

Best Practices for Oracle Database 10g Backup and Recovery

An Oracle White Paper
September 2005

Best Practices for Oracle Database 10g Backup and Recovery

Introduction	3
I. What's Important for Recovery.....	3
Recovery Time Objective (RTO)	4
Recovery Point Objective (RPO)	5
Backup Retention Policy.....	5
II. Oracle Best Practices for Backup and Recovery	6
Backup Best Practices	6
Incremental Backups	6
Backup Set Multiplexing.....	7
Input and Output Buffers	8
Speed Of Backup Devices	10
Media Recovery Best Practices	11
Application of Archived Logs and Incrementals.....	11
Parallel Recovery	11
General Database Performance	12
Flash Recovery Area Recommendations.....	13
Conclusion.....	14
Additional Resources	14
Metalink Notes	15
Acknowledgements	15

Best Practices for Oracle Database 10g Backup and Recovery

INTRODUCTION

A well-developed database protection plan is essential to any DBA's survival toolkit. The plan should minimally cover the 'what and how' in backing up critical data to local disk, onsite/offsite tape, or other tertiary storage:

- *What?* -- what are the recovery time objectives (RTO), recovery point objectives (RPO), and backup retention policies
- *How?* -- how to apply best practices to optimally backup and recover the database.

It is also imperative to include tested, documented procedures for restoring and recovering the data, in scenarios that can range from minor corruption to full scale site disaster.

Oracle Recovery Manager (RMAN) is the Oracle-suggested method for efficiently backing up and recovering your Oracle database. RMAN is designed to work intimately with the server, providing block-level corruption detection at backup and restore time, and the ability to merge incremental backups into image copy backups. RMAN automatically performs controlfile backups when the database structure changes, optimizes resource and space consumption during backup through block-level multiplexing and compression, and preserves a history of all backup activities. Out-of-the-box, RMAN can backup to leading tape and storage media vendors' products via the supplied Media Management Library (MML) API.

Part I of this paper discusses factors for determining RTO, RPO, and backup retention policies. Part II presents RMAN best practices for database backup and recovery.

I. WHAT'S IMPORTANT FOR RECOVERY

The essential questions for recovery are:

- How quickly should recovery complete for a datafile, tablespace, and database?
- What is the tolerance for data loss?
- How far back should backups be kept to satisfy the data retention policy?

Recovery Time Objective (RTO)

When considering business requirements of a data protection plan, it is vital to assess the impact of system downtime to the company's reputation, customer/partner relationships, and ultimately, top and bottom-line revenue. The recovery time objective is an upper bound of the period of time it will take to make the database operational. At a high level, recovery time consists of the time to repair hardware or storage failures, restore backups from disk and/or tape, recover the database by applying incremental backups and archived logs, and finally, open the database for user access.

Media recovery is needed if a database file is physically damaged and requires a restore and recovery operation. *Instance recovery* is needed when the database crashes and usually just requires a recover command (no restore operation).

Bounded and predictable instance recovery can be configured with the mean-time-to-recover target (MTTR) parameter, which adjusts the database checkpointing frequency. For additional details on instance recovery, refer to the [Oracle Backup and Recovery Advanced User's Guide](#).

The OS platform, disk/tape I/O thresholds, and other hardware options are significant factors in recovery time. The question boils down to:

"What is the business cost of downtime and data loss versus the hardware and software cost of reducing such downtime and data loss?"

Considerations in designing a backup and recovery strategy will help determine the trade-offs between required service levels and cost:

- What is the tolerance for system downtime, e.g. x days, y hours, z minutes/year? Consider all scenarios where media loss or corruption, caused by either hardware or software failures, can lead to system downtime.
- In light of the required service levels, evaluate I/O performance and cost of disk versus tape as the backup media. Consider backing up to disk for short-term, fast recovery, and periodically moving backups to tape for archival purposes.
- Are there different recovery service levels for different databases? Do service levels vary in relation to granularity of data, e.g. table, datafile, tablespace, whole database? Are RMAN backup sets sufficient or on-disk image copies necessary for faster recovery?
- Correcting logical error is easy with Oracle Database 10g Flashback Technologies. Additional disk space should be considered for flashback operations:
 - Undo tablespace (Flashback Query, Flashback Versions Query, Flashback Transaction Query, Flashback Table)
 - Flashback logs (Flashback Database)

Refer to [Application Developer's Guide – Fundamentals](#) for details on Flashback Query, Flashback Versions Query, and Flashback Transaction Query. Refer to the Backup and Recovery Basics Guide for additional details on [Flashback Database](#) and [Flashback Table](#).

