CHAPTER 2

Oracle8i Overview

In This Chapter

This chapter describes Oracle8i's client/server architecture. It also lists the major components that you get when you install Oracle8i's server software, and it contrasts that with the components you get as a result of installing Oracle8i's client software. Next, you will learn, step by step, how to start and stop the database. Finally, you'll see how to access and use Oracle's online documentation.

Introducing Oracle8i's Client/Server Architecture

Oracle8i is a client/server database. This means that the database server runs independently from the applications that access it. The server listens for, and accepts, requests from clients, processes those requests, and sends the results back to clients. Traditionally, most Oracle applications have been two-tier applications, but that is changing. The advent of the Web and the emphasis on Internet access to information are driving more and more applications to be implemented using a three-tier model.

Since Oracle8i is often used in a client/server architecture, you won't be surprised that it ships software both for the server and for the client. The server portion of the package includes the database software, as well as network software to enable communication to and from clients. The client portion of the package includes utilities and assistant programs, as well as a networking piece that enables communication with Oracle servers. On Windows platforms, the client portion usually includes a number of easy-to-use GUI-based programs.

Implementing a two-tier client/server architecture

Oracle applications are often implemented in a two-tier environment. This means that you have one server running the database, and one or more clients running software programs that interact with that database. Oracle's Net8 software is used to enable communications between the clients and the server. Figure 2-1 shows how this might look.

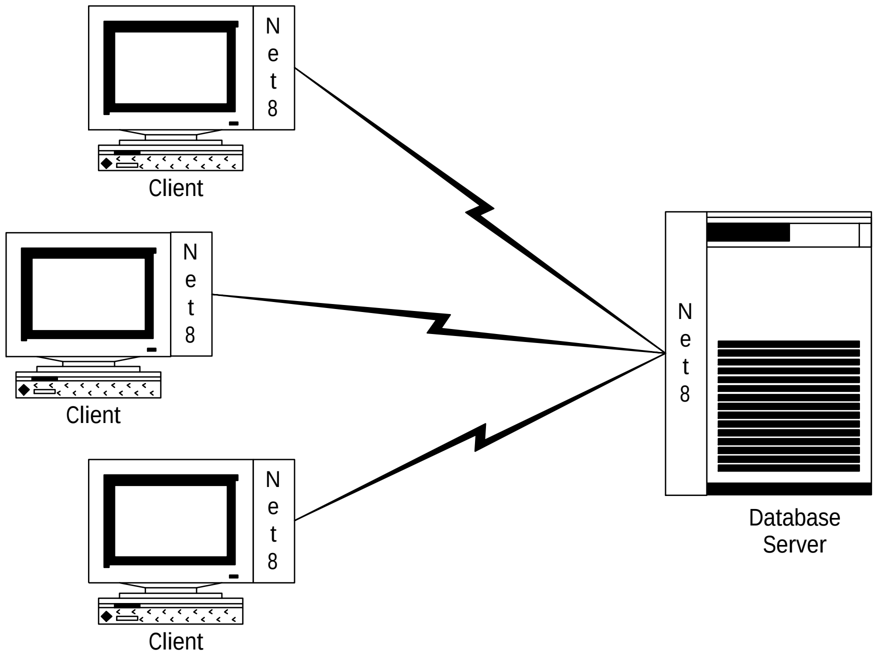


Figure 2-1: Oracle in a two-tier client/server environment

To enable easy connectivity between clients and a server, Oracle has defined a high-level networking protocol named Net8. All clients in a client/server environment have Net8 client software installed. The server will have Net8 server software installed. Net8 provides a common interface that programmers can code to, regardless of what the underlying networking protocol actually is. Oracle provides *network adapters* to transmit Net8 calls across different types of networks. TCP/IP is the most commonly used protocol, although Oracle supports a number of others as well.

Implementing a three-tier client/server architecture

Three-tier client/server configurations have gained popularity over the last few years. This is largely because of the growing importance of the Internet in the business world. The ability to interact with customers and potential clients via the Web is fast becoming a de facto requirement for being in business. Figure 2-2 shows how a typical three-tier application looks.

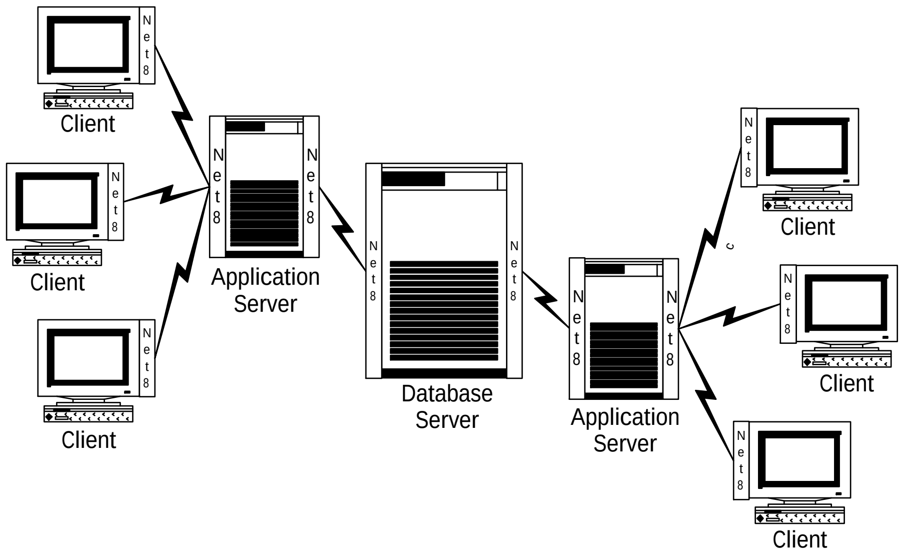


Figure 2-2: Oracle in a three-tier client/server environment

As you can see in Figure 2-2, a three-tier Oracle application still has an Oracle server and client PCs. However, an *application server* sits between the PCs and the database. Ideally, the application server contains all the application logic, the database server contains all the data, and the PCs simply manage the display and the user interaction. Three-tier configurations provide the following advantages over two-tier configurations:

- **Scalability.** A server can handle only a limited number of clients. Instead of having each client connect directly to the database server, the three-tier model allows you to spread clients over a number of application servers. As you add more clients, you can add more application servers to support them.
- **Ease of distribution.** This applies primarily to Web-based applications. Rather than distributing your software to every client using the system, you need to distribute it only to a small number of application servers, where anyone using a standard Web browser can access it.

Oracle provides a middle-tier solution called, not surprisingly, the Oracle Application Server (OAS). OAS allows you to serve up Web pages to clients and to write PL/SQL and Java code to control what goes on those pages.

In a two-tier, or even a three-tier, environment, each tier has different software installed. The next two sections describe the software that you get when you install Oracle on a server and on a client.

Taking inventory of Oracle8i's server software

When you install Oracle8i on a server, you will typically end up installing the database server software, the Net8 server software, and a core set of standard utilities. These are shown in Figure 2-3. This group of core utilities and the database behave identically — at least as far as the external world is concerned — on all platforms, whether UNIX, Windows NT, or otherwise. The differences are on the inside, where Oracle8i takes advantage of the native platform's capabilities to make the database run as efficiently as possible.

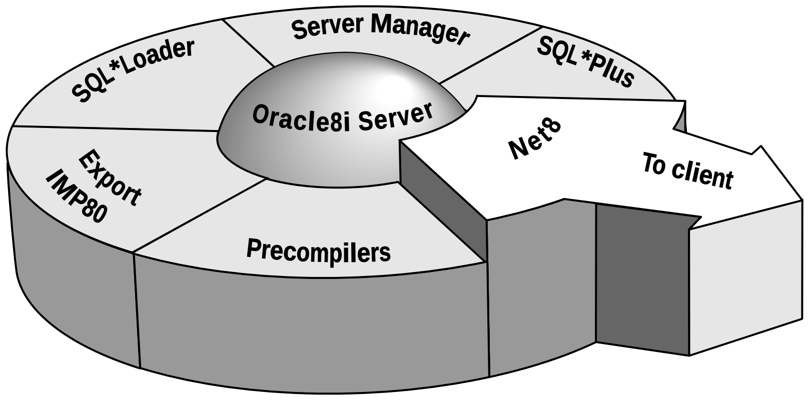


Figure 2-3: The Oracle8i database, with its core utilities, is supported on many operating system and hardware platforms.

