CHAPTER 4

Managing the Database Password File

In This Chapter

The database *password file* is an operating system file, separate from the database, in which Oracle stores the passwords of database administrators. The password file is key to authenticating DBAs who must connect remotely to a database over a network and who must perform administrative tasks such as starting and stopping that database. As a DBA, you need to be able to connect and perform these tasks even when the database isn't open. If a database isn't open, then the passwords stored within that database are not accessible. By storing the administrative passwords in a separate file that isn't part of the database, Oracle enables remote authentication even when the database is not running. This chapter covers creating and managing password files.

Connecting as an Administrative User

What does it mean to connect as a database administrator? Depending on your point of view, it may not mean a whole lot. You'll find that you can perform almost all administrative tasks, such as creating users, tables, tablespaces, and so forth, while logged on normally as a user with DBA privileges. However, the following commands may be issued only by someone connected as an administrator:

- `STARTUP`
- `SHUTDOWN`
- `ALTER DATABASE OPEN`
- `ALTER DATABASE MOUNT`

- `ALTER DATABASE BACKUP`
- `ARCHIVE LOG`
- `RECOVER`
- `CREATE DATABASE`

The brevity of this list may make it seem insignificant, but these are really some very significant commands. When you need to restore a damaged database file, the `RECOVER` command becomes quite significant indeed. Generally, connecting as an administrator allows you to perform tasks that involve the database being in some state other than fully open.

The internal connection

Years ago, the only way to connect as an administrator to an instance was to run Server Manager and connect internally by using the keyword `INTERNAL`. That method still works today, although it's not the method that Oracle recommends. The following example shows how it is done:

```
C:\>svrmgrl

Oracle Server Manager Release 3.1.5.0.0 - Production

(c) Copyright 1997, Oracle Corporation.  All Rights Reserved.

Oracle8i Release 8.1.5.0.0 - Production
With the Partitioning and Java options
PL/SQL Release 8.1.5.0.0 - Production

SVRMGR> connect internal
Connected.
SVRMGR>
```

Beginning with the release of Oracle8i, SQL*Plus can connect internally. Oracle added this ability in preparation for the eventual phase-out of Server Manager, so if you have Oracle8i installed, you should probably get used to using SQL*Plus instead. The only difference between using SQL*Plus and using Server Manager to connect internally is that you will want to start SQL*Plus using the `/nolog` option on the command line. Normally, SQL*Plus forces you to connect to the database as a normal user before it gives you a command prompt. The `/NOLOG` option gets you a command prompt right away. You can then issue a `CONNECT INTERNAL` command to connect internally. Here's an example:

```
$sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Wed May 19 22:24:15 1999
```

```
(c) Copyright 1999 Oracle Corporation.  All rights reserved.

SQL> connect internal
Connected.
SQL>
```

Even though Oracle no longer promotes the use of the `INTERNAL` keyword, you will likely see this method used quite often. It's easy, and it's simple. In a UNIX environment, DBAs are given UNIX accounts that are part of the dba group. Being part of the dba group allows DBAs to connect internally. Other UNIX users that are not part of the dba group won't be able to connect as internally.

For added security, or to enable internal connections to be made over the network, you can create a password file with just one password — for use when connecting internally. Once that's done, you can log on remotely.

Note

If you connect internally over the network to start a database, remember that the path to the initialization file is relative to your machine and not the database server.

The following example shows how you would connect to the `PROD` database as the internal user from a remote PC somewhere on the network:

```
SQL> connect internal@prod
Password:
Connected.
SQL>
```

Notice that the password characters were not echoed to the screen. That prevents any passersby from seeing them. If you prefer not to be prompted for the password, you can supply it with the `CONNECT` command. For example:

```
SVRMGR> connect internal/oracle@prod
Connected.
SVRMGR>
```

Remember, before you can connect as an internal user from a remote system, you need to create a password file. Otherwise, Oracle won't be able to authenticate you, and you won't be allowed to connect. You'll see how to do this later in the chapter.

The SYSOPER and SYSDBA connection

Another method for making an administrative connection to an Oracle database is to connect in one of two special roles known as `SYSDBA` and `SYSOPER`. This is the method that Oracle Enterprise Manager uses. To connect this way, you need to have an Oracle username and you need to have been granted either the `SYSDBA` or `SYSOPER` privilege.

The SYSDBA privilege gives you full control over the database. The SYSOPER privilege allows you to grant someone, perhaps a system operator, the ability to perform straightforward and routine tasks such as starting and stopping a database. Table 4-1 shows the capabilities that these two privileges provide and compares them to what you can do through an internal connection.

