Reducing Siebel Downtime with a Local Standby Database

Oracle Maximum Availability Architecture White Paper
November 2008

Maximum Availability Architecture

Oracle Best Practices for High Availability

ORACLE

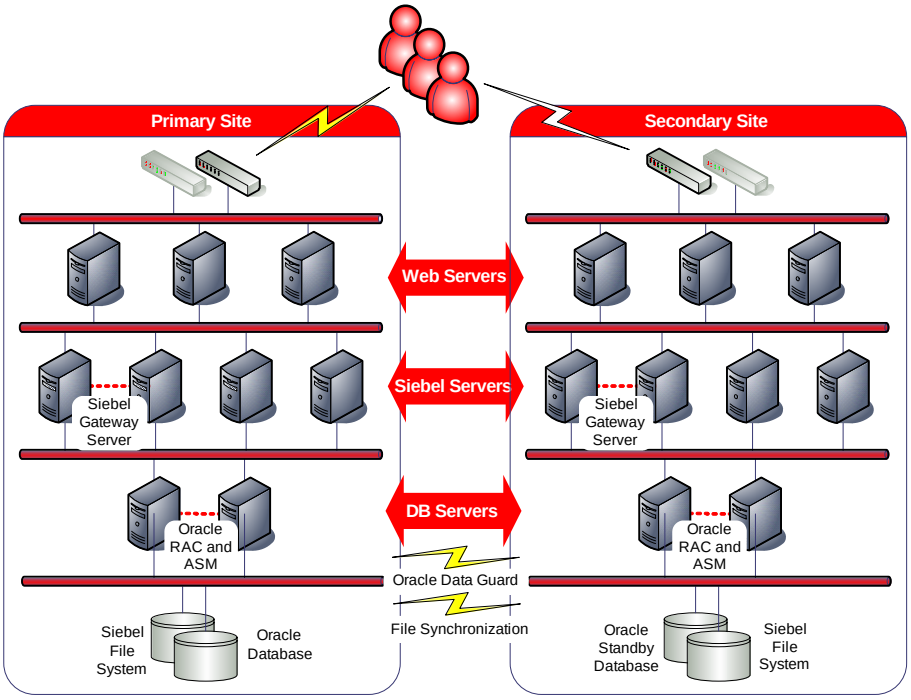
Reducing Siebel Downtime with a Local Standby Database

Reducing Siebel Downtime with a Local Standby Database

EXECUTIVE SUMMARY

Oracle Maximum Availability Architecture (MAA) is Oracle's best practices blueprint based on proven Oracle high availability technologies and recommendations. The goal of MAA is to achieve the optimal high availability architecture at the lowest cost and complexity. Papers are published on the Oracle Technology Network (OTN) - http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm.

The Siebel Maximum Availability Architecture (MAA) is a best practice blueprint for achieving an optimal Siebel high availability deployment using Oracle high availability technologies and recommendations.



For a complete description of Siebel MAA see –

http://www.oracle.com/technology/deploy/availability/pdf/siebelmaa.pdf.

In addition to Siebel MAA, the sections that follow move beyond Data Guard's traditional use for remote data protection by describing how to setup and operate a logical or physical standby database local to the primary database to achieve high availability in various recovery and maintenance scenarios. It is also worth noting that Data Guard's ability to support multiple standby databases in the same configuration means that the concepts of local and remote standby databases are not mutually exclusive; both can be deployed in the same configuration for optimal high availability and data protection.

All procedures were tested under load in Oracle's MAA lab. Siebel was configured to stay up during the database outage and reconnect and resume work when the database service returned. As a result, users only experienced a short pause during the outage and were able to continue working afterwards. There was no need to re-login or redo work, no errors were reported, and no transactions were lost.

The following table describes how a local standby database can be used to reduce downtime during unplanned outages:

Unplanned Outage	Oracle Solution	Downtime
Database Instance Hang or Failure	Diagnosis + Recovery	Hours
	Remote Site Failover	Minutes
	Local Standby Failover	Seconds (transparent)
	RAC Failover	Seconds (transparent)
Entire Database Failure or Corruption	Diagnosis + Recovery	Hours
	Remote Site Failover	Minutes
	Local Standby Failover	Seconds (transparent)

The following table describes how a local standby database can be used to reduce downtime during unplanned outages when compared with other solutions:

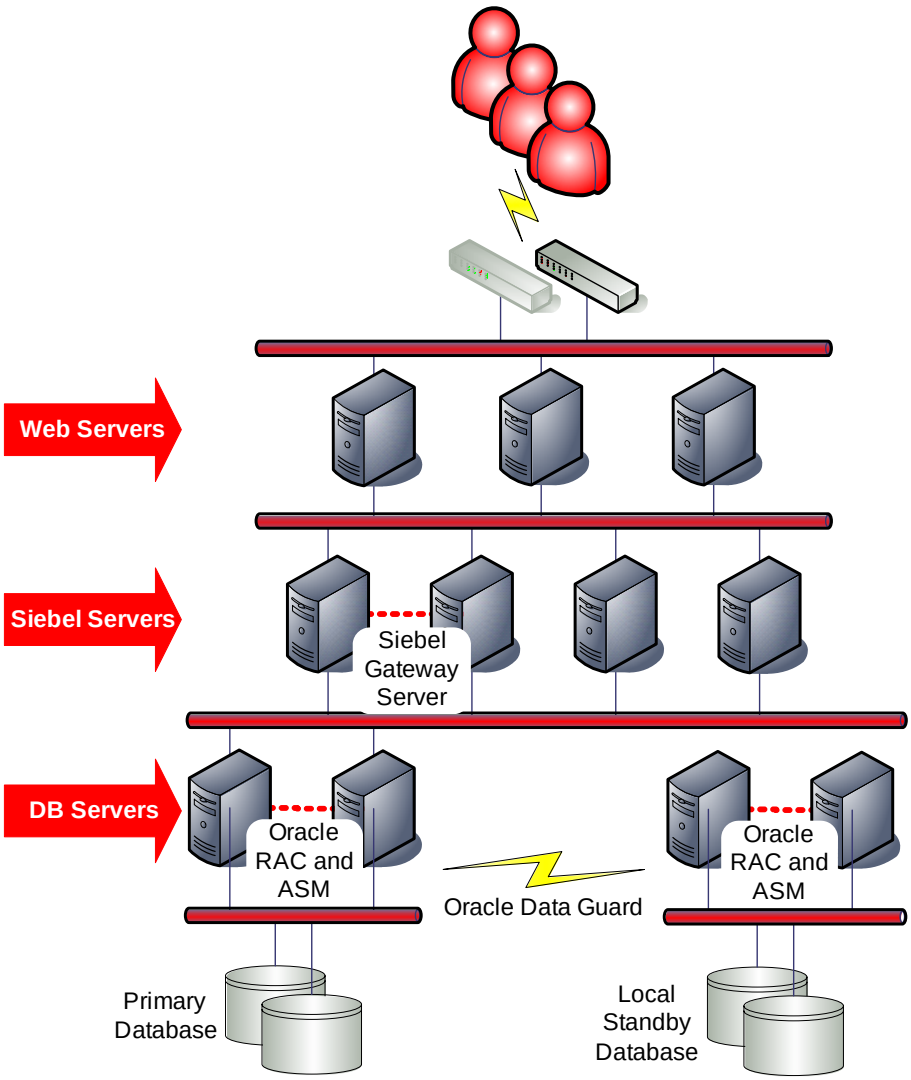
Planned Maintenance	Oracle Solution	Downtime
Database Interim Patch, DB Hardware or OS Upgrade	Shutdown + Upgrade + Startup	Minutes
	Remote Site Switchover	Minutes
	Local Standby Switchover	Seconds (transparent)
	RAC Rolling	No downtime
Database Patch Set or Upgrade	Shutdown + Upgrade + Startup	Hours
	Remote Site Switchover	Minutes
	Local Logical Standby Switchover	Seconds (transparent)
Database Transition to Oracle RAC and/or ASM, 10g ASM Upgrade	Shutdown + Upgrade + Startup	Hours
	Remote Site Switchover	Minutes
	Local Standby Switchover	Seconds (transparent)

For demonstrations of Siebel behavior during a database upgrade see http://www.oracle.com/technology/deploy/availability/demonstrations.html.

This document was written in the context of Siebel Business Applications, Version 8.0 running on Oracle Database 10g Release 2 and upgrading to Oracle Database 11g Release 1 and where necessary we make reference to the documentation of these product releases. In most cases, our recommendations and best practices can be applied to earlier releases too. If you are running earlier product versions we recommend you refer to the earlier product documentation.

ORACLE DATA GUARD AND ORACLE FLASHBACK DATABASE

In this paper we use a local Data Guard standby for our testing. By local we mean that the Siebel mid-tiers can connect to either the primary or standby databases and a switchover or failover can be achieved without restarting Siebel. This can be contrasted with a remote standby where the Siebel mid-tiers and database tier must failover together. Note: it is possible to provide a local standby solution even if the primary and standby databases are geographically separated provided the network can handle the load and response time requirements of the application.



ORACLE DATA GUARD

Oracle Data Guard 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 these 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 the application downtime caused by the outage. Data Guard can be used with traditional backup, restore, and clustering solutions to provide a high level of data protection and data availability. Siebel supports both physical and logical standby databases. See also: Oracle Data Guard Concepts and Administration.

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 Apply, which recovers the redo data received from the primary database and applies the redo to the physical standby database.

As of Oracle Database 11g release 1 (11.1):

> A physical standby database can receive and apply redo while it is open for read-only access and so may be used for other purposes as well as disaster recovery.
>
> With a single command, a physical standby database can also be converted into a Snapshot Standby and become an independent database open read-write, ideal for QA and other testing. The Snapshot Standby continues to receive and archive redo data from the primary database while it is open read-write, thus protecting primary data at all times. When testing is complete, a single command will convert the snapshot back into a standby database, and automatically resynchronize it with the primary.
>
> A physical standby database can be also used for rolling database upgrades using the SQL Apply process – and return to its function as a physical standby database once the upgrade is complete.

A logical standby database contains the same logical information as the production database, although the physical organization and structure of the data can be different. The logical standby database is kept synchronized with the primary database through SQL Apply, which transforms the data in the redo received from the primary database into SQL statements and then executes the SQL statements on the standby database. A logical standby database can be used for disaster recovery and reporting requirements, and can also be used to upgrade the database software and apply patch sets while the application is online and with almost no downtime.

It is possible to deploy a local standby database at the primary site as well as a remote standby at the secondary site. This offers the advantage that a failover to the local standby can be performed while the Siebel Servers continue running and can be done almost transparently to the end users. It also offers the ability to perform a rolling database upgrade without the need to switch to another site (Siebel Servers remain online throughout the rolling database upgrade). We would recommend that a local and remote standby be deployed to achieve maximum availability. Note, transparent failover (failover while Siebel stays up) should only be performed if no committed data is lost. This will constrain your choice of data protection mode, discussed below.