The following are the core pieces of an Oracle server installation, as shown in Figure 2-3:

- ✥ **Oracle8i Server.** This is the database server. The server software contains all the logic necessary to store data, process queries, run PL/SQL code, and run Java code.
- ✥ **Net8.** This is Oracle's own network communication protocol. It enables rapid and efficient transmission of data between the server and the clients accessing that server.

- **SQL*Plus.** This is a tool that lets you execute any SQL statement, and in the case of `SELECT` statements, to view the results. SQL*Plus provides some limited scripting capabilities and is frequently used to run scripts that automate database administration tasks. Beginning with Oracle8i, SQL*Plus can also be used to start up, shut down, and recover a database.
- **Export and Import.** These utilities enable you to export data from or import data to any Oracle8i database. For example, you can use the export utility on a PC to copy data and table definitions into a file. Then, you can move the file to a UNIX computer, an IBM Mainframe, or any other platform that has an Oracle8i database, and use the import utility to place the information in that database.
- **SQL*Loader.** This is a utility that allows you to load data into a table from a flat file. It is most commonly used to load data from text files, such as a comma-delimited file, but it can process some types of binary files as well. While it can be difficult to learn and use, SQL*Loader is a robust utility and can quickly load large amounts of data.
- **Precompilers.** A number of these tools are available. Oracle8i contains one precompiler for each supported programming language, including Java, COBOL, Ada, C, C++, Pascal, and FORTRAN. The mix varies depending on the platform.
- **Server Manager.** Often abbreviated to SVRMGR, this utility lets you execute commands to start or stop your database. You can also run SQL commands within Server Manager. While still widely used, Oracle is beginning to phase this tool out in favor of SQL*Plus. The version of SQL*Plus shipped with Oracle8i implements all the commands that were formerly unique to Server Manager.

The previous list shows the primary tools that come with Oracle8i, regardless of the platform on which you are installing. You'll get these tools on UNIX, and you'll get them on Windows NT. However, the Windows NT version of Oracle8i has several GUI-based tools that you don't usually get on a UNIX platform:

- **Web Publishing Assistant.** This allows you to take the results of an Oracle `SELECT` statement and place them into an HTML file. Any Web browser can access the HTML file. You can do this on an ad hoc basis, or you can schedule this to occur on a regular and recurring basis.
- **Database Configuration Assistant.** This is a GUI-based wizard that you can use to create a new Oracle database. The SQL commands necessary to create a database can be a bit intimidating, especially to people brought up in a Windows environment where everything is point and click, so Oracle developed this wizard to generate and execute the commands for you.
- **Oracle Administration Assistant for Windows NT.** This is a utility that lets you perform some common DBA tasks using a GUI interface instead of commands.

- **Oracle for Windows NT Performance Monitor.** You will see an icon for this utility in the Database Administration program group, but it's really Microsoft's performance monitor running with some extensions (provided by Oracle) that allow it to monitor database activity.
- **Net8 Assistant.** Net8 is normally configured by editing several text files. Net8 Assistant provides a nice-looking GUI interface for doing the same task. The nice part about Net8 Assistant is that you don't need to remember all the keywords and syntax to use in the configuration files. If you're like most DBAs, you probably can't remember half of them anyway. Instead, you only need to fill out some forms on the screen.
- **Net8 Configuration Assistant**. This is a GUI-based wizard that walks you through the steps of initially configuring Net8 Listener.
- **Net8 Easy Config**. This is a wizard that helps you add entries to your tnsnames file.

The next two sections discuss Oracle's client software. First, we'll look at the basic elements that you might install on a user's or developer's PC. Then we'll discuss Oracle's Enterprise Manager software.

Taking inventory of Oracle8i's client software

The Oracle8i client software is installed on those users' PCs who need to interact with an Oracle database. It consists of Net8 client software used to connect the client to the server and a collection of utilities, many of which match those that get installed on the server. The client package includes the following:

- **Net8 Client.** This is part of both client and server components in an Oracle environment. The Net8 software on the client talks to the Net8 software on the server. Net8 enables Oracle applications to use a common networking interface regardless of the underlying network protocol.
- **Net8 Easy Config.** If you are installing Oracle8i client software on a Windows machine, you will get the Net8 Easy Config utility.
- **SQL*Plus.** This is installed on client PCs as well as on servers. If you're using a Windows machine, you'll notice that Oracle has wrapped a GUI interface around this command-line utility.
- **SQL*Loader.** SQL*Loader is often installed on client PCs, but it doesn't have to be. It can be omitted if the user doesn't require it.
- **Export and Import.** Like SQL*Plus and SQL*Loader, these utilities are often installed on the client as well as the server.

Figure 2-4 shows the configuration of client-side tools, including the main components of Enterprise Manager.

Figure 2-4 shows only the major components of the Oracle8i client package. The complete list is rather long, and it consists of a number of drivers and similar items that, unlike utilities, aren't invoked directly by an end user. The installation guide for your platform will have a complete, detailed list of client components.

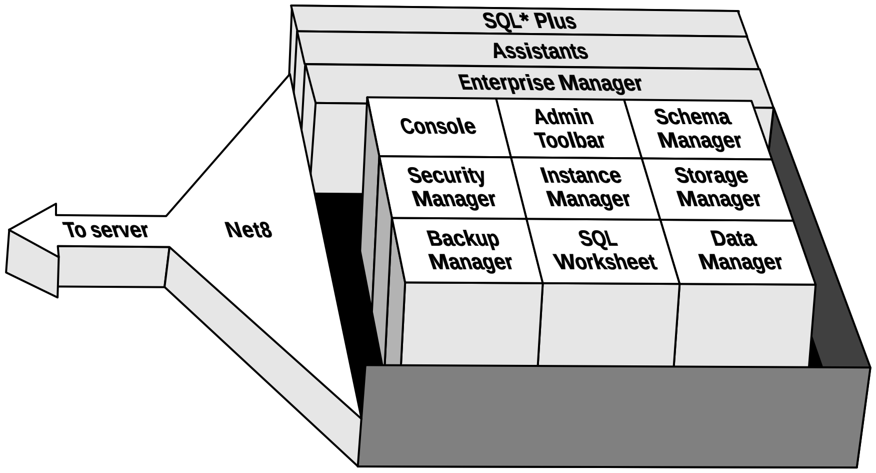


Figure 2-4: The Oracle client environment

Taking inventory of Oracle Enterprise Manager

Oracle Enterprise Manager is Oracle's GUI-based solution for managing Oracle databases. You likely already have the Oracle Enterprise Manager client software installed on your PC. The Enterprise Manager client consists of the following applications:

✥ **Enterprise Manager Console.** This is an application that enables you to monitor activity on multiple databases concurrently. The console allows you to submit database jobs to run on remote nodes, and it provides you with a unified interface that ties all the other Enterprise Manager applications together.

✥ **Schema Manager.** This tool enables you to create, alter, and drop objects such as tables, views, stored procedures, and so forth. You can navigate through lists of database objects using a tree-like interface similar to Windows NT Explorer.

✥ **Security Manager.** This tool makes creating users, setting and changing passwords, and assigning roles easy. You can also expire passwords, restrict reuse of old passwords, and more.

