CHAPTER 2

DB2 Server Management

This chapter will prepare you to configure and manage DB2 servers, instances, and databases. You will learn to use many of the DB2 autonomic computing features to improve system availability and performance. By the end of the chapter, you will also be able to install and configure IBM Data Studio to perform database administration activities, such as database maintenance job schedules and Visual Explain plan generation.

Exam Objectives

- ✓ Demonstrate the ability to create and manage DB2 instances
- ✓ Demonstrate the ability to view and modify DB2 system registry variables
- ✓ Demonstrate the ability to view and modify DB2 database manager configuration information
- ✓ Demonstrate the ability to view and modify database configuration information
- ✓ Demonstrate the ability to gain exclusive control of a database
- ✓ Demonstrate the ability to use automatic maintenance
- ✓ Demonstrate the ability to use the STMM
- ✓ Demonstrate the ability to use the Configuration Advisor
- ✓ Demonstrate the ability to use automatic storage features
- ✓ Demonstrate the ability to use data compression features
- ✓ Demonstrate the ability to throttle utilities
- ✓ Demonstrate the ability to schedule and manage jobs by using IBM Data Studio

Working with Instances

DB2 sees the world as a hierarchy of objects. *Workstations* or *servers* on which DB2 has been installed occupy the highest level of this hierarchy. When any edition of DB2 is installed on a workstation, program files for a background process known as the *DB2 database manager* are physically copied to a specific location on that workstation, and in most cases, an instance of the DB2 database manager is created.

Instances occupy the second level in the hierarchy and are responsible for managing system resources and databases that fall under their control. Although only one instance is created initially, several instances can coexist on a single server. Each instance behaves like a separate installation of DB2, even though all instances within a system share the same DB2 database manager program files (unless each instance is running a different version of DB2). And although multiple instances share the same binary code, each runs independently of the others and has its own environment, which can be modified by altering the contents of its associated configuration file.

Databases make up the third level in the hierarchy and are responsible for managing the storage, modification, and retrieval of data. Like instances, databases work independently of each other. Each database has its own environment that is controlled by a set of configuration parameters, as well as its own set of grantable authorities and privileges to govern how users interact with the data and database objects it controls. Figure 2.1 shows the hierarchical relationship between systems, instances, and databases.

Although most DB2 environments consist of one instance per server, at times it is advantageous to create multiple instances on the same physical server. Reasons for creating multiple instances include the following:

- To separate your development environment from your production environment
- To obtain optimum performance for special applications (for example, you may choose to create an instance for one or more applications, and then fine-tune each instance specifically for the applications it will service)
- To prevent database administrators from accessing sensitive data (for example, a company's payroll database could reside in its own instance, in which case owners of other databases in other instances on the same server would be unable to access payroll data)

Obviously, multiple instances will require additional system resources such as disk space, memory, and CPUs based on the workload you run on the databases created on the additional instance.

As you might imagine, DB2 provides several commands for creating and managing instances. Table 2.1 shows these commands, referred to as system commands because they are executed from the system command prompt rather than from the DB2 Command Line Processor (CLP).