Oracle Data Guard supports three distinct modes of data protection and it is important to select the appropriate mode to meet your needs:

Maximum protection This protection mode ensures that no data loss will occur if the primary database fails. To provide this level of protection, the redo data needed to recover each transaction must be written to both the local online redo log and to the standby redo log on at least one standby database before the transaction commits. To ensure data loss cannot occur, the primary database shuts down if a fault prevents it from writing its redo stream to the standby redo log of at least one transactionally consistent standby database.

Maximum availability This protection mode provides the highest level of data protection that is possible without compromising the availability of the primary database. Like maximum protection mode, a transaction will not commit until the redo needed to recover that transaction is written to the local online redo log and to the standby redo log of at least one transactionally consistent standby database. Unlike maximum protection mode, the primary database does not shut down if a fault prevents it from writing its redo stream to a remote standby redo log. Instead, the primary database operates in maximum performance mode until the fault is corrected, and all gaps in redo log files are resolved. When all gaps are resolved, the primary database automatically resumes operating in maximum availability mode.

Maximum performance This protection mode (the default) provides the highest level of data protection that is possible without affecting the performance of the primary database. This is accomplished by allowing a transaction to commit as soon as the redo data needed to recover that transaction is written to the local online redo log. The primary database's redo data stream is also written to at least one standby database, but that redo stream is written asynchronously with respect to the transactions that create the redo data.

ORACLE FLASHBACK DATABASE

Flashback Database enables you to rewind the database to a previous point in time without restoring backup copies of the data files. Flashback Database is a recovery feature that operates on only the changed data. With Flashback Database, the time it takes to correct an error is less than the time it takes to cause and detect the error, without recovery time being a function of the database size. You can flash back a database using a single RMAN command or SQL*Plus statement instead of using a complex procedure. For further details and best practices see: http://download.oracle.com/docs/cd/B28359_01/backup.111/b28270/rcmflash.htm#BGBDCAFA and Metalink Note 565535.1.

SETUP PROCEDURES

This section describes how to setup Siebel and standby databases to minimized downtime during database outages.

CONFIGURATION TO MINIMIZE DOWNTIME DURING DATABASE OUTAGES

Siebel supports Oracle Transparent Application Failover (TAF). TAF enables Siebel database connections to transparently failover to a surviving database or instance when a database connection is lost. TAF can be configured to fail over to another Oracle RAC instance, an Oracle Data Guard standby database, or even to the same database in the case of a database shutdown and restart. The Siebel Servers and Clients continue running during the failover and do not need to be restarted. The following table summarizes Siebel behavior during failover when TAF is configured for various client operations. Besides a short pause as the failover occurs, the failure is transparent to the end user.

Siebel Client Operation	Behavior
Web client user is updating data and steps-off (saves) the updates during or just after the DB failure.	Oracle reconnects and reconstructs the database session on a surviving node and Siebel resubmits the update.
Web client user is paging through queried data when the DB failure occurs.	Oracle reconnects and reconstructs the database session on a surviving node, re-executes the query, repositions the SQL cursor, and returns the next set of rows.
Web client user is issuing a new query or switching screens just after the DB failure.	Oracle reconnects and reconstructs the database session on a surviving node.

To minimize downtime during planned and unplanned database outages it is important that TAF is configured. To configure TAC and take full advantage of TAF functionality it is essential that the following configuration changes are made:

1. A database service is created for Siebel database connections and the service is configured for TAF.
2. The client side Oracle Net configuration points to the database service, not a specific instance, and includes all primary and standby database listeners. This will ensure that Siebel can connect regardless of where the service is started.

3. The client side Oracle Net configuration includes the SQLNET.OUTBOUND_CONNECT_TIMEOUT parameter. This will ensure that Siebel quickly skips unavailable listeners when connecting to the database.
4. In an Oracle RAC environment, a clusterware managed database service is created for the primary and standby databases. This is necessary to ensure that connections will only be made to open database instances.
5. TCP keep-alive timeout should be configured on Siebel Servers. This will ensure faster failure detection in the event of a database node failure.
6. For customers running Oracle Database 10.2.0.2 and 10.2.0.3, the prerequisites for automatic client failover are applied.
7. For customers running or upgrading to Oracle Database 11.1.0.6 with logical standby, the prerequisites are applied.

Creating a Database Service and Configuring for TAF

A database service provides a simple named access point to the database. A service can physically span multiple instances in an Oracle RAC cluster and can be moved from one database to another. By requiring Siebel to connect only through a service we are able to relocate or reconfigure the service without reconfiguring Siebel.

A database service can be created and configured through Enterprise Manager or using the dbms_service package. The "SELECT" failover type and "BASIC" failover method are supported by Siebel and should be adequate for most configurations. For example, this is how a service can be created with the dbms_service package:

```
SQL> EXECUTE DBMS_SERVICE.CREATE_SERVICE (-
SERVICE_NAME => 'SIEBEL' -
, FAILOVER_METHOD => DBMS_SERVICE.FAILOVER_METHOD_BASIC -
, FAILOVER_TYPE => DBMS_SERVICE.FAILOVER_TYPE_SELECT -
, CLB_GOAL => DBMS_SERVICE.CLB_GOAL_LONG);
```

Note, the failover_retries and failover_delay parameters have no effect on Siebel behavior. Siebel will always retry 24 times with a delay of 5 seconds.

See Metalink Note 460982.1 - How To Configure Server Side Transparent Application Failover for details on how to create a database service and configure for TAF.

