Outage Prevention, Detection, And Repair

Lawrence To

Center of Expertise
Worldwide Customer Support
Oracle Corporation

October 18th, 1995

Research Analysts: Lawrence To

Technical Writer: Brian Quigley

Contributors: Sameer Patkar

Introduction

This report outlines plausible database outages and focuses on procedures needed to **prevent**, **detect** and **repair** these outages. Sections have several introductory paragraphs describing the significance of a database component and follow up with suggestions and recommendations to prevent, detect and repair the loss of the component. We will incorporate some of our own testing and times on a 5 GB database to exemplify these procedures and scenarios wherever we find that the procedures are generic to all systems. However, we recommend that testing should continue on the actual Production environment to fine tune the recovery procedures.

Ideally, proactive mechanisms are implemented to reduce the likelihood of outages but if they do occur, recovery should be automated as much as possible. Again, detection and repair mechanisms and steps will be detailed throughout this report. The customer needs to dedicate resources on the automation of these procedures.

Control files

Control files are key components to recovery. A control file that reflects the physical structure and current status of the database must always be available. The control file should be backed up at the completion of every structural change. For example, after adding a data file or log file and after the completion of a hot backup to the database, the control file should be backed up.

Because a control file is an unreadable binary file, Oracle provides a method to create a readable SQL script. Such a script is generated for a given control file by issuing an *alter database backup control file to trace* command. Multiple backup copies of control files from different times should be kept to enable point-in-time recovery. Oracle recommends that a backup control file be created after the completion of a hot backup or after a structural change to the database. In addition, backing up of the control file to a trace file should be done as frequently as backing up the control files. The trace files should have a naming standard which includes the timestamp. Trace files with and without the resetlog option should be created and kept readily available in the event the DBA needs to recreate the control file quickly in a recovery situation. This should be standard practice in all database environments.

At least three control files should be used. Each should be on different controllers and disks. Since loss of any control file will result in the database crashing, it is always advisable to have the control files hardware mirrored as well. Hardware striping, which does not give any performance gains since control files are small, should not be used. By using hardware striping, the probability of losing a control file due to disk failure increases dramatically because the control file is striped or distributed across several disks.

In the event of losing a control file and its mirrored copy due to user error or corruption, the DBA can easily restart the instance with the remaining valid control files. If all control files are lost, the DBA needs to startup nomount the instance, recreate the control file by running a valid *create control file* script, and open the instance.

Failure to write to any one of the control files will result in the database aborting or crashing. Oracle does recommend having multiple copies of the control file spread over multiple disks to reduce the probability of losing all control files due to corruption or disk crash. However, having several copies of the control files should not be the only protection, we recommend that the control files should be mirrored on the hardware level.

If you lose a single control file and have at least one good control file, we recommend that you

- shutdown abort if the instance is not down yet.
- check background trace files to determine which control file is corrupted.
- copy the good control file over the bad one or modify the init.ora to reflect only good control files.
- startup

If you lose all your control files, then you should issue the

- create controlfile command which is available for all oracle versions 6.0.34 and up.

The create control file command should already be in a script and can easily be created while the database is up by issuing an *alter database backup controlfile to trace.* The output from the resetlog option is dramatically different from the output without resetlogs (noresetlogs is default).

The noresetlogs option produces a script that does not contain the thread numbers of each group. This may be serious problem if you modify this script to contain the resetlog option. This will ignore the other threads.

Risks and disadvantages of the create controlfile command:

- Lost previous log history.
- Database specific details must be recreated by looking at the file headers.
- Data file information is obtained by data files. If the data files are corrupted or not from a single backup, then there may be problems.
- Certain optimizing SCN structures are ignored with a backup controlfile
- User error possibility is high when there is not a create controlfile script already in place.
- Check if you are in archivelog mode.

When should you backup your controlfile?

- After the completion of a hot backup
- After a database structural change (i.e. add data file, change sizes of files, add log files)
- After a tablespace status change (i.e. READ-ONLY to READ-WRITE or vice versa)
- After backing up your control file, you should also create or adjust any create controlfile scripts. Your control file is a key component in recovery. It should **always** represent the physical configuration and the file status of the time you open the database. In most situations, it is best to recover with the current control file.

What to check for when using the **backup controlfile** option during recovery?

Make sure that the contents of the controlfile reflect the physical structure of the database you are recovering to.

Alter database open resetlogs is needed when using backup controlfile. A backup is always recommended after resetlogs because the logs are cleared and the sequence numbers are reinitialized.

In the event that media recovery needs to be performed, the current control file should be used if it does reflect the physical structure of the database that you are planning to recover to. The current control file preserves crucial timestamp and stop SCN information which may optimize and ease recovery while using the "backup control file" syntax needs to ignore some timestamp information found in the control file. The "backup control file" syntax relinquishes control of recovery to the user.

