Technical report:

Oracle9*i* for UNIX and IBM N series

Backup and recovery

Document NS3130-0

October 12, 2007

Table of contents

Abstract

An Oracle9i for UNIX database on IBM System Storage N series delivers Oracle IT administrators with efficiencies and capabilities in database backup and recovery. The IBM N series and snapshot functionality optimize backup and recovery operations and dramatically improve performance over traditional data protection regimes.

Introduction

This document covers the techniques for backing up and restoring an Oracle*9i* for UNIX® database when an IBM® System Storage™ N series is used for database storage. Specifically, this report covers the following issues:

- Backing up a database while the database is shut down (cold backup)
- Backing up a database while the database is running (hot backup)
- Recovering a database which failed while running in noarchivelog mode
- Recovering a database which failed while running archivelog mode.

Key advantages

The performance and reliability of the backup and recovery operations are critical to effective database operation. IBM N series provides unique functions that afford the database administrator significant reliability and performance benefits in both backup and recovery. The key feature of the IBM N series approach to database backup is the use of IBM System Storage N series with Snapshot™ software.

Snapshots are critical because they allow the database administrator to quickly and easily create a read-only image of the entire file system, including the database files. After a snapshot is taken, the database can be returned to normal operation. A backup can then be written to tape from the IBM N series Snapshot software directory while the database is being used from the active file system. Actual backup-to-tape performance is thus of secondary importance, because the impact of this operation on the database's performance is negligible. For the backup operation, the paramount issue is that the database must either be:

- Shutdown and restarted in the minimal amount of time (cold backup); or
- Remain in hot-backup mode for the minimal amount of time (hot backup).

During a cold backup, the database is shutdown and unavailable. Obviously, reducing the period of time during which this is the case provides a key advantage.

During a hot backup, the database runs in hot-backup mode. This results in significant performance overhead, particularly in write operations. Further, being in hot-backup mode is risky. Thus, getting the database out of hot-backup mode more quickly also provides a key advantage.

Because the time required for the backup operation is reduced so much, many database administrators (DBAs) may find that offline or cold backups become feasible when running a database with storage on a filer. Many databases cannot be shut down for the hours required to perform an offline backup; however, shutting down the database for only a few seconds is acceptable. The functionality of IBM N series with Snapshot affords the DBA the ability to shut down the database for only a few seconds, during which a snapshot is taken, and then the database is brought back up again. Cold backups are preferable to hot backups for many reasons. For more information on this issue see the section entitled Cold backup.

Using hot backups, you can also take snapshots several times per day, and back up only one of them to tape. This offers the DBA additional flexibility. Effectively, you can take an online backup of your entire database every few hours. (**Note:** The amount of storage overhead associated with an IBM N series Snapshot is based upon the number of blocks that are different between the snapshot and the active file system. As this increases, the size of the storage requirements for IBM N series Snapshot increases as well. Thus, recent snapshots are generally less onerous to store than older ones.) For more information on this issue see the section entitled Hot backup.

Snapshots also provide key advantages for the restore operation. If the data that you need is in a snapshot, you can restore the entire database, regardless of size, in a few minutes. This is accomplished with the IBM System Storage N Series with Data ONTAP® *snaprestore* command. If you save several days' worth of snapshots, the chances are that you will never need to restore from tape at all, barring a failure on the filer itself. (**Note:** It is always recommended that all data be backed up to tape often. Snapshots should be used as a supplement, not a replacement, for tape backup.)

Take the following scenario: A company runs a 200 GB database on a UNIX platform, with storage on local disk. A glitch in one of the database server's processors results in corrupt data being written to the data files. As a result, the entire database must be restored from tape. At a rate of about 50 GB per hour, it will take approximately four hours to restore the data.

Now consider the same scenario with storage on a filer. Because the DBA saved a snapshot from a period prior to the failure, all that is required is to restore the datafiles from the snapshot. Using the IBM System Storage N series with SnapRestore™ *snaprestore* command this can be accomplished in only a few minutes, providing vastly improved mean time to recovery (MTTR) over the local disk solution.

For more information on restoring a database using the *snaprestore* command see: Restoring a database that runs in noarchivelog mode and Restoring a database that runs in archivelog mode.

