

HPWORLD'97 Paper # 4095**Large Database Backup & Recovery Strategies For HP 9000**

by Jennie Hou & Harvey Skinner

Hewlett-Packard

19447 Pruneridge Ave

Cupertino, CA 95014

(Tel) (408) 447-5971 (Jennie)

(408) 447-6080 (Harvey)

OVERVIEW

The size of customer databases in all industries has been growing at a phenomenal rate in recent years and will continue to grow exponentially in the years to come. To meet the pressing demand of managing and supporting large databases, many Unix vendors have been developing robust media management products to backup and restore databases with focus on reliability, security, performance, and cost effectiveness, and high availability (reduction of down time). In parallel, customers are designing backup/recovery strategies to best meet their business requirements.

The objective of this paper is to focus on large backup/recovery strategy for the HP 9000 platform. This paper highlights the importance of having a solid backup/recovery strategy, factors that should be considered when planning and designing such a strategy, and the tradeoffs between various strategies in terms of database availability and ease of operation. This paper also documents HP's evaluation and testing of several media management products to perform backup, restore, and recover large Oracle & SAP/R3-Oracle databases on the HP 9000 platform.

This paper is written assuming that the target audience is somewhat familiar with the Oracle and SAP databases and applications. Basic administrative (DBA) commands such as opening and closing a database is expected. It is also expected that the target audience understands the concept of basic database composition in terms of data files, tablespaces, (online) redo logs, and (offline) archived redo logs.

RECOVERY NEEDS**Importance of Recovery**

When choosing a backup strategy, it is critical to remember the reason for data backup. The backup, though an common MIS operational procedure, is taken to ensure that data can be recovered in case of data loss or data corruption. The efficiency of the recovery process is based on how well planned the backup strategy is. If the backup strategy is poorly designed, the recover process can be long, tedious, and may also be incomplete. If the backup strategy is well-planned, the recovery process can be efficient, successful, and with minimal downtime for all levels of recovery (e.g., data file(s), tablespace(s), or full database).

The relationship between the backup process and the recover process is proportional. A simple analogy is to equate the database backup as an insurance policy. The more coverage you have, the better prepared you are for unexpected data loss or data corruption or even disaster. However, the premium can be costly in terms of backup time, system downtime, resources, complexity, and frequency. Therefore, it is very important that you have the right amount of insurance to ensure that

your business needs are being met and protected. Too much insurance will be costly upfront and with reduced marginal value. Too little insurance will definitely compromise your recovery process.

So, how does one identify what is the ideal backup strategy? There are so many options of doing backups as well as recoveries and they can be quite overwhelming. In this chapter, we'll first quickly mention some common causes of data loss or data corruption, backup types, recovery types, and archive mode versus no-archive mode. Secondly, we'll go over a list of recovery issues that you must consider. Lastly, we'll show you that with the answers that you have come up from the recovery issues, you can then derive your backup strategy.

In a way, the backup strategy is the by-product of how you want your recovery process to be. Because every business has different requirements and resources, each would have a customized backup/recovery strategy. It is important to realize that there is no off-the-shelf backup/recovery strategy planning product, but only a set of basic backup strategies that can be leveraged and tuned when designing your customized strategy from scratch that best meet your required business needs. However, there are off-the-shelf media management products that can assist you in handling your backup/recovery tactical functions once you have defined your strategy. Evaluation of several tools will be later discussed later in paper.

Common Causes Of Data Loss/Corruption

Data loss or data corruption can be attributed to the following failure categories:

- **Hardware:** CPU failure, media failure, disk controller failure, i/o path failure, etc.
- **Software:** Bugs in system, application, or database software can cause data integrity/loss problems.
- **Environmental:** Disasters such as fires, floods, earthquakes, power surges, abnormal temperatures, etc.
- **Operational:** Errors caused by human intervention such as poor DBA skills, operator errors, user errors, or system or database setup errors, as well as inappropriate backup/recovery procedures.

Types of Database Backup

So, how many ways are there for performing backups? Database backups can be classified into the two main categories: physical backups and logical backups. A physical backup is a backup where the actual physical database files are copied from one location to another (usually from disk to tape). System backups, cold backups, and hot backups are examples of physical backups. A logical backup is a backup that extracts the data using SQL from the database and stores it in a binary file. Usually, this file is kept on disk and then later backed off when the process is completed. This procedure is normally known as "export". The data can then be put back to the same database or another database at a later time. This procedure is normally known as "import". Exporting and importing can be done via Oracle's EXPORT/IMPORT utility. If the SAP/R3 applications are running on an Oracle database, then Oracle's export/import utility will also be applicable.

System Backups

This type of backup is the simplest to perform; however, it is very time consuming and requires the system to be unavailable during that timeframe. All applications and databases must be shutdown and

all users must be logged off during this backup. The backup is done by the system administrator/operator.

Since there is no user activity on the system during a system backup, it's assured that the data on the disk is consistent with the point in time when the system is taken down for backup. If this backup is to be used to restore a system at a later time, all changes made since this backup would be wiped out.

Cold Database Backups

Cold database backups involve shutting down the Oracle or SAP/R3-Oracle database in normal mode and backing up all required database files. This is also known as the "offline" backup. The offline backup procedure is very similar to the system backups except only a subset of the disk files are backed up to tape, namely only Oracle or SAP/R3-Oracle database files. Users at this point can still access other applications on the system except those affiliated with the Oracle or SAP/R3 Oracle database. Direct access to Oracle or SAP/R3-Oracle database via SQL commands is not permitted during offline backup.

Hot Database Backups

Hot database backups are taken while the Oracle or SAP/R3-Oracle database is open and operating in ARCHIVELOG mode. This is also known as an "online" backup. Although users can still have access to the database during a hot backup, it is important to schedule this activity when the database activity load is low. This will prevent too many archived redo logs from being generated. (Too many archived redo logs translate to longer recovery time after the database is restored from the backup media.) The online backup procedure consists of backing up all data files belonging to a tablespace or tablespaces, all archived redo logs (including the ones created during the period of the backup), and the control file. This is unlike an offline backup where the entire database is being backed up; the smallest logical unit in an online backup is a tablespace. Eventually, all tablespaces will need to be backed up.

Key advantages of an online backup are:

- The database is completely accessible to users while backups are being made, including the tablespaces that are being backed up.
- Ideal for a 7x24 shop.
- All tablespaces do not have to be backed up at the same time. With the use of archived redo logs, complete recovery can be performed.

Logical Backups

The logical backup, or EXPORT, creates a logical copy of the database objects and stores it in a binary file. The export utility is used to perform this function. The import utility then uses this file to restore these database objects back into the database at a later time. Both utilities are provided by Oracle as part of the RDBMS bundle.

This method of backup cannot provide point-in-time recovery. Also, it cannot be used with archived redo logs. One advantage of using a logical backup is that in case of a corrupted database block on disk, it will never be propagated to the logical backup because it would have been detected during the export process and the export would fail. The DBA can then take corrective action before attempting another logical backup. This would not be the case in physical backups. In physical backups, corrupted database blocks would have been copied to the backup copy. Usually, it is recommended that logical backups be taken occasionally if your main backup strategy is performing physical backups.

Since the database can be up and running during a logical backup, one disadvantage is that any changes made to the database during the export will not be incorporated into the export file. Since archived

redo logs can't be used with logical backups, there is no way to recover from these changes via logical backups.

Types of Database Recovery

Data recovery is really broken into two parts. The first part is to restore the needed database files from the backup media (if needed); the second step is to perform database recovery by issuing appropriate database commands so all data in the database can be recovered to the point of failure or a point in time depending on whether it's a complete recovery or an incomplete recovery respectively. In this section, the focus will be on the second step, database recovery. Restoring data will be discussed later in the backup strategy section.

Oracle provides three types of database recovery mechanism: block-level recovery, thread recovery, and media recovery.

Block Recovery

Block-recovery is the simplest type of recovery and it is done automatically by Oracle. This occurs when a process dies just as it is changing a buffer. The online redo log files for the current thread are used to reconstruct the buffer and write it to disk. This is transparent to both the user and the DBA. No backup media is required in this scenario.

Thread Recovery

Thread recovery is done automatically by Oracle as well. When Oracle discovers that an instance died leaving a thread open, thread recovery is performed as part of crash recovery or instance recovery. If the database has a single instance, then crash recovery is performed when the database is restarted by the DBA. If the database has multiple instances and one of these instances crashes, the second instance automatically then performs an instance recovery to recover the first thread. Regardless, the goal of thread recovery is to restore the data block changes that were in the cache of the instance that died and to close the thread that was left open. Online redo log files of the thread is used in this recovery. Again, no backup media is required in this scenario.

Media Recovery

Media recovery requires a recovery command to be issued manually by the DBA. This is not automatically done by the database as in the previous two recovery types. It is used to make backup data files become current after a restore from the backup media, or to restore changes that were lost when a data file went offline without a checkpoint. Both archived log files and online redo log files can be applied in this type of recovery. This type of recovery is most visible to DBAs since it requires human intervention.

Within the media recovery type, Oracle has provided three recovery options or commands. Recovery, after backup data is restored, can be made at data file level, tablespace level, or database level. Furthermore, these recovery options may or may not allow the database to be online during the recovery process depending on the failure and the recovery procedure that is going to be used.

