Data Guard in Oracle Database 10*g* SQL Apply Best Practices

An Oracle White Paper
October 2005

ORACLE

Data Guard in Oracle Database 10*g*
SQL Apply Best Practices

Data Guard in Oracle Database 10*g*
SQL Apply Best Practices

EXECUTIVE OVERVIEW

With Data Guard SQL Apply in Oracle Database 10*g*, Oracle is addressing the requirements of the business community for an online disaster recovery solution that also provides a means to offload reporting and decision support operations from the primary database.

Data Guard SQL Apply has the following benefits:

- Allows the standby database to be open for normal operations.
- Performs all of its operations on the standby node, and requires minimal additional processing on the primary nodes.

INTRODUCTION

This white paper describes the characteristics of the Logical Standby engine, and determines the configuration best practices for Oracle Data Guard Logical Standby database, thereby optimizing the application of the changes to the standby database.

Updates to this white paper may be found on the Oracle Technology Network Website at http://otn.oracle.com/deploy/availability/htdocs/maa.htm.

In addition, please reference this papers complementary MetaLink Note 312434.1 "*Oracle 10g Data Guard SQL Apply Troubleshooting*". Taken together, these references provide practical advice on best practices for configuring, managing, and tuning an Oracle 10*g* Data Guard Logical Standby database.

Data Guard in Oracle Database 10*g* database introduces a number of SQL Apply enhancements over that of Oracle9*i*R2 such as:

- Standby Redo Log files
- Real Time Apply
- Support for Maximum Protection mode
- Simpler SQL Apply configuration with less tuning

- Better SQL Apply performance
- Faster SQL Apply switchover
- Identifying log files that are no longer required
- Integration with flashback database to resolve logical failures and for re-instantiation after a failover
- More support for data types
- Zero downtime instantiation
- Upgrading the Oracle Database Software Version with SQL Apply
- Improved Redo Data Transmission Security

For an extensive list of the New Features available in Oracle Database 10*g*, please refer to the New Features Guide (http://download-west.oracle.com/docs/cd/B14117_01/server.101/b10750.pdf) available on OTN.

SQL APPLY BENEFITS AND GENERAL OBSERVATIONS

These following recommendations will provide significant Logical Standby gains and will provide additional insight in tuning your database for Logical Standby apply. These best practices were derived from extensive Logical Standby testing on Oracle9*i* and Oracle 10*g* databases as part of the performance studies within the Maximum Availability Architecture (MAA) projects. For more information about MAA, please refer to http://otn.oracle.com/deploy/availability/htdocs/maa.htm.

The next few sections provides further details on each of these capabilities:

Standby Redo Log files

With Oracle Database 10*g*, Logical Standby database are now able to utilize Standby Redo Log files. With Standby Redo Log Files, the Logical Standby database may be configured for Maximum Protection mode, resulting in guaranteed no data loss.

Real Time Apply

With Oracle Database 10*g*, Logical Standby databases are now able to utilize Real Time Apply, when Standby Redo Log files are present. With Real Time Apply the Logical Standby database can be configured with no delay, resulting in near real time information being present on the standby database.

Support for Maximum Protection mode

With Oracle Database 10*g*, the Logical Standby database can now be configured as a target for Maximum Protection mode. Maximum Protection mode guarantees no data loss in the event of a failure affecting the primary database.

Simpler SQL apply configuration with less tuning

Oracle Database 10*g* simplifies the configuration of the logical standby processes when compared with Oracle9*i*R2.

Better SQL apply performance

Oracle Database 10*g* running under the recommended configuration when compared to Oracle9*i*R2 SQL Apply performs approx 25% faster. This is with the Logical Standby configured for FULL transactional consistency, and with 27 apply processes.

Faster SQL apply switchover and failover

Oracle Database 10*g* has reduced the time for a scheduled switchover to complete by introducing a prepare phase that executes before the actual outage occurs. A scheduled switchover can complete in less than 30 seconds under Oracle Database 10*g*.

Identifying log files that are no longer required

Oracle Database 10*g* has introduced a PL/SQL procedure that reports the archived log file names of all log files that have been applied and are no longer required by the SQL apply engine.

Integration with flashback database to resolve logical failures in the logical standby database

Oracle Flashback Database when configured with Oracle 10*g* Logical Standby can be used to recover failures to the standby database more efficiently by rewinding the standby database to the time before the failure occurred. For additional information on Flashback database refer to Backup and Recovery Advanced User's Guide (ref. http://download-west.oracle.com/docs/cd/B14117_01/server.101/b10734/toc.htm).

