Banner General
Security Administration Handbook

Release 8.2
June 2009

SUNGARD HIGHER EDUCATION

Trademark, Publishing Statement and Copyright Notice

Prepared by: SunGard Higher Education

4 Country View Road
Malvern, Pennsylvania 19355
United States of America
(800) 522 - 4827

Customer Support Center Website

http://connect.sungardhe.com

Documentation Feedback

http://education.sungardhe.com/survey/documentation.html

Distribution Services E-mail Address

distserv@sungardhe.com

Revision History Log

Publication Date	Summary
June 2009	New version that supports Banner General 8.2 software.

Banner General 8.2
Security Administration Handbook

Contents

Chapter 1 Banner Security

Chapter 4 Security Auditing

Chapter 5 Security with Shared Connections

Chapter 6 Multi-Entity Processing (MEP)

1 Banner Security

Security means control over the use of a system, and specifically the ability to set up a system so that access to data is available only to properly authorized personnel. The simplest model of security involves requiring each user to log on with an ID and password. However, because of Banner's distributed architecture, the ID/password model is not enough to ensure security. Security must be enforced not just at the application level, but also at the database level.

Securing the Oracle Database

Banner's Oracle database can be accessed by any user on your network who has a valid Oracle user ID and password. Oracle does not tie the user ID and password to a specific means of access. Thus, anyone with a user ID to run Banner could use this ID to access the database from some other application (such as Banner Self-Service).

Many third-party tools that directly access Oracle databases can be easily installed on desktop computers. If users use these third-part tools in conjunction with their Banner IDs and passwords, they would be able to bypass all application-enforced validation and data integrity.

Users could also potentially bypass the Banner application by modifying or developing their own software. If a user has been given permission to run a Banner form, that permission should not extend to the user's own modified version of that same form.

The Banner security system was designed to meet the following two goals:

- Prevent privileges given to the user to run Banner from being used in external tools such as Excel or SQL*Plus.
- Provide object authentication to prevent obsolete or user-developed objects from accessing the database.

Preventing the user from using his Banner-required Oracle privileges in a non-Banner application has been solved using roles with passwords. The role has the privileges that the application requires, the end user does not. The role is password protected with a randomly-generated password that the user does not know. The application, working with some database procedures, can decipher the password and activate the role. This technique requires the end user to have a connect privilege only. The privileges required to run Banner will be activated as required.

The role activation technique also ties in with the second goal of the security system, object authentication. Object authentication guarantees that the object being executed by the end user is the authorized version of the Banner object and not a user-modified, counterfeit version.

Object authentication is achieved by the use of encryption keys or *seed* numbers. There are three of these numbers, and they are known only to the database and valid Banner objects. These encryption keys are used to decipher the password that is required to activate the role. These numbers are compiled into valid Banner objects. After a Banner upgrade or install is performed, the seed numbers are hidden to prevent the creation of counterfeit forms. Without these seed numbers, any forms created will not know how to activate the role required to access the database.

Banner Security in Action, Step by Step

When an object begins execution it must activate the role to obtain the correct permissions. This activation process is described below. The same basic procedure applies for forms, COBOL programs, and C programs.

1. The stored procedure `G$_VERIFY_PASSWORD1_PRD` is called, providing the object name and version as parameters.

2. The stored procedure checks to see what role, if any, the current user has with that object. Permissions for the object are checked in the following order:

   2.1. Direct authorizations to the user are checked first and take precedence over other authorizations.

   2.2. If the user did not have any direct permissions for the object, then classes and security groups that the user is enrolled in are checked.

   2.3. When a user has access to an form object through two or more security classes, only one class's permissions can be chosen and applied. The classes are given priority through an alphabetical rule.

   - The class names with maintenance roles for the object (roles ending with *_M*) are sorted alphabetically, and the first in the list is used.
   - If there are no classes with *_M* roles, all other classes are sorted alphabetically, and the first in the list is used.

Note

For local changes to this precedence order, please see the `g$_get_role_for_object_fnc` and `g$_get_tab_security_fnc` functions in gspsecr.sql. ■

3. At this point, one of several possibilities will happen based on the security level being enforced and the result of the object authorization lookup.

   3.1. The word *INSECURED* is returned if security is turned off. This will cause the object to run with the database privileges that have been given directly to the user.

Note

Running Banner without *ROLE* level security is not supported as of Release 8.0. Banner 8.0 only supports *ROLE* level security. ■

   3.2. The word *INSECURED* is returned if process level security is turned on and the user is authorized to use the object. This will cause the object to run with the database privileges that have been given directly to the user.

Note

Process level security is not supported as of Release 8.0. Banner 8.0 only supports ROLE level security. ■

   3.3. An encrypted password is returned if you are allowed to use the routine and role level security is active.

   3.4. An error condition is raised terminating execution of the object.

4. If the object being activated sees the word *INSECURED*, the security section exits and object execution will continue. The object execution will succeed if the user has sufficient privileges to the required database objects. If a password was returned, the local routine then does a partial decryption of the returned password.

5. The same stored procedure is called again passing it the partially decrypted password.

6. The stored procedure checks your work so far and will log a security violation if the value is not correct. If the application calculated the correct value, the next level of decryption is performed using seed number 2. The result is returned to the calling object.

7. The calling object completes the decryption using seed number 1. The result of this decryption is the actual password. The calling object activates the role and clears the password from memory.

The main purpose of the multi-level decryption is to support the object authentication process and to provide a point where attempted security violations can be logged.

Object Authentication

One of the security system's main responsibilities is to guarantee that the user is executing the proper form assigned to them, rather than a user-modified, counterfeit version. This is as important as controlling what object names the user has access to. Without object authentication, you are controlling the name of what the user executes without knowing what logic the object contains.

Object authentication is accomplished using three encryption keys or seed numbers. The password that is required to activate each Banner role is encrypted three times using each of the seed numbers as the encryption key. If the object does not know the numbers needed to reverse this process, it will not be able to activate the role required to access the required database objects.

`CREATE SESSION` is the only privilege that the user needs to start Banner. When the user that has only `CREATE SESSION` privilege connects to the database they can only access objects that are granted to public. The security routines are granted to public and that is all the user needs to get the security process rolling.

Tip

Banner delivers the `BAN_DEFAULT_CONNECT` role, which can typically be used as the user's default role.

Every form must essentially start from this *no privilege* state to enforce object authentication. Each form must deactivate its own role when it terminates or needs to call another form for any reason. If this is not done, the calling form can be replaced with the user-developed version that would be able to inherit the privileges of the calling form.

Banner Roles

A Banner Role is a definition that includes multiple Oracle privileges and grants. The definition of a role enables the security administrator to group privileges and grants into a common object that can then been associated with Banner objects. This eliminates the need to define and maintain the low level of access needed for each individual object. SunGard delivers multiple roles, such as

- `BAN_DEFAULT_M`: when associated with a object and a user, enables maintenance access
- `BAN_DEFAULT_Q`: enables query-only access.
- `BAN_DEFAULT_NO_ACCESS`: a role with no privileges that can be used to override object assignments from security classes or groups. It can also be assigned to terminated accounts.

Default Roles in Oracle Database 11g

If you are upgrading to Oracle Database 11g, you must plan for a change in the way that Oracle manages default roles. With Database 11g, roles that are password encrypted, such as the `BAN_DEFAULT_CONNECT` role delivered with Banner, can no longer be assigned as a default role. A user logging in with a password encrypted role in 11g will be stopped with error ORA-1045, *User lacks CREATE SESSION privilege; logon denied*.

At the time of Release 8.2, this remained an open issue associated with Defect 1-5DG7XF. One workaround is to remove the password on the BAN_DEFAULT_CONNECT role by issuing this command:

```
alter role ban_default_connect not identified;
```

For more advice on this workaround and other information on this issue, refer to FAQ 1-5BWTYS and FAQ 1-4W1JEA.

Security Classes

The security system supports the definition of user classes. A security class is functionally equivalent to an Oracle role but it controls access to Banner objects, not tables. This allows the permissions for the user type to be defined once for the class and then assigned to many users. This eliminates the need to define the security for each end user.

SunGard delivers class definitions for each Banner product. These classes will be maintained by the upgrade process and will allow access to every object (Form, C and COBOL program) delivered with the product. You may assign users to these classes but it is recommended that you do not modify them. If you need to add locally-created or modified objects, create a class containing just those objects and add the user to the new class.

Users may be enrolled in multiple classes. A Banner class should be built at the lowest level that you will be assigning to an end user. Users enrolled in multiple classes can have greater access, and be more powerful and responsible. This method will simplify adding new Banner objects to site-defined classes during future Banner upgrades.

A “Standard” Banner Class

The following list of forms are those that are typically viewed a minimally required for a Banner user. You can use this list of forms as a starting point for creating a standard Banner security class with the common, minimum permissions necessary at your institution.

Based upon implementation decisions at your institution, there may be other forms that you consider essential, and some of forms listed below may not be necessary. For example, if password control is not left up to the end user, it may be decided that no one needs access to GUAPSWD.

All the forms listed below require a default role of BAN_DEFAULT_M.

Minimum forms

- GUAABOT - Access to the *Help About* form
- GUACALC - Enables use of calculator
- GUACALN - Allows access to the calendar associated with entering dates
- GUAERRM - Enables display of multiple error messages
- GUAGMNU - Allows access to the menu
- GUAHELP - Allows access to form help
- GUAINIT - Required to be able to log into Banner
- GUIOBJS - Enables searching of menu items

Personalization Forms

- GUAMESG - Enables sending and receiving of messages from Banner users
- GUAPMNU - Enables maintenance of personal menus
- GUAPSWD - Enables changing of password
- GUAQFLW - Enables the starting of a quickflow as defined on GUAQUIK
- GUAUPRF - Enables setting of personal preferences
- GUQQUIK - Enables searching of quickflow processes

Submitting Jobs

Users who submit jobs need:

- GJRRPTS - Example Process being run via GJAPCTL
- GJAPCTL - Form needed to submit Job
- GJIREVO - Form needed to view results in Database
- GUQINTF - Form needed to submit Job
- GURINSO - Process needed to insert results into Database

Population Selection

For population selection (GLRSLCT) a user needs:

- GJAPCTL - To submit GLBDATA
- GJARSLT - To view error or results from POPSEL compile
- GLBDATA - To execute POPSEL

- GLOLETT - To compile POPSEL
- GLRAPPL - To define Application
- GLRSLCT - To define POPSEL
- GUQINTF - To submit POPSEL compile/execute

Variables

To do variable (GLRVRBL) processing a user needs:

- GJARSLT - To view error or results from POPSEL compile
- GLOLETT - To compile Variable
- GLRAPPL - To define Application
- GLRVRBL - To define Variable
- GUQINTF - To submit Variable compile

Security Groups

Release 8.0 delivered security groups, a new concept for grouping of users with similar security needs. A security group is a collection of individual objects as well as classes. When you associate a user with a security group, the user is assigned to the classes and objects of that group.

The security groups feature enables a security administrator to define common groups (correlated with, for example, specific job titles or responsibilities at the institution) and assign users to those groups. If the need of the group changes, the security administrator only needs to maintain the security group instead of having to maintain all the users individually.

Users may be assigned to multiple security groups. Users may also be assigned to additional classes or have individual object overrides.

Site Responsibilities

A successful Banner security setup provides each user with access to the Banner forms, tables, and other objects necessary for that user's activities, while preventing users from gaining access to data and objects that they should not be able to access. Each user's privileges can be a combination of direct user grants, class grants (for classes and groups the user belongs to), and role privileges.

Note

When planning a security setup, remember that a user needs access not only to application forms, but also to common utilities such as calendar, calculator, and comments forms.

Security Checklist

For a secure Banner setup, you should adopt all of the following practices:

1. Require that users have individual accounts.
2. Document site security procedures and end-user responsibilities.
3. Secure all Oracle accounts that have an *any* privilege.
4. Hide the routines that contain the seed numbers.
5. Monitor the security log tables.
6. Ensure that every Oracle ID has a default role.

Note

If a user does not have a default role, then *all* the roles that are assigned to the user are active when the user logs on to Oracle, whether that logon is through Banner or through a third party tool such as SQL*Plus. As a general rule, the default role should have minimal privileges and grants. In most cases BAN_DEFAULT_CONNECT will suffice for most users. ■

7. Assign different seed numbers to every instance. This will require compiling and generating programs and objects separately for each environment. This involves extra effort, but it provides a valuable measure of safety; for example, it guarantees that a test program will not inadvertently be run against the production database.

Note

Even if two instances have the same seed numbers, the user cannot run a form using the wrong instance if they do not have a valid Oracle account and password. ■

8. Secure Banner upgrade directories. The Banner upgrade scripts contain passwords and other information that could be used to compromise your database security. It must be protected from unnecessary user access and should be backed up and removed from your machine as soon as the upgrade is completed.

Managing Oracle Privileges

No security system is perfect, but the Banner security system provides a reasonable level of security that cannot be bypassed by accident.

A potential opening exists whenever users have extra, unnecessary Oracle privileges. For instance, any account that has `SELECT ANY` or `UPDATE ANY` privilege could insert rows directly into the security tables owned by BANSECR and bypass the intended security. Any account that has `ALTER USER` privilege could set his default role to one of the Banner-

secured roles. The user would not need to know the secured role's password to gain access.

Any user that can create a role could grant the Banner roles to the new role. Roles granted in this way do not require the use of the password. The following list of Oracle privileges must be closely monitored:

- SELECT ANY
- INSERT ANY
- UPDATE ANY
- DELETE ANY
- GRANT ANY ROLE
- CREATE ROLE
- CREATE USER
- ALTER USER

Any user that does not have a default role poses a security problem. The Banner roles are password protected but they are granted to the end users. This means that when the user does not have a default role, he gets all roles granted to him as the default role and does not need to know the password.

Super Roles

A performance problem was found while developing the security system. As an end user is granted more and more roles, the *set role* command takes longer and longer to get the roles set properly. Until this problem is fixed, SunGard will be delivering only Super Roles. This role will have access to all Banner database objects either directly or via a SELECT ANY privilege. This role will still be password protected and is inaccessible from third party tools.

Security External to Banner

One of the design goals of the Banner security system was to prevent Banner privileges from being misused via external tools. The seed numbers and object authentication satisfy these design goals. However, this technique can only be employed in languages or tools that can include the seed numbers and generate a static executable file.

The seed numbers exist in the GOQOLIB library and are referenced into the forms when they are compiled. Banner security enforces object authentication because the seed numbers are included in the generated FMX and not the library.

If you attempt to secure object types other than forms, C, and COBOL, make sure the seed numbers are compiled into the code and not simply referenced from somewhere else. If

you do, you are effectively disabling object authentication and creating a security compromise as severe as if the security had been disabled entirely.

The intended method for securing table and view access external to Banner is to control it via the end user's default role. If a user is allowed to run SQL*PLUS scripts (or is allowed to do ad hoc reporting), the user should have a default role that provides access to the required tables.

The Banner security maintenance system controls what the user can run within Banner. The Oracle default role controls what the user can access using the wide array of reporting tools that can access an Oracle database.