- In the event that the production database server suffers an outage, is there a requirement for failover within seconds or minutes to a standby database to continue handling the workload? If so, refer to [Oracle Data Guard Concepts and Administration Guide](#) for details.

Recovery Point Objective (RPO)

The recovery point objective is the point-in-time in the past to which your data should be recoverable. For example, a zero RPO means that no committed data should be lost when media loss occurs, while a 24 hour RPO can tolerate a day's worth of data loss. In both cases, recovery should complete within a stated RTO.

For RMAN, determining the RPO falls into whether point-in-time recovery (PITR) is needed or not. If PITR is needed, then ARCHIVELOG mode must be enabled. This is also a required mode for online backups, in addition to recovery at the datafile and tablespace level. If PITR is not required, then the database does not require ARCHIVELOG mode. With this mode turned off, backups can only be taken offline, with RPO high threshold determined by the time between successive backups. Also, with this mode turned off, only the whole database can be restored and recovered.

For the above reasons, it is recommended that ARCHIVELOG mode be enabled to allow the full set of recovery options.

Backup Retention Policy

The backup retention policy governs how long backups should be kept, or how many versions of backups to keep at any time. This policy should be set in accordance with company mandated and/or government regulated record retention periods.

In RMAN, this policy is set using the `CONFIGURE` command, and applies to all backups taken during the RMAN client session for the specified target database. If a long term backup is needed, outside of the normal retention policy, this can be made using the `KEEP` option on backup.

There are two types of retention policies, and only one can be set at any time:

- Recovery window: This policy establishes the number of days within which point-in-time recovery must be possible.
- Redundancy: Establishes a fixed number of backups that must be kept. Backups that are in excess of this can be deleted.

The default retention policy is redundancy 1.

Note: When using a third party media manager, RMAN cannot implement an automatic retention policy if the media manager deletes backups. The media manager retention policy should always be longer than the RMAN retention policy to prevent expiration of tape backups that are still listed as available in the RMAN repository.

II. ORACLE BEST PRACTICES FOR BACKUP AND RECOVERY

Backup Best Practices

Incremental Backups

Incremental backups provide the ability to backup only the changed blocks since the previous backup. The benefits of incremental backups over full database backups include:

- Decreased backup time
- Reduced backup storage
- Decrease recovery time

Oracle provides two levels (0 & 1) of incremental backups. A level 0 backup backs up all database blocks and becomes the basis for future incremental level 1 backups. A level 1 incremental only backs up changed blocks since the last backup. A level 1 incremental offers two types of backups:

- Differential – All data blocks modified since the last level 0 or level 1 are backed up. This is the default type of incremental.
- Cumulative – All data blocks modified since the last level 0 are backed up.

