Oracle Database 10*g* Best Practices: Data Guard Redo Apply and Media Recovery

An Oracle White Paper
September 2005

ORACLE

Oracle Database 10*g* Best Practices: Data Guard Redo Apply and Media Recovery

Executive Overview 3
Data Guard Redo Apply and Media Recovery Best Practices 4
Tuning Media Recovery Phases 4
Best Practices for Tuning Log Read Phase 5
Best Practices for Tuning Redo Apply Phase 5
Best Practices for Tuning Checkpoint Phase 8
Troubleshooting and Advanced Tuning 9
Assess system resources 9
Assess database waits 10
Conclusion 12
Appendix A – Recovery Rate Script 13
Appendix B – Recovery tuning Steps 14
References 16

Oracle Database 10*g* Best Practices: Data Guard Redo Apply and Media Recovery

EXECUTIVE OVERVIEW

With the increasing adoption of Oracle Data Guard as a comprehensive solution for enterprise disaster recovery, optimizing media recovery for Data Guard Redo Apply is an important consideration for a Data Guard configuration, in order to keep the physical standby in that configuration as current as possible with the primary database.

Media recovery occurs when one or more datafiles or the controlfiles are restored from a previous backup or when using Data Guard Redo Apply in managed recovery. The goal of media recovery is to recover the datafiles and the rest of the database to a consistent point in time or to apply all primary database transactions that have occurred when a physical standby database is used to protect the primary site.

This paper provides best practice recommendations for configuring media recovery in Oracle Database 10*g* – both in the case of a regular backup and Data Guard Redo Apply, such that the Service Level Agreement (SLA) associated with the recovery time can be achieved [1]. This paper does not cover block media recovery, crash recovery, instance recovery, or Data Guard SQL Apply with a logical standby database.

It may be noted that with some of the new features of Oracle Database 10*g*, such as Real Time Apply and Flashback Database, Data Guard Redo Apply can provide fast switchover or failover in the event of an outage while still being prepared to revert any logical corruption. It is essential that media recovery is tuned by following the best practices outlined in this paper so that it complements these new features in the most optimal manner.

Based on test results and customer experiences, following are examples of results obtained after adopting the best practices outlined in this paper:

- In Oracle Database 10*g*, the Data Guard Redo Apply instance achieved an apply rate of 14 MB/sec for a large OLTP application.

[1] This SLA metric is commonly referred to as the Recovery Time Objective (RTO).

- Data Guard Redo Apply doubled redo apply rate in Oracle Database 10*g* compared to Oracle9*i*. In an environment with 8 CPUs (@400 Mhz) and 8 GB RAM, redo apply rate improved from 6 MB/sec in Oracle9*i* to 14 MB/sec after upgrading to Oracle Database 10g.

DATA GUARD REDO APPLY AND MEDIA RECOVERY BEST PRACTICES

The best practices outlined in this paper have been derived after extensive media recovery testing on Oracle Database 10*g* as part of performance studies within the Maximum Availability Architecture (MAA) project[2]. For more information on MAA, please refer to [1]. For more information on Oracle 10*g* High Availability Practices or Data Guard, refer to [2], [3], [4] and [5]. Besides, some of these best practices were derived after extensive joint studies with real customer databases.

Tuning Media Recovery Phases

Media recovery consists of three distinct phases. Each phase must be assessed and tuned if the recovery rate is not sufficient.

1. **Log Read Phase** involves the reading of redo from the standby redo logs or archived redo logs by the recovery coordinator or Managed Recovery Process (MRP).
2. **Redo Apply Phase** involves the reading of data blocks into the buffer cache and the application of redo, by parallel recovery slave processes. The recovery coordinator (or MRP) ships redo to the recovery slaves using the parallel query (PQ) inter-process communication framework.
3. **Checkpoint Phase** involves the flushing to disk of modified data blocks and the update of data file headers to record checkpoint completion.

