Roles: A New Security Feature in INFORMIX-OnLine Dynamic Server, Version 7.10.UD1

by Lester Knutsen

Introduction

INFORMIX-OnLine Dynamic Server, version 7.10.UD1, introduced a new feature called "Roles." Roles provide a way to grant and revoke privileges to a function, rather than to individual users. In addition, a user is granted the privilege to use one or more Roles. When a user requires access to the privileges of a Role, the user or application sets the current access levels to the Role. Then, after the user has performed the functions for which the Role was granted, the Role can be deselected so that the privileges are no longer in effect. This article examines some examples which use Roles to improve security, and discusses the limitations of Roles. The examples used in this article were developed using the **stores** database provided with INFORMIX-OnLine Dynamic Server, version 7.10.UD1, on a Sun Sparc hardware platform.

This article begins with a simple example which illustrates the use of Roles. This example restricts insert, update, and delete access to a group of users in the Orders Department. First, all privileges are revoked from everyone in the table. Then, instead of granting the select, insert, update and delete privileges to each individual, three Roles are created. One Role, `read_ord`, provides select-only access. The next Role, `upd_ord`, provides select, update and insert access, and the final Role, `del_ord`, includes delete privileges. Then, individuals are granted the privilege to use these Roles. Finally, this article discusses how to set up the applications to use these Roles.

Notes

The examples contained in this article make use of the **stores** database and a table contained within this database called **orders**.

Creating Roles

Creating a Role begins with the `CREATE ROLE role_name` statement, where `role_name` is an eight-character name for the Role. The `role_name` cannot be the name of a user on the system, since it is stored in the system table **sysusers**. In order to create a Role, it is necessary to have **dba** privileges in the database. The following statements are used to create the three Roles:

```
create role read_ord;
create role upd_ord;
create role del_ord;
```

After creating the Roles, the following query can be performed on the system table **sysusers** to view the Roles:

```
select * from sysusers where usertype = "G";
```

The query returns the following data:

```
username    usertype            priority password
read_ord    G             5
upd_ord G                 5
del_ord G                 5
```

In the **sysusers** table, a usertype **G**, and a new usertype in INFORMIX-OnLine Dynamic Server, version 7.10.UD1, indicates a Role definition.

Privileges for a Role

Granting privileges to a Role is the same as granting privileges to a user, and uses the same syntax. For the purposes of the example, it is first necessary to revoke all privileges on the **orders** table. The following SQL statement displays all privileges which have been granted on the **orders** table:

```
select * from systabauth where tabid in
 (select tabid from systables where tabname = "orders");
```

To revoke privileges from public, use the following SQL statement:

```
revoke all on orders from public;
```

This command must be repeated for each user with privileges to the **orders** table.

Next, use SQL to grant privileges to each Role:

```
grant select on orders to read_ord;
grant select, insert, update on orders to upd_ord;
grant select, delete on orders to del_ord;
```

After granting privileges, run the query against the system tables to view the results:

```
select * from systabauth where tabid in
 (select tabid from systables where tabname = "orders");
```

Following are the results:

```
      grantor   grantee    tabid   tabauth
      lester    del_ord    101     s---d---
      lester    read_ord   101     s-------
      lester    upd_ord    101     su-i----
```

The results show that the user **lester** granted the privileges, the **grantee** column displays the Role name, and the **tabauth** column contains the privileges.

Adding Users to a Role

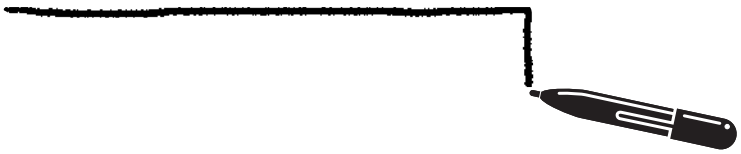
It is now necessary to add the users to the appropriate Roles. In the example, there are five users in the orders department: abby, joe, ron, jack, and linda. Of this group, everyone must have the ability to read orders, while only linda and abby must have the capability to add and update orders. In addition, abby must be able to delete orders. To accomplish this, use the following SQL statements.

```
grant read_ord to abby, joe, ron, jack, linda;
grant upd_ord  to abby, linda;
grant del_ord  to abby ;
```

