Multiple Standby Databases Best Practices:
Oracle Database 10*g* Release 2

Oracle Maximum Availability Architecture White Paper
December 2008

Maximum Availability Architecture

Oracle Best Practices For High Availability

ORACLE

Multiple Standby Databases Best Practices
Oracle Database 10*g* Release 2

Multiple Standby Databases Best Practices
Oracle Database 10*g* Release 2

INTRODUCTION

This Oracle Maximum Availability Architecture (MAA) best practices white paper discusses considerations for implementing multiple-standby database configurations and for performing role transitions in such configurations. A multiple-standby configuration can include standby databases using either Redo Apply (physical standby) or SQL Apply (logical standby), or a combination of both.

Data Guard configurations having multiple standby databases are deployed for the following purposes:

- To provide continuous protection following failover. The standby databases in a multiple-standby configuration that are not the target of the role transition (referred to as *bystander standby databases*) automatically apply redo data received from the new primary database.
- To achieve zero data loss protection while also guarding against widespread geographic disasters that extend beyond the limits of synchronous communication. For example, one standby database that receives redo data synchronously is located 200 miles away, and a second standby database that receives redo data asynchronously is located 1,500 miles away from the primary.
- To perform rolling database upgrades while maintaining disaster protection throughout the rolling upgrade process.
- To perform testing and other ad-hoc tasks while maintaining disaster-recovery protection. When desired, use some standby databases for such purposes while reserving at least one standby database to serve as the primary failover target.

MAA is a best practices blueprint for achieving high availability using Oracle technologies. The MAA testing described in this white paper used Data Guard with Oracle Database 10*g* Release 2. All testing was performed using both the Data Guard broker command-line interface (DGMGRL) and SQL*Plus statements.

Note: Cascaded standby databases, a configuration in which a standby database forwards the redo it receives from the primary database to other standby databases in a Data Guard configuration, are outside the scope of this paper. For more information about cascaded standby databases, see Oracle*MetaLink* note 409013.1

EXAMPLES OF MULTIPLE STANDBY CONFIGURATIONS

The best practices contained in this white paper supplement the information provided in the *Oracle Data Guard Concepts and Administration* [5] and the *Oracle Data Guard Broker* [6].

Multiple Physical Standby Databases

Figure 1 shows an example of a configuration with two physical standby databases.

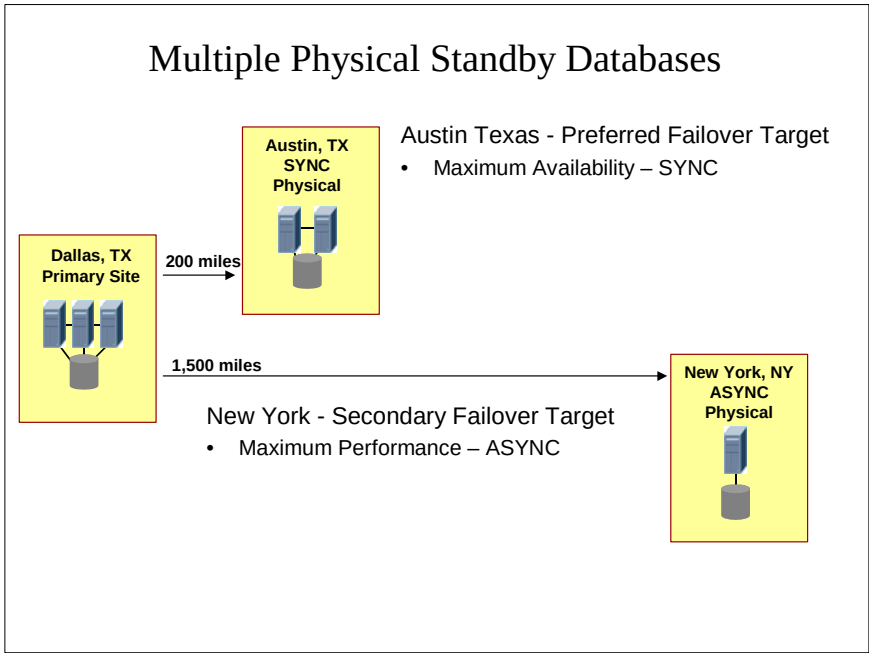

Figure 1: Configuration with Multiple Physical Standby Databases

In Figure 1, the preferred failover target database is located 200 miles away (in Austin) from the primary database in Dallas, a distance in which Data Guard can maintain synchronous redo transport without incurring unacceptable levels of overhead on the production database (due to network latency).

A secondary failover target is maintained 1,500 miles away in New York to protect against events of broader geographic scope. Asynchronous redo transport is used for the remote site, and the resulting small exposure to data loss is acceptable in

return for eliminating the impact of Wide Area Network Latency on production database performance.

A Combination of Physical and Logical Standby Databases

Figure 2 shows an example where standby databases using Redo Apply and SQL Apply are in the same Data Guard configuration.

The configuration in Figure 2 is different from Figure 1 in that it uses a logical standby database as the preferred failover target. A logical standby database can be open and used for queries while redo is being applied, a capability that does not exist for physical standby databases until Oracle Database 11*g* Release 11.1.