- **Instance Manager.** This tool enables you to see an overview of the activity on an Oracle database instance. You can view a list of users who are logged on, resolve in-doubt transactions, and change initialization parameters. You can also use Instance Manager to start and stop a database instance.
- **Storage Manager.** This tool allows you to manage database storage using a GUI interface. You use Storage Manager to add tablespaces and to add datafiles to existing tablespaces, increasing the amount of storage available for your data. You can also use Storage Manager to temporarily take a tablespace or datafile offline.
- **SQLPlus Worksheet.** This is a GUI tool that you can use as an alternative to running SQL*Plus. The SQLPlus Worksheet interface consists of two panes: an upper pane and a lower pane. The upper pane enables you to type and edit SQL commands. The lower pane displays the results from executing those commands. SQLPlus Worksheet makes editing SQL much easier than SQL*Plus does. It also has a very handy command-recall feature that allows you to review the commands that you've executed and select one to execute again.

Starting and Stopping an Oracle8i Instance

Before you can use an Oracle database, you have to start an Oracle instance and tell that instance to open the database. You can start an instance in several different ways, as described in this section. The startup process has several distinct phases, and it's important for you to understand these phases.

If you start a database instance, you'd probably assume that sooner or later, you would have to stop it. As with starting an instance, there are several ways to stop one. It's important that you understand your options here.

You might want your Oracle database instances to start automatically whenever you start (or reboot) your server. On UNIX machines, you accomplish this by having the `dbstart` script executed as part of the computer's startup process.

On a Windows NT machine, each Oracle instance is implemented as a Windows NT service. If you want an instance to start automatically, you can set the startup flag to automatic for the corresponding service, as described later in this chapter. If you'd rather start your database instances manually, you can use Instance Manager or SQL*Plus. You can also use the Services control panel to start the service. Instance Manager provides you with a GUI interface, while SQL*Plus allows you to issue commands.

Starting up in phases

If you listen to other DBAs talk, sooner or later, you will hear someone talk about "starting a database." What does it mean to "start" a database? First of all, you need to know that you really don't *start* a database. You *start* an instance, mount a database, and then you *open* a database. It's quite a mouthful, though, to say that you need to "start an instance, mount and then open a database," so most people tend to be a bit imprecise with their use of the terms.

To understand how Oracle startup works, you need to understand that a running Oracle database instance involves the following items:

- A collection of database files, including datafiles and control files
- Several processes, or programs, that operate on those files
- A shared memory area used by all of the processes to exchange data and coordinate activity

When you start an instance, you are really just starting the processes and allocating the shared memory area. Then, when you open a database, the instance opens all the files that make up the database. Figure 2-5 illustrates the phases in the startup process.

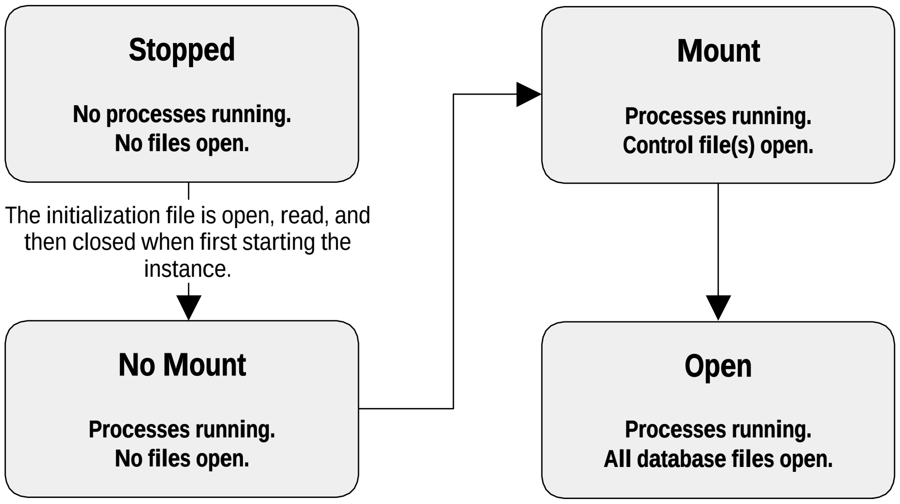


Figure 2-5: Oracle startup phases

A useful analogy is to compare Oracle to Microsoft Word. When you start up Word without opening a file, a blank screen greets you. The Word program is running, and certainly memory has been allocated to it, but that's about it. This state is comparable to the *nomount* state shown in Figure 2-5. After you've started Word, you can use the Open command on the File menu to open a file. Once you've opened a file, you are ready to get to work editing that file. This state is analogous to the *open* state shown in Figure 2-5.

Unlike Microsoft Word, however, Oracle has a state that falls between *nomount* and *open* called the *mount* state. A database is mounted when the instance has opened the control files but not any of the other database files. Figure 2-5 shows this in the frame labeled *mount*. The mount state is useful when you need to recover a database or when you need to move one or more of the datafiles. You can't move a datafile when it is open, but you must be able to record the new location of the datafile in the control file. The mount state works perfectly for that because the datafiles are closed, and the control file is open.

Shutting down an instance

When you shut down an Oracle instance, your options revolve around the questions of how quickly you want it down and how much or how little you want to inconvenience your users. You can choose from four modes when shutting down an instance:

Normal	Wait for all users to disconnect
Immediate	Wait for users to finish their current statement
Transactional	Wait for users to finish their current transaction
Abort	Don't wait for anything

The following subsections discuss each mode in detail.

Performing a Normal Shutdown

A normal shutdown generally represents the least inconvenience to your users. When you begin a normal shutdown, Oracle immediately bars any new user connections to the database. Any users currently logged on are allowed to continue whatever it is that they are doing. Eventually (hopefully), all users will log off of the database. Only when all the users have voluntarily disconnected will Oracle close the database and shut down the instance.

A normal shutdown is usually considered to be the safest choice. It can also be a frustrating choice because you effectively lose control over when the shutdown occurs. If your users tend to log on at 8:00 AM and stay logged on until 5:00 PM, you may not be able to complete a normal shutdown during business hours. Couple that with batch processing that kicks off in the evening, and you may find yourself waiting all night.

Performing an Immediate Shutdown

Doing an immediate shutdown is a good choice when you must have the database down quickly, even at the expense of inconveniencing any users who happen to be logged on at the time. When you begin an immediate shutdown, Oracle bars any new user connections to the database. So far, this is the same as a normal shutdown, but that's where the similarity ends. Any users not executing a SQL statement will have their connections terminated immediately. Users who are executing SQL statements will have their connections terminated as soon as their statements complete.

You can use immediate shutdowns before making cold database backups at night. A *cold backup* is one that occurs while the database is closed. If you're writing a script to make a cold backup of your database at 3:00 am, you don't want that script to be held up all night because one user left his or her application logged on to your database. An immediate shutdown avoids that possibility.

One other useful time for an immediate shutdown is when you need to *bounce* your instance during business hours. Hopefully, this doesn't happen to you much, because if it did, you might be out of a job. Bouncing an instance refers to the practice of quickly stopping, and then restarting, the instance. This is sometimes done when a critical initialization parameter needs to be changed. If you need to bounce an instance during the day, an immediate shutdown is often the quickest way to do it, and sometimes it's best to just get the painful process over with quickly.

Performing a Transactional Shutdown

Transactional shutdowns are a new Oracle8i feature. A transactional shutdown is just like an immediate shutdown, except that users are allowed to finish their current *transactions* rather than their current *statements*. Depending on the type and length of your transactions, this may take a bit longer, but at least you won't terminate anyone's connection in the middle of a logical unit of work.

Aborting

An abort is a good type of shutdown to avoid. When you shut down a database in the abort mode, everything just stops. All the users are disconnected immediately. All the processes are stopped. All the memory is released. Unfortunately, the database files are not left in a consistent state. It's as if you kicked the computer's plug out of the wall. Most likely, some data will exist in memory that should have been written to the datafiles, which didn't occur because of the abort. You won't lose any committed transactions because next time you start the instance, Oracle will recover those transactions from the redo log files. However, that recovery process can take some time.

