8i Hot Standby Databases - Usages, Implementation and Management

Stuart Speers
Senior DBA
IBM

Introduction

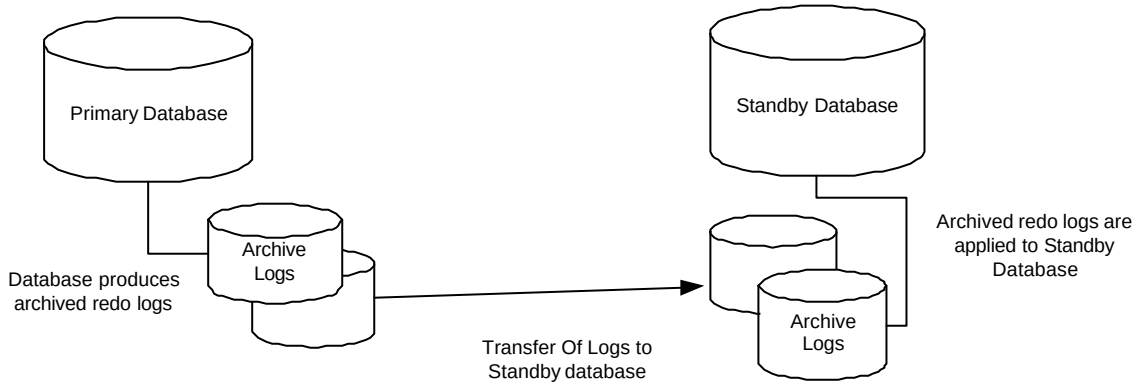
As most businesses develop and build their entire company around mission critical computer systems, database availability becomes one of the key components in keeping businesses profitable. Acronyms of RTO (Return To Operation) and MTTR (Mean Time To Recovery) are bandied around most IT shops these days and understanding what the options are for keeping critical applications available and which option is the best fit for an application is a time consuming process.

Oracle's Hot Standby Database feature provides one potential solution to maintain uptime and protect against potential disaster. Oracle's Hot Standby database uses the existing logic of Oracle RDBMS recovery by applying archive logs to a remote copy of the database.

This paper discusses the scenarios that best fit the use of Hot Standby databases, it's shortfalls, configurations and management once the solution is implemented.

What Is A Hot Standby Database?

A hot standby database is a copy of an Oracle database (referred to as the primary database). The standby database is kept in sync with the primary database by applying the primary databases archived redo logs.



The Standby Database Process:

- Create a Hot or Cold backup including all archive logs of the primary database
- Restore the backup to a new location (recommended to be on a separate server, preferably at a separate site)
- Create a standby control file from the primary database
- Mount the database using the standby control file (placing the database in standby mode)
- Set-up a method of archive log transfer between the primary and standby databases
- Begin manual or automatic application of the archived redo logs from the primary database
- When and if failover is required, apply as many outstanding logs to the standby database as possible
- Activate the standby database, this effectively does a reset logs and opens the database.

- If temporary tablespaces are used, you will need to manually add the datafiles (v$tempfile)
- Backup the new primary database
- Begin the process of recreating a new standby database on a third server or back onto the original server and repeat the previous steps

Advantages / Disadvantages Of A Standby Database

Advantages

- Comes licensed with the Oracle RDBMS
- Practically no overhead placed on the primary server (archive log shipping and management)
- Data loss can be minimal
- Easily configured

Disadvantages

- Requires significant monitoring processes to verify the transfer and application of archive logs
- Potential cost of additional hardware

What Are Some Of The Possible Usages Of A Standby Database?

As well as being a mechanism for protection from a site disaster, standby databases have other uses. Detailed below are a few examples.

Disaster Prevention

The primary use of a standby database is to provide a means of quickly bringing a system back online if a disaster occurs. A disaster could be a complete site taken offline or a single server failure. With careful planning, minimal data loss occurs and MTTR (Mean Time To Recovery) is low.