The most comprehensive recovery is the database recovery. It will recover all necessary data files associated with a specified database. During a database recovery, the database must be offline. This is done by mounting the database but not opening it. This command can be used for both complete and incomplete recovery. For a complete recovery, issuing the "recover database" command as it is will do the job. However, in case there can't be a complete recovery due to various circumstances, incomplete recovery can be performed by issuing the same "recover database" command but with options. These options are listed below :

- recover database until cancel

- recover database until time 'yyyy-mm-dd:hh:mm:ss'
- recover database until change integer
- recover database until cancel using backup controlfile

Tablespace level recovery is a subset of the database level recovery. The "recover tablespace" command recovers only the specified tablespace. In this scenario, the database must be set online but with the named tablespace set offline. Tablespace recovery can only be used when performing a complete recovery. It will not work when performing an incomplete recovery.

Data file level recovery is the most granular of the three options provided. The "recover datafile" command recovers a named data file. Usually, a data file is only one of many data files that constitute a tablespace. When issuing the "recover datafile" command, the database can be open or closed depending on the file being recovered. When recovering a SYSTEM data file, the database must be closed (but mounted) since the database cannot be open with SYSTEM data files offline. When recovering a data file belonging to a user tablespace, the database can be open but the file that is being recovered need to be offline. Data file recovery can only be used when performing a complete recovery; incomplete recovery cannot be done.

The table below summarizes which type of recovery can be performed online versus offline (database is mounted but not opened) while recovering the database, a tablespace, or a data file as described above.

Recovery Command	Database Online	Database Offline
Recover database	No	Yes
Recover tablespace	Yes	No
Recover datafile	Yes	Yes

Database Operating In NOARCHIVELOG Mode vs ARCHIVELOG Mode

If the database is in NOARCHIVELOG mode, the database can only be protected from instance failure and not from disk (media) failure. One can only restore (not recover) the database to the point where the most recent offline full database backup was taken. All subsequent transactions cannot be recovered. This is known as an incomplete recovery. On the other hand, in ARCHIVELOG mode, the data can be protected from media failure. One can restore and recover the database to the point of failure. This is possible due to offline archived redo logs. These are copies of online redo logs which contain committed transactions and therefore can be retrieved during recovery. This is known as a complete recovery. Furthermore, with the database set in ARCHIVELOG mode, online backups can be taken. Distributed recovery can also be performed if all nodes of a distributed database system are in ARCHIVELOG mode. Although ARCHIVELOG mode provides more protection and flexibility, the cost is in terms of additional disk space to store offline archived redo logs and more administrative work for the DBA to maintain those offline archived redo logs.

Recovery Issues You Should Consider

When planning a backup/recovery strategy, you first need to identify what are your business requirements or how much you can tolerate in case of data loss or data corruption. Once you have identified your business requirements or tolerance level, you have then established the framework or boundaries of your backup/recovery strategy.

The following questions will assist you in the identification process:

1. How much transactional data can I afford to lose?
None <1 Day < 1 Week < Month All
2. How long can the database be unavailable during the backup process?
None Few Hours 1/2 Day 1 Day Entire Backup Process
3. How frequently is the structure of my database changing?
Often Sometimes Rarely
4. What is the activity level on the database?
High Medium Low
5. How much time do I have in recovering lost data?
Recovering lost data varies due to the degree of recovery needed as well as what backup strategy was used. Regardless of how much time is required, this step must be taken.
6. How much trained resource do I have to perform backup and recovery?
Adequate Somewhat Adequate Not Adequate
7. How much hardware resource do I have to perform backup and recovery?
Adequate Somewhat Adequate Not Adequate

Your answer to the first, also the most important, question forms the foundation of your recovery process. Your answer to the second question forms the foundation of your backup process. Your answers to questions 3 and 4 indicate how often backups should be taken. The fifth question really has no off-the-shelf answer. However, you need to consider this issue anyway as part of your planning. Answers to questions 6 and 7 can be used as the gauge factors when selecting your ideal option(s) from all those backup/recovery options that are available to you.

How To Derive Backup Strategy To Meet Your Recovery Needs

So, how does one derive a backup strategy based on one's recovery needs? In general, if the answers from the first two questions above are toward the left of the scale, there are less backup options available and less flexibility. Conversely, if these answers are toward the right of the scale, there are more backup options available as well as flexibility. Let's discuss the options one has based on the answers provided from the questions above.

1. How much data can I afford to lose?

If you cannot afford to lose any data, then your database must be operated in ARCHIVELOG mode, ideally with multiplexed online redo logs. For large databases with lots of activities, it is usually unacceptable to have data loss. For businesses where data is critical, such as a bank's ATM transaction data, data loss is not acceptable.

If you can afford to lose some data, then your database can be operated in NOARCHIVELOG mode. This will avoid the extra work required to archive online redo logs when they are full. However, the consequence is that complete recovery to

the point of failure is not possible. This ramification must be understood by management.

2. How long can the database be unavailable during the backup process?

The answer to this question is critical when evaluating whether an online or offline database backup should be performed.

If you can afford to keep the database offline during the backup process, then you can operate the database in NOARCHIVELOG mode. However, the only protection against a disk failure is the most recent full backup of the database. Furthermore, before an offline backup can be taken, all users must close the files that are being backed up; usually, this entails forcing users off the system to ensure files are being closed. This can require an additional X minutes of downtime before the backup is started. After the offline backup is completed, applications and database must then be restarted. This usually causes another Y minutes of downtime until users are again productive in the application. The backup strategy in this case is to take full offline backups regularly and frequently according to the amount of work that you can afford to lose.

If your database can't be unavailable during the backup process, then you must perform online backup. In an online backup, it's required that the database be in ARCHIVELOG mode. Based on your backup window, you want to select the backup strategy that best meet your needs. Three common strategies are:

- Full backup with frequent backups of archived redo logs
- Full backup with backups of rotating tablespace and archived redo logs
- Rotating tablespace backups with archive redo logs backups

3. How frequently is the structure of my database changing?

Any structure change made to the database should always incur a backup, of at least the affected tablespaces, to reflect the new structure. If the structure of your database changes often, this means that backups need to be taken more frequently regardless of what backup method you use.

4. What is the activity level on my database?

If your database data is highly volatile, then you will need more frequent backups than database with more static data. The main advantages of performing backups more frequently are to reduce the total recovery time and to minimize data loss. For example, if your database has ARCHIVELOG turned on and you do full backups infrequently, then your total recovery time would be much longer since there will be a lot of archived redo logs that need to be applied after a restore. Another example, if your database has NOARCHIVELOG turned on and you do full backups infrequently, then you would lose more data in case of media failure than if you do backups more frequently. Again, these tradeoffs must be decided by you in terms of how they best fit in your business environment and requirements.

5. How much time do I have in recovering lost data?

Recovery process normally involves two steps. The first step is to restore data from the backup media. The second step is to do database recovery by issuing appropriate database commands. The actual restore time depends on how many files need to be

restored and the size of these files. Likewise, the actual database recovery time depends on how many offline archived redo logs need to be applied. Again, this is based on per case basis. Here are some tips in minimizing data recovery time:

- Whenever possible, keep the database operational for tablespaces that are not affected while performing restore and database recovery of a few affected tablespaces, as in the case of corrupted datafiles due to a disk failure. This can be easily done by taking those affected tablespaces offline while keeping other tablespaces online instead of taking the entire database offline. Furthermore, multiple concurrent Server Manager sessions can be used to simultaneously recover datafiles to expedite complete media recovery.
- It's essential to have the backup media at hand. If the backup media is stored at an offsite (for disaster recovery purpose), the time to get the required backup media to the target site can be time consuming. It is always wise to keep two sets of backup media whenever possible. One at the site while the other one is shipped away for safe keeping in case of a disaster. In case you can't have two identical sets due to a narrow backup window, then keep the most current backup media on site and keep the previous one at offsite.
- Performing recovery in parallel is also another alternative. Parallel recovery allows several processes to apply changes from the redo log files. One process (the recovery session) reads the redo log files and dispatches the redo entries to a specified number of recovery processes. These recovery processes then apply the changes to the appropriate datafiles respectively. Parallel recovery can be specified via either the `RECOVERY_PARALLELISM` parameter in the `init.ora` file or in the database `RECOVER` command. Here is an example via the Oracle `RECOVER` command:
SVRMGRL> RECOVER TABLESPACE <tablespace name> PARALLEL (DEGREE 8);

6. How much trained resource do I have to perform backup and recovery?

If you have a highly trained staff, complex backup/recovery procedures can be easily done. On the other hand, if the staff is unfamiliar with Oracle database backup using the provided media management tools, then the process needs to be scripted and well documented to minimize errors. Furthermore, any backup/recovery procedure that requires manual intervention must be kept simple versus complex. This may affect how you choose a backup/recovery strategy. Also, the more automated the process is, there is less opportunity for operator errors which can prolong the recovery process.

7. How much hardware resource do I have to perform backup and recovery?

If you have adequate equipment resource, you may be able to save time during backup and restore. For example, if you have a large database, backing up using DLT drives far improves the backup/restore time than using DDS drives. Moreover, if you have multiple DLT drives, you can complete the backup/restore faster than with less DLT drives. Configuring the drives to handle multiple data streams also improves the throughput.

Other Backup/Recovery Considerations

Earlier, we discussed some common backup strategies. Here is more comparative information regarding these strategies:

- **Full backup with frequent backups of archived redo log**

Full backups with archived redo logs are the most efficient ones to recover. Based on the backup window that one has, full database backups can be taken daily or weekly. Of course, a daily full backup will significantly reduce the recovery time required over that of a weekly full backup since less archived redo logs need to be applied to the database as part of database recovery.

When performing full database backups, one can choose offline backup if the production environment can afford to shut down the database and affiliated applications during the entire backup operation. An offline backup does not require any archived redo logs to be applied to achieve logical data integrity which can slightly reduce downtime for recovery; however, it is required to apply archived redo logs in order to roll forward to a time after the offline backup was taken. When calculating the application and database downtime required for an offline backup, be sure to include the additional time required to remove users, shutdown the applications and database, and to restart the applications and database after the backup is completed. In many customer production environments, this can add an additional 30 minutes to 1 hour.