Security with Multiple Database Instances

If different seed numbers are set for each database instance, the executables from one database (for example, a test instance) cannot be used to connect to another database (such as a production instance). With this kind of setup, every Banner object must be separately compiled for each database environment.

Role Maintenance and Multiple BANINST Accounts

Role maintenance functionality is done primarily with Oracle SYS level privileges that have been granted to the BANSECR account (and optionally BANSECR_*xxx* distributed security accounts).

When dealing with table and view grants, there is not a SYS level privilege that allows you to issue a grant for someone else's object. You can require that the BANSECR account be issued a `GRANT WITH GRANT OPTION` for every Banner object, but that will require a great deal of maintenance because of the many grants that are involved.

You could also use a database package called `G$_FOREIGN_SQL_PKG` that is owned by the BANINST1 account. This package can only be used by a BANSECR or BANSECR_*xxx* account. Attempted use from any other ID will result in an error. This package will execute a SQL command passed to it as a parameter. The GSASECR form takes advantage of this because a stored procedure executes with the Oracle privileges of the owner of the package or procedure. BANINST1 owns this procedure so the BANINST1 account is effectively issuing the grant. The BANINST1 account already has `GRANT WITH GRANT OPTION` on all the Banner objects so no additional object grants are necessary.

The GSASECR form supports sites with multiple BANINST accounts. Before the SQL is passed to the database procedure, the GSASECR form determines which BANINST account has the `GRANT WITH GRANT OPTION` privilege for the object and executes the correct version of the procedure.

Every BANINST account must own a copy of the G$_FOREIGN_SQL_PKG package. A public synonym called BANINST?_SQL_PKG must also exist for each version. (The question mark must be replaced with the number of the BANINST account.)

If you are using distributed security maintenance, make sure execute permission for each G$_FOREIGN_SQL_PKG is granted to the BANSECR_*xxx* user's default role. If the grant is directed to the user you may experience some permission errors in the package.

Modifying Local Forms

If you have locally developed or modified forms, you can include them in Banner security by following these step.

1. Open each form and add an appropriate audit trail.
2. Attach the GOQRPLS library program from the database.
   - 2.1. Click **Attached Libraries.**
   - 2.2. Click the green plus sign (+) to add a new library.
   - 2.3. Browse to locate *GOQRPLS.PLL*.
   - 2.4. Click **Attach**.
3. Reference the G$_REVOKE_ACCESS and the G$_VERIFY_ACCESS triggers from the GOQOLIB reference library form.
   - 3.1. Open the GOQOLIB.FMB form (while keeping your new form open as well).
   - 3.2. Right-click the form name of your new form and select **Property Palette**
   - 3.3. Click the **Subclass Information** to update the information
   - 3.4. Select the **Property Class** radio button
   - 3.5. Select *GOQOLIB* as the **Module**
   - 3.6. Select G$_FORM_CLASS, G$_APPL_FORM_CLASS, G$_INQ_FORM_CLASS, or G$_VAL_FORM_CLASS and click **OK**

   Your new form will automatically inherit multiple triggers, specifically PRE-FORM, POST-FORM, and LOAD_CURRENT_RELEASE
4. Modify the trigger named LOAD_CURRENT_RELEASE and set the assignment of the form's release number to the CURRENT_RELEASE item here by including the following statement:

```
:CURRENT_RELEASE := '8.0';
```

5. If you did not reference GOQOLIB as defined in step 3, you must *at a minimum* make the changes specified in this step. (If you completed step 3, you can skip this step, because these items will be automatically inherited.)

   5.1. Modify the PRE-FORM (create one if it does not already exist) to execute the newly created trigger (LOAD_CURRENT_RELEASE). Add the execution of the G$_VERIFY_ACCESS routine as the first step after setting CURRENT_RELEASE. The following code is an example:

```
BEGIN

    EXECUTE_TRIGGER('LOAD_CURRENT_RELEASE');

    G$_CHECK_FAILURE;

    EXECUTE_TRIGGER('G$_VERIFY_ACCESS');

    G$_CHECK_FAILURE;

END;
```

   5.2. Modify the POST-FORM (create one if it does not already exist) to execute the G$_REVOKE_ACCESS routine as the last step. The following code is an example:

```
BEGIN

EXECUTE_TRIGGER('G$_REVOKE_ACCESS';);

G$_CHECK_FAILURE;

END;
```

6. Modify all routines which call any other form to execute the following:

   6.1. EXECUTE_TRIGGER('G$_REVOKE_ACCESS';);

   6.2. G$_CHECK_FAILURE;

   6.3. G$_SECURED_FORM_CALL, instead of issuing their own calls to forms. Once a role is invoked, the grants provided by this role stay in effect until revoked.

Database Scripts

Scripts Called by Other Database Packages and Functions

gurcmpa.sql

This script will spool a script, which will compile all database objects (functions, views, packages, procedures, and triggers) not owned by SYS or SYSTEM, but it does not look only for scripts which are in an invalid state as `guraltr.sql` does.

This was created due to an issue with certain versions of Oracle showing database objects in a valid state after having the public *execute* privilege revoked and yet the objects didn't function correctly until recompiled.

gurgfix.sql

This script generates a grant script for objects which should be granted to the various table owners, but are missing. This ensures that `EXECUTE` permission is in place so the Banner products will work correctly.

It also grants one BANINST1 routine to the BANSECR user, and makes sure `EXECUTE ANY PROCEDURE` has been granted to the default roles.

Logic was added to support credit card payments (if installed). Logic was also added to support the new function that allows users to save the output over the Web.

gurgfix2.sql

This script is the same as gurgfix.sql but assumes that the Self-Service Banner products are not installed.

gurgrnt.sql

This script generates a file of GRANT statements (spooled as newgrnt.sql) based on a model user. You should be logged on as the grantor to run this routine (for example, the security administrator, such as BANSECR. The generated file may be reused, since *newuser* is a variable.

gurgrtb.sql

This script grants `EXECUTE` privilege on the BANINST1-owned stored procedure passed as the first argument to each of the Banner schema owners (ALUMNI, BANIMGR, FAISMGR, FIMSMGR, GENERAL, PAYROLL, POSNCTL, SATURN, TAISMGR, and

WTAILOR), user IDs (such as wfauto), and roles (BAN_DEFAULT_M and BAN_DEFAULT_Q). You may modify this file to add schema owners or roles, or to remove any that are not applicable to the local environment.

gurgrte.sql

The purpose of this script is to grant EXECUTE privilege on the stored procedure passed as the first argument to the *EPRINT* user (EPRINT).

gurgrth.sql

This script grants EXECUTE privilege on the BANINST1-owned stored procedure passed as the first argument to each of the local web server database user IDs (OAS_PUBLIC and BAN_DEFAULT_WEBPRIVS). By default, two of the most common user IDs are provided, but this script should be modified to model the local environment.

gurgrti.sql

The purpose of this script is to grant EXECUTE privilege on the stored procedure passed as the first argument to the Integration Manager (INTEGMGR).

gurgrts.sql

This script grants EXECUTE privilege on the BANINST1-owned stored procedure passed as the first argument to the Banner security owner (BANSECR).

gurgrtw.sql

This script grants EXECUTE privilege on the WTAILOR-owned stored procedure passed as the first argument to the Banner stored procedure owner (usually BANINST1), the Banner database roles (BAN_DEFAULT_M and BAN_DEFAULT_Q), and the local web server database user IDs (OAS_PUBLIC and BAN_DEFAULT_WEBPRIVS). By default, the two most common user IDs are provided, but this script should be modified to model the local environment.

Example Scripts

The examples that follow demonstrate how GRANT EXECUTE privileges are included in different kinds of scripts.

Script Creating a Database Function

```
PDFBDPL.SQL
CREATE OR REPLACE FUNCTION f_ptrbdpl_rowid
(bdca_code varchar2,
bdpl_code varchar2,
profile_date date)
return rowid
as
--
-- FILE NAME..: pdfbdpl.sql
-- RELEASE....: 4.1
-- OBJECT NAME: F_PTRBDPL_ROWID
-- PRODUCT....: PAYROLL
-- USAGE......: Select rowid from pdrbdpl table
--
-- DESCRIPTION:
--
-- This function returns the rowid of the appropriate PTRBDPL record
-- based on BDCA_CODE, BDPL_CODE (Plan) and the as-of-date
-- (profile_date).
..
--
-- DESCRIPTION END
-
,...
/

whenever sqlerror continue;
drop public synonym f_ptrbdpl_rowid;
whenever sqlerror exit rollback;
create public synonym f_ptrbdpl_rowid for f_ptrbdpl_rowid;
REM *** BEGINNING OF GURMDBP MODS ***
REM GRANT EXECUTE ON F_PTRBDPL_ROWID TO PUBLIC;
WHENEVER SQLERROR CONTINUE
start gurgrtb F_PTRBDPL_ROWID
WHENEVER SQLERROR EXIT ROLLBACK
REM *** END OF GURMDBP MODS ***
```

Note

It is very important that you include the `WHENEVER SQLERROR` statements before each command to make sure the script runs all the steps. ■

Script Creating a Database Package

```
PDKLIBS.SQL
create or replace package PDKLIBS is
--
-- FILE NAME..: pdklibs.sql
-- RELEASE....: 5.0
-- OBJECT NAME: PDKLIBS
-- PRODUCT....: PAYROLL
-- USAGE......: Provides common deduction processing for Banner HR.
--
-- DESCRIPTION:
--
-- This package provides common deduction processing for Banner HR and
-- declare externally visible constant,type,cursor,variable and exception.
--
-- Cursors:
--  DednPrecludesC     - select precluded deduction code.
--  FlxsAmountsC       - select flexible benefit data.
...
-- Functions:
--  F_DednAmountType         - retrieve deduction amount type.
--  F_DednFieldEntryAllowed  - determine if deduction entry is allowed.
...
--
-- Procedure:
--  P_DednPayPeriodEnd       - Retrieve the latest pay period end date.
--
-- DESCRIPTION END

...

END PDKLIBS;
/
show errors
set scan on

whenever sqlerror continue;
drop public synonym pdklibs;
whenever sqlerror exit rollback;
create public synonym pdklibs for pdklibs;
REM *** BEGINNING OF GURMDBP MODS ***
REM GRANT EXECUTE ON PDKLIBS TO PUBLIC;
WHENEVER SQLERROR CONTINUE
start gurgrtb PDKLIBS
WHENEVER SQLERROR EXIT ROLLBACK
REM *** END OF GURMDBP MODS ***
```

Note

It is very important that you include the `WHENEVER SQLERROR` statements before each command to make sure the script runs all the steps. ■

Script Creating a Top-Level Web Package

A top-level web package is called from the menu and has an entry in the Web Tailor menu table.

```
PDKLIBS.SQL
create or replace package PDKLIBS is
--
-- FILE NAME..: pdklibs.sql
-- RELEASE....: 5.0
-- OBJECT NAME: PDKLIBS
-- PRODUCT....: PAYROLL
-- USAGE......: Provides common deduction processing for Banner HR.
--
-- DESCRIPTION:
--
-- This package provides common deduction processing for Banner HR and
-- declare externally visible constant,type,cursor,variable and exception.
--
-- Cursors:
--  DednPrecludesC     - select precluded deduction code.
--  FlxsAmountsC       - select flexible benefit data.
...
-- Functions:
--  F_DednAmountType         - retrieve deduction amount type.
--  F_DednFieldEntryAllowed  - determine if deduction entry is allowed.
...
--
-- Procedure:
--  P_DednPayPeriodEnd       - Retrieve the latest pay period end date.
--
-- DESCRIPTION END

...

END PDKLIBS;
/
show errors
set scan on
whenever sqlerror continue;
drop public synonym pdklibs;
whenever sqlerror exit rollback;
create public synonym pdklibs for pdklibs;
REM *** BEGINNING OF GURMDBP MODS ***
REM GRANT EXECUTE ON PDKLIBS TO PUBLIC;
WHENEVER SQLERROR CONTINUE
start gurgrtb PDKLIBS
WHENEVER SQLERROR EXIT ROLLBACK
REM *** END OF GURMDBP MODS ***
```

Note

It is very important that you include the `WHENEVER SQLERROR` statements before each command to make sure the script runs all the steps. ■

Script Creating a Web Package (Not Top Level)

```
BWGKEPAY.SQL
CREATE OR REPLACE PACKAGE bwgkepay
AS
--
-- FILE NAME..: bwgkepay.sql
-- RELEASE....: 7.1
-- OBJECT NAME: BWGKEPAY
-- PRODUCT....: GENWEB
-- USAGE......: Process credit card payments via the EPOS Payment Server.
...

WHENEVER SQLERROR CONTINUE
DROP PUBLIC SYNONYM bwgkepay;
WHENEVER SQLERROR EXIT ROLLBACK
REM *** BEGINNING OF GURMDBP MODS ***
REM GRANT EXECUTE ON BWGKEPAY TO PUBLIC;
WHENEVER SQLERROR CONTINUE
START gurgrtb BWGKEPAY
WHENEVER SQLERROR EXIT ROLLBACK
REM *** END OF GURMDBP MODS ***
CREATE PUBLIC SYNONYM bwgkepay FOR bwgkepay;
```

Note

It is very important that you include the `WHENEVER SQLERROR` statements before each command to make sure the script runs all the steps. ■

Script Creating a Web Tailor Package

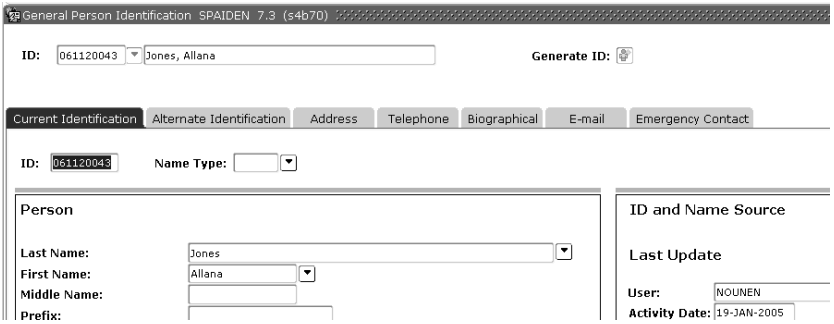
```
TWBKWMNU.SQL
CREATE OR REPLACE PACKAGE TWBKWMNU IS

/* Global type and variable declarations for package */

--
procedure P_OptionPgWebMain(return_code in varchar2 default null);
-- Page that gives user option of creating or updating an existing
--   Main Text page.
..
end TWBKWMNU;
/
show errors

whenever sqlerror continue;

drop public synonym TWBKWMNU;
whenever sqlerror exit rollback
create public synonym TWBKWMNU for TWBKWMNU;
REM *** BEGINNING OF GURMDBP MODS ***
REM GRANT EXECUTE ON TWBKWMNU TO PUBLIC;
WHENEVER SQLERROR CONTINUE
start gurgrtw TWBKWMNU
WHENEVER SQLERROR EXIT ROLLBACK
REM *** END OF GURMDBP MODS ***
set scan on
```

Note

It is very important that you include the `WHENEVER SQLERROR` statements before each command to make sure the script runs all the steps. ■

Tab-Level Security

Some Banner forms are displayed as two or more windows, with navigation between the windows provided by a set of tabs at the top of the form. By clicking the *Biographical* tab, for example, you can navigate to the Biographical window of the form shown below. Forms with this navigational structure are called *tab-enabled forms* or *tabbed forms*, and each window in one of these forms is commonly called a *tab*.