Control Files	Preventive Steps	Detection Mechanisms
Loss of any control file will result in an outage.	1. Hardware mirroring 2. Redundant Controllers 3. Three or more control files on separate disks and controllers. 4. Control file backups nightly, after hot backups and after structural changes. 5. Control file create scripts created with *alter database backup controlfile to trace* command at the same interval. These scripts should be modified to fit the needs of the database. Documented and tested procedures in place to resolve these outages.	1. Parsing the alert.log and background trace files for errors. 2. Monitoring disks and controllers for failures

Types of Outage	Steps	Tested TTR	Early Detection	Error Detection
Loss of a single control file Notes: The time metrics do not include analysis of the trace files and modification to the init.ora file. These tests were performed on a 5 GB database)	Shutdown Instance (abort) Start Background processes Mount exclusive Crash Recovery Open DB TOTAL	0:01 mins 0:14 mins 0:07 mins 3:05 mins Total= 3:27 mins	No detection is possible until a checkpoint is performed.	The CKPT process if present, otherwise the LGWR process will generate the error stack OPIRIP, ORA-00447, ORA-00201, ORA-00202. Error number 202 reports the file effected. The full syntax is; **`ORA-00202: CONTROL FILE: 'file name'`**
Loss of all control files Notes: The time metric assume that the 'alter database backup controlfile to trace' command has been correctly edited and is available for execution.	Shutdown Instance (abort) Start Background processes Create Controlfile Media Recovery Archive Log Current Open DB (includes crash & other checks) TOTAL	0:01 mins 0:20 mins 0:25 mins 1:41 mins 1:21 mins 1:56 mins Total: 5:44 mins	No detection is possible until a checkpoint is performed.	The CKPT process if present, otherwise the LGWR process will generate the error stack OPIRIP, ORA-00447, ORA-00201, ORA-00202. Error number 202 reports the file effected. The full syntax is; **`ORA-00202: CONTROL FILE: 'file name'`**

Online redo logs

On-line redo logs are crucial for instance and media recovery. They contain the redo to roll the database forward and the undo needed to rollback any uncommitted transactions. The loss of a current on-line redo log or loss of an unarchived on-line redo log will result in an outage. An unarchived log cannot be dropped unless NOARCHIVELOG mode is enabled; unfortunately, in order to switch to this mode, one needs to shutdown and remount the instance.

To protect from the above outages, Oracle recommends Oracle multiplexing of the redo log files over disk mirroring if both techniques cannot be used. Oracle multiplexing (multiple members per group) protects against user error (e.g. an accidental delete) while hardware mirroring does not. Ideally, each member of the on-line redo group should be on different drives and different controllers and not striped across disks. Here is an example which contrasts OS mirroring of log files versus log file groups. If there is a single operator or configuration error to a mirrored log, the mirrored logs become corrupt and operations are stopped. In contrast, Oracle continues to run if only one member of the multiplexed on-line log is corrupted or destroyed. Thus, Oracle multiplexing of groups are more resilient. Although it requires allocating more on-line redo logs and adds a small performance overhead, Oracle recommends its use in place of mirroring.

For extreme cases where the instance aborts due to a background process dying, the customer may need to bring up the database and perform crash recovery within a small interval of time. This interval of time should reflect the checkpoint interval setting or checkpoint_timeout setting in your database. A checkpoint is the process of writing out dirty buffers from the SGA to the physical data files and updating the files on completion to synchronize them to a point in time. Smaller checkpoint intervals will enable crash recovery to complete faster although it imposes a performance overhead of writing dirty buffers and updating file headers. Oracle, however, always does a checkpoint on a log switch. A typical recommended setting is a checkpoint every 10 to 15 minutes.

Furthermore, the customer should monitor the frequency of checkpoints and archiver wait problems. Although the database is up and running, since it is hung due to checkpoint problems or archiver wait problems, the database is essentially not available. If the checkpoint initiated in a log is not completed before the log needs to be recycled, the database will temporarily freeze all user activity until the checkpoint is finished. Async io should be available and should be default. With Async io, this problem does not usually appear. If async io is not available, multiple database writers (DBWR) and a checkpoint process (CKPT) can reduce checkpoint wait errors but usually is only a temporary fix. However, if this happens frequently, it may be due to the size and number of log files being small relative to the rate of changes being made in the database. A similar situation arises when log writer waits on archiver to complete archiving of a redo log. Again, the solution is to add more log file groups or increase the size of the log file groups to allow the archiver process more time to do its job. One can also increase log_archive_buffer_size to increase the size of a single read from the on-line redo logs by archiver (arch). Multiple archivers are a common solution to archive waits. One can enable another “archiver” process by systematically issuing an ‘alter system archive log current’. Checkpoints should always be monitored to determine if activity predictions are accurate or changing. *Log_checkpoints_to_alert* is set to TRUE in the init.ora to write checkpoint messages to the alert.log; however, this may cause a flood of messages in the alert.log. Alternatively, the checkpoints can be monitored via querying V$SYSSTAT.

Oracle recommends that log file status be monitored very closely, especially for the STALE or INVALID status. The INVALID log file errors do appear in the alert.log as an IO error and would be detected by alert.log parsing. The STALE status indicates that the state of any pending write on that log is unknown or the log is not complete (a log being written to prior to a shutdown abort). This potential problem does not appear in the alert.log and can only be monitored by V$LOGFILE. If frequent STALE or INVALID status is found on a running database, it may be indicative of hardware problems.

Log files have the highest IO activity along with rollback segments. They should be placed on their own set of disks if possible and definitely separated from the archive files.