If an offline backup is not practical, then use an online backup. The use of an online backup during off hours with little or no database activity will save the additional downtime incurred with the process of shutting down the database and applications. If there is little or no activity, then there is very little difference in recovery time compared to an offline backup since there will be minimal archived redo logs generated during the online backup process which are required during the database recovery process.

If there is not a backup window available every day to perform a daily full backup, then consider doing a full database backup at least weekly and backup all archived redo logs frequently, at least once per day. Full recovery from a weekly full database backup will require restoring all of the database data and then applying all the archived redo logs created since the full backup occurred. Performing more frequent full backups will reduce the number of archived redo logs which must be available and applied in order to recover completely.

Usually, in most database recovery situations, only a tablespace or a data file needs to be recovered and does not require a full recovery from the backup unless it is a complete disaster recovery situation. The tablespace that requires recovery can be taken from a full Oracle backup. For this reason, the weekly full backup with frequent backup of archived redo logs can provide an appropriate level of recovery for many environments and many different recovery scenarios, not just full recovery. For environments where high database activities and data turnaround occur, the number of archived redo logs created during a week can make recovery using a week worth of archived redo logs prohibitive. In cases like this, consider performing two or more full backups a week or even using other forms of disaster recovery like mirroring data to a remote site or using some form of replicated system. Using these alternative methods of disaster recovery will alleviate the dependency of full recovery from backup and can significantly reduce the downtime encountered when a disaster occurs. Please note that using other alternatives for disaster recovery does not replace the need for backup. The backup is still required for many non-full recovery scenarios or situations where a roll-back of the database data is required. In addition, mirroring of disks and other data replication methods do not provide for recovery from data corruption or loss.

- **Rotating tablespaces backups with frequent backup of archived redo logs**

If a full database backup, even if only weekly, cannot fit within your database availability requirements, then there is another backup plan available for your consideration. This option is to take full backups of sets of tablespaces, rotating the backup sets so that each tablespace is backed up at least once during the backup cycle. It is advisable to keep the backup cycle, the period of rotation of tablespace set backups, to one week or less. A tablespace backup allows the spreading of the backup overhead (resources and time) over a greater period of time; however, some recoveries requiring recovery of multiple or all tablespace is more complex and takes longer to recover than a full database backup.

Using the rotating tablespace backup plan does avoid the requirement for one large backup window by spreading the backup overhead and/or downtime over a longer period of time, up to a week. However, there are tradeoffs in risk and complexity which must be considered before putting this backup plan into use.

When recovering from the backup, for either full or a partial recovery of the database, the number of archived redo logs required to be applied using the tablespace backup plan may be more than required for a full backup. Similar to a full backup, you will need to apply any archived redo logs since the last full backup; but in this case it is from the last full backup of the specific table(s) needing recovery.

Recovery from a tablespace backup will often take longer than from full backup, especially if multiple tablespaces on different backups must be recovered. In some cases more archived redo logs will need to be applied. Also the fact that the recovery is really multiple recoveries from multiple backups will add additional recovery time since media swap is more likely to be done. In the case of a full database recovery from a tablespace backup plan, it will always require one tablespace backup cycle worth of archived redo log data since some tables being recovered are from the start of the backup cycle.

Planning and implementing a tablespace backup plan is very complex and very maintenance intensive. The planning and implementation of a tablespace backup plan is mostly a manual effort. Rather than the chosen backup utility determining which tablespaces are included in the backup, the user must now do this. The user must also order tablespaces into backup sets so that the backups of tablespace sets are approximately equal and then develop scripts to effect the backup of the determined tablespace sets. Continual monitoring of tablespace sizes and changes is required because as tablespace sizes change or as tablespaces are split, adding or deleting of tablespace backup scripts will need to be manually modified to incorporate the changes and ensure all database data is backed up and balanced.

The tablespace backup is not only manually intensive and complex to implement, but recovery from a tablespace backup is more complex and manually intensive than from a full backup. Scripts specifically restoring each of the tablespaces according to the tablespaces set backups taken must also be developed and maintained.

Below is a comparison chart for Full Backup versus Tablespace Backup plans:

Full Backup With Archived Redo Logs		Rotating Tablespace Backup With Archived Redo Logs	
Advantages	Disadvantages	Advantages	Disadvantages

Minimal complexity for recovery and backup.	Requires at least one large backup window once per week.	Can distribute time required for full backup over the week.	Very complex and much manual calculation and monitoring required to ensure all database files are saved in a cycle of backups.
Any tablespace/table changes (additions, deletions, or splits) are included automatically.		Provides needed recovery if a specific table, tablespace, or data file is damaged and needs recovery.	Any tablespace/table changes must be manually reflected in backup and recovery scripts. Otherwise, data and recoverability will be compromised.
Easier and quicker recovery of full database in case of disaster situation.			Full recovery of the database will take more time and more manual setup than from a full backup.
Only need to apply the offline redo logs since the full backup for full database recovery.			Will need to apply a full backup cycles worth of offline redo logs for a full database recovery.
Can provide for table, tablespace, or data file recovery if needed.			

- **Full backup with backups of rotating tablespace and archived redo logs**

If there are a few tablespaces that are used predominately during production operation, it is feasible to back these up separately between full backups to reduce the amount of recovery time required. You can perform an online backup to backup the most active tablespaces and prevent application downtime for end users. By having rotating tablespace backups in between full backups can help reducing recovery time.

For example, lets say the following backups took place:

Sunday: Full backup + archived redo logs
Monday: Tablespace A backup + archived redo logs
Tuesday: Tablespace B backup + archived redo logs

On Wednesday, due to a disk failure, a couple data files associated with Tablespace B were lost. Instead of recovering Tablespace B from Sunday's full backup, Tuesday's rotating Tablespace B backup can be used instead. This way, less archived redo logs need to be applied and therefore can reduce the total recovery time.

Here is another example:

Let's say on Wednesday morning, due to a disk failure, data files associated with Tablespace C were lost. Since no rotating Tablespace C backup took place because Tablespace C isn't used predominately, recovery must be done using Sunday's full backup.

- **Additional Tablespace Backups Outside of Implemented Backup Plans**

Ongoing maintenance of the database will require database structure changes; i.e., tablespace splits, additions, and deletions. When performing structure changes (new, changed, or deleted tablespaces; new data files, etc.) you should always backup up at least the changed tablespaces and the database control file before and after the modifications.

Backing up tablespaces that are changed frequently may reduce the time required for any necessary recovery of those tablespaces. When a more recent backup of an intensively used tablespace is available, fewer archived redo log entries will have to be processed in order to update the tablespace.

- **Backup of Archived Redo Logs**

Regardless of which strategies you have chosen, the backup of offline archived redo log is crucial because it is the most important element of successful recovery.

The data in the Oracle database redo logs, both online and offline (archived), are very critical to the recovery process. Without this critical data, the ability to fully recover the database in many error conditions decreases. Even in cases where the last full backup of database data files cannot be restored, the previous full backup can be restored and then all archived redo logs since that backup can be applied to fully recover the database, roll the database forward, until the current point in time. The archived redo logs are also used for many other non-full recovery operations. This makes the data in the online and offline redo log files one of the most important data elements required for recovery. As such, special consideration should be given to the protection of this very important data.

Whenever an online redo log is switched, because the online redo log became full or some other operation invoked a redo log switch like an online backup, the data of the current online redo log is copied to an archived redo log file in the designated archive log directory. This switching of redo logs occurs automatically if the database is run in ARCHIVELOG mode.

The amount of activity against the database will determine the number of archived redo logs written during any period of time. You should constantly monitor the archive log directory and backup and delete archived redo logs when necessary to ensure there is enough disk space available for archival of the online redo logs. The number of archived redo logs to apply to a backup also increases the time required for recovery.

Always backup the archived redo logs immediately after a database backup, either full or tablespace backup. Make sure that the archived redo logs created during an online backup are saved immediately after since the online backup is not recoverable without this redo information.

To perform a recovery of the database, the archived redo logs must be available in their original, uninterrupted sequence. If an archived redo log is lost, complete

recovery of the database after an error is no longer possible. You will only be able to recover the database up to the point in time of the missing archived redo log.

Be sure to backup the archived redo logs frequently to have plenty of space available in the archive log directory. A database shutdown may occur when no more space is available for new archived redo logs in the archive log directory. For this reason, you should regularly monitor archived redo log storage and store the archived redo logs to tape.

- **Logical Backups**

Logical backups can be taken using Oracle's EXPORT utility. This utility copies the data and database definitions and saves them in a binary file in Oracle internal format. When using the Export utility, the database must be open. Since a snapshot of the table is taken before exporting, read consistency for individual tables is guaranteed; however, inter-table consistency is not. Therefore, if you want a snapshot of all the tables in the entire database, no changes should be made to the database while taking an export of the database. This can be easily done by opening the database in RESTRICT mode where users cannot access the data during export.

It is important to know that a logical backup takes more time than a physical backup. If you are exporting to disk or if you have multiple tape drives, parallel export sessions can decrease the time to obtain a full export versus a long serial export session to tape.

So why choose logical export over physical export. Here is a chart outlining the advantages and disadvantages of logical backups.