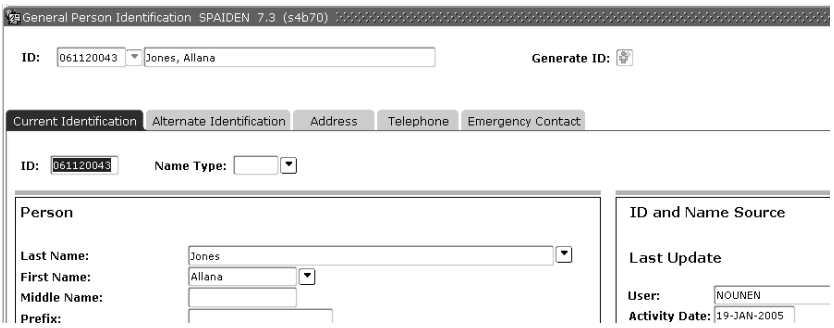


Banner security is based on controlling a user's access to Banner objects, such as forms. Through security setup in the Oracle/Banner Security Maintenance Form (GSASECR), each user is granted access to each of the forms that the user needs. A user is not able to access a Banner form unless he or she has been granted access in GSASECR.

Release 7.5 of Banner General introduced an extension of the user/object security model. Within certain forms, you can selectively restrict a user's access to tabs (where a tab is a window of a tabbed form).

Tab-level security was implemented first in identification forms, the forms that contain personal information tied to a person's ID record. Within these forms, you can now allow a user to access one tab while preventing that same user from accessing another tab within the same form.

For example, suppose a user's job involves updating student addresses, but that user has no need to see e-mail information. You could give the user access to the Address tab, while preventing access to the E-mail tab. The following example shows the SPAIDEN form with the E-mail, Biographical, and Additional ID tabs hidden from the current user.



The forms listed below have been enhanced for tab-level security.

- General Person Identification (SPAIDEN)
- Identification Form (PPAIDEN)
- Person Identification (FOAIDEN)
- Advancement Identification (APAIDEN)
- Banner Distributed Security (GSADSEC)

Note

Tab-level security cannot be applied to other baseline Banner forms until those forms have been enabled by SunGard. Adding tab security records on GSASECR for a form, without coding changes to that form, will not enable tab security on the form.

Resolving Tab-Level Permissions

If a security class has permissions for a form, and you *don't* set up tab-level security records, the members of that class will have full access to all of that form's tabs. There is no need to set up tab-level security records for a form, except where you wish to create restrictions for individual tabs.

A user's permissions for a tab can never exceed the permissions for the form containing the tab. For example, if a user has query-only access to the FOAIDEN form, you cannot give the user full access to the Address tab on FOAIDEN.

As with object permissions, a user's direct permissions for a tab take priority over permissions that are granted through a class.

When a user has access to an form object through two or more security classes, only one class's permissions can be chosen and applied. The classes are given priority through an alphabetical rule.

- The class names with maintenance roles for the object (roles ending with *_M*) are sorted alphabetically, and the first in the list is used.
- If there are no classes with *_M* roles, all other classes are sorted alphabetically, and the first in the list is used.

Note

For local changes to this precedence order, please see the `g$_get_role_for_object_fnc` and `g$_get_tab_security_fnc` functions in gspsecr.sql.

The class selected through the alphabetical rule controls the tab-level permissions that are applied in this instance. Tab-level permissions that exist in the user's other classes are ignored.

Setting Up Tab-Level Security for a Form

To set up tab-level security for SPAIDEN, PPAIDEN, FOAIDEN, and APAIDEN, or any other tab-enabled form.

1. Review the tab access restrictions and (optionally) create additional restrictions.

   Review the records for the each of the form objects in GSASECR's Objects window. (See the screen print on page 2-22.) For SPAIDEN, PPAIDEN, FOAIDEN, and APAIDEN, the Current Identification tab on each form has a *Q* restriction. This means that users with access to the form must have (at least) query access to the Current Identification tab; this tab cannot be hidden.

   If you wish, you can add restrictions for other tabs. Bear in mind that the tab restrictions set in GSASECR's Objects window are not restriction on users; instead, they restrict *your* ability to limit what users can see.

2. Add the appropriate tab-level security records for each security class, for each form that you want to restrict. (A security class is a collection of Banner objects grouped together for security setup purposes.)

   2.1. Navigate to the Classes window of GSASECR. Select a security class for which you want to set up tab-level permissions.

   2.2. Click the Objects button. Select an object (for example, SPAIDEN).

   2.3. Create the tab-level security records for each of the tabs that you want to restrict for that security class and object. (See "User/Class Privilege Maintenance" on page 2-10.)

3. Add tab-level security records for individual users, where necessary. This is done in the same way as setting up security classes, except that your starting point is the Users window (**Modify** option) of GSASECR.

4. Test the results to make sure that your tab-level security setup works as expected.

5. Review your locally-created options menu records on GUAOPTM and make any necessary adjustments.

Setting Up Tab-Level Security for a New Tab

When a new tab is added to a form that is already enabled for tab-level security, you should consider setting up tab-level security records for the new tab. If you do nothing, each user's form-level privileges will apply to the newly added tab. That is, any user with access to the form will have access to the new tab.

To set up security for a new tab:

1. In GSASECR's Objects window, select the form and review the tab's access restrictions.

2. Add the appropriate tab-level security records for each security class that will have restricted access to the new tab.

   2.1. Navigate to the Classes window of GSASECR. Select the security class.

   2.2. Click the Objects button. Select the form with the new tab.

   2.3. Create a tab-level security record for the new tab.

3. Add tab-level security records for individual users, where necessary. This is done in the same way as setting up security classes, except that your starting point is the Users window of GSASECR.

4. Test the results to make sure that your tab-level security setup works as expected.

Special Security Objects

There are several security objects that exist in the Security Object table (GURAOBJ), but do not exist in the Banner Object Base table (GUBOBJS). These objects represent specific functions in Banner that are controlled through the security system. When a user is granted access to any of these objects (directly, through a class, or through a security group) the user has privileges to perform the corresponding Banner function.

Each of these special security objects, along with the functionality that it controls, is described below.

BROADCAST Security Object

You can send broadcast messages to all users, which will appear on the General menu form in the Banner Broadcast Messages section. To send a broadcast message, create it on GUAMESG as usual, and use BASELINE as the recipient.

Whenever you receive a broadcast message:

- The Broadcast Message icon on the toolbar will be enabled.
- The message will appear on the General Menu form (GUAGMNU) in the Broadcast Messages section.

To give users the permission to send Broadcast messages you must give the user access to the BROADCAST security object directly or through a class or group which contains this object.

CHANNELS Security Object

During security setup, all users or classes that need access to Luminis Channels for Banner must have this object added to their privileges. See "Proxy Authentication for Channels and Other JDBC-Based Connections" on page 5-2.

EXTENDED_QUERY Security Object

The Extended Query function allows a user to enter a colon or an ampersand as a query condition in a field which will open the extended query window where an additional query condition can be specified.

The `EXTENDED_QUERY` security object provides a way to restrict which users can perform this function. The object is delivered as part of the `BAN_GENERAL_C` class.

If you want to restrict access to this feature, remove the `EXTENDED_QUERY` object from the `BAN_GENERAL_C` class and grant access through a more restrictive class or directly to specific users. If a user attempts to use Extended Query without having a grant for this security object the user will receive the following error: **WARNING* Extended Query functionality has been disabled.*

Note

In Oracle 10gR2, the Extended Query function is set to a default of *TRUE* (or disabled), to help prevent the possibility of SQL injection attacks.

If you wish to continue use of the `EXTENDED_SEARCH` feature in Banner for certain users, the `FORMS_RESTRICT_ENTER_QUERY` setting in the `default.env` file must be set to *FALSE*.

If `FORMS_RESTRICT_ENTER_QUERY` is set to *TRUE* and the user attempts to use Extended Query, the user will receive the following error: *FRM-40367: Invalid criteria in field nnn in example record.*

SSN_SEARCH Security Object

The `SSN_SEARCH` object gives users the ability to enter a value in a Banner ID field (SPRIDEN ID name search) in Banner INB and the Banner system will automatically search the SSN field (`SPBPERS_SSN`) for a matching value and return the person's ID to the ID field.

The SSN search feature is not automatically available to all Banner users. To make this feature available to Banner users, you must explicitly give them permission in the Banner Security Maintenance Form (GSASECR).

Permission for SSN searching is managed by giving users access to the object SSN_SEARCH.

As with any Banner object, you can give user permissions for this object by:

- directly adding the object to an individual user's list of objects
- adding the object to a security class which is assigned to the user

If you want to make this feature available to all users, you can add the SSN_SEARCH object to a class that is available to all users, for example BAN_GENERAL_C.

Security for Population Selection and Letter Generation

This section offers suggestions for setting up security for Banner's population selection and letter generation features. You can create special roles with just enough privileges for GLOLETT, GLBLSEL, and GLBDATA, following the examples shown below.

Automatic Letter Compilation Process (GLOLETT)

The Automatic Letter Compilation Process (GLOLETT) compiles variables and populations selection rules. These rules can then be used as parameters in other Banner processes. A simple way to set up security for GLOLETT is to give a user permission for the GLOLETT object with a role that has system privileges for selecting from any table (such as BAN_DEFAULT_M).

If you want to craft the GLOLETT user's security privileges more narrowly and limit the tables that GLOLETT can access, you can create a role called, for example, BAN_GLOLETT_BASE_GRANTS, that has only the base grants needed for running the process.

The table below shows the base grants that this role should be assigned on GSASECR's Role Privileges window.

Object Name	Owner	Object Type	Select	Insert	Update	Delete	Execute
ALL_IND_COLUMNS	SYS	VIEW	Y				
ALL_TAB_COLUMNS	SYS	VIEW	Y				
GJBPRUN	GENERAL	TABLE	Y			Y	
GJBRSLT	GENERAL	TABLE	Y	Y	Y	Y	
GLBSLCT	GENERAL	TABLE	Y				

Object Name	Owner	Object Type	Select	Insert	Update	Delete	Execute
GLBVRBL	GENERAL	TABLE	Y				
GLRAPPL	GENERAL	TABLE	Y				
GLRCMPL	GENERAL	TABLE	Y	Y	Y	Y	
GLRSFRM	GENERAL	TABLE	Y				
GLRDLCT	GENERAL	TABLE	Y				
GLRVFRM	GENERAL	TABLE	Y				
GLRVRBL	GENERAL	TABLE	Y				

This role should also be assigned the system privilege EXECUTE ANY PROCEDURE.

To appy this role to a specific user account:

- Add the user to a class that includes GLOLETT with the BAN_GLOLETT_BASE_GRANTS role
- Add the user to a security group that includes GLOLETT with the BAN_GLOLETT_BASE_GRANTS role
- Assign GLOLETT with the BAN_GLOLETT_BASE_GRANTS role directly to the user's privileges

With these base grants, the user can run the GLOLETT process (for compiling population selections and variables), but the user will not have the ability to select from any table via population selection. You can add additional table grants to the BAN_GLOLETT_BASE_GRANTS role (or a copy of this role) to allow a user to select from specific tables.

You might want to create separate roles for the GLOLETT process, with different additional tables, to be used for different groups of users.

Letter Extract Process (GLBLSEL)

Just as with the GLOLETT process described above, you can establish a role specifically for the Letter Extract Process (GLBLSEL). By creating a GLBLSEL role with specific limited privileges, you can prevent a user from selecting data from unauthorized tables and including that data in letter output.

The following table shows the setup for an example role, BAN_GLBLSEL_BASE_GRANTS, with the minimum grants needed to use GLBLSEL. You can add additional table grants for tables that the user should have permission to access. You may want to create separate roles for the process, with different additional tables, to be used for different groups of users.

Object Name	Owner	Object Type	Select	Insert	Update	Delete	Execute
GJBRSLT	GENERAL	TABLE	Y	Y	Y	Y	
GLBAPPL	GENERAL	TABLE	Y				
GLBEXTR	GENERAL	TABLE	Y				
GLBVRBL	GENERAL	TABLE	Y				
GLRCALC	GENERAL	TABLE	Y	Y	Y	Y	
GLRCMPL	GENERAL	TABLE	Y				
GLRCOLR	GENERAL	TABLE	Y	Y	Y	Y	
GLRORDR	GENERAL	TABLE	Y	Y	Y	Y	
GLRVRBL	GENERAL	TABLE	Y				
GTVLETR	GENERAL	TABLE	Y				
GUBINST	GENERAL	TABLE	Y				
GURMAIL	GENERAL	TABLE	Y				
GUVLETR	BANINST1	VIEW	Y				
ROBINST	FAISMGR	TABLE	Y				
RORVIEW	FAISMGR	TABLE	Y	Y	Y	Y	
SPRCOLR	SATURN	TABLE	Y	Y	Y	Y	
SPRIDEN	SATURN	TABLE	Y				
STVATYP	SATURN	TABLE	Y				

Population Selection Extract (GLBDATA)

If a user had compiled a population selection prior to assigning the new roles described above for GLOLETT and GLBLSEL, it is possible that the user could run GLBDATA and extract the population. In order to avoid this potential situation, you can establish base grants for GLBDATA in a role called, for example, BAN_GLBDATA_BASE_GRANTS, with the role privileges listed below.

Object Name	Owner	Object Type	Select	Insert	Update	Delete	Execute
ALL_TAB_COLUMNS	SYS	VIEW	Y				
GJBPRUN	GENERAL	TABLE	Y	Y	Y	Y	
GJBRSLT	GENERAL	TABLE	Y	Y	Y	Y	
GLBAPPL	GENERAL	TABLE	Y				
GLBEXTR	GENERAL	TABLE	Y	Y	Y	Y	
GLBSLCT	GENERAL	TABLE	Y	Y	Y	Y	
GLRAPPL	GENERAL	TABLE	Y				
GLRCMPL	GENERAL	TABLE	Y				
GLRSFRM	GENERAL	TABLE	Y				
GLRSLCT	GENERAL	TABLE	Y	Y	Y	Y	
GLRVRBL	GENERAL	TABLE	Y				
GUBINST	GENERAL	TABLE	Y				

Working with the BANSECR Account

The tables and views that are used to enforce the security are owned by an Oracle account called BANSECR. The objects that this account owns are rarely granted to anyone. All the security functions are provided by stored procedures that are granted to public. This is the only access other users typically need.

BANSECR is the only user that can change other users' passwords, unless you have created distributed security users that have been granted the `ALTER ID` privilege.

Objects Owned by BANSECR

BANSECR is the owner of many objects involved in Banner's security system. BANSECR's objects, other than tables, are listed in the table below. A second table follows, which lists all of the tables owned by BANSECR.