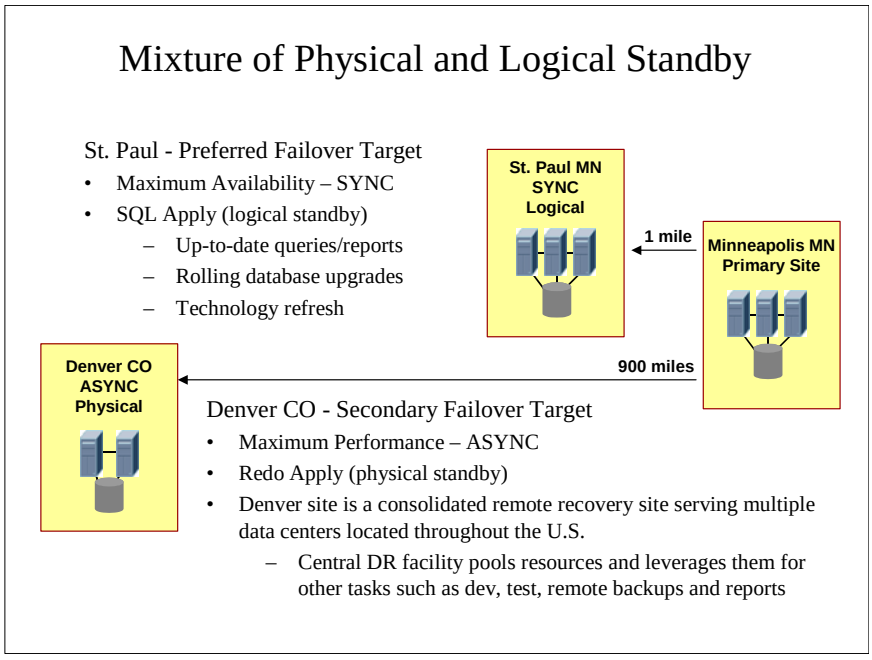

Figure 2: Configuration with Both Physical and Logical Standby Databases

BEST PRACTICES

Generally speaking, multiple standby database architectures are virtually identical to that of single standby database architectures. Therefore, when implementing multiple standby databases, you should use existing best practices for physical and logical standby databases. The MAA best practice papers for creating standby databases are available at Oracle Maximum Availability Architecture (MAA) [1]. *Oracle Data Guard Concepts and Administration* [5] and the *Oracle Data Guard Broker* [6] documentation also provide valuable information.

The primary differences between a single-standby and a multiple-standby configuration involve behavior and management during role transitions (switchover and failover operations). Depending on the configuration and the apply progress of the bystander standby databases at the time of the role transition, additional considerations and steps may be necessary. This paper concentrates on the actions required during role transitions.

Oracle recommends using the Data Guard broker to manage your configuration and perform role transitions. The best practices in the main body of this paper are presented from the perspective of using the broker. Best practices when using SQL*Plus can be found in Appendix C.

This section contains the following topics:

Configuration and Maintenance Best Practices

- Use Flashback Database to quickly reinstate the old primary as the new standby after a failover instead of re-creating the entire standby database from backups or from the new primary database. Set the `FLASHBACK_RETENTION_PERIOD` initialization parameter to the same value on all databases in the configuration. If Flashback Database is being used for the sole purpose of facilitating reinstatement following a failover, a `FLASHBACK_RETENTION_PERIOD` of 60 minutes is the minimum recommended value.
- Enable supplemental logging in configurations containing logical standby databases. When creating a configuration with both physical and logical standby databases, issue the `ALTER_DATABASE ADD SUPPLEMENTAL LOG DATA` statement to enable supplemental logging in the following situations:
  - When adding a logical standby database for the first time to an existing configuration, you must enable supplemental logging on all existing physical standby databases in the configuration.
  - When adding a new physical standby database to an existing configuration that contains a logical standby database, you must enable supplemental logging on the new physical standby database.

  Supplemental logging is enabled on the primary database when the logical standby dictionary is built. Enabling supplemental logging is a control file change and therefore the change is not propagated to each physical standby database. Supplemental logging is enabled automatically on a logical standby database when it is first converted from a physical standby database to a logical standby database as part of the dictionary build process.

To enable supplemental logging, issue the following SQL*Plus statement when connected to a physical standby database:

```
SQL> alter database add supplemental log data (primary key, unique
index) columns;
```

Tip: If the logical standby database will not perform real-time queries, then consider configuring it to use an apply delay. By delaying the application of redo, you can minimize the likelihood that the logical standby will require manual reinstatement after a failover to a physical standby database.

Role Transition Best Practices

This section describes the following topics:

Note: The term "bystander standby database" is used to describe any standby database in a multiple-standby Data Guard configuration that is not the target of a role transition.

Physical/Physical Standby Environment Best Practices

This section describes performing role transitions using the Data Guard broker in configurations consisting of only physical standby databases.

Switchover

No extra steps are necessary when performing a switchover. All bystander physical standby databases automatically apply redo data received from the new primary database.

Failover

As part of performing a failover operation, the broker determines whether bystander standby databases have applied more redo data than the failover target.

- Bystander standby databases that have applied more redo data than the failover target must be reinstated using the DGMGRL CLI `REINSTATE DATABASE` command. To see which databases require reinstatement after failover has completed, use the DGMGRL CLI `SHOW CONFIGURATION` and `SHOW DATABASE` commands.
- Bystander standby databases that have not applied more redo data than the failover target will automatically apply redo data received by the new primary database once failover has completed. No additional action is required.

The broker automatically reinstates the old primary database if fast-start failover is enabled. Otherwise, you must manually reinstate the old primary database using the DGMGRL CLI `REINSTATE DATABASE` command.

See Also:

- See Appendix A.1, "Physical/Physical Failover with Physical Bystander Ahead" for a detailed example without fast-start failover.
- Appendix A.2, " Physical/Physical Fast-Start Failover with Physical Bystander Ahead" for a detailed example using fast-start failover.

Physical/Logical Standby Environment Best Practices

This section describes performing role transitions in configurations consisting of both physical and logical standby databases.

Switchover

Switchover to a Physical Standby Database

When the switchover target is a physical standby database, all logical standby databases in the configuration automatically become logical standby databases of the new primary database.

See Appendix B.1, "Switchover to a Physical Standby in a Mixed Configuration" for a detailed example.

Switchover to a Logical Standby Database

When the switchover target is a logical standby database, the original primary database becomes a logical standby of the new primary. The bystander physical standby databases maintain their relationship as physical standbys of the original primary because they cannot apply redo data generated by the new primary database. As a result, the physical standby databases do not provide disaster recovery protection for the new primary database and are disabled by the broker.

Typically, the switchover target is a logical standby database when you need to facilitate planned maintenance or some other temporary procedure, after which you will switch back and return all databases to their original roles.

To prevent the physical standby databases from lagging too far behind the new logical standby database, you may manually configure the logical standby to ship its redo to the physical standbys. See Appendix B.3, "Switchover to a Logical Standby in a Mixed Configuration with Manual Configuration for a Bystander Physical Standby" for a detailed example.

After planned maintenance is finished, switch back to the original primary database and re-enable the bystander physical standby databases so that they may resume

disaster recovery protection for the original primary database. See Appendix B.2, "Switchover to a Logical Standby in a Mixed Configuration" for a detailed example.

Failover

Failover to a Physical Standby Database

After failing over to a physical standby database, you may reinstate the old primary database by using the DGMGRL CLI `REINSTATE DATABASE` command. The broker automatically reinstates the old primary database if fast-start failover is enabled. The broker's treatment of the bystander logical standby database depends upon how much redo data the logical standby has applied.

- If the bystander logical standby database has not applied more redo data than the physical standby database had applied at the time of the failover, the logical standby will automatically apply redo data received from the new primary database.