Standards:

- Naming convention should be established.
- Multiplexing should be mandatory if running Oracle7.
- Mirroring on the hardware level is extra protection.
- Redo logs (groups) should be isolated from other disk files.
- Redo log members should be the same size.
- Redo log should be sized to switch once every fifteen minutes or depending on your availability standards.

- Log checkpoint interval and log checkpoint timeout should be set.
- Sufficient redo logs (groups) should be created to prevent archiver induced
- file busy waits or checkpoint busy waits.

What do you do when you lose or discover a stale or invalid redo log member of a group?

- Investigate the problem and check if it is still accessible to oracle.
- Drop the invalid or stale member if problems still persist and recreate the log member somewhere else if possible. Stale logs can also be caused by switching a log. Stale implies that the log is not completely full or complete.

What if you lose the entire group that has been archived?

- Attempt to drop the group and recreate a new group.
- If this is successful, then initiate a hot backup because recovery can not skip missing logs.. If not, then this implies that this is the current or next log..
- You need to restore and perform incomplete database recovery to the log group preceding the crashed one **-** using **until cancel** or **until time** recovery.

What if you lose a group that has not been archived but is not the current or next group?

- One needs to shutdown the database.
- Startup mount, alter database noarchivelog, and drop the log group.
- Alter database archivelog and then startup. From testing, we sometimes see that if the archive log lossed has not checkpointed then the database will not allow you to startup with archivelog. One must startup with noarchivelog, shutdown, and then restart.
- Add subsequent groups.
- Strongly suggest running a hot backup at this time because recovery can not recovery pass missing logs.

What do you do if you lose the current online redo log?

- You need to restore and perform incomplete database recovery.

Multiplexing the logs would have the effect of LGWR multiplexing writes and ARCH multiplexing reads. Should one operation be lost, the affected instance would not be crippled. Using mirroring, a read or write could be lost only via a dual hardware or firmware failure; an operating system or mirroring bug; or, more likely, human error overwriting the partition. Here's another way to contrast OS mirroring of log files versus log file groups. If there is a single operator or configuration error to a mirrored log, the mirrored logs become corrupt and operations are stopped. In contrast, Oracle continues to run if only one member of the multiplex online log is corrupted or destroyed. Thus, Oracle multiplexing groups give more resilience. Although it requires allocating more online redo logs and adds a small performance overhead, we recommend its use in place of mirroring.

Online Redo Logs	Preventive Steps	Detection Mechanisms
Online redo logs are needed for recovery.	1. Oracle multiplexing of log groups 2. Hardware mirroring of logs 3. Never placed Online Redo logs on the same disks as archive logs and application data files 4. Make sure log switch rate is	1. check V$LOG for invalid or stale statuses 2. parse alert.log and background trace file for redo log errors 3. Always check V$LOG and V$LOGFILE to determine which log is current and which log has

	matched by DBWR activity and ARCH activity 5. Make sure log group size is acceptable to availability requirements 6. More than 3 log groups. 7. Implement multiple archivers.	been archived. 4. Monitor log switch rate and checkpoint rates. 5. Monitor if the archiver keeps up with the log writer.

Types of Outage	Steps	Tested TTR	Early Detection	Error Detection
Loss of one and not all of the online redo log members of a particular group	Drop log file member Add log file member (40 MB) TOTAL	0:01 mins 0:37 mins Total: 0:38 mins	No detection is possible until log switch is performed either exiting or entering the file.	The alert log records the following error stack when switching out of the redo log group. ORA-00316, ORA-00312, ORA-00321, ORA-00312. In addition the LGWR process will generate the same error stack Error number 312 reports the file effected. The full syntax is; **`ORA-00312: online log 1 thread 1: 'file name'`** **`The view, V$LOG, should be monitored for statuses of 'STALE' or 'INVALID'.`**
Loss of inactive archived redo log group Notes: 1. All recovery time are quite static except for the crash recovery. 2. The addition of new redo log group can be done afterwards.	Shutdown abort Startup mount Drop the problem redo log Alter database open (crash recovery) TOTAL	0:01 mins 0:30 mins 0:01 mins 3:00 mins Total: 3:32 mins	No detection is possible until the LGWR process write to the redo log	The LGWR process will generate the error stack OPIRIP, the general one ORA-00447 followed by more specific ones. In this test case, they are ORA-00313, ORA-00312 and ORA-07366. The failure of the LGWR will cause other background processes fail with some general error message produced and finally alert log will report a background process failure. (may not be the LGWR process)

Types of Outage	Steps	Tested TTR	Early Detection	Error Detection
Loss of an inactive redo log group that has not been archived Notes: 1. All recovery time are quite static except for the crash recovery. 2. The addition of new redo log group can be done afterwards.	Shutdown abort Startup mount Alter database noarchivelog Drop the problem redo log Alter database open (crash recovery) Shutdown normal Startup mount	0:01 mins 0:30 mins 0:01 mins 0:01 mins 3:00 mins 0:03 mins 0:30 mins 0:01 mins 0:05 mins	No detection until the ARCH process archive the redo log	The ARCH process will generate archival stoppage message - ARCH: Archival stopped, error occurred. Will continue retrying - followed by informative error messages (ORA-00255, ORA-00312) reporting the problem online redo log and more specific error message(s) telling the cause (in the test case, it is ORA-00286). The same set of error messages will also appear on the alert log file together with

3. Cannot drop the unarchived redo log without setting noarchivelog mode.(ORA-00350) 4. Cannot set archivelog after dropping the problem redo log group since instance recovery required.(ORA-00265)	Alter database archivelog Alter database open TOTAL	Total: 4:12 mins		archival stoppage message.