Table 4-1
Capabilities Provided by SYSDBA, SYSOPER, and INTERNAL

Command/Privilege/Feature	***SYSDBA***	***SYSOPER***	***INTERNAL***
Start the database	*	*	*
Close the database and shutdown the instance	*	*	*
Open the database	*	*	*
Mount the database	*	*	*
Place tablespaces into backup mode	*	*	*
Issue ARCHIVE LOG commands to view the status of archiving	*	*	*
Perform database recovery	*	*	*
Recover the database to a specific point in time	*		*
Create a new database	*		*
Connect when only DBAs are permitted to connect	*	*	*
Receive all system privileges, with the ADMIN option	*		*

A user connecting as SYSOPER will be limited and will be able to perform only the specific tasks shown in Table 4-1. Users connecting as SYSDBA have full control over the database and are not limited only to the tasks shown in Table 4-1. Since they have all possible system privileges, they can create tables, modify data, drop objects, and pretty much execute any valid SQL statement.

To connect in one of these administrative roles, you issue a special form of the CONNECT command that looks like this:

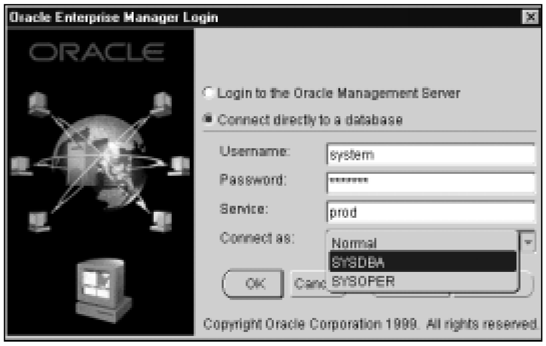
```
CONNECT username/password AS {SYSDBA|SYSOPER}
```

If you are connecting remotely, you need to include a Net8 service name:

```
CONNECT username/password@service AS {SYSDBA|SYSOPER}
```

If you are connecting through Oracle Enterprise Manager, you need to select either SYSDBA or SYSOPER from the drop-down list on the Oracle Enterprise Manager

Login screen. Figure 4-1 shows how this looks for Oracle Enterprise Manager 2.0. Note the Connect as drop-down list, and note that SYSDBA is being selected.


Oracle Enterprise Manager Login
ORACLE
Login to the Oracle Management Server
Connect directly to a database
OK
Canc
Copyright Oracle Corporation 1999. All rights reserved.

Username: system
Password: ********
Service: prod
Connect as: Normal / SYSDBA / SYSOPER

Figure 4-1: Select either SYSDBA or SYSOPER in the Oracle Enterprise Manager to be connected to your Oracle database.

A user can actually be granted both of these roles. In that case, the user has the choice of whether to connect as SYSOPER or as SYSDBA.

Note Strange as it may seem, when you connect remotely to a database as the user SYS, and you connect as SYSDBA or SYSOPER, you must use the INTERNAL password and not the SYS user's password. When you connect as SYS without specifying SYSOPER or SYSDBA, you use the SYS user's password.

Oracle's official position is that you should connect as SYSOPER or SYSDBA when performing administrative tasks. The internal connection is a deprecated feature that Oracle keeps around for backwards compatibility. A lot of Server Manager scripts still exist out there that make use of it. Someday, though, Oracle will probably remove the ability to connect internally.

The default schema

Each database user in an Oracle database is associated with a schema. Normally, the user's name and the schema's name are the same, and people often use the terms user and schema interchangeably. One of the few times that they are not the same is when you log on as an administrator. Any time that you log on as INTERNAL, SYSDBA, or as SYSOPER, your session is associated with the SYS schema.

What is the effect of this? What does it mean to be associated with the SYS schema? Take a look at the following example, where a user named Jaimie connects in the normal fashion and issues a SELECT statement to confirm her username:

```
SQL> connect jaimie/upton
Connected.
SQL> select user from dual;
USER
------------------------------
```

```
JAIMIE
1 row selected.
SQL>
```

As expected, the SELECT statement confirms that Jaimie is indeed connected as the user named “jaimie.” Now, take a look at a second example. This time, Jaimie connects as an administrator using the SYSDBA keyword:

```
SQL> connect jaimie/upton as sysdba
Connected.
SQL> select user from dual;
USER
------------------------------
SYS
1 row selected.
SQL>
```

Even though Jaimie logged on using her username and password, she is connected as the user SYS. If Jaimie were to create a table, a view, or any other schema object, it would end up being owned by SYS, not Jaimie. Because of this, and to reduce the possibility of mistakenly creating objects in the wrong schema, you should avoid creating any database objects while logged on as INTERNAL, SYSDBA, or SYSOPER.