Creating Triggers to Manage the Database Service

The database service created in the previous section should only be started when the database is in the primary role. If the database switches from primary to the standby role, the service should be stopped and all sessions should be

disconnected. We can automatically manage these transitions with database triggers - one that is triggered on database startup, and one that is triggered on a database role change. Note: The "startup" trigger is only required for logical standby database. Here are examples:

```
CREATE OR REPLACE TRIGGER startup_Siebel after startup on
database
DECLARE
  role VARCHAR(30);
BEGIN
  SELECT DATABASE_ROLE INTO role FROM V$DATABASE;
  IF role = 'PRIMARY' THEN
    DBMS_SERVICE.START_SERVICE('salesJDBC');
  ELSE
    DBMS_SERVICE.STOP_SERVICE('salesJDBC');
  END IF;
END;

CREATE OR REPLACE TRIGGER role_change_Siebel after
db_role_change on database
DECLARE
  role VARCHAR(30);
BEGIN
  SELECT DATABASE_ROLE INTO role FROM V$DATABASE;
  IF role = 'PRIMARY' THEN
    DBMS_SERVICE.START_SERVICE('salesJDBC');
  ELSE
    DBMS_SERVICE.STOP_SERVICE('salesJDBC');
  END IF;
END;
```

Client Side Oracle Net Configuration for TAF

To connect to the service the SERVICE_NAME parameter is used in the TNSNAMES.ORA file. Because the service may be running on any server it is important that all database listeners are included in the ADDRESS_LIST. Here is an example TNSNAMES.ORA configuration with a two node Oracle RAC primary and a two node Oracle RAC standby:

```
SEBL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS=(PROTOCOL=TCP)(HOST=p2-vip)(PORT=1521))
      (ADDRESS=(PROTOCOL=TCP)(HOST=p1-vip)(PORT=1521))
      (ADDRESS=(PROTOCOL=TCP)(HOST=s2-vip)(PORT=1521))
      (ADDRESS=(PROTOCOL=TCP)(HOST=s1-vip)(PORT=1521))
    )
    (LOAD_BALANCE=on)
    (CONNECT_DATA=
      (SERVER=DEDICATED)
      (SERVICE_NAME=SIEBEL)
      )
    )
  )
```

See Metalink Note 453293.1 - 10g: Configuration of TAF (Transparent Application Failover) and Load Balancing for further details.

When connecting or reconnecting to the database, Siebel will keep trying all the addresses in the address list until a connection is established. It is important that it doesn't get "hung-up" on any one address, as this will delay connection.

The SQLNET.OUTBOUND_CONNECT_TIMEOUT parameter dictates the maximum amount of time (in seconds) that we wait for a response from an address before skipping to the next address. A setting of 3 seconds is recommended and is usually acceptable in most cases. This parameter should not be set to less than 3 seconds. Here is an example SQLNET.ORA:

```
SQLNET.OUTBOUND_CONNECT_TIMEOUT=3
```

The above configuration changes should be made for all Siebel clients and servers that connect directly to the database.

Creating Cluster Managed Database Services

In an Oracle RAC environment it is recommended that an Oracle Clusterware managed database service be created for Siebel connections to the database. This is necessary to ensure that connections will only be made to open database instances. Oracle Clusterware managed database services are created through Enterprise Manager or with the srvctl command. If the service has already been created in the database then this command will merely add a cluster resource and use the existing service. Here is an example using srvctl:

```
srvctl add service -d SEBLRAC -s SIEBEL -r
"SEBLRAC1,SEBLRAC2" -P basic
```

If a service is created with srvctl, the server side TAF attributes are adjusted using DBMS_SERVICE.MODIFY_SERVICE.

Configuring TCP Keepalive Timeout

Siebel currently does not support Oracle Fast Application Notification (FAN) and so it is necessary to reduce the TCP Keepalive Timeout for the Siebel Servers to release database connections in the event of a database node crash. This is only for the rare case where the database node crashes before the TCP connections can be cleaned up, and only for connections where a database request was in-flight at the time of failure or a new request was started before the Virtual Internet Protocol (VIP) Address could be switched to a surviving node. In all other cases the database connection failure is detected immediately and a new connection is established on a surviving node.

Please refer to Metalink Note 249213.1 - Performance problems with Failover when TCP Network goes down (no IP address) for details on how to configure the TCP Keepalive timeout. Making these configuration changes may have adverse effect on network utilization and so all changes should be tested and monitored carefully.

Prerequisites for Oracle Database 10.2.0.2 and 10.2.0.3

See Metalink Note 405120.1.

Prerequisites for Oracle Database 11.1.0.6 with Logical Standby

If you are planning to use Oracle Database Release 11.1.0.6 in logical standby mode then you must apply patch 7198303 first. If you are upgrading to 11.1.0.6 using logical standby, apply the patch after the database upgrade but before the switchover. The patch is not required for Oracle Database Release 11.1.0.7 or later.

CREATING A PHYSICAL STANDBY

The process of creating a physical standby database is documented in the "Creating a Physical Standby Database" section of Oracle Data Guard Concepts and Administration. There are no special steps for a Siebel database. See also this paper for MAA best practices for an Oracle RAC physical standby - http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10g_RACPrimaryRACPhysicalStandby.pdf.

Note, if archiving is not enabled on your database it will be necessary to shutdown and restart the database to enable archiving. The steps in section "Transparent Database Shutdown and Restart" can be used to reduce Siebel downtime in this case.

CREATING A LOGICAL STANDBY

The process of creating a logical standby database is documented in the "Creating a Logical Standby Database" section of Oracle Data Guard Concepts and Administration. Please pay particular attention to the following steps in the creation process. See also this paper for MAA best practices for an Oracle RAC logical standby - http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10gR2_RACPrimaryRACLogicalStandby.pdf.