Requirements and assumptions

This technical report covers backup and recovery of an Oracle9*i* database running under UNIX with database storage on an IBM N series unit. It is assumed that you are familiar with Oracle9*i* and the operation of IBM N series. It is also assumed that you are familiar with the operation of your version of UNIX. All examples in this technical document are tested on Oracle*9i* Enterprise Edition under Sun Solaris Version 8. The scripts contained in this paper might require significant modifications to run under your version of UNIX.

You must have your UNIX host configured to perform remote shell operations on the filer. This technical report assumes that the Oracle server machine has this capability. The examples in this technical report use the following syntax:

```
rsh -l root filer4 SomeCommand
```

As long as the hosts.equiv file on the filer is set up correctly, this is a secure operation. On this test filer, the following entry is placed in the hosts.equiv file:

```
orasun oracle
```

Orasun is the name of the Oracle server machine, and *oracle* is the name of the Oracle user account. This entry in the hosts.equiv file on the filer means that the oracle user on orasun is allowed to become root on the filer for the purpose of running rsh commands. As long as this is properly administered, it does not create a security hole.

The sample scripts in this technical report assume the following:

The name of the filer is *filer4*.
The location of the database is *filer4:/vol/voldb/home/oracle*.
The name of the Oracle instance is *test*.
The *filer4:/vol/voldb* directory is mounted on the Oracle server machine at */filer4*.

Backup

The following sections contain SQL and operating system scripts that show the techniques for backing up an Oracle database with datafiles stored on an IBM N series unit.

Cold backup

If the database is run in noarchivelog mode, the only option is to take an offline, or cold backup. A cold backup is often periodically taken for a database that is run in archivelog mode as well. A cold backup requires that the DBA shut down the database, back up all critical files using operating-system commands, and then restart the database.

Taking a cold backup using an IBM N series is a simple and fast operation. It requires three scripts that have the following functions:

Script name	Function
shutdown.sql	Shuts down the Oracle instance.
startup.sql	Starts up the Oracle instance.
docoldbackup.sh	A wrapper shell script that calls shutdown.sql, takes a snapshot, calls startup.sql, and performs the backup-to-tape from the snapshot. This script also does some housecleaning to keep four snapshots online at a time. If the script is run every night as a cron job, that means that four days' worth of snapshots would be available at any given time. For more information, see the section entitled: Key advantages.

The following is the text of the shutdown.sql script:

```
CONNECT / as sysdba
SHUTDOWN IMMEDIATE
```

You might want to wrap your SHUTDOWN IMMEDIATE command inside a shell script that provides some warning and error processing. For more information on how to do this, see the documentation that came with your version of Oracle.

The following is the text of the startup.sql script:

```
CONNECT / as sysdba
STARTUP
```

The following is text of the docoldbackup.sh shell script:

```
#!/bin/csh -f
# Shutdown the Oracle instance
sqlplus <shutdown.sql

# Rename and delete old snapshots
rsh -l root filer4 snap delete voldb old3
rsh -l root filer4 snap rename voldb old2 old3
rsh -l root filer4 snap rename voldb old1 old2
rsh -l root filer4 snap rename voldb new old1

# Take a new snapshot
rsh -l root filer4 snap create voldb new

# Startup the Oracle instance again
sqlplus <startup.sql

# Perform your backup using dump
# or other operating system commands here.
# For example:
# rsh -l root filer4 dump 0ufbln rst0a 63
/vol/voldb/home/oracle/.snapshot/new/
```

The options for dumping data from a filer to tape are beyond the scope of this technical report.

You will need to set the privileges on the docoldbackup.sh shell script to allow it to be executed. The following commands will do this:

```
orasun% chmod 750 docoldbackup.sh
orasun% ls -l docoldbackup.sh
-rwxr-x---   1 oracle        496 Jul 6 15:06 docoldbackup.sh
```

At that point, calling the docoldbackup shell script will do the following:

```
$ ./docoldbackup.sh

SQL*Plus: Release 9.0.1.0.0 - Production on Mon Sep 24 15:00:51 2001

(c) Copyright 2001 Oracle Corporation. All rights reserved.

Connected to:
Oracle9i Enterprise Edition Release 9.0.1.0.0 - Production
With the Partitioning option
JServer Release 9.0.1.0.0 - Production

SQL> Database closed.
Database dismounted.
ORACLE instance shut down.

deleting snapshot...
creating snapshot...
SQL*Plus: Release 9.0.1.0.0 - Production on Mon Sep 24 15:04:20 2001

(c) Copyright 2001 Oracle Corporation. All rights reserved.

SQL> Connected to an idle instance.
SQL> ORACLE instance started.

Total System Global Area     235701300 bytes
Fixed Size                       279604 bytes
Variable Size                  167772160 bytes
Database Buffers                67108864 bytes
Redo Buffers                      540672 bytes
Database mounted.
Database opened.
```