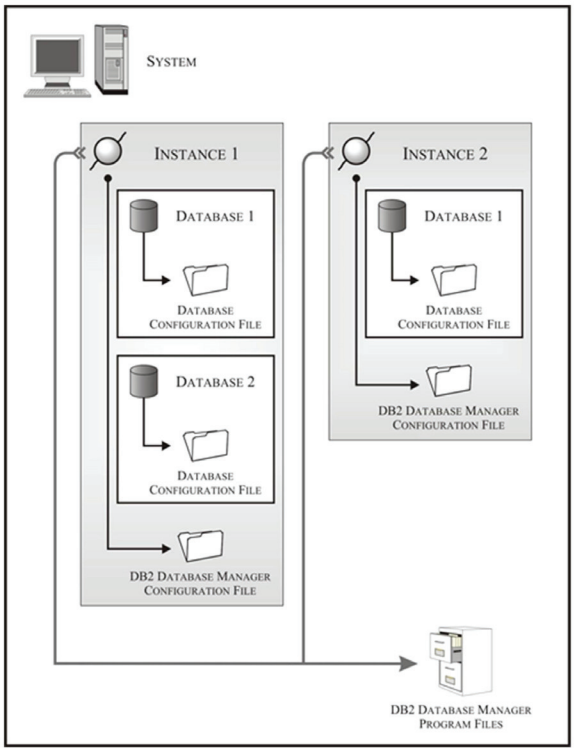

Figure 2.1: Hierarchical relationship between systems, instances, and databases

Table 2.1: DB2 instance management commands

Command	Purpose
db2icrt [InstanceName]	Create a new instance
db2idrop [InstanceName]	Delete (drop) an existing instance
db2ilist	List all instances that have been defined within one installation
db2ckupgrade [DatabaseName]	Verify that local databases are ready to be upgraded
db2iupgrade [InstanceName]	Upgrade an existing instance to a newer version of DB2
db2iupdt [InstanceName]	Update an existing instance to exploit new functionality that is provided when product fix packs are installed (also used to convert a 32-bit instance to a 64-bit instance)
db2start	Start the DB2 database manager background processes for the current instance
db2stop	Stop the DB2 database manager background processes for the current instance

Note: Although Table 2.1 presents the basic syntax for the instance management commands, the actual syntax supported may be more complex. To view the complete syntax for a specific DB2 command or to obtain more information about a particular command, refer to the IBM DB2 Version 10.1 Command Reference product documentation.

Attaching to an Instance

The default instance for a system is defined by the DB2INSTANCE environment variable, and often, this is the instance that all instance-level operations are performed against. If you need to perform an operation against a different instance, you must first change the value assigned to the DB2INSTANCE variable by executing the command "set DB2INSTANCE=[*InstanceName*]" (export DB2INSTANCE=[*InstanceName*] on Linux and UNIX), where *InstanceName* is the name assigned to the instance that you want to make the default instance, or you must *attach* to that instance.

Applications and users can attach to any instance by executing the ATTACH command. The basic syntax for this command is:

```
ATTACH TO [InstanceName]
USER [UserID] USING [Password]
```

where:

InstanceName	Identifies the name to assign to the instance to make an attachment (this instance must have a matching entry in the local node directory)
UserID	Identifies the user (by authorization ID) under whom the instance will attach
Password	Identifies the password that corresponds to the specified authorization ID

Thus, if you want to attach to an instance named db2inst1 by using the authentication ID db2admin and the password ibmdb2, you can do so by executing an ATTACH command:

```
ATTACH TO db2inst1 USER db2admin USING ibmdb2
```

Detaching from an Instance

Once you have made an attachment to an instance and all necessary tasks have been performed against that instance, you terminate the instance attachment if it is no longer needed. By terminating an instance attachment, you eliminate the potential to accidentally perform new

operations against the wrong instance. The easiest way to terminate an attachment to an instance is by establishing an attachment to another one. That is because an application or user can attach to only one instance at a time—if an attachment is made to another instance, the current instance attachment is automatically terminated.

Applications and users can also detach from an instance by executing the DETACH command. The syntax for this command is:

```
DETACH
```

As you will notice, the DETACH command requires no additional parameters.

Starting and Stopping an Instance

The DB2 database manager background processes that are associated with a particular instance must be active and ready to process requests before any operation can be performed against the instance or a database under the instance's control. If they are not already running, you can start these background processes by executing the START DATABASE MANAGER command. The basic syntax for this command is:

```
START [DATABASE MANAGER | DB MANAGER | DBM]
```

or

```
db2start
```

Thus, to start the DB2 database manager background processes for the default instance, you execute a command that looks something like this:

```
START DATABASE MANAGER
```

If at any time you want to stop the DB2 database manager background processes, you can do so by executing the STOP DATABASE MANAGER command. The basic syntax for this command is:

```
STOP [DATABASE MANAGER | DB MANAGER | DBM]
<FORCE>
```

or

```
db2stop
```

Therefore, to stop the DB2 database manager background processes for the default instance, execute a command that looks something like this:

```
STOP DATABASE MANAGER
```

Note: On Linux, UNIX, and Windows operating systems, instances created using db2icrt are set to a manual start after the system reboot. You can enable instances to start automatically by using the following steps.