The OSOPER and OSDBA roles

As an alternative to using a password file for authentication, on many platforms, Oracle uses two operating system roles named OSOPER and OSDBA to provide DBA operating system authentication. These correspond to SYSOPER and SYSDBA, respectively.

To use operating system authentication, your system administrator must grant you and the other DBAs either the OSOPER or the OSDBA role. Alternatively, all the DBAs can be made part of the dba group. Remember, OSOPER and OSDBA are operating system roles. The grants are done from the operating system and not from within Oracle. Your system administrator should be able to help you with that. Once granted the proper privilege, you can run SQL*Plus (or Server Manager) and connect to Oracle with a command like this:

```
   CONNECT / AS SYSDBA
```

The forward slash is used instead of a username because you don’t even need a database username at this point. You are connected to the SYS schema, just as if you had logged on internally. When you connect this way, Oracle checks to see if you have been granted the proper operating system role. If you try to connect as SYSDBA, Oracle checks to see if you have the OSDBA role. If you try to connect as SYSOPER, Oracle checks to see if you have the OSOPER role. If you have the correct role, you are allowed to connect. If not, you aren’t.

Creating a Password File

You will need a password file when you connect remotely. It's easy enough to rely on operating system authentication when you're running in a UNIX environment, where all the DBAs log on to the UNIX server to perform administrative tasks. It's a different matter when you are a DBA running Enterprise Manager on a PC and you are connecting to that same database over the LAN. Operating system authentication won't work because you aren't even logging on to the server operating system. You can't depend on the normal user passwords that are stored in the database either, because the database may not be up when you try to connect. The solution is to create a password file, allowing the DBA passwords to be stored outside of the database. To create a password file for a database, you need to perform these five steps:

1. Determine the proper location and name for the password file.
2. Shut down the database.
3. Use the `ORAPWD` utility to create the password file.
4. Set the value of the `REMOTE_LOGIN_PASSWORDFILE` initialization parameter.
5. Restart the database.

Properly setting the `REMOTE_LOGIN_PASSWORDFILE` initialization parameter is a key part of this procedure. First of all, it determines whether your password file even gets used. Secondly, it determines whether you can set administrative passwords for any user, or for just the `INTERNAL` user. This parameter is discussed further later in this section.

Password file name and location

The location in which Oracle expects to find password files is specific to each operating system. The same is true for the naming convention used to name these files. UNIX systems use one convention, and Windows NT systems use another. Table 4-2 shows the location and naming conventions used in these two environments.

Table 4-2
Location and Naming Conventions for Oracle Password Files

Operating System	***Password File Location***	***Naming Convention***
UNIX (and Linux too)	$ORACLE_HOME/dbs	orapwXXXX
Windows NT	c:\oracle\ora81\Database	PWDXXXX.ora

The letters XXXX in Table 4-2 represent the Oracle SID, or instance name. For example, if you had an instance named `PROD`, then your password file should be named `orapwPROD` in UNIX, and `PWDPROD.ora` in Windows NT. The password file location in the Windows NT environment is in the Database folder underneath your Oracle home folder. The first part of the path may vary depending on choices that you made when you installed Oracle.

If you don't get the file name right, Oracle will tell you about it when you try to start the database. You will see a message like this:

```
ORA-01990: error opening password file '/ORACLE_BASE/product/733/dbs/orapwPROD'
```

Fortunately, this message includes the full path and file name that Oracle is looking for. That's everything you need to solve the problem. Just re-create the password file where Oracle expects to find it.

The orapwd utility

Once you've figured out where to place the password file, and what to name it, it's time to create it. To create a password file, you need to run the Oracle command-line utility named `orapwd`. You run it from a command prompt and pass several pieces of information as parameters to the command. The syntax for running `orapwd` looks like this:

```
  orapwd file=filename password=password
  [entries=max_administrators]
```

The `filename` is the name to use for the password file that you are creating. You may include a path as part of the name. The `password` sets the password for use when connecting internally, and *`max_administrators`* is an optional parameter that specifies the maximum number of DBA users you can ever expect to support. This value is used to size the password file. It includes room for the internal password and for passwords of all users to which you ever expect to grant `SYSDBA` or `SYSOPER` privileges.