Real Application Clusters (RAC) provides additional fault tolerance to an existing Data Guard Redo Apply instance but does not help speed up recovery. For a RAC standby, in Data Guard Redo Apply, media recovery still runs on one instance, called the *apply instance*. However, for a RAC standby, Data Guard Broker makes it possible to achieve seamless high availability in the event of failures of one or more instances in a RAC standby. Redo transport and redo apply can be redirected to a surviving standby instance without any intervention from the user. For further details, refer to [6].

The following sections outline the best practices relevant to each phase.

[2] These general best practices should apply to most customer environments. However, these results are not indicative of what you may experience. Testing with serial recovery and different degrees of parallelism is imperative.

Best Practices for Tuning Log Read Phase

Maximize I/O rates on standby redo logs (SRL) and archived redo logs

Measure read I/O rates on the SRL and archived redo log directories. Keep in mind that the concurrent writing of shipped redo on a standby may reduce the redo read rate due to I/O saturation. The overall recovery rate will always be bounded by the rate at which redo can be read; so ensure that the redo read rate surpasses your required recovery rate.

The following UNIX example shows how to measure the maximum redo read rate for recovery. Oracle uses a 4 MB read buffer for redo log reads.[3]

```
% /bin/time dd if=/redo_logs/t_log8.f of=/dev/null bs=4096k

   50+1 records in
   50+1 records out

   real 6.4
   user 0.0
   sys 0.1
```

Estimated Read Rate (200 MB log file) = (50 * 4 MB) / 6.4s = 31.25 MB/sec

Best Practices for Tuning Redo Apply Phase

Assess Recovery Rate

Use the following queries to get several snapshots while a redo log is being applied to obtain the current recovery rate:

i) Determine Log Block Size (`lebsz`) since it is different for each operating system. This query only needs to be executed once.

```
select lebsz LOG_BLOCK_SIZE from x$kccle where rownum=1;
```

ii) Derive recovery blocks applied for at least 2 snapshots:

(a) Media Recovery Cases (e.g. `recover [standby] database`)

```
select TYPE, ITEM, SOFAR, TO_CHAR(SYSDATE, 'DD-MON-YYYY
HH:MI:SS') TIME from v$RECOVERY_PROGRESS where ITEM='Redo
Blocks' and TOTAL=0;
```

(b) Managed Recovery Cases (e.g. `recover managed standby database…`)

```
select PROCESS, SEQUENCE#, THREAD#, BLOCK#, BLOCKS,
TO_CHAR(SYSDATE, 'DD-MON-YYYY HH:MI:SS') TIME from
V$MANAGED_STANDBY where PROCESS='MRP0';
```

[3] If you repeat this simple test, use a different SRL or archive log since the data may be cached making the results artificially high and incorrect.

iii) To determine the recovery rate (MB/sec) for this archive, use one of these formulas with the information derived above:

(a) Media Recovery Case:

((SOFAR_END – SOFAR_BEG) * LOG_BLOCK_SIZE) /

((TIME_END – TIME_BEG) * 1024 * 1024)

(b) Managed Recovery Case:

((BLOCK#_END – BLOCK#_BEG) * LOG_BLOCK_SIZE)) /

((TIME_END – TIME_BEG) * 1024 * 1024)

To assess if more tuning is required, get the maximum and average redo generation rates at the primary database from the primary database's `v$sysstat`'s statistic "redo size"[4] and use the redo apply rate quick assessment chart below.

Table 1: Redo Apply Rate Quick Assessment

Redo Generation Rate vs Redo Apply Rate	Recommendation
2 * Max Primary Database Redo Generation Rate < Redo Apply Rate	Excellent - No Tuning Required
Max Primary Database Redo Generation Rate < Redo Apply Rate < 2 * Max Primary Redo Generation Rate	Good - Tuning is Optional
Avg Primary Redo Generation Rate < Redo Apply Rate	OK - Need Tuning
Avg Primary Redo Generation Rate > Redo Apply Rate	Bad - Need Tuning. Call Oracle Technical Support if all tuning steps have been followed and the redo apply rate is still too slow. Refer to Appendix B.