Protection against data corruption

Hot standby databases can, to some degree, protect from data corruption. However, due to the fact that the archive logs are transferred from the primary host and usually applied in a timely manner to the standby database, any data corruption caused from human or application error has usually been applied to the standby database before the error is detected on the primary database. If protection from data corruption is the primary reason for the standby database, the application of archive logs can be delayed to allow time for errors to be detected. The additional time to apply any outstanding archive logs needs to taken into consideration in working through recovery time (MTTR).

Decision Support System

Another feature of a standby database is the ability to open the database in read only mode and allow access for reporting users to query the database. Under version 8i of Oracle, applying archived redo logs and read only mode are mutually exclusive. Version 9i of Oracle allows these two functions to coexist. It is generally recommended that if both disaster recovery and DSS functionality are required that two standby databases be created.

Efficient Recovery Option For System Maintenance - Application Of Patches, Software Upgrades

When major upgrades are being performed that have some degree of risk, for example, a new release of an application; the standby database can have the last log(s) applied to it before the upgrade starts and then left in

that state for the duration of the upgrade and user acceptance testing. If the upgrade needs to be backed out then the standby database can be activated and used as the primary database. This gives some other key benefits:

- The amount of scheduled time required in the upgrade plan for recovery, should the upgrade fail is reduced, therefore leaving more time for the upgrade itself.
- The standby database can be opened as read only and any discrepancies between the original production database and the upgraded database can be identified and hopefully resolved.
- The old production database that has been unsuccessfully upgraded can be kept online so that application vendors can resolve any issues with the upgrade.

Hot Standby Terminology

The following table describes some of the hot standby terminology:

Terminology	**Description**
Primary database	The production (live) database
Standby database	Copy of production database
Redo logs	Current transaction record of all changes being made to a database. Logs are used in cyclic fashion and can be kept (archived) for database recovery.
Archive logs	Once a redo log is full or switched the logs become archived redo logs
SQL*Net	Oracle proprietary software used to communicate with Oracle databases
Bandwidth	The capacity of a network connection between two points (in this case the primary and standby servers)
Latency	Any additional delay that distance between physical servers causes. In summary this is the speed of light. Very little can be done to improve latency
Log Switches	The process when the current online redo log is closed and the next defined redo log is opened for writing, this occurs in a cyclic fashion. Generally, log switches should occur no more than every five minutes (peak use)
Resetlogs	The command that forces Oracle to reset the SCN (sequential Change Number) across all datafiles. A reset logs operation renders historical backups useless. Resetlog operations mostly occur as part of a point and time database recovery where you tell Oracle to forget data changes at a specific time.
Failover	The process of activating the standby database (via resetlogs) and making it production.
Switchover	The process of moving between the primary and standby databases without performing a resetlogs
Roll forward	The process of applying archived redo logs to bring a database back to a consistent state.

Archive Log Transfer – What Are The Options

The key component of a standby database is the availability of all archive logs from the primary database to the standby database. Many options exist for archive log transfer between the primary and standby sites. Oracle 8i supplies an automated recovery option via Sqlnet. This has restrictions (see below), so many DBA's prefer using standard Unix utilities such as ftp, rcp and nfs. Shareware or purchased utilities are also available (for example ssh, rdist). The method chosen is dependant on the following things:

- Bandwidth and latency delays
- Security requirements
- Skill sets / personal preference
- Budget

Sqlnet

Oracle 8i provides the ability to use standard Sqlnet configuration to transfer archive logs between the Primary and Standby servers. This process automates both the transfers and application of archive logs. A listener is configured on the standby server to accept requests. Oracle spawns additional server processes on the standby server to manage and process the archive logs.

Configuration

- Configure Oracle listener on standby database to receive archive logs
- Configure second archive log destination on primary database using Sqlnet connect string (parameters log_archive_dest_n and log_archive_dest_state_n)
- Place standby database in auto-recovery mode

Negatives With This Approach

- No delay can be applied to the application of the archive logs to the standby database
- Potential for Sqlnet to become a significant bottleneck in the transfer of logs
- Requires significant monitoring
- Requires significant maintenance when archive log issues arise

Ftp

Using the standard ftp process to transfer archive logs between servers is a relatively efficient approach, especially when they are compressed prior to transfer.