Supported Data Types and Storage Attributes

A logical standby database can maintain most, but not all Oracle database data types and storage attributes. The out-of-the-box Siebel database uses only supported types and attributes; however, any additional objects that have been created in the database must be checked. The procedure for checking the database is explained in the above guide.

Unique Row Identification

Oracle uses primary-key or unique-constraint/index supplemental logging to logically identify a modified row in the logical standby database. A "disabled primary-key RELY constraint" must be added to all tables that do not have a primary key. In an out-of-the-box Siebel database two tables were highlighted:

```
S_AUDIT_ITEM
S_DD_EXTRACT_CL
```

S_AUDIT_ITEM is unique in the ROW_ID column and is policed by the Siebel application. Run the following command to flag this for the logical standby database:

```
SQL> ALTER TABLE SIEBEL.S_AUDIT_ITEM ADD PRIMARY KEY
(ROW_ID) RELY DISABLE;
```

S_DD_EXTRACT_CL became obsolete in Siebel 7.7 and so is unlikely to be of concern.

OPERATIONAL PROCEDURES

In this section we describe the operational procedures associated with various Siebel database outage scenarios. Our goal is not to replicate the existing Data Guard documentation - Oracle Data Guard Concepts and Administration - rather to point out any additional steps or considerations when working with Siebel. It is important that you become familiar with the generic Oracle Data Guard documentation before working with Data Guard.

In the following sections we assume that the appropriate setup steps described in the previous section have already been performed.

In this section we use the word "transparent" to indicate the cases where the transition is achieved while Siebel stays up and the end users see a momentary pause (hour glass) but no error is reported and no data is lost.

TRANSPARENT FAILOVER PROCEDURES

Failover to a local standby, logical or physical, is typically used if the primary database service is lost. There are a number of scenarios where this may happen, for example if the database crashes and cannot be restarted, or if the database files have become corrupt. Having the facility to failover to a standby will typically save many hours of downtime in these scenarios. Having a local standby allows the failover to be performed transparently to the Siebel users.

It is also possible to transparently failover to the same database if it is shutdown and restarted.

Transparent Failover to Local Physical Standby Procedure

We assume the following pre-requisites:

- A local physical standby database is in place.
- Siebel is up and running against the primary database.
- A mechanism is in place to alert administrators in the event of a database failure so that they can respond accordingly. If fast start failover has been implemented then the database recovery is initiated automatically.

The loss of primary was simulated in our tests by a shutdown abort:

```
SQL> SHUTDOWN ABORT
```

When the primary database fails, all Siebel database connections are lost and TAF begins cycling through the address list trying to reconnect. The following actions are taken to complete the recovery:

1. Failover to the standby by executing the following commands against the standby database:

```
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE
FINISH FORCE;
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
SQL> ALTER DATABASE OPEN;
```

When the failover has completed the standby becomes the primary and the role change trigger starts the database service. TAF establishes connections to the new primary database and Siebel resumes working.

Note: If the standby database was ever opened read-only then it must be restarted prior to opening. For more details see: http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10gR2_SwitchoverFailoverBestPractices.pdf.

2. If and when the original primary database becomes available it can be reinstated as a standby for the new primary database. This can be done either by restoring a backup or by using flashback database. The steps are documented in Oracle Data Guard Concepts and Administration.

Discussion:

- The transparent failover to a local standby database should only be used if the standby database is kept in sync with the primary (Maximum Protection or Maximum Availability modes). If that is not the case then Siebel should be restarted and users should be alerted that data might have been lost.

Transparent Failover to Local Logical Standby Procedure

We assume the following pre-requisites:

- A logical standby database is in place.
- Siebel is up and running against the primary database.
- A mechanism is in place to alert administrators in the event of a database failure, or fast start failover has been implemented and the database recovery is initiated automatically.

The loss of primary was simulated in our tests by a shutdown abort:

```
SQL> SHUTDOWN ABORT
```

When the primary database fails, all Siebel database connections are lost and TAF begins cycling through the address list trying to reconnect. The following actions are taken to complete the recovery:

1. Failover to the logical standby by executing the following commands on the standby:

   ```
   SQL> ALTER DATABASE ACTIVATE LOGICAL STANDBY DATABASE
   FINISH APPLY;
   ```

   When the failover has completed the standby becomes the primary, the role change trigger starts the database service, TAF establishes connections to the new primary database, and Siebel resumes working.

2. If and when the old primary database becomes available it can be reinstated as a standby for the new primary database. This can be done by using flashback database or by following the steps to create a new logical standby. The steps are documented in Oracle Data Guard Concepts and Administration.

Discussion:

The transparent failover to a local standby database should only be used if the standby database is kept completely in sync with the primary. If that is not the case, Siebel should be restarted and users should be alerted that data might have been lost.

Transparent Database Shutdown and Restart Procedure

We assume the following pre-requisites:

- Siebel is up and running.
- There is a need to restart the database.

The following events unfold:

1. The primary database is shutdown:

   ```
   SQL> SHUTDOWN IMMEDIATE
   ```

   All Siebel connections are lost and TAF begins cycling through the address list trying to reconnect.

2. The database is down. Perform necessary configuration work.

3. Restart the database:

   ```
   SQL> STARTUP
   ```

   When the database opens the role change trigger starts the database service. TAF reestablishes connections to the database and Siebel resumes working.

TRANSPARENT SWITCHOVER PROCEDURES

Switchover to a local standby database, logical or physical, may become necessary if there is a degradation of the primary database service, for example, if the database becomes slow or hangs. It may also be used as part of a planned maintenance activity, such as a hardware upgrade or transition to Oracle RAC or ASM.

