Using Oracle Rdb with Oracle 10G Developer Suite 9.0.4 and Oracle 10G Application Server 9.0.4
(July 1 2004)

Introduction

This document addresses the use of testing selected Oracle 10G Developer Suite and 10G Application server products with Oracle Rdb databases. Unless otherwise stated it also applies to Oracle 9i release 2 versions (9.0.2) of Developer Suite and Application Server.

Oracle Developer Suite used for testing was installed on Windows 2000 and Windows XP platforms and the 10G Application Server, infrastructure and midtier, was installed on a Solaris 2.7 platform.

Developer Suite components tested

JDeveloper
Designer
Forms Developer
Reports Developer
Discoverer Administrator
Discoverer Desktop

Application Server components tested

Portal
Forms Services
Reports Services
Discoverer Plus
Discoverer Viewer
Discoverer Java command-line EUL tool

For testing purposes an Oracle Rdb 7.1 database on an OpenVMS system was created and prepared it for use with SQL*net for Rdb, so both the Oracle thin JDBC and the `Rdb Native thin JDBC Driver` could be used interchangeably where appropiate, or the database could be accessed by use of a SQL*Net connection (version 7.1-6) through local tnsnames.ora. Different connection methods are used by the various Oracle Application Server components as shown in the below matrix. In some cases an Rdb database can be accessed through a JDBC driver connection, whether the Oracle Rdb native JDBC driver, or the Oracle thin (or OCI) JDBC driver in conjunction with SQL*Net for Rdb. In other cases, such as with Forms or Discoverer, connection through a local SQL*Net client installation is required. Oracle Application Server consists of an Infrastructure layer, with an Oracle 9i/10G database as a repository and a Mid Tier layer, which contains a local SQL*Net connection to the Infrastructure.

Below you will find brief descriptions of each of the major Oracle Application Server and Oracle Developer Suite component products with any special configuration instructions for Oracle Rdb where appropiate. In all cases where a SQL*Net client or Oracle thin JDBC driver connection to an Rdb database is used, configuration proceeds according to the product-specific configuration documentation as described for an Oracle 8i, 9i or 10G database although there can be differences in the functionality of the product due to the differences in the underlying database. In other cases where the `Rdb Native thin JDBC Driver` is used, there may be special instructions in the Oracle documentation that refer to use of third party JDBC drivers that can be applied. In these cases we described typical configuration steps below.

Test Matrix

	Native Rdb JDBC driver support	Oracle thin JDBC with SQL*Net for Rdb	SQL*Net client with SQL*Net for Rdb
JDeveloper	Yes	Yes	No
Designer	No	No	No
Forms Developer	No	No	Yes
Reports Developer	Yes	Yes	Yes
Discoverer Administrator	No	No	Yes
Discoverer Desktop	No	No	Yes
Portal	Yes	Yes	Yes
Forms Services	No	No	Yes
Reports Services	Yes	Yes	Yes
Discoverer Plus	No	No	Yes
Discoverer Viewer	No	No	Yes
Discoverer command-line tool	No	No	Yes

Jdeveloper

Jdeveloper can use Oracle thin driver with a SQL*Net for Rdb prepared database or the `Rdb Native thin JDBC Driver` after registering it, within the Jdeveloper GUI, as a known third party JDBC driver and adding a reference to the rdbthin.jar jdbc driver file in the Jdeveloper configuration.

Jdeveloper and BC4J classes can be used with Rdb either with Oracle thin driver/SQL*Net for Rdb or with Rdb native thin JDBCdriver. Testing can be done within the Jdeveloper embedded OC4J server, and completed applications can be deployed to Application Server mid-tier OC4J servers.

There is a file called `NATIVEDRIVERSANDJDEV.TXT` distributed with the Rdb JDBC kit that describes in detail how to configure Jdeveloper, using the built-in third party JDBC driver functionality, for use with the Rdb thin driver.

Designer

Designer uses an Oracle database as a repository. Oracle Rdb cannot be used for this purpose although Designer can be used with Oracle Rdb databases, once a repository has been established.

Currently the only supported method for capturing the design of an Rdb database ("Capture server model from database") is using metadata extracted from an Rdb database using RMU/EXTRACT.

Forms Developer and Forms Services

Forms Developer, the tool used to create and edit Oracle forms is a SQL*Net based client product that requires a SQL*Net for Rdb prepared database and a local SQL*Net client installation and tnsnames.ora entry to connect to an Rdb database. Forms are displayed within the web browser (or Jinitiator) as a Java applet, although connection is still via SQL*Net rather than JDBC. There is a local OC4J server supplied with Developer Suite that allows local testing of the created form without deployment to an Application Server instance. Forms Services, the Application Server component that executes the form built through use of Developer Suite Forms builder can be executed under the local OC4J instance for testing prior to deployment.

