Oracle 10g Data Guard

(*Summary Sheets*) v. 2.0 Draft

Edited By: Ahmed Baraka

Version History

Version	Date	Updates
1.0	Sept, 2006	Initial document.
2.0 B	June, 2010	Updating and correcting the information. Draft copy.

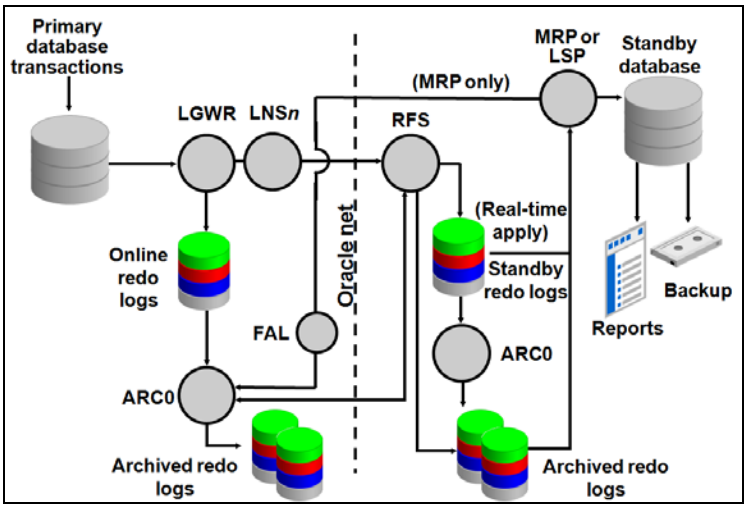
Document Purpose

This document is edited to be a quick hand book reference on using Oracle Data Guard in Oracle 10g Database. It is simply a summary of the Oracle documentation. However, many other resources were used a supporting materials.

1. Introduction to Oracle Data Guard

1.1 Data Guard Configurations and Architecture

- Primary Database
- Standby Databases
  - o Physical standby database
  - o Logical standby database



1.2 Data Guard Services

- Log Transport Services
- Log Apply Services: Redo Apply or SQL Apply
- Role Management Services

1.3 Data Guard Broker

The Data Guard broker is a distributed management framework that automates and centralizes the creation, maintenance, and monitoring of Data Guard configurations using either GUI (by intergration with EM) or CLI (in DGMGRL prompt).

1.4 Data Guard Protection Modes

- Maximum protection: no data loss. Primary database shuts down in case of transmission failure.
- Maximum availability: no data loss. Primary database tolerates one transmission failure.
- Maximum performance: a transaction is committed when its redo entries are written to local redo log.

1.5 Data Guard and Complementary Technologies

- Oracle Real Application Clusters (RAC)
- Flashback Database
- Recovery Manager (RMAN)

1.6 Summary of Data Guard Benefits

- Disaster recovery, data protection, and high availability.
- Complete data protection
- Efficient use of system resources
- Automatic gap detection and resolution
- Centralized and simple management
- Integration with Oracle Database

2. Getting Started with Data Guard

2.1 Standby Database Types

2.1.1 Physical Standby Databases

It works in two modes: Redo Apply (the database cannot be opened while redo is being applied), Open Read-Only and Open read/write.

Although the physical standby database cannot perform both Redo Apply and be opened in read-only mode at the same time, you can switch between them.

Benefits of a Physical Standby Database

- Efficient disaster recovery and high availability
- Data protection
- Reduction in primary database workload Performance

2.1.2 Logical Standby Databases

The logical standby database can be used concurrently for data protection and reporting operations. It has some restrictions on datatypes, types of tables, and types of DDL and DML operations.

Benefits of a Logical Standby Database

- It has same benefits as in a physical standby database.
- Efficient use of standby hardware resources.
- Reduction in primary database workload

2.2 User Interfaces for Administering Data Guard Configurations

- Oracle Enterprise Manager
- Command-line interface: SQL*Plus and some initialization parameters
- Data Guard broker command-line

2.3 Data Guard Operational Prerequisites

2.3.1 Hardware and Operating System Requirements

- The operating system and platform architecture on the primary and standby locations must be the same.
- The hardware (for example, the number of CPUs, memory size, storage configuration) can be different between the primary and standby systems.

2.3.2 Oracle Software Requirements

- The same release of Oracle Database Enterprise Edition must be installed on the primary database and all standby databases in a Data Guard configuration.
- The primary database must run in ARCHIVELOG mode.
- Keep the primary database in FORCE LOGGING mode as long as the standby database is required.
- Oracle recommends that when you set up Oracle Automatic Storage Management (ASM) and Oracle Managed Files (OMF) in a Data Guard configuration, set it up symmetrically on the primary and standby database.
- Consider setting the time zone for the primary and remote standby systems to be the same.

