Configuring Oracle TopLink Applications with Oracle Active Data Guard

Oracle Maximum Availability Architecture White Paper
September 2009

Maximum Availability Architecture

Oracle Best Practices For High Availability

ORACLE

Executive Overview

This paper provides the Oracle Maximum Availability Architecture (MAA) best practices describing how new or existing applications built using Oracle TopLink can be configured with Oracle Active Data Guard...

The combination of Oracle Active Data Guard and the Oracle TopLink can be used to:

- Increase database capacity and reduce the load on the primary database server
- Improve performance for all transactions, both read/write and read-only workloads
- Increase the utilization of existing standby databases

MAA is the Oracle best practices blueprint for implementing Oracle high-availability technologies. For more information about MAA, see the Web site at http://www.otn.oracle.com/goto/maa.

How It Works

This document guides you through the configuration of Oracle Active Data Guard and the Oracle TopLink Read Connection Pool. The following sections provide an introduction to these technologies.

Oracle Active Data Guard

Oracle Data Guard is an included feature of Oracle Database Enterprise Edition that provides a comprehensive set of services that create, maintain, manage, and monitor one or more standby databases to enable production Oracle databases to survive failures, disasters, user errors, and data corruption. Data Guard maintains standby databases as transactionally consistent copies of the production database. If the production database becomes unavailable due to a planned or an unplanned outage, Data Guard can switch any standby database to the production role, thus greatly reducing downtime.

A physical standby database provides a physically identical copy of the primary database, with on-disk database structures that are identical to the primary database on a block-for-block basis. A physical standby database is kept synchronized with the primary database through Redo

Transport Services, which transport the redo data from the primary database to the standby; and Redo Apply, which applies the redo data to the physical standby database. As of Oracle Database 11g Release 1 (11.1), a physical standby database can receive and apply redo data while it is open for read-only access, allowing the standby database to be used for other purposes in addition to disaster recovery (see Figure 1). This configuration is known as Oracle Active Data Guard (and sometimes referred to as "real-time query") and requires a license for the Active Data Guard Option.

Oracle Active Data Guard is a database option for Oracle Database 11*g* that extends Data Guard capabilities by enabling a physical standby database to be open for read-only access while it is simultaneously kept up-to-date with the primary database. Active Data Guard is the simplest, most transparent, and highest performing method of maintaining an active, synchronized replica of your primary database that is opened for read-only access. Active Data Guard is also the fastest way to make existing disaster recovery sites active, without the complexity of traditional replication solutions or their byproducts of increased management complexity, greater chance of human error, and more potential for data loss and downtime.

For more information about Oracle Active Data Guard, see http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardOverview.html

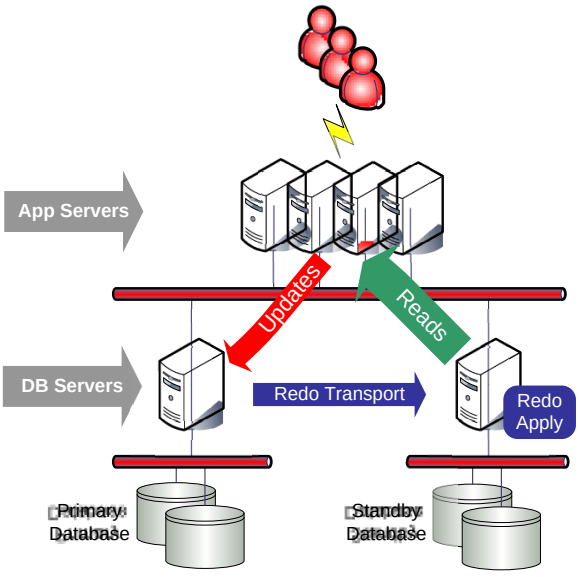


Figure 1: An Active Data Guard Configuration

Oracle TopLink and the TopLink Read Connection Pool

Oracle TopLink is a member of the Oracle Fusion Middleware family of products. It delivers a proven enterprise Java solution for all of your persistence needs based on high performance and

scalability, developer productivity, and flexibility in architecture and design. TopLink is an advanced, object-to-relational persistence framework for storing Java objects and Enterprise Java Beans (EJBs) in relational database tables. It offers developers tools and run-time capabilities that reduce development and maintenance efforts when working with any database, any application server, any development toolset and process, and any J2EE architecture,

Oracle TopLink offers the ability to configure a Read Connection Pool, which is a separate read-only database connection pool that allows read-only database traffic to be redirected to a separate, independently configured set of database connections. By default, non-transactional read I/Os use the read pool and all transactional reads and writes use the write pool. In general, read I/Os in TopLink are non-transactional. Transactional read I/Os typically occur only after a pessimistic lock (`SELECT ... FOR UPDATE`) request or when a user forces the transaction to begin.