Using Forms Developer and Forms Services, queries and DML operations such as insert, update and delete can be executed against an Rdb database.

Once built and tested within Forms Developer the form source file (.fmb extension) is copied to the Application Server <Appserv_home>\Forms90 directory (Midtier instance). Then f90genm is used to compile the forms source into the .fmx executable. This step is necessary if heterogenous operating systems are in use for Application Server and Developer Suite platforms. For example to compile a forms source file originally built on Windows 2000 platform to a Unix Application Server platform :-

(make sure DISPLAY variable set)
f90genm.sh Module=/private1/oracle/rdbtest.fmb userid=cts1/cts1@sid

..which builds an executable called rdbtest.fmx that can then be referenced in formsweb,cfg by adding a new entry.

[rdbtest]
form=rdbtest.fmx

The form can then be executed using a similar http request to the example below

http://applicationserver.domain.com:7779/forms90/f90servlet/?config=rdbtest

The Forms Runtime Server uses only a SQL*Net tnsnames.ora connection. There is no JDBC access for Forms at build or runtime so the connection has to use a SQL*Net-for-rdb pepared Rdb database and a tnsnames connection, rather than use the `Rdb Native thin JDBC Driver` or the Oracle thin JDBC driver. However an Rdb database can be accessed via JDBC drivers (either Oracle or Native Rdb) from within a java bean called from within a Forms application.

Portal

Portal 10G brings a new ready-made portlet type, 'Omniportlet', that allows easy creation of a portlet that accesses an Oracle Rdb database, either using Oracle thin JDBC driver and SQL*Net for Rdb, or the `Rdb Native thin JDBC Driver`. These drivers can also be used with early versions of Oracle Portal, including Portal 9i release 1 and 2, but have to be built like any other portal object with less automation.

(Note, the `Rdb Native thin JDBC Driver` is not listed as a supported JDBC driver for Oracle Portal in the product documentation - Only Oracle OCI8/thin and Datadirect drivers are listed as supported JDBC drivers).

Following are the configuration steps required to use the `Rdb Native thin JDBC Driver` with 'Omniportlet'.

First, add the `Rdb Native thin JDBC Driver`to the selection choice for Portal JDBC driver and add reference to the thin driver jar file. Edit the provider.xml file in the (midtier) <appserver_home>/j2ee/portal/applications/portalTools/omniPortlet/WEB-INF/providers/omniPortlet
directory.

Add these lines after a preceding <driverInfo></driverInfo> XML tag pair:-

```
<!-- registration of Connect for Native Rdb JDBC driver -->
<driverInfo class="oracle.webdb.reformlet.data.jdbc.JDBCDriverInfo">
  <name>Oracle Rdb</name>
  <sourceDataBase>other</sourceDataBase>
  <subProtocol>rdbThin</subProtocol>
  <connectString>mainProtocol:subProtocol://databaseName</connectString>
  <driverClassName>oracle.rdb.jdbc.rdbThin.Driver</driverClassName>

<connHandlerClass>oracle.webdb.reformlet.data.jdbc.JDBCODBCConnectionHandler</connHandlerClass>
  <connPoolSize>5</connPoolSize>
  <loginTimeOut>30</loginTimeOut>
</driverInfo>
```

Edit the application.xml file in the (midtier) <appserver_home>/j2ee/OC4J_Portal/config directory. Add this line (suitably modified to point to path of rdbthin.jar JDBC driver file on the target system) in a suitable place in the lines of library paths (e.g after <library path="../../../jlib/share.jar"/>

```
<library path="/private1/oracle/jdbc/rdbthin.jar"/>
```

Using Enterprise manager, restart OC4J_Portal.

Now build an Omniportlet instance that will connect to an Rdb database using the `Rdb Native thin JDBC Driver` ...

(a) On an existing Portal page, create a new Omniportlet in a selected region using provided documentation (select "Portlet Builders" from the Portlet Repository then select

"Omniportlet", then click on "Define your Omniportlet" and select SQL data view)

(b) Click on "Edit connection" . Enter userid, password for the database connection to the Rdb database, then enter connection string in rdb thin driver format:- e.g.

testserver.domain.com:1701/user_disk:[testbui3]mf_personnel

(This assumes a thin server listening on port 1701 on an Openvms system containing the database MF_PERSONNEL in directory 'user_disk:[testbui3]'.

Select "Oracle Rdb" from driver name drop down list box. Click the test button and click OK

(c) Modify the default SQL statement (select * from emp) to the desired SQL statement.

(d) Save omniportlet and use according to the documentation.