Caution Backups made after a shutdown abort are not reliable. You may not be able to restore them. If you are shutting down your database to make a backup, you should not use the abort mode. If you are forced to do a shutdown abort, you should restart your database and do a normal, immediate, or transactional shutdown before making a backup.

Consider using the abort mode most often after an instance has already crashed, and when that crash has left processes and memory in an unstable state. The characteristics of this state are that when you try to start your instance, you get an error message saying that it is already started, and when you try to stop the same instance, you get an error message saying that it is not running. Obviously, both can't be true at the same time. This state of affairs happens when an instance crashes and a few of the background processes are still running. The solution is to issue a shutdown abort command.

Using SQL*Plus to start and stop an instance

If you want to start or stop an instance by issuing commands, SQL*Plus is the tool to use. Even if you don't like to use commands, you should learn how in case GUI tools such as Enterprise Manager aren't available.

Note

Prior to the Oracle8i release (8.1.5), you could not use SQL*Plus to start or stop a database. If you are running an older release of Oracle, you will need to use Server Manager instead.

You can use SQL*Plus to start and stop an instance on the machine that you are directly connected to, or you can use it to start and stop an instance on a remote server that is connected to your network. It's easiest to use SQL*Plus to operate on a database instance when you are logged on to the server. You need to be aware of a couple of points to effectively use SQL*Plus to start or stop remote instances. We'll cover them later in this chapter.

Connecting So That You Can Start or Stop an Instance

If you are going to use SQL*Plus to start or stop a database instance, you first need to start SQL*Plus properly. SQL*Plus normally prompts you for a username and password when you first start it, and then it logs you on to the database as a normal user. You need to inhibit this behavior, because starting (or stopping) an instance requires that you connect not as a normal user, but as a DBA. Use the following command to start SQL*Plus:

```
sqlplus /nolog
```

The `/nolog` option tells SQL*Plus to start without automatically logging you on to the database. If you are running on Windows NT, you will find it easiest to issue this command from a Command Prompt window. Next, you need to connect to the instance that you want to start. Either of the following two commands will work:

```
CONNECT INTERNAL
CONNECT / AS SYSDBA
```

Of these two commands, Oracle prefers that you use the latter. Oracle intends to eliminate the `INTERNAL` user someday and discourages its use. It's included in this book because regardless of what Oracle might like, lots of DBAs still connect as the `INTERNAL` user. You need to be aware that it's an option, because you'll see people and scripts using it.

Note

For either of the previous two commands to work, you must be logged on to the server as a member of the dba group if you are using UNIX, or as an administrator if you are using Windows NT.

The `AS SYSDBA` syntax in the second `CONNECT` command is important. It's the reason that you must use the `/nolog` command-line option when you start SQL*Plus. `SYSDBA` is a special database role that gives you the necessary privileges to perform administrative functions such as starting a database.

Starting an Instance

Once you've connected to your instance in the SYSDBA role, you can issue one of the following STARTUP commands:

STARTUP	Use STARTUP when you want to both start the instance and open the database for general use. Most of the time, this is what you will want to do.
STARTUP NOMOUNT	Use STARTUP NOMOUNT when you just want to start the instance and nothing more. You have to use STARTUP NOMOUNT when you are going to create a new database, and that's pretty much the only time you'll need to use it.
STARTUP MOUNT	Use STARTUP MOUNT when you want to recover the database or when you want to issue ALTER DATABASE RENAME DATAFILE commands. The STARTUP MOUNT command causes the instance to open the control files, but the other database files are left closed.

Most of the time, you'll find yourself using the STARTUP command with no parameters to start an instance and open it for general use. The following example shows the entire sequence of starting SQL*Plus, connecting, and starting an instance. Listing 2-1 shows screen output from a Windows NT machine, but the commands are exactly the same on UNIX.

Listing 2-1: **Starting SQL*Plus, connecting, and starting an instance**

```
C:\>sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Fri Jun 25 19:19:58 1999

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

SQL> connect / as sysdba
Connected to an idle instance.
SQL> startup
ORACLE instance started.

Total System Global Area   38322124 bytes
Fixed Size                    65484 bytes
Variable Size              21405696 bytes
Database Buffers           16777216 bytes
Redo Buffers                  73728 bytes
Database mounted.
Database opened.
SQL>
```

If you've performed a default Oracle install on your server and you are starting the default instance, you should be able to start it by following the previous example.

Stopping an Instance

To stop an instance using SQL*Plus, issue one of the following commands:

`SHUTDOWN`	Performs a normal shutdown, waiting for all users to voluntarily connect.
`SHUTDOWN IMMEDIATE`	Does an immediate shutdown, forcibly disconnecting each user after his or her current SQL statement completes.
`SHUTDOWN TRANSACTIONAL`	Performs a transactional shutdown, forcibly disconnecting each user after his or her current transaction completes.
`SHUTDOWN ABORT`	Aborts the instance. Everything stops. Crash recovery is necessary, and the database files are not consistent with one another.

The following example shows how you would use SQL*Plus to shut down a database in the immediate mode:

```
C:\>sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Sun Jun 27 22:19:35 1999

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

SQL> connect / as sysdba
Connected.
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

The only difference that you'll notice with the other modes is that normal might take longer because you have to wait until all users disconnect, and with an abort, the database is not closed and dismounted first.

Dealing with Multiple Instances on One Server

It's common to have more than one Oracle database running on the same server. In a UNIX environment, this is the norm. When you're managing several databases on one server and you want to start or stop one of them, you need a way to designate which one it is that you want to stop.

On a Windows NT machine, you can set the ORACLE_SID environment variable to indicate which database you want to start or stop. To set ORACLE_SID, you have to use the SET command. Here's an example showing the ORACLE_SID environment variable being set to PROD:

```
C:\>SET ORACLE_SID=PROD

C:\>
```

You have to issue the SET command from a Command Prompt window. Once you've done that, any Oracle utilities that you run from that window will operate on the database instance that you've specified. The ORACLE_SID setting stays in effect until you issue another SET command that changes it or until you close the window.

Note The ORACLE_SID setting applies only to commands issued from the same Command Prompt window. Under Windows NT, it is entirely feasible to have multiple Command Prompt windows open, each with a different ORACLE_SID value.

You can view the current ORACLE_SID setting at any time by issuing the SET command with just the variable name as an argument, as shown in this example:

```
C:\>SET ORACLE_SID
ORACLE_SID=PROD
```

On UNIX systems, you also have to set the value of ORACLE_SID. However, UNIX systems usually provide an Oracle script named oraenv that you should run to do this. Here's an example showing oraenv being used to change the Oracle SID being changed on a UNIX box:

```
$ . oraenv
ORACLE_SID = [TEST] ? PROD
```

Note Notice the space between the dot and the oraenv command. If you're using the UNIX Korn shell, that space is significant. Omit it, and your new setting won't stick.

If you look carefully at your environment variable settings, you will see that oraenv changes more than just the ORACLE_SID. It also changes ORACLE_HOME and your path setting as well.

Once you've used either the SET command or the oraenv command to change your ORACLE_SID setting, you can run SQL*Plus as shown previously, and issue the STARTUP command to start that database.

Specifying a Nondefault Parameter File

When you use SQL*Plus to start a database, SQL*Plus looks for a text file known as the *database parameter file*, reads a number of parameter settings from that file, and uses those to control various aspects of the database instance being started. Sometimes this file is referred to as the init.ora file, or even just the init file. Chapter 3, “Oracle8i Architecture,” has more information about the parameter file, what it contains, and how it relates to the other database files.

SQL*Plus expects parameter files to be in a specific location and to be named according to a specific naming convention. The default location and name are different for UNIX and Windows NT. One of the following will apply:

Unix:	$ORACLE_HOME/dbs/initSID.ora
Windows NT:	c:\oracle\ora81\database\initSID.ora

Note

The `c:\oracle\ora81` portion of the path under Windows NT may be different if you installed Oracle on a drive other than `c:`, or if you overrode the default directory names during the installation.

As long as you place your parameter files where SQL*Plus expects them, SQL*Plus will find them, and your `STARTUP` commands will remain very simple. If, for some reason, you decide to place your initialization files somewhere else or to name them differently, then you must tell SQL*Plus where to look. Use the `PFILE` parameter of the `STARTUP` command to do this. The following example shows you how:

```
STARTUP PFILE=$ORACLE_BASE/admin/PROD/pfile/initPRODnight.ora
```

In this example, the `PFILE` parameter points SQL*Plus to a special parameter file used only for nightly processing.

The `PFILE` parameter is handy if you have more than one parameter file that you use to start a database. The `PFILE` parameter is also handy if you are starting a database remotely.

Starting and Stopping an Instance on a Remote Server

You can use SQL*Plus to start or stop a database on another computer. You won’t do this often, but it is possible. Generally, if the other computer is running UNIX, you’ll find it easier to Telnet in and run SQL*Plus on the server. If the other machine is Windows NT, however, Telnet won’t be available, so you either have to know how to start or stop the database remotely, or you have to go to the machine.

To start or stop an instance remotely, the following prerequisites apply:

1. You need to have Net8 configured so that you can connect to the remote machine. Chapter 5, “Configuring Net8,” can help you with this.

2. You need to have created a password file for the instance that you want to start, and you need to have been granted either the SYSDBA or SYSOPER role. Chapter 4, “Managing the Database Password File,” can help you with this.
3. You need access to the database’s parameter file. This applies only to starting. The parameter file isn’t used when stopping an instance. You either need a copy of the parameter file on your local PC, or it needs to be accessible over the network.

With these prerequisites out of the way, the process for starting a remote instance is much the same as for starting an instance on your local machine. The only differences will be that you have to specify the Net8 service name in your connect statement, and you will likely need to use the startup command’s `PFILE` parameter to specify the location of the database parameter file. Listing 2-2 shows the `bible_db` database being started from a remote PC:

Listing 2-2: **Starting the bible_db database from a remote PC**

```
C:\>sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Sun Jun 27 17:09:24 1999

(c) Copyright 1999 Oracle Corporation.  All rights reserved.
SQL> connect system/manager@bible_db as sysdba
Connected to an idle instance.
SQL> startup pfile=o:\admin\jonathan\pfile\init.ora
ORACLE instance started.

Total System Global Area   38322124 bytes
Fixed Size                    65484 bytes
Variable Size              21405696 bytes
Database Buffers           16777216 bytes
Redo Buffers                  73728 bytes
Database mounted.
Database opened.
SQL>
```

Stopping an instance remotely is even more similar to stopping an instance on your local machine. The `SHUTDOWN` command is the same. This next example shows how you would shut down the instance that was started in the previous example:

```
C:\>sqlplus /nolog

SQL*Plus: Release 8.1.5.0.0 - Production on Sun Jun 27 22:36:05 1999
```

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

SQL> connect system/manager@bible_db as sysdba
Connected.
SQL> shutdown
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
```

If you intend to start and stop your database remotely like this, you need to decide how you want to handle the parameter file. Do you want to have just one parameter file, placed on a shared drive so that a remote machine can access it, or do you want to make local copies of the parameter file on each PC that you might use to start the database? If you are going to maintain multiple copies of the parameter file, how will you keep them in sync with each other? You also have to think about include files. Parameter files may contain `ifile` commands that link to other parameter files. Those references are resolved by SQL*Plus.

If you place your parameter file on the network and access it from a remote PC, you have to be sure that the `ifile` file references can be resolved from that PC, as well as from the server. For example, your parameter file might include a file named `e:\admin\jonathan\pfile\configjonathan.ora`. If your PC maps the server's E drive to your O drive, then the `ifile` directive will fail. One way to deal with this would be to map the server's E drive to your local E drive. That way the drive letters remain the same, regardless of whether you are starting the database from the server or your PC.

Using Server Manager to start and stop an instance

If you've been using Oracle for any length of time, you know that historically, Server Manager, not SQL*Plus, was the tool used to start and stop an Oracle instance. If you are using any release of Oracle prior to 8.1.5 (the 8i release), you won't be able to use SQL*Plus to start and stop a database instance. You'll have to use Server Manager instead.

Note

If you go back far enough, you'll find that SQL*DBA was the tool to use, but hopefully you aren't using any release of Oracle old enough for that to be the case.

The commands you use with Server Manager are the exact same as those that you use with SQL*Plus. Everything that you've read previously in this book about SQL*Plus, with one exception, is applicable to Server Manager as well. The one exception is the `/nolog` command-line option. You don't need it with Server Manager because Server Manager doesn't automatically attempt to log you on to a database.

Listing 2-3 shows Server Manager being used to start the `bible_db` database:

Listing 2-3: **Using Server Manager to start the bible_db database**

```
C:\>svrmgrl

Oracle Server Manager Release 3.1.5.0.0 – Production

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

