Oracle Database Administration Fundamentals II (*Note Sheets*) v. 1.0

On Oracle 9i

By: Ahmed Baraka

Version History

Version	Individual Name	Date	Updates
1.0	Ahmed Baraka	Sept, 2003	Initial document.

Oracle Net Architecture and Configuration

Oracle Net Configuration Files

Following are the configuration files:

- listener.ora
- tnsnames.ora
- names.ora
- sqlnet.ora
- ldap.ora

Generally Oracle Net searches for those files in the following order:

1. The directory specified by the TNS_ADMIN environment variable.
2. The ORACLE_HOME\network\admin directory

Connection Methods

- **Spawn and Bequeath** : The listener passes or bequeaths the connection to a spawned process. This method is used in a dedicated server configuration only.
- **Direct Hand-Off Connections:** The listener will hand off a connection to a dispatcher when an Oracle Shared Server is used. This method is not possible with dedicated server processes.
- **Redirected Session:** A connection may be redirected by the listener to a dispatcher if a Shared Server is used.

Implementing Spawn and Bequeath Connections in Windows

Set USE_SHARED_SOCKET environment variable (in the registry) to TRUE to allow multiple connections to use a single socket. When the value is FALSE (default), bequeath connections are not possible so a redirect session is initiated instead.

The listener.ora File Parameters

```
LISTENER =
  (ADDRESS_LIST =
    (ADDRESS= (PROTOCOL= TCP)
              (Host= stc-sun02)
              (Port= 1521)))
SID_LIST_LISTENER =
 (SID_LIST =
   (SID_DESC =
     (ORACLE_HOME= /home/oracle)
     (GLOBAL_DBNAME = ORCL.us.oracle.com)
     (SID_NAME = ORCL)))
```

Other Parameters:

LOGGING_*lstname* : turn logging on and off (ON)

LOG_DIRECTORY_*lstname* : directory of the log file

LOG_FILE_*lstname* : filename of the log file

TRACE_LEVEL_*lstname:* Turns tracing off or on to a specific level. Possible values: *Off, User, Admin, Support*

TRACE_DIRECTORY_*lstname* : directory of the trace file

TRACE_FILE_*lstname* : filename of the trace file

SAVE_CONFIG_ON_STOP_*lstname* : whether changes made by LSNRCTL SET command are made permanent (FLASE)

Dynamic Service Registration

Configure Registration

The following initialization parameters must be configured:

- SERVICE_NAMES: specifies one or more names for the database service to which this instance connects.
- INSTANCE_NAME: the instance name

Examples

```
SERVICE_NAMES=sales.us.oracle.com
INSTANCE_NAME=salesdb
```

Registering Information with the Listener

- By default, PMON registers with a local listener on the server on the default local address of TCP/IP, port 1521
- PMON can register with a non default listener if:
  - LOCAL_LISTENER initialization parameter is defined
  - LISTENERS attribute of the DISPATCHERS initialization parameter is defined for Oracle Shared Server

Configure the Listener for Oracle9i JVM: HTTP

- If both the listener and database are Oracle9i, configuration occurs dynamically during service registration.
- If the database is Oracle8i or earlier, configure listening addresses statically using the following procedure, even if an Oracle9*i* listener is used.

1. Configure listener with TCP/IP or TCP/IP with SSL protocol
2. Enter the host name of the database in the Host field.
3. Enter port 2481 in the Port field if the chosen protocol is TCP/IP, or enter port 2482 in the Port field if the chosen protocol is TCP/IP with SSL.
4. Dedicate the address for JServer connections.

```
listener=
   DESCRIPTION_LIST=
     (DESCRIPTION=
(ADDRESS=(PROTOCOL=tcp)(HOST=server1)(PORT=2481)
)
  (PROTOCOL_STACK= (PRESENTATION=GIOP)
    (SESSION=raw))))
```

Listener Control Utility (LSNRCTL)

Invoke the utility

```
Lsnrctl
```

Starting a listener

START *listener_name*

Stopping a listener

STOP *listener_name*

Additional Commands:

- RELOAD: shuts down everything except listener addresses and rereads the listener.ora file.
- CHANGE_PASSWORD: dynamically changes the encrypted password of a listener.
- EXIT quits the LSNRCTL utility.
- QUIT same as EXIT
- HELP display list of the utility commands
- SAVE_CONFIG creates a backup of your listener configuration file (called listener.bak) and updates the listener.ora file itself to reflect any changes
- SERVICES provides detailed information about services and instances registered and the service handlers allocated to each instance.
- SET <par> sets a listener parameter. SET modifiers are:
  - CURRENT_LISTENER
  - LOG_DIRECTORY
  - LOG_FILE
  - LOG_STATUS
  - PASSWORD
  - SAVE_CONFIG_ON_STOP
  - TRC_DIRECTORY
  - TRC_FILE
  - TRC_LEVEL
- SHOW <par> lists the value of a listener parameter.
- STATUS provides basic status information about a listener, including a summary of listener configuration settings, the listening protocol addresses, and a summary of services registered with the listener.

Naming Methods

- Host naming
- Local naming
- Directory naming
- Oracle Names
- External naming

Host Naming Method

Prerequisites:

- Oracle Net Services software installed on client and server
- Client and server are connecting using TCP/IP protocol
- An IP address translation mechanism, such as Domain Name System (DNS) or a centrally maintained TCP/IP hosts file, to resolve names.
- No advanced features such as Oracle Connection Manager or security options are used

To configure the host naming method:

1. Configure the Listener

Register the database dynamically or statically.

2. Configure HOSTNAME as the First Naming Method

In the sqlnet.ora file:
NAMES.DIRECTORY_PATH=(hostname, tnsnames)

3. Set Up Host Name Resolution Environment

The service name must be resolved through an IP address translation mechanism, such as DNS, NIS, or a centrally-maintained TCP/IP host file.

