or other activities are needed to be performed at a known point in time. This is available where an Oracle RAC service can be gracefully shutdown. In such scenarios, any borrowed or in-use connections are not interrupted and closed until work is completed and control of the connection is returned to the pool. This provides an extremely efficient way in large heterogeneous customer environments to manage planned outages.

3.2.2 Unplanned outages

Oracle WebLogic Server provides support for the area of unplanned outages. Depending upon your configuration, when a Oracle RAC node fails, in-flight transactions are redirected to another node in the cluster either by Oracle WebLogic Server or by the Oracle THIN JDBC driver.

3.2.3 High Availability and Failover

There are three options for handling connection failover with Oracle WebLogic Server and RAC.

The first option, which is also the Oracle recommended approach, is to let failover be handled by multi data sources. A multi data source “pins” a transaction to one and only one Oracle RAC instance. Failover is handled at the multi data source level when an Oracle RAC instance becomes unavailable. If there is a failure on a RAC instance before the PREPARE phase in a Two-Phase Commit scenario, the operation is retried until the retry duration has expired. If there is a failure after PREPARE phase, the transaction is failed over to another instance. This option applies to scenarios with or without global transactions.

The second option is applying when using a multi data source is not an option for your environment. This involves the use of Oracle database connect-time failover and load balancing, by configuring the Oracle JDBC driver's property `CONNECTION_PROPERTY_THIN_READ_TIMEOUT1`, which specifies the period to wait while reading from the socket.

The final option is using Oracle JDBC Implicit Connection Cache's feature of Fast Connection Failover to quickly be able to detect dead connections, clean the connection pool, and failover the connections.

3.2.4 Load Balancing

Load Balancing is supported through the use of JDBC multi data sources with Oracle RAC nodes. The set of data sources that form the multi data source are accessed using a round-robin scheme. When switching connections, Oracle WebLogic Server selects a connection from the next data source in the order listed.

When multi data sources are not an option, load balancing is not supported when using XA. In a configuration without a multi data source, Oracle WebLogic Server relies on the connect-time failover feature provided by the Oracle JDBC THIN driver to work with Oracle RAC.

4.0 ORACLE WEBLOGIC SERVER AND ORACLE RAC CONFIGURATIONS

4.1. Configuring database listeners for Oracle RAC Instances

First, for all the Oracle WebLogic Server RAC configuration options, a database listener must be configured for Oracle RAC instances appropriately.

A local database listener is required configuring on each Oracle RAC instance. Each database instance within the cluster should be configured to register with its local listener only. Oracle instances can be configured to register with the listener statically in the listener.ora file, or register dynamically using the instance initialization parameter `local_listener`, or both.

¹ This option applies to the Oracle THIN driver only.

4.2. Configuring Multi data sources with or without global transactions

There are three primary considerations to determine how Oracle WebLogic Server should be configured to make use of OracleRAC. The table below shows the key considerations.

Does Your Application Require				
Load Balancing?	Failover?	Global Transactions (XA)?	JDBC Store?	Recommendation
•	•	•		Use Multi Data sources with Global Transactions configure option
•	•		•	Use Multi Data Sources without Global Transactions configure option
	•		•	Use Connect-Time Failover without Global Transactions configure option

Table 1: Choosing Configurations to Use with Oracle RAC

4.2.1. Multi data sources with global transactions

If your applications require global transaction support when accessing Oracle RAC, then consider using multi data sources with distributed transaction support configuration option. The failover will be handled by multi data sources, instead of Oracle RAC.

If your applications require global transaction support when accessing Oracle RAC, then consider using multi data sources with distributed transaction support configuration option.

In a two-phase commit scenario, if there is a failure on a Oracle RAC instance before PREPARE, the operation is retried until the retry duration has expired. If there is a failure after PREPARE, the transaction is failed over to another instance. This option requires that all of the data sources defined for the multi data source use a XA-enabled driver, or none of them do. In addition, all the XA-related properties must be set to the same values for each data source.