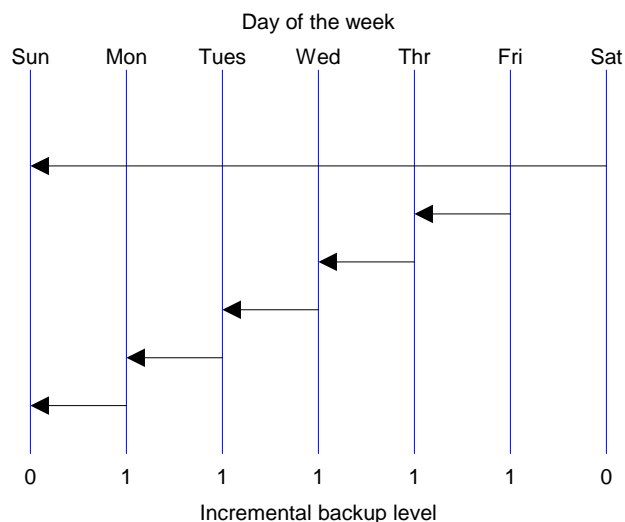


Figure 1: Differential Incremental Backups

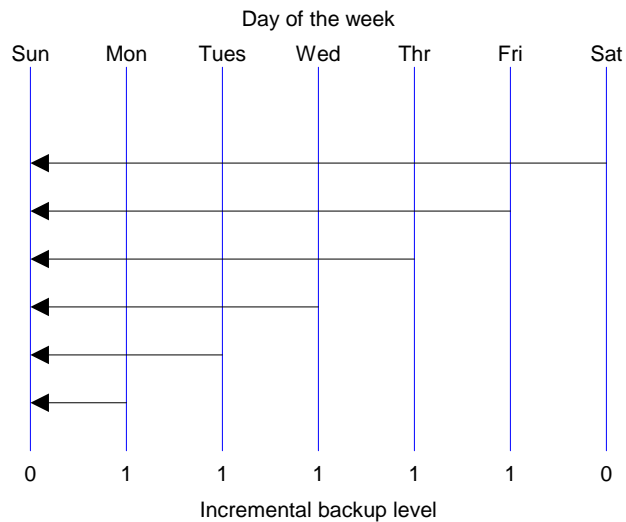


Figure 2: Cumulative Incremental Backups

Due to the reduced volume of data backed up using a differential backup, the backups will most often be smaller and also run faster than full or level 0 backups. Cumulative backups normally take longer than differential backups due to backing up more data, but during a restore, they will run faster because fewer incremental backups will need to be restored.

When looking at the timing of your own backups using the `ELAPSED_SECONDS` column in `v$backup_set`, it is important to note the amount of time spent taking the level 0 in comparison to the level 1 incrementals. If the time taking a level 1 gets close to the time for a level 0, it may be beneficial to make a 0 backup. The time taken to recover and restore using RMAN backups can be seen in the `ELAPSED_SECONDS` column of the `v$session_longops` view. Note that the contents of this view are cleared when the instance is shutdown and rows may get replaced with new sessions over time.

Optimized Incremental Backups

In Oracle Database 10g, enable the Block Change Tracking file, which greatly reduces incremental backup time. This file keeps track of changed data blocks using bitmap patterns to represent ranges of data blocks. Upon an incremental backup, instead of the entire datafile being scanned for changed blocks, just the changed blocks, as referenced in the file are read and backed up. Based on internal and external testing, this can result in 80%+ reduction in incremental backup times, depending on the locality of block updates.

Backup Set Multiplexing

Multiplexing allows RMAN to write blocks from multiple files, at the same time, to a single backup set, and provides better throughput than image copy of backups (e.g. never used blocks are not backed up in a backup set). The level of multiplexing

for a particular channel is determined by the minimum of the parameters `FILESPERSET` (default is 64), `MAXOPENFILES` (default is 8), and the number of files backed up on the channel (if specified).

However, there is a performance trade-off when restoring a subset of the datafiles or archived logs from multiplexed backup sets. It will take longer to read through a highly multiplexed backup to restore a single datafile, than from a smaller one. On the other hand, if the entire database is to be restored from tape, having a smaller multiplexing value and more backup sets may increase the time it takes to restore. This is due to more requests being sent to the media manager to retrieve the backup pieces. Consider your own experience to determine if single files or entire databases are restored more often.

Another consideration is time taken for resuming a backup that has failed. RMAN resumes the backup at the backup set level, so a large backup set, that was interrupted, will take more time to re-create than a smaller one (i.e. one with a smaller multiplexing value).

Input and Output Buffers

When RMAN takes a backup of datafiles it must read each block into an input buffer. The block is checked to make sure it needs to be backed up and then various block validation checks are made in order to detect corruptions. The block is then copied into the output buffers. For archive logs, each log file block also needs to be read and validated before it is written to the backup. Output buffers are used to store the multiplexed data blocks or archive log blocks, which are then written to the backup media (disk or tape).

Input Buffers

Oracle has documented the size and number of input buffers used during a backup in the [Backup and Recovery Advanced User's Guide](#):

Number of files per allocated channel	Buffer Size
<code>MAXOPENFILES < 4</code>	Each buffer = 1Mb, total buffer size for channel is up to 16Mb
<code>4 ≤ MAXOPENFILES ≤ 8</code>	Each buffer = 512k, total buffer size for channel is up to 16Mb. Numbers of buffers per file will depend on number of files.
<code>MAXOPENFILES > 8</code>	Each buffer = 128k, 4 buffers per file, so each file will have 512Kb buffer

Table 1: Read buffer allocation algorithm

The parameter that adjusts the number and size of read buffers allocated during the backup is the channel parameter `MAXOPENFILES`. This specifies the maximum number of datafiles that RMAN will concurrently open for input into the backup. The default value is *min(8, #files in backupset)*.

