DB2 Security

Presented by DB2 Developer Domain

http://www7b.software.ibm.com/dmdd/

Table of Contents

If you're viewing this document online, you can click any of the topics below to link directly to that section.

Section 1. Introduction

What this tutorial is about

This tutorial introduces you to DB2 security. To understand the concepts described in this tutorial, you should already have a basic knowledge of database concepts and an understanding of operating system security features.

This tutorial is the second in a series of six tutorials you can use to help prepare for the DB2 UDB V8.1 Family Fundamentals Certification (Exam 700). The material in this tutorial primarily covers the objectives in Section 2 of the test, which is entitled "Security." You can view these objectives at *http://www.ibm.com/certify/tests/obj700.shtml*.

DB2 installation is not covered in this tutorial. If you haven't already done so, we strongly recommend that you download and install a copy of *IBM DB2 Universal Database*, Enterprise Server Edition. Installing DB2 will help you understand many of the concepts that are tested on the DB2 UDB V8.1 Family Fundamentals Certification exam. The installation process is documented in the Quick Beginnings books, which can be found at the *DB2 Technical Support* Web site under the Technical Information heading.

In this tutorial, you'll learn about DB2 security features, including DB2 authentication, authorization, and privileges.

Pre-tutorial setup

The examples in this tutorial are specific to DB2 running on a Windows operating system (with native security features). However, the concepts and information provided are relevant to DB2 running on any distributed platform.

In order to work through the examples in this tutorial, the user should have:

1. Logged into a Windows machine as a user who is a member of the Administrators group. In the examples in this tutorial, we will be logged in with the user ID *LISAC*.
2. Installed DB2.
3. Created a new group on the machine on which DB2 was installed. In this tutorial, we will use the group ID *grp1*.
4. Created a second user ID on the machine on which DB2 was installed. In this tutorial, for this purpose we will use the user ID *tst1*. Note that the *tst1* user is not a member of the Administrators group.

About the author

Lisa Cawley, B.Sc., is a DB2 Certified Advanced Technical Expert and a senior member of the DB2 UDB Worldwide Support team at the IBM Toronto Software Laboratory. She currently provides technical support to DB2 customers worldwide as part of the DB2 Tools, Connectivity and Extenders front-line support team. You can contact Lisa by going to the IBM Global Directory at *http://www.ibm.com/contact/employees/us* and searching for her under the Canadian employees' directory.

Section 2. DB2 security

Aspects of database security

Database security is of utmost importance today. Your database may allow customers to purchase products over the Internet, or it may contain historical data used to predict business trends; either way, your company needs a sound database security plan.

A database security plan should define:

- Who is allowed access to the instance and/or database
- Where and how a user's password will be verified
- The authority level that a user is granted
- The commands that a user is allowed to run
- The data that a user is allowed to read and/or alter
- The database objects a user is allowed to create, alter, and/or drop

DB2 security mechanisms

There are three main mechanisms within DB2 that allow a DBA to implement a database security plan: *authentication*, *authorization*, and *privileges*.

Authentication is the first security feature you'll encounter when you attempt to access a DB2 instance or database. DB2 authentication works closely with the security features of the underlying operating system to verify user IDs and passwords. DB2 can also work with security protocols like Kerberos to authenticate users.

Authorization involves determining the operations that users and/or groups can perform, and the data objects that they may access. A user's ability to perform high-level database and instance management operations is determined by the authorities that they have been assigned. The five different authority levels within DB2 are SYSADM, SYSCTRL, SYSMAINT, DBADM, and LOAD.

Privileges are a bit more granular than authorities, and can be assigned to users and/or groups. Privileges help define the objects that a user can create or drop. They also define the commands that a user can use to access objects like tables, views, indexes, and packages.