Configuration

- Set-up ftp process following best security practices for FTP
- Write script to compile list of archive logs to transfer (always leaving any open files)
- Schedule ftp script to transfer logs at given interval (15 minutes for example)
- Write script to check for new archive logs on the standby database and apply

Negatives With This Approach

- Requires significant scripting
- Requires significant monitoring
- Potential security risk

Rcp

Using Unix's standard rcp (remote copy) utility, archive logs can be compressed and transferred efficiently.

Configuration

- Set-up rcp process following best security practices for rcp
- Write script to compile list of archive logs to transfer (always leaving any open files)
- Schedule rcp script to transfer logs at given interval (15 minutes for example)
- Write script to check for new archive logs on the standby database and apply

Negatives With This Approach

- Requires significant scripting
- Requires significant monitoring
- Potential security risk

Nfs

If bandwidth and the volume of archive logs permit, NFS (Network File System) can be used. A second archive log destination can be configured to write archive logs to a NFS mounted file system from the standby server to the primary server.

Configuration

- Configure and tune the NFS file system
- Configure second archive log destination for NFS file system
- Write script to check for new archive logs on the standby database and apply (ignoring any open files)

Negatives With This Approach

- Requires significant monitoring
- NFS performance may become a bottleneck.
- Requires significant testing (NFS can have issues)

Best Practice Configuration and Considerations

The following identifies key components that make up a secure and reliable standby database

Standards and Processes

Strict database standards improve the management, configuration and reliability of a standby database. Having naming standards for databases, directory structures (OFA - Oracle Flexible Architecture) and database parameters helps ensure that inconsistencies don't cause issues with running standby databases. Enforced source and change control for all standby database scripts and processes helps prevent human error.

Database Parameter Settings (initSID.ora)

Several database settings are required to be the same on the primary database and standby database:

- Compatible
- db_block_size
- log_archive_format

As stated earlier under the standards section, having database parameter files as standard as possible is best practice. Memory configurations should be the only parameters that may need to be different between the primary and standby databases.

If running the standby database on the same server as the primary, it is necessary to have the db_name set to a different value. This name is converted to hex via a logarithm when the database instance is started, this address is used to allocate real memory on the server. You may also need to use parameters to convert the name and location of datafiles and archive logs for the standby database (DB_FILE_STANDBY_NAME_CONVERT

LOG_FILE_STANDBY_NAME_CONVERT)

Backups

Oracle recommends backing up the standby database regularly as failure can still occur on the standby server. Though this is the case, depending on database size and backup windows available, primary backups are usually sufficient.

There is the possibility of using the standby database as the primary backup i.e. not backup the production database. This does complicate recovery and has significant risks around database corruption that may not have been detected but applied to the standby database (or nologging operations).

Bandwidth

Available bandwidth between the primary and standby sever is a critical component in the successful implementation of a standby database. If bandwidth is not calculated correctly this can cause significant bottlenecks which may cause the standby database to fall behind in the application of archive logs.

When calculating bandwidth requirements the following factors should be considered:

- What process is going to be used for creating the standby database i.e. will the primary database be remote copied between servers, or will the primary database be backed up to tape and then restored on the standby server?
- Volume of archive logs produced by the primary database that needs to be transferred to the standby server
- Latency may also be a factor to consider if primary and standby servers are physically separated.

Exports of read only db to /dev/null

Oracle recommends checking the database for corruption on a regular basis. The best approach is to open the standby database in read only mode and perform a full export. This will effectively read every database block. To improve performance and reduce disk requirements for the export, use the puesdo device /dev/null, therefore no export file is physically created.

Planned failover tests

It is vital that the standby database solution is regularly checked and practiced. By performing scheduled failover and allowing users to connect to the standby database for a period, you gain the following:

- Verification that the application connects correctly to the standby database
- Practice for staff who will need, if required, a clear understanding of all processes for successful transfers
- Amendments to documentation

Note: In this testing mode, you may prefer to use the graceful switchover (discussed later) to ensure no data loss occurs.