Object Name	Type	Description
BANINST1_SQL_PKG	Synonym	Synonym that points to a BANINST1 owned package that is granted only to BANSECR. The GSASECR form will pass grant commands to this procedure when a Banner-owned object has to be granted to a role.
BANNER_SECURITY_AUDIT_SEQUENCE	Sequence	Sequence used in Banner security audit tables to ensure unique primary key.
F_GETDEFROLES	Function	Function to return the default roles for the username
G$_AUTHORIZATION_PKG	Package	This package contains security routines that are used by job submission. It also contains routines used by the security front end to synchronize Oracle grants with the Banner class definitions.
G$_CHK_AUTH	Public Synonym	Synonym for BANSECR's G$_AUTHORIZATION_PKG.
G$_OBJECT_SECURITY	Package	Uses Oracle FGA feature to remove objects from being presented to a user if they are not permitted to access them.
G$_OREP_SECR_PKG	Package	Oracle Reports security package.
G$_OREP_SECR	Public Synonym	Synonym for BANSECR's G$_OREP_SECR_PKG
G$_SECURITY	Public Synonym	Synonym for BANSECR's G$_SECURITY_PKG
G$_SECURITY_PKG	Package	Procedures used by end users to verify their access and perform object authentication.
G$_VPDI_SECURITY	Package	Procedure to set the home context of the user logging in. This is also used when the user has access to more than one institution and chooses one

Object Name	Type	Description
G$_VPDI_SECURITY	Public Synonym	Synonym for BANSECR's G$_VPDI_SECURITY package
GP_UDC_ORACLE_ID	Package	This package performs tasks associated with create Oracle IDs related to UDC identity management.
GP_UDC_ORACLE_ID	Public Synonym	Synonym for BANSECR's GP_UDC_ORACLE_ID package
GSPCRPT	Package	This package is owned by the Bansecr owner. It handles encryption in Banner. This is a wrapper package for DBMS_CRYPTO.
GSPCRPT	Public Synonym	Synonym for BANSECR's GSPCRPT package.
GSPPRXY	Package	This package is the Oracle proxy security package. It checks user mappings and returns oracle user it should connect as.
GSPPRXY	Public Synonym	Synonym for BANSECR's GSPPRXY package.
GSPVPDI	Package	This package supports the VPDI Interface by returning valid MEP codes.
GSPVPDI	Public synonym	Synonym for BANSECR's GSPVPDI package.
GT_<tablename>_AUDIT_ROW	Trigger	Triggers to audit changes in Banner security related tables
GT_LOGIN_AUDIT_ACCESS	Trigger	Trigger to audit Oracle logons by Banner related User Ids
GT_LOGOFF_AUDIT_ACCESS	Trigger	Trigger to audit Oracle logoffs by Banner related Users Ids
GT_LOGIN_SET_VPDI_ CONTEXT	Trigger	Trigger to set Multi-Entity Processing (MEP) institution codes.
GUBOSEQ	Sequence	One up sequence number used to sequence the records in the GURSQLL table.

Object Name	Type	Description
GUVDFTR	View	Show a user's default role.
GUVOWNR	View	Banner security access of objects by distributed security users.
GUVRPRV	View	Returns the table permissions given to a role. This view de-normalizes the permissions stored in the system catalog. This view is used by the role maintenance screen of the GSASECR form.
GUVUACC	View	Banner security access of objects by user.
GUVUOBJ	View	Intermediate view used by GUVUACC.

The tables owned by BANSECR are listed below.

Object Name	Type	Description
GJRINVC	Table	GJRINVC: Job Submission Character Validation Table contains rows of characters which are either allowed or prohibited for use in various Job Submission parameters based on Type column.
GTVCALN	Table	GTVCALN: Validation entries for calendars used in security logon validation.
GTVCLAS	Table	GTVCLAS: Validation table of user classes defined to the BANNER security system.
GTVOWNG	Table	GTVOWNG: Validation entries for the security owner groups used in distributed security.
GTVSGRP	Table	GTVSGRP: Validation entries for security groups.

Object Name	Type	Description
GTVVPDI	Table	GTVVPDI: VPD Institution Code Validation Table
GUBAROL	Table	GUBAROL: This table stores audit information for the GUBROLE table.
GUBIPRF	Table	GUBIPRF: Site profile record. This table contains only one record. It defines what level of security is being used, seed numbers and if version number checking is active.
GUBROLE	Table	GUBROLE: This table stores the encrypted passwords for the Banner roles. This table is automatically maintained when passwords are generated or regenerated for the Banner roles.
GURAAOB	Table	GURAAOB: This table stores audit information for the GURAOBJ table.
GURAATB	Table	GURAATB: This table stores audit information for the GURATAB table.
GURABGP	Table	GURABGP: This table stores audit information for the GURBGRP table.
GURABPI	Table	GURABPI: This table stores audit information for the GURABPI table.
GURABPR	Table	GURABPR: This table stores audit information for the GORPBPR table.
GURACAL	Table	GURACAL: This table stores audit information for the GURCALN table.
GURACGP	Table	GURACGP: This table stores audit information for the GURCGRP table.
GURACLS	Table	GURACLS: This table stores audit information for the GURUCLS table.
GURADMN	Table	GURADMN: This table stores audit information for the GOBFDMN table.
GURADPI	Table	GURADPI: This table stores audit information for the GORFDPI table.

Object Name	Type	Description
GURADPL	Table	GURADPL: This table stores audit information for the GORFDPL table.
GURADSU	Table	GURADSU: This table stores audit information for the GURDSUR table
GURAEAC	Table	GURAEAC: This table stores audit information for the GOBEACC table.
GURAEOB	Table	GURAEOB: This table stores audit information for the GOBFEOB table.
GURAGAC	Table	GURAGAC: This table stores audit information for the GOBFGAC table.
GURAGBP	Table	GURAGBP: This table stores audit information for the GORFGBP table.
GURAGUS	Table	GURAGUS: This table stores audit information for the GORFGUS table.
GURAINV	Table	GURAINV: This table stores audit information for the GJRINVC table.
GURAIPF	Table	GURAIPF: This table stores audit information for the GUBIPRF table.
GURALGN	Table	GURALGN: This table stores information related to logins to Oracle by users of Banner as defined in GURUCLS.
GURALOG	Table	Banner Security Violation Log. This table contains a record of several types of Banner Security violations that have occurred.
GURAMSK	Table	GURAMSK: This table stores audit information for the GORDMSK table.
GURAOBJ	Table	GURAOBJ: This table defines all valid Banner objects and what the current version number is. This table also defines the default role to be used when the object is first granted to the user or a class.

Object Name	Type	Description
GURAOGP	Table	GURAOGP: This table stores audit information for the GUROGRP table.
GURAOWG	Table	GURAOWG: This table stores audit information for the GUROWNG table.
GURAOWN	Table	GURAOWN: This table stores audit information for the GUROWNR table.
GURAPRD	Table	GURAPRD: This table stores audit information for the GORFPRD table.
GURAPUD	Table	GURAPUD: This table stores audit information for the GOBFPUD table.
GURASGR	Table	GURASGR: This tables stores audit information for the GTVSGRP table.
GURATAB	Table	GURATAB: This table defines all the forms and their related tabs that can be used for tab based security.
GURAUGP	Table	GURAUGP: This table stores audit information for the GURUGRP table.
GURAULG	Table	GURAULG: This table stores information related to logins to Oracle by users of Banner. A Banner user is defined as a user having an entry in GURUCLS or GURUOBJ.
GURAUOB	Table	GURAUOB: This table stores audit information for the GURUOBJ table.
GURAUSI	Table	GURAUSI: This table stores audit information for the GURUSRI table.
GURAUTB	Table	GURAUTB: This table stores audit information for the GURUTAB table.
GURAVCL	Table	GURAVCL: This table stores audit information for the GTVCLAS table.
GURAVOG	Table	GURAVOG: This table stores audit information for the GTVOWNG table.

Object Name	Type	Description
GURBGRP	Table	GURBGRP: This table defines business profiles belonging to a security group.
GURCALN	Table	GURCALN: This table defines calendars used for logon verification.
GURCGRP	Table	GURCGRP: This table defines classes belonging to a security group.
GURDSUR	Table	GURDSUR: This table stores rules used in creating new distributed security users.
GURLOGN	Table	GURLOGN: This table stores information related to logins to Oracle by users of Banner as defined in GURUCLS.
GUROGRP	Table	GUROGRP: This table defines individual objects belonging to a security group.
GUROWNG	Table	GUROWNG: This table defines groups of users for distributed security.
GUROWNR	Table	GUROWNR: This table defines the objects owned by distributed security users and their access for each object.
GURSQLL	Table	GURSQLL: Log of SQL commands that are dynamically generated and executed by the security system front end.
GURUACC	Table	GURUACC: Object Access by User
GURUCLS	Table	GURUCLS: Table to track what BANNER security classes a user is authorized to access.
GURUGRP	Table	GURUGRP: This table defines users belonging to a security group.

Object Name	Type	Description
GURUOBJ	Table	GURUOBJ: This table defines the type of access, by User ID, for each Banner object.
GURUSRI	Table	GURUSRI: VPD Institution/Banner User Table.
GURUTAB	Table	GURUTAB: This table defines all the forms and their related tabs that are in use for tab based security for a user or a class.

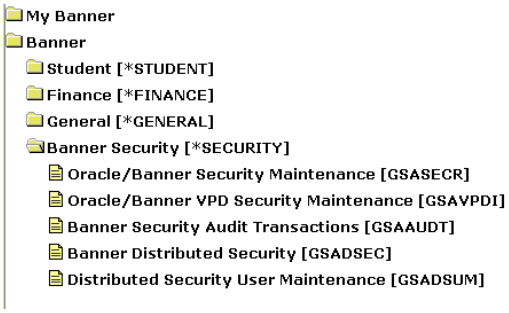
2 Maintaining User Accounts

A Banner user account is essentially an Oracle user ID tied to specific Banner permissions. The security maintenance forms GSASECR and GSADSEC are used to create and maintain Banner user accounts. GSASECR also lets you set institution-level options for security and review a log of security events and potential security violations.

*SECURITY Menu

*SECURITY

The *SECURITY menu, new with Release 8.0, provides convenient access to the security administration forms.



Before Release 8.0, the Banner security forms GSASECR and GSAVPDI were not accessible through menus; they could only be run as standalone forms in separate sessions.

Note

This *SECURITY menu is visible to any user who is a member of a security class with permissions for any of the Banner security forms. (Those users still cannot access the forms—only BANSECR and other explicitly assigned BANSECR_*xxx* user IDs can access the security forms). It is recommended that you review security classes and remove any permissions to security forms where necessary to prevent regular users from seeing the *SECURITY menu.

Security Maintenance (GSASECR)

The Security Maintenance form (GSASECR), helps you maintain Oracle and Banner accounts and control their access to Banner objects.

The GSASECR form must be executed from either the BANSECR account or a BANSECR_*xxx* account. There are only a few public synonyms for any of the BANSECR tables, and no end-user should ever obtain grants to the underlying tables and views.

GSASECR is a tabbed form with seven tabs. You can navigate to any of the seven functions by clicking the corresponding tab at the top of the page.

Tab	Description
Users	Oracle User Maintenance. In the Oracle user maintenance tab, Banner/ Oracle users can be created, altered or deleted. The objects that can be accessed from the user account are also maintained by this function.
Violations	Security Violations. This allows you to review the security log. This log indicates hack attempts and other failures, and it must be reviewed periodically. It also lets you clear out the log file.
Classes	Class Maintenance. In the class maintenance tab, object permissions that are common to many users can be defined and shared by the users.
Objects	Banner Object Maintenance. In the object maintenance tab, you can define Banner objects to the security system.
Roles	Role Maintenance. In the Oracle role maintenance tab, roles and their corresponding privileges are maintained.
Institution Profile	Profile Maintenance. The institution profile record controls security settings that affect the entire Banner instance.
Dynamic SQL History	The Dynamic SQL History window shows SQL history records stored in the GURSQLL table.

Users

The Users window is a combination Oracle and Banner user maintenance system.

Oracle IDs can be created, altered, or removed. A duplicate function is also provided, which clones the Oracle account and duplicates the Banner security information for the selected user. This includes direct object grants and class enrollment information.

Note

This user maintenance tab provides the functions necessary for Banner security setup. It is not intended to replace all the functionality of a standard Oracle user maintenance tool. ■

Warning

Do not delete any of the SunGard-delivered Oracle IDs! These are essential for the proper functioning of the Banner system. See the *Banner General Technical Reference Manual* for a list of SunGard-delivered IDs. Remember to change the passwords of these accounts. ■

Main Window

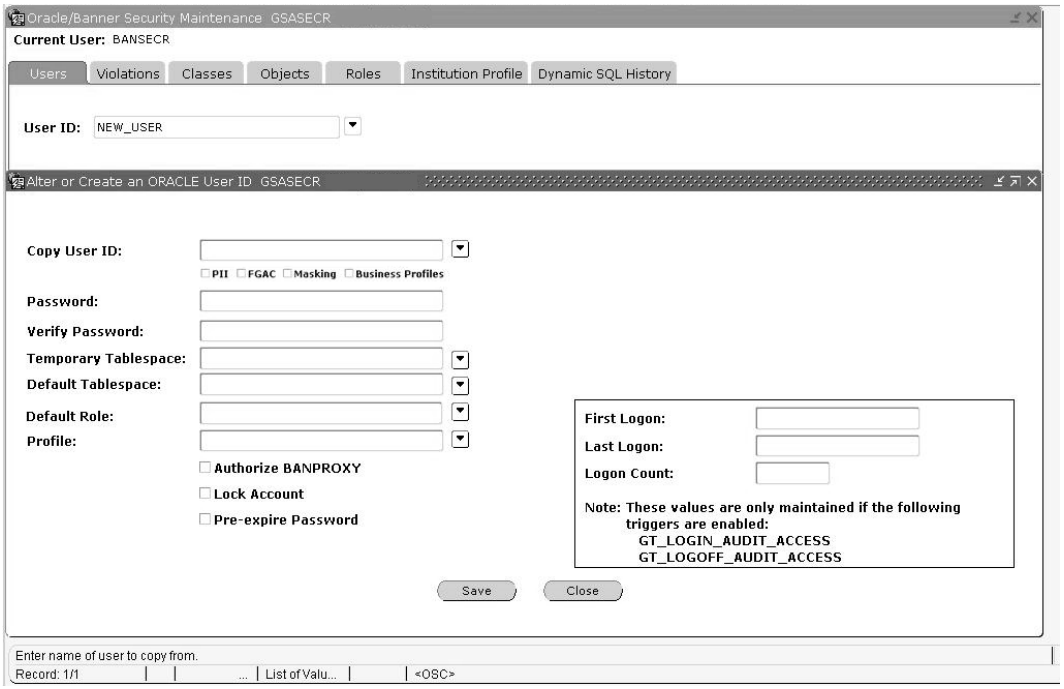
To get started in the Users window, enter or search for an existing Banner user ID, or click Create to create a new user account.