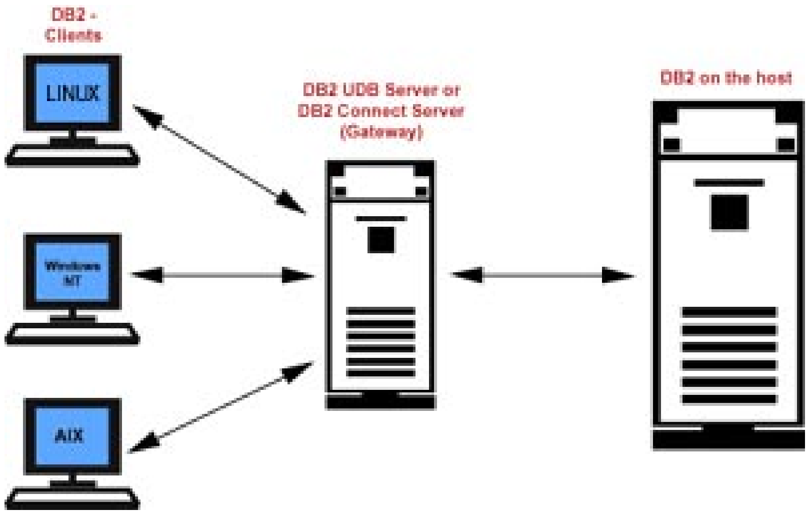
To prepare for the next section of the tutorial, you will need to create a database within the `db2inst1` instance. Make sure that the `db2instance` variable is still set to `db2inst1`, and then create the sample database using the command `db2sampl` *`drive`*, using the name of the drive where you want to create the sample. For the examples in this tutorial, we'll create the sample database on our C drive, as follows:

```
C:\PROGRA~1\SQLLIB\BIN>db2sampl c:
```

Clients, servers, gateways, and hosts

It is particularly important that you understand the terms *client, server, gateway,* and *host* when considering the security of the entire database environment. A database environment often consists of several different machines; you must safeguard the database at any potential data access point. The concepts of clients, servers, gateways, and hosts are particularly important when dealing with DB2 authentication.

The diagram below illustrates a basic client-server-host configuration.



The *database server* is the machine (or machines in a partitioned database system) on which the database physically resides. The DB2 database *clients* are machines that are configured to run queries against the database on the server. These clients can be local (that is, they can reside on the same physical machine as the database server) or they can be remote (that is, they can reside on separate machines).

If the database resides on a mainframe machine running an operating system like AS/400 (iSeries) or OS/390 (zSeries), it's called a *host* or *host server.* A *gateway* is a machine running the DB2 Connect product. Through the gateway, DB2 client machines can connect to a DB2 database that resides on a host machine. The gateway is also referred to as the DB2 Connect Server.

Section 3. DB2 authentication

When DB2 authenticates

DB2 authentication controls the following aspects of a database security plan:

- Who is allowed access to the instance and/or database
- Where and how a user's password will be verified

It does this with the help of the underlying operating system security features whenever an *attach* or *connect* command is issued. An attach command is used to connect to the DB2 instance, whereas a connect command is used to connect to a database within a DB2 instance. The examples below walk you through the different ways that DB2 will authenticate a user issuing these commands. In these examples, we've used the default authentication type of SERVER in the database manager configuration file.

Log on to the machine where DB2 is installed, with the user ID you used to create the `db2inst1` instance. Issue the following commands:

1. `db2 attach to db2inst1`
   Here, authentication is done implicitly. The user ID used to log onto the machine is used and is assumed to be already verified by the operating system.

2.
   ```
   db2 connect to sample user tst1 using mypass

   Database Connection Information

   Database server = DB2/NT 8.1.0

   SQL authorization ID = TST1

   Local database alias = SAMPLE
   ```
   Here, authentication is done explicitly. The user *tst1* with the password *mypass* is verified by the operating system. User *tst1* is successfully connected to the sample database.

3.
   ```
   db2 connect to sample user tst1 using mypass new chgpass
   confirm chgpass
   ```
   The user ID *tst1* with password *mypass* is verified by the operating system as in example 2. The password for *tst1* is then changed by the operating system from *mypass* to *chgpass*. As a result, the command in example 2 will fail if you reissue it.

DB2 authentication types

Authentication types are used by DB2 to determine where authentication is to take place. For example, in a client-server environment, will the client or the server machine verify the user's ID and password? In a client-gateway-host environment, will the client or host machine verify the ID and password?

The following table summarizes the available DB2 authentication types. In a client-gateway-host environment, these authentication options are set on the client and gateway, not on the host machine. We will discuss setting these options in more detail throughout this section. See Clients, servers, gateways, and hosts on page 5 for a refresher.

Type	**Description**
SERVER	Authentication takes place on the server.
SERVER_ENCRYPT	Authentication takes place on the server. Passwords are encrypted at the cli machine before being sent to the server.
CLIENT	Authentication takes place on the client machine (see Dealing with untrusted clients on page 8 for exceptions).
*KERBEROS	Authentication is performed by the Kerberos security software.
*KRB_SERVER_ENCRYPT	Authentication is performed by Kerberos security software if the client setting KERBEROS. Otherwise, SERVER_ENCRYPT is used.