4. Connect to the Database

CONNECT username/password@sales.us.acme.com

Local Naming

1. Configure tnsnames.ora File

```
SAMPLE =
 (DESCRIPTION =
  (ADDRESS_LIST =
   (ADDRESS =
    (PROTOCOL=TCP)(HOST=sun1)(PORT = 8461)))
  (CONNECT_DATA =
      (SERVICE_NAME = U461)))
```

2. Configure The sqlnet.ora File

NAMES.DIRECTORY_PATH= (TNSNAMES)

Configuration of the Oracle Shared Server

The SGA and PGA in the Shared Server

Cursor state and user session data will be stored in the large pool area, if configured. Otherwise they will be stored in SGA, specifically in shared pool.

Configuring Oracle Shared Server

- Required initialization parameters:
  - DISPATCHERS
  - SHARED_SERVERS
- Optional initialization parameters:
  - MAX_DISPATCHERS
  - MAX_SHARED_SERVERS
  - CIRCUITS
  - SHARED_SERVER_SESSIONS

Note: The parameters have reasonable defaults. On many systems, the only parameter that should be configured is DISPATCHERS.

Setting DISPATCHERS Parameter

- Parameter type: String (Specify as a quoted string)
- Parameter class: Dynamic (use ALTER SYSTEM to modify)
- Default value: NULL

Parameter Attributes:

- PROTOCOL: the network protocol
- ADDRESS : network protocol address of the endpoint on which the dispatchers listen

- DESCRIPTION: the network description of the endpoint on which the dispatchers listen, including the network protocol address For example: (DESCRIPTION=(ADDRESS=...))
- DISPATCHERS:The initial number of dispatchers to start (default is 1). Use the following formula: Initial number of dispatchers = CEIL (Average number of concurrent sessions/Connections per dispatcher)
- SESSIONS: The maximum number of network sessions for each dispatcher. Mostly the default is16K
- CONNECTIONS: the maximum number of network connections to allow for each dispatcher. The default is 1024 for Sun Solaris and Windows NT.

Example:

```
DISPATCHERS = "(PROTOCOL=TCP)(DISPATCHERS=3)"
```

Note: All attributes can be abbreviated using the first three or four letters of the attribute name.

Other Shared Server Configuration Parameters

- MAX_DISPATCHERS
  Description: Specifies the maximum number of dispatcher processes that can run simultaneously.
  Parameter type: Integer
  Parameter class: Static
  Default value5

- SHARED_SERVERS
  Description: Specifies the number of server processes created when an instance is started up.
  Parameter type: Integer
  Parameter class: Dynamic (can use ALTER SYSTEM to modify)
  Default value0

Note: Additional shared servers start automatically when needed and are deallocated automatically if they remain idle for too long. However, the initial servers always remain allocated, even if they are idle.

- MAX_SHARED_SERVERS
  Description: Specifies the maximum number of shared servers that can be started.
  Parameter type: Integer
  Parameter class: Static
  Default value: Derived from SHARED_SERVERS (either 20 or 2 * SHARED_SERVERS)

- SHARED_SERVER_SESSIONS
  Description: Specifies the total number of Oracle Shared Server user sessions to allow.
  Parameter type Integer
  Parameter class Static
  Default valuethe lesser of CIRCUITS and SESSIONS – 5
  Range of values 0 to SESSIONS – 5

- LARGE_POOL_SIZE

Configure the large pool to allocate shared server-related UGA (User Global Area), not the shared pool.

To Verify Shared-Server Setup

Verify that the dispatcher has registered with the listener when the instance was started by issuing:

```
lsnrctl services
```

Dynamic Views of Shared Server

- V$CIRCUIT
- V$SHARED_SERVER
- V$DISPATCHER
- V$SHARED_SERVER_MONITOR
- V$QUEUE
- V$SESSION

Using a Dedicated Server with Oracle Shared Server

You must use a dedicated server process when:

- Submitting batch jobs (no idle time)
- Connecting as sysdba for maintenance

In tnsnames.ora file:

```
(CONNECT_DATA=(SERVICE_NAME=TEST)(SERVER=DEDICATED
)
```

Configuring the Database Archiving Mode

Setting ARCHIVELOG mode

While database in mount mode:

```
ALTER DATABASE ARCHIVELOG | NOARCHIVELOG
```

Note: Backup the database before and after

View ARCHIVELOG mode of a Database

- V$Database (cols: log_mode)
- ARCHIVE LOG LIST (in SQL*Plus)

Setting Automatic Archiving

While Database open

```
ALTER SYSTEM ARCHIVE LOG START | STOP
```

Note: When DB starts up again, it reads its automatic archiving setting from init.ora file.

At Startup

```
LOG_ARCHIVE_START = TRUE | FALSE
```

Manually Archiving Logs

```
ARCHIVE LOG ALL | NEXT
```

Specifying Archive Destinations

Method1 (local or remote):

LOG_ARCHIVE_DEST_n *(dynamic)*

where: n is an integer from 1 to 10

Parameter Keywords:

LOCATION : a local file system

SERVICE : remote archival through Oracle Net service name

MANDATORY | OPTIONAL

REOPEN minimum number of seconds before the archiver process should try again to access a previously failed destination (default 300 seconds) (when setting to 0, it is turned off)

```
LOG_ARCHIVE_DEST_1 = 'LOCATION=/disk1/arc'
LOG_ARCHIVE_DEST_2 = 'SERVICE=standby1'
```

Method2 (local only):

LOG_ARCHIVE_DEST and

LOG_ARCHIVE_DUPLEX_DEST

Note: Any destination declared by LOG_ARCHIVE_DEST is mandatory.

```
LOG_ARCHIVE_DEST = '/disk1/arc'
```

LOG_ARCHIVE_MIN_SUCCEED_DEST Parameter

This parameter uses all mandatory destinations plus some number of optional *non-standby* destinations to determine whether LGWR can overwrite the online log.