Advantage of logical backup	Disadvantage of logical backup
Can detect data block corruption during export; thus, can fix this problem before propagate to the backup media	Export takes much longer than physical backups
There are three types of exports: COMPLETE, INCREMENTAL, CUMULATIVE	Cannot take advantage of offline archived redo logs to recover to the point of failure
Offers a great deal of flexibility in what data and data definitions to export	Backups are in binary format.
Provides an extra protection from user errors or structural failures. Can easily restore a 'dropped' table via the import utility versus an incomplete recovery from physical backup	Import takes longer than restoring from physical backup media.
Backups are portable and can be imported into any database on the same machine or over the network to another remote database	Cannot perform recovery to the point of failure. Only can perform a point-in-time recovery.

In addition to three export options (complete, incremental, cumulative), there are also three modes that can be used when exporting data. These modes are:

- **Table mode:** Objects that can be exported in this mode are table definitions, table data, owner's grants, owner's indexes, table constraints, and table triggers.
- **User mode:** Objects that can be exported in this mode are all "table mode" objects, clusters, database links, views, private synonyms, sequences, snapshots, snapshot logs, and stored procedures.
- **Full database mode:** All "user mode" objects, roles, all synonyms, system privileges, tablespace quotas, tablespace definitions, rollback segment definitions, system audit options, all types of triggers, and profiles.

In summary, logical backups have certain advantages over physical backups. However, it is most ideal if logical backups are taken in addition to the physical backups. This way, based on the failure, you have various options. For example, if a user drops a read/only table accidentally and you only need to recover the table, it's quicker to recover using an export backup. However, if a data file is lost due to a disk failure and the you need to recover the database, which is in ARCHIVELOG mode, to the point of failure, then a physical backup is quicker and can ensure complete recovery.

Backup/Recovery Strategy Comparison Chart

Below is a chart depicting various backup options that were described earlier and their associated tradeoffs in terms of data loss, database down time, recovery time range, system administrative efforts, archive mode versus non-archive mode. The intention of this chart is to give you a quick glance of the backup options that are available for you to choose from.

Backup Strategy	Data Loss (None,Some)	DB Downtime (None,Some)	Recovery Time (1-4, 1=least)	Ease of Operation (1-2, 1=least)	DB Mode (Archive, NoArchive)
Full Weekly + Rotating TB + Archived logs (Online)	None	None	3	2	Archive
Full Weekly + Archived logs (Online)	None	None	4	1	Archive
Rotating TB + Archived logs (Online)	None	None	4	2	Archive
Full Daily + Archived logs (Online)	None	None	2	1	Archive
Full Weekly + Archived logs (Offline)	None	Some	4	1	Archive
Full Daily +					

Archived logs (Offline)	None	Some	2	1	Archive
Full Weekly + w/o Archive logs (Offline)	Some	Some	1	1	NoArchive
Full Daily + w/o Archive logs (Offline)	Some	Some	1	1	NoArchive
Logical backup via Export Utility (Full DB)	Some	None (May have restricted mode)	4	1	N/A

Basic Database Design Considerations & Backup Rules

Good database design is critical in maximizing your backup/recovery efficiency. Here are some simple rules that can make your backup scheme robust and minimize your recovery time as in the case of a media failure.

- Always create appropriate user tablespaces to keep non-system data. Don't create user tables in the system tablespace. It becomes extremely difficult to separate user data with system data when performing backup/restore since allocation of these tables within the system tablespace is managed by Oracle versus the DBA. The DBA has no way of knowing what system data file contains what table(s).

An good example is designing a database for Oracle Applications. Oracle Applications is Oracle's financial, manufacturing, and human resources product suite. Currently, there are over 40+ modules in Oracle Applications Release 10.7. Although a customer can choose and pick whatever the modules one needs, it is important to design the database with modularity. Let's say this customer has purchased the following modules: General Ledger (GL), Account Payable (AP), Account Receivable (AR). The database should be designed by creating a user tablespace for each module for storing data and creating another user tablespace for storing indexes. Each tablespace created should have associated physical data files(s) in mnemonics for easy identification. Here is an example what tablespaces should be created for the above modules:

Tablespace Name	Associated Physical Data Files
GL	/oracle/apps107/glfile1.dbf, /oracle/apps107/glfile2.dbf
GLINDEX	/oracle/apps107/glindx1.dbf
AP	/oracle/apps107/apfile1.dbf
APINDEX	/oracle/apps107/apindx1.dbf
AR	/oracle/apps107/arfile1.dbf, /oracle/apps107/arfile2.dbf, /oracle/apps107/arfile3.dbf

ARINDEX

/oracle/apps107/arindx1.dbf,
/oracle/apps107/arindx2.dbf

- Offline archived redo logs are most efficient when they are put on disk by default and then later backed up to tape. This is done by specifying the archive destination directory to be a location on the disk in your initialization file (init.ora). However, the archived redo log destination should not reside on the same physical disk device as any database data file or online redo log files. If a data file or an active redo log file is lost due to a disk failure, then these offline archived redo logs will be needed in the recovery process. If they are on the same physical disk, then they would be lost as well.

If the archived redo log file or an online redo log file (that is not active currently) is lost, then the current database should be backed up using either an online or offline option to a backup device. Operations then can safely continue.

- Maintain multiple copies of the control file. A copy of the control file should be placed on several different disk devices mounted under different disk controllers. A control file can be added to the database by shutting down the database, copying the control file, altering the init.ora parameter CONTROL_FILES, and restarting the database.
- If database files are backed up to disk, a database file residing on the same physical device as its backup copy is not adequately backed up. In case of a physical device failure, the backup database files are lost as well. It is important to have a separate disk or disks to maintain the backup copy of the database files. Backing up database files to disk can speed recovery since the required file need not be restored from tape. In general, backing up to disk often allows recovery to run in a shorter amount of time.
- Online redo logs should be multiplexed and should maintain a minimum of two members for each group. Two members of a log group should not reside on the same physical device since it defeats the purpose of multiplexing log files.
- Maintain multiple copies of archived redo log files to allow recovery in case of multiple media failures. It is recommended to keep one set on disk and the other one on tape.
- The procedure of rolling forward a database or database file from a backup can be simplified and made faster by keeping on disk all archived redo log files needed to roll forward the least recently backed up database file of a database. In many cases, much of the time necessary for recovery is spent restoring archived red logs from tape.
- Whenever the database structure is changed by adding, deleting, renaming, or dropping a log file or a data file, the control file should be backed up since the control file stores the schema of the database. In addition, any data file that is added should be backed up as well. The control file can be backed up while the database is open using the following command: *alter database backup controlfile to 'filename'*
- Keep an extra disk if possible. An extra empty disk can be configured and brought online in case of a disk failure instead of waiting for that disk to be fixed. The lost data can be restored onto that extra disk and the database can be recovered then.

Raw Database versus File System Database

A database can be built on either raw devices or filesystems. Filesystem based databases, in either HFS (Hierarchical File System) or JFS (Journal Files System) format, are more common due to ease of configuration, maintenance, and abundant third-party tools. Raw-device based databases, where data files and online redo log files are set on raw devices instead of filesystems, were becoming more popular because they provide higher performance, as much as 50% more. The performance improvement is done by bypassing the filesystem buffer when writing from Oracle's SGA (System Global Area) buffer area to disk directly. This process can reduce CPU activity, avoid paging within memory between the buffer cache and the SGA, and avoid filesystem overhead as well as read ahead overhead. Furthermore, raw backup can be taken and this is considered to be significantly faster than filesystem backups, up to twice as fast in some cases. This helps tremendously when the database is very large and the backup window is very narrow. Although there are advantages of building databases on raw devices, the tradeoffs are less third-party system management tools, more complex configuration, more maintenance, and less granularity in backup and recovery.

There are significant enhancements and performance improvements being implemented in the JFS filesystem which decrease the performance advantages of RAW backups of filesystem data. This is described in more detail in the Filesystem Backup & Recovery section of this paper.

Raw Backup & Recovery

A raw backup is an image backup of the individual disk volumes, either physical for logical (via LVM - Logical Volume Manager), that contain the database data. With a raw backup, the recovery choices are somewhat limited because one has to recover the entire volume image in order to access any part of the backup. There are some backup utilities that have provided functions to interpret the raw backup image data to locate and recover individual datafiles; however, this is not the norm and also this can be very time consuming depending on how these functions are implemented. Furthermore, it is important to recover the raw volume data onto a volume of at least the same size to ensure successful recovery.

Raw backup tends to be faster in general, there are also scenarios where raw backups are slower than the filesystem based backups. For example, if the volumes that are being backed up have actual files with only 50% or less of the storage space allocated, the raw backup will copy the entire volume regardless. Unlike in a filesystem backup, only the actual allocated storage space is being backed up and therefore can be faster in such scenario.

There are also cases where raw backup is the only solution. For example, if the database is built on raw devices, then only raw backup is viable.

Filesystem Backup & Recovery

Filesystem based backup is the most common method used today since majority of the databases are built on filesystems. This backup method allows backup of complete filesystems or sets of files in a filesystem. Many backup utilities multiplex multiple data streams together to get more optimized throughput performance. To ensure optimal recovery, these recovery utilities must be able to demultiplex the data stream from the backup device to have the same number of active data streams during the recovery process as in the backup process. Otherwise, multiple restore operations will need to be performed (one per multiplexed data stream) in order to recover all the data.

HFS Versus JFS filesystem

There are different types of filesystems you can use with HP-UX, the major ones being the HFS filesystem and the JFS filesystem. Our investigations have shown that backup utilities performing a filesystem backup can achieve a significant backup performance gain when using a JFS filesystem

instead of a HFS filesystem, assuming of course that the backup utility has been tuned for use with a JFS filesystem.