*These settings are valid only for Windows 2000 operating systems.

Setting authentication on the server

Authentication is set on the database server within the Database Manager Configuration (DBM CFG) file using the AUTHENTICATION parameter. Remember, the DBM CFG file is an instance-level configuration file. Thus, the AUTHENTICATION parameter affects all databases within the instance. The following commands illustrate how this parameter can be altered.

- To view the authentication parameter in the configuration file:

  ```
  db2 get dbm cfg
  ```

- To alter the authentication parameter to `server_encrypt`:

  ```
  C:\PROGRA~1\SQLLIB\BIN>db2 update dbm cfg using authentication
  server_encrypt

  C:\PROGRA~1\SQLLIB\BIN>db2stop

  C:\PROGRA~1\SQLLIB\BIN>db2start
  ```

Setting authentication on the gateway

Authentication is set on the gateway using the catalog database command. For the examples on this panel, we'll use a host database named *myhostdb*.

To set the gateway authentication type to SERVER, you would issue the following command on the gateway machine:

```
db2 catalog database myhostdb at node nd1 authentication dcs
db2 terminate
```

Note that authentication is never performed on the gateway itself. In DB2 Version 8, authentication must always occur at either the client or the host database server.

Setting authentication on the client

Let's consider two scenarios on two separate client machines. We'll configure one to connect to a database on a server machine (DB2 distributed platform), and the other to connect to a database on a host machine (DB2 for OS/390, for example).

- **Client connecting to a server database:** The client authentication setting must match that of the database server to which the client is connecting (with the exception of KRB_SERVER_ENCRYPT). Client authentication is set using the catalog database command.

  Let's assume the server authentication type is set to SERVER. The following command would then be issued to catalog the server database named *sample:*

  ```
  db2 catalog database sample at node nd1 authentication server
  ```

  If the authentication type is not specified, the client will try to use SERVER_ENCRYPT by default.

- **Client connecting to a host database:** Let's assume that the authentication type on the gateway is set to SERVER. If an authentication type is not specified, SERVER authentication is assumed when accessing a database through DB2 Connect. Authentication will take place on the host database server. The following command issued from the client will cause the password to be encrypted on the client before being sent to the gateway:

  ```
  db2 catalog database myhostdb at node nd1 authentication server_encrypt
  ```

  Now let's assume authentication is set to SERVER_ENCRYPT on the gateway. Authentication will once again take place on the host database server. The password is encrypted on the client before being sent to the gateway, and encrypted on the gateway before being sent to the host machine.

Dealing with untrusted clients

If the server or gateway machine has authentication set to `CLIENT`, this implies that the client is expected to authenticate a user's ID and password. However, some client machines may not have operating systems with native security features. Such *untrusted* clients include DB2 clients running on Windows 98 and Windows ME.

There are two additional parameters in the DBM CFG file used to determine where authentication should take place when the server or gateway authentication method is set to `CLIENT` and untrusted clients are attempting to connect to the database or attach to the DB2 instance. These are the TRUST_ALLCLNTS and TRUST_CLNTAUTH parameters.

When the server or gateway authentication type is `CLIENT`, there are two other factors that come into play in addition to the TRUST_ALLCLNTS and TRUST_CLNTAUTH parameters. The first is whether a user ID and password were explicitly supplied and the second is the type of client connecting. The three DB2 clients are:

- **Untrusted clients:** As described above
- **Host clients:** Clients running on host operating systems like OS/390
- **Trusted clients:** Clients running non-host operating systems that have native security features

When authentication is set to CLIENT

The table below summarizes where authentication will take place when a connect or attach command is issued by each type of client to a server whose authentication type is set to `CLIENT`.

User ID/Password Supplied?	**TRUST_ALLCLNTS**	**TRUST_CLNTAUTH**	**Untrusted Cli**
No	Yes	`CLIENT`	`CLIENT`
No	Yes	`SERVER`	`CLIENT`
No	No	`CLIENT`	`SERVER`
No	No	`SERVER`	`SERVER`
No	`DRDAONLY`	`CLIENT`	`SERVER`
No	`DRDAONLY`	`SERVER`	`SERVER`
Yes	Yes	`CLIENT`	`CLIENT`
Yes	Yes	`SERVER`	`SERVER`
Yes	No	`CLIENT`	`SERVER`
Yes	No	`SERVER`	`SERVER`
Yes	`DRDAONLY`	`CLIENT`	`SERVER`
Yes	`DRDAONLY`	`SERVER`	`SERVER`

`DRDAONLY` refers to host clients only, despite the fact that DB2 Version 8 clients connect using DRDA as well.