Other Online Redo Log Outages	**Detection**	**Steps**
Loss of the current online redo log group.	The LGWR process will generate the error stack OPIRIP, the general one ORA-00447 followed by more specific ones. In this test case, they are ORA-00313, ORA-00312 and ORA-07366. The failure of the LGWR will cause other background processes fail with some general error message produced and finally alert log will report a background process failure. (may not be the LGWR process). V$LOG and V$LOGFILE will indicate if this is the current log. If so, we must switch to CRF.	1. Restore and commence incomplete recovery. 2. Commence Disaster Recovery Plan
Silent Redo Log Corruption	Error (ORA 600[3020]) during application of archive log on the standby database site.	1. Rebuild standby database if there is one present. 2. Investigate Primary for any problems. Check alert.logs.
Internal Redo Corruption	ORA 600 [3020] implies that this change in the redo log is corrupted or inconsistent with the changes in the data block. All ORA 600 errors during application or writing of redo logs may be evidence to a corruption in the online redo log.	1. If it does not affect primary, then refresh or rebuild standby database. 2. If primary is down, commence disaster recovery plan

System tablespace

The system tablespace is the most crucial of all tablespaces because it contains the data dictionary tables and objects. The data files should be protected by redundant hardware and controllers. It is not advisable to stripe the system data files across too many disks since this increases the number of potential failure points. Loss of the system data file will result in database recovery or switching to a disaster recovery site.

The system tablespace should be configured and sized properly. By convention, no user objects or temporary segments are created in SYSTEM. This should be enforced using privileges and roles with periodic monitoring to catch any deviations. Basically, the RESOURCE role and UNLIMITED TABLESPACE privilege should not be granted to an user. A typical size of 30-50 MB is common for the system tablespace, while an additional 300 MB - 500 MB may be required if PL/SQL and/or stored procedures are used. Free space and fragmentation should be monitored.

Loss of a system data file:

- The file needs to be restored and rolled forward. The database will be unavailable.
- Or you can recreate the database.
- Or restore from backup and roll forward
- Or commence disaster recovery plan.

System Tablespace	Preventive Steps	Detection Mechanisms
System tablespace contains the data dictionary tables.	1. No non-data dictionary objects should be found in SYSTEM 2. Monitor Space usage 3. Hardware mirroring with redundant controllers 4. No one should have privileges to write to the system tablespace 5. No auditing. 6. Spare disks in case of disk failures.	1. Monitor threshold on space usage within system 2. Monitor for non-system objects found in system tablespace 3. Parse alert.log and background trace files for ORA 600 errors with your monitoring tool. 4. Monitor disk failures to check if it affects system data files.

System Tablespace Outages	Comments	Steps
	/Detection Mechanisms	
Lack of space or system tablespace fragmentation.	This should be monitored very closely and the monitor tool should ensure that space is sufficient in all their tablespaces. However, if this occurs, the customer needs to add a datafile.	1. add another data file
Loss of system tablespace.	Monitor tool should track disk failures and correlate with the corresponding data file.	1. restore and recover. 2. commence disaster recovery plan.
Corruption of data dictionary object.	ORA-1578 on a data dictionary object, and ORA 600 may be an indication of data dictionary corruption. These errors should be parsed from the alert.log by the monitoring tool.	1. restore and recover. 2. commence disaster recovery plan.

Rollback segment tablespace

Since rollback segment tablespaces typically experience the highest IO activity, RAID 1 is a good choice over RAID 5 to allow for sufficient redundancy without sacrificing too much performance.

Rollback segment configuration is important to allow for continual availability for all applications modifying data. Rollback segments should be configured to minimize space usage, to reduce undo contention, to distribute IO, and to minimize rollback segment maintenance due to application error. The optimum number of rollback segments will be determined by the peak transactional concurrency the database needs to support while the size is determined by the largest amount of rollback that a single transaction can generate The rollback segment tablespace should have ample free space to allow for rollback segment growth while the OPTIMAL parameter should be set on all rollback segments to reclaim space in case of an exceptionally large transaction extend the rollback segment pass average size.

If the system is limited for disk space, the system tablespace and rollback segment tablespace can coexist on the same disk units. Loss of any one of these data files will result in database recovery or resorting to a disaster recovery site. The rollback segments contain information to undo the changes of an aborted transaction and are also used for read consistency. Loss or corruption of an active rollback segment can result in serious data inconsistency. We cannot guarantee the integrity of the database if we lose an active rollback segment.