On this test system (a Sun e3500 running Oracle*9i* Enterprise Edition) the entire operation took approximately about one minute. (**Note:** The size of the database will not appreciably affect this interval, because the time required to take a snapshot is not dependent upon the size of the file system, or the files within the file system. Rather, a snapshot operation triggers a consistency point within the filer, and then simply copies the master inode of the file system to a new location. Thus, only about 4 Kb of data is actually copied.)

Hot backup

If the database is run in archivelog mode, you have the option to take online or hot backups. The database is open for user access while a hot backup is running. However, there is a significant performance hit, and an increased level of redo is generated. For this reason, it is critical that the database remain in hot backup mode for the absolute minimum period of time possible. IBM N series affords the DBA the opportunity to reduce the interval during which the database is in hot backup mode to only a few minutes.

Because a database in hot backup mode can be restored up to the point of failure in most instances, you should plan for recovery by organizing your files differently on the filer. The configuration recommended for IBM N series and Oracle databases running in archivelog mode is to place the datafiles on one volume, and the control file and log files (including the archived log files) on a second volume. This enables you to recover the datafiles using the *snaprestore* command, while maintaining the control and log files intact. After you apply the log files, your database is current up to the moment of failure. For more information on this, see the section entitled: Restoring a database that runs in archivelog mode.

There is less margin for error when taking a hot backup. Because a cold backup using an IBM N series takes only about one minute, a cold backup may be a feasible solution in many databases. For more information see the section entitled: Cold backup.

Taking a hot backup using an IBM N series is a simple and fast operation. It requires five scripts, two of which are created during the operation. These scripts have the following functions:

Script name	Function
dobegin.sql	Writes out the begin.sql script and then calls it.
begin.sql	Places all of the tablespaces that need to be backed up into hot-backup mode.
doend.sql	Writes out the end.sql script, and then calls it.
end.sql	Takes all of the tablespaces that need to be backed up out of hot-backup mode.
dohotbackup.sh	A wrapper shell script that calls dobegin.sql, takes a snapshot, calls doend.sql, and performs the backup-to-tape from the snapshot. This script also does some housecleaning to keep four snapshots online. If the script is run every night as a cron job, that means that four days' worth of snapshots are available at any given time. Alternatively, you can take a hot backup more frequently than once a day. For more information, see the following section: Key advantages.

The following is the text of the dobegin.sql script:

```
CONNECT system/manager
SET FEEDBACK off
SET PAGESIZE 0
SPOOL begin.sql
SELECT
    'ALTER TABLESPACE ' ||
    tablespace_name ||
    ' BEGIN BACKUP;'
FROM
    dba_tablespaces;
SPOOL off
@begin
EXIT
```

On our system (admittedly a small test database), the following begin.sql script is generated by dobegin.sql:

```
ALTER TABLESPACE SYSTEM BEGIN BACKUP;
ALTER TABLESPACE TOOLS BEGIN BACKUP;
ALTER TABLESPACE RBS BEGIN BACKUP;
ALTER TABLESPACE USERS BEGIN BACKUP;
ALTER TABLESPACE INDX BEGIN BACKUP;
ALTER TABLESPACE SUN2 BEGIN BACKUP;
```

The following is the text of the doend.sql script:

```
CONNECT system/manager
SET FEEDBACK off
SET PAGESIZE 0
SPOOL end.sql
SELECT
    'ALTER TABLESPACE ' ||
    tablespace_name ||
    ' END BACKUP;'
FROM
    dba_tablespaces;
SPOOL off
@end
EXIT
```

On our system the following end.sql script is generated by doend.sql:

```
ALTER TABLESPACE SYSTEM END BACKUP;
ALTER TABLESPACE TOOLS END BACKUP;
ALTER TABLESPACE RBS END BACKUP;
ALTER TABLESPACE USERS END BACKUP;
ALTER TABLESPACE INDX END BACKUP;
ALTER TABLESPACE SUN2 END BACKUP;
```