2.4 Standby Database Directory Structure Considerations

Standby database on the same system as the primary database

- You must use a different directory structure.
- You must set the `DB_UNIQUE_NAME` initialization parameter.
- You can either manually rename files or set up the `DB_FILE_NAME_CONVERT` and `LOG_FILE_NAME_CONVERT` initialization parameters on the standby database to automatically update the path names for primary database datafiles and archived redo log files and standby redo log files in the standby database control file.
- You must explicitly set up unique service names for the primary and standby databases with the `SERVICE_NAMES` initialization parameter.
- Disasters recovery is not supported by switchover is supported for planned maintenance.

Standby database on a separate system from the primary database but same directory structure

- You do not need to rename primary database files, archived redo log files, and standby redo log files in the standby database control file, although you can still do so if you want a new naming scheme.

Standby database on a separate system from the primary database and different directory structure

- For *physical* standby database, you can either manually rename files or set up the `DB_FILE_NAME_CONVERT` and `LOG_FILE_NAME_CONVERT` initialization parameters on the standby database to automatically update the path names for primary database datafiles and archived redo log files and standby redo log files in the standby database control file.

2.5 Online Redo Logs, Archived Redo Logs, and Standby Redo Logs

Redo data transmitted from the primary database is received by the *remote file server* (RFS) process on the standby system where the RFS process writes the redo data to archived log files or standby redo log files.

2.5.1 Online Redo Logs and Archived Redo Logs

- Online Redo Logs apply in the primary database and does ***not*** apply in the *physical* standby database.
- Primary databases, and both physical and logical standby databases each have an archived redo log.

2.5.2 Standby Redo Logs

- A standby redo log is similar in all ways to an online redo log, except that a standby redo log is used only when the database is running in the standby role to store redo data received from the primary database. It is highly recommended to configure.

3. Creating a Physical Standby Database

3.1 Preparing the Primary Database for Standby Database Creation

3.1.1 Enable Forced Logging

```
ALTER DATABASE FORCE LOGGING; -- may take long time
```

3.1.2 Create a Password File

Create a password file if one does not already exist.

The password for the SYS user must be identical on every system for redo data transmission to succeed.

3.1.3 Configure a Standby Redo Log

Determine the appropriate number of standby redo log file groups:

appropriate number of standby redo log file groups:

(maximum number of logfiles for each thread + 1) * maximum number of threads

Verify the values used for the MAXLOGFILES and MAXLOGMEMBERS clauses allows creating the required number of standby redo log group:

```
-- issue the command
alter database backup controlfile to trace;
-- check the generated file in:
SHOW PARAMETER user_dump_dest
```

Create standby redo log file groups:

- group number must be between 1 and MAXLOGFILES
- don't skip group numbers
- practically, a group may have two members or more

```
ALTER DATABASE ADD STANDBY LOGFILE GROUP 4
 ('E:\oracle\oradata\ora10g\standby_groups\log4a.rdo',
  'E:\oracle\oradata\ora10g\standby_groups\log4b.rdo') SIZE 50M;
ALTER DATABASE ADD STANDBY LOGFILE GROUP 5
 ('E:\oracle\oradata\ora10g\standby_groups\log5a.rdo',
  'E:\oracle\oradata\ora10g\standby_groups\log5b.rdo') SIZE 50M;
```

- Verify the standby redo log file groups were created.

```
SELECT GROUP#,THREAD#,SEQUENCE#,ARCHIVED,STATUS FROM V$STANDBY_LOG;
```

3.1.4 Setting Primary Database Initialization Parameters

Primary Database: Primary Role Initialization Parameters

```
# below is the primary db name
# for all standby databases, use the same name
DB_NAME=chicago
# should be unique for all databases
DB_UNIQUE_NAME=chicago
SERVICE_NAMES=chicago
# list the DB_UNIQUE_NAME of the primary and standby databases
# after a role transition, you may need to specify SEND, NOSEND, RECEIVE, or NORECEIVE
keywords
LOG_ARCHIVE_CONFIG='DG_CONFIG=(chicago,boston)'
# used by primary db for local archiving
LOG_ARCHIVE_DEST_1='LOCATION=/arch1/chicago
VALID_FOR=(ALL_LOGFILES,ALL_ROLES)DB_UNIQUE_NAME=chicago'
LOG_ARCHIVE_DEST_2='SERVICE=boston
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)DB_UNIQUE_NAME=boston'
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_STATE_2=ENABLE
```

```
# it can also take SHARED
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
LOG_ARCHIVE_FORMAT=%t_%s_%r.arc

Changing parameters commands:
  alter system set LOG_ARCHIVE_CONFIG='DG_CONFIG=(chicago,boston)'

  show parameter LOG_ARCHIVE_DEST_1
  alter system set LOG_ARCHIVE_DEST_1='LOCATION=/arch1/chicago
  VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=chicago'

  show parameter LOG_ARCHIVE_DEST_2
  alter system set LOG_ARCHIVE_DEST_2='SERVICE=boston LGWR ASYNC
  VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=boston'

  alter system set LOG_ARCHIVE_DEST_STATE_1=ENABLE;
  alter system set LOG_ARCHIVE_DEST_STATE_2=ENABLE;

  show parameter REMOTE_LOGIN_PASSWORDFILE
  alter system set REMOTE_LOGIN_PASSWORDFILE=exclusive scope=spfile;

  show parameter LOG_ARCHIVE_FORMAT
  alter system set LOG_ARCHIVE_FORMAT='arc_s%S_r%R_t%T.arc' scope=spfile;
```