To illustrate these concepts, Figure 2 walks you through the life cycle of an object being processed in a TopLink and Active Data Guard configuration. In Figure 2, the object is being read through the read-only pool that is connecting to the standby database and is updated through the read/write pool that is connecting to the primary database

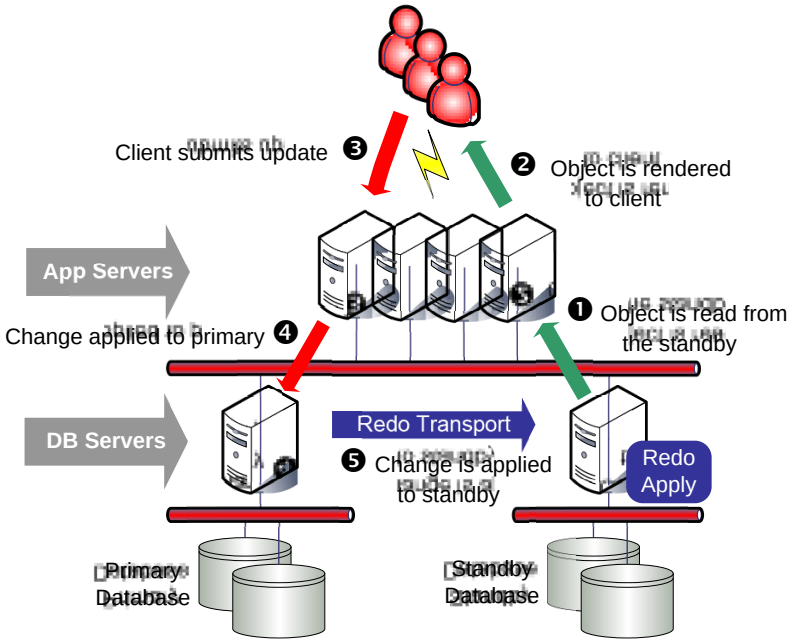


Figure 2 TopLink Object Lifecycle

The following list describes the notations shown in Figure 2:

❶ TopLink reads objects using the read-only database connection.

For example:

```
SELECT t0.EMP_ID, t0.GENDER, t0.F_NAME, t0.L_NAME, t0.VERSION,
t1.SALARY FROM EMPLOYEE t0, SALARY t1 WHERE (((t0.F_NAME LIKE
%Richard%') OR (t0.L_NAME LIKE '%Exley%')) AND (t1.EMP_ID =
t0.EMP_ID))
```

TopLink uses an optimistic locking mechanism so that, if the user chooses to update the object later, TopLink can detect a conflict situation.

In our example, the `VERSION` column is used to achieve optimistic locking.

❷ The object is passed to higher layers in the application where it may be converted to a different representation, such as the one shown in the following example in which an HTML page is rendered:

```
<form method="POST" action="UpdateEmployee">
  <input type="HIDDEN" name="id" value="164205">
  <input type="HIDDEN" name="version" value="50">
  First Name: <input type="TEXT" name="firstName" value="Richard">
  Last Name: <input type="TEXT" name="lastName" value="Exley">
  Gender:
    <select name="gender">
      <option value="Male">Male</option>
      <option value="Female" selected>Female</option>
    </select>
  Salary: <input type="TEXT" name="salary" value="349845">
  <input type="submit" value="Update">
</form>
```

Note that the `id` and `version` are hidden in the HTML so that they can be returned to the server when the form is submitted.

❸ The object is returned to the application server and reconstituted as a TopLink object.

❹ The TopLink object, along with its changes, is saved to the database through the read/write connection pool, for example:

```
UPDATE EMPLOYEE SET GENDER = 'M', VERSION = 51 WHERE ((EMP_ID =
164205) AND (VERSION = 50))
```

The `EMP_ID` and `VERSION` columns are specified in the `WHERE` clause of the `UPDATE` statement. If the `VERSION` has changed in the database, then the query will not find any rows to update, causing TopLink to raise a locking exception.

❺ As soon as the `UPDATE` statement is committed, Oracle Data Guard transports the commit redo record to the standby database where it is applied to make the transaction updates visible.

Benefits

The combination of Oracle Active Data Guard and Oracle TopLink provides the following benefits:

- Increased database capacity and reduced workload on the primary database server
- Improved performance for all transactions
- Increased utilization of standby databases for other purposes in addition to disaster recovery

Increased Database Capacity and Reduced Primary Workload

By adding an Active Data Guard standby database to your existing environment, you increase the read capacity of the system and offload the read-only workload from the primary database to the standby, as shown in Figure 3.