Loss of a rollback segment data file:

- Attempt to offline all online rollback segments within that tablespace.
- Attempt to drop the rollback segments within that tablespace. You may be force to create other rollback segments in a different tablespace because the database needs at least one active rollback segment other than SYSTEM rollback segment.
- Attempt to drop the tablespace. If this fails, then there are active transactions in the rollback segment that has not been applied to your "data" blocks.
- Call Oracle Support and see if they can workaround the issue. In most cases, you need to restore from backup and go through recovery.

Rollback segment with "needs recovery" status:

- A rollback segment is considered "corrupt" when it cannot rollback an active transaction. Depending on the objects involved and the extent of the "problem", you may be able to startup the database by just taking the segment name out of the init.ora file.
- You should also check if all data files are online. Often, this error can be resolved by onlining your data files and allowing Oracle access to apply the undo to the data files' data blocks.
- Set event 10015 in the init.ora file if the database cannot startup.

  event="10015 trace name context forever"
  It prints out the rollback segments it is recovering while it is rolling back active transactions during startup.
  At times, we can drop the object that needs the undo and the "needs recovery" status will go away because there is no undo to apply to existing objects.
  Call Support for further suggestions.
  If the rollback segment is indeed corrupted or the rollback segment is lost that contains active transaction, we then suggest restore and attempt database recover to bring the database back to a consistent state. Sometimes, recreation of the database is necessary if no other backup is available.

One can check XIDUSN from V$TRANSACTION to find the active rollback segments by comparing it with the SEGMENT_ID column in DBA_ROLLBACK_SEGS.

Rollback Tablespace	Preventive Steps	Detection Mechanisms
Rollback segments contain the undo needed to rollback any statement and for transaction consistency.	1. Configure and tune rollback segments to the proper size and number to accommodate for the highest TPS and for the largest DML and query. 2. Hardware Mirroring with redundant controllers. 3. No other objects should subside in the rollback segment tablespace. 4. Monitor for disk failures.	1. Monitor V$ROLLSTAT for threshold and indications that the rollback segment may need to reconfigured. 2. Parse alert.log and trace files for rollback segment errors. (ORA 600) 3. check sys.**dba_rollback_segs** view for any '**needs recovery'** status

Rollback Tablespace Outages or Rollback related problems.	Detection Mechanisms/Comments	Realistic/ Tested TTR	Steps
Cannot allocate extents or tablespace fragmentation. Rollback segment maintenance errors.	The customer should monitor any ORA 1538, 1551, 1552, 1553, 1554, 1555, 1556, 1557, 1558, 1559, 1562.	Database availability is not lost. Application errors may occur.	1. Add more space by adding a datafile. 2. Add larger rollback segments or reconfigure to accommodate for larger txn.
ORA-1555 or snapshot too old errors.	ORA 1555	Database availability is not lost. Application errors may occur.	1. A larger rollback segment is needed.
Rollback segment corruption.	ORA 600, 1555, 1578	commence disaster recovery plan	1. attempt to drop rollback segment. If not successful, then commence disaster recovery plan. 2. investigate why rollback segment got corrupted. check trace files and call support
Loss of a rollback segment data file that does not contain active transactions.	Monitor tool needs to detect disk failures and correlate OS files to Oracle files.	Several rollback segment tablespaces are available. Database availability is not lost. Application	1. Drop rollback segments in that data file and drop tablespace. 2. Recreate rollback segment tablespace

		errors may occur.	
Loss of rollback segment data files that currently being used.	Monitor tool needs to detect disk failures and correlate OS files to Oracle files. If we cannot offline and drop affected rollback segment tablespace, then there are active transactions.		1. Restore and recover 2. Commence Disaster Recovery Plan.

Temp tablespace

Although the loss of the temporary tablespaces will not result in losing access to data, the application, in particular operations that require sort space or temp space, may falter. Temporary tablespace should be hardware mirrored to protect from disk failure. At least two temporary tablespaces should be available while precreated *alter user* scripts are established to prevent system tablespace fragmentation or application failure in case of loss of a temporary tablespace. Furthermore, .the default *initial* and *next* extent sizes for TEMP, with a *pctincrease* of 0, should be set to ensure that the largest sort space requirement can be met without running out of extents (121 for 2K block size). The initial and next extents should be set greater than sort_area_size and larger than the average sort if it does not conflict with available memory requirements.

Multiple temporary tablespaces will be beneficial in distributing sort allocations for different users and improve the performance of concurrent sorts. Furthermore, if several index rebuilds have to occur due to corruption or maintenance, multiple temp tablespaces will allow the parallelism to restore and create the indexes quickly. For index rebuilds, TEMP space of up to twice the size of indexed data may need to be allocated. As this TEMP space may not be required during normal operation, it is useful to keep spare disk space.

Following the initial index building phase, temporary segments usage is limited to sort activity. For performance reasons, the sort_area_size setting should be high enough to keep disk sorts to a minimum assuming there is enough real memory available. All complex queries which may potentially require disk sorts will then write to the temp tablespace.

Temporary Tablespace	Preventive Steps	Detection Mechanisms
Temporary segments are used for sorting and temporary scratch work when the allotted space in memory is insufficient.	1. Temporary tablespaces should be hardware mirrored with redundant controllers. 2. Alter user scripts should be available to switch users to an alternative temporary tablespace. 3. Enough space should be	1. Monitor disks and controllers for failure.

	available if more is needed for the temp. 4. Test and measure the amount temp space needed as your application changes. 5. Extra disks should be available.	