Note

The `orapwd` utility does not allow for spaces on either side of the equal sign. You must type `file=`*`filename`*, not `file = `*`filename`*.

The following example shows how to use the `orapwd` utility. It creates a password file for a database named `PROD`. The password file is created large enough to hold six database administrator passwords in addition to the internal password. The password for `INTERNAL` is set to *gorf*.

```
$ orapwd file=$ORACLE_HOME/dbs/orapwPROD password=gorf entries=7
```

The password file has seven entries and not six because the internal password always gets one slot. That leaves six remaining slots for the individual DBAs.

Tip

Always be generous with the number of entries that you allow for. These files aren't expandable. To add more administrators than you originally planned for, you need to re-create the file. That's a bit of a pain. Password files aren't that large, so allow plenty of room for growth.

The REMOTE_LOGIN_PASSWORDFILE parameter

The `REMOTE_LOGIN_PASSWORDFILE` initialization parameter, which, as noted earlier, is a key part of creating a password file, controls how your database instance uses the password file. You can set three possible values:

- **None** — No password file is used. This is the default setting.
- **Shared** — Multiple databases may share one password file. You are limited to one entry for the internal password.
- **Exclusive** — The password file is used exclusively by one database. It may contain entries for any number of users.

You set `REMOTE_LOGIN_PASSWORDFILE` by placing a line like the following in your database's parameter file:

```
REMOTE_LOGIN_PASSWORDFILE = EXCLUSIVE
```

In the above example, the exclusive value is set. The next subsections describe each of the options in more detail.

Using the None Option

The default value, none, is used for the starter database that you get when you first install Oracle. If you are using operating system authentication, or if you don't want to allow for remote administration by tools such as Oracle Enterprise Manager, you can leave the setting at none.

Using the Shared Option

Setting the `REMOTE_LOGIN_PASSWORDFILE` parameter to shared allows you to share one password file among several databases. When you share a password file, you are limited to just one entry for the internal password. You won't be allowed to grant `SYSDBA` or `SYSOPER` privileges. The practical effect of sharing a password file is that the internal password will be the same for all of the databases using that file. Once you give a DBA access to one database, you've given him or her access to all.

Sharing password files can be a bit of a trick. Remember, the SID forms part of the password file name, so if you have two databases, you will have two SIDS, and Oracle will be looking for two different file names. To get two databases to use the same password file, you need to create one of the files as a link to another. For example:

```
$ orapwd file=orapwGOLD password=justin
$ ln -f orapwGOLD orapwSEED
```

The UNIX ln command in this example creates a link from orapwSEED to orapwGOLD. In effect, UNIX is making one file look like two. When Oracle tries to open orapwSEED, it will really be opening orapwGOLD. By linking your files in this way, you can have several databases sharing the same password file.

Using the Exclusive Option

You use the exclusive option when you want a password file to be used by only one database. When you have an exclusive, one-to-one relationship between password file and database, Oracle allows you to grant SYSDBA and SYSOPER to users other than INTERNAL and SYS. The number of users to which you can grant those privileges is determined by the number of entries that you allocate when creating the password file. The following command, for example, creates a password file sized to hold records for ten users:

```
  orapwd file=orapwGOLD password=justin entries=10
```

If you are using Oracle Enterprise Manager to manage your databases and you want to allow yourself and other database administrators to connect remotely, then you need to use the exclusive option.

Managing Administrative Users

Once you've created the password file and set the REMOTE_LOGIN_PASSWORDFILE parameter to *exclusive*, you can enroll database users as administrators. You do this by connecting as an administrator yourself and granting the appropriate privileges (either SYSDBA or SYSOPER) to the users who need them. Once you've granted a user the SYSDBA privilege or the SYSOPER privilege, that user gets an entry in the password file. The entry remains until you revoke the privilege.