To show how the algorithm equates to the actual buffer allocation and size during a backup, refer to the following table:

MAXOPENFILES	Block size (bytes)	Buffer Size (Kb)	#Buffers per file	#files open at one time	Max. Total Buffer Size (MB)
1	8192	1024	16	1	16
2	8192	1024	8	2	16
3	8192	1024	5	3	15
4	8192	512	8	4	16
5	8192	512	6	5	15
6	8192	512	5	6	15
7	8192	512	4	7	14
8	8192	512	4	8	16
9	8192	128	4	9	4.5
10	8192	128	4	10	5

Table 2: Disk Read Buffer Allocations for Datafile Backups

The number and the size of the buffers allocated for each file (datafiles and archive logs), during a backup, can be determined using the following query:

```
SELECT set_count, device_type, type, filename,
       buffer_size, buffer_count, open_time, close_time
FROM v$backup_async/sync_io
ORDER BY set_count, type, open_time, close_time;
```

Monitor `v$backup_async/sync_io` to see how much memory is being allocated to input buffers, and adjust `MAXOPENFILES` if needed, to avoid memory starvation issues.

Output Buffers

The output buffers are sized differently for tape and disk devices:

Device Type	Number of Buffers Per Channel	Buffer Size	Total Buffer Size Allocated
DISK	4	1Mb	4Mb
SBT	4	256Kb	1Mb

Table 3: Default write buffer allocation

The SBT (tape device) output buffers are smaller than the disk channels due to the fact that the devices write slower, and hence a larger memory area is not required. However, it is possible to modify the size of the write buffers allocated to channels, if desired, e.g. to take advantage of larger block sizes or additional caching on tape devices. This is bounded, of course, by the physical I/O rate limit of the device.

The size for each write buffer can be adjusted using the `BLKSIZE channel` parameter. Increasing the buffer can reduce the number of I/O calls to the MML and subsequently reduce the amount of I/O time. It is recommended to monitor the I/O rates being achieved using `V$BACKUP_SYNC_IO` or `V$BACKUP_ASYNC_IO` and then adjust the `BLKSIZE channel` parameter accordingly.

Where is the Memory Allocated?

The memory for the read and write buffers are allocated from the PGA unless I/O slaves are being used. Each channel allocated creates a separate connection into the database and will have a separate PGA allocated. If I/O slaves are used, then the memory is allocated from the shared pool because the slaves will need to communicate data between themselves. If a large pool area is allocated using the `LARGE_POOL_SIZE` parameter, this will be used instead of the shared pool. It is recommended to use the large pool to reduce contention in the shared pool.

Oracle automatically uses asynchronous or synchronous I/O for backups to disk, based on the underlying operating system support.

Backups to tape also utilize synchronous or asynchronous I/O. With synchronous I/O, the channel process will wait for the write to tape to complete before continuing and filling the read buffers. When filling the read buffers, the tape device will wait for the next set of write buffer data. This prevents continuous tape streaming and is not the optimized way for taking backups. It is recommended to set the initialization parameter `BACKUP_TAPE_IO_SLAVES` to `TRUE`, to optimize overall backup to tape performance. In this way, the channel process allocates the writes to the slave processes and does not wait for them to complete before refilling the buffers.

Speed Of Backup Devices

The absolute maximum speed at which a backup can run is dictated by:

$$\text{Max Mb/Sec} = \min (\text{disk read Mb/s, tape write Mb/s})$$

It is not possible to make a backup go faster than this, period.

The first part of a backup, as already explained, involves reading each datafile/archive log file and placing the blocks into input buffers. The maximum speed at which these buffers can be filled depends on the physical location/caching and maximum read speed of the disks. The throughput being achieved through RMAN from the disks can be monitored using the

effective_bytes_per_second column in the V\$BACKUP_SYNC_IO/V\$BACKUP_ASYNC_IO views, where type= ' INPUT' . If this is lower than the expected read rate advertised for your disks, it is recommended to investigate OS resources using 'sar' and disk monitoring data of the particular disk/logical volume manager implementation.

The speed at which the devices used for writing the backup can also be monitored using V\$BACKUP_ASYNC/V\$BACKUP_SYNC_IO, using the effective_bytes_per_second column where type= ' OUTPUT' . If you are seeing a slower I/O rate than expected on the tape devices, then investigate MML and tape drive options to tune the physical tape block size (normally, the larger the better), tape compression settings, and ensure that adequate data is streaming to the drive.

Media Recovery Best Practices

This section assumes datafiles have already been restored from RMAN using a base level backup (level 0 or a full backup), and media recovery is now required.

Application of Archived Logs and Incrementals

After the base level backup is restored by RMAN, the recovery phase begins. RMAN will first look for any suitable incremental backups that can be used to roll the database forward. After the last incremental is applied (or, if no suitable incrementals are found), then the required archive logs are applied for complete recovery or to the desired point-in-time.