  See Appendix B.4, "Failover to Physical Standby with Logical Bystander Behind in a Mixed Configuration" for a detailed example.

- If the bystander logical standby database has applied more redo data than the physical standby database at the time of failover, then the broker disables the logical standby database.

  You cannot use the broker's `REINSTATE DATABASE` command to enable the bystander logical standby database. Instead, you must manually reinstate the bystander logical standby with the following steps:

  1. Follow the steps documented in the *Oracle Data Guard Concepts and Administration* [5] and the *Oracle Data Guard Broker* [6], Section 12.5.2 "Flashback a Logical Standby Database After Flashing Back the Primary."
  2. After reinstating the logical standby database, enable the database using the `ENABLE DATABASE` command.

     See Appendix B-5: Failover to a Physical Standby with Logical Bystander Ahead in a Mixed Configuration, for a detailed example.

Failover to a Logical Standby Database

After failing over to a logical standby database, you may reinstate the old primary database by using the DGMGRL CLI `REINSTATE DATABASE` command. The broker automatically reinstates the old primary database if fast-start failover is enabled. The broker always disables all bystander standby databases after a failover to a logical standby and you must reinstate the physical standby databases manually before they can be reenabled.

> **Note:** The new primary database will have disaster recovery protection only when the old primary has been reinstated and is resynchronized.

You can manually reinstate the bystander physical standby databases as standbys of the new primary database. However, the bystander physical standbys do not provide disaster recovery protection for the new primary database. The physical standby databases will provide disaster recovery protection only after a switching back to the original primary database.

Follow these steps to restore the configuration to its original state:

1. Switch back to the original primary database.
2. Manually reinstate the bystander physical standby databases using SQL*Plus commands described in *Oracle Data Guard Concepts and Administration* [5], in Section 12.5.1 "Flashing Back a Physical Standby Database to a Specific Point-in-Time".
3. Issue the DGMGRL CLI `ENABLE DATABASE` command for the physical standby databases to return the bystander physical standbys to the configuration. This will resynchronize the bystander physical standbys with the original primary database.

   See Appendix B.6, "Failover to a Logical Standby in a Mixed Configuration" for a detailed example.

CONCLUSION

Configurations that include multiple standby databases can provide even higher levels of availability, data protection, and operational flexibility. By virtue of Data Guard's design, deploying multiple standby databases can have minimal additional configuration and management requirements beyond those of a single standby database. Role transitions with multiple standby databases are handled efficiently and without disruption to other standby databases in the same Data Guard configuration properly prepare for them, as described in this paper.

APPENDIX A – DETAILED BROKER ROLE TRANSITION EXAMPLES FOR A PHYSICAL/PHYSICAL CONFIGURATION

A.1 Physical/Physical Failover with Physical Bystander Ahead

The following example uses the sample configuration from Figure 1 to illustrate how to perform a failover to a physical standby database in a configuration that contains only physical standby databases. The example shows the status and steps after a failover from Dallas, the original primary database, to Austin, a physical standby database. At the time of the failover, the physical standby database in New York had applied more redo than the physical standby database in Austin. After the failover, the example shows how to manually reinstate the physical standby database in New York to remove the extra redo that had been applied.

1. Show the status of the overall configuration and the status of New York after the failover:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    dallas   - Physical standby database (disabled)
    austin - Primary database
    newyork  - Physical standby database (disabled)

Current status for "DRMulti":
SUCCESS

DGMGRL> show database newyork

Database
  Name:            newyork
  Role:            PHYSICAL STANDBY
  Enabled:         NO
  Intended State:  ONLINE
  Instance(s):
    newyork

Current status for "newyork":
Error: ORA-16661: the standby database needs to be
reinstated
```

2. Reinstate New York:

```
DGMGRL> reinstate database newyork;
Reinstating database "newyork", please wait...
Reinstatement of database "newyork" succeeded

DGMGRL> show configuration

Configuration
```

```
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    dallas   - Physical standby database (disabled)
    austin - Primary database
    newyork  - Physical standby database

Current status for "DRMulti":
SUCCESS
```

A.2 Physical/Physical Fast-Start Failover with Physical Bystander Ahead

The following example uses the sample configuration in Figure 1 to illustrate how to perform a fast-start failover to a physical standby database in a configuration that contains only physical standby databases. At the time of the failover, New York had applied more redo than the fast-start failover target, Austin, and the broker disables both Dallas and New York. After the failover, New York must be reinstated to remove the extra redo data that had been applied.

1. Show the configuration status before the failover:

```
DGMGRL> show configuration verbose

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: ENABLED
  Databases:
    dallas - Primary database
    austin - Physical standby database
         - Fast-Start Failover target
    newyork - Physical standby database

Fast-Start Failover
  Threshold: 30 seconds
  Observer:  halinux06
```

2. Show the configuration status after fast-start failover has occurred. Notice the status of both the original primary database and the bystander physical standby database:

```
DGMGRL> show configuration verbose

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: ENABLED
  Databases:
    dallas - Physical standby database (disabled)
          - Fast-Start Failover target
    austin - Primary database
    newyork - Physical standby database (disabled)

Fast-Start Failover
```

```
  Threshold:           30 seconds
  Observer:            halinux06
  Shutdown Primary:    TRUE

Current status for "DRMulti":
Warning: ORA-16608: one or more databases have warnings


DGMGRL> show database dallas

Database
  Name:            dallas
  Role:            PHYSICAL STANDBY
  Enabled:         NO
  Intended State:  ONLINE
  Instance(s):
    dallas

Current status for "dallas":
Error: ORA-16661: the standby database needs to be
reinstated

DGMGRL> show database newyork

Database
  Name:            newyork
  Role:            PHYSICAL STANDBY
  Enabled:         NO
  Intended State:  ONLINE
  Instance(s):
    newyork

Current status for "newyork":
Error: ORA-16661: the standby database needs to be
reinstated
```

3. Reinstate the bystander standby database:

```
DGMGRL> reinstate database newyork;
Reinstating database "newyork", please wait...
Reinstatement of database "newyork" succeeded
```

APPENDIX B – DETAILED BROKER ROLE TRANSITION EXAMPLES FOR A PHYSICAL/LOGICAL CONFIGURATION

B.1 Switchover to a Physical Standby in a Mixed Configuration

The following example used the sample configuration in Figure 2 to illustrate how to perform a switchover to a physical standby database in a configuration that contains both physical and logical standby databases. Immediately after the switchover occurs to the physical standby database, Denver, the bystander logical standby database, St. Paul, becomes a logical standby database to Denver (now running in the primary database role) and continues to apply redo.

1. Show the configuration prior to the switchover:

```
DGMGRL> show configuration;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Primary database
    denver – Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

2. Perform the switchover to Denver:

```
DGMGRL> switchover to denver;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "denver"
```

3. Show the configuration immediately after the switchover occurred. Notice that all of the databases are enabled:

```
DGMGRL> show configuration;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Physical standby database
    denver - Primary database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

B.2 Switchover to a Logical Standby in a Mixed Configuration

The following example uses the sample configuration in Figure 2 to illustrate how to perform a switchover to a logical standby database in a configuration that

contains both physical and logical standby databases. In this example, the target of the switchover is the logical standby database, St. Paul. The broker disables the bystander physical standby database, Denver. After switching back to the original configuration in which Minneapolis becomes the primary database again, you can enable the broker to manage the Denver database.

1. Show the configuration prior to the switchover:

```
DGMGRL> show configuration verbose;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

2. Perform the switchover to St. Paul:

```
DGMGRL> switchover to stpaul;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "stpaul"
```

3. After the switchover, Denver is disabled and no longer receives or applies redo, as shown by issuing the following command:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Logical standby database
    denver - Physical standby database (disabled)
    stpaul  - Primary database

Current status for "DRMulti":
SUCCESS

DGMGRL> show database denver;

Database
  Name:            denver
  Role:            PHYSICAL STANDBY
  Enabled:         NO
  Intended State:  ONLINE
  Instance(s):
    denver

Current status for "denver":
Error: ORA-16795: database resource guard detects that
database re-creation is required
```

4. Perform another switchover to return Minneapolis to its original role as the primary database in the configuration. Notice that Denver remains disabled:

```
DGMGRL> switchover to minneapolis;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "minneapolis"

DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database (disabled)
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

5. Enable Denver to have it managed by the broker again:

```
DGMGRL> enable database denver;
Enabled.

DGMGRL> show configuration verbose;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

B.3 Switchover to a Logical Standby in a Mixed Configuration with Manual Configuration for a Bystander Physical Standby

The following example uses the sample configuration in Figure 2 to illustrate how to perform a switchover to a logical standby database in a configuration that contains both physical and logical standby databases. In this example, the broker disables the physical standby database, Denver, since it cannot apply redo generated by the new primary database. As a result, it cannot provide disaster recovery protection for the new primary database St. Paul.

To prevent Denver from lagging too far behind the new logical standby database, manually configure redo transport between Minneapolis and Denver. Later, perform the following tasks to switch back to the original configuration in which Minneapolis becomes the primary database again:

- Disable redo transport between Minneapolis and Denver
- Perform the switchover to Minneapolis
- Enable broker management of database Denver

1. Show the configuration prior to the switchover:

```
DGMGRL> show configuration verbose;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

2. Perform the switchover to St. Paul:

```
DGMGRL> switchover to stpaul;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "stpaul"
```

3. After the switchover, Denver is disabled and it no longer receives or applies redo, as shown by the following command:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Logical standby database
    denver - Physical standby database (disabled)
    stpaul  - Primary database

Current status for "DRMulti":
SUCCESS

DGMGRL> show database denver;

Database
  Name:            denver
  Role:            PHYSICAL STANDBY
  Enabled:         NO
  Intended State:  ONLINE
  Instance(s):
    denver

Current status for "denver":
Error: ORA-16795: database resource guard detects that
database re-creation is required
```

4. Leave Denver disabled in the broker configuration.

5. Using SQL*Plus, configure one of the LOG_ARCHIVE_DEST_*n* database initialization parameters on Minneapolis to send redo data to Denver:

```
SQL> alter system set log_archive_dest_3='service=denver
LGWR SYNC AFFIRM
valid_for=(online_logfiles,standby_role)
db_unique_name=denver'
```

6. Display the configuration:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Logical standby database
    denver - Physical standby database (disabled)
    stpaul  - Primary database

Current status for "DRMulti":
SUCCESS
```

7. At a later time, to start the switchover to return Minneapolis to its original role as the primary database in the configuration, in SQL*Plus, clear the LOG_ARCHIVE_DEST_*n* database initialization parameter previously set on Minneapolis:

```
SQL> alter system set log_archive_dest_3='';

System altered.
```

8. Perform the switchover to Minneapolis:

```
DGMGRL> switchover to minneapolis;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "minneapolis"

DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database (disabled)
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

9. Enable Denver:

```
DGMGRL> enable database denver;
Enabled.
```

```
DGMGRL> show configuration verbose;

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

B.4 Failover to Physical Standby with Logical Bystander Behind in a Mixed Configuration

The following example uses the sample configuration in Figure 2 to illustrate how to perform a failover to a physical standby database in a configuration that contains both physical and logical standby databases. At the time of failover, the broker has determined the logical standby database, St. Paul, has not applied more redo than Denver, so St. Paul remains in an enabled, active state and serves as a logical standby database to the new primary database after the failover operation. Only the original primary database, Minneapolis, needs to be reinstated.

1. Show the configuration prior to the failover:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

2. After the loss of Minneapolis, the primary database, issue the following command to manually failover to Denver:

```
DGMGRL> failover to denver;
Performing failover NOW, please wait...
Failover succeeded, new primary is "denver"
```

3. The following command shows the configuration after the failover. Notice that St. Paul is not disabled:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
```

```
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Physical standby database (disabled)
    denver - Primary database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

4. Reinstate the original primary Minneapolis:

```
DGMGRL> reinstate database minneapolis;
Reinstating database "minneapolis", please wait...
Reinstatement of database "minneapolis" succeeded
```

5. Switch back to the original configuration with Minneapolis as the primary database:

```
DGMGRL> switchover to minneapolis;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "minneapolis"
```

B.5 Failover to Physical Standby with Logical Bystander Ahead in a Mixed Configuration

The following example uses the sample configuration in Figure 2 to illustrate how to perform a failover to a physical standby database in a configuration that contains both physical and logical standby databases. In this example, the broker has determined that the bystander logical standby, St. Paul, has applied more redo at the time of failover than the target for the failover, physical standby database Denver. Therefore, the broker disables St. Paul. St. Paul must be manually reinstated before it can be reenabled.