The examples below illustrate setting authentication types and parameters on the server and client:

- Setting authentication on the server:

```
db2 update dbm cfg using authentication client
db2 update dbm cfg using trust_allclnts yes
db2 update dbm cfg using trust_clntauth server
db2stop
db2start
```

- Setting authentication on the client:

```
db2 catalog database sample at node nd1 authentication client
```

If the command `db2 connect to sample` is issued from any client, authentication takes place on the client. If the command `db2 connect to sample user tst1 using mypass` is issued from any client, authentication takes place on the server.

Section 4. DB2 authorities

Introduction to authorities

DB2 authorities control the following aspects of a database security plan:

- The authority level that a user is granted
- The commands that a user is allowed to run
- The data that a user is allowed to read and/or alter
- The database objects a user is allowed to create, alter, and/or drop

Authorities are made up of groups of privileges and higher-level database manager (instance-level) maintenance and utility operations. Of the five authorities available in DB2, SYSADM, SYSCTRL and SYSMAINT are *instance-level authorities.* That means that their scope includes instance-level commands as well as commands against all the databases within the instance. These authorities can only be assigned to a group; you can do so through the DBM CFG file.

The DBADM and LOAD authorities are assigned to a user or group for a particular database. This can be done explicitly using the GRANT command.

The following panels describe how each authority is assigned and what commands users with that authority are allowed to perform. Note that any reference to group membership implies that the user and group names have already been defined at the operating system level.

Users can determine what authorities and database-level privileges they have by issuing the following command:

```
db2 get authorizations
```

Obtaining SYSADM authority

SYSADM authority in DB2 is comparable to root authority on UNIX or Administrator authority on Windows. Users with SYSADM authority for a DB2 instance are able to issue any DB2 commands against that instance, any databases within the instance, and any objects within those databases. They also have the ability to access data within the databases and grant or revoke privileges and authorities. SYSADM users are the only users allowed to update the DBM CFG file.

SYSADM authority is controlled in the DBM CFG file via the SYSADM_GROUP parameter. When the instance is created, this parameter is set to Administrator on Windows (although it appears blank if you issue the command `db2 get dbm cfg`). On UNIX, it is set to the primary group of the user who created the instance.

Since SYSADM users are the only users allowed to update the DBM CFG, they are also the only ones allowed to grant any of the SYS* authorities to other groups. The following example illustrates how to grant SYSADM authority to the group *grp1*:

```
db2 update dbm cfg using SYSADM_GROUP grp1
```

Remember, this change will not take effect until the instance is stopped and then restarted. Also, keep in mind that if you are not currently logged in as a member of *grp1*, you may not have authority to restart the instance! You would have to log out and log back in with an ID in the correct group, or add your current ID to *grp1*.

Obtaining SYSCTRL authority

Users with SYSCTRL authority can perform all administrative and maintenance commands within the instance. However, unlike SYSADM users, they cannot access any data within the databases unless they are granted the privileges required to do so. Examples of commands a SYSCTRL user can perform against any database in the instance are:

- `db2start/db2stop`
- `db2 create/drop database`
- `db2 create/drop tablespace`
- `db2 backup/restore/rollforward database`
- `db2 runstats` (against any table)
- `db2 update db cfg for database` *dbname*

A user with SYSADM authority can assign SYSCTRL to a group using the following command:

```
db2 update dbm cfg using SYSCTRL_GROUP group name
```

Obtaining SYSMAINT authority

The commands that a user with SYSMAINT authority can issue are a subset of those allowed to users with SYSCTRL authority. SYSCTRL users can only perform tasks related to maintenance, such as:

- `db2start/db2stop`
- `db2 backup/restore/rollforward database`
- `db2 runstats` (against any table)
- `db2 update db cfg for database` *dbname*

Notice that users with SYSMAINT cannot create or drop databases or tablespaces.

They also cannot access any data within the databases unless they are granted the privileges required to do so.