You may notice that the recovery rate may vary depending on the primary's transaction activity. Typically recovery rate is much higher when the number of distinct blocks being changed is small or during batch processing. In most applications, a predictable pattern surfaces after monitoring for several days.

---

[4] You can derive the Redo Generation Rate manually by querying `v$sysstat` and getting 2 snapshots. The formula of Redo Generation Rate:

(EndTime.redo size – StartTime.redo size)/time interval

You can leverage the following query to get a snapshot of redo size:

```
select name, value, to_char(sysdate, 'dd-mon-yyyy HH:MI:SS') from
v$sysstat where name = 'redo size';
```

Please refer to the recovery rate script in Appendix A.

Use defaults for DB_BLOCK_CHECKING and DB_BLOCK_CHECKSUM

The default settings are `DB_BLOCK_CHECKING = FALSE` and `DB_BLOCK_CHECKSUM = TRUE`. Setting `DB_BLOCK_CHECKING` to `TRUE` can potentially **halve** the recovery rate. Block checking is always recommended on the primary database and may still be enabled on the standby if the recovery rate meets expectations. Block checksum should **always** be enabled for both primary and standby databases and will catch most block corruptions while incurring negligible overhead. [5]

Set recovery parallelism to the number of CPUs

Parallel recovery is enabled by default for media and crash recovery with the default and optimal degree of parallelism set to the number of CPUs available. The same default settings are used by managed recovery or Data Guard Redo Apply beginning with 10.1.0.5, and for Oracle Database 10g Release 2 beginning with 10.2.0.1. Prior to these releases it is necessary to explicitly set the PARALLEL attribute in the MANAGED RECOVERY clause. For example,

```
RECOVER MANAGED STANDBY DATABASE PARALLEL;
```

If you want to experiment with an higher degree of parallelism, you can explicitly dictate the degree of parallelism.

```
RECOVER MANAGED STANDBY DATABASE PARALLEL <#>;
```

Set PARALLEL_EXECUTION_MESSAGE_SIZE (PEMS) = 8192

Increasing the PEMS parameter to 8192 may improve recovery by as much as 20%, over the default PEMS setting of 2152. The message size parameter is used by parallel query operations so there must be sufficient shared pool to support this increase. On a 64-CPU box with a 32-bit address space, an increase in message size to 8K may cause parallel query operations to fail, because of a blowout in addressable memory. Most of the recovery performance gains can be realized by increasing PEMS to 4K (4096).

---

[5] Redo apply always does simple fundamental checks such as the header is formatted correctly and comparing the version of header block with the tail block for accuracy. Setting `DB_BLOCK_CHECKSUM` compares current block checksum with the calculated value. Checksums catch most data block inconsistencies. Additionally `DB_BLOCK_CHECKING` validates more internal data block data structures such as Interested Transaction Lists (ITLs), free space and used space in the block.

Set DB_CACHE_SIZE >= Primary's settings, DB_KEEP_CACHE_SIZE=0, and DB_RECYCLE_CACHE_SIZE=0

Having a large database cache size can improve media recovery performance significantly. Since media recovery does not require `DB_KEEP_CACHE_SIZE` and `DB_RECYCLE_CACHE_SIZE` or require a large `SHARED_POOL_SIZE`, the memory can be reallocated to the `DB_CACHE_SIZE`.

The only complication is resetting it to the primary database setting prior to changing to a primary role. If these parameters are different, you may require 2 initialization parameter files (init.oras or spfiles) for standby and primary roles.