For filesystem based databases, use of the JFS filesystem can significantly increase the backup performance and at the same time lower system bus utilization. This is done by taking advantage of the Direct I/O (DIO) feature available with the JFS filesystem to reduce the number of memory to memory copies of data performed during the backup.

For example, with a filesystem backup of an HFS filesystem, data is first read from the disk into the HP-UX I/O buffer cache. The data is then copied from the I/O buffer cache to the backup application's data buffer. Then the backup utility writes the data from the application's data buffer to the backup device.

With the JFS filesystem, a feature called Direct I/O (DIO) can be used to eliminate the memory copy of the data from the I/O buffer cache to the backup application's data buffer. Thus, with a filesystem backup of a JFS filesystem using the DIO feature, the data is read directly from disk into the backup application's data buffer and then written to the backup device. The elimination of the memory copy from the I/O buffer cache to the applications memory buffer can both increase the backup performance and reduce system bus utilization significantly. Use of the DIO feature makes a filesystem backup of a JFS filesystem nearly as efficient as a RAW backup of the logical or physical disk device, yet retaining the advantages of better storage management capabilities that are inherent with filesystem based data.

Since most of the new Oracle database servers, are being installed with JFS filesystems, those systems should experience better backup performance than if they had used HFS filesystems. This of course assumes that the backup utilities have been tuned like OmniBack II to take advantage of the JFS DIO feature.

Users which have their databases implemented using an HFS filesystem should consider migrating them to a JFS filesystem. This requires backing up the data in the HFS filesystem, removing the HFS filesystem, making the JFS filesystem, and then restoring the data which was previously backed up.

TOOLS EVALUATION

Although your ideal backup/recovery strategy must be built from scratch to best meet your business requirements, the actual tactical management of backing up and restoring data can be done with the help from various off-the-shelf media management products. As mentioned earlier, several media management tools have been evaluated and tested against large databases. These products are: Hewlett-Packard's OmniBack II (version 2.1), Oracle EBU (version 2.0.14.20) integrated with OmniBack II, and SAP/R3 (version 3.0C) backup/recovery utilities integrated with OmniBack II. Please keep in mind that there are other third party media management tools available; however, they will only be listed and not evaluated in this paper.

OmniBack II

OmniBack II is a Hewlett-Packard product that provides backup and restore services for most Unix based computer platforms, PC server platforms, and Windows based PCs. This product offers centralized management of multiple heterogeneous systems, rich media management, support for stacker, library, and silo based backup devices. OmniBack II can be used by itself or it can be integrated with other specific database backup utility such as SAP/R3 backup utility and Oracle EBU.

Evaluation Summary

Usability

In our evaluation of Using OmniBack II, version 2.1, by itself to perform Oracle database backups and restores, we find the product to be much more efficient than using the UNIX provided tools (e.g, tar, cpio, dd, etc). OmniBack II is easy to install, it provides high level administration via GUI interfaces, it can perform backups in both online and offline mode, and it works with various media devices (e.g, DLTs, DATs).

Similar to other standard Unix backup/restore utilities, OmniBack II doesn't understand the concept of a database (meaning what data files constitute a database). Therefore, whenever the database grows or shrinks in terms of data files, the associated datalists must be manually edited to reflect the latest changes. Failure to do this will inhibit full backup and recovery of the database. Along the same line, a tablespace concept is not comprehended by OmniBack II; therefore, backing up or restoring of a tablespace must be done manually by selecting the appropriate data files and of the correct timestamped versions.

Backup/Restore

OmniBack handles backup extremely well for both raw devices and file system database files. Multiple filesystems or raw-device datafiles that constitute a database can all be backed up under one datalist (script). However, when restoring the entire database (as from a full offline backup), there are variations based on whether its raw devices or filesystems. If the database is built on raw devices, restoring a database can be done under one datalist. If the database is built on one filesystem only, then restoring the database can also be done under one datalist. However, if the database is built on multiple filesystems, then restoring the entire database must be done on per filesystem basis where each filesystem has its own datalist. Therefore, restoring a database with large number of filesystems can become quite tedious.

Online/Offline

OmniBack II is ideal for performing offline backups. Online backup of Oracle databases can be done using OmniBack II by itself; however, this would require extra steps such as having the DBA issuing appropriate database commands before starting a backup and after the backup is completed. Because performing online backup using OmniBack II is not transparent to the DBA and requires more administration, this may have an impact on your backup/recovery strategy depending on how well trained is your DBA on performing Oracle online backup.

Performance

OmniBack II provides scalability by taking advantages of parallel hardware such as SMP, multiple storage devices, and multiple disks. It also can reduce backup time by providing parallel processing capabilities where multiple data files can be backed up simultaneously on multiple devices in parallel. Furthermore, OmniBack II provides buffered I/O where all I/O is buffered in shared memory and thus enables all physical devices to operate at their peak rates.

Oracle Enterprise Backup Utility (EBU)

Oracle7 Enterprise Backup Utility (EBU) must be integrated with a third-party media management product for it to provide complete backups and restores of the target databases. The interface to a media management product is provided by EBU API which is a set of routines that allows the reading and writing of data to the media device. In addition to handling direct interaction with the media devices (such as Digital Linear Tapes, 4-mm tapes, 8-mm tapes, and library CDs), the media management product also provides features such as scheduling, data compression, media aging reports and labeling.

The EBU provides an intelligent interface to Oracle databases. Once a database identification information is provided, the EBU can locate the tablespaces, data files, control file(s), and archived redo log files associated with that database. All this information is stored in a catalog called the

BACKUP CATALOG. This catalog can save a user the effort of having to specify all the physical details about the database structure.

One thing to note though, the EBU addresses all backup and restore processes but it does not address the database recovery process. The DBA must apply standard database recovery procedures after EBU completes the restore process.

Currently, Oracle has approved three media management products. These are: Hewlett-Packard's "OmniBack II", Legato System's "Legato Networker", and EMC's "Enterprise Backup". In this white paper, only the OmniBack II product is evaluated.

Evaluation Summary

Usability

Overall, Oracle's EBU integrated with Hewlett-Packard's OmniBack II media management product demonstrated that management of data backup and restore is much easier and quicker than using OmniBack II alone and significantly over the standard Unix backup/restore utilities. Furthermore, EBU understands the concept of a full database and a tablespace; thus, invoking backup or restore at database level or tablespace level can be done not only easily but can also greatly reduce potential human errors.

On the other hand, the installation of the Oracle EBU and integration with OmniBack II are extra steps. High level understanding of how this product works is very important. Backup of the Oracle Catalog Database, which contains configuration information and detailed backup histories of all your target databases, must be taken regularly which would require additional maintenance.

Backup/Restore

Backup or restore can be initiated via either the EBU **obackup** commands or using the GUI interfaces provided by OmniBack. EBU also can handle both raw devices and filesystems with a consistent interface so the DBA does not need to deal with two set of OmniBack GUI screens or obackup commands. Furthermore, when performing aggregated restore, EBU will automatically retrieve the most current files, even across multiple backups, to ensure the shortest possible recovery time.

Online/Offline

Lastly, Oracle EBU offers both online backup as well as offline backup options. The online backup option has been made even simpler by having EBU transparently issuing the online backup database commands (begin backup and end backup) instead of having the DBA issuing them. This online backup option is extremely important for shops that require 7x24 operation. Without this option, only offline backup can be performed. In addition, online restore (at tablespace/datafile level) is also offered by EBU. This enable the database to be operational during the restore process, of course, excluding the part that is being restored.

Performance

In addition to leveraging all the performance advantages that OmniBack II offers (as mentioned above), EBU also offers the added backup/restore performance via datafile multiplexing capability and null block compression. Especially with the null block compression capability, performance benefit can be realized over other backup utilities.

Miscellaneous

EBU also offers both checksum and block header verification capabilities to ensure that the restore is successful. EBU also has the auto-configuration capability where database configurations are automatically verified and comprehensive backups are guaranteed even when database structures have changed.

Architecture and Function of EBU

The EBU consists of several executables and a catalog that maintains all current and historical information about the databases in your backup strategy. These executables are briefly described below.

obackup:

- User interface for EBU operations
- Coordinates other EBU processes
- Communicates with the target database and the Backup Catalog
- Spawns the instance Manager if one is not already running

brtio:

- Coordinates between obackup, brdk, and brtp processes
- One brtio process per parallel I/O stream
- Must have same number of brdk processes as in brtio processes

brdk:

- Handles read/write of disk files
- One brdk process per I/O stream per file by default
- Mux specifier can create multiple concurrent brdk processes within one I/O stream

brtp:

- Handles read/write of tape files
- One brtp per parallel I/O stream

brd:

- It's the daemon process that monitors the Backup Catalog and obackup
- Manages cleanup for all backup and restore operations which terminate abnormally

Figure 1 below illustrates the functional process flow of EBU integrated with OmniBack II.