Temporary Tablespace Outages or Temporary tablespace related problems.	**Detection Mechanisms/ Comments**	**Realistic/ Tested TTR**	**Steps**
Cannot allocate extents or tablespace fragmentation.	These errors do not appear in the alert.log; thus, it must be trapped within the application. ORA 1547, ORA 1562 during a sort operation indicates temp tablespace configuration problem.	The customer should configure the TEMP tablespace to avoid problems like this. To fix this problem once it is detected encompasses adding another data file or creating another tablespace.	1. add another data file. .
Losing temporary tablespace data file.			1. run alter user scripts to switch temporary tablespace usage for users. 2. create another temp tablespace if possible.

Types of Outage	**Steps**	**Early Detection**	**Error Detection**
Loss of temporary tablespace data file	Drop current temporary tablespace. Create another temporary tablespace. Alter user to use new temporary tablespace.	Monitor tool needs to detect disk failure and correspond OS files to Oracle files.	**`Monitor tool needs to detect disk failures and correspond OS files to Oracle files.`**

Loss of an application data file

All application data files should be hardware mirrored on different controllers and disks for additional resilience. More importantly, the application should have integrity checks and error checking mechanisms to prevent errors and to automatically recover in case of errors. Security policies and proper prevention techniques will be essential to avoid dropping, deleting or modifying critical data. We also recommend partitioning of data and implementing business rules and database integrity constraints to detect for application data inconsistencies.

Even with built in prevention, user errors, application bugs, hardware failures, etc still occur which will result in some type of recovery. There are several possible solutions to resolve these problems:

1. recreate the tablespace and its corresponding objects (only relevant for indices),
2. restore from backup and recover the tablespace or
3. implement object level recovery plan or
4. initiate disaster recovery plan.

Application Data	Preventive Steps	Detection Mechanisms
Index segments should only hold indexes.	1. Sufficient space should be allocated for growth of indices. 2. Hardware mirroring and redundant controllers are recommended. 3. Recreate index scripts should be pre-created. 4. Partitioned accordingly to functionality.	1. Monitor space usage and b*tree levels with analyze commands. 2. Monitor disk and IO errors 3. Parse and check trace files and alert.log for ORA 1578 that affects indices
Data segments should hold both tables and clusters.	Same as above. 5. Object level safeguards some be implemented if possible. (e.g. exports, unloader, snapshots, standby database, etc.)	Same as above.

Case 1: Loss of index data file

Tablespace recovery and object recovery are two options that are available. Tablespace recovery would entail restoring the affected tablespace from the hot backup and rolling forward. Object recovery consists of recreating the tablespace and recreating the indices. Precreated *create index* scripts (implemented in parallel or with degrees of parallelism) should be kept ready and ran concurrently touching different temporary tablespaces to expedite recovery. In addition, large temporary tablespaces should be made available to allow for successful index creation .

Although both recovery descriptions allow for full availability of the database and the tables, this is still a serious outage due to the slow performance.

Types of	Steps	Early Detection	Error Detection

Outage/ Recovery Scheme			
Loss of Index Data file/ Recreate Index Tablespace	Drop Index tablespace Create another index tablespace Create all indices in the tablespace in parallel.	**`Monitor tool should detect disk failure. If the failure is an index tablespace, automated steps are needed to recreate the indices.`**	**`Monitor tool should detect disk failures.`**
Loss of index data file/ Restore from Backup and Recover	Offline tablespace. Restore from hot backup. Set autorecovery on. Bring up recovery processes. Alter database recover automatic tablespace. Alter tablespace online.	**`Monitor tool should detect disk failure. If the failure is an index tablespace, automated steps are needed to be restore and recover.`**	**`Monitor tool should detect disk failures.`**
Loss of index data file/ Switch to disaster recovery plan.	Switch to standby database or replicated database.	Monitor tool needs to detect disk failure and correspond OS files to Oracle files.	**`Monitor tool needs to detect disk failures and correspond OS files to Oracle files.`**

Case 2: Loss of application table datafiles

Hopefully, the customer does implement some type of object level backup and recovery strategy to reduce MTTR. If they do not, they must either restore from backup and recover or implement their disaster recovery plan.

Object level recovery only works when there are mechanisms within their application to resynchronize from a determined point of time. When the subset has been restored, then the subset should be able to resynchronized with the rest of the database via the application.

The database should be partitioned in such a way that recovery can occur to subsets of the database while other functionality is not impeded.