In general, recovering the database using incremental backups is faster than using archived redo log files. This can be tested by first taking a base level 0 backup of the database in ARCHIVELOG mode. For a period of time, run DML against the database so that it creates between 10 and 20 archived logs, and then take an incremental backup. Delete the database and restore the base level 0 backup. Then, use RMAN to recover the database, which will restore the incremental backup, and time it. Restore the base level 0 again, but instead of using RMAN to recover the database, use SQL*Plus with a 'RECOVER AUTOMATIC DATABASE' command, and compare the time it takes with the incremental-based recovery.

As previously discussed, the type of incremental backup (cumulative or differential) will also help determine if applying archived logs are slower than applying incremental backups.

Parallel Recovery

By default, Oracle uses a single process, which is the one issuing the recovery command, to carry out media recovery. This single process will read the archive logs and determine which changes are required for the specific datafiles being recovered. The data block is read into the buffer cache, and the redo is applied to it. Each time an archive log is completely applied, a media recovery checkpoint occurs, which signals DBWR to write all the dirty blocks in the buffer cache to the

datafiles. The file headers and control file also get updated with checkpoint information. This is a lot to do for a single process, especially when applying a large amount of redo to recover a large number of datafiles.

To decrease the time it takes to carry out media recovery, Oracle provides the Parallel Recovery feature, which is automatically enabled for Oracle Database 10g. On multiple CPU machines, Oracle automatically configures the degree of parallelism for use with the RECOVER command (with the exception of managed recovery), assigning one recovery process for each CPU. The process that issues the recovery command reads the archive logs as before, but instead of reading the blocks in the cache and applying the redo to them directly, it passes that work to the parallel execution slave processes. To make sure the redo is applied to the datafiles in SCN order, the parallel slaves work on different ranges of data blocks, so they will not interfere with each other, and the redo gets applied in SCN order for each data block.

When recovering a small number of datafiles using parallel recovery, it may take longer to perform due to the extra computation involved in partitioning the work, plus the extra IPC communication between the slave and coordinator processes. Monitor `v$session_wait` and `v$system_events` for the top wait events. If you are seeing 'PX Deq' events, for which there are several different types, then reducing the degree of parallelism may increase the recovery time.

You will also notice an increase in CPU usage when using parallel recovery due to the fact that the coordinator process is calculating the work partitioning for each redo change in the archive log, and there are more processes carrying out the recovery. As long as the CPU is not being saturated (look at something equivalent to 'sar -u'), then parallel recovery is not causing CPU issues.

The final thing to note about parallel recovery is the way it uses memory. By default, the `init.ora` parameter `PARALLEL_AUTOMATIC_TUNING` is set to `FALSE` and the buffers used for messaging between the parallel processes is 2148 bytes and is allocated from the shared pool. If the `PARALLEL_AUTOMATIC_TUNING` is set to `TRUE`, the default buffer is 4096 bytes and allocated from the large pool. By adjusting the size of the buffers, with `PARALLEL_EXECUTION_MESSAGE_SIZE` `init.ora` parameter, the speed of parallel recovery may be decreased, to save resource utilization. For more information on media recovery, take a look at the [Maximum Availability Architecture \(MAA\) white paper](#) on best practices for recovery.

General Database Performance

If the database performs slowly in day-to-day activities you shouldn't expect a fast recovery time. Recovery uses the underlying database server processes to carry out its tasks, so if general performance issues already exist, recovery may have similar issues.

Common areas that can be optimized to help recovery times include:

- I/O – Recovery is very read and write intensive due to having to read all the archive log contents, read the data blocks from the datafiles, and then write the dirty blocks to disk once the redo has been applied. By monitoring `v$filestat` along with OS-specific tools, you need to make sure the read and write times are acceptable for the hardware being used. Slow I/O can significantly slow down the time it takes to carry out media recovery.
- DBWR performance – DBWR’s main responsibility is to ensure there are enough clean buffers in the buffer cache to be used when data is being read from the datafiles. When a block has been updated, it is DBWR’s job to write the buffer to disk and return it for reuse within the buffer cache. If DBWR cannot keep up with cleaning the buffers, you will notice waits for the ‘free buffer waits’ event. If the I/O is not a problem, then using multiple DBWR process (or DBWR slaves if your OS does not support asynchronous I/O or asynchronous I/O is disabled) should reduce the waits.
- CPU Utilization – Because each data block that requires recovery is read into the buffer cache before the redo change is applied to it, there are a number of latches that must be acquired first. This includes the *cache buffers chains* and the *cache buffers lru chain*. Acquiring latches and making the changes to the blocks all takes CPU cycles, so you should make sure there is enough CPU bandwidth for the database during media recovery. If you are using parallel recovery, CPU usage will be even higher, as previously discussed.