Zero Downtime Instantiation

Oracle Database 10*g* allows for the instantiation of a Logical Standby database from a RMAN database backup or traditional hot backup, without the need to interrupt the production database. For further details refer to Data Guard Concepts and Administration manual (ref. http://download-west.oracle.com/docs/cd/B14117_01/server.101/b10823/create_ls.htm#76903) on OTN.

More support for data types

Oracle Database 10*g* continues to lift the restrictions on unsupported data types and now supports Index Organized Tables without LOB's or Overflow segments, and LONG & LONG RAW and NCLOB columns. For a complete list of supported and unsupported data types, refer to Data Guard Concepts and Administration manual on OTN.

Upgrading the Oracle Database Software Version with SQL Apply

Starting with Oracle Database 10*g* Release 1 Patchset 1 (Oracle Database Release v10.1.0.3), a Logical Standby database can be used to support upgrading the Oracle Database Software Version. For additional information, please refer to Oracle10g Database readme file (ref. http://download-west.oracle.com/docs/cd/B14117_01/server.101/b10823/manage_ls.htm#1028400).

Improved Redo Data Transmission Security

Oracle Database 10*g* improves the security of redo transmitted between the source and target database with support for data encryption. For additional information, please refer to Data Guard Concepts and Administration manual on OTN.

LOGICAL STANDBY BEST PRACTICIES CONFIGURATION

Standby Redo Log files

With Oracle Database 10*g*, Logical Standby database are now able to utilize Standby Redo Log files. Without the presence of Standby Redo Log files, Logical Standby must wait for an archive log file to be transferred to the standby database and registered, before the Logical Standby process could mine the log.

Additionally, with Standby Redo Log Files, the Logical Standby database may be configured for Maximum Protection mode, resulting in guaranteed no data loss.

Create the standby redo log files on the standby database.

```
SQL> ALTER DATABASE ADD STANDBY LOGFILE THREAD
  <thread#> SIZE <size>;
```

The size of the Standby Redo Log file must be the same size as the Online Redo Log file on the primary database.

Real Time Apply

With Real Time Apply the Logical Standby database can be configured with no delay, resulting in near real time information being present on the standby database. In order to utilize Real Time Apply, Standby Redo Log files must be created.

To utilize Real Time Apply,

To configure the standby database to archive the standby redo logfiles.

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_n =
  'location=<standby dest> ARCH
  VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE)';
```

To configure the primary database to ship the redo stream in real time.

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_n =
  'service=<TNSALIAS> LGWR <transport options>
  VALID_FOR=(ONLINE_LOGFILES, PRIMARY_ROLE)
  DB_UNIQUE_NAME=<DB_UNIQUE_NAME>';
```

To start Logical Standby with Real Time Apply enabled.

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

Support for Maximum Protection mode

To utilize Maximum Protection mode and to have a logical standby database as the means for guaranteeing no data loss, the standby database must be configured with Standby Redo Log files.

To configure the primary instance to transmit the data in real time

```
SQL> ALTER SYSTEM SET LOG_ARCHIVE_DEST_n =
'service=<service_name> LGWR SYNC AFFIRM …';
```

To configure the primary database in Maximum Protection mode, enter the following command on the primary database

```
SQL> ALTER DATABASE SET STANDBY DATABASE TO MAXIMIZE
  PROTECTION;
```

For more information regarding the use of Maximum Protection mode, including the network requirements, please refer to the Primary Site and Network Configuration Best Practices paper on the MAA web site.

Simpler SQL apply configuration with less tuning

The tuning methodology for a Logical Standby database maintained by SQL apply follows standard Oracle procedures as described in the Performance Tuning Guide (http://download-west.oracle.com/docs/cd/B14117_01/server.101/b10752/toc.htm)

Before discussing SQL apply configuration and tuning it is helpful to understand how the workload to maintain a Logical Standby database is different from the original workload executed on the Primary database. Because of these differences, it cannot be assumed that a well-tuned Primary database will result in identically configured Logical Standby database being similarly well tuned. For example, applications make use of shared pool differently from how the SQL apply process

uses shared pool. A Primary database supports SQL Queries as well as DML operations against the database, and therefore the shared pool needs to support additional SQL statements. Additionally, DML statements issued against the Primary database can take multiple forms for a given table. In contrast, for a Logical Standby databases the SQL Apply processes do not issue SQL Queries against the application tables, and all DML is reduced to three statements per table, one for insert DML, one for update DML and one for delete DML. A Logical Standby database utilizes the shared pool for the "LCR-Cache" which is a region of the shared pool used exclusively by SQL Apply processes. These differences may require different shared pool configuration for the primary and standby databases in order to optimize the performance of each database.