Having the facility to switchover to a standby will typically save many hours of downtime in these scenarios. Having a local standby allows the switchover to be performed transparently to the Siebel users.

Transparent Switchover to Local Physical Standby Procedure

We assume the following pre-requisites:

- A local physical standby database is in place.
- Siebel is up and running against the primary database.

The following steps are performed to complete the switchover:

1. The primary database commits to switch over to the standby role by running the following commands on the primary:

   ```
   SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO STANDBY WITH
   SESSION SHUTDOWN;
   ```

   All Siebel connections are lost and TAF begins cycling through the address list trying to reconnect.

2. Shutdown the primary database and restart in mount mode:

   ```
   SQL> SHUTDOWN IMMEDIATE
   SQL> STARTUP MOUNT
   ```

3. On the standby, wait for switchover status to become 'TO PRIMARY' or 'SESSIONS ACTIVE':

   ```
   SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE;
   ```

   Then, commit the standby to switch over to the primary role and open the database:

   ```
   SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
   SQL> ALTER DATABASE OPEN;
   ```

   When the database opens in the primary role the startup trigger starts the Siebel database service. TAF establishes connections to the new primary

database and Siebel resumes working.

Note: If the standby database was ever opened read-only then it must be restarted prior to opening. For more details see: http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10gR2_SwitchoverFailoverBestPractices.pdf.

3. The original primary now becomes the standby database. Begin redo apply on the new standby:

```
SQL> RECOVER MANAGED STANDBY DATABASE USING CURRENT
LOGFILE DISCONNECT;
```

4. Switch logs on the new primary:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
```

Discussion:

In this case all committed transactions on the primary will be recovered on the standby and no data is lost.

Because the original primary becomes a standby, it is possible to switch back if necessary. This "back-out" facility can significantly reduce the risk associated with system changes.

Transparent Switchover to Local Logical Standby

We assume the following pre-requisites:

- A local logical standby database is in place.
- Siebel is up and running against the primary database.

The following steps are performed to complete the switchover:

1. The primary database prepares to switch over to the standby role by running the following command on the primary:

```
SQL> ALTER DATABASE PREPARE TO SWITCHOVER TO LOGICAL
STANDBY;
```

2. The standby database prepares to switch over to the primary role by running the following command on the standby:

```
SQL> ALTER DATABASE PREPARE TO SWITCHOVER TO PRIMARY;
```

3. Wait until the switchover status on the primary changes to 'TO LOGICAL STANDBY':

```
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE;
```

Note that Siebel continues running normally up to this point.

4. On the primary, stop the database service, disconnect all the Siebel connections, and commit to the standby role:

```
SQL> EXECUTE DBMS_SERVICE.STOP_SERVICE('SIEBEL');
SQL> EXECUTE DBMS_SERVICE.DISCONNECT_SESSION('SIEBEL');
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO LOGICAL
STANDBY;
```

5. On the standby, wait until the switchover status becomes 'TO PRIMARY':

```
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE;
```

Then, commit the standby to switch over to the primary role:

```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
```

When the database changes to the primary role the role change trigger starts the Siebel database service. TAF establishes connections to the new primary database and Siebel resumes working.

6. The original primary now becomes the standby database. Begin logical apply on the new standby:

```
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY
IMMEDIATE;
```

7. Switch logs on the new primary:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
```

Discussion:

In this case all committed transactions on the primary will be recovered on the standby and no data is lost.

Because the original primary becomes a standby, it is possible to switch back if necessary. This capability can significantly reduce the risk of system changes.

Note: the procedure to switchover during a database upgrade is different from a regular switchover. See the "Transparent Database Upgrade using Local Logical Standby" section for details.

TRANSPARENT DATABASE UPGRADE PROCEDURE

In this scenario a database version upgrade is performed while Siebel continues running. There is a short pause during the switchover to the upgraded database and so the end users should be warned that this would happen.

Note, the Data Guard Broker does not support the rolling upgrade process and so disable the broker for the period of the upgrade.

The following steps are performed to complete the upgrade:

1. To begin with, a local logical standby database is established for the primary Siebel database and the database must be brought up-to-date by applying SQL. This can be done with a new database, or by temporarily converting a physical standby to logical.
2. Stop SQL Apply the logical standby:

```
SQL> ALTER DATABASE STOP LOGICAL STANDBY APPLY;
```

3. Upgrade the logical standby using the standard database upgrade procedure.
4. Restart SQL Apply and bring the standby up-to-date:

```
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY
IMMEDIATE;
```

Note, it is possible to test the newly upgraded database at this point before the switchover. See the section entitled "Testing with Logical Standby:" for details.

5. On the primary, stop the database service, disconnect all the Siebel connections, and commit to the standby role:

```
SQL> EXECUTE DBMS_SERVICE.STOP_SERVICE('SIEBEL');
SQL> EXECUTE DBMS_SERVICE.DISCONNECT_SESSION('SIEBEL');
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO LOGICAL
STANDBY;
```

At this point the Siebel end users will pause when they submit work and they must wait until the switchover has completed.

6. On the standby, wait until the switchover status becomes 'TO PRIMARY':

```
SQL> SELECT SWITCHOVER_STATUS FROM V$DATABASE;
```

Then, commit the standby to switch over to the primary role:

```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
```

When the database changes to the primary role the role change trigger starts the Siebel database service. TAF establishes connections to the new primary database and Siebel resumes working.