If you have SYSADM authority, you can assign SYSMAINT authority to a group using the following command:

```
db2 update dbm cfg using SYSMAINT_GROUP group name
```

Obtaining DBADM authority

DBADM authority is a database-level authority rather than an instance-level authority. In summary, DBADM users have complete control over a database -- almost. DBADM users cannot perform such maintenance or administrative tasks as:

- `drop database`
- `drop/create tablespace`
- `backup/restore database`
- `update db cfg for database` *`db name`*

However, they can perform the following tasks:

- `db2 create/drop table`
- `db2 grant/revoke` (any privilege)
- `db2 runstats` (any table)

DBADM users are also automatically granted all privileges to the database objects and their contents. Since DBADM authority is a database-level authority, it can be assigned to both users and groups. The following commands illustrate different ways in which you can give DBADM authority.

- `db2 create database test`
  This command gives implicit DBADM authority on the database named *test* to the user who issued the command.

- `db2 connect to sample`

  `db2 grant dbadm on database to user tst1`
  This command can only be issued by SYSADM users; it issues DBADM authority to the user *tst1* on the sample database. Note that the issuing user must be connected to the sample database before granting DBADM authority.

- `db2 grant dbadm on database to group grp1`
  This command grants DBADM authority to everyone in the group *grp1*. Again, only SYSADM users can issue this command.

Obtaining LOAD authority

LOAD authority is also considered a database-level authority, and can therefore be granted to both users and groups. As the name implies, LOAD authority allows users to issue the LOAD command against a table. The LOAD command is typically used as a faster alternative to insert or import commands when populating a table with large amounts of data. Depending on the type of LOAD you wish to perform, having LOAD authority alone may not be sufficient. Specific privileges on the table may also be required.

The following commands can be run by users with LOAD authority:

- `db2 quiesce tablespaces for table`
- `db2 list tablespaces`
- `db2 runstats` (any table)
- `db2 load insert` (must have insert privilege on table)
- `db2 load restart/terminate after load insert` (must have insert privilege on table)
- `db2 load replace` (must have insert and delete privilege on table)
- `db2 load restart/terminate after load replace` (must have insert and delete privilege on table)

Only users with either SYSADM or DBADM authority are permitted to grant or revoke LOAD authority to users or groups. The following examples illustrate how LOAD authority can allow our user to `LOAD` data into a table called *sales.* Assume that the command `db2 connect to sample` has already been issued.

- `db2 grant load on database to user tst1`

  `db2 grant insert on table sales to user tst1`
  With LOAD authority and insert privilege, *tst1* could issue a `LOAD INSERT` or a `LOAD RESTART`, or `TERMINATE` after a `LOAD INSERT` against the sales table.

- `db2 grant load on database to group grp1`

  `db2 grant delete on table sales to group grp1`

  `db2 grant insert on table sales to group grp1`
  With LOAD authority, as well as delete and insert privileges, any member of *grp1* could issue a `LOAD REPLACE` or a `LOAD RESTART`, or `TERMINATE` after a `LOAD REPLACE` against the sales table.

Section 5. DB2 privileges

Database and object privileges

In the preceding section, we briefly touched on the concept of privileges. Privileges can generally be placed into two main categories: database-level privileges, which span all objects within the database, and object-level privileges, which are associated with a specific object.

The database-level privileges that a user might be given are:

- CREATETAB: Users can create tables within the database.
- BINDADD: Users can create packages in the database using the BIND command.
- CONNECT: Users can connect to the database.
- CREATE_NOT_FENCED: Users can create unfenced user-defined functions (UDFs).
- IMPLICIT_SCHEMA: Users can implicitly create schemas within the database without using the CREATE SCHEMA command.
- LOAD: Users can load data into a table
- QUIESCE_CONNECT: Users can access a database while it is in a quiesced state.
- CREATE_EXTERNAL_ROUTINE: Users can create a procedure for use by applications and other users of the database.

Database *objects* include tables, views, indexes, schemas, and packages. Fortunately, most of the object-level privileges are self explanatory. The following table summarizes these privileges.

Privilege name	**Relevant object(s)**	**Description**
CONTROL	Table, View, Index, Package, Alias, Distinct Type, User Defined function, Sequence	Provides full authority on the object. Use grant or revoke privileges on the object t
DELETE	Table, View	Allows users to delete records from the c
INSERT	Table, View	Allows users to insert records into the ob IMPORT commands.
SELECT	Table, View	Provides the ability to view the contents statement.
UPDATE	Table, View	Allows users to modify records within the statement.
ALTER	Table	Allows users to alter the object definition
INDEX	Table	Allows users to create indexes on the ob statement.
REFERENCES	Table	Provides the ability to create or drop fore object.