If you specify the LGWR process to transmit redo data to both the local and remote destinations, also include the NET_TIMEOUT attribute on the LOG_ARCHIVE_DEST_2 initialization parameter.

Note: check the other directory dependent parameters (like USER_DUMP_DEST) to make their values the same in the standby database.

Primary Database: Standby Role Initialization Parameters

```
FAL_SERVER=boston
FAL_CLIENT=chicago
# any matching pattern found in the file name will be converted
DB_FILE_NAME_CONVERT='boston','chicago'
LOG_FILE_NAME_CONVERT='/arch1/chicago/','/arch1/boston/'
STANDBY_FILE_MANAGEMENT=AUTO

the commands:
  alter system set FAL_SERVER='boston';
  alter system set FAL_CLIENT='chicago';
  show parameter DB_FILE_NAME_CONVERT
  alter system set DB_FILE_NAME_CONVERT='boston','chicago' scope=spfile;
  alter system set LOG_FILE_NAME_CONVERT='/arch1/chicago/','/arch1/boston/' scope=spfile;

  show parameter STANDBY_FILE_MANAGEMENT
  alter system set STANDBY_FILE_MANAGEMENT=AUTO;
```

3.1.4 Enable Archiving

```
SHUTDOWN IMMEDIATE
STARTUP MOUNT
ALTER DATABASE ARCHIVELOG;
```

3.2 Creating a Physical Standby Database

3.2.1 Create a Backup Copy of the Primary Database Datafiles

You can use any backup copy of the primary database to create the physical standby database, as long as you have the necessary archived redo log files to completely recover the database.

3.2.2 Create a Control File for the Standby Database

Create control files for standby database using the following commands:

```
STARTUP MOUNT;
```

```
ALTER DATABASE CREATE STANDBY CONTROLFILE AS '/tmp/boston.ctl';
ALTER DATABASE OPEN;
```

3.2.3 Prepare an Initialization Parameter File for the Standby Database

```
CREATE PFILE='/tmp/initboston.ora' FROM SPFILE;
```

Initialization Parameters for a Physical Standby Database

```
# parameters to modify are in bold:
DB_NAME=chicago
DB_UNIQUE_NAME=boston
SERVICE_NAMES=boston
LOG_ARCHIVE_CONFIG='DG_CONFIG=(chicago,boston)'
CONTROL_FILES='/arch1/boston/control1.ctl', '/arch2/boston/control2.ctl'
DB_FILE_NAME_CONVERT='chicago','boston'
LOG_FILE_NAME_CONVERT='/arch1/boston/','/arch1/chicago/'
LOG_ARCHIVE_FORMAT=log%t_%s_%r.arc
LOG_ARCHIVE_DEST_1=
'LOCATION=/arch1/boston/
VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
# this parameter does not take effect
#  when the db is on standby mode
LOG_ARCHIVE_DEST_2='SERVICE=chicago LGWR ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_STATE_2=ENABLE
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
STANDBY_FILE_MANAGEMENT=AUTO
# used when the primary and standby
#  databases reside on the same system.
INSTANCE_NAME=boston
# Oracle Net service name of the FAL server
FAL_SERVER=chicago
FAL_CLIENT=boston
```

- If you specify the LGWR process to transmit redo data to both the local and remote destinations, also include the NET_TIMEOUT attribute on the LOG_ARCHIVE_DEST_2 initialization parameter.
- Ensure the COMPATIBLE initialization parameter is set to the same value on both the primary and standby databases.
- Modify directory-based parameters.

3.2.4 Copy Files from the Primary System to the Standby System

- Copy the following files to the standby system:
  - Backup datafiles created in Section 3.2.1
  - Standby control file created in Section 3.2.2
  - Initialization parameter file created in Section 3.2.3

3.2.5 Set Up the Environment to Support the Standby Database

- `WINNT> oradim -NEW -SID boston -INTPWD password -STARTMODE manual`
- In Non-Windows platform, create a password file, and set the password for the SYS user to the same password used by the SYS user on the primary database.

```
orapwd file=$ORACLE_HOME/dbs/orapwposton password=<password> entries=<users>
```

Note: In Windows, password file is automatically created by the previous step.