Granting administrator privileges

To grant a user either the SYSDBA or SYSOPER privilege, you need to be connected to Oracle as SYSDBA or using the internal connection. You might think that you could log on as a user with full DBA privileges, such as the user SYSTEM, and grant SYSDBA privileges, but you can't. Here's what happens if you try:

```
SQL> connect system/manager
Connected.
SQL> grant sysdba to jaimie;
grant sysdba to jaimie
*
ORA-01031: insufficient privileges
```

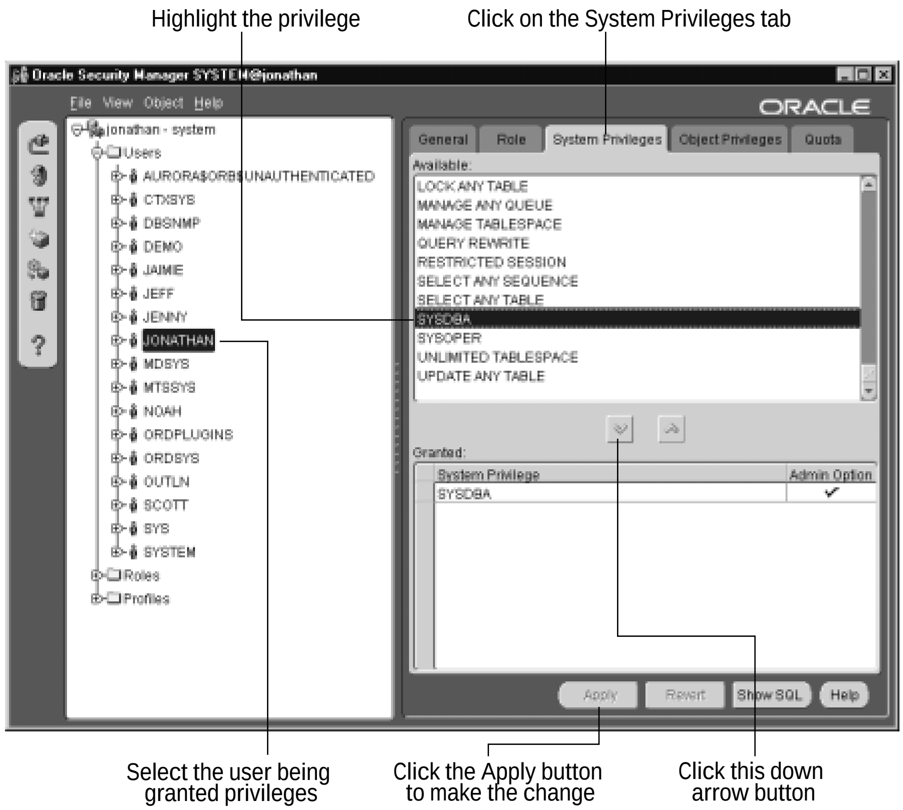
The reason the grant failed is because you logged on normally and not as SYSDBA. In this next example, the user correctly logs on as an administrator before attempting the grant:

```
SQL> connect system/manager as sysdba;
Connected.
SQL> grant sysdba to jaimie;
Statement processed.
```

This time the statement worked. The user Jaimie now has the SYSDBA role. She has something else, too — a record in the password file. From this point forward, Jaimie's database password will be stored both within the database and inside the password file. When Jaimie issues an ALTER USER command to change her password, that password is changed in the password file as well.

Using Oracle Security Manager

SYSDBA and SYSOPER privileges may also be granted from Oracle Security Manager, which is part of the Oracle Enterprise Manager toolset. Security Manager treats SYSDBA and SYSOPER as if they were just another system privilege. Figure 4-2 shows SYSDBA being granted to a user named JONATHAN:



System Privilege	Admin Option
SYSDBA	✓

Figure 4-2: The SYSDBA role has been granted to JONATHAN.

Just as with SQL*Plus and Server Manager, you need to be connected as SYSDBA when using Security Manager to grant either the SYSOPER or SYSDBA privileges.

Making the First Privilege Grant

One problem that you might encounter when you first create a password file is that none of your users will have the SYSDBA privilege to start with. If no one has it, how can you grant it? You can usually get around this by connecting internally. When you connect using the keyword INTERNAL, you have full control over the database, and you will be able to grant the SYSDBA privilege. Here's an example:

```
SQL> connect internal
Connected.
SQL> grant sysdba to system;
Statement processed.
```

Another solution, and probably the one that Oracle would recommend, would be to log on to the server as the Oracle software owner and connect as SYSDBA using operating system authentication. For example:

```
SQL> connect / as sysdba
Connected.
SQL>
```

Connecting this way using operating system authentication should work if you have logged on as the Oracle software owner or as a member of the dba group.