On the Windows platform, change the property of the DB2 services in the services panel to automatic. On UNIX and Linux platforms, run the command db2iauto –on <*InstanceName*>, where *InstanceName* is the name of the instance that needs auto-starting.

Quiescing an Instance

Because any number of users can be granted access to an instance or one or more databases under an instance's control, it can be difficult, if not impossible, to coordinate the work efforts of everyone who is using a specific instance at a given point in time. This can present a problem if a database administrator needs exclusive access to a particular instance for a short time (for example, to perform a maintenance operation). Therefore, individuals holding the proper authority can place an instance in a "restricted access" or "quiesced" state. When an instance is quiesced, all users are forced off the instance, all active transactions are immediately rolled back, and all databases under the instance's control are put into quiesced mode. You can place instances (and databases) in quiesced mode by executing the QUIESCE command. The basic syntax for this command is:

```
QUIESCE [INSTANCE [InstanceName ]
<USER [UserName] | GROUP [GroupName]>
[RESTRICTED ACCESS] [IMMEDIATE | DEFER <WITH TIMEOUT [Minutes]>]
<FORCE CONNECTIONS>
```

where:

InstanceName	Identifies the name to assign to the instance to place in quiesced mode
UserName	Identifies the name of a specific user who has permission to access the specified instance or database while it is in quiesced mode
GroupName	Identifies the name of a specific group of users that has permission to access the specified instance or database while it is in quiesced mode
Minutes	Specifies a time, in minutes, to wait for applications to commit their current transactions before quiescing the instance; if no value is specified, the default value is 10 minutes

Thus, to place an instance named db2inst1 into quiesced mode immediately, but allow a user named db2admin to continue to have access to it, you execute a QUIESCE command:

```
QUIESCE INSTANCE db2inst1 USER db2admin IMMEDIATE
```

Eventually, you will need to return the instance or database put into quiesced mode to a normal state. You can remove instances and databases from quiesced mode by executing the UNQUIESCE command. The basic syntax for this command is:

```
UNQUIESCE [INSTANCE [InstanceName] | DB]
```

where:

InstanceName	Identifies the name assigned to the instance that is to be taken out of quiesced mode

Therefore, if you want to take an instance named db2inst1 out of quiesced mode, you can do so by executing an UNQUIESCE command that looks like this:

```
UNQUIESCE INSTANCE db2inst1
```

You can also quiesce and unquiesce instances and databases by selecting the **Quiesce** or **Unquiesce** action from either the **Instances** menu or the **Databases** menu in IBM Data Studio. Figure 2.2 shows the Data Studio menu items that must be selected to activate the Quiesce instance dialog.

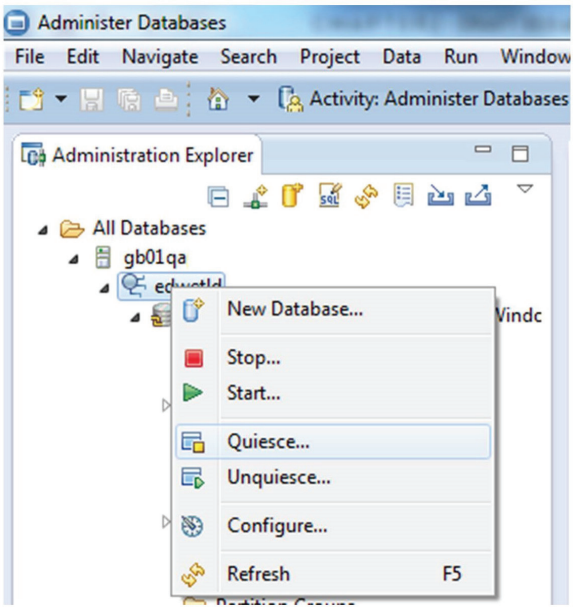

Figure 2.2: Invoking the Quiesce dialog from IBM Data Studio

It is important to note that only users with System Administrator (SYSADM) or System Control (SYSCTRL) authority are allowed to quiesce an instance. Once an instance is in a quiesced state, only users with System Administrator (SYSADM), System Control (SYSCTRL), or System Maintenance (SYSMAINT) authority; users who are members of a specified group (if a group name was specified when the instance was placed in quiesced mode); and users with a specified user name (if a user name was specified when the instance was placed in quiesced mode) can connect to the instance.