- Configure listeners for the primary and standby databases.
- On **both the primary and standby systems**, use *Oracle Net Manager* to create a network service name for the primary and standby databases that will be used by log transport services. The connect descriptor must also specify that a dedicated server be used.

```
ORA10G2 =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = srv02)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
```

```
      (SERVICE_NAME = ora10g2)
    )
  )

ORA10G =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = srv01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ora10g)
    )
  )
```

- Create SPFILE:

```
set ORACLE_SID=boston
conn sys as sysdba
create spfile from pfile='E:\temp\ora10g\initboston.ora';
```

- Create directories used by the parameters.

3.2.6 Start the Physical Standby Database

- A physical standby database must be in the mounted state (or open in read-only mode) to receive redo data.

```
STARTUP MOUNT;
```

Note: if you receive the error ORA-02778, it means some directories in the pfile aren't there. Check and create them.

- Start Redo Apply (must be issued after every startup):

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

- Test archival operations to the physical standby database:

```
ALTER SYSTEM SWITCH LOGFILE; -- on primary db
```

3.2.7 Verify the Physical Standby Database Is Performing Properly

On the standby:

```
SELECT SEQUENCE#, FIRST_TIME, NEXT_TIME FROM V$ARCHIVED_LOG ORDER BY SEQUENCE#;
```

On the primary:

```
ALTER SYSTEM ARCHIVE LOG CURRENT;
```

Then on the standby:

```
SELECT SEQUENCE#, FIRST_TIME, NEXT_TIME FROM V$ARCHIVED_LOG ORDER BY SEQUENCE#;
```

also verify new archived redo log files were applied:

```
SELECT SEQUENCE#,APPLIED FROM V$ARCHIVED_LOG ORDER BY SEQUENCE#;
```

3.3 Further Preparations

You can further apply the following:

- In both the primary and standby databases, set CONTROL_FILE_RECORD_KEEP_TIME. The range of values for this parameter is 0 to 365 days. The default value is 7 days.

```
show parameter CONTROL_FILE_RECORD_KEEP_TIME
alter system set CONTROL_FILE_RECORD_KEEP_TIME=30 ;
```

- Upgrade the data protection mode. It is in the maximum performance mode by default.
- You can enable Flashback Database on the primary database, the standby database, or both.

4. Creating a Logical Standby Database

4.1 Preparing for Logical Standby Database Creation

4.1.1 Determine Support for Datatypes and Storage Attributes for Tables

Unsupported Datatypes:

- BFILE
- ROWID
- UROWID

Unsupported user-defined types:

- object types REFs
- varrays
- nested tables
- XMLType

Unsupported Tables, Sequences, and Views

- Most schemas that ship with the Oracle database are skipped by SQL Apply
- Tables with unsupported datatypes
- Tables using table compression

Changes made to unsupported datatypes, table, sequences, or views on the primary database will not be propagated to the logical standby database and no error message will be returned.

```
SELECT DISTINCT OWNER,TABLE_NAME FROM DBA_LOGSTDBY_UNSUPPORTED ORDER BY OWNER,TABLE_NAME;
```

```
SELECT COLUMN_NAME,DATA_TYPE FROM DBA_LOGSTDBY_UNSUPPORTED WHERE OWNER='OE' AND TABLE_NAME =
'CUSTOMERS';
```

Skipped SQL Statements on a Logical Standby Database

```
ALTER DATABASE
ALTER SESSION
ALTER MATERIALIZED VIEW
ALTER MATERIALIZED VIEW LOG
ALTER SYSTEM
CREATE CONTROL FILE
CREATE DATABASE
CREATE DATABASE LINK
CREATE PFILE FROM SPFILE
CREATE SCHEMA AUTHORIZATION
CREATE MATERIALIZED VIEW
CREATE MATERIALIZED VIEW LOG
CREATE SPFILE FROM PFILE
DROP DATABASE LINK
DROP MATERIALIZED VIEW
DROP MATERIALIZED VIEW LOG
EXPLAIN
LOCK TABLE
SET CONSTRAINTS
SET ROLE
SET TRANSACTION
```

DBMS_JOB Support

Specific support for DBMS_JOB has been provided. Job execution is suspended on a logical standby database and jobs cannot be scheduled directly on the standby database. However, jobs submitted on the primary database are replicated in the standby database. In the event of a switchover or failover, jobs scheduled on the original primary database will automatically begin running on the new primary database.