Field	Description
User: Create	Displays the Alter or Create an ORACLE User ID window to create a new user account.
User: Banner Rules	Displays the Setup Login Rules for a User window to create or maintain login rules for a user account.
User: Alter	Displays the Alter or Create an ORACLE User ID window to modify an existing user account.
User: Delete	Deletes this Oracle user and everything the account owns. For technical details, see "Deleting a User Account" on page 2-41.
Permissions: Modify	Reviews or modifies direct object grants to the user and the classes the account is enrolled in. Also provides a copy function to copy privileges from another user.
Permissions: Summary	Creates an alphabetized pop-up list summarizing all objects that the user can access. It is created from the objects directly granted to the user, and the objects that are granted to the classes and security groups in which the user is enrolled.

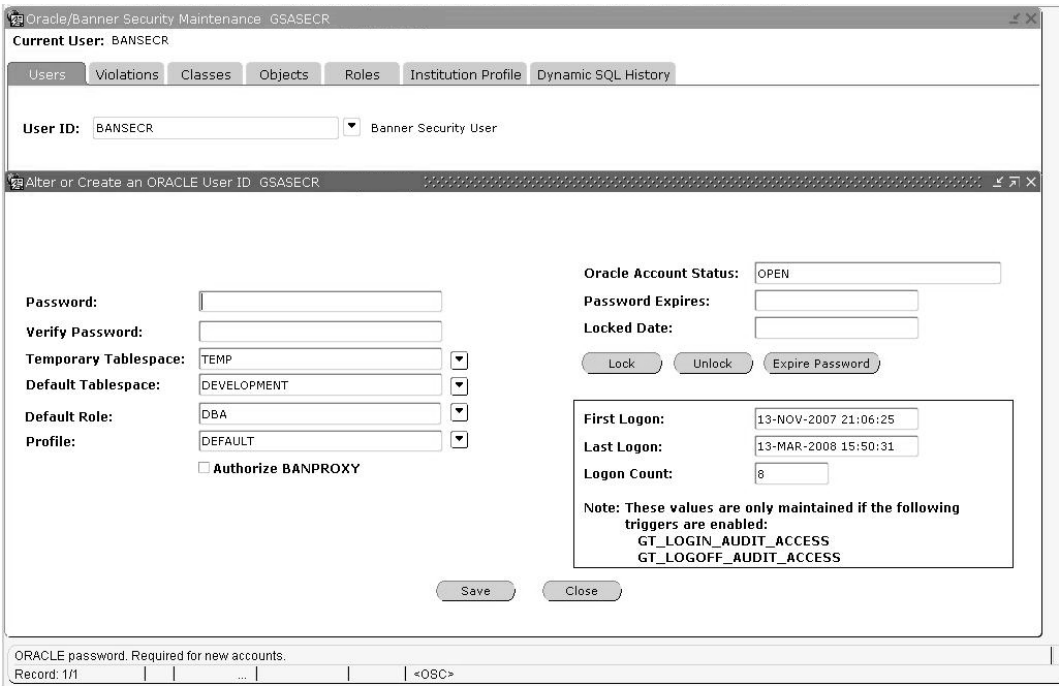
Alter or Create an ORACLE User ID

This window is presented when you click the **Create User** button or the **Alter User** button on the main window of the Users tab. This window looks basically the same in either case, with just a few exceptions:

- In Create mode, you see an option to **Copy User ID** and check boxes for **Lock Account** and **Pre-expire Password**.



- In Alter mode you see buttons to **Lock Account**, **Unlock Account**, and **Expire Password** and also the display of **Current Status**, **Date of Password Expiration**, and **Date Account was Locked**.



For more information on creating new accounts, see "Creating a New User" on page 2-38.

Note

Global rules for creating account names and passwords may have been established on the Institution tab on GSASECR. Also, rules for expiring a newly created account or forcing profile validation may have been established.

Field	Description
Copy User ID	Specify an existing user ID that will be the basis of the new account. Security permissions from the original account are duplicated. For a list of information copied from the original account and other details, see "Copying Another User Account" on page 2-40. **Note:** This field displays only when creating a new User ID.
PII	When copying an existing user ID to create a new user ID, select this check box to copy the existing user ID's PII restrictions. If the check box is grayed out (inactive), the selected existing user does not have any PII information to copy.

Field	Description
FGAC	When copying an existing user ID to create a new user ID, select this check box to copy the existing user ID's FGAC value-based security restrictions. If the check box is grayed out (inactive), the selected existing user does not have any FGAC information to copy.
Masking	When copying an existing user ID to create a new user ID, select this check box to copy the existing user ID's masking restrictions. If the check box is grayed out (inactive), the selected existing user does not have any masking information to copy.
Business Profile	When copying an existing user ID to create a new user ID, select this check box to copy the existing user ID's business profile information. If the check box is grayed out (inactive), the selected existing user does not have any business profile information to copy.
Password	Enter a password for the user. **Note:** There are institutional settings that affect your choice of password. See "Creating a New User" on page 2-38.
Verify Password	Enter the password again to verify it.
Temporary Tablespace	An Oracle tablespace where temporary tables and sort areas are to be created.
Default Tablespace	An Oracle tablespace where permanent tables are to be created if a specific local is not given. Most users should not have permission to create tables so this setting does not matter. If they do have create table permission, it is best if their default tablespace is not a Banner tablespace. This way end-users do not use space allocated for Banner tables and they will not fragment the Banner tablespace with odd size tables.
Default Role	Specify the default role(s) for the account. See "Default Role" on page 2-40 for an explanation of the default role and why it is necessary.
Profile	Enables the execution of an package for stronger password validation as well as password life, reusability, and grace period rules.
Authorize BANPROXY	Allows this user to connect to Banner through proxy connections by way of the Oracle user BANPROXY.

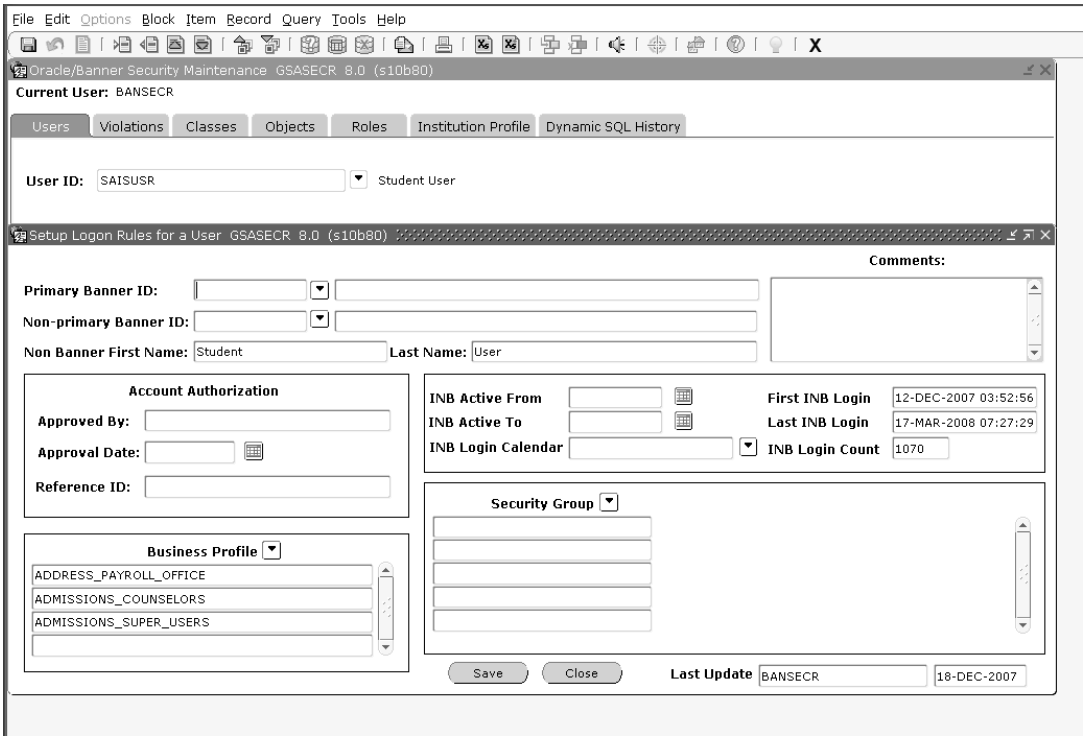
Field	Description
Lock Account (check box)	Check this box to lock the account so the user cannot access it. **Note:** This field displays only when creating a new User ID.
Lock Account (button)	Click this button to lock the account so the user cannot access it. **Note:** This button displays only when altering an existing User ID.
Unlock Account	Click this button to unlock a previously locked account so the user can access it. **Note:** This button displays only when altering an existing User ID.
Pre-Expire Password (check box)	Causes the account password to expire immediately. The user will be required to change the password on the next logon attempt. **Note:** The Institution Profile tab on GSASECR can establish rules that force new accounts to be created in a pre-expired state. **Note:** This field displays only when creating a new User ID.
Expire Password (button)	Causes the account password to expire immediately. The user will be required to change the password on the next logon attempt. **Note:** This button displays only when altering an existing User ID.
Oracle Account Status	The current status of the account (whether the user is able to log in). **Note:** This field displays only when altering an existing User ID.
Password Expires	The date on which the user's password expired. **Note:** This field displays only when altering an existing User ID.
Locked Date	If the account is currently locked, the date on which it became locked. **Note:** This field displays only when altering an existing User ID.

Field	Description
First Logon	The date and time of the user's earliest logon recorded on the GURALGN audit table. This field is populated only if the GT_LOGIN_AUDIT_ACCESS trigger is enabled.
Last Logon	The date and time of the user's most recent logon recorded on the GURALGN audit table. This field is populated only GT_LOGOFF_AUDIT_ACCESS trigger is enabled.
Logon Count	The total number of times the logons for the user, as recorded in the GURALGN table. This field is populated only if the GT_LOGIN_AUDIT_ACCESS trigger is enabled.
Save	Saves your changes without closing the window.
Close	Closes this window without saving any changes.

Setup Logon Rules for a User

The Setup Logon Rules for a User window, introduced in Release 8.0, allows you to associate a Banner user ID with an existing Banner ID record (a PIDM). It also provides a convenient place to make many security choices for the Banner user ID.

To access this window, click the **Banner Rules** button on the Users window.



Field or Button	Description
Primary Banner ID	A Banner ID (PIDM) record to be associated with the Banner/ Oracle user ID. Entering an ID in this field creates an Enterprise Access (GOBEACC) record, which associates the user ID to the Banner ID.
Non-primary Banner ID	A second Banner ID record associated with the Banner/Oracle user ID. This field allows you to associate the Banner ID with the user ID without creating a GOBEACC record.
Non Banner Name	A name associated with the user ID, if different from the names associated with the Banner ID records. This can be used when creating a user ID for a user with no Banner ID (SPRIDEN PIDM) record. Separate fields are available to enter a first and last name.
Comments	Comments on the user ID record. For example, you can use this field to make note of the reasons for specific security settings.
Account Authorization	Use this section to record formal approvals for the creation of the account.
Approved By	The person who provided approval for the creation of this user ID.
Approval Date	The date the account creation approval was received.
Reference ID	An optional field to record account creation approval information. This can be used to store a document reference number related to the approval of this user account.
INB Active From	The beginning date of the period when this user will be permitted access to Banner. You can use this field to create a user ID ahead of time, before an account is active. If no date is entered, the account is active as soon as you finish creating it in GSASECR. **Note:** This date controls only INB access. It has no impact on the status of the Oracle account or a Self-service Banner account.
INB Active To	The ending date of the period when this user will be permitted access to Banner. You can use this field to disable a user ID ahead of time, for example, if an employee's departure date is known in advance. If no date is entered, the user's access is open-ended. **Note:** This date controls only INB access. It has no impact on the status of the Oracle account or a Self-service Banner account.

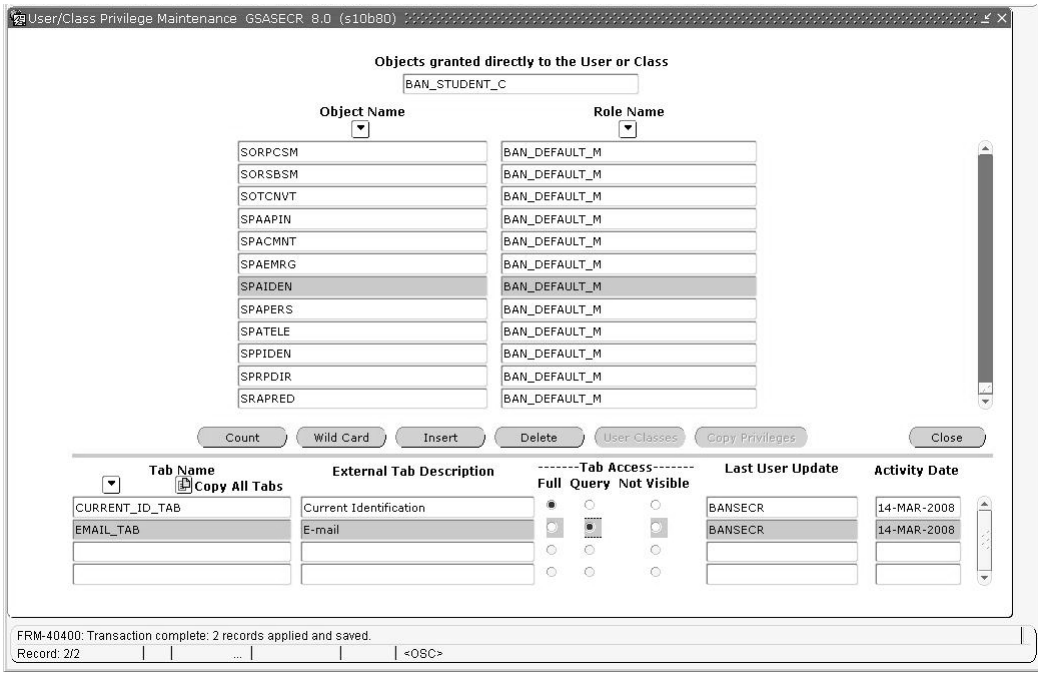
Field or Button	Description
INB Login Calendar	You can optionally assign a login calendar (created on the GSADSEC form) to the User ID. A login calendar limits the days of the week and times of day when a user is permitted to log in.
First INB Login	The date and time of the user's first login.
Last INB Login	The date and time of the user's most recent login.
INB Login Count	The total number of times the user has logged into INB as a Banner user.
Business Profile	Use this section to assign the user ID to one or more business profiles. Business profiles are groups of users assigned to specific FGAC rules for Value-Based Security (VBS) and Personally Identifiable Information (PII) security, as well as rules for data masking.
Security Group	Use this section to assign the user ID to one or more security groups. **Note:** Security groups are created on the GSADSEC form.
Last Update	The user ID and date of the latest update to this user's security record.
Save	Click the **Save** button to save all changes to the record.
Close	Click the **Close** button to return to the Users window.