If there is a failure during the switchover then it is possible to resume normal operation on the pre-upgraded database by executing the following steps:

1. On the original primary:

```
SQL> ALTER DATABASE COMMIT TO SWITCHOVER TO PRIMARY;
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
```

When this completes the role change trigger will restart the service and Siebel will reconnect.

2. Begin logical apply on the original standby:

```
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY
IMMEDIATE;
```

Discussion:

In this case all committed transactions on the primary will be recovered on the standby and no data is lost.

Because of incompatibility in redo, it is not possible to convert the original primary to standby operation and so a new standby database should be established as quickly as possible.

Note: the two-phased (prepare then commit) procedure is not supported for database upgrade switchovers.

TESTING PROCEDURES

A standby database can be tested or used for testing while the primary database is in live operation and Oracle Flashback Database can be used to quickly restore the database to standby operation afterwards.

This facility can be used to test new application functionality before a production rollout. It can also be used during a transparent database upgrade to test the new database version prior to the switchover.

This process described below applies to local and remote physical standby databases for Oracle Database 10g.

Testing with Physical Standby Database in Oracle Database 10g

In this scenario we assume the following pre-requisites:

- A physical standby database is in place with flashback database enabled.
- Siebel is up and running against the primary database.

The following events unfold on the standby database:

1. Recovery is cancelled on the standby database:

```
SQL> RECOVER MANAGED STANDBY DATABASE CANCEL;
```

2. A guaranteed restore point (named "TESTING _STARTS" in this example) is created on the standby database:

```
SQL> CREATE RESTORE POINT TESTING_STARTS GUARANTEE
FLASHBACK DATABASE;
```

3. On primary database, switch the current log and then defer the archive destination for the standby:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=DEFER;
```

4. On standby, activate and open the database:

```
SQL> ALTER DATABASE ACTIVATE STANDBY DATABASE;
SQL> ALTER DATABASE SET STANDBY DATABASE TO MAXIMIZE
PERFORMANCE;
SQL> ALTER DATABASE OPEN;
```

Note, if you have a database trigger that will start the production database

service on roll change then the trigger should be disabled before executing this step.

5. The database is now open and can be used for testing. Any changes that are made to the database will be rolled back afterwards using flashback database. You can start additional databases instances and test the application as you wish. Note that the standby will be getting behind on redo application during the testing period and so make sure that you do not get too far behind.
6. When you have finished testing on the standby, shutdown all but one database instance.
7. Flashback the standby and start managed recovery:

```
SQL> STARTUP MOUNT FORCE;
SQL> FLASHBACK DATABASE TO RESTORE POINT TESTING_STARTS;
SQL> DROP RESTORE POINT TESTING_STARTS;
SQL> ALTER DATABASE CONVERT TO PHYSICAL STANDBY;
SQL> STARTUP MOUNT FORCE;
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE
USING CURRENT LOGFILE DISCONNECT;
```

8. On the primary site, enable the archive destination:

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=ENABLE;
```

Testing with a Physical Standby Database in Oracle Database 11g

Data Guard Snapshot Standby is an Oracle Database 11g capability that enables a physical standby database to be open read-write for test purposes while continually protecting primary data. A single command automates steps 1-5 above, and converts a physical standby database to snapshot standby that is open read-write. Likewise, a single command converts it back into a synchronized physical standby database once testing is complete. Data Guard Snapshot Standby is an ideal complement to Oracle Real Application Testing, another new Database 11g capability that greatly improves the quality and efficiency of test efforts by easily capturing and replaying actual production workload on a snapshot standby database.

Testing with a Logical Standby Database

In this section we document how the standby database can be tested or used for testing while the primary database is in live operation, and how Oracle Flashback Database can be used to quickly restore the database to standby operation afterwards.

This process applies to local and remote logical standby databases.

In this scenario we assume the following pre-requisites:

- A logical standby database is in place with flashback database enabled.
- Siebel is up and running against the primary database.

The following events are performed on the standby database:

1. Cancel recovery on the standby database:

```
SQL> ALTER DATABASE STOP LOGICAL STANDBY APPLY;
```

2. Create a guaranteed restore point (named "TESTING _STARTS" in this example) on the standby database:

```
SQL> CREATE RESTORE POINT TESTING_STARTS GUARANTEE
FLASHBACK DATABASE;
```

3. On the primary database, switch the current log and then defer the archive destination for the standby:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=DEFER;
```

4. Remove the guard on the standby database:

```
SQL> ALTER DATABASE GUARD NONE;
```

5. The database is now open and can be used for testing. Any changes that are made to the database will be rolled back afterwards using flashback database. You can start additional databases instances and test the application as you wish. Note that the standby will be getting behind on SQL application during the testing period and so make sure that you do not get too far behind.
6. When you have finished testing on the standby, shutdown all instances:

```
SQL> SHUTDOWN IMMEDIATE
```

7. Flashback the standby and start managed recovery:

```
SQL> STARTUP MOUNT
SQL> FLASHBACK DATABASE TO RESTORE POINT TESTING_STARTS;
SQL> DROP RESTORE POINT TESTING_STARTS;
SQL> ALTER DATABASE GUARD ALL;
SQL> ALTER DATABASE OPEN RESETLOGS;
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY
IMMEDIATE;
```

8. On the primary, switch the current log file:

```
SQL> ALTER SYSTEM ARCHIVE LOG CURRENT;
```

9. On the primary site, enable the archive destination:

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=ENABLE;
```

ORACLE RAC CONSIDERATIONS

Additional steps required for Oracle RAC 10g databases involved in Data Guard switchovers and failovers

For configurations having a logical standby with compatibility set to less than 10.2 and for all configurations having a physical standby, if the primary is an Oracle RAC database ensure that all but one instance is shut down before performing a switchover. If the standby is an Oracle RAC database, ensure that all instances except the one where the apply process is running are shut down before performing a switchover or failover. In addition, for configurations having a logical standby with compatibility set to less than 10.2, all of the redo threads that correspond to all closed instances (primary and standby) must be disabled before initiating a switchover or failover. The threads can be re-enabled and the instances restarted after the switchover operation has completed successfully. Although the instances are shut down, the role change will be automatically propagated to these instances when they are restarted.