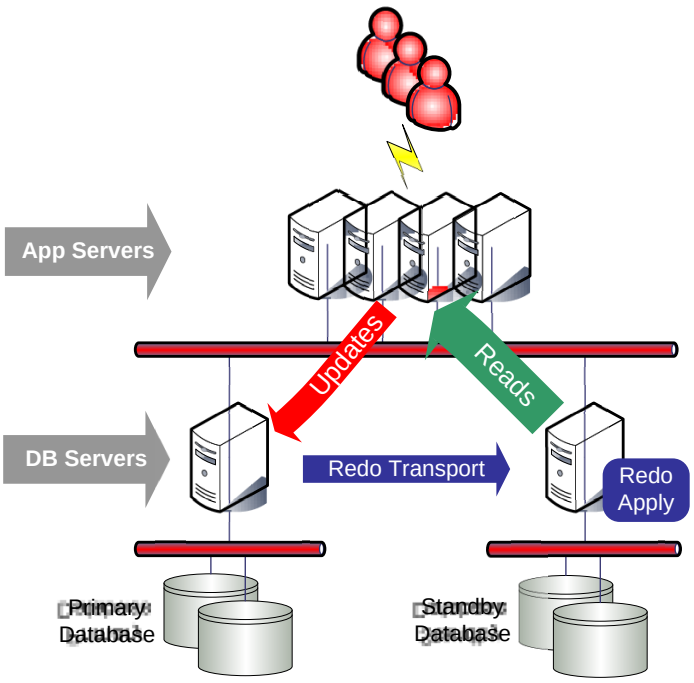


Figure 3: Increasing Database Capacity and Reducing Primary Database Load

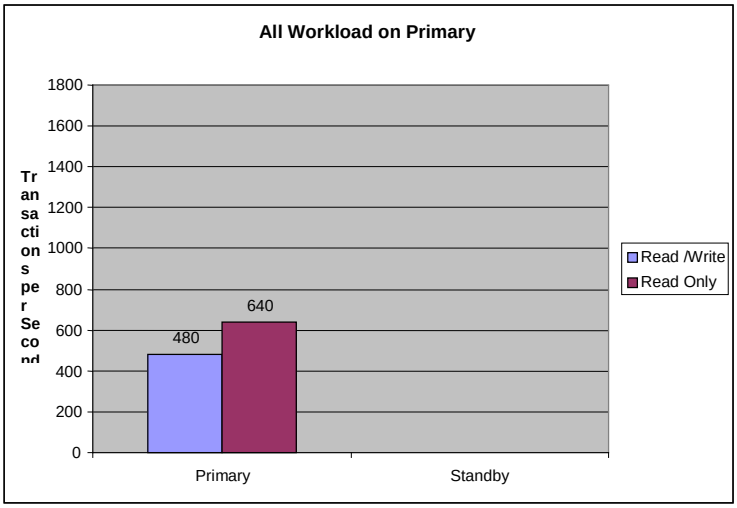
Improved Performance for All Transactions

Improving the performance of read-only queries on a standby database can improve the performance and scalability for all workloads. This is illustrated in MAA tests using Swingbench, which is an Oracle load-generation tool that includes an Order Entry application. The application consists of five transaction types typical of any order-entry application: Creating a Customer, Browsing Products, Placing Orders, Processing Orders, and Reviewing Orders.

All workloads are initially run on the primary database to establish a performance baseline. The workloads include:

- A read/write database service for adding new customers and order processing (50 users)
- A read-only database service for browsing catalogs and verifying orders (100 users).

When both workloads run on the primary database, the production host is operating at 100% capacity. Figure 4 shows how many transactions per second you can achieve when all workloads



run on the primary database.

Figure 4: All Workloads Running on the Primary Database

The second phase of the MAA testing moved the read-only service to an Active Data Guard standby database. By using the standby database for read-only processing, you can eliminate I/O contention for scarce system resources between read/write and read-only workloads.

Figure 5 shows how the primary and standby databases are each able to process two- to-three times as many transactions in the same unit of time. As an added benefit, CPU utilization on the primary host has decreased from 100% to just 60% of capacity, providing more room for future growth.

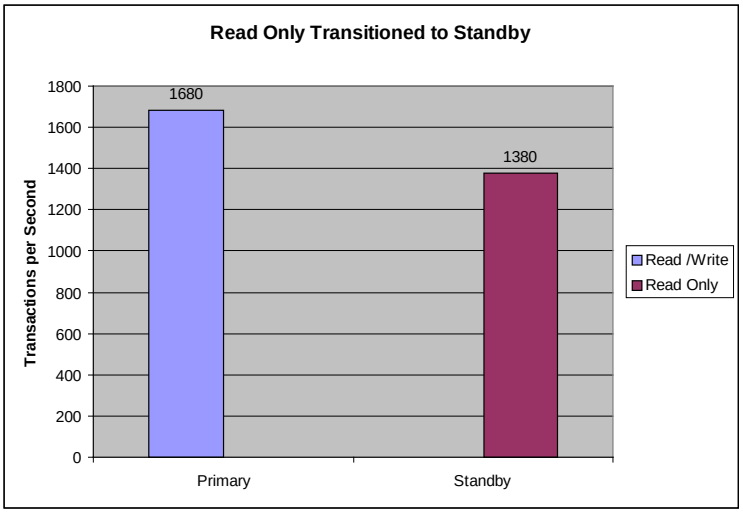


Figure 5: Read-Only Transactions Transitioned to Standby Database

Active Data Guard can also provide additional benefits for globally distributed user populations that access a central database for read-only queries. (For example: *follow-the-sun* call centers, e-commerce sites, Web access to online financial statements.) Remote users may suffer long query times due to limitations in application throughput and network latency. Active Data Guard can be deployed regionally to improve query performance, while providing high availability and disaster protection in the case of network outages or primary database failure.

Figure 6 shows a globally distributed configuration where two Active Data Guard standby databases have been configured at remote geographies to provide read-only access to local users.