Similarly, only users with System Administrator (SYSADM) or Database Administrator (DBADM) authority are permitted to quiesce a database. After a database is in a quiesced state, only users with System Administrator (SYSADM), System Control (SYSCTRL), System Maintenance (SYSMAINT), or Database Administrator (DBADM) authority; users who are members of a specified group (if a group name was specified when the database was placed in quiesced mode); and users with a specified user name (if a user name was specified when the database was placed in quiesced mode) can connect to the database.

The basic syntax for QUIESCE and UNQUIESCE database are:

```
QUIESCE DATABASE [IMMEDIATE | DEFER <WITH TIMEOUT [Minutes]>]
<FORCE CONNECTIONS>
```

The database connection must be established before running the QUIESCE command on the database. There are no additional parameters necessary for running the UNQUIESCE command:

```
UNQUIESCE DATABASE
```

Upgrading Instances

To upgrade your DB2 server from DB2 9.5, DB2 9.7, or DB2 9.8 to DB2 10.1, you must upgrade the instance by using the db2iupgrade command. If the DB2 servers are running on a release before DB2 9.5, migrate the instance by using the db2imigr command to the latest DB2 9.5 fix pack, and then follow the DB2 10.1 upgrade procedure.

Table 2.2 lists the four major steps to follow during the upgrade process (these steps are specifically for UNIX systems).

Table 2.2: DB2 10.1 upgrade procedure

Step	Task	Detailed Information
1	Prerequisites	List the db2 instances and the installation locations by using the db2ls command. Understand the compatibilities between the operating system and DB2 10.1. Check the known issues for DB2 on your platform by using the links below: ● AIX®: *www-01.ibm.com/support/docview.wss?uid=swg21165448* ● HP-UX: *www-01.ibm.com/support/docview.wss?uid=swg21257602* ● Solaris: *www-01.ibm.com/support/docview.wss?uid=swg21257606* ● Linux: *www.ibm.com/software/data/db2/linux/validate* Download the DB2 10.1 product and the license from IBM Passport Advantage®. Run the db2prereqcheck command on the server to check whether the system meets DB2 10.1 prerequisites. Extract all the registry, instance, and database parameter current values: db2set –all db2cfexp cfexp backup LIST DB DIRECTORY LIST NODE DIRECTORY LIST DCS DIRECTORY GET DBM CFG SHOW DETAIL cp -p ~/sqllib/userprofile userprofile.bak cp -p ~/sqllib/db2nodes.cfg db2nodes.cfg.bak

Table 2.2: DB2 10.1 upgrade procedure (continued)

Step	Task	Detailed Information
2	Preupgrade	Install the DB2 10.1 software on the server by using the db2setup silent installation procedure. You can make use of the install.rsp response file present in the samples directory to run through the silent installation. Perform the installation file validation by using the db2val command, which in turn does the following validations for you: ● Installation file sets ● Embedded run-time path for DB2 executables and libraries ● Accessibility to the installation path ● Accessibility to the /etc/services file Apply the DB2 license by using the db2licm command. Check the database upgrade by using the db2ckupgrade command. Increase the LOGSECOND database configuration parameter value to double the current. Update the database manager configuration parameter DIAGLEVEL to 4.
3	Upgrade	Upgrade the existing instance to the newer version; the commands are: db2 LIST APPLICATIONS db2 FORCE APPLICATIONS ALL db2 DEACTIVATE DATABASE <dbname> db2stop db2diag –A ipclean;ipcrm db2_kill exit As the root authority, run the db2iupgrade command to upgrade an instance from the old release to a new release: db2iupgrade –u <*FencedUser*> <*InstanceName*> where: ● *FencedUser* runs UDFs and stored procedures outside the address space used by the DB2 database. You can find the fenced user for the instance by using the db2pd command: db2pd -fmp | grep -i fenced Trusted Path: /db2home/db2inst1/sqllib/function/unfenced Fenced User: db2fence ● *InstanceName* is the instance you would like to upgrade. Start the DB2 database manager by using the db2start command. Perform the database upgrade by using the UPGRADE command: UPGRADE DATABASE <DatabaseName> where: DatabaseName is the name of the database you want to upgrade to the new release.
4	Postupgrade	Activate all the databases. Perform the RUNSTATS on all the tables, and rebind all packages. Update the DB2 database manager configuration parameter DIAGLEVEL to 3. Restart the instance by using the db2stop and db2start commands.