Range of values: 1-10 in method 1 and 1-2 in method 2.

Formatting ArchiveLog Filenames

%s log sequence number

%S log sequence number, zero filled

%t thread number

%T thread number, zero filled

```
LOG_ARCHIVE_FORMAT = "LOG%s_%t.ARC"
```

Multiple ARCn Processes

LOG_ARCHIVE_MAX_PROCESSES (dynamic)

number of archiver background processes (ARC0 through ARC9) Oracle initially invokes. It can be set after startup.

Dynamic Performance Views

- V$ARCHIVE_DEST
  all the archived redo log destinations.
- V$ARCHIVE_DEST_STATUS
  displays runtime and configuration information for the archived redo log destinations.
- V$ARCHIVE_PROCESSES
  provides information about the state of the various ARCH processes for the instance.
- V$ARCHIVED_LOG
  displays archived log information from the control file, including archive log names.
- V$LOG
  contains log file information from the control files.
- V$LOG_HISTORY
  contains log history information from the control file.

User-Managed Backup

Cold Backup

Mandatory Steps:

1. Shutdown the database cleanly
2. OS Copy of data files and control files.
3. restart the database

Note: It is highly advisable to copy redo logs, parameter file and archived logs as well.

Hot or Online Backup

1. Put tablespace in backup mode using the command:

```
ALTER TABLESPACE <tname> BEGIN BACKUP
```

2. OS copy tablespace data files
3. End backup mode for the tablespace

Note: It is advisable to copy parameter file, archived logs and as well.

Note: database must be in ARCHIVELOG mode

Backing up Control Files

- Binary copy of the control file

```
Alter database backup controlfile to '…'
```

- ASCII copy of the control file:

```
Alter database backup controlfile to trace
```

Cleaning Up Failed Online Backups

1. Check data files that have STATUS column value equal to ACTIVE in the view V$BACKUP.
2. Issue the following commands for them:

```
ALTER DATABASE DATAFILE '<filename>' END BACKUP
```

or

```
ALTER DATABASE END BACKUP
```

or

```
ALTER TABLESPACE <tname> BEGIN BACKUP
```

Backing Up the Initialization Parameter File

```
CREATE PFILE = '/fname.ora' FROM SPFILE;
```

DBVERIFY Utility

External utility that can be used to ensure that a backup database or data file is valid before a restore. DBV has the following parameters

- FILE data file to be verified
- START starting block
- END ending block
- BLOCKSIZE blocksize of the data file
- LOGFILE log file to store the results
- FEEDBACK display operation progress with dots

```
dbv file=/ORADATA/u03/users01.dbf logfile=dbv.log
```

User-Managed Complete Recovery

User-Managed Complete Recovery in NOARCHIVEMODE

Backup of Redo Log is available

1. Shutdown the database
2. Copy all data files, control files and redo logs from a cold backup.
3. Startup the database

Backup of Redo Log is unavailable

After copying the backup files, issue the following:

```
RECOVER DATABASE USING BACKUP CONTROLFILE
UNTIL CANCEL ;
CANCEL
ALTER DATABASE OPEN RESETLOGS;
```

User-Managed Complete Recovery in ARCHIVEMODE

1. Take the recovered data file offline.

```
ALTER DATABASE DATAFILE '..' OFFLINE
```

2. Copy the data file(s) from backup to the original location

Note: do not restore redo logs or control files

3. Issue the command:

```
Recover Datafile '..'
or
Recover Tablespace <number> | <name>
or
Recover Database (in mount state)
or
Alter Database Recover
```

4. Take the data file online

Recovery of a Datafile Without a Backup

1. Take the tablespace offline
2. `ALTER DATABASE CREATE DATAFILE '..'`
   or
   `ALTER DATABASE CREATE DATAFILE '..' AS '..'`
3. Recover tablespace
4. Take the tablespace online

Read-Only Tablespace Recovery

RO backup and RO Recovery

Only copy tablespace data files.

RO backup and RW Recovery or RW backup and RO Recovery

Copy data files
Apply archived logs

Determining Which Files Need Recovery

- V$RECOVER_FILE
  to determine which data files need recovery.
- V$ARCHIVED_LOG
  for a list of all archived redo log files for the database.
- V$RECOVERY_LOG
  for a list of all archived redo log files required for recovery.

Using Archived Redo Log Files During Recovery

Oracle server can be notified before or during recovery, by one of the following methods:

- at the recover prompt:
  `Specify log: {<RET>=suggested | filename | AUTO | CANCEL}`
- `ALTER SYSTEM ARCHIVE LOG START TO <new location>`
- `RECOVER FROM '<new location>' DATABASE`

To apply redo log files automatically, you can issue the `SET AUTORECOVERY ON` command before starting media recovery. Remember that you can control location of archived logs by setting the parameter: `LOG_ARCHIVE_DEST`

Restoring Data Files to Different Locations

1. Startup Mount
2. `Alter Database Rename File '..' To '..'`

Recover Database

User-Managed Incomplete Recovery

Cancel-Based Recovery

1. Shutdown the database cleanly
2. Backup database files (optional but highly recommended)
3. Restore only ALL the backup data files. (Do ***not*** restore control files or redo log files)

4. You may also need to restore archived logs. If there is enough space available, restore to the LOG_ARCHIVE_DEST location or use the ALTER SYSTEM ARCHIVE LOG START TO <location> command or the SET LOGSOURCE <location> command to change the location.
5. Startup in mount state
6. Verify all data files you need to recover are online
7. Recover Database Until Cancel
8. Open database with RESETLOGS option
9. Complete Backup the database. (existing backup no longer valid)

Note: It is advisable to SHUTDOWN and then STARTUP the database after completing the recovery. Also take full database backup.