Additionally, if it is determined that less shared pool is required to support SQL Apply than is required for the primary database, then it is possible to allocate more memory to the buffer cache of the Logical Standby database to further enhance SQL Apply performance.

With these facts in mind, the tuning of a Logical Standby database should begin the same way that a primary database should be tuned. AWR reports should be generated over relevant periods of time, and the resulting information used to tune the database, such as analyzing the "Top 5 Wait Events", Latching and System Statistics.

Database 10*g* further simplifies the configuration of the logical standby processes when compared with Oracle9*i*R2. With Oracle Database 10*g*, there are only two parameters specific to logical standby that may need to be configured

MAX_SERVERS

The SQL Apply parameter MAX_SERVERS controls the number of processes that will be created when the Logical Standby process is started. The recommended initial setting for this parameter remains unchanged from Oracle 9*i*R2 and should be set to (3 * CPU) + 3.

```
SQL> execute DBMS_LOGSTDBY.APPLY_SET('MAX_SERVERS',nn);
```

A second consideration when it comes to setting the MAX_SERVERS parameter is to examine Maximum Transaction Concurrency of the Primary database. This value is reported in the primary database AWR report under the “Undo Segment Status” section of the "Undo Segment Summary". This value is the highest number of concurrent transactions seen during the two snapshots contained in an AWR report. If the value for Maximum Transaction Concurrency is larger than the MAX_SERVERS setting recommended above, then increase the MAX_SERVERS parameter to a value of (3 * Max Tx Concurrency) / 2.

If the MAX_SERVERS parameter is increased in this fashion, then CPU utilization needs to be reviewed to ensure CPU / RUN Queues have not become excessive. The I/O subsystem must also be monitored to insure sufficient

performance needed to support the additional processes, especially if the storage is shared by multiple applications.

Additionally, as with Oracle 9*i*R2, the MAX_SERVERS parameter is dependent upon the init.ora parameter parallel_max_servers, which should be increased by or set to the value specified in the APPLY_SET procedure.

```
SQL> ALTER SYSTEM SET PARALLEL_MAX_SERVERS=nn;
```

TRANSACTION_CONSISTENCY

SQL Apply in Oracle Database 10*g* supports two levels of transactional consistency, FULL and NONE. When the ordering of different transactions is critical, then FULL consistency should be used, but when the ordering is not important, then NONE consistency maybe used.

```
SQL> execute DBMS_LOGSTDBY.APPLY_SET(
   'TRANSACTION_CONSISTENCY' ,'FULL');
```

It may be noted that irrespective of the SQL Apply TRANSACTION_CONSISTENCY setting, Oracle database always maintains consistency within a transaction – whether it is the primary database or the logical standby database. When the value of "NONE" is specified for TRANSACTION_CONSISTENCY, the commit order of the various transactions in the logical standby database will not match that of the primary database. However, following a switchover or a failover, full consistency is established in the new primary database.

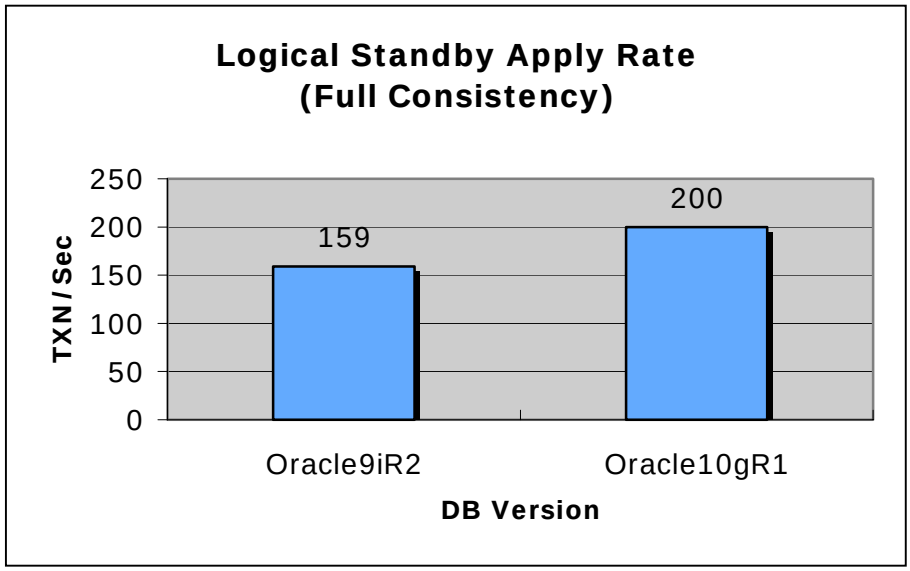
Better SQL apply performance

A workload consisting of 5 distinct transactions was run against the primary database. The two primary transaction's consisting of approx 44% of all transactions each were complex transactions. The first modifying data in 6 different tables, and with a mixture of DML operations manipulated approx 25 rows. The second transaction updated information in 3 different tables and inserted data into a 4th table.