Dropping Instances

To drop your DB2 root instance, issue the db2idrop command; to drop nonroot instances, uninstall your DB2 database product. It is necessary to stop the instance before dropping it, and a good practice is to back up the $INSTHOME/sqllib directory before dropping the instance.

The instance drop command is:

```
db2idrop <InstanceName>
```

where:

InstanceName Identifies the name of the specific instance to upgrade; to list all the instances, use the db2ilist command on the server

A Word About the DB2 Administration Server (DAS)

The DB2 Administration Server (DAS) has been deprecated in DB2 9.7 and will receive no more enhancements. Furthermore, the DAS might be removed in the near future. The reason for discussing DAS here is simply for completeness of the topic.

The tools that used to come with DB2, such as the Control Center (a discontinued DB2 GUI tool), require a separate instance that operates independently of, yet concurrently with, all other instances that have been defined for a particular workstation. For this reason, a special instance, known as the *DB2 Administration Server (DAS)* instance, is also created as part of the DB2 installation process. In contrast to other instances, only one DAS instance can exist on a single workstation. The DB2 global-level profile registry variable DB2ADMINSERVER contains the name of the DAS instance that has been defined for a particular workstation.

Once created, the DAS instance runs continuously as a background process whenever the system it was created on is online; the DAS instance is usually activated automatically each time the workstation it resides on is started (or rebooted). Furthermore, the DAS instance must be running on every DB2 server that you wish to administer remotely. That is because, among other things, the DAS instance provides remote clients with the information needed to establish communications with other instances.

It is important to note that to administer a server from a remote client, a user must have System Administration (SYSADM) authority for the DAS instance used. Furthermore, once a remote instance and database have been registered on a client workstation, the user must hold the authorities and privileges needed to perform administrative tasks.

In addition to enabling remote administration of DB2 servers, the DAS instance assists the Control Center and the Configuration Assistant in the following:

- Providing job (task) management, including the ability to schedule and run user-defined shell scripts and batch files that contain both DB2 and operating system commands
- Scheduling jobs, viewing the results of completed jobs, and performing administrative tasks against jobs executed either remotely or locally (by using the Task Center)

Note: The Control Center tools and associated DB2 commands such as db2am, db2ca, db2cc, db2eva, db2hc, db2indbt, db2lc, and db2tc have been discontinued. In DB2 10.1 onward, it is advisable to use IBM Data Studio, which is discussed at the end of the chapter.

Configuring the DB2 System Environment

During normal operation, the DB2 database manager's behavior is controlled, in part, by a collection of values that defines the DB2 operating environment. Some of these values are operating system environment variables, and others are special DB2-specific system-level values known as *environment* or *registry* variables. Registry variables provide a way to centrally control the database environment. Four different registry profiles are available, and each controls the database environment at a different level. The registry profiles are as follows:

- **The DB2 global-level profile registry**—All machine-wide environment variable settings reside in this registry; one global-level profile registry exists on each DB2 workstation. To set an environment variable for all instances, use this profile registry.
- **The DB2 instance-level profile registry**—The environment variable settings for a particular instance are kept in this registry; this is where you set the majority of the DB2 environment variables. The values defined in this profile registry override any corresponding settings in the global-level profile registry.
- **The DB2 instance node-level profile registry**—This profile registry level contains variable settings that are specific to a partition (node) in a multi-partitioned database environment. The values defined in this profile registry override any corresponding settings in the global-level and instance-level profile registries.
- **The DB2 user-level profile registry**—This profile registry level contains variable settings that are specific to each user and takes higher precedence over other registry settings.