Time-Based Recovery

Same as above except step 6 which will be
RECOVER DATABASE UNTIL TIME 'yyyy-mm-dd:hh24:mi:ss'

Change-Based Recovery

Same as above except step 6 which will be
Recover Database Until Change <n>

Note: You can query V$LOG_HISTORY and V$ARCHIVED_LOG to display high and low SCN number in every archived log file.

Using a Backup Control File During Recovery

This is needed when a physical structure of the database has been changed (like accidentally dropping a tablespace).

Same as above except step 6 which will be:

Restore from the appropriate backed up Control File then issue the command:

```
RECOVER DATABASE UNTIL TIME '2002-03-09:11:44:00'
USING BACKUP CONTROLFILE
```

Notes:

- Opening a database with RESETLOGS option recreates any missing log files.
- Alert files are important source of failures occurred and to confirm success of recovery.

Oracle Recovery Manager Configuration

Connecting to RMAN without a Recovery Catalog

1. In OS command line, set ORACLE_SID environment variable
2. rman target user/pswd nocatalog

RMAN Command Line Arguments

- Writing RMAN output to a log file:

```
log $HOME/ORADATA/u03/rman.log append
```

- Executing a command file when RMAN is invoked:

```
@'$HOME/STUDENT/LABS/my_rman_script.rcv'\
```

Configuring the RMAN Environment

- **To list current configuration**

```
SHOW ALL
```

- **To set value for a configuration setting**

```
CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT '/?/%U';
CONFIGURE CONTROL FILE AUTOBAKCUP ON
```

- **To reset the setting value**

```
CONFIGURE CONTROL FILE AUTOBAKCUP CLEAR
```

RMAN Channel Commands

- **Manual Channel Allocation**

```
RUN {
 ALLOCATE CHANNEL c1 TYPE disk
 FORMAT = '/db01/BACKUP/usr0520.bak';
 BACKUP DATAFILE '/db01/ORADATA/users01.dbf';}
```

- **Automatic Channel Allocation**

```
CONFIGURE DEFAULT DEVICE TO DISK | SBT
CONFIGUE DEVICE TYPE DISK PARALLELISM n
```

- **Automatic Channel Options**

```
CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT =
'/BACKUP/RMAN/%U'
CONFIGURE CHANNEL DEVICE TYPE DISK MAXPIECESIZE 2G
```

Duration in days of RMAN information in control file

This is controlled by the initialization parameter
CONTROL_FILE_RECORD_KEEP_TIME

RMAN Backups

Backup Piece Size

```
ALLOCATE CHANNEL … MAXPIECESIZE = integer
CONFIGURE CHANNEL … MAXPIECESIZE = integer
```

Backup Command

Mandatory Steps

- Database must be OPEN or in MOUNT state
- Allocate Channel (manually or automatic)

BACKUP command options

- FULL : copies all data blocks (except not used ones).
- INCREMENTAL LEVEL n
- INCLUDE CURRENT CONTROLFILE

- FILESPERSET n :maximum number of input files in each backup set
- SKIP OFFLINE | READONLY | INACCESSIBLE
- MAXSETSIZE n K|M|G
- DELETE INPUT : useful when backing up archived redo logs, datafile copies or backup sets.
- FORMAT :%c copy number
  %p backup piece number
  %s backup set number
  %d database name
  %n database name padded with 8 characters
  %t backup set time stamp
  %u compressed version of %s and %t
  %U (default) equivalent to %u_%p_%c

To make a whole database backup
```
BACKUP DATABASE FORMAT '/tmp/%U' TAG='weekly_bak'
SQL 'ALTER SYSTEM ARCHIVE LOG CURRENT';
```

Backing Up Tablespaces
```
BACKUP TABLESPACE system, users, tools;
LIST BACKUP OF TABLESPACE
```

Backing up Datafiles and Datafile Copies
```
BACKUP DATAFILE 1,2,3,4, DATAFILECOPY
'/tmp/system01.dbf'
LIST BACKUP OF DATAFILE 1,2,3,4;
```

Backing Up Backup Sets (from disk to tape or from disk to disk)
```
BACKUP DEVICE TYPE sbt BACKUPSET ALL;
```

Control File Backups

Control File Autobackup
- Use the CONFIGURE CONTROLFILE AUTOBACKUP command to enable
- When enabled, RMAN automatically performs a back up of the control file and current server parameter file after BACKUP or COPY commands

Control File Backup Format
```
SET CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE
disk TO 'controlfile_%F';
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT '..'
```

Backing Up the Control File Manually
```
BACKUP CURRENT CONTROLFILE TAG = mondaypmbackup
LIST BACKUP OF CONTROLFILE
```

Note: two control file backups and server parameter file are created, if control file AUTOBACKUP is enabled.