4.1.2 Ensure Table Rows in the Primary Database Can Be Uniquely Identified

The following query displays a list of tables that SQL Apply might not be able to uniquely identify:

```
SELECT OWNER, TABLE_NAME,BAD_COLUMN FROM DBA_LOGSTDBY_NOT_UNIQUE WHERE TABLE_NAME NOT IN
(SELECT TABLE_NAME FROM DBA_LOGSTDBY_UNSUPPORTED);
```

Note: Some of the tables displayed in the DBA_LOGSTDBY_NOT_UNIQUE view can still be supported because supplemental logging adds information that uniquely identifies the row containing the redo data.

If your application ensures the rows in a table are unique, you can create a disabled primary key RELY constraint on the table.

```
ALTER TABLE mytab ADD PRIMARY KEY (id, name) RELY DISABLE;
```

Note: if data in constraint columns are not unique, Redo Apply will fail.

Note: To improve the performance of SQL Apply, add an index to the columns that uniquely identify the row on the logical standby database.

4.2 Creating a Logical Standby Database

4.2.1 Create a Physical Standby Database

You create a logical standby database by first creating a physical standby database and then transitioning it into a logical standby database.

4.2.2 Stop Redo Apply on the Physical Standby Database

```
ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
```

Note: If the database is a RAC, then you must first stop all RAC instances except one before issuing the statement.

4.2.3 Prepare the Primary Database to Support a Logical Standby Database

4.2.3.1 Prepare the Primary Database for Role Transitions

Add the parameter:

```
-- ignored when db in primary role
LOG_ARCHIVE_DEST_3='LOCATION=/arch2/chicago/
VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE) DB_UNIQUE_NAME=chicago'
LOG_ARCHIVE_DEST_STATE_3=ENABLE
```

Commands to issue:

```
-- create the directory here
alter system set LOG_ARCHIVE_DEST_3='LOCATION=/arch2/chicago/
VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE) DB_UNIQUE_NAME=chicago' scope=both;
alter system set LOG_ARCHIVE_DEST_STATE_3=ENABLE scope=both;
```

4.2.3.2 Build a Dictionary in the Redo Data

As part of building LogMiner Multiversioned Data Dictionary, supplemental logging is automatically set up to log primary key and unique-constraint/index columns.

```
-- the following command waits for all existing transactions to complete:
EXECUTE DBMS_LOGSTDBY.BUILD;

-- set the following parameter in both primary and standby dbs:
show parameter UNDO_RETENTION
alter system set UNDO_RETENTION=3600;
```

4.2.4 Transitioning the Physical Standby to a Logical Standby Database

4.2.4.1 Convert to a Logical Standby Database

-- in the standby db (boston):

```
shutdown immediate
startup mount exclusive
ALTER DATABASE RECOVER TO LOGICAL STANDBY boston;
```

Note: If you're using pfile, you may get the error ORA-16254. Shutdown the db, change the db_name in the pfile and try again.

Note: The statement waits, applying redo data until the LogMiner dictionary is found in the log files. This may take several minutes. If a dictionary build is not successfully performed on the primary database, this command will never complete. You can cancel the SQL statement by issuing the `ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL` statement from another SQL session.

4.2.4.2 Create a New Password File

- If db_name changed in the step above, you must re-create the password file.

```
orapwd file=$ORACLE_HOME/dbs/orapwposton password=<password> entries=<users>
```

4.2.4.3 Adjust Initialization Parameters for the Logical Standby Database

```
-- in db2:
SHUTDOWN
STARTUP MOUNT

show parameter LOG_ARCHIVE_DEST_1
alter system set
LOG_ARCHIVE_DEST_1='LOCATION=/arch1/boston/ VALID_FOR=(ONLINE_LOGFILES,ALL_ROLES)
DB_UNIQUE_NAME=boston'
scope=both;

show parameter LOG_ARCHIVE_DEST_2
alter system set
LOG_ARCHIVE_DEST_2='SERVICE=chicago LGWR ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=chicago'
scope=both;

show parameter LOG_ARCHIVE_DEST_3
-- create the directory here
alter system set
LOG_ARCHIVE_DEST_3='LOCATION=/arch2/boston/ VALID_FOR=(STANDBY_LOGFILES,STANDBY_ROLE)
DB_UNIQUE_NAME=boston'
scope=both;

alter system set LOG_ARCHIVE_DEST_STATE_1=ENABLE scope=both;
alter system set LOG_ARCHIVE_DEST_STATE_2=ENABLE scope=both;
alter system set LOG_ARCHIVE_DEST_STATE_3=ENABLE scope=both;
```