Table 2.3 shows the order in which DB2 resolves the registry settings and the environment variables when configuring the system.

Table 2.3: DB2 Registry location and precedence

Profile Registry	Precedence	Location on Windows platform	Location on Linux and UNIX platform	Authorization required on Windows platform	Authorization required on Linux and UNIX platform
Environment variables	1	Not applicable	For Bourne or korn shell: instance_home/sqllib/db2profile For C shell: instance_home/sqllib/db2cshrc	Not applicable	-rwxr-xr-x on db2profile or db2cshrc files, and part of the SYSADM group
User level	2	Lightweight Directory Access Protocol (LDAP) directory	Not applicable	Member of DB2 administrators group (DB2ADMNS)	Not applicable
Instance node level	3	\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\PROFILES\instance_name\NODES\node_number	$INSTHOME/sqllib/nodes File Name: <nodenumber>.env	Member of DB2 administrators group (DB2ADMNS)	drwxrwxr-x on the nodes directory and -rw-rw-r— on the env file, and part of the SYSADM group
Instance level	4	\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\PROFILES\instance_name	$INSTHOME/sqllib/profile.env	Member of DB2 administrators group (DB2ADMNS)	-rw-rw-r—on the file profile.env, and part of the SYSADM group
Global level	5	\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\GLOBAL_PROFILE	For root installations: /var/db2/global.reg For nonroot installations: home_directory/sqllib/global.reg	Member of DB2 administrators group (DB2ADMNS)	For root installation, root authority is required; for nonroot installation, the user who installed the product can modify the global setting

You can use the `db2greg` command to view and alter the global registry settings, which then modifies the file `/var/db2/global.reg` in root installations and `$HOME/sqllib/global.reg` in nonroot installations.

The command output in a root installation looks like this:

```
db2greg -dump
V,DB2GPRF,DB2SYSTEM,gb01qa,/opt/ibm/db2/V10.1,
S,TSA,3.2.2.5,/opt/IBM/tsamp,-,-,0,0,-,1389969042,0
S,DB2,10.1.0.3,/opt/ibm/db2/V10.1,,,3,0,,1389969093,0
I,DB2,10.1.0.3,db2inst1,/home/db2inst1/sqllib,,1,0,/opt/ibm/db2/V10.1,,
V,DB2GPRF,DB2INSTDEF,db2inst1,/opt/ibm/db2/V10.1,
```

The global registry consists of three record types:

- **Service (S)**—This records product-level information, such as version and install path.
- **Instance (I)**—This records instance-level information, such as instance name, instance path, DB2 version, and the start-at-boot flag.
- **Variables (V)**—This records variables and the value settings.

You can edit the global registry setting by using the `db2greg` command; editing in root installation needs a root privilege.

A wide variety of registry variables are available, and they vary depending on the operating system being used. Chapter 22 of the *Database Administration Concepts and Configuration Reference* manual contains a complete listing.

So how do you determine which registry variables have been set and what they have been set to? Or more important, how do you assign values to one or more registry variables? One way is by executing the `db2set` system command. The syntax for this command is:

```
db2set [variable=[value]]
        [-g|-i instance [member-number]]
        [-all]
        [-null]
        [-r [instance] [member-number]]
        [-im|-immediate]
        [-info]
        [-n DAS Node [-u user [-p password]]]
        [-l|-lr]
        [-v]
        [-ul|-ur]
        [-?|-h]
```

where:

Variable	Identifies the registry variable whose value is to be displayed, set, or removed
Value	Identifies the value to assign to the specified registry variable; if no value is provided but a registry variable is specified, the specified registry variable is deleted
Instance	Identifies the instance profile with which to associate the specified registry variable
Member-number	Identifies the node number of the instance in cases regarding the use of a DPF database
DAS Node	Identifies the name of the node where the DAS instance resides (this is deprecated in DB2 9.7 and is no longer required to be created)
User	Identifies the authentication ID to use to attach to the DAS instance
Passwords	Identifies the password (for the authentication ID) to use to attach to the DB2 Administration Server instance

Table 2.4 describes all other options with this command.