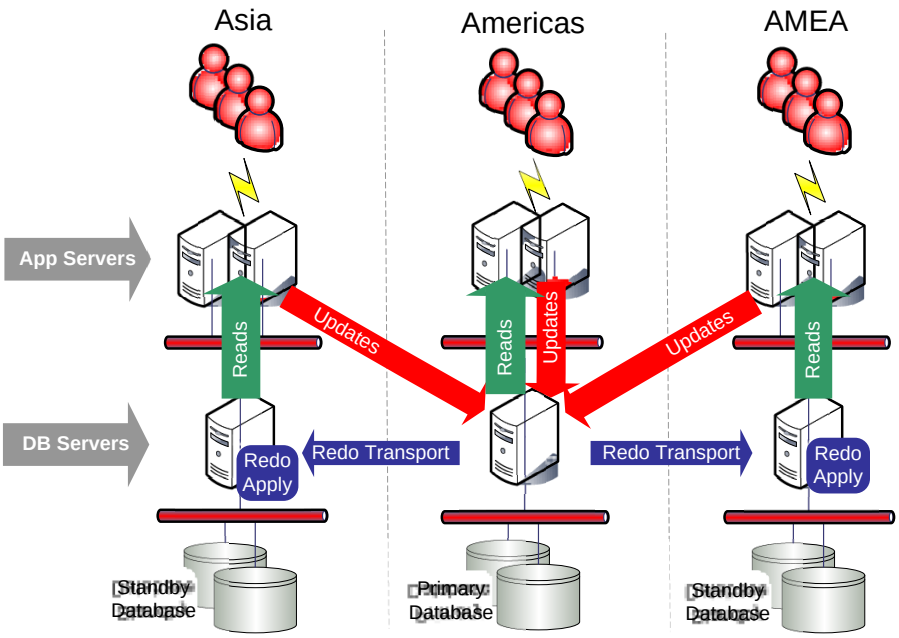


Figure 6: Improving Query Performance for Geographically Distributed Users

Increased Use of Disaster Recovery Assets

It is important to have a secondary site to protect against data loss and to maintain high availability in the event of a planned or unplanned outage of your primary database. Active Data Guard provides this level of protection and high availability while enabling the secondary site to be used for read-only workloads while serving in the standby role.

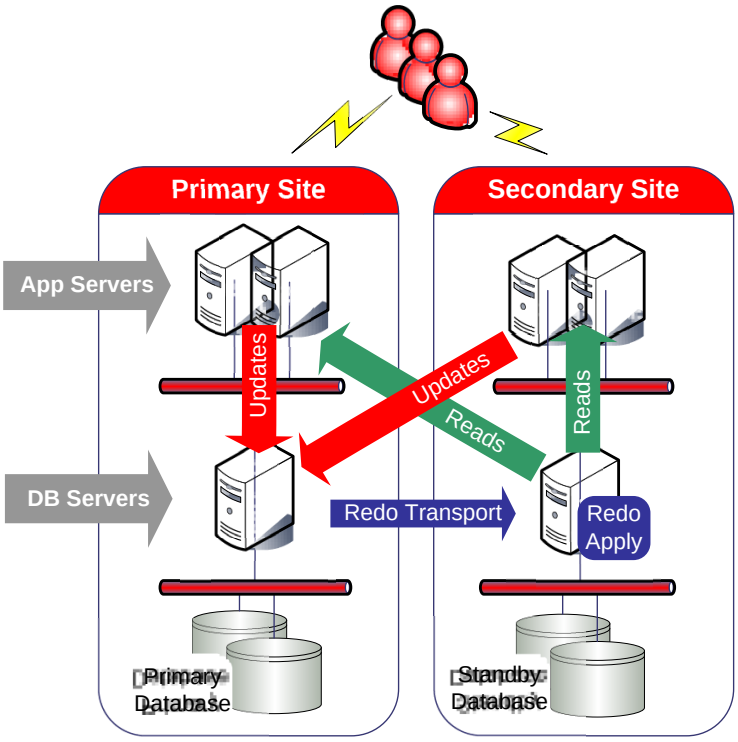


Figure 7: Increasing ROI of Disaster Recovery Assets

Design and Implementation Considerations

Redo Transport and Apply Services

The Active Data Guard standby database is kept up-to-date through the transportation and application of redo records received from the primary database. While offloading the read-only workload from the primary database to the Active Data Guard standby database can improve performance, the time taken for a change committed on the primary database to become visible on the standby can be influenced by transaction load, server capacity, network latency, and bandwidth, and the Oracle Data Guard configuration practices used. See the following

references to optimally configure and tune the Data Guard redo transport and apply according to your requirements:

- "Active Data Guard Best Practices" white paper (includes Redo Apply Best Practices)
- *Oracle Data Guard Concepts and Administration*
- "Data Guard Redo Transport & Network Configuration" white paper

However small, there will inevitably be some delay in applying changes to the standby database and some data read from the standby database may not be completely current with the primary database. In most cases, the lag time will be undetectable to the end user and totally acceptable when balanced with the additional benefits. However, the delayed effect should be considered carefully when deploying an Active Data Guard database solution.

For users and components that must always have the most current data, you can configure a separate TopLink session that uses the production database instead of the read-only connection.

Optimistic Locking

Data locking is an important concern when implementing a multiple-user system. How does TopLink handle the case where two or more users wish to update the same data at the same time?

One approach, known as **pessimistic locking**, locks the data when it is first selected (for example, using a `SELECT FOR UPDATE` statement). However, locking the data reduces throughput by serializing transactions and it is rarely used in highly concurrent environments. The pessimistic locking option cannot be used when reading from an Active Data Guard standby database.