4.2.5 Open the Logical Standby Database

```
-- in the standby database db2
ALTER DATABASE OPEN RESETLOGS;

-- Create standby redo logs
-- determin the log group size
select bytes/1024/1024 mb from v$log ;

alter database add standby logfile group 4 ('/u01/oracle/oradata/db2/redo04.log') size 50M;
alter database add standby logfile group 5 ('/u01/oracle/oradata/db2/redo05.log') size 50M;
alter database add standby logfile group 6 ('/u01/oracle/oradata/db2/redo06.log') size 50M;
alter database add standby logfile group 7 ('/u01/oracle/oradata/db2/redo07.log') size 50M;

-- begin applying redo data to the logical standby
ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
```

4.2.6 Verify Operation of the Logical Standby Database

```
/* Monitoring Log File Archival Information */
-- Determine the status of redo log files (db1)
SELECT THREAD#, SEQUENCE#, ARCHIVED, STATUS FROM V$LOG;

-- Determine the most recent archived redo log file (db1)
SELECT MAX(SEQUENCE#), THREAD# FROM V$ARCHIVED_LOG GROUP BY THREAD#;

-- Determine the most recent archived redo log file at each destination (db1)
--  should be the same in all destinations
SELECT DESTINATION, STATUS, ARCHIVED_THREAD#, ARCHIVED_SEQ#
```

```
 FROM V$ARCHIVE_DEST_STATUS
 WHERE STATUS <> 'DEFERRED' AND STATUS <> 'INACTIVE';

-- Find out if an archived redo log file was not received at a particular site (db1)
select dest_id , dest_name from V$ARCHIVE_DEST
  where dest_name like '%_1' or dest_name like '%_2' ;

SELECT LOCAL.THREAD#, LOCAL.SEQUENCE#
FROM
  (SELECT THREAD#, SEQUENCE# FROM V$ARCHIVED_LOG WHERE DEST_ID=1) LOCAL
WHERE
LOCAL.SEQUENCE# NOT IN
   (SELECT SEQUENCE# FROM V$ARCHIVED_LOG WHERE DEST_ID=2 AND THREAD# = LOCAL.THREAD#);

-- Trace the progression of transmitted redo on the standby site
```
see Setting archive tracing.

4.2.7 Monitoring the Performance of Redo Transport Services

ARCn Process Wait Events

ARCH wait on ATTACH	All ARCn processes to spawn an RFS connection.
ARCH wait on SENDREQ	All ARCn processes to write the received redo data to disk as well as open and close the remote archived redo log files.
ARCH wait on DETACH	All ARCn processes to delete an RFS connection.

LGWR SYNC Wait Events

LGWR wait on LNS The LGWR process waiting to receive messages from the LNSn process.

LNS wait on ATTACH All network servers to spawn an RFS connection.

LNS wait on SENDREQ All network servers to write the received redo data to disk as well as open and close the remote archived redo log files.

LNS wait on DETACH All network servers to delete an RFS connection.

LGWR ASYNC Wait Events

LNS wait on DETACH All network servers to delete an RFS connection.

LNS wait on ATTACH All network servers to spawn an RFS connection.

LNS wait on SENDREQ All network servers to write the received redo data to disk as well as open and close the remote archived redo log files.

True ASYNC Control FileTXN Wait The LNSn process to get hold of the control file transaction during its lifetime.

True ASYNC Wait for ARCH log The LNSn process waiting to see the archived redo log (if the LNSn process is archiving a current log file and the log is switched out).

Waiting for ASYNC dest activation The LNSn process waiting for an inactive destination to become active.

True ASYNC log-end-of-file wait The LNSn process waiting for the next bit of redo after it has reached the logical end of file.

4.3 Further Preparations

- Upgrade the data protection mode
- Enable Flashback Database

5 Redo Transport Services

5.1 Using the Log Writer Process (LGWR) to Archive Redo Data

```
-- It can be synced (waits for the network I/O to complete) (default)
LOG_ARCHIVE_DEST_2='SERVICE=boston LGWR SYNC NET_TIMEOUT=30'
LOG_ARCHIVE_DEST_STATE_2=ENABLE

-- or Asynced
LOG_ARCHIVE_DEST_2='SERVICE=boston LGWR ASYNC'
```

5.2 Specifying Role-Based Destinations with the VALID_FOR Attribute

- It is recommended that you define a VALID_FOR attribute for each destination so that your Data Guard configuration operates properly, including after a role transition.

```
VALID_FOR=(redo_log_type,database_role)
redo_log_type: ONLINE_LOGFILE, STANDBY_LOGFILE, or ALL_LOGFILES
database_role: PRIMARY_ROLE, STANDBY_ROLE, or ALL_ROLES
```