User/Class Privilege Maintenance

The same window is used to maintain privileges for both users and classes. To access this window, click the **Modify Permissions** button on the Users window or the **Objects** button on the Classes window.

The only difference in behavior is that the **User Classes** and **Copy Privileges** buttons are available only during user privilege maintenance.

From this window you can define which objects are given to a user or class, and what role the objects use when executed.



Field or Button	Description
Objects granted directly to the User or Class	The user ID or class identifier.
Object Name	The name of a Banner object which the user or class has permission to access.
Role Name	The role that applies when the user or class accesses the object.
Count	Presents the number of objects given the user or class by product character (the first character of the object name).
Wild Card	Gives a group of objects to the user or class (by using wild-card characters in the object name).
Insert	Creates an empty record so a new object can be entered.
Delete	Removes the object the cursor is positioned on from the user or class definition.
User Classes	Opens the User Class Enrollment window. **Note:** This button is available only if you are maintaining a user.

Field or Button	Description
Copy Privileges	This button launches the Copy Privileges popup so you can apply another user's full set of privileges to this user. **Note:** This button is available only if you are maintaining a user.
Close	Saves any changes made and closes this window.

Wild Card Additions or Deletions

You can add or delete Banner objects using an Oracle wild card symbol such as *G%*.

Button	Description
Insert	Adds all currently existing Banner objects that match the mask entered to the current user or class. For example, *G%* would cause all objects that begin with the letter *G* to be added.
Delete	Removes all currently given Banner objects that match the mask entered from the current user or class. For example, *G%* would cause all objects that begin with the letter *G* to be removed.
Close	Closes this window.

Tab-Level Security Settings

The User/Class Privilege window has a tab information block to allow you to set up tab-level security for forms which have been enabled for tab-level security. (For most objects, which are not enabled for tab-level security, this area will be blank.)

The **Copy All Tabs** button adds records for all of the selected form's tabs. After adding an object to a user or class, you can navigate to the tab block and click **Copy All Tabs** to add all tabs that do not currently exist for the user or class. You can also select individual tabs by using the list of values button.

Note

When establishing tab privileges for the APAIDEN form, it is recommended that the Household Members tab have the same tab privileges as the Address tab. ■

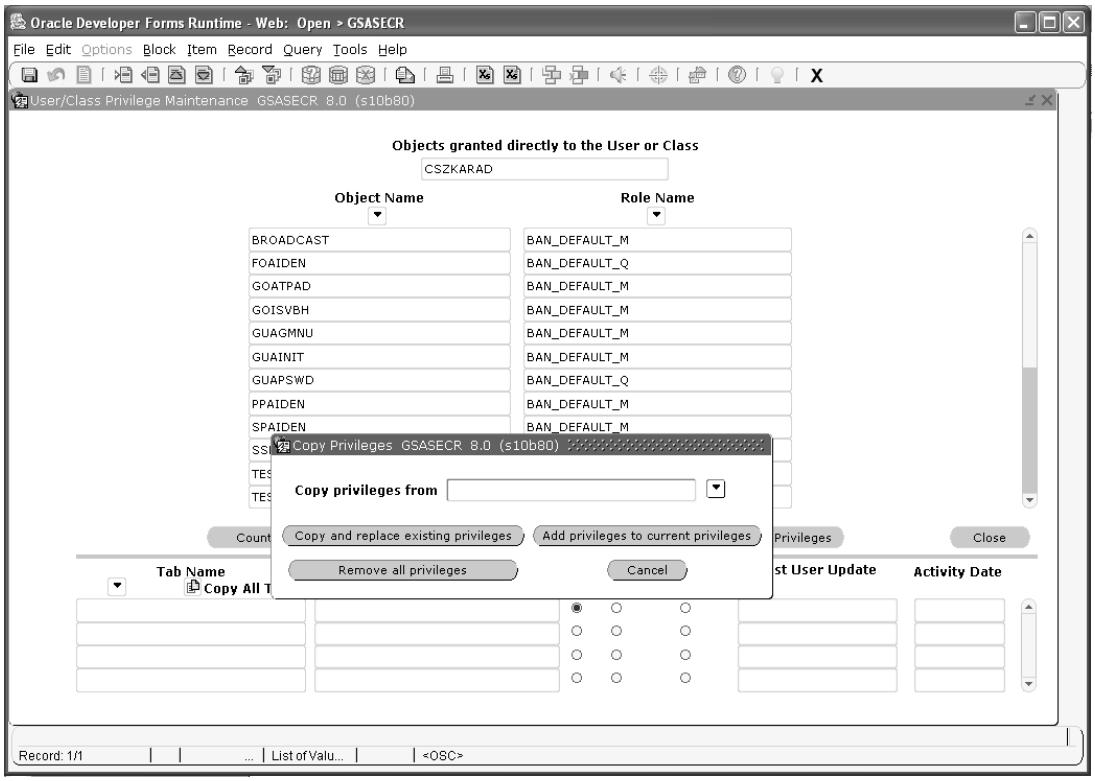
Note

If form and tab privileges have been established for a user both through a security class and through direct user permissions, the user-level security records take precedence over the class-level records. The tab records set

up at the class level are ignored if the user has a direct object grant for the form. If a particular tab is not defined at the user level, then the user's access to the tab will default to the user's overall form privilege.

Field	Description
Tab Name	The tab name as referenced internally by the Banner system (as defined by Oracle Forms Developer).
External Tab Description	The tab name that displays on the Banner form. **Note:** This value is descriptive information only. It does not control what is actually displayed on the tab.
Tab Access	Select one of the three radio buttons to specify the user/class access privileges for the selected tab. • *Full:* The user/class has full access to this tab. (Note that access for a tab may never exceed access defined at the form level. For example, if a user's access to the form is query-only, then full access for all the tabs will, in practice, be query-only access.) • *Query:* The user/class can view this tab but cannot change any data. • *Not Visible:* The user/class will not even be able to see this tab. Depending on the settings for this tab (as displayed in the tab information block of the Objects window), your choice of options here may be limited. For example, if a tab is defined on the Objects window with a *Full Access* restriction then you cannot set the tab to *Not Visible* for a user or class here.
Last User Update	User ID of the user who created or last updated the record.
Activity Date	Date record was created or last updated.

Copy Privileges



During user maintenance, the Copy Privileges popup lets you copy privileges (as defined in the GURUTAB, GURUOBJ, and GURUCLS tables) from one user to another.

You can access this popup by clicking the **Copy Privileges** button on the User/Class Privilege Maintenance window. This popup is only accessible during user privilege maintenance, and is not available during class privilege maintenance.

Here you can select another user and click one of the following buttons:

- **Copy and replace existing privileges**: Click this button to revoke all privileges from the user being edited and replace them with the privileges of the user selected in the popup.
- **Add privileges to current privileges**: Click this button to increase the privileges of the user being edited by adding all privileges of the user selected in the popup.
- **Remove all privileges**: Click this button to revoke all privileges from the user being edited. (If you select this button, it is not necessary to select a user in the popup.)

User Class Enrollment

This window shows all classes defined and indicates which ones the user is currently enrolled in.

Note

This screen is only accessible during user privilege maintenance, not during class privilege maintenance. ■

To enroll in a class, click the class. *Wait* is displayed while the user is being enrolled in a class. This can take a while because the form is verifying that the user has grants to all the roles needed to execute every object defined in the class.

To remove enrollment in a class, click the class.

Button	Description
Show Classes (Radio Group)	Toggle the selection criteria to show: • All classes • Only classes the user is enrolled in. • Only classes the user is not enrolled in.
Close	Close this window.

Viewing All of a User's Object Privileges

The Object Access by User View (GUVUACC) is the most convenient way to see a user's complete set of object privileges.

Because a user can have direct object privileges as well as privileges obtained through membership in security classes and security groups, there is no quick way to see all of a user's privileges in GSASECR.

For for information on GUVUACC, see "Reviewing Users' Object Access" on page 2-37.

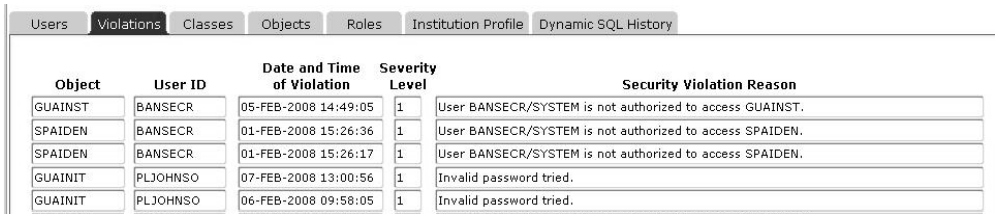
Violations

The Violations window queries the security log created by BANSECR's stored procedures. The log records security-related events, including hack attempts and Oracle errors. Records are sorted by three levels of severity; within each level they are listed chronologically.

Level 1 events are considered most important. For example, if an object fails to pass the decryption test, that could be a sign of a hack attempt, and a level 1 message is triggered.

When a user attempts to run a form the user is not authorized to run, a level 3 event is recorded.

This log file should be reviewed on a periodic basis. Also, the records in the log should be deleted periodically to prevent the log from exceeding its maximum capacity.



Button	Description
Delete All	Deletes all records from the security log table. This should be done periodically, in order to manage the size of the table.

Following is a partial list of the messages that can be written to the Security Violation Log:

Severity level	Message	Description
1	*No parameters passed*	The `G$_VERIFY_PASSWORD1_PRD` procedure was not passed the correct number of parameters.
1	*No password found on GUBROLE*	An encrypted password for the role being used could not be found on the GUBROLE table. Run **Encrypt All** from the profile maintenance tab to correct the problem.
1	*No records found on GUBIPRF*	The security profile record was not found. Create one using the profile maintenance tab.
1	*Invalid password tried*	An object that did not know the correct seed numbers tried to connect to this database.
2	*Invalid version of object being used*	The security profile record has turned on version checking and an object tried to connect to the database that was not the correct version. Either the form is old or the version number is wrong in the GURAOBJ table.

Severity level	Message	Description
2	*Oracle error message*	An unexpected Oracle error occurred in the security stored procedures.
3	*User _____ not authorized access to _____*	The user tried to access an unauthorized object from the **Go To** field of the Main Menu Form (GUAGMNU), from the direct access window (F5), or through any other method.

System User and OS User

Security messages that mention a specific user may list two user IDs: the system user (`user_id`) and the operating system user (`os_user_id`). Logging both IDs can help detect certain kinds of security violations.

The system user is the database connection user ID. For a typical INB connection, this will literally be *SYSTEM*. The `os_user_id` is, for example, the user's Unix or Windows login ID.

When two IDs appear in an error message, they will follow one of the following patterns:

- `user_id`/`os_user_id`
- `os_user_id` *as* `user_id`

Cross-Site Scripting Violations

When the log entry lists the object as *CROSS SITE SCRIPTING*, this indicates that a cross-site scripting attempt was detected on a Self-Service page. For example, a user may have entered some malicious code in a Self-Service text entry field.

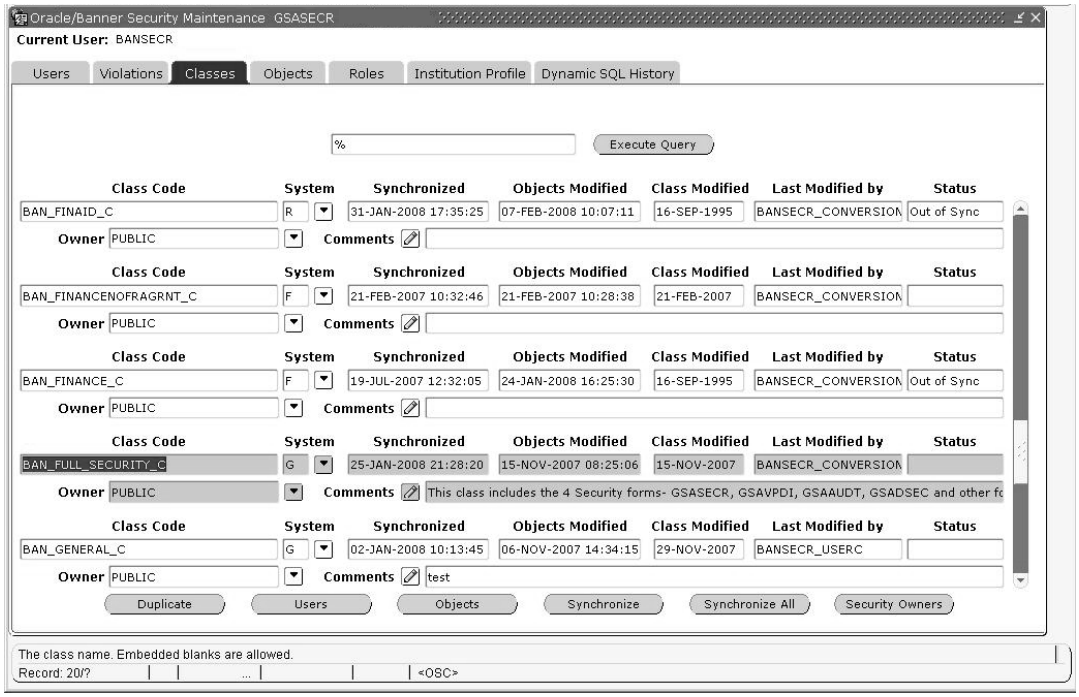
The **User ID** field indicates the IP address. The **Security Violation Reason** shows:

- the PIDM that the user logged on with (if a secure login) or an AIDM if the user was in the Apply for Admission page
- the ID associated with the PIDM or AIDM
- the Oracle User ID that the user is logged on with
- the web page and options (including the text the user entered that was identified as malicious scripting)

If the information exceeds the maximum 250 characters available in the **Security Violation Reason** field, the message is split into two security log entries. When this occurs, each entry's reason text begins with a sequence number, for example *(1)* or *(2)*.

Classes

The Classes window allows the creation, deletion, and maintenance of security classes. Classes are a group of object permissions that are common to more than one user at your site. Classes in Banner are similar to roles in Oracle. The use of classes simplifies security setup when several users need the same set of object grants.



Key Block

Field	Description
[Query field]	When the Class window is presented, all defined classes are displayed. To reduce the number of classes displayed, enter selection criteria.
Execute Query	Click this button to narrow the list of classes accorded to the selection criteria in the query field.

Main Window

After object permissions are changed for a class, the class must be synchronized for the changes to take effect. See "Synchronizing Classes" on page 2-20.