There are several required attributes that need to be configure. These are needed for each data source used in the application:

First, the Oracle JDBC THIN driver is required for the connection, i.e.:

```
<url>jdbc:oracle:thin:@lcqsol24:1521:SNRAC1</url>
<driver-
name>oracle.jdbc.xa.client.OracleXADataSource</driver-name>
```

- Configure to set `KeepXAConnTillTxComplete="true"`, this will force the data sources to reserve a physical database connection and stay in the same transaction scope during entire life cycle for this particular application.
- Configure to set `XARetryDurationSeconds="300"`, the value is the time that Oracle WebLogic Server transaction manager will retry for XA recover, commit and rollback calls.
- Configure to set `TestConnectionsOnReserve="true"`, this could be used to test/verify the connections to the backend database. This could be accompanied with an attribute `TestTableName`.

Here is a partial configuration sample that includes the properties required to configure multi data source for use with Oracle RAC supporting global transactions:

```
<jdbc-driver-params>
  <url>jdbc:oracle:thin:@lcqsol24:1521:SNRAC1</url>
  <driver-
name>oracle.jdbc.xa.client.OracleXADataSource</driver-name>
  .....
</jdbc-driver-params>
<jdbc-connection-pool-params>
  <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
  <profile-type>0</profile-type>
</jdbc-connection-pool-params>
<jdbc-data-source-params>
  <jndi-name>oracleRACXAJndiName</jndi-name>
  <global-transactions-protocol>TwoPhaseCommit
    </global-transactions-protocol>
</jdbc-data-source-params>
<jdbc-xa-params>
  <keep-xa-conn-till-tx-complete>>true</keep-xa-conn-till-
tx-complete>
  <xa-end-only-once>>true</xa-end-only-once>
  <xa-set-transaction-timeout>>true</xa-set-transaction-
timeout>
  <xa-transaction-timeout>120</xa-transaction-timeout>
  <xa-retry-duration-seconds>300</xa-retry-duration-
seconds>
```

4.2.2 Multi data sources without global transactions

If your applications do not require global transaction support, then you can configure to use multi data sources for handling the failover and load balancing, but without specifying the XA configuration properties.

From the configuration perspective, the only difference from configuration shown in section 4.2.1 is that XA-related attributes are not required to be specified. Here is a sample configuration:


```

<jdbc-driver-params>
  <url>jdbc:oracle:thin:@lcqsol24:1521:snrac1</url>
  <driver-name>oracle.jdbc.OracleDriver</driver-name>
  <properties>
    <property>
      <name>user</name>
      <value>wlsqa</value>
    </property>
  </properties>
  <password-encrypted>{3DES}aP/xScCS8uI=</password-encrypted>
</jdbc-driver-params>
<jdbc-connection-pool-params>
  <test-connections-on-reserve>true</test-connections-on-reserve>
  <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
</jdbc-connection-pool-params>
<jdbc-data-source-params>
  <jndi-name>jdbcDataSource</jndi-name>
</jdbc-data-source-params>

```

4.3. Configuring Connect-Time Failover

Oracle RAC's connect-time failover can be used in your application for handling failover. Note that this option doesn't support load balancing and global transactions.

In this use case, each data source configured to point to multiple Oracle RAC nodes. In the failure scenarios, the failover time on the database connection is as long as the TCP timeout that is set at your operating system level.

The configuration to make use of connect time failover is set in the JDBC driver URL attribute. An additional property, **ConnectionReserveTimeoutSeconds**, also needs to be set to specify the time an application will wait for a connection to become available.

Here is an example:

```

<jdbc-driver-params>
  <url>jdbc:oracle:thin:@(DESCRIPTION=
    (ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)
      HOST=lcqsol24)(PORT=1521))(ADDRESS=(PROTOCOL=TCP)
      (HOST=lcqsol25)(PORT=152))(FAILOVER=on)
    (LOAD_BALANCE=off))(CONNECT_DATA=(SERVER=DEDICATED)
    (SERVICE_NAME=snrac))</url>
  <driver-name>oracle.jdbc.OracleDriver</driver-name>
  .....
</jdbc-driver-params>
<jdbc-connection-pool-params>
  <test-connections-on-reserve>true</test-connections-on-reserve>

```

```
<test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
<profile-type>4</profile-type>
</jdbc-connection-pool-params>
<jdbc-data-source-params>
  <jndi-name>oracleRACJndiName</jndi-name>
  <global-transactions-protocol>OnePhaseCommit
    </global-transactions-protocol>
</jdbc-data-source-params>
```

4.4. Configuring Fast Connection Failover (FCF)

Within Oracle JDBC driver Implicit Connection Cache (ICC) implementation, a feature commonly known as Fast Connection Failover (FCF) is available. This is an Oracle RAC/Fast Application Notification (FAN) client implementation. By using the Oracle JDBC drivers and ICC, Java clients can leverage RAC failover and load balancing features.