Standby Server Capacity

In an ideal world the standby server would be of the same configuration as the primary server. This is often not the case due to budget constraints. Standby database servers are usually sized to only cope with the mission critical business requirements, with more focus on rebuilding the production environment as soon as possible after failover. Using a non-production server to act as the standby server is often a good compromise. When failover occurs the normal users of the standby server (e.g. developers) stop work to allow production users to use the resources on the server. When using a non-dedicated standby server, consider the following:

- Implementation across different domain names
- Enforced change control and strict security
- Maintenance of consistent standby server capacity (compared to the primary server).

Regular Refreshes

Undetected errors on the standby database may render the standby database useless. To ensure that the standby database is in the correct state, scheduled refreshes should occur. Doing scheduled refreshes in conjunction with the regular export of the standby database to /dev/null (discussed earlier) and the planned failover tests provide a robust solution.

Delay in applying logs

If you decide to manually transfer archive logs to the standby database, a delay can be implemented in the application of the archive logs. This can allow some time to detect and stop data errors from being propagated to the standby database. Consider the following when calculating delays in applying archive logs:

- Number and size of the archive logs produced
- Disk storage for archive logs not yet applied to the standby database
- Return to operation timing
- A 20M log file takes approximately 40 seconds to apply (this needs to be tested on the appropriate hardware)

Log Switches

To ensure redo log switches occur at regular intervals, a process should be implemented to force the database to switch logs. This can either be a DBA job function or a cron job.

NOLOGGING and UNRECOVERABLE Clauses: What Effect Do These Have On A Standby Database

Oracle provides the ability to run certain operations without producing redo log information (unrecoverable clause). These operations generally produce a lot of redo log information, hence may take significant time to run, by using the unrecoverable option the time to run is greatly reduced. Creation of indexes is an example of this type of operation. Because no redo information is recorded the statement cannot be applied to the standby database, creating an inconsistency between the two databases. Nologging operations cause that object to be in an incorrect state, and the only option is to refresh the effected tablespace(s) on the standby database. Unrecoverable operations must be detected and corrected immediately and should be avoided in a hot standby configuration.

Directory Structures

To ease the management of the standby database, ensure that database directory structures are the same on the primary database and standby database. Oracle Flexible Architecture (OFA) is an example of naming standards.

Common Checks And Processes Required

Below is a summary of processes that need to have automated processes written:

- Detection of Nologging / Unrecoverable statements
- Reporting on the number of logs not applied to the standby database
- Checks for errors in /dev/null export
- Correct log transfer
- Collection of statistics on transfer rates and number of logs being transferred
- Automated log switching

What Is Graceful Switchover and Switchback?

Normally when failing over (activating the standby database) Oracle performs a resetlogs operation. This forces Oracle to reset the SCN (sequential change number) number back to one. After a resetlogs is performed it is impossible to roll forward (apply archive logs) through the time of the resetlogs command. This renders all historical backups useless, as Oracle can no longer apply archive logs in sequence because the sequence has been reset back to one. Customers requested the ability to failover without losing data, so Oracle have provided a process to switchover without data loss, but it is not officially supported and requires thorough testing for each configuration.

Graceful switchover is generally only useful for planned outages as the primary database must be cleanly shutdown and the online redo logs available to the standby server for copy (unless remote mirroring of online redo logs is implemented).

The following criteria must be meet to perform graceful switchover:

- Primary database online redo logs must be available to the standby database
- An up to date recreate control file script (generated from the primary database) must be available

Process For Graceful Switchover:

- Create a standby database as previously discussed
- Disconnect all users from the primary database
- At the time of switchover, switch logs on the primary database and apply all outstanding archive logs to the standby database
- Create a backup controlfile to trace (alter database backup controlfile to trace) edit and change any of the location of datafiles. Copy the new create control file script to the standby site.
- Shutdown the primary database
- Shutdown the standby database
- Copy the online redo logs to the standby database
- Run the recreate controlfile script
- Recover the standby database (recover database) and open

Activation Of The Standby Database - The Decision To Failover

Depending on the businesses requirements for RTO (Return To Operation), a choice must be made whether to automate failover or to rely on manual failover (human interaction). If automated failover is required, careful scripting must be implemented to ensure only real failures trigger failover. A five-minute network outage is not a valid reason to failover.