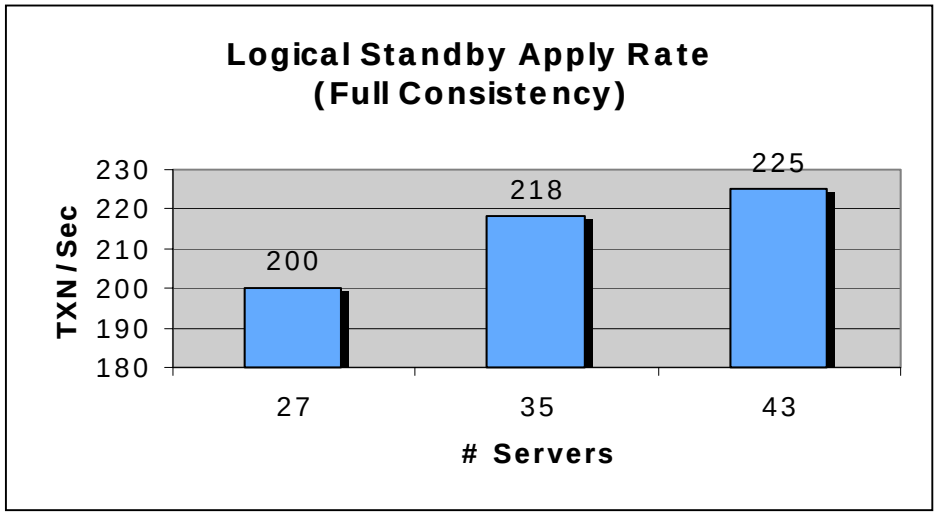
A third transaction modified approx 150 rows over 4 distinct tables, and the final 8% of transactions were split between two complex query transactions.

All transactions utilized either a primary or unique index in there access plans. No DDL transactions were performed in the transaction mix.

Oracle Database 10*g* running under the recommended configuration when compared to Oracle9*i*R2 SQL Apply performs approx 25% faster. This is with the Logical Standby configured for FULL transactional consistency, and with 27 processes.

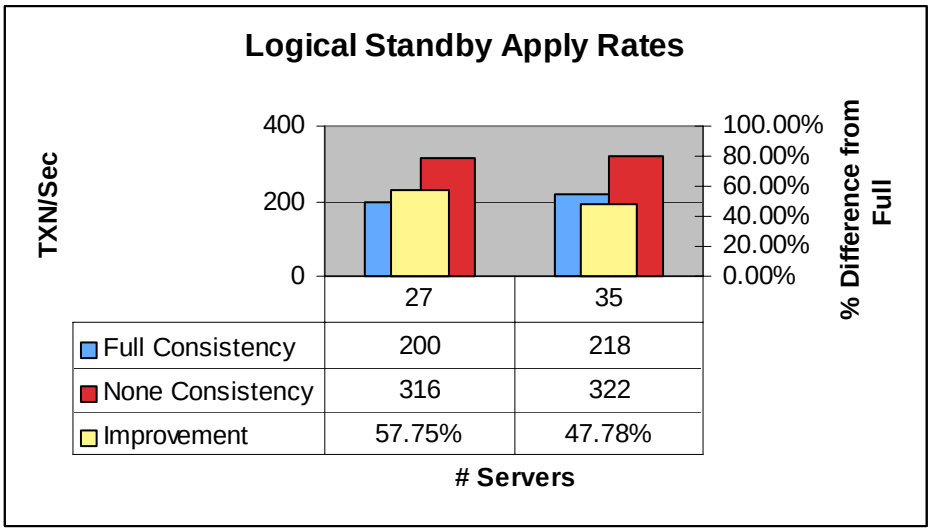


From the following chart, if we look at the scaling of the SQL Apply engine in Oracle Database 10*g* as servers are added to the configuration, we can see that there is minimal gain as additional processes are added with a maximum gain of 12% over that of the recommended configuration.