Including the Control File in a Backup Set
```
BACKUP TABLESPACE users INCLUDE CURRENT CONTROLFILE;
```

Backing Up the Server Parameter File

- Automatically backed up when CONFIGURE CONTROLFILE AUTOBACKUP = ON
- Explicitly backed up with BACKUP SPFILE

```
BACKUP COPIES 2 DEVICE TYPE sbt SPFILE
```

Backing Up Archived Redo Logs

Using BACKUP ARCHIVELOG command
```
BACKUP ARCHIVELOG ALL
```

To delete backed up copy of the archived log file
```
BACKUP ARCHIVELOG ALL DELETE INPUT
```

To delete logs from all enabled archiving destinations.
```
BACKUP ARCHIVELOG ALL DELETE ALL INPUT
```

To specify a range of archived redo logs by time
```
BACKUP ARCHIVELOG UNTIL TIME 'SYSDATE-7';
BACKUP ARCHIVELOG FROM TIME 'SYSDATE-30' UNTIL
TIME 'SYSDATE-7';
```

To specify a range of archived redo logs by SCN
```
BACKUP ARCHIVELOG UNTIL SCN = 320
BACKUP ARCHIVELOG SCN BETWEEN 205 AND 320
```

To specify a range of archived redo logs by log sequence number
```
BACKUP ARCHIVELOG UNTIL SEQUENCE = 501
BACKUP ARCHIVELOG FROM SEQUENCE integer
```

Using BACKUP ... PLUS ARCHIVELOG:
```
BACKUP DATABASE PLUS ARCHIVELOG;
```

Note: In Oracle9*i*, Release 2, you can use the NOT BACKED UP *integer* TIMES clause of the BACKUP ARCHIVELOG command to back up only those logs that have not been backed up at least *integer* times.

Multiplexed Backup Sets

Multiplexing is controlled by the following:
- The FILESPERSET parameter on the BACKUP command
- The MAXOPENFILES parameter of the ALLOCATE CHANNEL and CONFIGURE CHANNEL commands (default is 8)

Parallelization of Backup Sets

Parallelization of backup sets is achieved by:
- Configuring PARALLELISM to greater than 1 **or** allocating multiple channels
- Specifying many files in the BACKUP command

By default, RMAN determines which channels should back up which database files. You can use the BACKUP ... CHANNEL command to manually assign a channel to back up specified files.

```
BACKUP
  (DATAFILE 1,2,3
   FILESPERSET = 1
   CHANNEL ORA_DISK_1)
  (DATAFILECOPY '/tmp/system01.dbf',
   '/tmp/tools01.dbf'
   FILESPERSET = 2
   CHANNEL ORA_DISK_2);

RUN
  {
   ALLOCATE CHANNEL c1 DEVICE TYPE sbt PARMS="ENV=
  (BACKUP_SERVER=tape_server1)";
   ALLOCATE CHANNEL c2 DEVICE TYPE sbt PARMS="ENV=
  (BACKUP_SERVER=tape_server2)";
   BACKUP
   (DATAFILE 1,2,3 CHANNEL c1)
```

```
(DATAFILECOPY '/tmp/system01.dbf',
 '/tmp/tools01.dbf' FILESPERSET = 2 CHANNEL c2)
}
```

Duplexed Backup Sets

You can use the following commands to produce a duplexed backup set:

- **BACKUP COPIES**
  ```
  BACKUP COPIES 3 INCREMENTAL LEVEL = 0
  DATABASE;
  BACKUP COPIES 2 DATAFILE 1, DATAFILE 2
  FORMAT '/BACKUP1/%U','/BACKUP2/%U';
  ```
- **SET BACKUP COPIES** within a run block
- **CONFIGURE … BACKUP COPIES**
  ```
  CONFIGURE CHANNEL DEVICE TYPE DISK FORMAT
  '/save1/%U', '/save2/%U';
  CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE
  TYPE sbt TO 2;
  CONFIGURE ARCHIVELOG BACKUP COPIES FOR
  DEVICE TYPE sbt TO 2;
  ```

Note: there is little value in creating multiple copies on the same physical media. For sbt channels, if you use a media manager that supports Version 2 of the SBT API, then the media manager will automatically put each copy onto a separate medium.

Note: You must set the BACKUP_TAPE_IO_SLAVES initialization parameter to TRUE in order to perform duplexed backups to an sbt device.

Image Copies

```
COPY DATAFILE { 'filename'| integer }
     | DATAFILECOPY {'filename' |TAG='tag_name'}
     | ARCHIVELOG 'filename'
     | CURRENT CONTROLFILE
     | CONTROLFILECOPY {'filename'|TAG='tag_name'}
  TO AUXNAME | 'filename'
```

```
COPY DATAFILE '/ORADATA/users_01_db01.dbf' to
'/BACKUP/users01.dbf' tag=DF3 ;
```

Note: Oracle server process copies the file and performs additional actions such as checking for corrupt blocks and registering the copy in the control file. To speed up the process of copying, you can use the NOCHECKSUM parameter.

Image Copy Parallelization

You can parallelize the copy operation by:

- Using the CONFIGURE DEVICE TYPE … PARALLELISM **or** allocating multiple channels (required in Oracle8i)
- Specifying one COPY command for multiple files

```
CONFIGURE DEVICE TYPE disk parallelism 4;
COPY      # 3 files copied in parallel
 datafile 1 TO '/BACKUP/df1.dbf',
 datafile 2 TO '/BACKUP/df2.dbf',
 datafile 3 TO '/BACKUP/df3.dbf';
```

Copying the Whole Database

- Use the REPORT SCHEMA command to list the files.
- Use the COPY command or make an image copy of each datafile.

Incremental Backups

Differential incremental: backups contain only modified blocks from level n or lower.

```
BACKUP INCREMENTAL LEVEL N ...
```
(n range: 1 to 4)

Cumulative incremental: backups contain only modified blocks from level n-1 or lower. (faster recovery, slower and larger backups)

```
BACKUP INCREMENTAL LEVEL 2 CUMULATIVE ...
```

Note: In BACKUP command, you must set one of the following parameters: DATAFILE, DATAFILECOPY, TABLESPACE, or DATABASE.

Backup in NOARCHIVELOG Mode

1. Shut down cleanly
2. Mount the database.
3. Allocate multiple channels, if not using automatic.
4. Run the BACKUP command.

