CHAPTER 3

RMAN Setup and Configuration
Well, let’s get started with this RMAN thing, shall we? I’ll just reach down, pull on the handle…. I said pull on the handle…and, it doesn’t start. Like many other things, we first need to set up RMAN and our database for backup and recovery operations before we can actually do anything. In this chapter, we look at initial RMAN setup requirements and options. First, we look at putting the database in ARCHIVELOG mode, in case you want to do online backups. We then look at the basic RMAN interface, so that you can get into RMAN itself. Next, we discuss configuring RMAN for database backup operations. Finally, we discuss the RMAN recovery catalog, including why you might want to use it and how to configure it for use.

**Configuring to Operate in ARCHIVELOG Mode**

The first thing you need to decide when setting up RMAN for backups is what mode you are going to run your database in, ARCHIVELOG mode or NOARCHIVELOG mode. Chapter 1 discussed the benefits of running in each of these modes. In this section, we address the configuration of ARCHIVELOG mode. First, we look at the process of putting the database in ARCHIVELOG mode and taking it out. Then, we examine the process of configuring the database ARCH process.

**Preparing to Put the Database in ARCHIVELOG Mode**

A very common mistake that new DBAs make is to fail to enable the ARCH process after putting the database in ARCHIVELOG mode. As discussed in Chapter 1, the ARCH process is responsible for making copies of online redo logs and making archive redo logs out of them.

The ARCH process is signaled by the Oracle LGWR process as soon as an online redo log fills up, and LGWR switches to another online redo log group. ARCH will respond to the log switch by making copies of the online redo log in the locations defined by the Oracle database parameter `log_archive_dest_n`. Until the ARCH process has completed the creation of the ARCHIVED redo log’s copies, that online redo cannot be reused by Oracle.

The `log_archive_dest_n` parameter can be used to define up to ten different archive log destinations. These destinations can be local directories, network directories (for example, NT Folders), or even a defined database service name if you are using Oracle’s standby database product. You can define each location as a mandatory or optional location.
In addition to the LOG_ARCHIVE_DEST_n parameter(s) to the location(s) you wish Oracle to create archived redo logs, you need to set the LOG_ARCHIVE_START parameter to TRUE. It is this parameter that actually tells Oracle to fire up the ARCH process when the database instance is started. Fortunately, this parameter is dynamic as well, so if you forget to set it, you can issue the command 

```sql
alter database set log_archive_start=TRUE
```

and the ARCH process will start.

The following are other parameters that you need to consider with regard to the ARCH process:

- **LOG_ARCHIVE_STATE_n** Defines one of two different states for each log archive destination. If set to ENABLE, the ARCH process will consider the destination associated with this state as a valid archive log destination. If set to DEFER, the ARCH process will not archive logs to the related LOG_ARCHIVE_DEST_n location.

- **LOG_ARCHIVE_FORMAT** Provides a template for Oracle to use when naming archived redo logs. As Oracle creates the archived redo logs, it renames them in such a way that each of the archived redo logs has a unique name assigned to it. Using the LOG_ARCHIVE_FORMAT parameter, you can manipulate the default naming standard as you require.

- **LOG_ARCHIVE_MIN_SUCCEED_DEST** Allows the DBA to define a minimum number of archive log destination copies that must succeed in order for Oracle to be able to reuse the associated online redo log again.

Each of the different parameters mentioned thus far is defined in the Oracle9i Database Reference manual (which is part of the overall Oracle documentation), should you need further information on them.

In the following example, we have a database in ARCHIVELOG mode. We will create three different archive log destination directories, including one to a service name that supports an Oracle standby database. We will also enforce the requirement that at least two of these destinations must be written to in order for the movement of the archived redo log to be considered complete, and that the standby database must be one of those two locations. Here is an example of the use of the various database parameter file parameters related to ARCHIVELOG mode operations:

```sql
log_archive_dest_1='location=d:\oracle\oraarc\robt mandatory'
log_archive_dest_2='location=z:\oracle\oraarc\robt optional'
log_archive_dest_3='service=recover1 mandatory'
log_archive_min_succeed_dest=2
log_archive_format="robt_%s_%t.arc"
log_archive_start=TRUE
```
In this example, our first archive log destination goes to d:\oracle\oraarc\robt. The second archive log destination is to a secondary location on the Z: drive. We have made this an optional archiving location because it is a networking device (which may not be all that reliable). The third destination is to an Oracle NET service (probably a standby database) called recover1. This will cause Oracle to send the archived redo logs through net8 as they are generated.

Proceeding through the example, we have indicated that the archived redo logs must be successfully copied to at least two different locations via the `log_archive_min_succeed_dest` parameter. The format of the archived redo log is defined with the `log_archive_format` parameter and finally the ARCH process configured to start up at database startup with the `log_archive_start` parameter.

**NOTE**

If you forget to set the ARCH process to start in the initialization parameter file, you can issue the command `alter system archive log start;` to dynamically start the process with the database running. This is true of most of the parameters that are associated with archive logs.

**Switching Between ARCHIVELOG Modes**

To switch a database from NOARCHIVELOG mode to ARCHIVELOG mode, you must first shut down the database in a consistent state using one of these commands: `shutdown`, `shutdown immediate`, or `shutdown transactional`. Once the database has been cleanly shut down, restart the database instance by issuing the `startup mount` command. Finally, to put the database in ARCHIVELOG mode, issue the command `alter database archivelog`.

**RMAN Workshop: Put the Database in ARCHIVELOG Mode**

**Workshop Notes**

For this workshop, you will need an installation of the Oracle software, and a database that is up and running in NOARCHIVELOG mode.