Optimistic locking offers an alternative approach that provides higher concurrency and it can be used with Active Data Guard. See the Descriptors and Locking section of the *Oracle Fusion Middleware Developer's Guide for Oracle TopLink* for details about how optimistic locking is configured in Oracle TopLink. In most cases, you should configure optimistic locking for objects that are to be read from the standby database and updated on the primary.

Load Balancing

The placement of read-only services must be considered carefully so that the database load is distributed evenly across the available database servers. In many applications, the read load is much greater than the update load. Thus, moving the entire read load to the standby database may leave the primary with little to do. It is possible to start the read-only service on the primary and standby databases, if necessary, to balance the load more evenly.

Application Tuning

Some applications may be coded to re-read an object from the database immediately after an update. This is unnecessary because the object is already available and so re-reading the object is wasteful of system resources. When using the Active Data Guard standby database, there is an additional adverse effect.

If the object is re-read from the standby database immediately after it has been updated in the primary database, then the pre-updated version of the object will most likely be returned because of the apply delay described earlier in the "Redo Transport and Apply Services" section. This will be confusing for the end user (their update will seem to have disappeared) and will have knock-on effects, such as a false locking exception if the object is further updated. If this condition is observed in the application, then there are two ways to remedy the problem:

- Fix the application code to remove the additional read I/Os after the update is performed. See Task 7 in "Appendix A" for an example of how to remove the read I/Os.
- Implement TopLink object caching so that re-reads are returned from cache memory.

Alternative Replication Technologies

This paper focuses on the use of Oracle Active Data Guard to provide a read-only database replica. Although not explored here, it is also feasible to implement a similar solution using Oracle Advanced Replication, Oracle Streams, or the Oracle Data Guard logical standby database. All of these solutions provide read access to replicated data, along with other significant benefits of a target database that is opened for read/write I/O. As always, your requirements should drive the solution you choose.

If read-only access will address your business requirement, then Active Data Guard is:

- The simplest, most transparent, and highest performing method of maintaining an active, synchronized replica of your primary database that is open read-only.
- The fastest way to make existing disaster recovery sites active, without the complexity of traditional replication solutions.
- The best way to avoid byproducts of increased management complexity, greater chance of human error, and more potential for data loss and downtime.

Application and Database Configuration

This section explains how you can reconfigure an existing TopLink application and database to establish an Active Data Guard standby database and begin to route read-only traffic to the standby.

Configuration examples are taken from the TopLink Servlet JSP Example Application. The full steps for configuring this example are included in Appendix A.

Task 1: Configure an Active Data Guard Standby Database

Follow the steps in *Oracle Data Guard Concepts and Administration* to create and configure the standby database. In particular, see the following:

- Follow the instructions in the "Creating a Physical Standby" section to create and start a Data Guard physical standby database.
- Follow the instructions in the "Opening a Physical Standby" section to open the standby database for read-only access (Active Data Guard).

Note: Oracle Active Data Guard is supported beginning with Oracle Database 11*g* Release 1 (11.1) Enterprise Edition.

Task 2: Create and Start a Read-Only Database Service

Using SQL*Plus statements:

1. Connect to the standby database with system privileges
2. Execute the following commands to start a read-only database service:

```
EXECUTE DBMS_SERVICE.CREATE_SERVICE(SERVICE_NAME => 'RO'-
, NETWORK_NAME => 'RO');

EXECUTE DBMS_SERVICE.START_SERVICE(SERVICE_NAME => 'RO');
```

Task 3: Configure an Additional Connection Pool

The way you configure an additional connection pool differs depending on your application and how it is deployed. For example, the `DATA-SOURCES.XML` configuration would be configured as follows:

```
<managed-data-source connection-pool-name="TopLink Examples RO
Connection Pool"
  jndi-name="jdbc/TopLinkDSRO"
```

```
name="TopLinkDS"
tx-level="local"/>
```

See Appendix A, Task 3 and Task 4, for a detailed example.

If the read-only database service is to be started on several database servers, then all servers should be listed in the database connection URL, for example:

```
url= "jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)
(ADDRESS=(PROTOCOL=TCP)(HOST=<DB_HOST1>)(PORT=<DB_PORT1>))
(ADDRESS=(PROTOCOL=TCP)(HOST=<DB_HOST2>)(PORT=<DB_PORT2>))
(CONNECT_DATA=(SERVICE_NAME=RO)))"
```