Best Practices for Tuning Checkpoint Phase

Increase primary and standby log group size

Increase the primary database's online redo log and standby database's standby redo logs sizes to a default of 1 GB. Oracle does a full checkpoint and updates all the file headers (in an optimized manner) at each log file boundary during media recovery. To reduce the frequency of a full database checkpoint and updating all the file headers, increase the log group size so that a log switch is occurring at a minimum of 15 minutes interval[6]. If Real Time Apply is being used and redo is being sent synchronously or asynchronously via LGWR process, then there's no additional data loss risk with this change. If archiver is sending the redo or the primary database is converting to ARCH mode due to heavy load, then you have to balance between faster recovery rates and higher data loss risk.

Tune I/O

DBWR needs to write out modified blocks from the buffer cache to the data files. Always use native asynchronous I/O by setting DISK_ASYNCH_IO=TRUE (default). In the rare case that asynchronous I/O is not available, use DBWR_IO_SLAVES to improve the effective data block write rate with synchronous I/O.

Ensure that you have sufficient I/O bandwidth and the I/O response time is "reasonable" for your system either by doing some base I/O tests, comparing the I/O statistics with primary, or looking at some historical I/O metrics. Be aware that I/O response time may vary when many applications share the same storage infrastructure such as with a Storage Area Network (SAN) or Network Attached Storage (NAS).

---

[6] To ensure that that the primary database's crash recovery time is minimized even with very large redo group sizes, set FAST_START_MTTR_TARGET to a non-zero value to enable incremental checkpointing. If it is currently not set, then set `FAST_START_MTTR_TARGET = 3600`. This initialization parameter is only relevant for the primary database.

TROUBLESHOOTING AND ADVANCED TUNING

The physical standby database or a recovery instance is unlike the primary database. While the typical primary instance may comprise of 95 percent or more query activity with many CPU intensive operations, the media recovery instance is very write and update intensive. The recovery instance's goal is to apply changes to data blocks and write them to the data files. In many cases, the media recovery instance requires much less overall CPU resources but equal or greater I/O or memory capacity. Instead of possibly hundreds of CPU intensive operations on the primary, only the recovery coordinator (PID of the foreground process in `v$process`) or MRP process (MRP0 PID found in `v$managed_standby`) is generally CPU intensive. In many cases, fewer but faster CPUs will typically enhance recovery performance. There is no simple formula to predict the standby database system utilization. Here are some general observations that we found during our testing.

- The higher the read ratio on primary, the greater the difference in CPU utilization between the primary and standby databases.
- Higher numbers of sorts or complex queries executed will require more CPU utilization on the primary database. Queries and sorts do not create additional redo and thus do not create additional work on the standby database.
- Additional standby database CPU utilization is required when unique blocks are updated to account for the application of redo to the distinct blocks. The SQL*Loader runs modified 5 times less unique blocks compared to an OLTP run leading to 30% - 40% less CPU utilization on the standby while having almost three times the apply rate of the OLTP runs.

Therefore, tuning media recovery focuses primarily on removing system resources or database wait constraints.

Assess system resources

Use system commands such as UNIX `sar` and `vmstat` or system monitoring tools to assess system resources.

- If there are I/O bottlenecks or excessive wait I/Os, then stripe across more spindles/devices or leverage more controllers. A stripe size between 256KB to 1MB is optimal to leverage your I/O subsystem. Verify that this is not a bus or controller bottleneck or any other I/O bottleneck. The read I/O from the standby redo log should be greater than expected recovery rate.
- Check for excessive swapping or memory paging.
- Check to ensure the recovery coordinator or MRP is not CPU bound during recovery.

Assess database waits

- Database wait events from `v$system_events` and `v$session_waits`
  - Refer to the top 3 system wait events and tune the biggest waits first. You can determine the top system and session wait events by querying `v$session_wait` and `v$system_event` and taking the top waits with the largest "`TIME_WAITED`" value

    If recovery is applying a lot of redo efficiently, the system will be I/O bound and the I/O wait should be reasonable for your system. The following are the top recovery related waits that you may observe. Only apply the tuning tips if the recovery events are in the top 10 waits.