Before starting the workshop, determine where you want Oracle to copy the archived redo logs to.

Putting the database in ARCHIVELOG mode will add some additional overhead onto the system in terms of CPU and disk I/O requirements on the system. This additional overhead should be minimal.
Step 1. Modify the database parameter file and set the `log_archive_dest_1` parameter so that it is pointing to the correct location. Also set `log_archive_start` to TRUE.

Step 2. Shut down the database:

```
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
```

Step 3. Mount the database:

```
SQL> startup mount
ORACLE instance started.
Total System Global Area 84700976 bytes
Fixed Size 282416 bytes
Variable Size 71303168 bytes
Database Buffers 12582912 bytes
Redo Buffers 532480 bytes
Database mounted.
```

Step 4. Put the database in ARCHIVELOG mode:

```
SQL> alter database archivelog ;
Database altered.
```

Step 5. Open the database:

```
SQL> alter database open;
Database altered.
```

Taking the database out of ARCHIVELOG mode is as simple as reversing the process. Shut down the database, restart the database instance by issuing the `startup mount` command, and put the database in NOARCHIVELOG mode by issuing the command `alter database noarchivelog`. Note that there is no requirement to shut down the database in a consistent manner when moving from ARCHIVELOG mode to NOARCHIVELOG mode. Here is an example of switching back into NOARCHIVELOG mode:

```
SQL> shutdown
ORACLE instance shut down.
SQL> startup mount
ORACLE instance started.
```
Total System Global Area 84700976 bytes
Fixed Size 282416 bytes
Variable Size 71303168 bytes
Database Buffers 12582912 bytes
Redo Buffers 532480 bytes

Database mounted.
SQL> alter database noarchivelog;
Database altered.
SQL> alter database open;
Database altered.

Finally, you should do a backup of the database once you have completed either task.

The RMAN Command Line

Now that the database is in ARCHIVELOG mode (if you are going to do online backups), we are ready to configure RMAN and our database for backups. Before we can do that, it would be nice to actually know how to use the RMAN executable. So, let’s take a slight detour in our setup discussion to look at the RMAN command-line interface (CLI) and how to use it.

There are two different ways to get to RMAN. The first is from the command line and the second is by using OEM. We will deal with the OEM interface in more detail in Chapter 10. Most of the examples you will see in this book, however, will be done using the CLI. We figure that if you can do it from the command line, you can do it from anywhere.

You can start RMAN from the OS prompt simply by typing the command RMAN. Once you have started the RMAN command interpreter, you can perform whatever operations you might need to perform. Often, it’s much easier to get some of the preliminary work done by using command-line parameters. Thus, when we start RMAN, we can pass several command-line parameters. You can use the command-line parameters to connect RMAN to the database you are going to back up (known as the target database), the recovery catalog, or a number of other tasks. Table 2-1 provides a list of the command-line parameters, the data type for the argument of the parameter (if there is one), and the purpose of the parameter.

Here are some examples of starting RMAN with some command-line parameters (and you will see others later):

```
RMAN target=systen/manager@robt nocatalog
RMAN target=sys/robert as sysdba@robt nocatalog
RMAN target=systen/manager@robt catalog=systen/manager@catalog log="RMAN.log"
RMAN target system/manager@robt nocatalog log "RMAN.log"
```
### Table 3-1. RMAN Command-Line Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Argument Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Character string</td>
<td>Defines the username, password, and service name of the target database to connect to.</td>
</tr>
<tr>
<td>catalog</td>
<td>Character string</td>
<td>Defines the username, password, and service name of the recovery catalog.</td>
</tr>
<tr>
<td>nocatalog</td>
<td>No arguments</td>
<td>Indicates that no recovery catalog is going to be used by this session. This parameter is the default parameter in Oracle8i and Oracle9i.</td>
</tr>
<tr>
<td>cmdfile</td>
<td>Character string</td>
<td>Indicates the name of a command file script to execute.</td>
</tr>
<tr>
<td>log</td>
<td>Character string</td>
<td>Indicates that the RMAN session should be logged. The log file will take the name of the argument to this parameter. Also causes all RMAN messages to the screen to be suppressed (except the RMAN prompt).</td>
</tr>
<tr>
<td>trace</td>
<td>Character string</td>
<td>Indicates that the RMAN session should be traced. The trace file will take the name of the argument to this parameter.</td>
</tr>
<tr>
<td>append</td>
<td>No arguments</td>
<td>Indicates that the log file (defined by the log parameter) should be appended to.</td>
</tr>
<tr>
<td>debug</td>
<td>Various arguments</td>
<td>Indicates that RMAN should be started in debug mode.</td>
</tr>
<tr>
<td>msgno</td>
<td>No arguments</td>
<td>Indicates that the RMAN- prefix should be shown with each error message. If this option is not selected, then certain non-error messages will not include a message number with them.</td>
</tr>
</tbody>
</table>
NOTE
The = sign between the command-line parameter and the value of that parameter is optional. Also, if you are running Oracle9i Real Application Clusters, you can only connect to one instance of that cluster.