Failover is a management decision due to the fact that data loss will occur when the standby database is activated. A failure maybe repairable in a short time (disk failure for example) so recovering the primary database may have less risk than failing over.

Key factors determining whether or not to failover:

- Time for error to be repaired
- Time and day of week / month may play a factor
- Failover server capacity

So That's The Database What About The Application?

If the application has a component that resides with the database, application synchronization between the primary and standby servers may also be required. There are many options available for this, from standard Unix utilities to shareware and manufactured products. Below is an example of such synchronisation.

The application is a Unix based combination of C++ and shell scripts. The application writes log files and reports required to be available in a failover situation. A shareware product called "rdist" is used to keep all files and directories synchronised on an hourly basis. The application is stored under the following directory structure on the primary server, /app/fin_app. Rdist builds an initial list of all files to keep synchronised and does a once off copy of all of the files to the remote server using the same directory structure. For each refresh that occurs rdist compares all of the files between the primary and standby servers and copies only changed / updated files to the standby server. Monitoring of this process will need to be configured to ensure updates are successful.

Separate schedules can be created to accommodate the requirements of each directory. For example the log directory may need to be updated every 5 minutes.

Application Failover and Sqlnet

One thing not yet discussed is, how to change Sqlnet configuration when the standby database is activated. This primarily depends on how you have Sqlnet configured. I recommend the following configuration.

- An IP alias (cname) should be configured for each database on the server. This interface is started on the server where the database currently resides. (This effectively gives you the ability to telnet to the name of the database, you no longer need to worry about which host the server resides on)
- The IP alias should be registered in DNS
- Each database should run it's own listener with dedicated port number. The host name defined in the listener.ora should be set to the database name (being the above configured IP alias)
- The tnsnames.ora configuration should specify the IP alias name for the host, not the physical server name.

When failover is required, the listener and IP interface should be shutdown on the primary server and started on the standby server. Connections will now connect to the standby server instead of the primary server. Note: If the IP address changes as part of the failover (i.e. a different subnet for example) DNS may also need updating.

9i what's Available / what's Different

Under Oracle 9i release 2 (9.2) standby databases have been renamed to a product set call Data Guard. Data Guard provides a managed process for standby databases. It includes full-automated recovery (still via Sqlnet) with various options for tuning the performance of the transfer and application of archived redo logs. Monitoring for errors is still a manual task.

Key Enhancements

- Delay in applying archive logs now available via automated option
- Many configurable parameters available in Data Guard
- Graceful failover now supported and automated (with no data loss – requires remote mirroring of online redo logs)
- NOLOGING operations can be forced to still produce redo information (at the database level)
- Improved archive log management performance via new specific task background processes
- User friendly gui

Reference Papers: Available from Metalink (metalink.oracle.com)

The following papers have step-by-step examples for setting up a standby database including syntax.

Oracle8i Standby Database by Lawrence To

Graceful Switchover and Switchback by Lawrence To

About the Author

In an IT career spanning almost ten years, Stuart has gained business experience in the following New Zealand Industries; manufacturing, transportation, distribution, healthcare and research. Stuart's strength is the ability to apply Industry best practices to all facets of Information Technology, specialising in Oracle Database solutions.

Stuart prides himself on his strong business and customer focus.

Skills & Experiences

- Principal Consultant – Providing business consultancy, Unix, Oracle database and Financials administration services to Key New Zealand Health care providers, Insurers and Atmospheric research Industries
- Oracle Architect – Providing Oracle database strategic direction to a leading New Zealand Airline
- Senior Consultant – Providing Senior Unix and Oracle systems administration skills to Major New Zealand fruit exporter and Meat manufacturer.
- Senior Database Administrator – Providing Unix and Oracle database and financial systems support for a key New Zealand Airline.
- Senior Oracle Database Administrator – Providing Unix and Oracle database and financial systems support for a significant Food Manufacturer
- Stuart is an Oracle Certified Professional (OCP) in Database Administration.
- Stuart has extensive experience in running systems 24 hours a day 7 days a week.