Flash Recovery Area Recommendations

In Oracle Database 10g, the Flash Recovery Area, a filesystem directory or ASM disk group, can be setup to manage all recovery-related files, including archivelogs and backups, and automate the deletion of files that fall outside of the retention policy.

The recovery area requires a space quota, so all necessary files must be considered. If only archive logs and control file backups are needed, then estimate how many archive logs are generated between backups on the busiest day, and multiply their size by two to leave a margin of error.

If archive logs and flashback database logs should be kept, then multiply the archive log sizes between backups by four. If incremental backups are also kept, then look at the typical size of your incrementals and add that value. The size of an incremental backup is very dependent on the database workload.

Finally, if archive logs, flashback logs, incrementals, and an on-disk backup must be kept, then add the size of the database minus the size of the temp files. A rough rule of thumb for this final case is 2x the database size (minus temp files).

Additional examples for sizing the recovery area can be found in the Backup and Recovery Basics [documentation](#).

CONCLUSION

Developing best practices for backup and recovery of your Oracle databases begins with a candid assessment of all service level criteria for your environment, including recovery point and time objectives, data availability requirements, and data criticality. Once these criteria are identified, one can investigate the size and type of disk and/or tape storage needed, the type of server hardware required based on availability and performance, and then apply RMAN best practices to tune backup and recovery.

This paper has discussed several areas where RMAN can be optimized for performance, availability, and manageability:

- Differential vs cumulative incremental backup
- Backup set multiplexing
- Size of input and output memory buffers
- Speed of backup devices

Factors affecting media recovery performance:

- Application of archived logs and incrementals
- Incrementally updated backups
- Parallel recovery
- General database performance

Get the most out of Oracle backup and recovery, by putting these techniques to the test *today!*

ADDITIONAL RESOURCES

Oracle Backup and Recovery Advanced User's Guide, [Tuning Backup and Recovery](#)

[Tuning Oracle Recovery Manager](#), OTN White Paper

[Media Manager Troubleshooting Guide](#), OTN White Paper

[Oracle Database 10g Best Practices: Data Guard Redo Apply and Media Recovery](#), OTN White Paper

[OTN Case Studies](#)

- [Fannie Mae: Breaking the 1 TB/Hour Backup Barrier](#)
- [Starwood Hotels: RMAN in Oracle Database 10g Best Practices for Maximum Benefit](#)
- [CSX: Online RMAN Backups Protect over 16 TB of Data](#)

Oracle9i RMAN Backup and Recovery, Freeman and Hart, Oracle Press

Oracle Database Backup and Recovery Advanced User's Guide, [“Flashback Technology: Recovering from Logical Corruptions”](#)

Oracle Application Developer's Guide – Fundamentals, [“Using Flashback Features”](#)

Oracle Database 10g: The Top 20 Features for DBAs

- [Flashback Versions Query](#)
- [Flashback Table](#)

Metalink Notes

[Using V\\$BACKUP_ASYNC_IO / V\\$BACKUP_SYNC_IO to Monitor RMAN Performance](#) (Note:237083.1)

[RMAN 9i: Backup Optimization](#) (Note:142962.1)

[RMAN: I/O Slaves and Memory Usage](#) (Note:73354.1)

[RMAN: Quick Debugging Guide](#) (Note:132941.1)

[RMAN: Checking the Progress of RMAN Backup/Recovery](#) (Note:109159.1)

[RMAN: Resolving an RMAN Hung Job](#) (Note:145624.1)

[How to Configure RMAN I/O Block Size to Improve Backup and Recovery Performance](#) (Note:107213.1)

ACKNOWLEDGEMENTS

The author wishes to thank Stephan Haisley for the information in [Factors Affecting Backup and Restore Performance](#) and [Factors Affecting Media Recovery Performance](#). This comes from his [Backup and Recovery Optimization](#) paper on OTN.



Best Practices for Oracle Database 10g Backup and Recovery

September 2005

Author: Timothy Chien (Oracle Corporation), Saravanan Shanmugam (The Hartford)

Contributing Authors: Stephan Haisley (Oracle Corporation)

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

Copyright © 2005, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.