1. Perform a manual failover to physical standby database Denver:

```
DGMGRL> failover to denver
Performing failover NOW, please wait...
Failover succeeded, new primary is "denver"
```

2. Show the configuration after the failover. Notice that St. Paul has been disabled:

```
DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Physical standby database (disabled)
    denver - Primary database
    stpaul  - Logical standby database (disabled)

Current status for "DRMulti":
SUCCESS
```

3. The following command shows that St. Paul cannot be reinstated using the `REINSTATE DATABASE` command:

```
DGMGRL> reinstate database stpaul;
Reinstating database "stpaul", please wait...
Error: ORA-16795: database resource guard detects that
database re-creation is required

Failed.
Reinstatement of database "stpaul" failed
```

4. Manually reinstate St. Paul using Flashback Database as described in *Oracle Data Guard Concepts and Administration* [5], in Section 12.4.3 "Flashing Back a Logical Standby Database to a Specific Applied SCN." After reinstatement, enable St. Paul:

```
DGMGRL> enable database stpaul;
Enabled.

DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Physical standby database (disabled)
    denver - Primary database
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

B.6 Failover to Logical Standby in a Mixed Configuration

The following example uses the sample configuration in Figure 2 to illustrate how to perform a failover to the logical standby database, St. Paul, in a configuration that contains both physical and logical standby databases. The broker disables the bystander physical standby database, Denver. You must either re-create Denver as a standby from a backup of the new primary database St. Paul, or enable Denver after returning to the original configuration in which Minneapolis is the primary database.

1. Show the configuration:

```
DGMGRL> show configuration;

Configuration
  Name:                DMulti
  Enabled:             NO
  Protection Mode:     MaxAvailability
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Primary database
    denver - Physical standby database
```

```
    stpaul  - Logical standby database

Current status for "DMulti":
SUCCESS
```

2. Fail over to the logical standby database, St. Paul. Notice the status of both the original primary database and the bystander physical standby:

```
DGMGRL> failover to stpaul;
Performing failover NOW, please wait...
Failover succeeded, new primary is "stpaul"

DGMGRL> show configuration

Configuration
  Name:                DMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Logical standby database (disabled)
    denver - Physical standby database (disabled)
    stpaul  - Primary database

Current status for "DMulti":
SUCCESS
```

3. As shown below, you cannot use the broker to reinstate Denver, so you must reinstate it manually:

```
DGMGRL> reinstate database denver;
Reinstating database "denver", please wait...
Error: ORA-16795: database resource guard detects that
database re-creation is required

Failed.
Reinstatement of database "denver" failed
```

4. Reinstate Minneapolis:

```
DGMGRL> reinstate database minneapolis;
Reinstating database "minneapolis", please wait...
Reinstatement of database "minneapolis" succeeded
```

5. Switch back to the original configuration

```
DGMGRL> switchover to minneapolis;
Performing switchover NOW, please wait...
Switchover succeeded, new primary is "minneapolis"

DGMGRL> show configuration

Configuration
  Name:                DRMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis   - Primary database
```

```
    denver - Physical standby database (disabled)
    stpaul  - Logical standby database

Current status for "DRMulti":
SUCCESS
```

6. Manually reinstate Denver. Use Flashback Database to revert the bystander physical standby to a point in time prior to the failover performed in Step 2. After Denver has been flashed back, you can enable it:

```
DGMGRL> enable database denver;
Enabled.

DGMGRL> show configuration verbose

Configuration
  Name:                DMulti
  Enabled:             YES
  Protection Mode:     MaxPerformance
  Fast-Start Failover: DISABLED
  Databases:
    minneapolis - Primary database
    denver - Physical standby database
    stpaul  - Logical standby database

Current status for "DMulti":
SUCCESS
```

APPENDIX C: SQL*PLUS MANAGED CONFIGURATION BEST PRACTICES

This appendix contains best practices when using SQL*Plus statements rather than the Data Guard broker to manage your configuration. This appendix also includes detailed role transition examples in the following sections:

- Best Practices for Managing Configurations with SQL*Plus
- Best Practices for Managing Role Transitions with SQL*Plus in a Physical/Physical Configuration
- Best Practices for Managing Role Transitions with SQL*Plus in a Mixed Configuration

C.1 Best Practices for Managing Configurations with SQL*Plus

- The following database initialization parameters need special consideration when configuring for multiple standby databases. The initialization parameters represent the settings you would use for the multiple-standby database configuration in Figure 1.
  - Set `LOG_ARCHIVE_CONFIG` to list all databases to allow log shipment from any of the databases in the configuration. For example:

    ```
    SQL> alter system set log_archive_config=
        'dg_config=(dallas, austin, newyork)' scope=spfile;
    ```

  - Set `DB_FILE_NAME_CONVERT` to define all pairs of database filename conversions in anticipation of a role transition converting any of the standby databases in the configuration to the primary database role. For example:

    ```
    SQL> alter system set db_file_name_convert=
    '/austin/','/dallas/','/newyork/','/dallas/'
    scope=spfile;
    ```

  - Set `LOG_FILE_NAME_CONVERT` to define all pairs of log filename conversions in anticipation of a role transition converting any of the standby databases in the configuration to the primary database role. For example:

    ```
    SQL> alter system set log_file_name_convert=
    '/austin/','/dallas/','/newyork/','/dallas/'
    scope=spfile;
    ```

  - Set `FAL_SERVER` to define all other databases in anticipation of any of the databases in the configuration becoming the primary database. For example:

```
SQL> alter system set fal_server='austin','newyork'
scope=spfile;
```

- Set LOG_ARCHIVE_DEST_*n* so that each database will have log archive destinations defined for every other database in the configuration in anticipation of assuming the primary role:

```
SQL> alter system set log_archive_dest_1='service=austin
valid_for=(online_logfiles,primary_role) lgwr sync
affirm db_unique_name=austin' scope=spfile;
```