Architecture Of EBU Integrated With OmniBack II

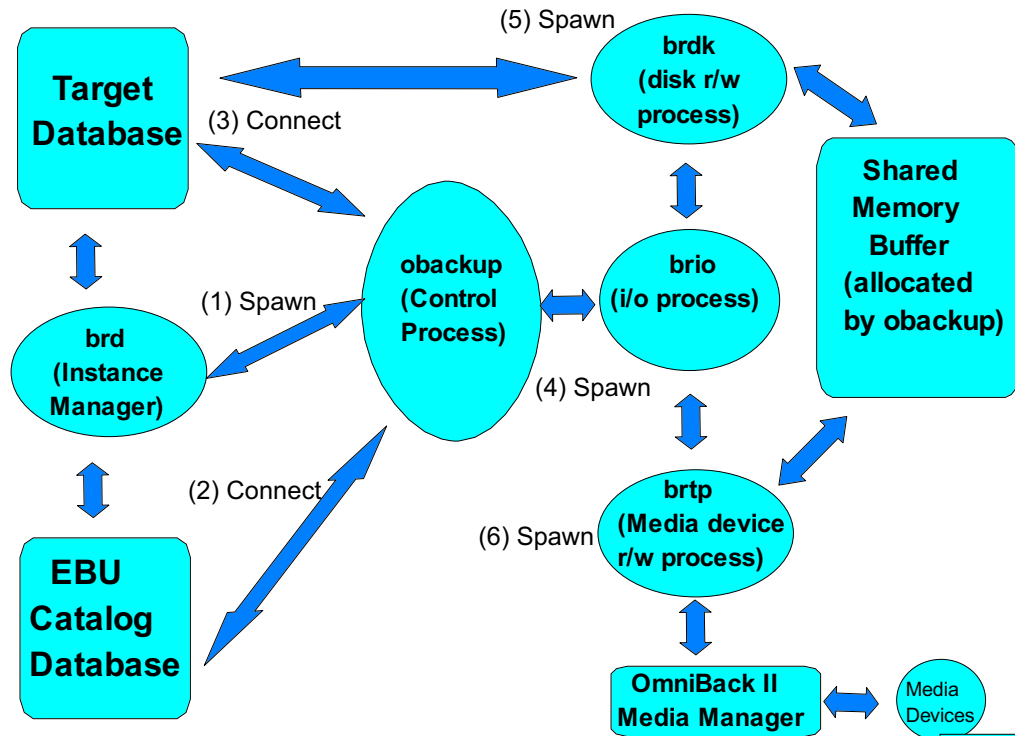


Figure 1

In Figure 1 above, a single database is being backed up onto or restored from a media device using EBU integrated with OmniBack II. The backup/restore process can be initiated by either OmniBack II GUI interface or EBU obackup command.

During the backup process, the following steps take place:

- 1) The **obackup** control process spawns the **instance manager (brd)**, if there isn't one already running.
- 2) **Obackup** connects to the **EBU catalog database** to get target database configuration information, verifies the registration information, and then builds the list of files to be backed up.
- 3) **Obackup** connects to the **target database** if an online backup is performed by automatically putting the database into backup mode.
- 4) **Obackup** spawns the **i/o process (brio)**. (Multiple brio processes can be spawned if parallel data streaming capability is invoked.)
- 5) **Brio** spawns the **disk process (brdk)**; the database files are read by the disk process into a **shared memory buffer** area allocated by obackup. This buffer alleviates any temporary speed mismatches between disk and tape throughputs. (Multiple brdk processes can be spawned for corresponding brio processes.)
- 6) **Brio** spawns the **tape process (brtp)**; the tape process reads the backup data from the **shared memory buffer** and writes it into the **media device** via **OmniBack II**. (Multiple brtp processes can be spawned for corresponding brio processes.)

During the restore process, the following steps take place (steps 5 and 6 from the backup process are reversed in this case):

- 1) The **obackup** control process spawns the **instance manager (brd)**, if there isn't one already running.
- 2) **Obbackup** connects to the EBU **catalog database** to get target database configuration information and the registration information, builds a list of files to be restored, and locates the backup fileset for the files to be restored.
- 3) **Obbackup** connects to the **target database** and verifies that the files to be restored are set offline.
- 4) **Obbackup** spawns the **i/o process (brio)**. (Multiple brio processes can be spawned if parallel data streaming capability is invoked.)
- 5) **Brio** spawns the **tape process (brtp)**; the tape process reads the backup data from the **media device** via **OmniBack II** and writes it into a **shared memory buffer** allocated by obbackup. (Multiple brtp processes can be spawned for corresponding brio processes.)
- 6) **Brio** spawns the **disk process (brdk)**; the disk process then reads data from the **shared memory buffer** area and writes the data to disk. (Multiple brdk processes can be spawned for corresponding brio processes.)

In both cases above, the I/O and control (brio and obbackup) processes also serve managerial functions to process exceptions like errors, shutdowns, and cleanup operations. The catalog database tracks state information about each backup and restore job and provides detailed history information for reporting and browsing functions.

Backup and restore processes can be significantly accelerated by taking advantage of EBU's parallel data streaming capabilities. If multiple media devices are available, a single obbackup control process can spawn multiple brio processes to read/write all available media devices concurrently.

SAP/R3 Internal Backup/Recovery Utilities

SAP/R3 is one of the industry-leading applications that provides components to manage comprehensive financial, manufacturing, human resources, and sales & distribution functions. In most of SAP/R3 installations, the underlying database is Oracle.

SAP/R3 provides its own backup and restore utilities with the application. These utilities are: BRBACKUP, BRARCHIVE, BRRESTORE, and SAPDBA. These utilities provide the ability to perform online backup of Oracle database data using standard Unix utilities (e.g., cpio, cp, tar, dd) by default or by integrating with third-party external utilities, such as HP OmniBack II and Legato Networker (in our evaluation OmniBack II was used). Furthermore, SAP/R3 provides its own backup media volume management, software compression, and backup optimization techniques.

Evaluation Summary

Usability

In our evaluation of using SAP/R3, we found that by integrating it with external tools, such as HP OmniBack II, a much richer backup and recovery solution can be provided for the SAP/R3-Oracle database. Furthermore, limitations invoked by the defaulted standard Unix tools can be overcome. Overall, this makes backup and recovery more robust. Some key advantages are:

- Perform online and offline backups
- Perform complete database or tablespace backups and recovery because SAP/R3 is database aware, similar to Oracle EBU

- Can use a separate backup session to backup other supporting files and non-database files
- Avoid backup and recovery of extraneous files left in the database filesystems
- Allows use of high level database management tool, SAPDBA,, to aid in database recovery
- so explicit Oracle database recovery commands don't need to be issued manually. This truly is a big advantage!
- Any database changes (such as added or deleted database filesystems) are automatically included in the backup and recovery.

Backup/Restore

SAP/R3 integrated with OmniBack II enable both online and offline backups to be taken for both raw-device and filesystem database files. Initiation of a backup can be done using either SAP/R3 GUI (Administration) , SAP/R3 backup or archive log utility (BRBACKUP or BRARCHIVE), SAP/R3 database utility (SAPDBA), or OmniBack II GUI. Of the four methods, we found the use of the SAP/R3 SAPDBA utility, which also invokes the BRBACKUP and BRARCHIVE programs, to be the most helpful and successful. We usually use the Monitor window of OmniBack II to monitor the backup activity and verify that tape media is being mounted as required.

For restore and recovery of the SAP/R3-Oracle database, either SAP/R3 BRRESTORE or SAP/R3 SAPDBA can be used. From our evaluation, again the SAPDBA utility is the best method available. The SAPDBA utility not only invokes restore of the database data, but will also perform a database integrity check, identify redo log data required, and invoke restore and application of the redo log data for full database recovery. If the BRRESTORE or OmniBack II Restore utilities are used, the steps needed to bring the restored data up-to-date and restore logical data consistency must be invoked manually.

Online/Offline

SAP/R3 offers online backups. If integrated with OmniBack II, then both online and offline backups can be taken. When performing online backup, SAPDBA will automatically issue the appropriate Oracle database commands (e.g., ALTER TABLESPACE <tablespace name> BEGIN BACKUP and ALTER TABLESPACE <tablespace name> END BACKUP). No intervention from the DBA is required.

Performance

By integrating SAP/R3 with OmniBack II, all the performance advantages from OmniBack II can be leveraged. There are additional load leveling features available which may allow further optimization.

Architecture and Function of SAP/R3 Integrated With OmniBack II

SAP/R3 backup and recovery utilities consists of several executables. These executables are briefly described below.

• **BRBACKUP**

The SAP/R3 backup utility, BRBACKUP, is used to perform both offline and online backups of the entire or specific tablespaces of the SAP/R3-Oracle database (e.g. data files, online redo log files, and the control file). You could also use BRBACKUP to backup non-database files or directories, but this is not recommended by SAP or HP because of the complexity of recovery of this non-database data.

• **BRARCHIVE**

This program provides services for backup of the offline redo log files.

- **BRRESTORE**

This is the restore program that restores the files (data files, online and offline redo log files, control file, and other files) that were backed up by the BRBACKUP and BRARCHIVE utilities. It allows you to restore all or part of the database data files and/or offline redo log files which were backed up. BRRESTORE only restores the requested files, it does not perform the additional recovery operations needed to recover the database. To do this, use SAPDBA utility.

- **SAPDBA**

SAPDBA is a higher level utility which uses the above utilities (BRBACKUP, BRARCHIVE, BRRESTORE) to perform backup and recovery functions. The main difference between the SAPDBA utility and the other three utilities, is the added value that SAPDBA provides. For example, the BRRESTORE utility will simply restore database files while SAPDBA will restore database files (by invoking BRRESTORE for the restore) but will then also automatically apply redo log data to attain full recovery and logical data integrity.

SAPDBA also provides guided steps to perform various recovery operations for either full or partial recovery. We have found the SAPDBA utility the better utility to use for both full and partial database recovery.

- **BACKINT**

BACKINT is the interface between SAP/R3 and the external backup/recovery utilities. All requests made by BRBACKUP, BRARCHIVE, BRRESTORE are passed to OmniBack II as in our case.