Types of Outage/ Recovery Plan	Steps	Early Detection	Error Detection
Loss of Data File: 1. Recreate Table Tablespace	Only applicable if exports or unloads of the tables are taken. Only applicable if there is an object level recovery plan. (snapshots, unloader, replicated database, etc)	**`Monitor tool should detect disk failure. If the failure is an index tablespace, automated steps are needed to be restore and recover.`**	**`IO errors or ORA 1100s errors.`**
Loss of Data File:	Offline tablespace. Restore from hot backup	**`Monitor tool should detect disk failure.`**	**`Monitor tool should detect`**

2. Restore from Backup and Recover	Set autorecovery on. Bring up recovery processes. Alter database recover automatic tablespace. Alter tablespace online.	**`If the failure is an index tablespace, automated steps are needed to be restore and recover.`**	**`disk failures.`**
Loss of Data File: 3. Commence disaster recovery plan	Only applicable if there is a standby database or a replicated database. This can also be accomplished by breaking three-way mirrors to maintain hot backups on site.	Monitor tool needs to detect disk failure and correspond OS files to Oracle files.	**`Monitor tool needs to detect disk failures and correspond OS files to Oracle files.`**

Types of Outage	Steps	Early Detection	Error Detection
Cannot allocate extents or tablespace fragmentation	Add datafile	Monitor Space Usage and fragmentation of tablespaces.	**`ORA-1547 or other error traps need to be trapped within the application`**

Loss of Index due to user error or corruption.	Drop index.
	Create index.
		`Monitor alert.log for ORA-1578 errors that affect index data.`
		`Periodic index validation with the analyze command.`
			`ORA-1578 while using or validating an index.`

Single Table Loss or Corruption of Single Table.	Object level recovery plan.
	Restore and recover.
		`Monitor alert.log for ORA-1578 errors that affect table data.`
		`Table can be analyze to check for block corruptions.`
			`ORA-1578 while accessing or analyzing table data.`

Reorganize Table - Scheduled Outage.	Drop / Create Unload/ Direct Load Use of Parallelism would be helpful.	Monitor and alarm when objects exceed more than 20 extents or some threshold.	**`ORA-1556 max extent error should be trapped within the application.`**
Reorganize Index - Scheduled Outage	Drop/ Create	Monitor and alarm when objects exceed more than 20 extents or some threshold.	**`ORA-1556 max extent error should be trapped within the application.`**

Read-only tablespaces

Read only tablespaces contain information that is static and the data is accessible only for viewing and not for modification. Read-only tablespaces **NEED** to be backed up once it becomes **READ -ONLY**. The recovery session will get an error if a file is **READ-ONLY** and a **backup controlfile** is being utilized for recovery.. All read-only tablespace files must be taken offline prior to incomplete recovery. This is what makes backing up READ-ONLY tablespace after changing to READ-ONLY very important. The moral of the story is that you need to restore the READ-ONLY tablespace from your previous backup and offline the read-only tablespace files prior to recovery with a backup controlfile. After opening the database, you then can online the READ-ONLY data files.

If you use the current control file, then recovery will complete successfully even if the tablespace changes from read-write to read-only through the redo logs. Read-only tablespace files are still online files and recovery will still apply changes to these files. Changing a file from read-write or read-only will cause some redo and during recovery, we will apply the redo. Thus the flags are changed.

Read-only tablespaces reduce the amount of work during checkpoints. Read only data files are not checkpointed and updated while only read-write files are. Recovery will ignore read-only files when possible; thus, streamlining recovery even further.

Although read-only files can be restored from backup and should not affect the rest of the database if it is loss, the application may depend on the data and may stop if the data is not accessible. In that case, we recommend that read-only data should also be mirrored to reduce the possibility of an outage due to media failure.

Read-Only Data	**Preventive Steps**	**Detection Mechanisms**
Index segments should only hold indexes.	1. Partition data according to functionality. 2. Hardware mirroring and redundant controllers	1. Monitor disks and for IO errors
Data segments should hold both tables and clusters.	1. Partition data according to functionality. 2. Hardware mirroring and redundant controllers	1. Monitor disks and for IO errors

Read-Only Data	**Steps**
Accidental drop of read-only data.	1. Restore from backup 2. commence Disaster Recovery Plan.
Loss of a read-only data file.	1. Restore from backup 2. commence Disaster Recovery Plan.

Archive logs

Archive log files provide the means to roll forward from a previous backup of the database. Since archive log files are applied in order, it is extremely important that all the archive log files be available in order to roll forward to the desired point in time. For that reason, Oracle recommends that archive files are also mirrored to protect from disk failure. For extra protection, copies of the archive files can be kept onsite and offsite in case of some large disaster. Furthermore, archive files from all threads in a parallel instance environment are required for media recovery.

One of the most common errors found on customer sites is a poorly organized archiving procedure which may be backing up incomplete archive files or not detecting corrupted tapes. A scripted methodology and well-tested tape management tool may be required to safeguard from corrupted archive files. Prior to backing up or copying an archive file, the procedure should check from V$LOG table that the corresponding on-line redo log has been completely archived. Archive files in Oracle7 may vary in size due to a log switch or a shutdown abort; thus , checking archive file sizes are not an adequate verification test.

Another common problem is that the archive destination becomes full or inadequate to support the number of archive files being created. First, adequate disk space should be provided to allow for sufficient time for the log files to be backed up to tape and to allow at least a day's worth of archive logs to be onsite or all archive logs starting from the previous database backup (cold or hot) to remain onsite. Second, multiple archival destinations should be created with automated switchover of destination when a certain threshold for available free space is reached. Finally, the archive procedure should dynamically monitor the archive destinations for lack of free space. The following procedure describes in detail how this may be accomplished.