Tags for Backups and Image Copies

```
BACKUP .. TAG='tag_name'
```

```
COPY DATAFILECOPY TAG='tag_name' TO
'\tmp\test.dbf'
```

RMAN Dynamic Views

- V$ARCHIVED_LOG
  shows which archives have been created, backed up, and cleared in the database.
- V$BACKUP_CORRUPTION
  shows which blocks have been found corrupt during a backup of a backup set.
- V$COPY_CORRUPTION
  shows which blocks have been found corrupt during an image copy.
- V$DATABASE_BLOCK_CORRUPTION
  displays information about database blocks that were corrupted after the last backup.
- V$BACKUP_DATAFILE
  is useful for creating equal-sized backup sets by determining the number of blocks in each datafile. It can also help you find the number of corrupt blocks for the datafile.

- V$BACKUP_REDOLOG
  shows archived logs stored in backup sets.

- V$BACKUP_SET
  shows backup sets that have been created.

- V$BACKUP_PIECE
  shows backup pieces created for backup sets.

Monitoring RMAN Backups

To correlate a process with a channel during a backup:

1. In each session, set the COMMAND ID to a different value

```
RUN
 {
  ALLOCATE CHANNEL c1 TYPE sbt;
  SET COMMAND ID TO 'sess1';
  BACKUP DATABASE;
 }
```

2. Query the joined V$SESSION and V$PROCESS views

```
SELECT SID, SPID, CLIENT_INFO
FROM   V$PROCESS p, V$SESSION s
WHERE  p.ADDR = s.PADDR
AND s.CLIENT_INFO LIKE '%id=sess%';
```

   The CLIENT_INFO column displays in the following format:
   `id=command_id,rman channel=channel_id`

3. Query the V$SESSION_LONGOPS view to get the status of the backup or copy.

Detecting Corruption

- To view corrupt blocks encountered during backups from the control file, view either V$BACKUP_CORRUPTION for backup sets or V$COPY_CORRUPTION for image copies.
- RMAN tests data and index blocks for logical corruption and logs any errors in the alert.log and server session trace file. By default, error checking for logical corruption is disabled.

RMAN Complete Recovery

Recover a Database in ARCHIVELOG Mode

```
STARTUP MOUNT;
RESTORE DATABASE;
RECOVER DATABASE;
```

Restore Datafiles to a New Location

```
run{set newname for datafile 1 to
     '/newdir/system01.dbf';
    restore database;
    switch datafile all; # record in control file
    recover database;}
```

Recover a Tablespace

```
run{
 sql "alter tablespace users offline immediate";
 restore tablespace users;
 recover tablespace users;
 sql "alter tablespace users online";        }
```

Relocate a Tablespace

```
RUN{
 SQL "alter tablespace users offline immediate";
 SET NEWNAME FOR DATAFILE
'/ORADATA/u03/users01.dbf'
  TO '/ORADATA/u04/users01.dbf';
 RESTORE TABLESPACE users;
 SWITCH datafile 3; # Update the control file and
  recovery catalog
 RECOVER TABLESPACE users; #Recover the tablespace
 SQL "alter tablespace tbs1 online";}
```

RMAN Incomplete Recovery

Incomplete Recovery of a Database

1. Mount the database.

2. The following steps should be followed:

```
RUN {
    # multiple channels for parallelization
    ALLOCATE CHANNEL c1 TYPE DISK;
    ALLOCATE CHANNEL c2 TYPE DISK;
    # recover until time, SCN or sequence
    SET UNTIL TIME = '2001-12-09:11:44:00';
    SET UNTIL TIME  "to_date('09-05-2004
    00:00:20', 'dd-mm-yyyy hh24:mi:ss')" ;
    SET UNTIL SEQUENCE 120 THREAD 1;#120 not
    included
    # ALL datafiles must be restored
    RESTORE DATABASE;
    RECOVER DATABASE;
    ALTER DATABASE OPEN RESETLOGS; }
```

3. If using a recovery catalog, register the new incarnation of the database using the command: RESET DATABASE

4. Perform a whole database backup.

Note: Insure that NLS_LANG and NLS_DATE_FORMAT environment variables are set appropriately.

Note: check the alert.log for any errors during recovery.

Note: If you need to restore archived redo log files to a new location use the SET ARCHIVELOG DESTINATION TO <location> command.

RMAN Maintenance

Cross Checking Backups and Copies

Use CROSSCHECK command to ensure that data about backup sets and image copies in the recovery catalog or control file is synchronized with corresponding files on disk or in the media management catalog.

```
CROSSCHECK BACKUPSET OF DATABASE;
CROSSCHECK BACKUP OF TABLESPACE users DEVICE
TYPE sbt COMPLETED BEFORE 'SYSDATE-31';
```

```
CROSSCHECK BACKUP OF ARCHIVELOG ALL SPFILE;
CROSSCHECK BACKUPSET 1338, 1339, 1340;
CROSSCHECK BACKUPPIECE TAG = 'nightly_backup';
CROSSCHECK CONTROLFILECOPY '/tmp/control01.ctl';
CROSSCHECK DATAFILECOPY 113, 114, 115;
```

Note: If the backup or copy is no longer available, then RMAN marks it as EXPIRED. You can determine which files are marked EXPIRED by issuing a LIST EXPIRED command.