Table 2.4: The db2set command options

Option	Meaning
-g	Indicates that a global profile variable is to be displayed, set, or removed
-gl	Indicates that a global profile variable stored in LDAP is to be displayed, set, or removed; this option is effective only if the registry variable DB2_ENABLE_LDAP has been set to YES
-i	Indicates that an instance profile variable is to be displayed, set, or removed
-all	Indicates that all occurrences of the registry variable, as defined in the following, are to be displayed: • The environment (denoted by [-e]) • The node-level registry (denoted by [-n]) • The instance-level registry (denoted by [-i]) • The global-level registry (denoted by [-g])
-null	Indicates that the value of the variable at the specified registry level is to be set to NULL
-r	Indicates that the profile registry for the given instance is to be reset
-n	Indicates that a remote DAS instance node name is specified
-u	Indicates that an authentication ID that will attach to the DAS instance is specified
-p	Indicates that a password for the specified authentication ID is provided
-l	Indicates that all instance profiles will be listed
-lr	Indicates that all registry variables supported will be listed
-v	Indicates that the db2set command is to be executed in verbose mode
-ul	Accesses the user profile variables (this parameter is supported only on Windows operating systems)

Table 2.4: The db2set command options (continued)	
Option	**Meaning**
-ur	Refreshes the user profile variables (this parameter is supported only on Windows operating systems)
-h | -?	Displays help information; when this option is specified, all other options are ignored, and only the help information is displayed

It is important to note that if you execute the db2set command without options, a list containing every registry variable that has been set for the current (default) instance, along with its value, will be returned.

Thus, if you want to determine which registry variables have been set for each profile, execute the db2set command:

```
db2set -all
```

And the resulting output might look something like this:

```
[i] DB2FCMCOMM=TCPIP4
[i] DB2_SKIPINSERTED=ON
[i] DB2_OBJECT_TABLE_ENTRIES=10000
[i] DB2_USE_ALTERNATE_PAGE_CLEANING=ON
[i] DB2_LOAD_COPY_NO_OVERRIDE=nonrecoverable
[i] DB2_INLIST_TO_NLJN=YES
[i] DB2_REDUCED_OPTIMIZATION=ON
[i] DB2_EVALUNCOMMITTED=ON
[i] DB2_EXTENDED_OPTIMIZATION=Y
[i] DB2_ANTIJOIN=Y
[i] DB2TCPCONNMGRS=16
[i] DB2_SKIPDELETED=ON
[i] DB2DBDFT=SAMPLE
[i] DB2COMM=TCPIP
[i] DB2_PARALLEL_IO=*:5
[i] DB2AUTOSTART=YES
[g] DB2SYSTEM=prodbcuapp001
[g] DB2INSTDEF=db2inst1
```

Alternatively, to see the current value of the DB2COMM registry variable for all DB2 instances, execute a db2set command that looks something like this:

```
db2set -l DB2COMM
```

And finally, if you want to assign a value to the DB2COMM registry variable for all DB2 instances on a server, you can do so by executing a db2set command that looks something like this:

```
db2set -g DB2COMM=[Protocol, ...]
```

where:

Protocol — Identifies one or more communications protocols to start when the DB2 database manager for the instance is started; any combination of the following values is valid: NPIPE, TCPIP, and SSL

Thus, to set the DB2COMM instance-level registry variable such that the DB2 database manager will start the TCP/IP communication manager each time any instance is started, execute a db2set command that looks like this:

```
/home/db2inst1/sqllib/adm/db2set -g DB2COMM=TCPIP
```

You can unset the value assigned to any registry variable by providing just the variable name and the equal sign as input to the db2set command. Thus, if you want to disable the DB2COMM instance-level registry variable for an instance named db2inst1, you can do so by executing a db2set command that looks like this:

```
db2set -i DB2INST1 DB2COMM=
```

A Word About Aggregate Registry Variables

An aggregate registry variable is a group of several registry variables as a configuration that is identified by one registry variable name. Each registry variable that is part of the group has a predefined setting. The purpose of an aggregate registry variable is to ease registry configuration for broad operational objectives.