First, two archive destinations or more need to be created with sufficient disk space. Archiving procedure should switch from one destination to another when the previous destination reaches a certain space usage threshold by issuing *alter system archive log to destination*. More scripts should be developed to periodically monitor free space of the current archive destination. When space usage reaches a certain threshold (75% or some value that will allow for at least two more logs in the current ARCH destination), the monitoring process should alter the archive destination, preferably the oldest archive destination. Concurrently, a separate process or shell script can be created to clean up and backup archive destinations to tape starting from the oldest to the most recent archive destination. The backup script should check that the archive files have been successfully archived and then back them up to tape. Removal of the oldest archive files should only occur prior to switching to the oldest archive destination. The archive destinations should always accommodate at least one day's worth of archive files onsite. If the archive files are onsite, the recovery time will be reduced by the time needed to restore from tape backup.

Since compression is usually done, we suggest that the archive files are thoroughly tested to ensure that the compression utility is not inadvertently corrupting the archive files or backups. This may be done by comparing checksums of the of the source and tape backed up files or restoring from this backup and checking if the database starts up.

In the event of incomplete or complete database recovery, caution must be exercised to prevent confusion between the current set of archive files and the archive files from the previous version of the database. For example, after successful incomplete recovery, the archive log sequence numbers will reset to one. Therefore, new archive files will be created with sequence number starting from one and up which should not be confused with the older archive logs with the same number. Another example is restoring from a old cold backup and starting up without any roll forward media recovery. The archive logs generated from that point will begin with the number that was stored in the control file. Again, it is necessary to distinguish between the current database's logs and the previous versions of those logs.

What if you lose an archive log or set of logs?

- A complete backup of the database is necessary and should be accomplish as soon as possible: cold backup or hot backup.

Archive Log Files	Preventive Steps	Detection Mechanisms
Archive log files are required for media recovery and for standby databases.	1. Scripted procedure to archive to the archive destination and to archive to tape. 2. Tape management tool with error checking mechanisms should be utilized. 3. Hardware mirroring of archive files. 4. Two sets of backups.	1. Error checking for successful completion of archiving from database to tape 2. Space usage of archive destination needs to be monitored.

Archive Log Files and Archivelog	Detection Mechanisms	Steps
Archiver stuck due to lack of free space in the archive destination.	Monitor free space in the archive destination and alarm when close to reasonable threshold (70%)	1. Free up space in the archive destination . 2. Switch to different archived destination.
Loss of archive file(s) due to media failure or user error.	Monitor tool should check for disk failures. Alerts should be available when media failure affects archive logs. When this occurs, automatic procedures should switch to another archive destination.	1. Backup primary. 2. Refresh standby database if one exists.
Archiver can not write to the archive destination due to media failure.	Monitor tool should check for disk failures. Alerts should be available when media failure affects archive logs. When this occurs, automatic procedures should switch to another archive destination.	1. Switch archive destination.
Archive logs are not successfully backed up.	Checksums must be in place during the backup. Investigate problem and retry.	If backup is not successful because of a bad archive log file, one needs to backup Primary and recreate the standby site.
Archiver hangs because someone deactivates archivelog or disable archiving.	This should be checked before every open command. The *archive log list* command will	shutdown database startup mount; alter database archivelog alter database open.

Tape Management

A crucial step to backup and recovery is the efficiency of the tape management system. It must be capable to backup and restore at a rate that meets availability needs. For example, if the MTTR requirements allows for only a three hour outage and the only backups are on tape, it is crucial that the tape restore rates are adequate to allow the system administrator and DBA to recover within requirements. If not, then a copy of the backups may need to remain on disk.

A good reporting tool and error checking mechanism should be built on top of the tape management system or tool. This will be necessary to catch any errors during the backup/restore process. It will also be helpful for capacity planning.

Tape Management System	Preventive Steps	Detection Mechanisms
All backups eventually go to tape.	1. Error checking mechanisms and verification of backups. 2. Two backup sites.	1. Error checking via monitoring the tape backup logs.

Tape Management System	Steps
Failure to archived to tape due to tape errors.	1. Correct tape problems. 2. Backup to tape.
Checksum errors found during archiving to tape.	1. Hot backup 2. Check Standby database 3. Investigate tape errors.
Tape management utility failures due to media failure, software error or user error.	1. Failover to secondary backup site.

Miscellaneous

Machine clock change

Machine clock changes can affect recovery. Oracle stores both timestamps as well as SCNs for every committed transaction.. When recovering to a specific time, Oracle may stop recovery sooner if the system clock was pushed back. This may lead to unpleasant results in both your database as well as your distributed database environment.

By the way, Oracle does group commits which implies that one to many transactions will have the same timestamp. Oracle does guarantee that each committed transaction will have its own SCN. When doing time based recovery, Oracle completes after finding a timestamp higher than the requested timestamp.

Incomplete backup or Using backup controlfile

You always need to **alter database open resetlogs** after recovery is complete.

- A backup is necessary after any resetlogs.
- Log sequence numbers are reset.
- Previous backups can not rolled pass this point.
- We are not permitted to cross the RESETLOGS barrier using either a new or old controlfile.

alter database create datafile

- You need to have all the archive logs since the creation of the data file.
- Controlfile needs to have data file information.
- Full recovery has to be completed.