```
ORA-12560: TNS:protocol adapter error
SVRMGR> connect system/manager@bible_db.gennick as sysdba
Connected.
SVRMGR> startup pfile=o:\admin\jonathan\pfile\init.ora
ORACLE instance started.
Total System Global Area                        38322124 bytes
Fixed Size                                         65484 bytes
Variable Size                                   21405696 bytes
Database Buffers                                16777216 bytes
Redo Buffers                                       73728 bytes
Database mounted.
Database opened.
SVRMGR>
```

Note The ORA-12560 error that you see in this example occurs because NT was used, and because the ORACLE_SID environment variable was not set. It's safe to ignore the error in this case because the `connect` command includes a Net8 service name.

In this example, the `svrmgrl` command was used to start Server Manager. That command works with any UNIX release of Oracle, as well as with the 8.1.5 release on Windows NT. Previous releases of Oracle for Windows NT had the release number attached to the executable name, so the command would be `svrmgr30` for the 8.0.*x* release, `svrmgr23` for the 7.3.*x* release, and so forth.

Oracle is still shipping Server Manager with Oracle8i and will continue to do so in the future (probably for all the 8.1.*x* releases). However, Server Manager is now considered a deprecated feature, and Oracle intends to remove it someday, leaving SQL*Plus as the only command-line interface into Oracle. Therefore, avoid using Server Manager and instead get used to using SQL*Plus.

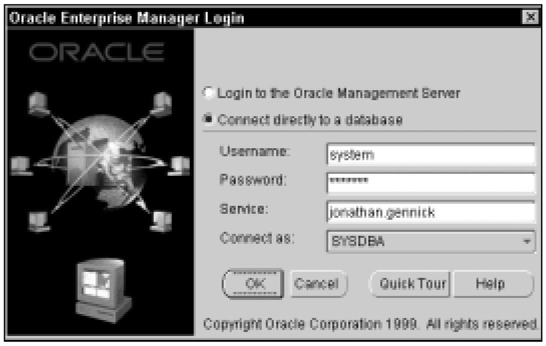
Using Instance Manager to start and stop an instance

You can also use Oracle's Instance Manager, part of Enterprise Manager's DBA Management Pack, to start or stop an Oracle instance. You can run Instance Manager as a stand-alone application, or you can run it in conjunction with an

Oracle Management server. If you run it as a stand-alone application, and if you run it from a machine other than the database server, all the prerequisites for starting a remote database need to be in place. Your database needs to have a password file, Net8 needs to be configured, and you need access to a database user who has been granted the SYSDBA or SYSOPER role.

Starting an Instance

First start the Instance Manager application. Click Start, point to Programs, point to Oracle-OraOEM, point to DBA Management Pack, and select Instance Manager. You will see a login screen similar to the one shown in Figure 2-6.


Oracle Enterprise Manager Login
ORACLE
Login to the Oracle Management Server
OK
Cancel
Quick Tour
Help

Connect directly to a database

Username: system
Password: *******
Service: jonathan.gennick
Connect as: SYSDBA

Copyright Oracle Corporation 1999. All rights reserved.

Figure 2-6: The Oracle Enterprise Manager Log in dialog box

Be sure to click the option labeled Connect Directly to a Database, and then enter your username and password in the appropriate text boxes. To start a database instance, you need to connect as either SYSOPER or SYSDBA, so select one of those options in the Connect as drop-down list box. Figure 2-6 shows everything filled out correctly for the system user to log on to a database. When you have everything correct, click OK.

Note If you are running on Windows NT, you must have a password file for your database in order to log on as SYSDBA or SYSOPER. Alternatively, you must be logged on as a Windows NT administrator.

After connecting to your instance, you'll see a screen similar to the one shown in Figure 2-7 but without the traffic light on the right-hand side. The traffic light shows the current status of your instance. To see it, click the database icon in the left-hand pane. That's the icon shown highlighted in Figure 2-7.

The traffic light not only shows you the status of your instance, but it lets you start and stop it as well. If your instance is stopped, the red light will be lit and the Shutdown option will be selected.

Note Just like when you use SQL*Plus to start a remote instance, you need to be able to pass a parameter file name to Instance Manager. You can keep a copy of the parameter file on your local PC for this purpose, or you can access one on the network.

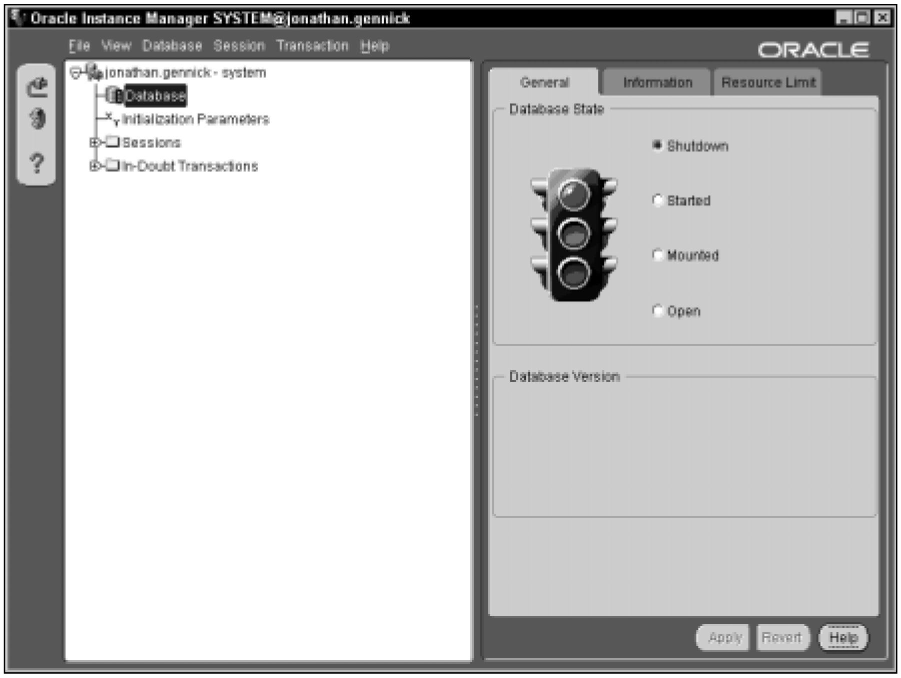


Figure 2-7: Instance Manager's traffic light

To start your instance, select the Open option and click the Apply button at the bottom of the window. Instance Manager will open a dialog box prompting you for the name of the parameter file for the instance that you are starting. Type the name and location of your parameter file, click OK, and Instance Manager will start the instance.

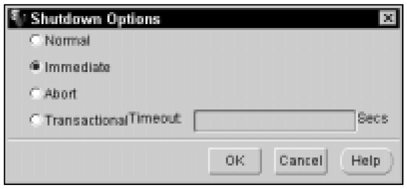
Tip

You don't have a copy of your parameter file handy? You can use Instance Manager to create one, but you need to do it while the database is open. Click the icon labeled Initialization Parameters, and then click the Save button.

Stopping an Instance

Using Instance Manager to stop an instance is pretty much the reverse of using it to start one. It's actually a bit easier because you don't have to worry about the parameter file. To start it, click the database icon in the left pane to get the traffic light in the right pane. Next, select the Shutdown option and click Apply. Instance Manager will present you with the shutdown options shown in Figure 2-8.

These are the same options — normal, immediate, abort, and transactional — that you have when you use SQL*Plus to shut down an instance. Select the option for the shutdown mode that you want and click OK. Instance Manager will close the database and shut down the instance.

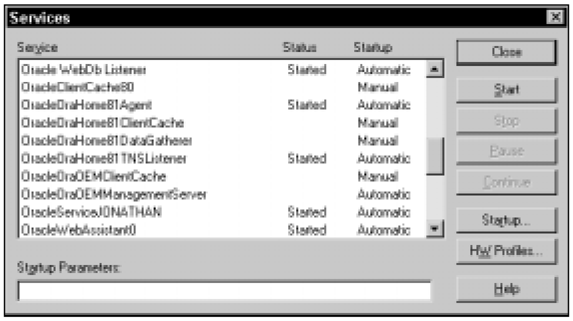


Normal [✓ Immediate] Abort Transactional
Timeout: Secs

Figure 2-8: Instance Manager's Shutdown Options dialog box

Using the services control panel to start and stop an instance

Under Windows NT, the processes that make up an Oracle instance are implemented as a standard Windows NT service. You can see this service, as well as other Oracle-related services, listed in the Services control panel. Figure 2-9 shows the Oracle services on an NT server.



Service	Status	Startup
Oracle WebDb Listener	Started	Automatic
OracleClientCache80		Manual
OracleOraHome81Agent	Started	Automatic
OracleOraHome81ClientCache		Manual
OracleOraHome81DataGatherer		Manual
OracleOraHome81TNSListener	Started	Automatic
OracleOraOEMClientCache		Manual
OracleOraOEMManagementServer		Automatic
OracleServiceJONATHAN	Started	Automatic
OracleWebAssistant0	Started	Automatic

Startup Parameters:

Figure 2-9: The Oracle services shown in the Services control panel applet

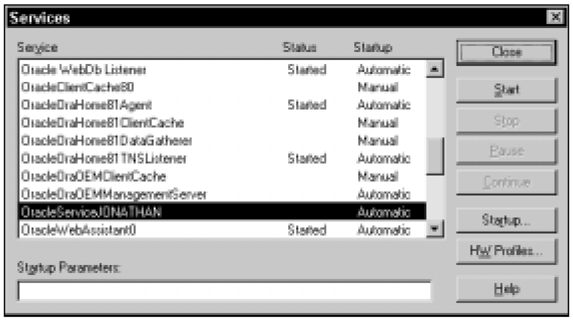
Like any other Windows NT service, you can start and stop the Oracle services via the Services control panel. In addition, you can also use the Oracle-supplied program named `oradim` to start and stop the services. Starting or stopping the service for a database instance implies starting or stopping the instance itself.

Note

Although stopping a service implies stopping an instance, the converse is not true. Stopping an instance does not stop the associated service.

Starting or stopping an Oracle service using the Services control panel is an almost trivial exercise. Figure 2-10 shows you what you will see from the control panel if the service for the instance named jonathan is not running.

To start the service, all you need to do is click the service once to highlight it, and then click the Start button. You'll see a spinning dial while Windows NT attempts to start the service, and then the status will change to read "started." To stop the service, simply click the Stop button.

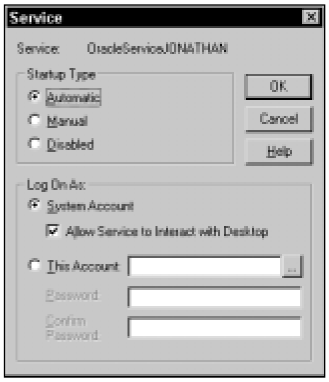


Service	Status	Startup
Oracle WebDb Listener	Started	Automatic
OracleClientCache80		Manual
OracleOraHome81Agent	Started	Automatic
OracleOraHome81ClientCache		Manual
OracleOraHome81DataGatherer		Manual
OracleOraHome81TNSListener	Started	Automatic
OracleOraOEMClientCache		Manual
OracleOraOEMManagementServer		Automatic
OracleServiceJONATHAN		Automatic
OracleWebAssistant0	Started	Automatic

Figure 2-10: The Oracle service for jonathan is not running.

Setting the Startup Mode for an Instance

One other task you can perform from the Services control panel is to set the startup type for an Oracle database instance's service. This controls whether the database starts automatically whenever the server starts. Select the service you are interested in and click the Startup button. This opens the Service dialog box shown in Figure 2-11.



Service: OracleServiceJONATHAN

Startup Type
[✓ Automatic] Manual Disabled

Log On As:
[✓ System Account]
[✓] Allow Service to Interact with Desktop
This Account:
Password:
Confirm Password:

Figure 2-11: Setting the startup type for a service

As you can see in Figure 2-11, you have three choices for Startup Type: Automatic, Manual, and Disabled. Automatic means that Windows NT will automatically start the service. The manual setting means that you will have to manually start the service. When a service is disabled, Windows NT won't allow you to start it at all.

Using oradim to Start and Stop a Service

With Windows NT distribution, Oracle includes a utility named `oradim` that you can use to start and stop the service for a database instance. The `oradim` utility is useful if you are writing a batch file and you need to include a command to stop or start a database. The syntax for the `oradim` commands used to start and stop a database instance's service appears as follows:

```
ORADIM -STARTUP -SID sidname [-USRPWD password]
       [-STARTTYPE srvc|inst|srvc,inst] [-PFIFLE filename]
ORADIM -SHUTDOWN -SID sidname [-USRPWD password]
       [-SHUTTYPE srvc|inst|srvc,inst] [-SHUTMODE a|i|n]
```

The following list describes the elements in this syntax:

- ✥ `oradim`: The command for starting and stopping a database.
- ✥ `STARTUP`: Indicates that you want to start a service or an instance.
- ✥ `SID sidname`: Specifies the instance whose service you want to start.
- ✥ `USRPWD password`: Specifies the password for the internal user. You won't need this parameter if you are logged on to NT as an administrator.
- ✥ `STARTTYPE srvc|inst|srvc,inst`: Indicates what to start. Your choices are as follows:
  - `srvc` — Starts just the service.
  - `inst` — Starts the instance. The service must already be running for this to work.
  - `srvc,inst` — Starts both the service and the instance.
- ✥ `SHUTTYPE srvc|inst|srvc,inst`: Indicates what to stop. Your choices are the same as for `STARTTYPE`.
- ✥ `PFILE filename`: Points to the parameter file for the instance. You don't need this if the parameter file is in the default location where Oracle expects it to be.
- ✥ `SHUTMODE a|i|n`: Indicates the shutdown mode to use when stopping the instance. Your choices are as follows:
  - `a` — An abort
  - `i` — An immediate shutdown
  - `n` — A normal shutdown
- ✥ The `oradim` utility doesn't support the transaction shutdown option.

To use `oradim` to start the NT service named OracleServiceJONATHAN shown in Figure 2-11, you would issue a command like this:

```
oradim -startup -sid jonathan
```

To stop the same instance, you would issue this command:

```
oradim -shutdown -sid jonathan
```

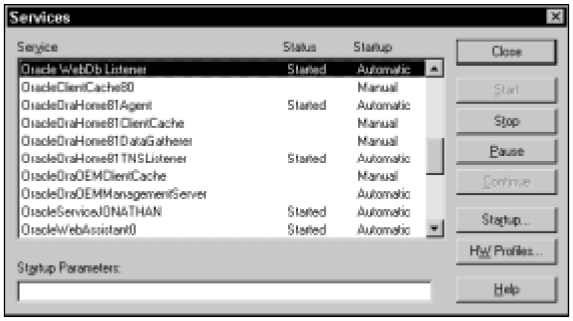
According to the documentation, it is possible to use `STARTTYPE srvc` to start only the service, or `STARTTYPE inst` to start only the instance. However, reality doesn't seem to match the documentation. In release 8.1.5, at least, you can't use this command to start and stop an instance independent of the service.

Starting and Stopping WebDB Listener

You have two options for starting and stopping WebDB Listener. One is to use the Services control panel to start and stop the service named Oracle WebDb Listener. The other way is to use the Windows NT `net start` and `net stop` commands.

Using the services control panel to start and stop WebDB Listener

The easiest way to control WebDB Listener is to use the Services control panel. Figure 2-12 shows the control panel with the WebDB Listener service at the top of the Services control panel.



Service	Status	Startup
Oracle WebDb Listener	Started	Automatic
OracleClientCache80		Manual
OracleOraHome81Agent	Started	Automatic
OracleOraHome81ClientCache		Manual
OracleOraHome81DataGatherer		Manual
OracleOraHome81TNSListener	Started	Automatic
OracleOraOEMClientCache		Manual
OracleOraOEMManagementServer		Automatic
OracleServiceJONATHAN	Started	Automatic
OracleWebAssistant0	Started	Automatic

Figure 2-12: The WebDB Listener process in the Services control panel

The process for starting and stopping the WebDB Listener service is the same as for any other service. First highlight the service, and then click the Start button to start it or click the Stop button to stop it. You can also click the Startup button to open up a window that lets you set the startup type to either automatic or manual. When the startup type is automatic, Windows NT will start the service whenever the system starts; otherwise, you will have to start the service manually.

Using the wdblsnr utility to start and stop WebDB Listener

To control WebDB Listener from the command line, or from a batch file, use the `net start` and `net stop` commands. The command to start the listener is the following:

```
net start "Oracle WebDb Listener"
```

The command to stop the listener is similar, and looks like this:

```
net stop "Oracle WebDb Listener"
```

Because the WebDB Listener name contains spaces, you must enclose it in quotes. The name needs to match what is shown in the Services control panel.

Note

While the name used with the `net start` and `net stop` commands must match a service name as shown in the Services control panel, Windows NT is not case sensitive. A value of “oracle webdb listener” or “ORACLE WEBDB LISTENER” will work just as well as “Oracle WebDb Listener.”

Using Oracle8i’s Online Documentation

Most of the utilities that come with Oracle8i have some sort of built-in online help. For the command-line utilities, the help is often a bit skimpy, but the GUI utilities, particularly Enterprise Manager, have fairly extensive online documentation.

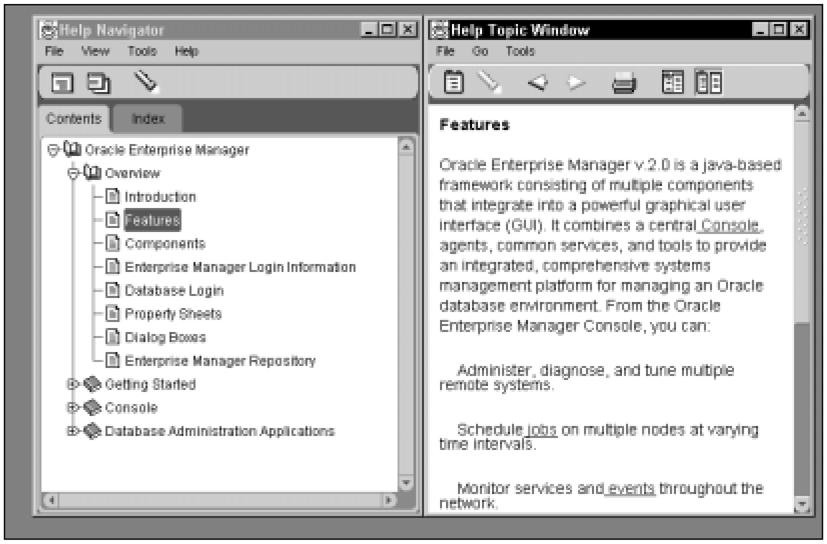
You can access Oracle’s online-help style documentation in the following two ways:

- Click the Help button you find in almost every Oracle8i window.
- Click one of the help icons in the Oracle8i program groups.

Most of the time, you access online help by clicking the Help button from an application screen. Sometimes, however, Oracle includes an icon on the Start menu that takes you directly to the online help for a particular topic. For example, there are icons in the Network Administration program group that take you to online help for Oracle’s ODBC drivers.

If you’re used to standard Windows-based online help, you’re in for a pleasant surprise when you start using help from any of Oracle’s new Java-based utilities. You view the online help in Java-based programs, such as the Enterprise Manager applications, by using two windows. One window is a Help Navigator window. The other is a Help Topic window. These windows are shown in Figure 2-13.

The Help Navigator window allows you to browse help topics using a tree-style menu, and it also contains a searchable index. When you click on a topic in the Help Navigator, you’ll see the help text for that topic in the topic window.



- Oracle Enterprise Manager
  - Overview
    - Introduction
    - Features
    - Components
    - Enterprise Manager Login Information
    - Database Login
    - Property Sheets
    - Dialog Boxes
    - Enterprise Manager Repository
  - Getting Started
  - Console
  - Database Administration Applications

Features

Oracle Enterprise Manager v.2.0 is a java-based framework consisting of multiple components that integrate into a powerful graphical user interface (GUI). It combines a central Console, agents, common services, and tools to provide an integrated, comprehensive systems management platform for managing an Oracle database environment. From the Oracle Enterprise Manager Console, you can:

Administer, diagnose, and tune multiple remote systems.

Schedule jobs on multiple nodes at varying time intervals.

Monitor services and events throughout the network.

Figure 2-13: The Help Navigator and Help Topic windows

In addition to the online help, you can access Oracle's full manual set using a Web browser. Like the online help, you navigate the online manual set using a tree-style menu. Here's how you bring up the online manual set and navigate to the *Oracle8i SQL Reference* manual:

1. Place the Oracle8i distribution CD in your CD-ROM drive. You don't need to do this if you installed the online help to your local hard drive, but the default installation option leaves the help files on the CD.
2. From the Start menu, select Programs, OraHome81, and Documentation. Your Web browser will open, and after a few more seconds, the Oracle Information Manager window will open. Your screen will now look similar to Figure 2-14.

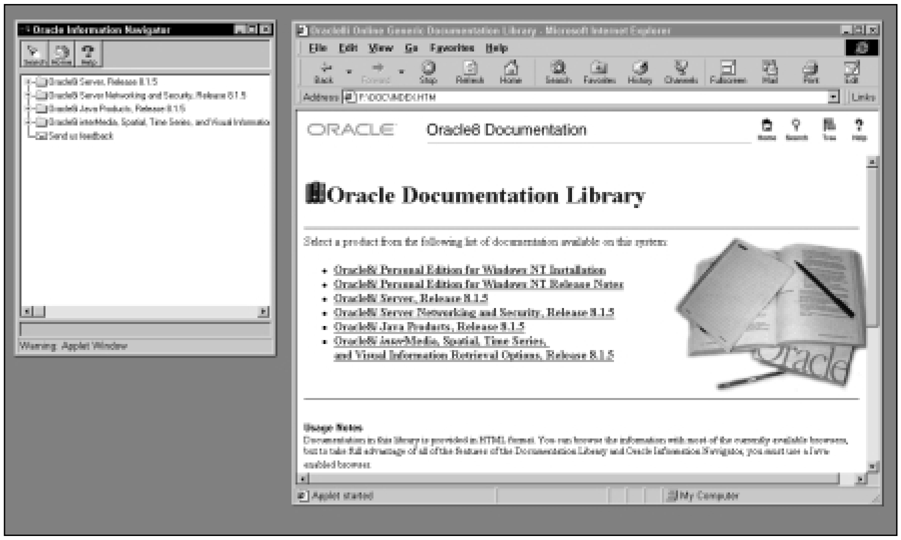
Note

Oracle also makes available a PDF version of the documentation that you can read using Adobe Acrobat. To use the PDF version, start by opening the file named index.pdf in the doc\server.815 directory on the Oracle8i distribution CD.

Note

If you don't have a Java-enabled browser, or if you have a very old browser, you won't see the Oracle Information Navigator window. In addition, you might frequently experience problems with the Oracle Information Navigator window starting improperly. You might need to try two or three times, terminating and restarting your browser each time, before it comes up properly.

At this point, you can browse the manual set either by using the tree-style navigator or by clicking the HTML links in the browser window.


Oracle Information Navigator
Oracle8i Server, Release 8.1.5
Oracle8i Server Networking and Security, Release 8.1.5
Oracle8i Java Products, Release 8.1.5
Oracle8i interMedia, Spatial, Time Series, and Visual Informatio
Send us feedback
Warning: Applet Window
Oracle8i Online Generic Documentation Library - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail Print Edit
Address F:\DOC\INDEX.HTM
Links
ORACLE
Oracle8 Documentation
Home Search Tree Help
Applet started
My Computer

Oracle Documentation Library

Select a product from the following list of documentation available on this system:

- Oracle8i Personal Edition for Windows NT Installation
- Oracle8i Personal Edition for Windows NT Release Notes
- Oracle8i Server, Release 8.1.5
- Oracle8i Server Networking and Security, Release 8.1.5
- Oracle8i Java Products, Release 8.1.5
- Oracle8i interMedia, Spatial, Time Series, and Visual Information Retrieval Options, Release 8.1.5

Usage Notes
Documentation in this library is provided in HTML format. You can browse the information with most of the currently available browsers, but to take full advantage of all of the features of the Documentation Library and Oracle Information Navigator, you must use a Java-enabled browser.

Figure 2-14: Oracle's online documentation

3. To find the *Oracle8i SQL Reference* manual, click the plus (+) sign next to the folder labeled Oracle8i Server, Release 8.1.5.
4. Open the Reference folder. You should see a list of manuals, each with an associated blue book icon.
5. Expand the SQL Reference Manual icon by either clicking the plus sign next to the manual name or by double-clicking the manual name. You will see a list of manual sections.
6. Double-click any of the manual sections, and you will see the text for that section appear in your browser window.
7. To shut down the documentation, close your browser as you normally would. The Oracle Information Navigator window will automatically close as well.

Tip

Install the HTML documentation on your local workstation. Even though it takes upwards of 140MB of space, you'll find it well worthwhile for the convenience of having the documentation always available, regardless of whether you have the CD handy.

Many people don't like to read the documentation from the CD. If you find yourself referring to a particular manual frequently, consider purchasing a printed copy from Oracle. Given the number of companies that don't like to spend money on manuals for their employees, having the documentation available electronically is a real blessing. Do everything you can to make it available to your developers and fellow database administrators.

Summary

In this chapter, you have learned:

✥ Oracle applications are implemented as client/server applications. The database server runs independent of the clients, accepts requests from clients, processes those requests, and sends results back to clients.

✥ An Oracle server software installation consists of the database software, Net8 software, and a collection of utilities.

✥ An Oracle client software installation consists of Net8 software, utilities, and possibly the Enterprise Manager applications.

✥ SQL*Plus, `oradim`, and Instance Manager are all tools that you can use to start and stop an Oracle database instance.

✥ You can stop and start WebDB Listener from the Services control panel. If you prefer to use commands, you may also stop and start it using the `net stop` and `net start` commands.

✥ The entire Oracle8i manual set is available in HTML format. Consider installing it on your workstation so that you can easily access it when needed. Make it available to developers and other database users.

✥ ✥ ✥