5.3 What to Do If Errors Occur

```
/* ReOpen */
-- the minimum number of seconds that must elapse following an error
--  before the archiving process will try again to access a failed destination
--  default 300
LOG_ARCHIVE_DEST_2= '.. REOPEN=150'


/* Using an Alternate Destination */
-- eg 1
LOG_ARCHIVE_DEST_1='LOCATION=/disk1 MANDATORY ALTERNATE=LOG_ARCHIVE_DEST_2'
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_2='LOCATION=/disk2 MANDATORY'
LOG_ARCHIVE_DEST_STATE_2=ALTERNATE

-- eg 2
-- This example shows how to define an alternate Oracle Net service
--  name to the same
standby database.
LOG_ARCHIVE_DEST_1='LOCATION=/disk1 MANDATORY'
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_2='SERVICE=stby1_path1 OPTIONAL ALTERNATE=LOG_ARCHIVE_DEST_3'
LOG_ARCHIVE_DEST_STATE_2=ENABLE
LOG_ARCHIVE_DEST_3='SERVICE=stby1_path2 OPTIONAL'
LOG_ARCHIVE_DEST_STATE_3=ALTERNATE


/* Controlling the Number of Retry Attempts */
-- he maximum number of consecutive times that redo
--  transport services attempt to transmit redo data to a failed destination
LOG_ARCHIVE_DEST_1='LOCATION=/arc_dest REOPEN=60 MAX_FAILURE=3'
```

5.4 Setting Up a Data Protection Mode

5.4.1 Data Protection Mode Requirements

- **Maximum Protection Mode**
  - At least one standby instance has a standby redo log and the LGWR, SYNC, and AFFIRM attributes be used on the LOG_ARCHIVE_DEST_n parameter for this destination.
- **Maximum Availability Mode**
  - Configure standby redo log files on at least one standby database.
  - Set the SYNC, LGWR, and AFFIRM attributes of the LOG_ARCHIVE_DEST_n parameter for at least 1 standby database.
- **Maximum Performance Mode**
  - The maximum performance mode enables you to either set the LGWR and ASYNC attributes, or set the ARCH attribute on the LOG_ARCHIVE_DEST_n parameter for the standby database destination.

5.4.2 Steps to Setup the Data Protection Mode

Step 1 Configure the LOG_ARCHIVE_DEST_n parameters on the primary database

Minimum requirement for data protection modes:

	Maximum Protection	Maximum Availability	Maximum Performance
Redo archival process	LGWR	LGWR	LGWR or ARCH
Network transmission mode	SYNC	SYNC	SYNC or ASYNC when using LGWR process. SYNC if using ARCH process
Disk write option	AFFIRM	AFFIRM	AFFIRM or NOAFFIRM
Standby redo log required?	Yes	Yes	No, but it is recommended

The following example shows how to configure the maximum availability mode:

```
ALTER SYSTEM SET LOG_ARCHIVE_DEST_2='SERVICE=chicago
 OPTIONAL LGWR SYNC AFFIRM
 VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=chicago';
```

Step 2: If you are upgrading the protection mode, perform the following:

```
-- the primary db
--  for RAC: shut down all the primary instances but start and mount only one primary instance
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
-- set the data protection mode
ALTER DATABASE SET STANDBY DATABASE TO MAXIMIZE {PROTECTION | AVAILABILITY | PERFORMANCE};
ALTER DATABASE OPEN;
```

Step 3: Configure the LOG_ARCHIVE_DEST_n parameters on standby databases

```
-- On the standby databases
ALTER SYSTEM SET LOG_ARCHIVE_DEST_2='SERVICE=boston
 OPTIONAL LGWR SYNC AFFIRM
 VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=boston';
```

Step 4 Confirm the configuration

```
SELECT PROTECTION_MODE, PROTECTION_LEVEL FROM V$DATABASE;
```

5.5 Planning for Growth and Reuse of the Control Files

5.5.1 Sizing the Disk Volumes that Contain the Control Files

The maximum control file size is 20000 database blocks. If DB_BLOCK_SIZE equals 8192, then the maximum control file size is 156 MB.

If the control file becomes full, then existing records will be overwritten. This is indicated by the message in the alert log: krcpwnc: following controlfile record written over.

5.5.2 Specifying the Reuse of Records in the Control File

Set CONTROL_FILE_RECORD_KEEP_TIME. The range of values for this parameter is 0 to 365 days. The default value is 7 days.

```
show parameter CONTROL_FILE_RECORD_KEEP_TIME
alter system set CONTROL_FILE_RECORD_KEEP_TIME=30;
```

5.6 Managing Archive Gaps

5.6.1 Manually Determining and Resolving Archive Gaps

On a physical standby database:

Determine if there is an archive gap:

```
SELECT * FROM V$ARCHIVE_GAP;
```

Locate the archived redo log files on your primary database.