BIND	Package	Allows users to rebind existing packages
EXECUTE	Package, Procedure, Function, Method	Allows users to execute packages and r
ALTERIN	Schema	Allows users to modify definitions of obje
CREATEIN	Schema	Allows users to create objects within the
DROPIN	Schema	Allows users to drop objects within the s

Information on object-level privileges is stored in the system catalog views. The view names are `syscat.tabauth`, `syscat.colauth`, `syscat.indexauth`, `syscat.schemaauth`, `syscat.routineauth`, and `syscat.packageauth`.

Explicit privileges

Privileges can be *explicitly* granted and revoked to users or groups using the GRANT and REVOKE commands. Let's take a look at how you could use these commands on various objects.

While logged in as a user with Administrator authority on Windows, bring up two DB2 command windows. Make sure that the `db2instance` variable is set to `db2inst1` in both windows!

From Window 1, issue the following command:

```
db2 connect to sample
```

Now, from Window 2, issue this command:

```
db2 connect to sample user tst1 using passwd
```

Remember, the commands in Window 1 are being issued by a user with SYSADM authority. The commands in Window 2 are being issued by *tst1*, a user with no specific authority or privileges on the sample database. Note that the schema name associated with the tables in your sample database will be the name of the user that issued the `db2sampl` command. In these examples, that user is *LISAC*.

Now, from Window 2, issue the following command:

```
db2 select * from lisac.org
```

You should see this response:

```
SQL0551N  "TST1" does not have the privilege to perform operation "SELECT" on object "L
```

To correct the situation, issue the following command from Window 1:

```
db2 grant select on table lisac.org to user tst1
```

Now our earlier command will succeed! Next, let's issue a more ambitious command from Window 2:

```
db2 insert into lisac.org values (100, 'Tutorial', 1, 'Eastern', 'Toronto')
```

Again, you'll see an error message:

```
SQL0551N  "TST1" does not have the privilege to perform operation  "INSERT" on object "
```

So, enter the following command from Window 1:

```
db2 grant insert on table lisac.org to group grp1, user mytest
```

Our earlier failed INSERT command should now complete successfully, because *tst1* is a member of group *grp1*.

Now, let's enter the following command in Window 2:

```
db2 drop table lisac.emp_photo
```

Again, you'll see an error message:

```
SQL0551N  "TST1" does not have the privilege to perform operation "DROP TABLE" on objec
```

So, we'll have the grant that privilege. Enter the following from Window 1:

```
db2 grant dropin on schema lisac to all
```

Our `DROP TABLE` command should now complete successfully.

Now that we've finished our example, let's revoke all the privileges we just granted. Do so by issuing the following commands from Window 1:

```
db2 revoke select on table lisac.org from user tst1
db2 revoke insert on table lisac.org from group grp1
db2 revoke dropin on schema lisac from all
```

Note that revoking privileges from a group does not necessarily revoke it from all members of that group. For example, the following command could have been used to revoke all privileges (except CONTROL) from *grp1* on the table lisac.org:

```
db2 revoke all on table lisac.org from group grp1
```

However, the user *tst1* (who is a member of *grp1*) would have kept the select privileges on that table, since he or she had been granted that privilege directly.

Implicit privileges

DB2 may grant privileges automatically when certain commands are issued, without the need for an explicit GRANT statement to be issued, as you saw in the previous panel. The table below summarizes some commands that result in privileges being implicitly granted by the database manager. Note that these privileges are implicitly revoked when the object created is dropped. They are not, however, revoked when higher-level privileges are explicitly revoked.

Command issued	**Privilege granted**	**To whom it is**
`CREATE TABLE mytable`	CONTROL on *mytable*	User issuing c
`CREATE SCHEMA myschema`	CREATEIN, ALTERIN, DROPIN on *myschema*, plus the ability to grant these to others	User issuing c
`CREATE VIEW myview`	CONTROL on *myview* only if CONTROL is held on all tables and views referenced in the definition of myview	User issuing c
`CREATE DATABASE mydb`	SELECT on *mydb*'s system catalog tables, IMPLICIT_SCHEMA on mydb *	PUBLIC**

*When a user creates a database, that user is implicitly granted DBADM authority on that database. With DBADM authority comes implicit CONNECT, CREATETAB, BINDADD, IMPLICIT_SCHEMA, and CREATE_NOT_FENCED privileges. These privileges will remain with the user even if the DBADM authority is revoked.