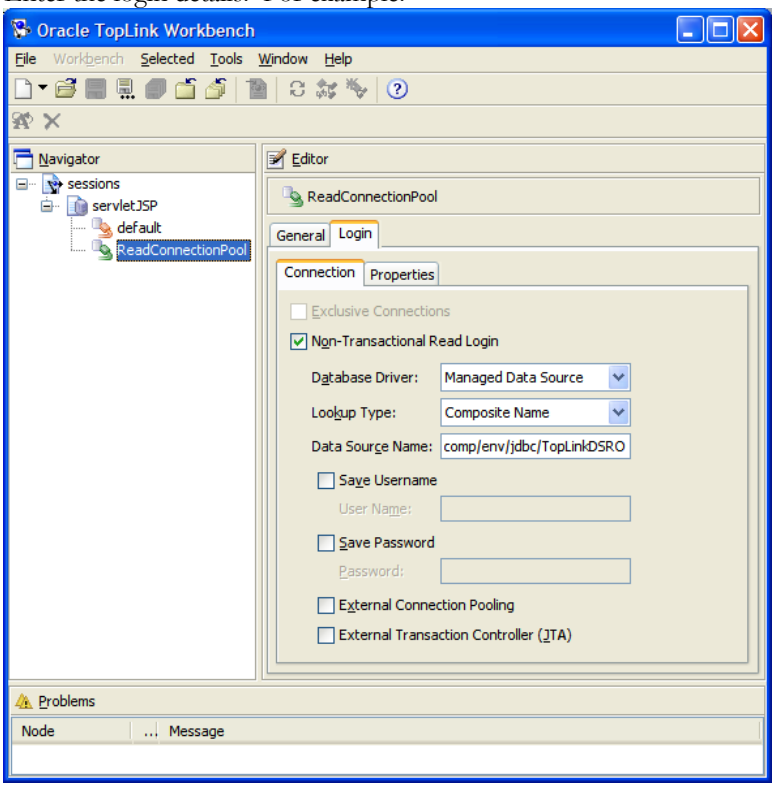
In the example:

- `<DB_HOST1>`, `<DB_HOST2>` are the network names of the database hosts
- `<DB_PORT1>`, `<DB_PORT2>` are the network port numbers of the database hosts

Task 4: Add the ReadConnectionPool Login to Sessions.xml

1. Open the applications `SESSIONS.XML` file using the Oracle TopLink Workbench.
2. Select **ReadConnectionPool** option.
3. Select **Non-Transactional Read Login**.

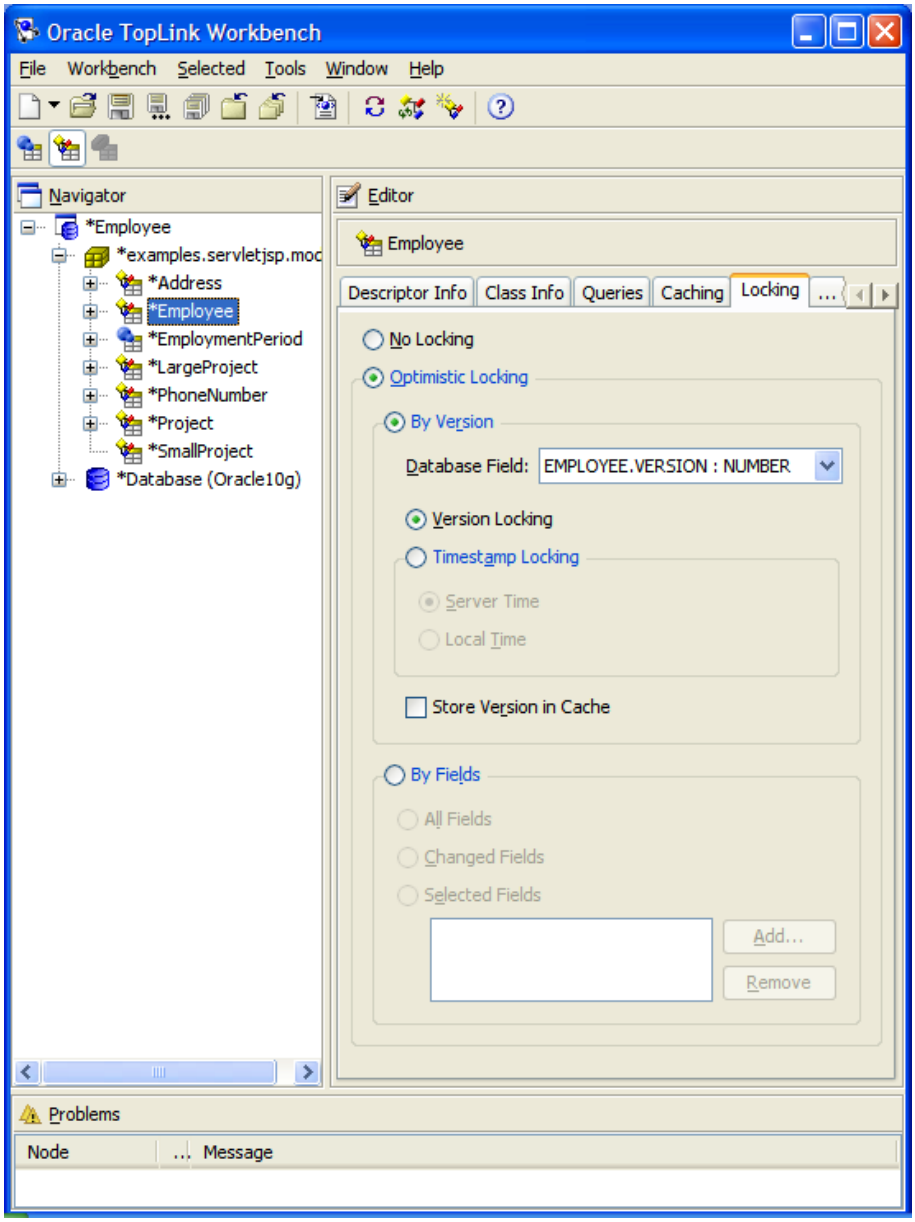
4. Enter the login details. For example:



The `Data Source Name` refers to the JNDI name of the data source created in Task 3.