Oracle WebLogic Server has a built-in connection pooling infrastructure for connection management, load balancing and transaction management, independent of JDBC driver's implementations like ICC. As a result, Oracle WebLogic Server only works with FCF in ICC for limited use cases. Specifically, in the current release global transactions are not supported when Oracle WebLogic Server is configured to use FCF through ICC. For the majority of Oracle WebLogic Server and RAC integration use cases, the native WebLogic Server multi data source, not using FCF in ICC, is a strongly recommended best practice.

To use FCF in Oracle WebLogic Server, rather than using multi data sources, a single data source is used. In this mode, failover, load balancing and transaction management are deferred to the Oracle JDBC ICC implementation rather than Oracle WebLogic Server pooling infrastructure. The following section discusses the additional details and steps necessary to use this approach.

4.4.1 Oracle RAC and Fast Connection Failover

FCF manages pooled connections for high availability and provides the following features:

- FCF supports unplanned outages. Dead connections are rapidly detected and are aborted and removed from the pool. Connection removal relies on abort to rapidly sever socket connections in order to prevent hangs. Borrowed and in-use connections are interrupted only for unplanned outages.
- FCF supports planned outages. Borrowed or in-use connections are not interrupted or closed until work is completed and control of the connection is returned to the pool.
- FCF encapsulates fatal connection errors and exceptions into the `isvalid` API for robust and efficient retries.

FCF manages pooled connections for high availability, supports unplanned, planned outages.

- FCF recognizes new nodes that join an Oracle RAC cluster and associates new connections with that node as appropriate in order to maximize quality of service. This facilitates middle-tier integration of Oracle RAC node joins and work-request routing from the application tier.
- FCF distributes runtime work requests to all active Oracle RAC instances.

4.4.2 Configuring Fast Connection Failover

Regardless of Oracle WebLogic Server versions, in order to use FCF with Oracle WebLogic Server data sources, Oracle Notification Service (ONS) and RAC must be setup appropriately.

Regardless of Oracle WebLogic Server versions, in order to use FCF with Oracle WebLogic Server data sources, Oracle Notification Service (ONS) and RAC must be setup appropriately. This is probably the most critical part of a correct FCF configuration. The detection and clean up of JDBC connections by the middle tier depends completely on the correct propagation of database events from the RAC nodes to the middle tiers. There are different ONS versions bundled with different releases. Oracle Database version 10g (10.1.0.3) RAC or better is required along with the necessary patches.

Oracle WebLogic Server needs to be started with the Oracle ONS libraries. You will need the ons.jar file and it needs to be in the CLASSPATH of the managed server. One way is that in the WLS Admin Console under Servers, your managed server, in the "Server Start" tab, specify \$CLASSPATH:<full path to ons.jar>, in the "Class Path" field. For example: \$CLASSPATH:/home/oracle/ons.jar. This ensures that the Node manager starts the managed server with the correct CLASSPATH.

To enable Fast Connection Failover on a data source, the following connection pool properties must be considered:

Modify the URL parameter, so that it uses a service name instead of SID. For example,

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host1)
(PORT=1522))(ADDRESS=(PROTOCOL=TCP)(HOST=host2)(PORT=1522))
(LOAD_BALANCE=yes)(CONNECT_DATA=(SERVICE_NAME=RCLB)))
```

- In Driver Class Name — set the class name to oracle.jdbc.pool.OracleDataSource.
- The Properties need to be set to the following values:
 - connectionCachingEnabled : true (to enable connection caching in the oracle JDBC driver)
 - fastConnectionFailoverEnabled : true (to enable FAN events)
- Set the ONS configuration string to remotely subscribe the Oracle RAC nodes to Oracle FAN/ONS events. For example:

```
"nodes=hostname1:port1,hostname2:port2"
```

Here is an example:

```
user=scott
fastConnectionFailoverEnabled=true
connectionCachingEnabled=true
ONSConfiguration=nodes=host1:6251,host2:6251
```

Oracle WebLogic Server will need to be restarted after the configuration change. You could do this using the Admin Console or boot WLS directly from the scripts. You could get the details from Oracle WebLogic Server documentation.

5.0 SAMPLE CONFIGURATION AND APPLICATION

Now, let's take a look at a working example of a configuration that uses Oracle RAC in conjunction with Oracle WebLogic Server JDBC multi data source. The application and all the configuration steps are available as a "How-To" at the link listed in the References section.

5.1. Configuring Oracle RAC and Defining a Database Service

First, let's see how to configure Oracle RAC appropriately so it can be used by your middle-tier applications. Here are the steps:

- Log onto Oracle Enterprise Manager Database Control as SYS.
- Click on the link "Availability" at the top of the page.
- Click on the link "Cluster Managed Database Services".
- Supply the cluster credentials as well as the database credentials.
- Create the services.
- Define a services as defined in the below table:

Service Name	MYHASERVICE
Enable Load Balancing Advisory	SELECTED
Throughput Radio Option	SELECTED
Connection Load Balancing Goal	LONG

After you create the service, "Cluster manage Database Services" page can be used to verify/check the service status as shown below. Of course you also could use the SQL Plus to test the service.

The multi data source is a composite of individual data sources, each of which is configured to connect to one instance of the Oracle RAC backend. Deployed applications program obtain connections to the Oracle RAC backend using the specified JNDI name of the multi data source.

ORACLE Enterprise Manager 11g
Database Control

Cluster Database: orcl.apemrac.au.oracle.com >

Cluster Managed Database Services

The following shows the status of all cluster managed services defined for the current database. Select a service to manage the states of its instances. Page Refreshed 11/18/08 2:43 PM Refresh

Create Service

Select	Service Name	Status	Running Instances	Response Time (milliseconds)(Last 5 Minutes)	% CPU Load (Last 5 Minutes)	Service related alerts among all Instances	Status Details
<input checked="" type="radio"/>	MYHASERVICE	Running	orcl1, orcl2	0 0 0 0			Service is running on all preferred instances.

5.2. Configuring Multi Data Source in Oracle WebLogic Server

The following configuration example uses Oracle WebLogic Server multi data sources for Oracle RAC. The multi data source is a composite of individual data sources, each of which is configured to connect to one instance of the Oracle RAC backend. Deployed applications program obtain connections to the Oracle RAC backend using the specified JNDI name of the multi data source.

First, a data source needs to be created for each Oracle RAC node that is to be used. Here is how this can be accomplished using the Oracle WebLogic Admin Console.

Create a New JDBC Data Source

Back Next Finish Cancel

JDBC Data Source Properties

The following properties will be used to identify your new JDBC data source.

* Indicates required fields:

What would you like to name your new JDBC data source?

Name:

What JNDI name would you like to assign to your new JDBC Data Source?

JNDI Name:

What database type would you like to select?

Database Type:

What database driver would you like to use to create database connections?

Database Driver:

Back Next Finish Cancel

You can test the connection as well as target the data source to your particular managed server.

Then a multi data source needs to be created as shown below.

Create a New JDBC Multi Data Source

Back Next Finish Cancel

Add Data Sources

What JDBC Data Sources would you like to add to your new JDBC Multi Data Source?

Data Sources:

Available		Chosen
<input type="text" value="jdbc/scottDS"/>	<input type="button" value="➔"/> <input type="button" value="➜"/>	<input type="text" value="Node1DS"/> <input type="text" value="Node2DS"/>

Back Next Finish Cancel

The domain's config/jdbc directory will look as following once this is complete, with our newly created data sources and multi data source:

```
[.../user_projects/domains/hadomain/config/jdbc]$ d
total 56
-rw-r--r--  1 oracle oinstall 135 Nov 26 09:40 readme.txt
drwxr-xr-x 11 oracle oinstall 4096 Nov 27 12:31 ../
-rw-r--r--  1 oracle oinstall 2259 Dec  3 07:48 jdbc2fscottDS-8614-jdbc.xml
drwxr-xr-x  2 oracle oinstall 4096 Dec  4 11:14 ./
-rw-r--r--  1 oracle oinstall  857 Dec  4 11:15 jdbc2fracDS-4177-jdbc.xml
-rw-r--r--  1 oracle oinstall 2508 Dec  4 12:23 Node1DS-0882-jdbc.xml
-rw-r--r--  1 oracle oinstall 2508 Dec  4 12:27 Node2DS-9921-jdbc.xml
```

Here is the multi data source that was created:

```
<?xml version='1.0' encoding='UTF-8'?>
  <jdbc-data-source
xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-
source
  http://www.bea.com/ns/weblogic/jdbc-data-
source/1.0/jdbc-data-source.xsd">
  <name>jdbc/racDS</name>
  <jdbc-connection-pool-params>
    <test-frequency-seconds>5</test-frequency-
seconds>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/racDS</jndi-name>
    <algorithm-type>Load-Balancing</algorithm-type>
    <data-source-list>Node1DS,Node2DS</data-source-
list>
    <failover-request-if-busy>>false</failover-
request-if-busy>
  </jdbc-data-source-params>
  </jdbc-data-source>
```

It is possible to test the configuration by using the simple web application that is provided by the How-To linked from this paper.

5.3. Testing the Configurations

To test the application, the sample web application needs to be deployed. Once deployed, the sample web application will look like the below.

This is the main web page that allows you to specify the data source location, how many connections to retrieve and the sleep time between connection requests. The test could be made to ensure that the load balancing between the Oracle RAC nodes works properly. And the failover happens when one of the Oracle RAC node has been shutdown. The detailed testing steps are illustrated in the How-To.



RAC Integration with Oracle WebLogic Server 10.3

[Fill in the details below to test your newly created RAC enabled data source](#)

Data Source Location:

Connection to Retrieve:

Sleep Time b/w connection gets:

6.0 SUMMARY

Oracle WebLogic Server and Oracle RAC are designed to work together to provide an environment for highly available and scalable applications. There are different configuration options available to support enabling different use case scenarios. With Oracle WebLogic Server 10.3, Oracle recommends the use of JDBC multi data sources for handling database connection failover and load balancing.

7.0 REFERENCES

How to Configure Oracle WebLogic Server 10.3 with Oracle Real Application Clusters (RAC):

<http://www.oracle.com/technology/products/weblogic/howto/rac/index.html>

Using WebLogic Server with RAC:

http://e-docs.bea.com/wls/docs103/jdbc_admin/oracle_rac.html#wp1080135

OTN Article on Implicit Connection Caching Support in OC4J 10g (10.1.3) Data Sources:

http://www.oracle.com/technology/pub/notes/technote_ds_caching.html

Oracle Database 11g documentation:

<http://www.oracle.com/technology/documentation/database11gR1.html>

Oracle WebLogic Server 10.3 documentation:

<http://edocs.bea.com/wls/docs103/index.html>

Oracle Universal Connection Pool for JDBC Developer's, using RAC features:

http://download.oracle.com/docs/cd/B28359_01/java.111/e10788/rac.htm

Oracle Database 10g JDBC Documentation on Implicit Connection Caching:

http://download.oracle.com/docs/cd/B19306_01/java.102/b14355/concachel.htm#CDEGABII

Oracle Database 10g JDBC Documentation on Fast Connection Failover:

http://download.oracle.com/docs/cd/B19306_01/java.102/b14355/fstconfo.htm#CIHJBFFC

Oracle WebLogic Server 10.3 and Oracle Real Application clusters (RAC)

January 2009

Author: Frances Zhao

Contributing Authors:

Pas Apecilla

Erik Bergenholtz

Steve Button

Mike Lehmann

Oracle Corporation

World Headquarters

500 Oracle Parkway

Redwood Shores, CA 94065

U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

oracle.com

Copyright © 2005, Oracle. 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

ORACLE FUSION MIDDLEWARE

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, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.