Deleting Backups and Copies

Deleting Specified Backups and Copies

```
DELETE BACKUPPIECE 101;
DELETE CONTROLFILECOPY '/tmp/control01.ctl';
DELETE NOPROMPT ARCHIVELOG UNTIL SEQUENCE = 300;
DELETE BACKUP OF TABLESPACE users DEVICE TYPE sbt;
DELETE COPY OF CONTROLFILE LIKE '/tmp/%';
DELETE NOPROMPT BACKUP OF SPFILE COMPLETED BEFORE
'SYSDATE-7';
DELETE NOPROMPT ARCHIVELOG ALL BACKED UP 3 TIMES
TO sbt;
```

Deleting Expired Backups and Copies

```
DELETE EXPIRED BACKUP;
DELETE EXPIRED COPY;
DELETE NOPROMPT EXPIRED BACKUP OF TABLESPACE users
DEVICE TYPE sbt COMPLETED BEFORE 'SYSDATE-31';
```

Deleting Backups and Copies Rendered Obsolete by the Retention Policy

```
DELETE OBSOLETE;
```

Deleting Backups and Copies Defined as Obsolete by the DELETE Command

```
DELETE OBSOLETE REDUNDANCY = 3;
DELETE OBSOLETE RECOVERY WINDOW OF 7 DAYS;
```

Forcing the Deletion of Backups and Copies

```
DELETE FORCE NOPROMPT BACKUPSET TAG 'weekly_bkup';
```

Changing the Availability of RMAN Backups and Copies

Use the command CHANGE ... UNAVAILABLE

```
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf'
UNAVAILABLE;
CHANGE BACKUP OF CONTROLFILE UNAVAILABLE;
CHANGE BACKUP OF CONTROLFILE AVAILABLE;
CHANGE COPY OF ARCHIVELOG SEQUENCE BETWEEN 230 AND
240 UNAVAILABLE;
```

Note: If a file is marked UNAVAILABLE, RMAN will not use the file when a RESTORE or RECOVER command is issued.

Exempting a Backup or Copy from the Retention Policy

Use the command CHANGE ... KEEP FOREVER|UNTIL

```
CHANGE BACKUPSET 123 KEEP FOREVER NOLOGS;
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf'
KEEP UNTIL 'SYSDATE+60';
```

Note: Specify KEEP ... LOGS to save archived logs for a possible incomplete recovery and KEEP ... NOLOGS not to save archived logs for a possible incomplete recovery.

Note: The KEEP FOREVER clause requires the use of a recovery catalog.

Note: Use CHANGE ... NOKEEP to make the file conform to the retention policy.

The CATALOG Command

Use CATALOG command to make RMAN aware of the existence of archived logs that are not recorded in the repository as well as file copies that are created through means other than RMAN.

```
CATALOG DATAFILECOPY '/DB01/BACKUP/users01.dbf';
CATALOG CONTROLFILECOPY '/DB01/BACKUP/db.ctl';
CATALOG ARCHIVELOG
'/ORADATA/ARCHIVE1/arch_12.arc',
'/ORADATA/ARCHIVE1/arch_13.arc';
```

Note: You need to make RMAN aware of the existence of archived redo log files that are not recorded in the repository, if you manually have restored your control file from a backup.

The CHANGE ... UNCATALOG Command

Run the CHANGE ... UNCATALOG command to perform the following actions on RMAN repository records:

- Delete a specific backup or copy record from the recovery catalog
- Update a backup or copy record in the target control file repository to status DELETED

RMAN does not touch the specified physical files: it only alters the repository records for these files. You can use this command when you have deleted a backup or copy through a means other than RMAN.

```
CHANGE ARCHIVELOG … UNCATALOG;
CHANGE DATAFILECOPY '/DB01/BACKUP/users01.dbf'
UNCATALOG;
```

Recovery Catalog Creation and Maintenance

Creating Recovery Catalog

1. Create tablespace
2. Create catalog owner
3. Grant privileges
   ```
   GRANT connect, resource, recovery_catalog_owner
   ```
4. Create catalog
   ```
   rman catalog rman_db1/rman_db1@catdb
   RMAN>  create catalog tablespace rman_ts;
   ```
5. Connect to target database as SYSDBA
   ```
   RMAN TARGET SYS/SYS@TEST2 CATALOG
   RMAN/RMAN@TEST1
   ```
6. Register target database
   ```
   REGISTER DATABASE;
   ```

To Update The Recovery Catalog Manually

Use the CATALOG, CHANGE, and DELETE commands. See previous section

Resynchronization of the Recovery Catalog

- RMAN performs partial or full resynchronizations ***automatically*** as needed when you execute certain commands, including BACKUP and COPY.
- Resynchronization of the recovery catalog happens ***manually*** with RESYNC CATALOG command. You may use it in the following situations:
  - If you have run your backups in NOCATALOG mode.
  - Run it periodically every CONTROL_FILE_RECORD_KEEP_TIME days.
  - The recovery catalog has been rebuilt for any reason like recovery in catalog database.

Note: During resynchronization, Recovery Manager may add records for files that no longer exist, because files being re-cataloged are not verified. Remove such records by issuing the CHANGE ... UNCATALOG command.

Resetting a Database Incarnation

- Run the RESET DATABASE command in RMAN after executing the ***SQL statement*** ALTER DATABASE OPEN RESETLOGS.
- Use RESET DATABASE TO INCARNATION n command to undo the effects of a RESETLOGS operation by restoring backups of a prior incarnation of the database. The identifier n is obtained by the LIST INCARNATION OF DATABASE command or V$DATABASE_INCARNATION view

RMAN Catalog Reporting

- Use REPORT and LIST commands.
- Alternatively you can use SQL commands to query the RMAN data dictionary views. Examples of those view are the following:
  - RC_DATABASE
  - RC_DATAFILE
  - RC_STORED_SCRIPT
  - RC_STORED_SCRIPT_LINE
  - RC_TABLESPACE

Stored Scripts

- Stored scripts are created using the CREATE SCRIPT command

```
Create Script Level0Backup {
        backup incremental level 0
        format '/u01/db01/backup/%d_%s_%p'
        fileperset 5
        (database include current controlfile);
        sql 'alter system archive log current';}
```

- To run a script use EXECUTE SCRIPT

```
RMAN> run {execute script Level0Backup;}
```

- To list the text of a specified stored script
  - Use PRINT SCRIPT command
  - Query RC_STORED_SCRIPT_LINE
- To rewrite a script use REPLACE SCRIPT command

```
REPLACE SCRIPT Level0Backup { ... }
```

- To remove a script use DELETE SCRIPT

Export and Import Utilities

Requirements

To export tables owned by another user, you must have privileges contained in the role EXP_FULL_DATABASE

Invoking Export

```
exp username/password PARAMETER=value
or
exp username/password
PARAMETER=(value1,value2,...,valuen)

exp hr/hr TABLES=employees,departments  rows=y
file=exp1.dmp

exp system/manager OWNER=hr direct=y
```

Note: The default export filename is expdat.dmp.