SQL Apply in Oracle Database 10*g* supports two modes of consistency, FULL and NONE. FULL consistency model should be used under normal circumstances when dependencies between transactions must be maintained. NONE consistency model should only be used when the standby database is being used for Disaster

Recovery purposes, and dependencies between unrelated transactions do not need to be maintained. The following chart shows the performance improvement that can be achieved when the logical standby is running in NONE consistency



Faster SQL apply switchover and failover

Both switchover and failover operations can be completed in less than 30 seconds.

In Oracle Database 10*g*, scheduled switchover can be optimized if

- Sessions are manually disconnected prior commencing switchover
- Real Time Apply is leveraged and minimum or zero recovery is required at the time of switchover
- Prepare phase is executed prior to the actual outage

In Oracle Database 10*g*, failover is optimized if

- Real Time Apply is leveraged and minimum or zero recovery is required at the time of switchover

Oracle Database 10*g* has reduced the time for a scheduled switchover to complete by introducing a prepare phase that executes before the actual outage occurs. Through this mechanism, prior to the point of the scheduled switchover, the primary and standby database must be *prepared* which causes the Logical Standby meta data to be built.

On the current primary database prior to the switchover

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

On the current standby database prior to the switchover

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

After both the prepare phases are complete and at the time of the switchover, disconnect all applications before proceeding.

On the current primary database

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

On the current standby database

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

Switchover and failover times may increase depending upon the size and number of the Standby Redo Logfiles which must be cleared during a switchover / failover operation. Besides, with Enterprise Manager and Data Guard broker, switchover times are higher because the prepare phase is in-line with the switchover downtime.

Switchover and failover times may also vary when the primary/and or logical standby database is a Real Application Clusters (RAC) database. For additional information on logical standby switchover steps involving RAC primary and/or standby databases, please refer to MetaLink Note 298452.1 "*Switchover of Logical Standby Database in RAC environment*".

Identifying log files that are no longer required

Oracle Database 10*g* has introduced a PL/SQL procedure that clears up the meta data in the Logical Standby database and generates a report of the archived log file names that have been applied and are no longer required by the SQL apply engine.

To purge the logical standby session of metadata that is no longer needed, enter the following PL/SQL statement:

```
SQL> EXECUTE DBMS_LOGSTDBY.PURGE_SESSION;
```

This statement also updates the DBA_LOGMNR_PURGED_LOG view that displays the archived redo log files that are no longer needed.

Query the DBA_LOGMNR_PURGED_LOG view to list the archived redo log files that can be removed:

```
SQL> SELECT * FROM DBA_LOGMNR_PURGED_LOG;
FILE_NAME
-----------------------------------
/boston/arc_dest/arc_1_40_509538672.log
/boston/arc_dest/arc_1_41_509538672.log
/boston/arc_dest/arc_1_42_509538672.log
/boston/arc_dest/arc_1_43_509538672.log
/boston/arc_dest/arc_1_44_509538672.log
/boston/arc_dest/arc_1_45_509538672.log
/boston/arc_dest/arc_1_46_509538672.log
```

/boston/arc_dest/arc_1_47_509538672.log

Use an operating system-specific command to delete the archived redo log files listed by the query.

For additional information, please refer to Data Guard Concepts and Administration.

CONCLUSION

Oracle Database 10*g* has further enhanced the features and performance of Data Guard SQL Apply making it a viable option for customers needing to implement a disaster recovery solution as well as utilizing the resources for reporting and decision support operations.

APPENDIX

Upgrading from Oracle9*i*R2 to Oracle Database 10*g* Release 1

When updating from Oracle9*i*R2 to Oracle10*g*, the primary enhancement that should be adopted is Real Time Apply. This feature allows for real time reporting against the standby database as well as reducing the time taken to switchover or failover to the standby database should the need arise.

Oracle Database 10*g* also removes the need to utilize un-documented parameters such as _eager_size as well as removes the READ_ONLY transaction consistency setting, so these parameters should be unset or changed to a more appropriate value.

To remove the _eager_size parameter

```
SQL> execute DBMS_LOGSTDBY.APPLY_UNSET('_EAGER_SIZE');
```

To change the transaction consistency parameter

```
SQL> execute DBMS_LOGSTDBY.APPLY_SET
  ('TRANSACTION_CONSISTENCY','FULL');
```

For additional information on upgrading from Oracle 9*i*R2 to Oracle 10*g*, please refer to Meta Link note# 278108.1, "*Upgrading to 10g with a Logical Standby in Place*".

Data Guard in Oracle Database 10*g* SQL Apply Best Practices
October 2005
Author: Andrew Babb
Contributing Authors: Lawrence To, Doug Utzig, Ashish Ray, Raymond Dutcher, Michael Smith

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
www.oracle.com