Listing administrators

The dynamic performance view named V$PWFILE_USERS reflects the current contents of the password file, and you can query it to see which users currently have administrator privileges. To see a list of administrators, issue a SELECT query such as the following:

```
SQL> SELECT * FROM v$pwfile_users;
USERNAME                       SYSDBA SYSOPER
------------------------------ ------ -------
INTERNAL                       TRUE   TRUE
SYS                            TRUE   TRUE
JAIMIE                         TRUE   FALSE
SYSTEM                         TRUE   FALSE
JENNY                          FALSE  TRUE
JEFF                           FALSE  TRUE
6 rows selected.
```

The results are easy to interpret. Users with a value of TRUE in the SYSDBA column have the SYSDBA privilege. Users with a value of TRUE in the SYSOPER column have the SYSOPER privilege. Users not in the list have neither privilege.

Revoking administrator privileges

To revoke SYSOPER and SYSDBA privileges, you need to log on either as SYSDBA or connect internally. Once you've done that, you'll be able to revoke the privileges from users who no longer need them. The following example shows the SYSDBA privilege being revoked from the user named JEFF:

```
SQL> $sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Wed May 19 22:24:15 1999

(c) Copyright 1999 Oracle Corporation.  All rights reserved.

SQL> CONNECT system/manager AS SYSDBA;
Connected.
SQL> REVOKE SYSDBA FROM jeff;
Statement processed.
SQL>
```

Once you've revoked a user's administrator privileges, his or her entry in the password file is no longer necessary. The entry will be deleted, and the space will be made available for use next time you grant a user SYSDBA or SYSOPER privileges.

You can never revoke SYSDBA or SYSOPER privileges from the internal connection. Oracle simply won't let you do that. If you try, you will get an error. Anyone connecting as internal always has full control over the database.

Deleting a Password File

If you currently have a password file for your database, you can remove it if you decide that you no longer want to support remote connections. To delete a password file, you should follow these steps:

1. Shut down the instance to close the database.
2. Change the value of the REMOTE_LOGIN_PASSWORDFILE parameter to none.
3. Delete the password file.
4. Restart the instance and open the database.

With the password file deleted, remote users will no longer be able to connect as SYSDBA, SYSOPER, or using the INTERNAL keyword.

Rebuilding a Password File

Someday, you may find that you need to add more users to a password file than you had originally planned for. You'll know when this happens because you'll get the error shown in the following example:

```
SVRMGR> grant sysdba to noah;
grant sysdba to oem
*
ORA-01996: GRANT failed: password file 'C:\Oracle\Ora81\DATABASE\PWDcoin.ORA' is
 full
SVRMGR>
```

When your password file is full and you still want to add another user, your only recourse is to delete the existing password file and create a new one with more entries. Here is the procedure that you should follow to do this:

1. Connect to the database and grab a list of current password file entries. You can use `SELECT * FROM v$pwfile_users` to do that.
2. Revoke each user's `SYSDBA` and `SYSOPER` privileges. Remember that you must be connected as `SYSDBA` to do this.
3. Make a note of the current internal password if you want to keep it.
4. Shut down the database.
5. Create a new password file with a sufficient number of entries. You can supply your current internal password to the `orapwd` command if you would like to keep it the same.
6. Restart the database.
7. Grant `SYSDBA` and `SYSOPER` to the users who need those privileges.

The key to this procedure is the first step, where you make a list of current password file entries. This won't give you the passwords for those users, but you don't need their passwords anyway. After you've re-created the password file, you need to regrant the same privileges to the same users. As you do that, Oracle creates a password file entry for each user. When you are done, everyone's privileges will be back to the way they were before you started.

Summary

In this chapter, you learned:

✦ Oracle supports three ways to connect to an instance when you want to perform administrative tasks such as starting or stopping the instance. You can connect internally, which uses a deprecated feature of Oracle, or you can connect in either the `SYSDBA` or `SYSOPER` role.

- Oracle uses password files to authenticate database administrators who must connect remotely from another computer on the network. These password files are external to the database so that they can be accessed even when the database is closed. This is necessary, for example, when a DBA connects in order to start a database.
- To grant `SYSDBA` and `SYSOPER` privileges to your DBAs, enabling them to connect remotely, you must have a password file with more than just one entry. You must also set the value of the `REMOTE_LOGIN_PASSWORDFILE` initialization parameter to *exclusive*.
- On Windows NT, password files are located in the Database folder underneath the Oracle home directory (`c:\oracle\ora81\database`, for example), and password files are named `PWDXXXX.ora`, where `XXXX` represents the instance name.
- On most UNIX systems, the password file location is `$ORACLE_HOME/dbs`, and the naming convention is `orapwXXXX`, with no file extension being used.

✦ ✦ ✦