```
SQL> alter system set log_archive_dest_2='service=newyork
valid_for=(online_logfiles,primary_role) arch async affirm
db_unique_name=newyork' scope=spfile;
```

- Configure the LOG_ARCHIVE_DEST_*n* parameter on all databases in the configuration as they are added.

- In a configuration containing both physical standby and logical standby databases, do not define LOG_ARCHIVE_DEST_*n* parameter entries that point to bystander physical standby database locations when configuring the logical standby database. The bystander physical standby database in this configuration is a block-for-block copy of only the original primary database. It has no relevance to the logical standby database, thus the physical standby database cannot receive redo from a logical standby database that has been transitioned to the primary database role. Using the example configuration in Figure 2, the logical standby database St. Paul should only have a LOG_ARCHIVE_DEST_*n* parameter pointing to primary database Minneapolis. It should not have a LOG_ARCHIVE_DEST_*n* parameter pointing to physical standby database Denver.

- For physical standby databases, start Redo Apply with the THROUGH ALL SWITCHOVER clause so that when a switchover occurs, media recovery continues on the bystander standby databases.

  Note: Using THROUGH_ALL_SWITCHOVER simplifies most role transition operations. However you must stop Redo Apply on the standby database that is the target of the switchover before you issue the ALTER DATABASE COMMIT TO PRIMARY command. Also, in a situation where a bystander physical database is ahead of the new primary at the time of failover, the standby database will continue to apply redo generated by the new primary database until an inconsistency is found that generates an ORA-600 error. At this point, you can flash back the bystander standby database and restart Redo Apply.

C.2 Best Practices for Managing Role Transitions with SQL*Plus in a Physical/Physical Configuration

This section describes best practices for using SQL*Plus statements to perform role transitions in a configuration containing only physical standby databases.

C.2.1 Switchover Target is a Physical Standby Database

When performing a switchover, no extra steps are necessary. All bystander physical standby databases automatically transition to serving as physical standby databases for the new primary database and continue to receive and apply redo.

C.2.2 Failover Target is a Physical Standby Database

The steps for performing a failover to a physical standby database depend on the Redo Apply progress of the new primary database and any bystander physical standby databases at the time of the failover.

- If the new primary database has applied more redo than all of the bystander physical standby databases, no additional steps are required. Only the original primary database needs to be reinstated, using the steps documented in *Oracle Data Guard Concepts and Administration*[5], Section 12.4.1 "Flashing Back a Failed Primary Database into a Physical Standby Database.".
- If any bystander physical standby database has applied more redo than the new primary database, then perform the following steps to reinstate the bystander physical standby:

*SQL*Plus Physical/Physical Failover with Physical Bystander Ahead*

1. Determine `STANDBY_BECAME_PRIMARY_SCN` from the new primary.

   ```
   SQL> select STANDBY_BECAME_PRIMARY_SCN from
   v$database;
   ```

2. On the bystander physical standby, flash back to `STANDBY_BECAME_PRIMARY_SCN` from the new primary database.

   ```
   SQL> flashback database to scn
   <STANDBY_BECAME_PRIMARY_SCN>;
   ```

3. On the bystander physical standby, delete divergent archived redo logs created at the time of, or after, the failover.

   ```
   RMAN> delete archivelog from scn
   <STANDBY_BECAME_PRIMARY_SCN>;
   ```

4. On the new primary database, enable the redo transport destination for this bystander physical standby and archive the current redo log.

   ```
   SQL> alter system set
   log_archive_destination_2=enable;
   SQL> alter system archive log current;
   ```

5. After the logs have been received by the bystander physical standby, start Redo Apply on the bystander physical standby.

```
SQL> alter database recover managed standby
database using current logfile through all
switchover disconnect;
```

The bystander standby database is now reinstated.

C.3 Best Practices for Managing Role Transitions with SQL*Plus in a Mixed Configuration

This section describes best practices for using SQL*Plus statements to perform role transition in a configuration containing both physical and logical standby databases.

C.3.1 Switchover Target is a Physical Standby Database

If the target of the switchover is a physical standby database, any bystander logical standby databases in the configuration automatically become logical standby databases to the new primary database without requiring any additional steps.

C.3.2 Switchover Target is a Logical Standby Database

If the target of the switchover is a logical standby database, you must manually update the `LOG_ARCHIVE_DEST_`*n* parameter of the original primary database pointing to the bystander physical standby to continue shipment of online redo log files in the standby role to keep the bystander physical standby database up to date.

Bystander physical standby databases remain as physical standby databases of the original primary database. The bystanders continue to receive redo from the original primary database. The `LOG_ARCHIVE_DEST_`*n* parameter from the original primary to this bystander normally has a `VALID_FOR = (ONLINE_LOGFILES,PRIMARY_ROLE)` clause defined. The original primary database, now converted to a logical standby database, must continue to ship redo data to the bystander physical standbys in its new role of a standby database, so the `VALID_FOR` clause must contain `(ONLINE_LOGFILES,STANDBY_ROLE)`. Because the recommended target for a role transition is a physical standby database, you should use `VALID_FOR = (ONLINE_LOGFILES,PRIMARY_ROLE)` attribute in the `LOG_ARCHIVE_DEST_`*n* parameter after a switchover from the primary database to a physical standby, and change it to `VALID_FOR = (ONLINE_LOGFILES,STANDBY_ROLE)` after a role transition to a logical standby database.

The following show the steps necessary to perform a swithover to a logical standby database in a mixed configuration. See Figure 2 for the configuration.

*C.3.2.1 SQL*Plus Physical/Logical Switchover to Logical*

Original Switchover

1. Prepare the primary database Minneapolis for switchover to a logical standby database role:

```
SQL> select database_role, switchover_status from
v$database;
```

```
DATABASE_ROLE    SWITCHOVER_STATUS
---------------- --------------------
PRIMARY          SESSIONS ACTIVE

SQL> alter database prepare to switchover to logical
standby;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE    SWITCHOVER_STATUS
---------------- --------------------
PRIMARY          PREPARING SWITCHOVER
```

2. Prepare the logical standby St. Paul for switchover to the primary database role:

```
SQL> alter database prepare to switchover to primary;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   PREPARING SWITCHOVER
```

3. Query V$DATABASE to check the role transition progress on Minneapolis:

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           TO LOGICAL STANDBY
```

4. Commit the switchover to the logical standby database on Minneapolis:

```
SQL> alter database commit to switchover to logical
standby;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   NOT ALLOWED
```

5. Commit the switchover to the primary database role on St. Paul:

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   TO PRIMARY

SQL> alter database commit to switchover to primary;

SQL> select database_role, switchover_status from
v$database;
```

```
DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           SESSIONS ACTIVE
```

6. On Minneapolis, reset the LOG_ARCHIVE_DEST_*n* parameter pointing to the bystander physical standby database Denver:

```
SQL> show parameter log_archive_dest_2

VALUE
------------------------------
service=denver LGWR ASYNC AFFIRM
db unique name=denver valid
for=(online_logfile,primary_role)

SQL> alter system set
log_archive_dest_2='service=denver LGWR ASYNC AFFIRM
valid_for=(online_logfile,standby_role)
db_unique_name=denver';
```

7. Start SQL Apply on the new logical standby database Minneapolis:

```
SQL> alter database start logical standby apply
immediate;
```

Switch back to the original configuration

8. Start the switchover to return to the original configuration in which Minneapolis is the primary database and St. Paul is the logical standby database, starting the switchover process on St. Paul to convert St. Paul to a logical standby database:

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           SESSIONS ACTIVE

SQL> alter database prepare to switchover to logical
standby;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           PREPARING SWITCHOVER
```

9. Prepare Minneapolis for switchover to the primary database role:

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   NOT ALLOWED

SQL> alter database prepare to switchover to primary;
```

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   PREPARING SWITCHOVER
```

10. Commit the switchover to the logical standby database role on St. Paul:

```
SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           TO LOGICAL STANDBY

SQL> alter database commit to switchover to logical
standby;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   NOT ALLOWED
```

11. Reset the LOG_ARCHIVE_DEST_*n* on Minneapolis that points to Denver:

```
SQL> select database_role, switchover status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
LOGICAL STANDBY   TO PRIMARY

SQL> show parameter log_archive_dest_2

VALUE
------------------------------
service=denver LGWR ASYNC AFFIRM
db unique name=denver valid
for=(online_logfile,standby_role)

SQL> alter system set
log_archive_dest_2='service=denver LGWR SYNC AFFIRM
valid_for=(online_logfiles,primary_role)
db_unique_name=denver';

SQL> alter database commit to switchover to primary;

SQL> select database_role, switchover_status from
v$database;

DATABASE_ROLE     SWITCHOVER_STATUS
----------------  --------------------
PRIMARY           SESSIONS ACTIVE
```

12. Start SQL Apply on St. Paul:

```
SQL> alter database start logical standby apply
immediate;
```

C.3.3 Failover Target is a Physical Standby Database

- When failing over to a physical standby database, the reinstatement steps depend on how much redo has been applied to the new primary database and how much redo has been applied to any bystander logical standby databases at the time of the failover. The primary database can be reinstated in the same manner that you would use for a single standby configuration using the steps documented in *Oracle Data Guard Concepts and Administration*[5], Section 12.4.1 "Flashing Back a Failed Primary Database into a Physical Standby Database.".

  The procedure for bystander logical standby databases depends on how much redo has been applied:

  - If the bystander logical standby database has not applied more redo than the new primary at the time of failover, no extra steps are required. The bystander logical standby will start receiving redo from the new primary database and apply it.
  - If the bystander logical standby database has applied more redo than the new primary database at the time of failover, you must manually reinstate the bystander logical standby. Use the steps documented in *Oracle Data Guard Concepts and Administration* [5], in Section 12.4.3 "Flashing Back a Logical Standby Database to a Specific Applied SCN". to reinstate the bystander logical standby database.

C.3.4 Failover Target is a Logical Standby Database

- If the failover occurs to a logical standby database, the state of any physical bystander database in relation to the new primary does not matter. The bystander physical standby does not become a physical standby to the new primary database; the bystander physical standby databases provide no disaster recovery protection for the new primary database. The bystander physical standby is orphaned until the original primary database is reinstated and returned to its original role. To return to the original configuration:

  1. Use the steps documented in *Oracle Data Guard Concepts and Administration* [5], Section 12.4.2 "Flashing Back a Failed Primary Database into a Logical Standby Database" to reinstate the original primary database.
  2. Switchback to the original configuration.
  3. Use the steps documented *Oracle Data Guard Concepts and Administration* [5], in Section 12.5.1 "Flashing Back a Physical Standby Database to a Specific Point-in-Time" to reinstate the physical standby database.

  The following example shows the necessary steps to failover to a logical standby database in a mixed configuration. See Figure 2 for the environment configuration.

*C.3.4.1 SQL*Plus Failover to a Logical Standby Database in a Mixed Configuration*

1. Failover occurs to the logical standby database St. Paul.
2. Follow the steps documented in *Oracle Data Guard Concepts and Administration* [5], in Section 12.4.2 "Flashing Back a Failed Primary Database into a Logical Standby Database" to reinstate the original primary database, Minneapolis

   ```
   SQL> flashback database to scn <flashback scn>;

   SQL> ALTER DATABASE GUARD ALL;

   SQL> alter database open resetlogs;

   SQL> create public database link stpaul
     2  connect to system identified by <password>
     3  using 'stpaul';

   SQL> ALTER DATABASE START LOGICAL STANDBY APPLY
   NEW PRIMARY stpaul;
   ```

3. Switch over to Minneapolis.
4. Follow the steps documented in *Oracle Data Guard Concepts and Administration* [5], in Section 12.5.1 "Flashing Back a Physical Standby Database to a Specific Point-in-Time" to reinstate the physical standby database, Denver:

   a) Stop Redo Apply on Denver

   b) Determine `RESETLOGS_CHANGE#` from Minneapolis, and use Flashback Database to revert Denver to that SCN.

   ```
   SQL> flashback database to scn
   <resetlogs_change# -2>;
   ```

   c) Restart Redo Apply on Denver:

   ```
   SQL> recover managed standby database using
   current logfile through all switchover
   disconnect;
   ```

APPENDIX D – PROTECTION MODE IN A MULTIPLE STANDBY DATABASE CONFIGURATION

Setting the Protection Mode

In a multiple standby database configuration, the protection mode (maximum protection, maximum availability, or maximum performance) and protection level (an aggregation of the protection modes for all standby destinations) work very much the same as they do in a single standby configuration.

On the primary database, the protection mode is based on the maximum protection, maximum availability, or maximum performance value specified for the overall configuration, while the protection level is the current status.

On each standby database, the protection mode is based on the `LogXptMode` property setting (`SYNC`, `ASYNC`, or `ARCH`) for that particular standby, while the protection level is that particular standby's current status.

> **Note:** The primary database also has a `LogXptMode` property definition, but the property value is relevant only when the database becomes a standby database after a role transition.

Regardless of how many standby databases you may have in your configuration, you specify only one protection mode for the configuration. You must have at least one standby database configured to support the protection mode. However, you can have any mix of redo transport modes (`SYNC`, `ASYNC`, or `ARCH`) among the standby databases. For example, if you plan to set the overall Data Guard configuration to the `MAXIMUM AVAILABILITY` protection mode, you must set the `LogXptMode` property to `SYNC` on at least one of the standby databases.