Additional steps required for Oracle RAC 11g databases involved in Data Guard switchovers and failovers

No additional steps are required for Oracle RAC 11g databases having a logical standby with compatibility set to 10.2 or later.

For all Oracle RAC 11g configurations having a physical standby, if the primary is an Oracle RAC database ensure that all but one instance is shut down before performing a switchover. If the physical standby is an Oracle RAC database, ensure that all instances except the one where the apply process is running are shut down before performing a switchover or failover. The instances can be restarted after the switchover operation has completed successfully. Although the instances are shut down, the role change will be automatically propagated to these instances when they are restarted.

Procedure for shutting down instances and disabling threads

Additional instances can be shutdown abort:

```
SQL> SHUTDOWN ABORT
```

A thread is disabled as follows:

```
SQL> ALTER DATABASE DISABLE THREAD 2;
```

In Oracle Database 11g the multiple threads are handled automatically.

AUTOMATING SWITCHOVER AND FAILOVER PROCEDURES

The Oracle Data Guard broker is a distributed management framework that automates and centralizes the creation, maintenance, and monitoring of Data Guard configurations. The broker automates and simplifies the following operations:

- Managing an entire Data Guard configuration, including all databases, redo transport services, and log apply services, through a client connection to any database in the configuration.
- Managing the protection mode for the broker configuration.
- Invoking switchover or failover with a single command to initiate and control complex role changes across all databases in the configuration.
- Configuring failover to occur automatically upon loss of the primary database, increasing availability without manual intervention. This is known as Fast-Start Failover (FSFO). Use of this feature is optional.
- Monitoring the status of the entire configuration, capturing diagnostic information, reporting statistics such as the log apply rate and the redo generation rate, and detecting problems quickly with centralized monitoring, testing, and performance tools.

All management operations can be performed locally or remotely through the broker's easy-to-use interfaces: the Data Guard management pages in Oracle

Enterprise Manager, which is the broker's graphical user interface (GUI), and the Data Guard command-line interface called DGMGRL.

Follow the steps in Oracle Database 10g Release 2 Best Practices: Data Guard Fast-Start Failover to configure Fast-Start Failover.

TESTING OF OPERATIONAL PROCEDURES

The operational procedures described in this paper were tested under the following conditions

- The database upgrade was performed from Oracle Database 10.2.0.4 to 11.1.0.6. All other procedures were performed on Oracle Database 10.2.0.4.
- All tests were performed using single instance (non-ORACLE RAC) databases.
- All procedures were performed on Siebel 8.0.
- Tests were performed while 100 concurrent virtual users were running against the system. The test user scripts consisted of a combination of account insert, update and delete operations driven through the Siebel client.

LAB CONFIGURATION

The tests were developed and executed using HP Systems hardware and software

HP Hardware

- HP DL145 G2 running HP Load Runner v8.1 for load generation.
- HP Integrity rx2620 servers for the Siebel and Web Servers.
- HP Integrity rx4640 servers for the database tier.
- HP StorageWorks EVA4000 for database storage.

HP Software

- HP-UX 11i v2 B.11.23 for Web and Siebel Servers.
- HP-UX 11i v3 B.11.31 for Database Servers.

RESULTS

In all tests, the Siebel server processes stayed running during the database outage and reconnected and resumed work when the database transition was completed. In all cases, users were able to continue working where they left off with no need to redo work or re-login and no errors were reported to them.

REFERENCES

MAA home page:
http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm

MAA demonstrations page:
http://www.oracle.com/technology/deploy/availability/demonstrations.html

Creating an Oracle RAC Physical Standby for an Oracle RAC Primary:
http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10g_RACPrimaryRACPhysicalStandby.pdf

Creating an Oracle RAC Logical Standby for an Oracle RAC Primary:
http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10gR2_RACPrimaryRACLogicalStandby.pdf

Overview of Oracle database high availability products and features:
Oracle® Database High Availability Overview

Definitive guide to Oracle Data Guard concepts and administration:
Oracle Data Guard Concepts and Administration

Rewinding a Database with Flashback Database:
Oracle Database Backup and Recovery User's Guide

Data Guard 11g Snapshot Standby
http://download.oracle.com/docs/cd/B28359_01/server.111/b28295/cli.htm - BGBJIGJH

Oracle Real Application Testing
http://www.oracle.com/technology/products/manageability/database/index.html

MAA best practices for Data Guard Fast-Start Failover deployment:
Oracle Database 10g Release 2 Best Practices: Data Guard Fast-Start Failover

Metalink Note 460982.1 - How To Configure Server Side Transparent Application Failover

Metalink Note 453293.1 - 10g: Configuration of TAF (Transparent Application Failover) and Load Balancing

Metalink Note 249213.1 - Performance problems with Failover when TCP Network goes down (no IP address)

Metalink Note 565535.1 - Flashback Database Best Practices & Performance

Siebel Maximum Availability Architecture
November 2008
Author: Richard Exley
Conributing Authors: Ray Dutcher, Mike Smith, Joe Meeks

Oracle USA, Inc.
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