Table 2: Key Wait Event Table

Wait Name	Description	Tuning Tips
Log File Sequential Reads	Coordinator (recovery session or MRP process) wait for log file read I/O.	Tune Log Read I/O
PX Deq: Par Recov Reply	Coordinator synchronous wait for Slave (wait for checkpoints)	Increase PARALLEL EXECUTION MESSAGE SIZE to 8192
PX Deq Credit: send blkd	Coordinator streaming wait for Slave (wait for apply)	Increase PARALLEL EXECUTION MESSAGE SIZE to 8192
Free buffer waits	Foreground waiting available free buffer in the buffer cache	Increase DB CACHE SIZE and remove any KEEP or RECYCLE POOL settings.
Direct path read	Coordinator wait for file header read at log boundary checkpoint	Tune File Read I/O
Direct path write	Coordinator wait for file header write at log boundary checkpoint	Tune File Write I/O
RELEVANT FOR SERIAL RECOVERY ONLY		
Checkpoint completed	Wait for checkpoint completed	Tune File Write I/O Increase number of DB WRITER PROCESSES
db file parallel read	Wait for data block read	Tune File Read I/O

Appendix A provides an easy approach for deriving the recovery rate using queries.

Appendix B provides a sample diagnostic approach to help assess a more intricate recovery performance issue.

CONCLUSION

With Oracle Database 10*g* and its default settings, you should be able to inherently achieve fast media recovery performance. The practices described above are a checklist to ensure that media recovery is not being constrained by any bottlenecks. Optimized media recovery leads to reduced Data Guard switchover, failover or database media recovery times. This equates to more uptime and higher availability in the case of an unplanned or planned outage and helps enterprises meet the SLAs associated with recovery time objectives.

APPENDIX A – RECOVERY RATE SCRIPT

Here's a sample script to determine recovery rate:

REM First determine Log Block Size (LEBSZ) since it's different for each operating system. This query only needs to be executed once.

```
SELECT LEBSZ FROM X$KCCLE WHERE ROWNUM=1;
```

Start an iteration and record data in output file:

If media recovery was invoked:

REM Summary of max sequence per thread that has been applied

```
SELECT THREAD#, MAX(SEQUENCE#)
FROM V$LOG_HISTORY GROUP BY THREAD#
```

REM This query illustrates how many redo blocks have been

REM applied for a specific log sequence#.

```
SELECT TYPE, ITEM, SOFAR,
TO_CHAR(SYSDATE, 'DD-MON-YYYY HH:MI:SS') time from
V$RECOVERY_PROGRESS
       WHERE ITEM='Redo Blocks' and TOTAL=0;
```

Else managed recovery was invoked

REM This query describes the number redo blocks (block#) for a specific log sequence#

```
SELECT PROCESS, SEQUENCE#, THREAD#, block#, BLOCKS,
TO_CHAR(SYSDATE, 'DD-MON-YYYY HH:MI:SS') time
from v$MANAGED_STANDBY WHERE PROCESS='MRP0';
```

Sleep <interval>

To determine the recovery rate (MB/sec) for a specific archive sequence number, use one of these formula:

((SOFAR_END – SOFAR_BEG) * LOG_BLOCK_SIZE) /

(TIME_END – TIME_BEG) * 1024 * 1024

OR

((BLOCK#_END – BLOCK#_BEG) * LOG_BLOCK_SIZE) /

(TIME_END – TIME_BEG) * 1024 * 1024

APPENDIX B – RECOVERY TUNING STEPS

When debugging or attempting to tune Data Guard redo apply or media recovery, please follow the following steps.

1. Follow best practices prescribed in this document.

2. Assess I/O read rates from standby redo logs or archives

```
ALTER SYSTEM DUMP LOGFILE '<redo log name>' VALIDATE;
```

A trace file is generated with the redo read rate from a recovery perspective. As discussed earlier, this redo read rate is an upper bound on the recovery rate.