Task 5: Review the Optimistic Locking Configuration

Open the Oracle TopLink Workbench and select the "Locking" tab for each object that is used being used by the application. For example:



It is recommended that you configure optimistic locking for all objects that may be read from the standby database and updated on the primary database.

Task 6: Build and Deploy the Application

Build and deploy the application in a test environment and perform functional and load testing.

Appendix A: Configuring the ServletJSP Example

This appendix provides detailed steps to configure a ServletJSP example application for operation with Oracle Active Data Guard. The ServletJSP example is located in Oracle Containers for J2EE (OC4J) 10.1.3 Examples bundle referenced from Oracle TopLink Examples.

The discussion in this appendix assumes the ServletJSP example is installed, configured and running against an Oracle database (as described in the example application's documentation).

> **Note:** this configuration must be running Oracle Database 11*g* Release 1 (11.1) Enterprise Edition (or later) to support Oracle Active Data Guard.

After the ServletJSP example is up and running, then complete the following tasks to reconfigure the application for use with Active Data Guard:

Task 1: Configure an Active Data Guard Standby Database

Follow the instructions in *Oracle Data Guard Concepts and Administration* to create and configure the standby database. In particular, perform the following tasks:

1. Create a physical standby database

   See the "Creating a Physical Standby Database" chapter, which steps you through the process of creating a physical standby database

2. Configure real-time apply

   See the "Apply Services" chapter

3. Open the standby database for read-only access (Active Data Guard)

   See the "Opening a Physical Standby Database" section (Note that the documentation describes Active Data Guard in terms of opening the standby database for *real-time queries*

Task 2: Create and Start a Read-Only Database Service

Using SQL*Plus statements:

1. Connect to the standby database with system privileges.
2. Execute the following commands:

```
EXECUTE DBMS_SERVICE.CREATE_SERVICE(SERVICE_NAME => 'RO'-
, NETWORK_NAME => 'RO');

EXECUTE DBMS_SERVICE.START_SERVICE(SERVICE_NAME => 'RO');
```

Task 3: Add Read-Only Data Source to DATA-SOURCES.XML

Edit the `DATA-SOURCES.XML` file, which is located in the `…\toplink\examples\oc4j\examples\servletjsp\config` folder file and add the following definitions:

```
<connection-pool name="TopLink Examples RO Connection Pool">
  <connection-factory factory-
class="oracle.jdbc.pool.OracleDataSource"
    user="scott"
    password="tiger"
    url=
"jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=(ADDRE
SS=(PROTOCOL=TCP)(HOST=<DB_HOST>)(PORT=<DB_PORT>)))(CONNECT_DATA=(SER
VICE_NAME=RO)))"
>
  </connection-factory>
</connection-pool>

<managed-data-source connection-pool-name="TopLink Examples RO
Connection Pool"
  jndi-name="jdbc/TopLinkDSRO"
  name="TopLinkDS"
  tx-level="local"/>
```

In the example:

- `<DB_HOST>` is the network name of the database host
- `<DB_PORT>` is the network port name of the standby database host

The `URL` of the `connection-pool` points to the database service created in Task 2.

Task 4: Add the New Data Source to web.xml:

Edit the `WEB.XML` file that is located in the `…\toplink\examples\oc4j\examples\servletjsp\config` folder and add the following definitions:

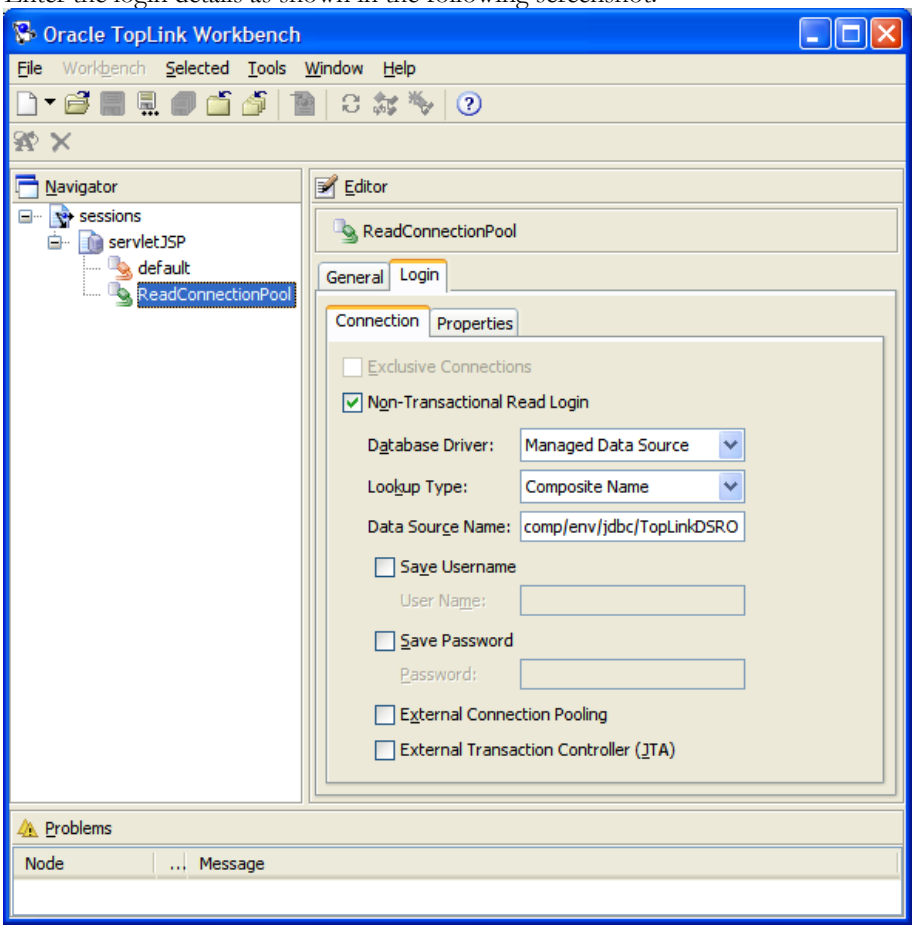
```
<resource-ref>
  <description>Non JTS RO DataSource</description>
  <res-ref-name>jdbc/TopLinkDSRO</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>SERVLET</res-auth>
</resource-ref>
```