The integration of external backup utilities with the SAP/R3-Oracle database environment provides a very ideal and complete backup and recovery solution. The strengths of each are brought together in a complementary manner:

- BRBACKUP/BRARCHIVE takes care of the database handling and generating lists of files to backup or restore.
- The external backup utility is responsible for moving the data from disk to the backup media and for managing the backup media.
- BRBACKUP/BRARCHIVE use the *backint* program to pass backup requests to the external backup utility. The backup requests are in the form of a list of files which are to be backed up. The external backup utility can then do further processing on the backup file list to further optimize the backup by ordering and separating the file list into multiple parallel backup datalists which will backup the database files to multiple backup devices concurrently.
- The BRRESTORE utility also uses the *backint* program interface to invoke the external backup utility to restore requested files. BRRESTORE can also query the external backup utility for what versions of the data files are available to be recovered. This along with the higher level recovery features of the SAPDBA utility provides a very usable and successful method to perform either full or partial database recoveries.
- The external program provides all the data file backup and restore operations as requested by BRBACKUP, BRARCHIVE, and BRRESTORE.
- BRBACKUP, BRARCHIVE, and BRRESTORE evaluate confirmation messages returned by the external backup utility which reflect the status of the requested operation.

SAP/R3 Backup Process and Data Flow

The process structure, as shown in Figure 2 below, starts with the invocation of the backup from either of the four methods available and ends with the database data or offline redo log files being written to tape. In summary, the actual process structure and data flow is as follows:

1. The backup process (database files or offline redo logs) is invoked by the SAPDBA utility, BRBACKUP/BRARCHIVE program directly, SAP GUI, or the OmniBack II GUI or command
2. Which ever method is used to start the backup, either BRBACKUP (for database data files) or BRARCHIVE (for offline redo log files) is invoked
3. The backint program, delivered with the external backup utility, is then called as the go-between for BRBACKUP/BRARCHIVE and the external backup utility. In this case the external backup utility is OmniBack II.
4. The *backint* program then launches the OmniBack sapback process, which is specially written for SAP/R3 backup. A second sapback process is also created by the first sapback process. For an SAP/R3 backup, the OmniBack sapback process replaces the OmniBack vdba process used for other types of backup.
5. As can be seen in the figure below, one of the sapback processes reads data from the disk into a shared memory buffer. The other sapback process performs messaging required by OmniBack, either locally or over the network. Note that there is a pair of the sapback processes started for each disk agent required.
6. The sapback process invokes the bma process, which in turn invokes another bma process.
7. Of the bma processes reads the backup data from the shared memory buffer and writes it to the tape device. The other bma process performs messaging required by OmniBack, either locally or over the network. Note that there is a pair of the bma processes started for each backup device used in parallel for the backup.

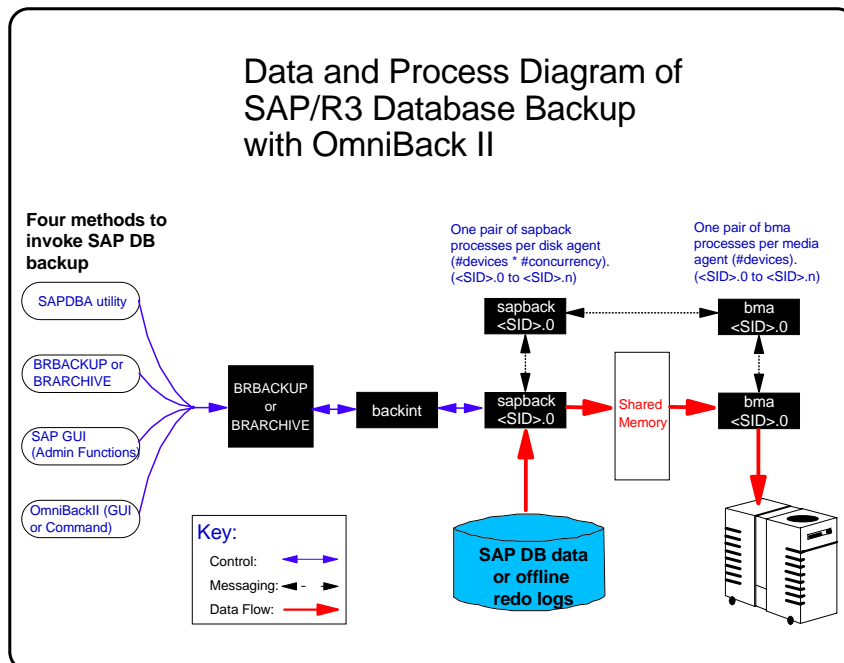


FIGURE 2

SAP/R3 Restore Process & Data Flow

The process structure, as shown in Figure 3 below, starts with the invocation of the restore operation from either of the two methods available and ends with the database data or offline redo log files being written to disk. In summary, the actual process structure and data flow is as follows:

1. The database restore process (database files or offline redo logs) is invoked by the SAPDBA utility, or the BRRESTORE program directly.
2. Which ever method is used to start the restore, the SAP/R3 BRRESTORE program is invoked
3. The *backint* program, delivered with the external backup utility, is then called as the go-between for BRRESTORE and the external backup utility. In this case the external backup utility is OmniBack II.
4. The *backint* program then launches the OmniBack saprest process which is specially written for SAP/R3 restore. A second saprest process is also created by the first saprest process. For an SAP/R3 restore, the OmniBack saprest replaces the OmniBack vdba process used for other types of data restore.
5. As can be seen in the figure below, one of the saprest processes reads data from a shared memory buffer and writes the data to the disk. The other saprest process performs messaging required by OmniBack, either locally or over the network. Note that there is a pair of the saprest processes started for each data stream, disk agent, used when the SAP/R3 backup was created.
6. The saprest process invokes the rma process, which in turn invokes another rma process.
7. One of the rma processes reads the backup data from the tape device and writes it to a shared memory buffer. The other rma process performs messaging required by OmniBack, either locally or over the network. Note that there is a pair of the rma processes started for each backup device used in parallel for the restore.

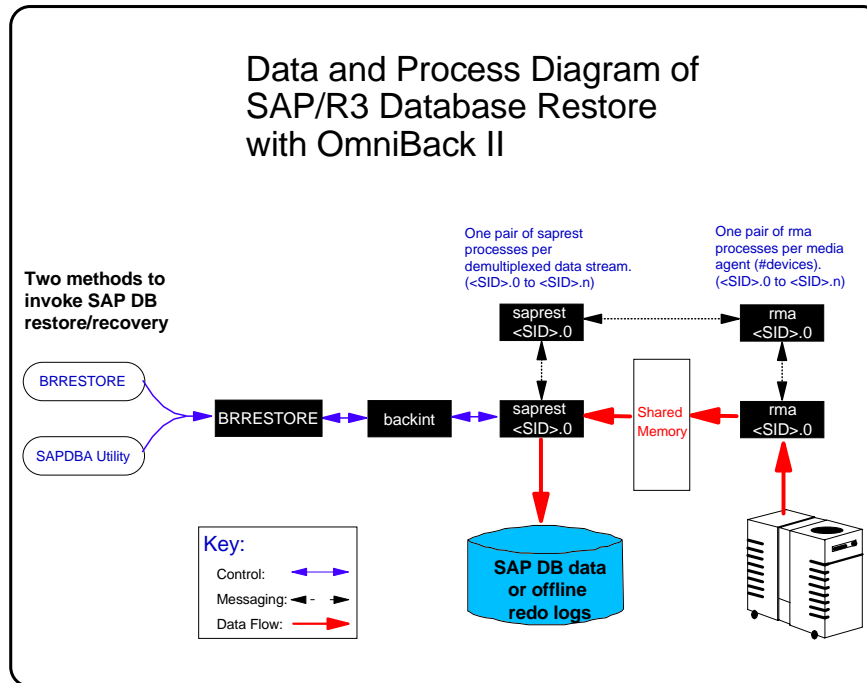


Figure 3

SAP/R3 Database Recovery

The SAPDBA high level database management utility is extremely useful and valuable when performing database recovery of the SAP/R3-Oracle database. SAPDBA works closely with the BRBACKUP, BRARCHIVE, and BRRESTORE utilities and in turn the external backup utility via the *backint* program interface to know which database data was backed up and which versions are available for recovery. SAPDBA assists with the entire recovery operation, both full and partial recovery, and provides a step-by-step recovery process which can be easily followed. By using SAPDBA, no explicit Oracle database recovery commands need to be issued manually. This greatly simplifies the database recovery process.

The following requirements must be met to run a SAPDBA assisted database recovery successfully and usually automatically:

- The complete, undamaged logs for BRBACKUP and BRARCHIVE operations (located in the <ORACLE_HOME>/sapbackup and <ORACLE_HOME>/saparch directories) must be available for SAPDBA to determine the location of the required database files and archived redo log files.
- Undamaged copies of the required database data files must be available on backup media
- Undamaged copies of all redo log files that were written between the time the database data file was backed up and the time the recovery is required must be available on disk or on backup media
- The control file, and all copies of this file, must be available in undamaged form. If this is not the case then you must manually resolve this, usually by copying the most recent and valid copy of the control file (*cntrl.dbf*) to the other locations, either sapdata0, sapdata1, and/or sapdata2.

- At least one member of each of the online redo log groups must be available in undamaged form. If this is not the case, you will have to first rebuild the necessary redo logs. SAPDBA determines whether any copies of the online redo logs are missing or damaged.