Reports Developer and Reports Services

Reports Builder tool allows a reports data source to be built from a query of the following types - Oracle Express Server, SQL, JDBC, Text or XML. Application Server 9i Release 2 and 10G also support use of the `Rdb Native thin JDBC Driver` for development instead of (or in addition to) the Oracle JDBC drivers with SQL*Net for Rdb , as was required for reports development in earlier releases. Oracle Rdb can be accessed either with an SQL query (using local tnsnames.ora and a local client SQL*Net installation) or with a JDBC query. The JDBC query can use the pre-installed Oracle thin JDBC driver with an Rdb database prepared for use with SQL*Net for Rdb (select "Oracle thin" in the standard listbox selection in the connection GUI) or it can use the `Rdb Native thin JDBC Driver`, after suitable configuration changes. See the whitepaper at:-

http://otn.oracle.com/products/reports/htdocs/getstart/whitepapers/pbr_jdbcpds.pdf

Using SQL data source type:-

Specify database via local tnsnames.ora entry which references a SQL Services service referencing an Rdb database prepared for use with SQL*Net for Rdb

Using JDBC data source type:-

Add an entry to the jdbcpds.conf file referred to in the above whitepaper, to allow the display and selection of the Oracle Rdb thin driver in the connection GUI mentioned above (file is in <appserver_home>/reports/conf/jdbcpds.conf).

```
<driver
name = "Oracle Rdb thin"
subProtocol = "rdbThin"
connectString = "mainProtocol:subProtocol://databaseName"
class = "oracle.rdb.jdbc.rdbThin.Driver"
connection = "oracle.reports.plugin.datasource.jdbcpds.JDBCConnectionHandling"
loginTimeout = "0">
</driver>
```

Now the `Rdb Native thin JDBC Driver` can be selected. When prompted for 'Database' not the entire Rdb database path specification has to be provided. For example, specify the string as:-

host.mydomain.com:1701/user_disk:[testbui3]mf_personnel

Note this string is specified without the leading '//' as this part of the string has already been specified in the jdbcpds.conf file.

Or, use the Oracle thin JDBC driver with a database exposed via SQL Services and prepared for use with SQL*Net for Rdb as used above with the SQL data source connection type. The database specification to be used will be, for example,

nodename:1527:cts

.. where 'nodename' is the host name, 1527 the port and cts the service name.

To deply these reports to the application server, one method is to simply move the .jsp to <appserver_home>/j2ee/OC4J_BI_Forms/applications/reports/web and deploy with a similar url to:-
http://applicationserver.domain.com:7779/reports/MODULE1.jsp?userid=cts1/cts1@cts

Discoverer

Oracle Discoverer consists of 4 products. Two of them, Discoverer Administrator and Discoverer Desktop are part of the Developer Suite and the other two, Discoverer Viewer and Discoverer Plus are part of the Application server. Before any database can be used with Discoverer Desktop, Discoverer Plus or Viewer, tables and views that constitute an End User Layer (EUL) have to be created within it, and one or more Business Areas created, using Discoverer Administrator. Once this has been done the Discoverer developer can use either Discoverer Desktop or Discoverer Plus to create workbooks that constitute the business intelligence reports that are viewed by the end user, using Discoverer Viewer within the Application Server.

Application Server 10G introduces a java command line tool for Unix, the *Java Command-Line Tool for EUL Maintenance,* that alows EULs (but not Business Areas) to be created and deleted, and allows various synchronization tasks to be completed without use of Discoverer Administrator.

Discoverer Administrator and the Java Command-Line tool rely on a local SQL*Net installation and tnsnames.ora connection to an Rdb database prepared for use with SQL*Net for Rdb. Only an Rdb database prepared for such use, into which an EUL and at least one Business Area has been created, can be used in the other Discoverer products.

Connection string used:- cts1/cts1@sid where sid is a local tnsnames.ora entry referring to the Rdb database.

Discoverer Plus and Discoverer Viewer, both Web based tools that access the Application Server Midtier, can be used to access the Business Areas created using Developer Suite Discoverer Administration Edition. Discoverer Plus has the capability of creating and modifying workbooks and also viewing them. Discoverer Viewer is a viewer for existing workbooks only.

There is a known problem with attempting to save a Workbook to an Oracle Rdb database using Discoverer Plus or Desktop. An ORA-12151 error is returned and is due to missing SQL*Net for Rdb support for certain types of Blob handling. The workaround is to save Workbooks to a local or remote operating system file system. These Workbooks can be restored from the file system to be used in a later Discoverer Desktop session on the same platform. A future release of SQL*Net for Rdb will provide additional BLOB support. Meanwhile however, use of Discoverer is limited by this issue.