**PUBLIC is a special DB2 group that includes all users of a particular database. Unlike the other groups we've discussed thus far, PUBLIC does not have to be defined at the operating system level. There are some privileges granted to PUBLIC by default. For example, this group receives CONNECT privilege on the database and SELECT privilege on the catalog tables automatically. GRANT and REVOKE commands can be issued against the PUBLIC group, like so:

```
db2 grant select on table sysibm.systables to public
db2 revoke select on table sysibm.systables from public
```

Indirect privileges

Privileges can be obtained indirectly when *packages* are executed by the database manager. A package contains one or more SQL statements that have been converted into a format that DB2 uses internally to execute them. In other words, a package contains multiple SQL statements in an executable format. If all the statements in the package are static, a user would only require EXECUTE privilege on the package to successfully execute the statements in the package.

For example, assume *db2package1* executes the following static SQL statements:

```
db2 select * from org
db2 insert into test values (1, 2, 3)
```

In this case, a user with EXECUTE privilege on *db2package1* would indirectly be granted SELECT privilege on the table org and INSERT privilege on the table test.

Section 6. Summary, resources, and feedback

Summary

Now that you've completed this tutorial, you should have a fundamental understanding of the following topics:

Elements of a DB2 security plan: You should understand the structure of the entire DB2 environment, which includes client, servers, gateways, and hosts. You should also understand authentication, authorization, and privileges.

DB2 authentication types: You should know how to set authentication types using the `db2 update dbm cfg using authentication` *`type`* command on the server, and using the `db2 catalog database` command on the gateway and client.

DB2 authorities: You should understand the basics of the SYSADM, SYSCTRL, and SYSMAINT authorities, which are set in the DBM CFG file, and DBADM and LOAD authorities, which are set via the GRANT command and revoked using the REVOKE command. You should definitely know what command each authority is allowed to run.

DB2 privileges: You should have an understanding of the different types of privileges and what they allow a user to do. Examples are CONTROL, INSERT, DELETE, CREATEIN, DROPIN, REFERENCES, and SELECT. You should also know how a privilege is obtained/revoked explicitly (GRANT/REVOKE commands), implicitly, or (for packages only) indirectly.

Resources

You can learn more about DB2 instances from the following resources:

- *DB2 Version 8 Administration Guide: Implementation, Chapter 4. Controlling Database Access*
- *DB2 Version 8 Connect User's Guide, Chapter 14. Security*

For more information on the DB2 Fundamentals Exam 700:

- *IBM Data Management Skills information*
- Download a *self-study course for experienced Database Administrators (DBAs)* to quickly and easily gain skills in DB2 UDB.
- Download a *self study course for experienced relational database programmers* who'd like to know more about DB2.
- *General Certification information* -- including some book suggestions, exam objectives, and courses.

Check out the other parts of the DB2 V8.1 Family Fundamentals Certification Prep tutorials:

- *DB2 V8.1 Family Fundamentals Certification Prep, Part 1 of 6: DB2 Planning*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 3 of 6: Accessing DB2 UDB Data*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 4 of 6: Working with DB2 UDB Data*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 5 of 6: Working with DB2 UDB Objects*
- *DB2 V8.1 Family Fundamentals Certification Prep, Part 6 of 6: Data Concurrency*

Feedback

Colophon

This tutorial was written entirely in XML, using the developerWorks Toot-O-Matic tutorial generator. The open source Toot-O-Matic tool is an XSLT style sheet and several XSLT extension functions that convert an XML file into a number of HTML pages, a zip file, JPEG heading graphics, and two PDF files. Our ability to generate multiple text and binary formats from a single source file illustrates the power and flexibility of XML. (It also saves our production team a great deal of time and effort.)

You can get the source code for the Toot-O-Matic at www6.software.ibm.com/dl/devworks/dw-tootomatic-p. The tutorial Building tutorials with the Toot-O-Matic demonstrates how to use the Toot-O-Matic to create your own tutorials. developerWorks also hosts a forum devoted to the Toot-O-Matic; it's available at www-105.ibm.com/developerworks/xml_df.nsf/AllViewTemplate?OpenForm&RestrictToCategory=11. We'd love to know what you think about the tool.