The following is text of the dohotbackup.sh shell script:

```
#!/bin/csh -f
# Place all of the critical tablespaces in
# hot backup mode.
$ORACLE_HOME/bin/sqlplus system/manager @dobegin.sql

# Rename and delete old snapshots
rsh -l root filer4 snap delete voldb old3
rsh -l root filer4 snap rename voldb old2 old3
rsh -l root filer4 snap rename voldb old1 old2
rsh -l root filer4 snap rename voldb new old1

# Take a new snapshot
rsh -l root filer4 snap create voldb new

# Remove all affected tablespaces from hot
# backup mode.
$ORACLE_HOME/bin/sqlplus system/manager @doend.sql

# Perform your backup using dump
# or other operating system commands here.
# For example:
# rsh -l root filer4 dump 0ufbln rst0a 63
/vol/voldb/home/oracle/.snapshot/new/
```

The options for dumping data from a filer to tape are beyond the scope of this technical report.

You will need to set the privileges on the dohotbackup.sh shell script to allow it to be executed. The following commands will do this:

```
orasun% chmod 750 dohotbackup.sh
orasun% ls -l dohotbackup.sh
-rwxr-x---   1 oracle   dba     496 Jul 6 15:06 dohotbackup.sh
```

At that point, calling the dohotbackup shell script will do the following:

```
$ ./dohotbackup.sh

SQL*Plus: Release 9.0.1.0.0 - Production on Mon Sep 24 15:09:57 2001

(c) Copyright 2001 Oracle Corporation. All rights reserved.

Connected to:
Oracle9i Enterprise Edition Release 9.0.1.0.0 - Production
With the Partitioning option
JServer Release 9.0.1.0.0 - Production

ALTER TABLESPACE SYSTEM BEGIN BACKUP;
ALTER TABLESPACE TOOLS BEGIN BACKUP;
ALTER TABLESPACE RBS BEGIN BACKUP;
ALTER TABLESPACE USERS BEGIN BACKUP;
ALTER TABLESPACE INDX BEGIN BACKUP;
ALTER TABLESPACE SUN2 BEGIN BACKUP;
Disconnected from Oracle9i Enterprise Edition Release 9.0.1.0.0 -
Production
With the Partitioning option
JServer Release 9.0.1.0.0 - Production
deleting snapshot..............
creating snapshot..............

SQL*Plus: Release 9.0.1.0.0 - Production on Mon Sep 24 15:11:32 2001

(c) Copyright 2001 Oracle Corporation. All rights reserved.

Connected to:
Oracle9i Enterprise Edition Release 9.0.1.0.0 - Production
With the Partitioning option
JServer Release 9.0.1.0.0 - Production

Connected.
ALTER TABLESPACE SYSTEM END BACKUP;
ALTER TABLESPACE TOOLS END BACKUP;
ALTER TABLESPACE RBS END BACKUP;
ALTER TABLESPACE USERS END BACKUP;
ALTER TABLESPACE INDX END BACKUP;
ALTER TABLESPACE SUN2 END BACKUP;
Disconnected from Oracle9i Enterprise Edition Release 9.0.1.0.0 -
Production
With the Partitioning option
JServer Release 9.0.1.0.0 - Production
```

On this test system (a Sun e3500 running Oracle*9i* Enterprise Edition) the entire operation took approximately one minute. (**Note:** The size of the database will not appreciably affect this interval, because the time required to take a snapshot is not dependent upon the size of the file system, or the files within the file system. Rather, a snapshot operation triggers a consistency point within the filer, and then simply copies the master inode of the file system to a new location. Thus, only about 4 KB of data is actually copied.).

Restore

The examples concerning restore in this technical report assume that the data required is still stored in a snapshot. If that is not the case (either because the snapshot has been deleted, or because the failure was on the filer itself) then you will need to perform an restore-from-tape operation. The requirements to do this are comparable to a similar operation on a UNIX system.

Restoring a database that runs in noarchivelog mode

If your database was running in noarchivelog mode at the time of the failure, point-in-time recovery is the only option. You simply recover the database up to the time when the last cold backup was made.

Recovery is a manual process. The DBA should be intimately involved in each step of this process. For this reason, we do not supply a set of canned scripts to do a recovery. Instead, we illustrate the steps involved.