In the example, the *res-ref-name* element refers to the `jndi-name` of the `managed-data-source` created in Task 4.

Task 5: Add the ReadConnectionPool Login to SESSIONS.XML

The `SESSIONS.XML` file is located in the `…\toplink\examples\oc4j\examples\servletjsp\config` folder. To add the Read Connection Pool login details to the `SESSIONS.XML` file:

1. Open the file using the Oracle TopLink Workbench.
2. Select **ReadConnectionPool**.
3. Select **Non-Transactional Read Login**.
4. Enter the login details as shown in the following screenshot:

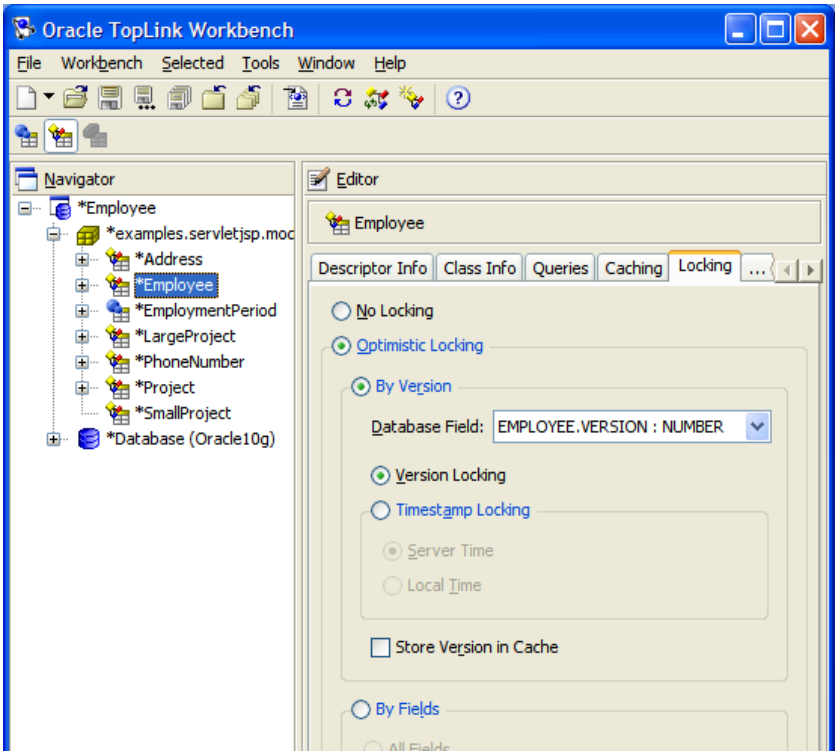


The Data Source Name refers to the `jndi-name` of the `managed-data-source` that was created in Task 4.

Task 6: View the Optimistic Locking Configuration

The `EMPLOYEE.MWP` file is located in the `…\toplink\examples\oc4j\examples\servletjsp\mw` folder. To view the optimistic locking configuration, perform the following steps:

1. Open the file using the Oracle TopLink Workbench
2. Select the **Locking** tab for the "Employee" object, as shown in the following screenshot:



Task 7: Remove the Re-Read After Update

One case of re-reading after update was found in the example application. The following code line in `UpdateEmployee.java` file causes the object to be re-read by querying on the object ID:

```
response.sendRedirect("EditEmployee?id=" + id);
```

This line was replaced by the following code, which passes the existing object—as an attribute—to the `editEmployee.jsp` file:

```
// Store employee in attribute for editEmployee.jsp
request.setAttribute("employee", employee);

request.getRequestDispatcher("./editEmployee.jsp").forward(request,
response);
```

Note: TopLink object caching is implemented by default in this application. Thus, the problem will not become apparent unless caching is switched off.

Task 8: Build and Deploy the Application

Perform the steps in the example application's documentation to build and deploy the application.

Configuring Oracle TopLink Applications with Oracle Active Data GuardSeptember 2009
Author: Richard Exley
Contributing Authors: Joseph Meeks
Editor: Viv Schupmann

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

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

0109