Assuming the local archive destination on the primary database is LOG_ARCHIVE_DEST_1:

```
SELECT NAME FROM V$ARCHIVED_LOG WHERE THREAD#=1 AND DEST_ID=1 AND
SEQUENCE# BETWEEN 7 AND 10;
```

Copy these log files to your physical standby database and register them:

```
ALTER DATABASE REGISTER LOGFILE '/physical_standby1/thread1_dest/arcr_1_7.arc';
ALTER DATABASE REGISTER LOGFILE '/physical_standby1/thread1_dest/arcr_1_8.arc';
```

After you register these log files on the physical standby database, you can restart Redo Apply.

Make sure there is more gaps:

```
SELECT * FROM V$ARCHIVE_GAP;
```

On a logical standby database:

Determine if there is an archive gap.

If there are no gaps, the query will show only one file for each thread:

```
COLUMN FILE_NAME FORMAT a55
SELECT THREAD#, SEQUENCE#, FILE_NAME FROM DBA_LOGSTDBY_LOG L
WHERE NEXT_CHANGE# NOT IN
(SELECT FIRST_CHANGE# FROM DBA_LOGSTDBY_LOG WHERE L.THREAD# = THREAD#)
ORDER BY THREAD#,SEQUENCE#;
THREAD# SEQUENCE# FILE_NAME
---------- ----------
1     6      /disk1/oracle/dbs/log-1292880008_6.arc
1     10     /disk1/oracle/dbs/log-1292880008_10.arc
```

In this case, copy log files of sequence numbers 7, 8, and 9, to the logical standby system and register them:

```
ALTER DATABASE REGISTER LOGICAL LOGFILE '/disk1/oracle/dbs/log-1292880008_10.arc';
```

Restart SQL Apply.

5.7 Verification

5.7.1 Monitoring Log File Archival Information

Step 1 Determine the status of redo log files on the primary database.

```
SELECT THREAD#, SEQUENCE#, ARCHIVED, STATUS FROM V$LOG;
```

Step 2 Determine the most recent archived redo log file on the primary database to determine recently archived thread and sequence number:

```
SELECT MAX(SEQUENCE#), THREAD# FROM V$ARCHIVED_LOG GROUP BY THREAD#;
```

Step 3 On the primary database, determine which archived redo log file was most recently transmitted to each of the archiving destinations:

```
SELECT DESTINATION, STATUS, ARCHIVED_THREAD#, ARCHIVED_SEQ#
FROM V$ARCHIVE_DEST_STATUS
WHERE STATUS <> 'DEFERRED' AND STATUS <> 'INACTIVE';
```

Step 4 Find out if archived redo log files have been received.

You can query the DEST_ID column of the V$ARCHIVE_DEST fixed view on the primary database to identify each destination's ID number.

Assume the current local destination is 1, and one of the remote standby destination IDs is 2. To identify which log files are missing at the standby destination, issue the following query:

```
SELECT LOCAL.THREAD#, LOCAL.SEQUENCE#
  FROM (SELECT THREAD#, SEQUENCE# FROM V$ARCHIVED_LOG WHERE DEST_ID=1) LOCAL
  WHERE
   LOCAL.SEQUENCE# NOT IN
    (SELECT SEQUENCE# FROM V$ARCHIVED_LOG WHERE DEST_ID=2 AND THREAD# = LOCAL.THREAD#);
```

Step 5 Trace the progression of transmitted redo on the standby site

see Setting Archive Tracing

6 Log Apply Services

6.1 Log Apply Services Configuration Options

Use the ALTER DATABASE statement to enable the real-time apply feature, as follows:

For physical standby databases, issue the ALTER DATABASE RECOVER MANAGED STANDBY DATABASE USING CURRENT LOGFILE statement.

For logical standby databases, issue the ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE statement.

Standby redo log files are required to use real-time apply.

Setting Archive Tracing

- The LOG_ARCHIVE_TRACE parameter controls output generated by the ARCn, LGWR, and foreground processes on the primary database, and the RFS and FAL server processes on the standby database.

```
-- location of generated trace files
SHOW PARAMETER USER_DUMP_DEST

ALTER SYSTEM SET LOG_ARCHIVE_TRACE=level;
```

Where level is a combination of:

0	Disables archived redo log tracing (default setting)
1	Tracks archiving of log files
2	Tracks archive status by archive log file destination
4	Tracks archive operational phase
8	Tracks archive log destination activity
16	Tracks detailed archive log destination activity
32	Tracks archive log destination parameter modifications
64	Tracks ARCn process state activity
128	Tracks FAL server process activity
256	Track RFS Logical Client
512	Tracks LGWR redo shipping network activity
1024	Tracks RFS physical client
2048	Tracks RFS/ARCn ping heartbeat
4096	Tracks real-time apply activity
8192	Tracks Redo Apply activity (media recovery or physical standby)