The steps for point-in-time recovery are as follows:

1. Shut down the database with the *shutdown abort* command (if the database is still running).
2. Restore the datafiles (including the control file) from the last snapshot taken using the *snaprestore* command.
3. Restart the database.

The following commands should be issued from within Server Manager to shut down the database prior to recovery:

```
CONNECT / as sysdba
SHUTDOWN ABORT
```

Next, from the filer's console or a telnet connection, issue the following command:

```
vol snaprestore voldb -s new
```

A reboot of the filer will occur. After the filer has rebooted, you can start the database back up again:

```
CONNECT / as sysdba
STARTUP
```

The time required to do a recovery is minimal, only two or three minutes. This is true regardless of the size of the database, because the act of restoring a snapshot is merely the copying of a single 4 KB block. However, a reboot of the filer is involved, which takes approximately two minutes. This allows the DBA to recover a database that can be many gigabytes in size very quickly.

Restoring a database that runs in archivelog bode

Restoring a database that was running in archivelog mode at the time of the failure presents a much greater range of options. Depending on the needs of the moment, the DBA can do a point-of-failure recovery, a cancel-based recovery, or a point-in-time recovery. You can also restore individual tablespaces or files. If the affected tablespaces or files are taken offline, you can even do the recovery while the database is open. The full range of techniques covering all of these options is far beyond the scope of this technical report. The example shown herein assumes that the DBA wishes to perform a point-of-failure recovery of all tablespaces while the database is closed. (That is, the recovery will take the database forward to the point where the failure occurred.) The current control file and all archived and online redo log files are assumed to be available. The most current backup of the datafiles is assumed to be in the IBM N series Snapshot called *new*.

As stated in the section entitled Hot backup, the database must be stored on two volumes to support point-of-failure recovery using *snaprestore*. One volume should contain the datafiles, and the other volume should contain the control file and the log files (including the archived log files). This allows you to run the *snaprestore* command against the datafile volume only, and thus preserve your control and log files intact.

Recovery is a manual process. The DBA should be intimately involved in each step of the process. For this reason, we do not supply a set of canned scripts for recovery. Instead, we illustrate the steps involved.

The steps for point-of-failure recovery are as follows:

1. Shut down the database with the *shutdown abort* command (if the database is still running).
2. Restore the datafiles from the last snapshot taken using the *snaprestore* command. (A current copy of the control file should be stored on the volume where the log files are located.)
3. Mount the database using the current control file referred to above.
4. Recover the database from the archived and online redo log files.
5. Open the database for user access.

The following commands should be issued from Server Manager to shut down the database:

```
CONNECT / as sysdba
SHUTDOWN ABORT
```

To restore the datafiles, run the following command from the filer console or a telnet connection:

```
vol snaprestore voldb -s new
```

Again, it is assumed that the control and log files are stored on a separate volume (that is, vol1). Thus, this operation will not affect the control and log files. After the *snaprestore* command is issued, the filer will reboot. At that point, you should issue the following commands from within Server Manager to complete the recovery:

```
CONNECT / as sysdba
STARTUP MOUNT
SET AUTORECOVERY on
RECOVER DATABASE;
ALTER DATABASE OPEN;
EXIT
```

At that point, the database should be running in normal mode, with all data intact up to the point of failure.

Summary

An IBM N series system offers the Oracle DBA compelling advantages in terms of database backup and recovery. The use of IBM N series with Snapshot software, combined with conventional backup-to-tape techniques, can dramatically optimize the Oracle database backup operation. Retaining a number of online snapshots allows the DBA to restore the database without the necessity of restoring from tape in many circumstances. Backup and recovery performance is dramatically improved over conventional local-disk configurations.

Caveats

The use of network-attached storage is supported by Oracle in the context of IBM N series. However, this configuration has not been tested with any version of UNIX other than Sun Solaris, and has certainly not tested with all of the combinations of hardware and software options available on Solaris. There may be significant differences in your configuration that will alter the procedures necessary to accomplish the objectives outlined in this paper.

Trademarks and special notices

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

Network Appliance, the Network Appliance logo, Data ONTAP, SnapDrive, SnapRestore,and Snapshot are trademarks or registered trademarks of Network Appliance, Inc., in the U.S. and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.