In DB2 10.1, the only valid aggregated registry variable is DB2_WORKLOAD, and the valid values for this variable are:

Value	Description
1C	1C application-specific workload setting
CM	Content Manager–specific workload setting
COGNOS_CS	Cognos® Content Server–specific workload setting
FILENET_CM	FileNet® Content Manager–specific workload setting
INFOR_ERP_LN	Infor ERP Baan–specific workload setting
MAXIMO	Maximo®-specific workload setting
MDM	Master Data Management–specific workload setting
SAP	SAP application–specific workload setting
TPM	Tivoli® Provisioning Manager–specific workload setting
WAS	WebSphere® Application Server–specific workload setting
WC	WebSphere Commerce–specific workload setting
WP	WebSphere Portal–specific workload setting

You can use an aggregate registry variable to explicitly define any registry variable that is implicitly configured, which in a way overrides the aggregated registry variable implicit value.

If you attempt to modify an explicitly set registry variable by using an aggregate registry variable, a warning is issued and the explicitly set value is kept. This warning tells you that the explicit value is maintained and will override the implicit value. For example, setting DB2_REDUCED_OPTIMIZATION to YES and then setting the DB2_WORKLOAD to SAP will generate a warning message something like the following:

```
db2set DB2_REDUCED_OPTIMIZATION=YES
db2set DB2_WORKLOAD=SAP
DBI1319W The variable "DB2_REDUCED_OPTIMIZATION" has been explicitly set and
will not be affected by the configuration of the aggregate variable "DB2_
WORKLOAD".
```

If the aggregate registry variable is used first, and then you specify an explicit registry variable, no warning is given. To identify the override settings, use the db2set -all command and check for [O] displayed next to its value, as follows:

```
[i] DB2_INLIST_TO_NLJN=YES [O]
[i] DB2_REDUCED_OPTIMIZATION=YES [O]
[i] DB2COMM=TCPIP [O]
```

And wherever the DB2_WORKLOAD setting value is active, you will see [DB2_WORKLOAD] appear next to its value something like this:

```
[i] DB2_ROWCOMPMODE_DEFAULT=STATIC [DB2_WORKLOAD]
[i] DB2_INDEX_PCTFREE_DEFAULT=0 [DB2_WORKLOAD]
[i] DB2_SKIP_VIEWRECREATE_SAP=TRUE [DB2_WORKLOAD]
```

Configuring DB2 Instances and Databases

Along with the comprehensive set of registry variables, DB2 uses an extensive array of configuration parameters to control how system resources are allocated and used on behalf of an instance and a database. The default values provided for many of these configuration parameters were produced with very simple systems in mind. The goal was for DB2 to run out of the box, on virtually any platform, not for DB2 to run optimally on the platform on which it is installed.

Thus, even though the default values for these configuration parameters are sufficient to meet most database needs, you can usually greatly improve overall system and application performance simply by changing the values of one or more configuration parameters. In fact, the values assigned to DB2 configuration parameters must always be modified or set to AUTOMATIC if your database environment contains one or more of the following:

- Large databases
- Databases that normally service large numbers of concurrent connections
- One or more special applications that have high performance requirements
- A special hardware configuration
- Unique query or transaction loads
- Unique query or transaction types

Chapter 5, "Monitoring DB2 Activity," discusses ways to measure transaction loads and performance to determine how to alter configuration parameter values. For now, let's examine the configuration parameters.

The DB2 Database Manager Instance Configuration

Whenever an instance is created, a corresponding DB2 database manager configuration file is also created and initialized as part of the instance creation process. Each DB2 database manager configuration file consists of about 96 different parameter values, and most control the amount of system resources that are allocated to a single DB2 database manager instance. Table 2.5 shows the parameters that make up a DB2 10.1 DB2 database manager configuration file.