If you forget the command-line arguments to RMAN (and somehow manage to leave this book and your documentation at home), then there is a way to get RMAN to display the valid command-line parameters. Simply start RMAN with an invalid parameter. As you can see in the following example, RMAN will return an error, but will also provide you with a list of valid command-line parameters (we removed some of the errors at the bottom of the listing for brevity):

```
D:\oracle\oradata\robt>RMAN help
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>quoted-string</td>
<td>connect-string for target database</td>
</tr>
<tr>
<td>catalog</td>
<td>quoted-string</td>
<td>connect-string for recovery catalog</td>
</tr>
<tr>
<td>nocatalog</td>
<td>none</td>
<td>if specified, then no recovery catalog</td>
</tr>
<tr>
<td>cmdfile</td>
<td>quoted-string</td>
<td>name of input command file</td>
</tr>
<tr>
<td>log</td>
<td>quoted-string</td>
<td>name of output message log file</td>
</tr>
<tr>
<td>trace</td>
<td>quoted-string</td>
<td>name of output debugging message log file</td>
</tr>
<tr>
<td>append</td>
<td>none</td>
<td>if specified, log is opened in append mode</td>
</tr>
<tr>
<td>debug</td>
<td>optional-args</td>
<td>activate debugging</td>
</tr>
<tr>
<td>msgno</td>
<td>none</td>
<td>show RMAN-nnnn prefix for all messages</td>
</tr>
</tbody>
</table>

TABLE 3-1. RMAN Command-Line Parameters (continued)
Chapter 3: RMAN Setup and Configuration

send         quoted-string  send a command to the media manager
pipe         string         building block for pipe names
timeout      integer        number of seconds to wait for pipe input
-----------------------------------------------------------------------
Both single and double quotes (" or ") are accepted for a quoted-string.
Quotes are not required unless the string contains embedded white-space.
RMAN-00571: =================================================================
RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
RMAN-00571: =================================================================

Using the RMAN connect Command

If you start RMAN and realize that you either have not connected to the correct
database or wish to connect to a different database (target, catalog, or auxiliary),
you can use the connect command to change which database RMAN is connected
to. To change to another target database, use the connect target command. To
change to a different recovery catalog, use the connect catalog command. To
connect to a different auxiliary database, use the connect auxiliary command.
Here are some examples of the use of the connect command:

connect target sys/password@testdb;
connect catalog rcat_user/password@robdb;

Configuring the Database
for RMAN Operations

Now that you know how to start RMAN, we need to deal with some configuration
issues. While it is possible to just fire up RMAN and do a backup, it’s a better idea
to deal with some configuration questions before you do. First, you need to set up
the database user that RMAN will be using. Next, you can configure RMAN to use
several settings by default, so we will look at those settings as well.

Set Up the Database User

By default, you can use RMAN with the SYS account (as sysdba) without any
configuration being required. Of course, that’s probably not the best account to
use when you are doing production backups. We recommend, before you use
RMAN to do a backup, that you create a separate account setup that is designated
for RMAN backups.
RMAN Workshop: Create the Target Database RMAN Backup Account

Workshop Notes
For this workshop, you will need an installation of the Oracle software, and a database that is up and running. You will need administrative privileges on this database.

Step 1. Determine the user account name that you want to use, and create it with the database create user command:

CREATE USER backup_admin IDENTIFIED BY backupuserpassword
DEFAULT TABLESPACE users;

Step 2. Grant the sysdba privilege to the backup_admin user. We need to grant this privilege because RMAN always connects to the database using the sysdba login. Here is an example of granting the sysdba privilege to the backup_admin account:

GRANT sysdba TO backup_admin;

So, what happens if you try to connect RMAN to an account that is not properly created? The following error will occur:

D:\oracle\oradata\robt>RMAN target=backup/backup@robt
Recovery Manager: Release 9.2.0.1.0 - Production
Copyright (c) 1995, 2002, Oracle Corporation. All rights reserved.
RMAN-00571: ===========================================================
RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
RMAN-00571: ===========================================================
RMAN-00554: initialization of internal recovery manager package failed
RMAN-04005: error from target database:
ORA-01031: insufficient privileges

Now that we have created the user and granted it the privileges it will need, we are a step closer to being ready to use RMAN. Still, we have some RMAN default settings we need to configure, so let’s look at those next.

Set Up Database Security
We need to discuss briefly the differences between connecting to RMAN on the local server and connecting to it via NET8. When you start RMAN, you might be
logged on to the same server as the database. In this case, if you are logged on using a privileged OS user account, you will not need to do anything beyond the Steps 1 and 2 in the preceding RMAN Workshop. How do you know whether your user account is a privileged one? It depends on the OS you are using. If you are using Unix, there is generally a Unix group called dba (though it may be called something else) that is created when the Oracle owning account (usually called Oracle) is created. If your Unix user account is assigned to this group, then you will be able to connect to a target database without any additional work. If you are using Windows NT, then the privileged users are assigned to an NT group, generally called ORA_DBA.

If you are not logging on to the local server using a privileged account, or if you are connecting to the target database using NET8 from a client workstation (for example, you are connecting using system/manager@testdb), then you need to configure your database to use a password file. To do so, you first need to create the password file. To do so, you first need to create the password file, and then configure the database so that it knows to use it. Let’s look at each of these steps in detail.

Create the Password File
To create the database password file, you use the Oracle utility `orapwd`. This command takes three parameters:

- **file** The password filename
- **password** The password for the sys user
- **entries** Any number of entries to reserve for additional privileged Oracle user accounts

By default, the Oracle database (on NT) will expect the password file to take on this naming standard, PWDSid.ora, where *sid* is your database name. Here is an example of the creation of a password file:

```
orapwd file=PWDrobt.ora password=robert entries=20
```

So, now that we have created the password file, we need to configure the database to use it, and thus allow us to do remote backups via NET8.

Configure the Database to Use the Password File
By default, an Oracle database is not configured to use the password file. To configure the database, edit the parameter file (init.ora) in your favorite editor. The parameter
we are interested in is REMOTE_LOGIN_PASSWORDFILE. This parameter can be set to one of three values in Oracle9i:

- **none**  The default value. In this case, Oracle will ignore the password file, and only local privileged logins will be recognized for sysdba access.
- **shared**  This parameter indicates that multiple databases can use the same password file. When in this mode, only the SYS user account password can be stored.
- **exclusive**  This parameter indicates that the password file is used by only one database. In this mode, the password file can contain passwords for several privileged Oracle accounts.

If you are using Oracle9i's spfile instead of a text-based parameter file, then use the alter system command to modify this parameter setting:

```
alter system set REMOTE_LOGIN_PASSWORDFILE=NONE
```

Finally, the REMOTE_LOGIN_PASSWORDFILE parameter is not dynamic, so you cannot change it with the database up and running.

### Setting the CONTROL_FILE_RECORD_KEEP_TIME Parameter

When configuring your database for RMAN, you should consider how long you wish backup records to be stored in the control file. This includes records of full database backups, specific datafile, control file, parameter file, and archive log backups. The database parameter CONTROL_FILE_RECORD_KEEP_TIME is defined in days (the default is 7). Thus, by default, Oracle will maintain RMAN backup and recovery records for a period of seven days.

This parameter can have a number of operational database impacts. First, it directly impacts the size of the database control file because, as RMAN backups occur, records relating to these backups are stored in the control file. As records are saved in the control file, the control file might well run out of space. In this case, Oracle will expand the control file to accommodate the storage of the required number of backup records. Additionally, setting this parameter to 0 will disallow any control file growth. The result will be an uncertain RMAN backup history retention period.

We suggest that you set the CONTROL_FILE_RECORD_KEEP_TIME to a value no less than your selected database backup retention period. Otherwise, you risk having database backups available on your backup media without related backup records available in the control file. This can cause serious complications if you need to recover these older backups for some reason!
Configuring RMAN Default Settings

RMAN allows you to perform automated database backup and recovery, as you will see in later chapters. To support this feature, RMAN allows you to define default values for a number of settings, such as channel configuration. In this section, we look at the configuration of default RMAN settings. Of course, if you can configure something, you will want to be able to change that configuration, and even remove it completely if required. We will look at that, too. So, what will be the benefit of all of this configuration work? It will make the process of actually doing backups much easier in the end. First, we will quickly examine the `configure` command in RMAN and all that it provides us. Then, we will look at several of the different defaults you might want to configure using the `configure` command.

Throughout this section, we use a number of terms that you might not yet be familiar with, because they are covered in later chapters. Many of the terms were introduced in Chapter 2, though others may seem not quite clear to you yet. That’s okay because, to use RMAN, none of the default configuration options are really required. We suggest you skim this section and get a feel for the various default values that you can set. Then, after you have read and applied later chapters, return here and reread this section. At that point, you will be ready to decide what defaults you want to apply to your Oracle database.

Introducing the configure Command

In Oracle9i and later, RMAN allows you to perform automated backup and recovery operations. Because of this, RMAN provides the `configure` command, which allows you to define default values to be applied when doing automated backup and recovery. Using the `configure` command, RMAN allows you to make changes to the default values of the various parameters that are persistent until cleared or changed again. The ability to customize default configuration settings allows you to execute automated RMAN operations. The following are several of the different settings that you can configure:

- A default device type, such as disk or sbt (tape), to use for RMAN jobs.
- The number of channels that are automatically allocated when performing automated backup and restore jobs.
- A tablespace exclusion policy to configure specific tablespaces to be excluded during full database backup operations.
- The maximum size for any given backup piece and the size of any backup set when doing an automated backup.
Backup optimization to default to ON or OFF. Backup optimization eliminates duplicate backups of identical datafiles (for example, those associated with read-only tablespaces) and archived redo logs.

The default filename for the snapshot control file (refer to Chapter 2 for more on the snapshot control file).

The default for automated backups of the control file to ON or OFF, as well as the default format for the control file backup output files and the default device on which to create these backups.

The default filenames for files of an auxiliary database.

A default retention policy, which determines which backups and copies are eligible for deletion because they are no longer needed.

Each configurable setting has a default value assigned to it. The defaults are stored in the database control file (as are any configured values). This is true even if you are connecting to a recovery catalog. You can see what the current defaults are by using the `show` command. The `show` command also tells you if a specific configuration is using the default setting and the value for that default setting. Any nondefault RMAN configured settings are also listed in the V$RMAN_CONFIGURATION database view. Here are some examples of the `show` command’s use:

```sql
show default device type;
show maxsetsize;
show retention policy;
show all;
```

### Configuring Various RMAN Default Settings

This section looks at setting RMAN defaults. First, let’s look at configuration of channel default settings. You can configure channels in different ways. You can configure defaults for all channels with the `configure channel device type` command, or configure defaults for specific default channels with the `configure channel n device type` command.

You can clear channel defaults for all channels with the `configure channel device type clear` command, and clear channel defaults for specific default channels with the `configure channel n device type clear` command.

When you allocate a channel with the `allocate channel` command, you can specify the assigned names to the channels that you allocate. For example, the `allocate channel d1 device type disk` command will create a channel called d1. When automated channels are allocated, Oracle assigns default names to these channels. These default names depend on the type of default device used. The following table provides an example of the default name format that will be used.
Chapter 3: RMAN Setup and Configuration

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Default Name Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk</td>
<td>ORA_DISK_n</td>
<td>ORA_DISK_1, ORA_DISK_2</td>
</tr>
<tr>
<td>Tape</td>
<td>ORA_SBT_TAPE_n</td>
<td>ORA_SBT_TAPE_1,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ORA_SBT_TAPE_2</td>
</tr>
</tbody>
</table>

The number of channels that are automatically allocated depends on the default level of parallelism defined (which we will discuss later in this chapter).

Now, let’s look at some of the number of ways that you can use the configure command to automate the backup and restore process with RMAN.

### Examples of the Use of the configure Command

This section presents some examples of using the configure command to define default values. Let’s start with an example of configuring the default backup/restore device to tape or to disk. In this case, all channels assigned to backups will be allocated to disk:

```bash
configure default device type to sbt;
configure default device type to disk;
```

Now let’s look at an example of configuring the number of channels to be allocated during an automated backup or recovery operation. Also in this example, we have set the default level of parallelism for disk operations to two. Thus, if you start an automated backup, two channels will be allocated to perform the backup in parallel.

```bash
configure device type disk parallelism 2;
configure channel 1 device type disk format 'd:\backup\robt\backup_%U';
configure channel 2 device type disk format 'e:\backup\robt\backup_%U';
```

**NOTE**

*Generally, when setting the default level of parallelism, you should set it to the number of disks or tape drives attached to which you will be backing up.*

Here is an example of controlling the size of a backup set piece or the entire backup itself. In this example, we are limiting channel 1 to create each individual backup piece at a maximum size of 100MB. Note that this command does not limit the overall size of the backup.

```bash
configure channel 1 device type disk maxpiecesize 100m;
```
If we had wished to limit all channels, we could have issued the command slightly differently:

```
CONFIGURE CHANNEL DEVICE TYPE DISK MAXPIECESIZE 100m;
```

So, why might we want to change the maximum size a given backup set piece can be? First, we might have some specific file size limitations that we have to deal with. Tapes can only handle so much data, and some disk file systems have limits on how large a given datafile can be.

We might also want to set a tape device as the default device for all channels, along with some specific parameter settings. In this case, our `configure` command might look like this:

```
-- Note that we could have used the = sign after the PARMS clause if we preferred.
-- like this PARMS='ENV=(NB_ORA_CLASS=RMAN_rs100_tape). This is true with many parameters.
CONFIGURE CHANNEL DEVICE TYPE sbt MAXPIECESIZE 100m PARMS 'ENV=(NB_ORA_CLASS=RMAN_rs100_tape)';
```

You may wish to configure a default maximum size for an entire backup set, in which case you would use this slightly modified syntax (it is followed by an example of resetting this value back to the default, which is unlimited):

```
CONFIGURE MAXSETSIZE TO 7500K;
CONFIGURE MAXSETSIZE CLEAR;
```

**CAUTION**

Be careful when you are configuring the maxsetsize for the entire database. While your database might be smaller than the maxsetsize defined initially, it could quickly grow beyond the maxsetsize, causing your database backups to fail.

If you are going to be using the automated backup feature, you are going to want to clear a given configuration, so that you can use the default. To do this, use the `configure` command with the `clear` option. In this example, we are clearing out the default options set for default channel 1:

```
configure channel 1 device type disk clear;
```

As you will see in later chapters, you can configure the backup process to create duplexed backups; in other words, multiple copies of the backup can be created at different locations. You can also configure database default settings such that automatic backups will be duplexed using the `configure` command. Here is an example where we have defined that all backups to disk by default will be duplexed, with two copies:
configure datafile backup copies for device type disk to 2;

We discussed the snapshot control file in Chapter 2. This file is a point-in-time copy of the database control file that is taken during RMAN backup operations. This ensures that the backup is consistent to a given point in time. Thus, if you add a tablespace or datafile to a database after the backup has started (assuming an online backup, of course), that tablespace or datafile will not be included in the backup. Sometimes it is desirable to have RMAN create the backup control file in a location other than the default location. In this event, you can use the configure command to define a new default location for the snapshot control file:

configure snapshot control file name to 'd:\oracle\backup\scontrolf';

You may wish to exclude specific tablespaces during an automated backup, which Oracle allows you to do with the configure command. Here is an example of excluding a tablespace by default:

configure exclude for tablespace old_data;

The configure command allows you to enable or disable backup optimization. When enabled, backup optimization will cause Oracle to skip backups of files that already have identical backups on the device being backed up to. Here is an example of configuring backup optimization:

configure backup optimization on;

Note that for optimization to occur, you must have enabled it. In addition, you must issue the backup database or backup archivelog command with the like or all option. Alternatively, you can use the backup backupset all command (more information on these types of backups is provided in later chapters). Finally, you can disable the setting for backup optimization by using the force parameter of the backup command.

Automated Backups of the Control File and the Database Parameter File

RMAN in Oracle9i offers the ability to back up the control file and the database parameter file, and you can configure these backups to take place by default. Again, the configure command can be used to configure this automated backup process to happen automatically during a backup. Here is an example of configuring automated backups of these important database files and turning the default configuration off:

configure controlfile autobackup on;
calculate controlfile autobackup off;
When autobackup of the control and parameter files is configured, the following rules apply:

- The control file and the server parameter file will be automatically backed up with each RMAN backup or copy command issued that is not included in a run block.
- If a run block is used, then the control files and parameter files will be backed up at the end of the run block if the last command is not backup or copy.

In addition to the last two types of automated control file backups, a special type of control file backup can be configured to occur as a direct result of database changes such as adding new tablespaces, adding datafiles, adding online redo logs, and so on. This type of automatic backup can only happen to disk. A special option of the configure controlfile autobackup command can be used to facilitate this backup. Here is an example:

```
RMAN> configure controlfile autobackup format for device type disk to 'd:\backup\contf\robt_%F'
```

When this option is used, the Oracle RDBMS will automatically back up the control file during database structure changes that impact the control file. These changes might include adding a new tablespace, altering the state of a tablespace or datafile (for example, bringing it online), adding a new online redo log, renaming a file, adding a new redo thread, and so forth. Note that this automated backup can only be to disk, as tape is not supported. These backups can get a bit large (since the control file contains a history of many of the past backups), so make sure you allocate enough disk space to the backup directory. In spite of the additional space that will be required, these backups can be incredibly handy to have for recovery. Finally, be aware that if the backup fails for any reason, the database operation itself will not fail.

**NOTE**

If you are not going to use a recovery catalog, and you wish to be able to recover your control file after an automated control file backup, you must know the database ID (DBID) of the database. You should, as a part of your initial setup and configuration of RMAN, note the DBID of the databases that you will be backing up and save that list somewhere safe.
Backup Retention Policy

So, how long do you want to keep your database backups? RMAN gives you the ability to configure a backup retention policy, using the `configure retention policy` command. Configuring a retention policy will not cause backups to be deleted automatically, but will cause expired backup sets to appear when the `report obsolete` command is executed (see Chapter 13 for more details on the RMAN `report` command).

Let’s look at an example. First, let’s configure a retention policy of three days:

```
RMAN> configure retention policy to recovery window of 3 days;
new RMAN configuration parameters:
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 3 DAYS;
new RMAN configuration parameters are successfully stored
```

Now that we have configured our retention policy, let’s see which backups that we might have already done show up to be obsolete:

```
RMAN> report obsolete;
RMAN retention policy will be applied to the command
RMAN retention policy is set to recovery window of 3 days
Report of obsolete backups and copies
+--------------------+-----+------------------+------------------+
<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Completion Time</th>
<th>Filename/Handle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup Set</td>
<td>4</td>
<td>02-JUN-02</td>
<td>\D:\ORACLE\ORA912\DATABASE\0ADQ1RV0_1_1</td>
</tr>
<tr>
<td>Backup Piece</td>
<td>4</td>
<td>02-JUN-02</td>
<td>\D:\ORACLE\ORA912\DATABASE\0ADQ1RV0_1_2</td>
</tr>
<tr>
<td>Backup Set</td>
<td>5</td>
<td>02-JUN-02</td>
<td>\D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_1</td>
</tr>
<tr>
<td>Backup Piece</td>
<td>5</td>
<td>02-JUN-02</td>
<td>\D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_2</td>
</tr>
<tr>
<td>Backup Set</td>
<td>6</td>
<td>02-JUN-02</td>
<td>\D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_1</td>
</tr>
<tr>
<td>Backup Piece</td>
<td>6</td>
<td>02-JUN-02</td>
<td>\D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_1</td>
</tr>
<tr>
<td>Archive Log</td>
<td>4</td>
<td>02-JUN-02</td>
<td>\D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_201.ARC</td>
</tr>
<tr>
<td>Archive Log</td>
<td>3</td>
<td>02-JUN-02</td>
<td>\D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_200.ARC</td>
</tr>
<tr>
<td>Archive Log</td>
<td>2</td>
<td>02-JUN-02</td>
<td>\D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_199.ARC</td>
</tr>
</tbody>
</table>
```

In this example, we have two backup sets, and four related backup pieces that are obsolete based on our backup retention policy. Additionally, we have three
archived redo logs that are ready to be removed as well. You can easily remove the backup sets using the **delete obsolete** command:

```
RMAN> delete obsolete;
```

The RMAN retention policy will be applied to the command:

```
RMAN retention policy is set to recovery window of 3 days
```

Using channel **ORA_DISK_1**

```
using channel ORA_DISK_2
```

Deleting the following obsolete backups and copies:

<table>
<thead>
<tr>
<th>Type</th>
<th>Key</th>
<th>Completion Time</th>
<th>Filename/Handle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backup Set</td>
<td>4</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ORA912\DATABASE\0ADQ1RV0_1_1</td>
</tr>
<tr>
<td>Backup Piece</td>
<td>4</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ORA912\DATABASE\0ADQ1RV0_1_2</td>
</tr>
<tr>
<td>Backup Set</td>
<td>5</td>
<td>02-JUN-02</td>
<td>D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_1</td>
</tr>
<tr>
<td>Backup Piece</td>
<td>6</td>
<td>02-JUN-02</td>
<td>D:\BACKUP\ROBT\BACKUP_0BDQ1SA2_1_2</td>
</tr>
<tr>
<td>Archive Log</td>
<td>4</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_201.ARC</td>
</tr>
<tr>
<td>Archive Log</td>
<td>3</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_200.ARC</td>
</tr>
<tr>
<td>Archive Log</td>
<td>2</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ADMIN\ROBT\BACKUP_0BDQ1SA2_1_2</td>
</tr>
<tr>
<td>Archive Log</td>
<td>1</td>
<td>02-JUN-02</td>
<td>D:\ORACLE\ADMIN\ROBT\ARCH\ROBT_199.ARC</td>
</tr>
</tbody>
</table>

Do you really want to delete the above objects (enter YES or NO)? <enter YES or NO>?

You can also configure the RMAN retention policy using the **configure** command:

```
RMAN> configure retention policy to redundancy 3;
```

Old RMAN configuration parameters:

```
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 2 DAYS;
```

New RMAN configuration parameters:

```
```

Note in the preceding example that the system will ask you to confirm that you really want to remove the objects that are slated to be removed. If any of the listed objects are not available to be removed, then you will need to run the **crosscheck** command (discussed in Chapter 12). Otherwise, each item listed as deleted in the **delete obsolete** output will be deleted by Oracle.

If you back up your database infrequently, you probably will prefer a redundancy policy that is stated in terms of number of backups rather than backups later than x days old. In this case, you can use the **configure** command again, this time using the **redundancy** parameter:

```
RMAN> configure retention policy to redundancy 3;
```

Old RMAN configuration parameters:

```
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 2 DAYS;
```

New RMAN configuration parameters:

```
```
CONFIGURE RETENTION POLICY TO REDUNDANCY 3;
new RMAN configuration parameters are successfully stored

The report obsolete and delete obsolete commands work just the same when using this retention policy.
Finally, if you want to disable the retention policy, you use the command
configure retention policy to none, and no retention policy will be applicable.
Use the configure retention policy clear command to reset the retention policy to the default value (three days in 9iR2).

NOTE
If you are using a tape management system, it may have its own retention policy. If the tape management system’s retention policy is in conflict with the backup retention policy that you have defined in RMAN, the tape management system’s retention policy will take precedence and your ability to recover a backup will be in jeopardy.

If You Are Using Shared Servers
If you are using Oracle’s shared servers option (known as Multi-Threaded Server, or MTS, in previous Oracle versions), then you have to configure a dedicated server for use with RMAN because RMAN cannot use a shared server session to connect to the database. If you are using a shared server architecture, refer to the Oracle9i Recovery Manager Reference manual (see Chapter 8) for more information on how to configure RMAN for use with the Oracle9i shared server option.

Summary of RMAN Configuration Tasks
We have thrown a great deal of information at you in this chapter. Let’s summarize some of the tasks that we suggest you perform on any database that you intend to install RMAN on. Here is a summary list of tasks:

1. Determine whether you wish to run the database in ARCHIVELOG mode or NOARCHIVELOG mode. Configure the database accordingly.
2. Set up a separate database user account (not sys) for use with RMAN.
3. In the database parameter file, set the CONTROL_FILE_RECORD_KEEP_TIME to a number of days equivalent to or greater than the number of days you wish to retain database backups.

4. If you are using shared servers, set up a dedicated server address for RMAN to connect to.

5. Using RMAN, connect to the target database to ensure the database is set up correctly.

6. If you intend to use automated backups, configure your default RMAN values. In particular, consider configuring the following:
   - The default degree of parallelism for tape or disk backups. Set to a default value equivalent to the number of disks or tape drives that you will be backing up to.
   - Automatic channels and device types. Configure as many channels as you have individual devices.
   - Automated control file/database parameter file auto backups.

7. Configure the retention policy as required. Make sure this retention policy is in sync with any other retention policies, such as those associated with tape management systems.

8. Configure RMAN such that it will use the new feature that causes the control file to be backed up after database changes.

9. Before you use it for production database backups, test your RMAN configuration using the methods you will find in later chapters.

The Recovery Catalog

When configuring and using RMAN, one of the initial things you need to decide is whether you are going to use a recovery catalog. In RMAN9i, RMAN defaults to the NOCATALOG connect option (previous versions expected you to connect to a recovery catalog unless you used the NOCATALOG parameter). In this section, first we look at what the recovery catalog is and when you need to use it. Then, we look at how you create a recovery catalog, and discuss backup and recovery of the recovery catalog.
**What Is the Recovery Catalog?**

The recovery catalog is an optional component of RMAN that stores historical backup information from RMAN backups. Unlike the database control file’s RMAN information, the recovery catalog data is not purged on a regular basis. Thus, the historical information in the recovery catalog tends to retain more historical information than the control file. Using a recovery catalog does have a few additional benefits over just using the database control file. Some of these benefits include:

- You must use a recovery catalog if you wish to use stored RMAN scripts.
- A recovery catalog allows you to restore a database from any previous incarnation that might be stored in the recovery catalog. An incarnation of a database is the logical lifespan of a given database. An incarnation of one database is the period of time between when the database is created and the execution of RESETLOGS on that database. Subsequent incarnations are created with each use of the RESETLOGS parameter.
- A recovery catalog offers a single, enterprise-wide repository of RMAN information. This provides an easier and more flexible central repository of enterprise backup information.
- The loss of a control file can be mitigated in several ways with the presence of a recovery catalog.
- A recovery catalog allows more flexibility when doing reporting, since you can report on the target database at a time other than the current time.
- With a recovery catalog, certain default database RMAN channel configuration information will still be maintained.

With the added benefits of using a recovery catalog, we recommend that if you have a database environment with many databases in it, you should consider using a recovery catalog. Generally, the added flexibility and centralized enterprise-wide reporting benefits of the recovery catalog outweigh the additional maintenance and administrative requirements that are added with the use of a recovery catalog.

When connecting to RMAN, you must use the CATALOG command-line parameter to indicate that you want RMAN to connect to a recovery catalog. By default, RMAN uses the NOCATALOG option, which indicates that a recovery catalog will not be used. After using the CATALOG parameter, indicate the userid and password of the
recovery catalog schema that contains the recovery catalog objects. Here is an example of connecting to the recovery catalog using the RMAN command line:

```
RMAN target='sys/robert as sysdba@robt'
catalog='cataloguser/password@bcatalog'
```

### Creating the Recovery Catalog

As you might expect, some setup is required before we can actually connect to the recovery catalog. First, we need to create the recovery catalog user and grant it the appropriate privileges. Then, we need to connect to it and create the recovery catalog schema objects. Let’s look at each of these steps next.

#### Creating the Recovery Catalog User

Generally, the recovery catalog should reside in its own database, because the recovery catalog is pretty useless if it is in the same database that you are trying to recover. The next RMAN Workshop section provides a set of detailed instructions on creating the recovery catalog user account.

---

**RMAN Workshop: Create the Recovery Catalog User Account**

**Workshop Notes**

For this workshop, you will need an installation of the Oracle software. You will need to identify a database to create the recovery catalog schema in. You will need administrative privileges in this database to create the recovery catalog user account. Finally, determine the name and password you will assign to the recovery catalog user account.

You should create a tablespace for the recovery catalog schema objects. We suggest that you size the tablespace at about 15MB to start.

**Step 1.** Create the recovery catalog user. Make sure you do not use the SYSTEM tablespace as the temporary tablespace (check out the new Oracle9i default temporary tablespace feature!). Assign the recovery catalog tablespace that you have created (as suggested in the Workshop Notes) to this schema as its default tablespace. Also, assign the recovery catalog user to an unlimited quota on the recovery catalog tablespace. Here is an example of this operation:

```
CREATE USER rcat_user IDENTIFIED BY rcat_password  
DEFAULT TABLESPACE catalog;
```

**Step 2.** Grant the following roles to the recovery catalog user:
Here is an example of granting the RCAT_USER user we created earlier the roles it requires:

```
GRANT connect, resource, recovery_catalog_owner TO rcat_user;
```

**NOTE**
The recovery catalog user account is somewhat of a privileged database account. Secure it as you would sys or system.

### Creating the Recovery Catalog Schema Objects

Now that you have created the recovery catalog database and user, it’s time to actually create the recovery catalog. This is a pretty simple process in Oracle9i. All you need to do is use RMAN. When you start RMAN, use the `target` parameter to connect to the target database, and use the `catalog` parameter to connect to the recovery catalog database schema (which you just created).

At the RMAN prompt, you then issue the `create catalog` command. Optionally, you can use the `tablespace` parameter to define a tablespace in which to create the RMAN schema objects in. The next RMAN Workshop section provides an example of using the `create catalog` command to create the recovery catalog schema.

#### RMAN Workshop: Create the Recovery Catalog

**Workshop Notes**

For this workshop, you should have completed the previous RMAN Workshop ("Create the Recover Catalog User Account"). Also, we assume that you have created a tablespace called `catalog_tbs`, and we will be creating the RMAN schema objects in that tablespace.

**Step 1.** Connect to the recovery catalog with RMAN:

```
RMAN catalog=rcat_user/rcat_password
```

**Step 2.** Issue the `create catalog` command from the RMAN prompt:

```
create catalog tablespace catalog_tbs;
```
Register the Database with the Recovery Catalog

Now that you have prepared the recovery catalog for use, you need to register databases with it. This is required before you can perform a RMAN backup of a database using the recovery catalog. This is a rather simple process, as you can see in the associated DO IT NOW section.

RMAN Workshop: Register Your Database in the Recovery Catalog

Workshop Notes
For this workshop, you should have completed the previous RMAN Workshop (“Create the Recover Catalog”).

Step 1. Using RMAN, sign into the database and the recovery catalog at the same time:

```sql
set ORACLE_SID=main_db
RMAN target=backup_admin/backupuserpassword
CATALOG=rcat_user/rcat_password@recover
```

Step 2. Register the database with the recovery catalog:

```sql
RMAN> Register database;
```

Backing Up and Recovering the Recovery Catalog

We will look at how to actually use the recovery catalog in later chapters as we discuss RMAN backups in general. Since RMAN can back up databases without a recovery catalog, it makes sense that you can use RMAN to actually back up the recovery catalog itself. If you choose to use RMAN to back up your recovery catalog, it would be a very good idea to create backups of your control file and store them in a separate location from your RMAN backups of the recovery catalog. Since recovering the control file can be a time-consuming process during a non-recovery catalog restore of your database, a separate backup of the control file is something to consider.
Other Backup and Recovery Setup and Configuration Considerations

Finally, let’s consider the other backup and recovery implications of your database. There are certain things that RMAN will not back up that you need to consider as a part of your overall backup and recovery strategy planning. These include such things as the base Oracle RDBMS software and the parameter files (tnsnames.ora, names.ora, sqlnet.ora, and so on). You need to make plans to back up and recover these files as a part of your overall backup and recovery planning.

You also need to consider your disaster planning with regard to RMAN and non-RMAN backups. How will you protect these backups from flood, fire, and earthquake? The beginning is a very good time to consider these questions, not when the fire is burning two flights below!

Summary

Whew! We have covered a great deal of ground in this chapter, and indeed there are several things you need to do before you start using RMAN. First, we looked at how to set up the database in ARCHIVELOG mode, if that is what you wish to do. Next, we looked at the RMAN command line and then at how to configure your database for use with RMAN, including setup of the password file and configuring a user account for use with RMAN. We also looked at configuring RMAN default settings. We strongly suggest you take advantage of this feature in RMAN, as it can make your life much easier. We then provided you with a summary of RMAN configuration tasks. Finally, we discussed the recovery catalog, including configuration and backup issues.