Version 7.1 of INFORMIX-OnLine Dynamic Server provides a new system table called **sysroleauth**, which stores information about users' access to Roles. A select on the table returns the following information:

```
rolename        grantee     is_grantable
read_ord        abby        n
read_ord        joe         n
read_ord        ron         n
read_ord        jack        n
read_ord        linda       n
upd_ord         abby        n
upd_ord         linda       n
del_ord         abby        n
```

The above information displays the Role name, the users who have access to that Role, and an "N" (No, cannot grant this Role to someone else) or a "Y" (Yes, can grant this Role to someone else).



Using a Role: the SET ROLE Statement

Once a user is granted the privilege to use a Role, he or she does not yet have automatic access to the privileges of the Role. The user, or the application executed by the user, must first execute the SET ROLE statement. Note that any user with SQL knowledge and connect privilege to the database can use the SET ROLE command to activate a role.

If the user Joe attempts to select data from the **orders** table before the SET ROLE statement is executed, he receives the following error message:

```
select * from orders;
#             ^
#  272: No SELECT permission.
```

However, when Joe—or the application he is using—sets the current Role to the proper privileges, he can read the data. The following SQL command sets the Role and selects all data from the **orders** table:

```
set role read_ord ;
select * from orders;
```

When a user no longer requires a Role, the Role can be set to NONE or NULL, which removes the privileges of the Role. In an application, use the SET ROLE NONE or NULL statement to end the Role's privileges when those privileges are no longer required. Note the following syntax:

```
set role none;
select * from orders;
#                   ^
#  272: No SELECT permission.
```

Roles in Applications

Roles are designed for use in applications. The application can set a Role, perform the tasks, and then deselect the Role. In this way, a user only retains privileges while the application runs. Once the application is complete, the user's privileges are revoked.

To use a Role in an application, it is necessary to prepare and execute a statement which sets the Role for the application. The following statements are examples, in the context of an INFORMIX-4GL program, that set the Role to `read_ord`:

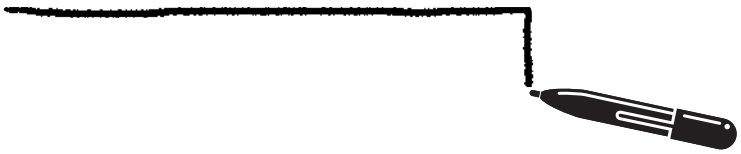
```
prepare role_stmt from "set role read_ord"
execute role_stmt
if ( sqlca.sqlcode != 0 ) then
       error "Cannot use this role"
fi
```

After executing the statement, ensure that it was successful. Otherwise, the user will attempt to perform functions without the proper privileges; this will generate numerous other SQL errors.

There are several new error messages in INFORMIX-OnLine Dynamic Server, version 7.10.UD1, which handle Roles. For example, if a user does not have permission to use a Role, the sqlca.sqlcode is `19805: No privilege to set to the Role.`

Conclusion

Especially when many users are involved, Roles provide useful security features. Roles enable the Database Administrator (DBA) to effectively control database privileges. The only drawback, which is often true with any new features, is that the DBA must change existing applications to take advantage of Roles.



About the Author

Lester Knutsen is a database consultant who has used Informix products since 1983. He is the president of Advanced DataTools Corporation, a company which provides DBA training, support and consulting services to major corporations. Lester is also president of the Washington D.C. area Informix User Group. Over the last six years, he has guided its growth into one of the largest and most active Informix user groups in the U.S. He is also one of the founding members of the International Informix User Group. Lester can be contacted via e-mail at lester@access.digex.net.

Advanced DataTools Corporation

Advanced DataTools Corporation—an Informix Solutions Alliance Partner—provides consulting in database design, application development, web database access, decision support systems/data warehousing, performance enhancement, and project management using Informix products. The company developed DB Privileges, a tool for managing Informix database security.

Advanced DataTools Corporation (ADTC) is located at 4216 Evergreen Lane, Suite 136, Annandale, VA 22003. For more information, contact ADTC at 703 256 0267 or 800 807 6732, or access the ADTC web site at www.access.digex.net/~lester. TN