3. Assess Top 10 Database Wait Events

When debugging an immediate problem, execute this query at 30-60 second intervals during active recovery for a span of 10 minutes. Furthermore, execute the query at 30-60 minute intervals and maintain a minimum of 2 days data for more extensive recovery performance analysis. Focus on tuning the top 10 wait events if redo apply rate is not sufficient and leverage the Key Wait Event Table (Table 2):

```
select event, total_waits,
round(time_waited/100) "TIME(s)",
average_wait*10 "AVG(ms)",
TO_CHAR(SYSDATE, 'DD-MON-YYYY HH:MI:SS') time
from v$system_event where time_waited > 100 and
event not in ('rdbms ipc message','pmon timer','control
file heartbeat','smon timer')
order by time_waited;
```

To assess the actual wait time for a particular event, you need compare the difference between 2 snapshots.

$$\text{Snapshot Total Waits} = \text{End\_Total\_Waits} - \text{Beg\_Total\_Waits}$$

$$\text{Snapshot Avg Wait} = \frac{\text{End\_Time(s)} - \text{Beg\_Time(s)}}{\text{Snapshot Total Waits}}$$

If the snapshot Avg Wait is similar to the long running average from `v$system_event`, then you may be able to skip the step of deriving the snapshot average wait. The `v$system_event` wait average is the average since the startup which usually quite different from the snapshot average.

4. Use Log_Archive_Trace=8192

You can temporarily enable a special recovery tracing that gathers additional recovery metrics for more detailed debugging. The trace information is stored in the recovery process trace file. You should only enabled the tracing on the redo apply instance during active recovery for a span of 10 minutes.

a) Enable tracing by:

```
alter system set log_archive_trace = 8192;
```

b) Apply several archive redo logs or standby redo logs.

c) Disable tracing by:

```
alter system set log_archive_trace = <previous value or 0>;
```

5. Gather key system and recovery statistics

When debugging an immediate problem, execute this query at 30-60 second intervals during active recovery for a span of 10 minutes. Furthermore, execute the query at 30-60 minute intervals and maintain a minimum of 2 days data for more extensive recovery performance analysis. These metrics will be leveraged to derive system resource, recovery, checkpoint, and write rates. To get the snapshot statistic, you need to do the following:

$$\text{Snapshot Statistic} = \frac{\text{Stat\_End} - \text{Stat\_Beg}}{\text{Duration in Seconds}}$$

i) Gather system resource statistics using system monitoring utilities. Capture CPU, I/O and memory consumption.

ii) Gather recovery stats

```
select name, value from v$sysstat where name like
'recovery%' and value > 0;
```

iii) Gather checkpoint statistics

```
column name format a50
set pagesize 1000
select name, value, to_char(sysdate, 'hh:mi:ss') time
from v$sysstat
where name = 'DBWR checkpoint buffers written' or
name = 'DBWR checkpoints';

column event format a20
SELECT * FROM V$SYSTEM_EVENT WHERE EVENT LIKE
'%checkpoint%';
```

iv) Gather database File I/O Stats

```
select * from v$filestat where PHYWRTS > 1000 order by
writetim ;

alter system set log_archive_trace = 8192;
```

REFERENCES

1. MAA web site on OTN: http://otn.oracle.com/deploy/availability/htdocs/maa.htm
2. Oracle Database High Availability Architecture and Best Practices: http://download-west.oracle.com/docs/cd/B12037_01/server.101/b10726/toc.htm
3. Oracle Data Guard Concepts and Administration: http://download-west.oracle.com/docs/cd/B12037_01/server.101/b10823/toc.htm
4. Oracle Database 10*g* HA site on OTN: http://otn.oracle.com/deploy/availability
5. Oracle Data Guard site: http://otn.oracle.com/deploy/availability/htdocs/DataGuardOverview.html
6. MetaLink Note: 275977.1 – *Data Guard Broker High Availability*

Oracle Database10*g* Best Practices: Redo Apply and Media Recovery
September 2005
Authors: Lawrence To, High Availability Systems Team, Vinay Srihari, Recovery Team

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