As long as at least one appropriately defined standby destination supporting the protection mode is accessible, Data Guard can maintain the protection mode for the configuration. If all standby destinations supporting the protection mode become inaccessible, the Data Guard broker will not automatically modify any destination to change its transport mode in an attempt to maintain support of the protection mode. You must manually edit the value of the `LogXptMode` property if it needs to be changed.

> **Note:** The broker prevents you from configuring an elevated protection mode when there are no standby databases whose `LogXptMode` property is set to `SYNC`.

The best practice is to define multiple destinations to support the elevated protection mode—primarily with a protection mode of `MAXIMUM PROTECTION`—because it allows the primary database to continue processing if you lose access to a `SYNC` destination. The configuration can use a second defined `SYNC` destination to support the elevated protection mode.

Configuring the Protection Mode for Optimal Performance During Role Transitions with the Broker

This section describes how the protection mode can affect performance after a role transition in a broker configuration with multiple standby databases. With the broker, the `LogXptMode` property for a database should be viewed only in reference to the current primary. Depending on which standby database is the target of a switchover, you may need to perform additional manual processing to ensure optimal configuration performance after the switchover.

In the example configuration from Figure 1 with Dallas (primary database), Austin (`SYNC` standby database) and New York (`ASYNC` standby database), the protection mode is set to `MAXIMUM AVAILABILITY`, and the Austin standby database is the destination supporting the protection mode. In this configuration, it is likely that Dallas would be defined as a `SYNC` destination. If you perform a switchover to New York, the switchover will retain the `MAXIMUM AVAILABILITY` protection mode as both Dallas and Austin can support this protection mode. Processing on New York (the new primary database) could be affected because of the distance between New York and the standby databases and because both Austin and Dallas are configured with the `LogXptMode` property set to `SYNC`.

To mitigate the effect, whenever performing a role transition to a destination that will not support the current protection mode, the best practice is to lower the protection mode for the overall configuration and verify/modify the `LogXptMode` property of each affected database as necessary prior to performing the switchover. Upon switching back to the original configuration (to return Dallas to the primary database role), you would reverse this process.

For example, to perform a switchover from Dallas to New York:

1. Edit the configuration to reset the protection mode:

   ```
   DGMGRL> edit configuration set protection mode as
   maxperformance;
   ```

2. Edit the `LogXptMode` property for Dallas and Austin, setting it to `ASYNC`:

   ```
   DGMGRL> edit database dallas set property
   LogXptMode='ASYNC';
   DGMGRL> edit database austin set property
   LogXptMode='ASYNC';
   ```

3. Perform the switchover:

   ```
   DGMGRL> switchover to newyork;
   ```

To return to the original configuration:

1. Perform the switchover:

   ```
   DGMGRL> switchover to dallas;
   ```

2. Edit the `LogXptMode` property for Dallas and Austin, setting it to `SYNC`:

```
DGMGRL> edit database dallas set property
LogXptMode='SYNC';
DGMGRL> edit database austin set property
LogXptMode='SYNC';
```

3. Edit the configuration to reset the protection mode:

```
DGMGRL> edit configuration set protection mode as
maxavailability;
```

Managing the Protection Mode and Role Transitions with SQL*Plus

This section describes protection mode considerations during role transitions when using SQL*Plus to manage a configuration with multiple standby databases. With SQL*Plus, the redo log transport mode can be defined in each database and the protection mode will take effect when that database assumes the primary database role. Because of this, the behavior can be slightly different with SQL*Plus compared to when using the broker.

When using SQL*Plus to perform a role transition, if there are no `LGWR SYNC` destinations defined on the new primary database, the switchover will complete. However, if the configuration has an elevated protection mode (`MAXIMUM PROTECTION` or `MAXIMUM AVAILABILITY`), the protection mode will drop to `MAXIMUM PERFORMANCE` during the switchover. This differs from broker processing in that the broker will not allow the switchover to proceed if there will be no standby databases that support the elevated protection mode available after the role transition.

In the example configuration from Figure 1 with Dallas (primary database), Austin (`LGWR SYNC` standby database) and New York (`LGWR ASYNC` standby database), the protection mode has been set to `MAXIMUM AVAILABILITY` and Austin is the destination supporting the protection mode. If you need to switchover to New York, the `LOG_ARCHIVE_DEST_`*n* parameters in New York should already define Dallas and Austin as `ASYNC` destinations. Because there would be no `SYNC` destinations available (required to support `MAXIMUM AVAILABILITY`) after the switchover, Data Guard would automatically drop the protection mode to `MAXIMUM PERFORMANCE`.

Upon returning to the original configuration, Data Guard will NOT automatically raise the protection mode. You must first switchover to Dallas and then manually raise the protection mode to the desired `MAXIMUM AVAILABILITY`.

APPENDIX E – CONSIDERATIONS FOR PERFORMING ROLLING UPGRADES IN CONFIGURATIONS WITH MULTIPLE STANDBY DATABASES

If the configuration contains multiple standby databases, you must upgrade each physical standby database. Prior to upgrading the original primary database, you must execute the standard procedure for upgrading a physical standby database in place for each physical standby database in your configuration. See the Oracle Database 10*g* Release 2 Best Practices: Rolling Database Upgrades Using SQL Apply [2] for the steps and processing required to upgrade these bystanders.

REFERENCES

1. Oracle Maximum Availability Architecture
   http://www.oracle.com/technology/deploy/availability/htdocs/maa.htm
2. SQL Apply Rolling Upgrade Best Practices: Oracle Database 10*g* Release 2
   http://www.oracle.com/technology/deploy/availability/pdf/MAA_WP_10gR2_RollingUpgradeBestPractices.pdf
3. *Oracle Database High Availability Best Practices 10g Release 2* (Part B25159)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b25159
4. *Oracle Database High Availability Best Practices 11g Release 1* (Part B28282)
   http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28282
5. *Oracle Data Guard Concepts and Administration* (Part B14239)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14239
6. *Oracle Data Guard Broker 10g Release 2 (10.2)* (Part B14230)
   http://otn.oracle.com/pls/db102/db102.to_toc?partno=b14230
7. *Oracle Data Guard*
   http://www.oracle.com/technology/deploy/availability/htdocs/DataGuardOverview.html

Multiple Standby Databases Best Practices: Oracle Database 10*g* Release 2
December 2008
Author: Frank Kobylanski
Contributing Authors: Andrew Babb, Nitin Karkhanis, Joseph Meeks, Vivan Schupmann, Michael Smith, Lawrence To

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