Field	Description
Class Code	The class's unique identifier.
System	One-letter code for the Banner product associated with the security class.
Synchronized	Date when the class was most recently synchronized.
Objects Modified	Date of the most recent change to the class's object permissions. **Note:** The class's objects are stored in the User/Class Privilege Table (GURUOBJ). When a new object is added to the class's privileges, the **Objects Modified** field will be updated to show the date that the object was added. But when an object is removed from the class's privileges, its row is actually deleted from GURUOBJ and, as a consequence, that object cannot be considered when calculating the **Objects Modified Date**. If the object privilege that was deleted happened to be the last-added object, the **Objects Modified Date** could actually revert to an earlier date: the date for the next-newest object added to GURUOBJ. The **Objects Modified Date** field will show the date of the latest change made to the class's privileges *only if* the latest changes included adding at least one object.
Class Modified	Date of the most recent change to the class record.
Last Modified by	The ID of the user that made the latest change to the class.
Status	Indicates *Out of Sync* if the class needs to be synchronized.
Owner	The user ID assigned as the class's owner for distributed security purposes. If the owner of the class is *PUBLIC*, all distributed users have privileges for maintaining and assigning this class.
Comments	Comments about the class and its security setup.
Duplicate	Copies a class to a new name. The new class name is generated automatically but may be changed after the duplication is complete. The new class is created with access to the same objects as the class that was copied.
Users	Open the Class/User Maintenance window to add or remove users for the class.
Objects	Open the User/Class Privilege Maintenance window that allows maintenance of what objects are contained in the class.

Field	Description
Synchronize	After changes are made to a class definition, this function ensures that all users enrolled in the class have the proper role grants to use every object in the class.
Synchronize All	Synchronizes all classes at once. **Note:** This function may take a while to complete.
Security Owners	Navigates to the Class Owners window of the Banner Distributed Security form (GSADSEC).

Creating a New Security Class

To create a new class:

1. Insert a new, blank record on the Classes window.
2. Specify the class code and other information.
3. Click the Objects button to define the class's object permissions on the User/Class Privilege Maintenance window.

To create a new class with similar privileges to an existing class:

1. Select the existing class.
2. Click the Duplicate button. A new class record is created.
3. Change the new class's code (which was generated automatically) to a meaningful code to identify the new class.
4. Click the Objects button to modify the class's object permissions on the User/Class Privilege Maintenance window, where they differ from the prior class's permissions.

Synchronizing Classes

Any changes made to a class are available to all users in the class only after the class is synchronized (by clicking the **Synchronize** button). If the class's **Status** is *Out of Sync*, there may be a mismatch between the class permissions and the permissions of users in the class.

To synchronize a class:

1. Select the class.
2. Click the **Synchronize** button.

Warning

You must click **Synchronize** after making changes to a class, including adding or removing objects, in order to apply the changes to all users in the class. The *Out of Sync* indicator will not always appear when you make changes, but you must synchronize every time.

The BAN_FULL_SECURITY_C Class

This security class (introduced in Release 8.0) is delivered with permissions for the Banner security objects and other forms needed for security administration. BAN_FULL_SECURITY_C includes permissions for the following objects:

- EXTENDED_QUERY
- GSAAUDT
- GSADSEC
- GSADSUM
- GSASECR
- GSAVPDI
- GUAABOT
- GUACALN
- GUAERRM
- GUAGMNU
- GUAHELP
- GUAINIT
- GUAPMNU
- GUAPSWD
- GUAUPRF
- GUIALTI
- GUIOBJS
- SOACOMP
- SOAIDEN
- SOQMENU

It is recommended that you assign this class to the BANSECR user ID.

You can copy BAN_FULL_SECURITY_C to create a separate class with some limitations for distributed users. These users, for example, should have a _Q role establishing query-only access to the new GSAAUDT form, so they will be unable to delete security audit records.

Note

With the introduction of this new security class, you should use it as the basis of permissions for BANSECR and other security administrators. It is recommended that you remove GSASECR and GSAVPDI permissions from classes such as BAN_GENERAL_C and BAN_ADMIN_C, and any other class which contains non-security users.

The BAN_SHOWALLMENU_C Class

Typically, users will only be able to view menu entries for those objects that they have permission to access. However, in some cases user may need to see the contents of the full menu even though they do not have access to all the objects. Typically, this would only be

users that are responsible for menu maintenance. In those cases, you must assign the user to the BAN_SHOWALLMENU_C class so that they will be able to view the full menu. Those users will see the *SECURITY menu, even though they will not have access to the security forms on *SECURITY menu.

Class/User Maintenance

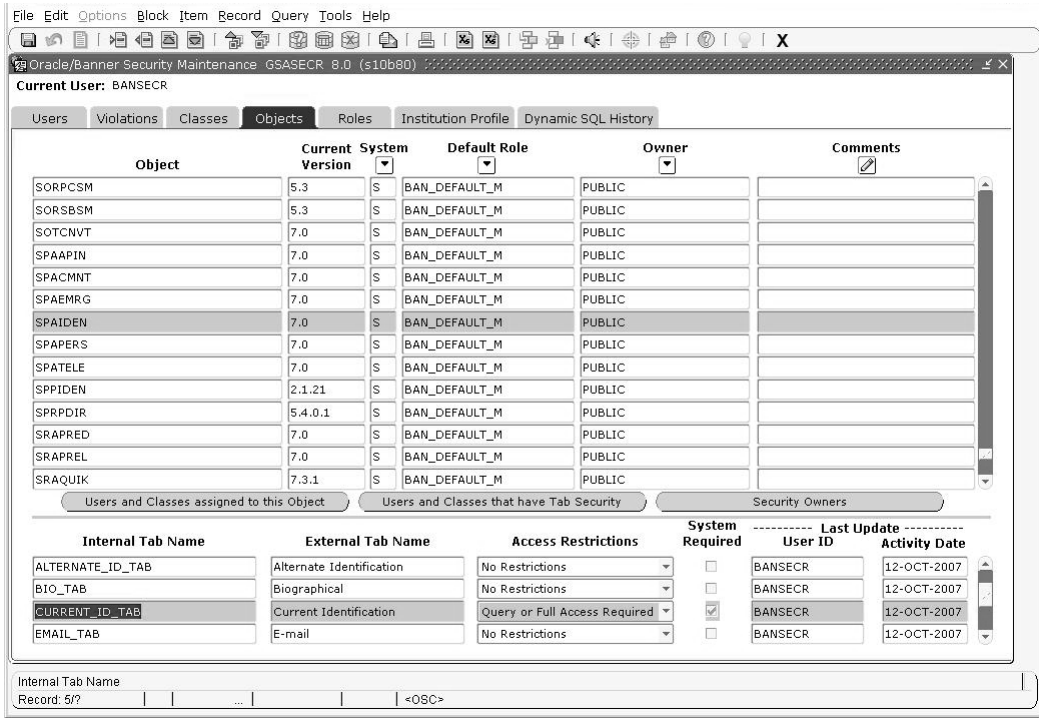
The Class/User Maintenance window is used to assign users to a class, or remove users from a class. This window is accessed by clicking the **Users** button at the bottom of the Class window. Information appears for the currently selected class in the Class window.

User/Class Privilege Maintenance

The User/Class Privilege Maintenance window is used to maintain privileges for both users and classes. This window is accessed by clicking the **Objects** button at the bottom of the Class window. See "User/Class Privilege Maintenance" on page 2-10.

Objects

The Objects window maintains a list of valid Banner object names and their default roles. Normal maintenance for the underlying table (GURAOBJ) is done during Banner upgrades. To define locally-developed Banner objects, you can insert new rows on this tab.



Field or Button	Description
Object	The name of a Banner object that is being defined.
Current Version	The version number indicated in the object. This value will be used if Version Checking has been enabled on the Institution Profile of GSASECR.
System	A one character identifier specifying which application the object belongs to (for example, *S* indicates Student).
Default Role	The default role that applies when the object is accessed.
Owner	The user ID assigned as the object owner for distributed security purposes.
Comments	Comments about the object.
Users and Classes assigned to this Object	List the class and users that have access to this object.
Users and Class that have Tab Security	List the class and users that have tab security defined for this object.
Security Owners	Displays the Object Tab on GSADSEC showing the owners and their privileges for this object

Tab-Level Security Settings

A block at the bottom of the Objects window displays tab information for the object currently selected in the list of objects.

Note

Since relatively few Banner objects are tabbed forms, this block will be blank for most objects. ■

For tabbed forms, the tab block will allow you to view the list of tabs and the options that are available (to you, the security administrator) for restricting access to each of the tabs. The **Access Restrictions** field for each tab can have one of three values:

- *Full Access Required* means that you cannot restrict access to this tab. Any user who is able to access this form will have full access to the tab.

Note

Tab access is always limited by a user's level of access to the form containing the tab. For example, if a user has query-only access to the

form, then a the user will have query-only access to the tab even if given *Full Access*. A tab may never provide more access than the form that contains it.

- *Query or Full Access Allowed* means that you can give some users query-only access to the tab while other users have full access, but you cannot hide the tab completely. Any user with access to this form will at least be able to query and view data in the tab.
- *No restrictions* means that you have a full range of options for restricting access. You can give some users full access, others query-only access, and for other users you can hide the tab, making it completely inaccessible for those users.

You cannot change the system required **Access Restrictions** setting for a tab. These settings are determined by SunGard based on analysis of each form.

Field	Description
Internal Tab Name	The tab name as referenced internally by the Banner system (as defined by Oracle Forms Developer).
External Tab Name	The tab name that displays on the Banner form. **Note:** This value is descriptive information only. It does not control what is actually displayed on the tab.
Access Restrictions	Assign access restrictions for the tab. No restrictions (N), Full Access Required (F), or Full access or Query required (Q).
System Required	Indicates whether the record is system required. If it is, then no changes are allowed to the record.
User ID	User ID of the User who created or last updated the record.
Activity Date	Date record was created or last updated.

Finding All Users with Access to an Object Privileges

The Object Access by User View (GUVUACC) is the most convenient way to see which users have access to a given object.

Because users can have direct object privileges as well as privileges obtained through membership in security classes and security groups, there is no quick way in GSASECR to see the full list of users who might have access to an object.

For for information n GUVUACC, see "Reviewing Users' Object Access" on page 2-37.

Roles

The Roles window provides a front end to manage Oracle roles used in Banner. It can create or delete a role, or change a role's permissions.

Main Window

Button	Description
Create New Role	Create a new Banner role with no initial permissions. Clicking the **Save** button will create the role. If a role to copy from was not specified, the created role will have no privileges. If a copy from role was specified, the table, view, and system privileges that were granted to the original role will be duplicated for the new role.
Delete Role	Delete the named role and remove all grants to and from it.
Role Privileges	Open the role privileges window to provide maintenance functions for the contents of the role.
Used by Objects	Show the users, classes, or groups that have access to a Banner object that uses this role. This is a summary of the information in the GURUOBJ table for this role.
Granted to	Shows the users that actually have grants to this role. It also indicates if it is their default role and if they have ADMIN permission on the role.

Role Naming Conventions

The roles that can be created, maintained, and used with Banner objects must adhere to strict naming conventions. First, roles must begin with either *BAN_* or *USR_*. (Only roles that begin with one of those two prefixes will be visible in the role maintenance tab.)

Roles that begin with the prefix *BAN_* will automatically be password-encrypted for use with a Banner object. Roles that begin with *USR_* are not password-encrypted and may be given to users as default roles. This default role would then define the tables and views the user can write ad hoc reports against.

The roles to be used with Banner objects must follow one of the following two patterns: `BAN_DEFAULT_`*something* or `BAN_`*objectname_something*.

This standard is enforced in the security maintenance form. It is also implemented in the LOV that displays the roles that can be used with a specific form. The *BAN_DEFAULT* roles can be used with all forms while the *BAN_objectname* roles can be only used with the specific object. Both *BAN_DEFAULT* and *BAN_objectname* roles will be given a random password when they are created.

Roles that begin with *USR_* will not be given a password and must be used as user default roles, not roles to be associated with Banner objects.

The Role Name Suffix: _Q or _M

Role names must end in *_Q* or *_M*. The suffix determines whether a user will access a form in query-only mode (if the applicable role ends in *_Q*) or is able to add, modify, and delete data (if the role ends in *_M*).

The *_Q* and *_M* convention must be followed in order for security to work correctly when a user accesses any API-enabled form. With the implementation of the Banner APIs, forms not being called in query mode could allow the user to perform other DML operations even if the user's role for that form only has *select* privileges. Banner security relies on the *_Q* convention for role names to know when to call a form in query mode.

Note

Some roles delivered with Banner, such as *BAN_DEFAULT_CONNECT* and *BAN_DEFAULT_WEBPRIVS*, are exceptions to the *_Q* and *_M* rule.

The BAN_DEFAULT_NO_ACCESS Role

This security role, delivered with Release 8.1, provides the ability to directly limit a user's access to an object. If a user is given a direct object grant with a role of `BAN_DEFAULT_NO_ACCESS` this overrides any other privileges that have been established for this user/object.

This new role can be thought of as an *exception* to a user's object permissions established through the user's memberships in security classes and security groups.

The advantage to using this role is that a user can now be enrolled in a class where the user has access to most of the objects and the user can be given a direct object grant for an object to exclude that object for the user. Previously, the way to accomplish this same result would have been to create a new class that contained the subset of objects the user is permitted to access.

The gchksecrole.sql Script

The `gchksecrole.sql` script can be run as needed to check for roles which do not adhere to Banner's naming conventions for roles. The script identifies role names which do not start with *BAN_* or *USR_* and end with *_M* or *_Q* so you can change the role names where needed.

The script also checks to make sure that no privileges have been assigned to the `BAN_DEFAULT_NO_ACCESS` role. Attaching privileges to this role would defeat its purpose of preventing access to an object.

Create New Role

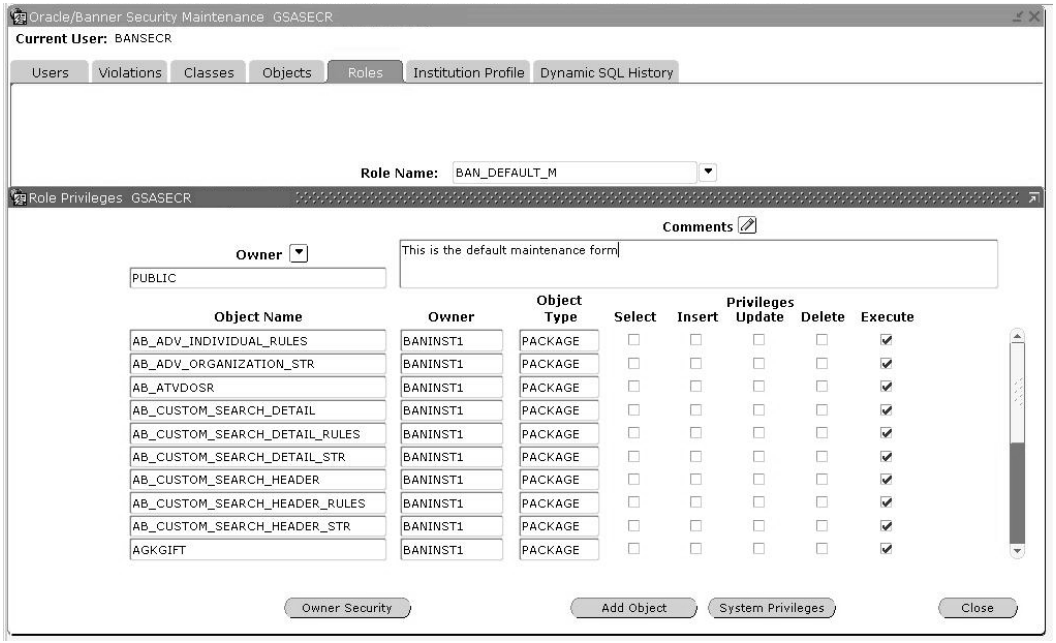
Button	Function	Description
Save	Save	This will create the role. If a role to copy from was not specified, the created role will have no privileges. If a copy from role was specified, table, view, and system privileges that were granted to the original role will be duplicated for the new role.