Export Modes

- **Full Database Mode**
  Full=Y
- **Tablespace Mode**
  TABLESPACES=<tablespaces_list>
- **User Mode**
  Owner=<user_name>
- **Table Mode**
  Tables=<tables_list>

  You can export tables whose names match specific patterns:
  TABLES=(scott.%P%,blake.%,scott.%S%)

  You can export a table with a specified partition:
  TABLES=(emp:m,emp:sp4)

Direct Path mode

- Can be set by specifying the DIRECT=Y parameter
- Restrictions:
  - The direct-path option cannot be invoked interactively.
  - Client-side and server-side character sets must be the same.
  - The BUFFER parameter has no affect. Use RECORDLENGTH instead.

Invoking Import

```
imp hr/hr TABLES=employees,departments rows=y
file=exp1.dmp

imp system/manager FROMUSER=hr TOUSER=scott
file=exp2.dmp
```

Note: If the file parameter is not specified, Import looks for the default file expdat.dmp

Import Modes

- **Full Database Mode**
  Full=Y
- **Tablespace Mode**
  TABLESPACES=<tablespaces_list>
- **User Mode**
  FROMUSER=<u1> TOUSER=<u2> TABLES=<tables_list>
- **Table Mode** (*see Export Modes*)
  Tables=<tables_list>

Invoking Import as SYSDBA

- You need to invoke Import as SYSDBA when importing a transportable tablespace set
  imp \'username/password@instance AS SYSDBA\'
- If either the username or password is omitted, Import will prompt you for it.

Import Process Sequence

1. New tables are created
2. Data is imported
3. Indexes are built
4. Triggers are imported
5. Integrity constraints are enabled on the new tables
6. Any bitmap, functional, and/or domain indexes are built

Manually Creating Tables Before Importing Data

When tables are manually created before data is imported, the CREATE TABLE statement in the export dump file will fail because the table already exists. To avoid this failure and continue loading data into the table, set the import parameter IGNORE=y. Otherwise, no data will be loaded into the table because of the table creation error.

Using Parameter File

Parameter values can be stored in a parameter file that can be then used by export or import utilities using the PARFILE parameter

```
imp PARFILE=filename
```

Using SQL*Loader

Direct-Load Insert Operations

Serial: uses one server process to insert data beyond the high-water mark.

```
INSERT /*+ APPEND */
INTO T1_NEW
NOLOGGING
SELECT * FROM T1
```

Parallel: the statement or the table is put into parallel mode. The database must have parallel query salves configured in its initialization parameter file.

Also, you must enable parallel DML for your session

```
ALTER SESSION ENABLE PARALLEL DML
```

Use hint or place the table to be inserted into parallel mode.

```
INSERT /*+ PARALLEL (SCOTT.T1_NEW,4)*/
INTO T1_NEW
NOLOGGING
SELECT * FROM T1
or
ALTER TABLE T1_NEW PARALLEL (DEGREE 4)
```

Issuing SQL*Loader

```
sqlldr test/test control=invoice_header.ctl
```

Control File

```
-- Invoice Header Sample Control File
LOAD DATA
INFILE 'mydata.dat'
BADFILE 'baddata.dat'
DISCARDFILE 'skippeddata.dat'
REPLACE
INTO TABLE invoice_header
WHEN SLAESPERSON(100)='EDI'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY
'"'
TRAILING NULLCOLS
( COMPNO     decimal external
, INV_NO     decimal external
, DISCOUNT_RATE     decimal external
":discount_rate * .90"
, DUEDATE    date "SYYYYMMDDHH24MISS"
, INVDATE    date "SYYYYMMDDHH24MISS"
, CUST_NO    char
, CUST_CAT   char NULLIF cust_cat=BLANKS
, CO_OBJ     decimal
, SALESMAN   char    "UPPER(:salesman)"
, CUSTREF    char
)
```

- `REPLACE` keyword will delete existing data from the table. `APPEND` is used for non-empty tables and `INSERT` for empty tables.
- `TRAILING NULLCOLS` tell SQL*Loader to handle any other columns that are not present in the record as null columns.

Conventional, Direct-Path and External-Path Loads

- **Conventional load** is the default method that SQL*Loader uses. The **direct-path load** is initiated by using the `DIRECT=TRUE` keyword on the command line.
- **The external-path load** is specialized to external tables.

SQL*Loader Parallel Load Methods

- **Parallel conventional load** is performed by issuing multiple SQL*Loader commands, each with their own control file and input data file, all to the same table.
- **Intersegment concurrency with direct-path load** is performed in the same way that parallel conventional load is, but it adds the `DIRECT=TRUE` keyword and uses different table.
- **Intrasegment concurrency with direct-path load** is performed by using direct-path load to load data into a single table or partition. This is performed by placing the `DIRECT=TRUE` and `PARALLEL=TRUE` option on the command line. In this parallel server, processes load the data into temporary segments and then merge them into the individual segments.