Tools Summary Comparison Chart

Below is a "at-a-glance" comparison chart of the three products we evaluated above:

Media Management Products Comparison Chart

Advantages	EBU w/ OBII	OBII Alone	SAP/R3 w/ OBII
1. Consistent backup and restore procedures; by having standardized and automated procedures can minimize human errors	X		X
2. High level administration	X	X	X
3. Uniform interface between raw devices and filesystems	X		X
4. Checksum capabilities and block header verification for database consistency check	X		
5. Online and offline database backup capabilities	X		X
6. Online restore capability; this allows the database to be operational for unaffected parts	X		SAP/R3 limitation
7. Auto-configuration capability where database configurations are automatically verified and comprehensive backups are guaranteed even when database structures have changed	X		X
8. Aggregated restore automatically retrieves the most current files, even across multiple backups, to ensure the shortest possible recovery time. No human intervention is required which can minimize errors	X		X
9. Point-in-time recovery capability where you only need to specify what point in time and restore will be performed automatically	X		X
10. Parallel processing capability where multiple data files or tablespaces can be backed up simultaneously on multiple devices in parallel; thus can reduce backup time	X	X	X
11. Multiplexing feature which allows simultaneous multiple data streams to be send to a single high speed device and demultiplexed automatically on restore	X		X
12. Provide scalability by taking advantage of parallel hardware (SMP, multiple storage devices, multiple disks)	X	X	X
13. Provides buffered I/O where all I/O is buffered in shared memory; thus forces all physical devices such as disks and tapes to operate at their peak rates	X	X	X
14. The Backup Catalog centralizes all configuration information and detailed backup histories for all databases	X		
15. Provides a Backup Catalog management utility which	X		

allows you to generate reports on target databases and their corresponding backup and restore operations			
16. Null block compression where null blocks will not be backed up	X		
17. Simple installation	X	X	X
18. Automatically issuing appropriate Oracle database online backup commands; transparent to DBA			X
19. Performs database recovery automatically after restore; transparent to DBA			X

Complete Versus Incomplete Database Recovery

Recovering the database from a media failure without losing any data is known as *complete recovery*. If some data is lost after recovering the database, it is known as *incomplete recovery*. Complete recovery should be implemented when you have all the required archived redo logs, backup data files of those damaged, and a current valid control file. Incomplete recovery should be implemented only if a complete recovery is not possible or if you want to restore the database to a point in time intentionally. When incomplete recovery is done, the database must be opened using the *alter database open resetlogs command* to reset redo log sequence to one again; however for SAP, this can be done for you via the SQLDBA utility. At this point, you must do a backup again because all your previous archived redo logs are no longer valid if want to perform complete recovery from this point on at a later time.

Database Recovery

As we mentioned earlier, that full data recovery is done in two steps. The first step is to have the necessary database files restored from the backup media and the second step is to perform database recovery to ensure that all the data is logically consistent to the point of failure or to a point in time. Database recovery usually needs to be done by issuing the appropriate Oracle database commands; however, there are some utilities that can do this automatically (such as SAP/R3's SQLDBA utility). There are three kinds of recovery that can be used depending on the kind of failure and on whether the database needs to be open during recovery. These are:

- RECOVER DATABASE
- RECOVER TABLESPACE
- RECOVER DATAFILE.

If a database is open while recovering a data file or a tablespace, this is an online recovery. If a database is closed when performing recovery, this is an offline recovery. Online recovery is applicable to data files and tablespaces. Offline recovery is applicable to databases and data files.

Below are more descriptions of these database recovery options.

Recover Database

Database recovery recovers all the data files in the database that are online. Both complete and incomplete recovery is possible. You can recover from an online or offline backup. You have to perform an offline recovery (database is mounted but not open) and all datafiles that need to be recovered should be online. You must have the following files: archived and/or online redo log files, current or backup control file, backup of data files (for the lost or damaged data files). The advantages of using RECOVER DATABASE are: the database is recovered in one step; incomplete recovery can be done; and data file belonging to SYSTEM tablespace can be recovered. The disadvantages are: the

database is inaccessible during recovery; and the recovery process can take a long time based on the amount of redo to be applied and frequency of backups.

When performing an incomplete recovery using RECOVER DATABASE command, you can have various options:

- recover database until cancel: this performs recovery until you issue the command cancel
- recover database until time 'YYYY-MM-DD:HH:MM:SS': this performs a point-in-time recovery
- recover database until change *integer* : this performs recovery up to a specific SCN (System Change Number: A crucial data structure that defines a committed version of the database at a precise moment in time. When a transaction commits, it is assigned an SCN that uniquely identifies the transaction.)
- recover database until cancel using backup controlfile: this is identical to the first command above; however, a backup controlfile is used to do recovery.

Recover Tablespace

Tablespace recovery can be only used to perform a complete recovery; incomplete recovery cannot be done. This command does media recovery on all the data files in the tablespace(s) listed. Oracle knows what data files constitute a tablespace. Oracle knows which tablespace contains what data only when the database is open. Thus, an online recovery can be done using either online or offline backup. This means the database must be open; however, the tablespace(s) that need to be recovered must be set offline. You must have the following files: archived redo and online redo logs, current control file, and backup of the data files that constitute the tablespace(s). The advantages of using RECOVER TABLESPACE command are: recovers all lost or damaged data files in the listed tablespace(s) in one step; it is faster than doing database recovery since redo log data doesn't need to be applied to all data files; other tablespaces in the database are accessible to users during recovery; and multiple SVRMGR or SQLDBA sessions can be used to recover tablespaces in parallel. The disadvantages are: You cannot perform online recovery for tablespaces that cannot be taken offline (such as SYSTEM); and incomplete recovery cannot be done.

Recover Datafile

Data file recovery can only be used when performing a complete recovery. Recover datafile recovers all the data files listed using either online or offline backup. Furthermore, it allows both online and offline recovery. When performing online recovery, the data file must be taken offline. You need the following files: archived redo and online redo logs, current control files, and backup of data files. The advantages of using recover datafile command are: offline or online recovery can be performed; multiple SVRMGR or SQLDBA sessions can be invoked to recover data files in parallel. The disadvantages are: data file must be taken offline for online recovery so you cannot perform online data file recovery for data files belonging to SYSTEM tablespace; and incomplete recovery cannot be done.

SUMMARY

Because data is critical in any business environment, it is vitally important that data is protected, whether it be stored in a small database or a large database (e.g., in terabytes). Due to the increasing size of those databases, managing and supporting them has become more complex while offline backup windows are constantly decreasing or non-existent. Therefore, it is essential to have a well planned backup/recovery strategy so your data can be safely protected and thus prevent/minimize downtime and data loss in your business operation.

In this paper, we have emphasized the importance of recovery and also provided a list of questions or issues that you must address first before designing your customized backup strategy. Many backup concepts, options, and tradeoffs have been provided for your information. Furthermore, various media management tools that can assist you in performing backup/restore and even database recovery are summarized in this paper. We hope the information provided in this paper can help you in your design or redesign of your backup/recovery strategy.

We conclude this paper with two at-a-glance matrices. The first matrix summarizes whether you can do a complete database recovery based on various backup scenarios using various media management tools. The second matrix presents various recovery scenarios and lists whether complete or incomplete recovery can be achieved using backup methods described in the first matrix. By having these matrices, we hope you can easily identify how protected your data is today and what you would like to use for future designs/adjustments of your backup/recovery strategy.

Backup Strategy Matrix

Backup Mode	Scenario	Backup w/ Unix System Utility	Backup w/ OmniBack II (OB2)	Backup w/ Oracle EBU & OB2	Backup w/ SAP/R3 Backup Util. & OB2
Online	Full DB (archive)	1 N/A	9 Complete	17 Complete	25 Complete
Online	Full DB (no-archive)	2 N/A	10 N/A	18 N/A	26 N/A
Online	Partial DB (archive)	3 N/A	11 Complete	19 Complete	27 Complete
Online	Partial DB (no-archive)	4 N/A	12 N/A	20 N/A	28 N/A
Offline	Full DB (archive)	5 Complete	13 Complete	21 Complete	29 Complete
Offline	Full DB (no-archive)	6 Incomplete	14 Incomplete	22 Incomplete	30 Incomplete
Offline	Partial DB (archive)	7 Complete	15 Complete	23 N/A	31 N/A
Offline	Partial Db (no-archive)	8 Incomplete	16 Incomplete	24 N/A	32 N/A

Recovery Scenarios & Corresponding Backup Methods

Recovery Scenario	Complete Recovery (Map # to Backup Strategy matrix)	Incomplete Recovery (Map # to Backup Strategy matrix)	Oracle Database Recovery Command Level
Full Database	5, 9, 13, 17, 21, 25, 29	6, 14, 22, 30	Recover database
System Tablespace	5, 9, 13, 17, 21, 25, 29, 7, 11, 15, 19, 27	6, 14, 22, 30, 8, 16	Recover database
Non-System Tablespace	5, 9, 13, 17, 21, 25, 29, 7, 11, 15, 19, 27	6, 14, 22, 30, 8, 16	Recover database; Recover tablespace
System Datafile	5, 9, 13, 17, 21, 25, 29, 7, 11, 15, 19, 27	6, 14, 22, 30, 8, 16	Recover database
Non-System Data file	5, 9, 13, 17, 21, 25, 29, 7, 11, 15, 19, 27	6, 14, 22, 30, 8, 16	Recover database; Recover tablespace; Recover datafile
Control File	No recovery needed if have mirrored control files on various disks	N/A - if all data files and log files are saved, you can recover the control file by creating a new control file or using the backup control file	Recover database using backup controlfile; Must open the database with "resetlogs" option. Must do a new backup!
Online Redo Log File	No recovery needed if there are multiplexed online redo logs on various disks and at least one is working	If not archived yet and have no multiplexed online redo logs, data loss will occur	Recover database is needed for an incomplete recovery
Archive Redo Log Files	17, 21, 25, 29 automatically backup archived redo logs files as part of the backup process	If archived redo log files are not backed up manually, then data loss will occur	N/A - will be applied by all three Oracle recovery commands when needed