Role Privileges

You can change the types of grants that the underlying object can have from the role privileges screen. You may also grant new tables, views, sequences, packages, functions, and procedures to the role. A function is provided that will also allow system privileges to be granted or revoked from the role.

You may maintain comments related to the role as well as the owner of the role for distributed security maintenance purposes. If the owner of the role is *PUBLIC*, all distributed users who have privileges to maintain roles will have privileges for maintaining and assigning this role.

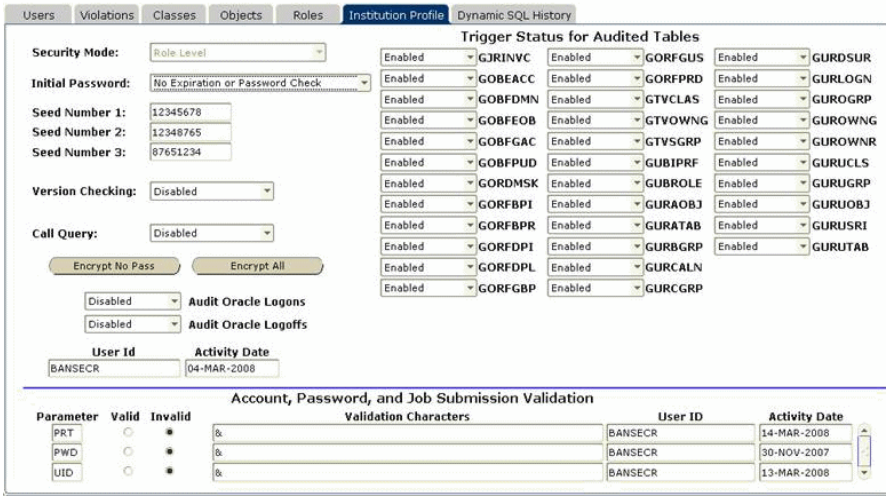
Existing permissions to objects in the role are presented to you as check boxes. A check box is provided for **Select**, **Insert**, **Update**, **Delete**, and **Execute**. Select the check box to grant one of these permissions to the role. Clear the check box to revoke the permission. If you clear all the check boxes, the object is removed from the role.



Button	Description
Owner Security	This button opens the Role Owners window on the GSADSEC form, where distributed ownership of the role can be maintained.
Add Object	This button opens a window that allows you to grant an object to the role that was not previously granted. When a new object is granted to the role, the minimum privilege possible for the type of object is granted. For tables, views, and sequences only, *select* will be granted. For procedures, functions, and packages, *execute* will be granted.
System Privileges	This button opens a window that allows you to grant system privileges to the role. All system-wide privileges defined in the Oracle data dictionary are available to be granted here. All the functions take place immediately; therefore, a rollback option is not available.
Close	Close this window and return to the main Role window.

Institution Profile

The Institution Profile window is used to set security options that apply across the whole institution, to manage security audit triggers, and to change the seed numbers.



Field	Description
Security Mode	A level of security. See "Security Modes" on page 2-33 for details on the options available.
Initial Password	Settings for password security. See "Initial Password" on page 2-34 for details on the options available.
Seed Number 1, 2, 3	Seed numbers for encryption. See "Seed Numbers" on page 2-35 for more information on what seed numbers do and how to change them.
Version Checking	Version checking checks the version number of Banner objects to prevent certain techniques that might be used to circumvent Banner security. As a general rule, version checking should be turned on.
Call Query	Call Query is a security feature that can be turned on or off. See "Call Query" on page 2-34.
Encrypt No Pass	This function will password-protect all Banner roles that are not currently password protected. Banner roles all start with the prefix *BAN_*.
Encrypt All	This function will password-protect all Banner roles even if they already have a password. Banner roles all start with the prefix *BAN_*. This function would be necessary if you change the seed numbers.
User ID	The ID of the user who made the last change to the Institution Profile record.
Activity Date	Date of the latest change to the Institution Profile record.

Audit Trigger Status for Security Tables

Use this group of fields to turn on or off the triggers for security auditing for specific tables. After you enable security auditing for a table, you can view the audit records in the Banner Security Table Audit form (GSAAUDT). See Chapter 4, "Security Auditing," for more information on GSAAUDT and security audit records.

Note

If a table is enabled for auditing, switching it to *Disabled* will not delete any existing audit records. Existing records will be retained and can still be viewed on GSAAUDT, but new audit records will not be created after auditing is disabled for a given table. ■

Field	Description
(Table name)	If *Enabled*, turns on the trigger for the specified table so that audit records will be recorded.
Audit Oracle Logons	If *Enabled*, turns on the `GT_LOGIN_AUDIT_ACCESS` trigger to record Oracle logons for Banner users. **Note:** INB logons will always be audited regardless of the value of this trigger.
Audit Oracle Logoffs	If *Enabled*, turns on the `GT_LOGOFF_AUDIT_ACCESS` trigger to record Oracle logoffs for Banner users. **Note:** INB logoffs will always be audited regardless of the value of this trigger.

Temporarily Suspending Security Audit Triggers

During upgrades or selective mass updates, you can use a set of SQL scripts to temporarily disable all triggers and then re-enable them after the process has been completed. These scripts (delivered with Release 8.0) save the current status of the entire set of triggers before disabling them, and then restore the full set of triggers to their saved state afterwards.

Script Name	Purpose
gsavtrig.sql	This script saves the current status of the audit triggers so that they can be restored.
grestrigs.sql	This script restores audit triggers to their saved state.
gursavt.sql	This script contains support SQL used when gsavtrig.sql is run.

Setting Triggers On or Off During Upgrade

A switch can be set during the Banner General installation process to have all security audit triggers initially enabled or disabled. See the *Banner General Upgrade Guide* for details.

Character Validation for User ID, Password, and Job Submission

Use this group of fields to define, for your institution, special characters that are not supported. You can maintain separate lists of unsupported characters for passwords *(PWD)*, user IDs *(UID)*, and print commands in Job Submission *(PRT)*.

If you prefer, you can instead explicitly list all *supported* characters, and all other characters will be unsupported.

The UID and PWD parameters are used during account creation of GSASECR and during password changes on GUAPSWD. The UID, PWD, and PRT parameters are all used during Job Submission to validate the data being processed.

Note

The validation function validates the parameters on a character-by-character basis. It does not validate strings of characters.

Tip

See "Creating a New User" on page 2-38 for additional limitations that affect user IDs.

Field	Description
Parameter	Type of validation parameter for which the listed characters apply. UID (user ID), PWD (password), or PRT (print command).
Valid/Invalid	Whether the characters listed are considered valid characters (only the explicitly listed characters are permitted, and all other characters are prohibited) or invalid characters (the listed characters are prohibited, and all other characters are permitted).
Validation Characters	The list of characters considered valid or invalid (depending on the Valid/Invalid selection) for this parameter. **Note:** If the list of validation characters is null, the parameter is ignored. For example, if the parameter is *PWD*, the condition is *Valid*, and the Validation Characters field is left empty, then all characters will be accepted for the password.
User ID	The ID of the user who made the last change to the Character Validation record.
Activity Date	Date of the latest change to the Character Validation record.

Job Submission Security

In Job Submission, a user could enter a special print command that would be passed along to the Operating System and run at the Operating System level. In addition, a user ID or a password could contain special characters that could also be passed to the Operating System through Job Submission.

Before Release 7.5, a hard-coded edit in GJAPCTL prevented the use of the *&* (ampersand) character in a print command, password, or user ID. There was also an edit in GTVPRNT that prevented using *&* in a print command. But there are other characters of

concern besides the ampersand, and a predefined, hard-coded solution did not provide enough flexibility.

Preventing Specific Characters

The Character Validation Table (GJRINVC) stores the list of supported or unsupported characters for each of three parameter types: user ID, password, and print command.

Note

You can set up only one validation row per parameter type. It is not possible, for example, to create a row listing User ID valid characters and another row listing User ID invalid characters. ■

To establish a list of unsupported characters:

1. Select the record for the parameter type you wish to edit.
2. Click the **Invalid** radio button.
3. Paste or type the unsupported (invalid) characters in the **Validation Characters** field.

Users will no longer be able to use any of the characters you listed for the parameter type that you selected.

The example below would prevent several special characters from being used for print command parameters.

Parameter type	Valid or Invalid	Characters
PRT	*Invalid*	!&#*

To establish a list of supported characters:

1. Select the record for the parameter type you wish to edit.
2. Click the **Valid** radio button.
3. Paste or type the supported (valid) characters in the **Validation Characters** field.

Users will now be able to use *only* the characters that you have listed for the parameter type that you selected.

The following example would allow only letters and digits in password parameters in Job Submission.

Parameter type	Valid or Invalid	Characters
PWD	*Valid*	ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890

If You Don't Plan to Restrict Job Submission Characters

If you don't have a need to limit your users' choice of characters in Job Submission for user ID, password or special print command, you do not need to take any action. The seed data records for GJRINVC will permit your users to use any character in their job submission parameters, as before Release 7.5.

Warning

You should take care not to delete the seed data records in the GJRINVC table, even if you don't plan to use this feature. Those records must be in place for Job Submission to work correctly. ■

Security Modes

The system provides the following three levels of security:

1. *Oracle grants only* – In this mode the security is turned off and end-users can execute any object. This mode depends on the user having sufficient privileges to all the required database objects.

Note

As of Banner 8.0, *Oracle grants only* security is no longer supported. Role-level security is the only supported mode. ■

2. *Process-level security* – This method is functionally equivalent to the old forms-level security. It guarantees that the user only runs the objects that you grant access to, but it depends on the user having sufficient privileges to all the required database objects.

Note

As of Banner 8.0, process-level security mode is no longer supported. Role level security is the only supported mode. ■

3. *Role-level security* – This is the highest level of security. This method not only guarantees that the user runs the objects that you grant them access to, but the object activates a role that allows the object to run without requiring the user to have direct permissions to all the database objects.

   The role-level security provides security in two distinct forms. First, it prevents an end-user from taking permissions, given to them to run Banner, into a third party tool that can access an Oracle database. Second, it provides object authentication,

preventing an end-user from creating his own forms by the same name as Banner forms. Because of the availability of third party tools that can access Oracle, the role level security mode is the only mode that is adequate in a client/server environment.

Initial Password

There are four options available to control the status of an account upon initial creation.

FORCE PASSWORD EXPIRATION	The password will be pre-expired and the creator will not be able to override this on GSASECR. If this option is not selected, the creator would still be able to manually select this option on GSASECR.
FORCE PASSWORD CHECK IN PROFILE	This option will ensure that the initial password given to an account will have to pass the password validation rules established in the Profile. Password changes will always have to pass the profile password validation regardless of this option.
FORCE PASSWORD CHECK AND EXPIRATION	This option enforces both password expiration and initial password verification by the profile.
NO EXPIRATION OR PASSWORD CHECK	This option does not automatically force expiration, nor does it enforce the password verification rules established in the profile.

Call Query

Call Query is the name of a feature that provides an optional, additional layer of form security. When the **Call Query** field is set to *Enabled*, any form that is called through a query role (a role with a name that ends with *_Q*) will be called in query mode. In query mode, a user can see the form's data, but cannot make any changes to the data.

When Call Query is enabled, query mode is "sticky." In other words, when a user in query mode navigates from one form to another form, the second form will also be in query mode regardless of the user's permissions for the second form. (This is native Oracle behavior, inherent in Oracle's implementation of Call Query.)

Following is an example of this behavior.

A user calls form A. The user's role for that form ends in *_Q*, so the form is called in query mode. The user then navigates directly to form B. The user's role for form B ends in *_M*, which would normally give the user the ability to edit data in that form. Nonetheless, form B appears in query mode, which it inherited from form A. The user cannot edit any data in form B.

The solution, in this example, is for the user to return to the menu and navigate to form B directly from the menu. Then the *_M* role will apply, and the user will be able to edit data in form B.

When **Call Query** is set to *Disabled*, a user's access to each form is determined directly by that user's permissions as set up through GSASECR. In the example above, the user will see form A in query mode, but *will* be able to edit data in form B, even if the user navigates directly from form A to form B. (Leaving **Call Query** blank has the same effect as setting it to *Disabled*.)

Seed Numbers

The role level security system is based on encryption that uses three seed numbers. Each Banner object uses these seed numbers to activate its role and gain access to the database.

The three seed numbers are stored in the following places:

1. The GUBIPRF table owned by BANSECR. These values are used by the database security packages.

2. A COBOL program called guasetr.pco. The object code for this program must be available when compiling COBOL programs.

3. A C header file called guassed.h. This header file must be available when compiling PRO*C programs.

4. The `G$_VERIFY_ACCESS` trigger in the property class `G$_FORM_CLASS` of the GOQOLIB reference library. This library is *not* loaded to the database. The source copy of this library must be available when generating forms. There is no object version of the Banner reference libraries; the referenced code becomes part of the generated FMX.

   The seed numbers should be stored in `G$_FORM_CLASS`, which is then propagated to the other property classes (`G$_APPL_FORM_CLASS`, `G$_INQ_FORM_CLASS`, `G$_VAL_FORM_CLASS`). These other property classes inherit the triggers from `G$_FORM_CLASS` and, therefore, do not need to be modified.

5. The bannerid.jar file, which handles secure access for Oracle*Reports. See Chapter 1, "Configuring INB," in the *Middle Tier Implementation Guide*. for instructions on changing the seed numbers and repackaging this file.

Note

Since Release 7.1, it is no longer necessary to generate a checksum for Oracle*Reports. The Checksum Generator Program (gurchks.exe) is no longer used. ■

6. The GODDTOP.DLL file. This file handles secure Banner access for Desktop Tools. See the *Desktop Tools* chapter of the *Banner General Technical Reference Manual* for

instructions on changing the seed numbers, recompiling the file, and updating Desktop Tools client machines.

7. The BatchSecurity.java file. Edit `BANNER_HOME/general/java/BatchSecurity.java`. Change these lines

```
static final long SECRET_SEED1 = 12345678L;
static final long SECRET_SEED3 = 87651234L;
```

Save changes. Recompile gurjbif.jar. (Make sure that the Java environment is set correctly).

OS	Command to recompile `gurjbif.jar`
UNIX	`cd $BANNER_HOME/general/misc` `gupdjar.shl`
VMS	`set def BAN_HOME:[general.com]` `@gupdjar.com`
Windows	`cd %BANNER_HOME%\general\misc` `perl gupdjar.pl`

Changing Seed Numbers

When changing seed numbers, the numbers must be changed in all the places where they are maintained. Then all SQL*FORMS must be regenerated, and all C and COBOL programs must be recompiled.

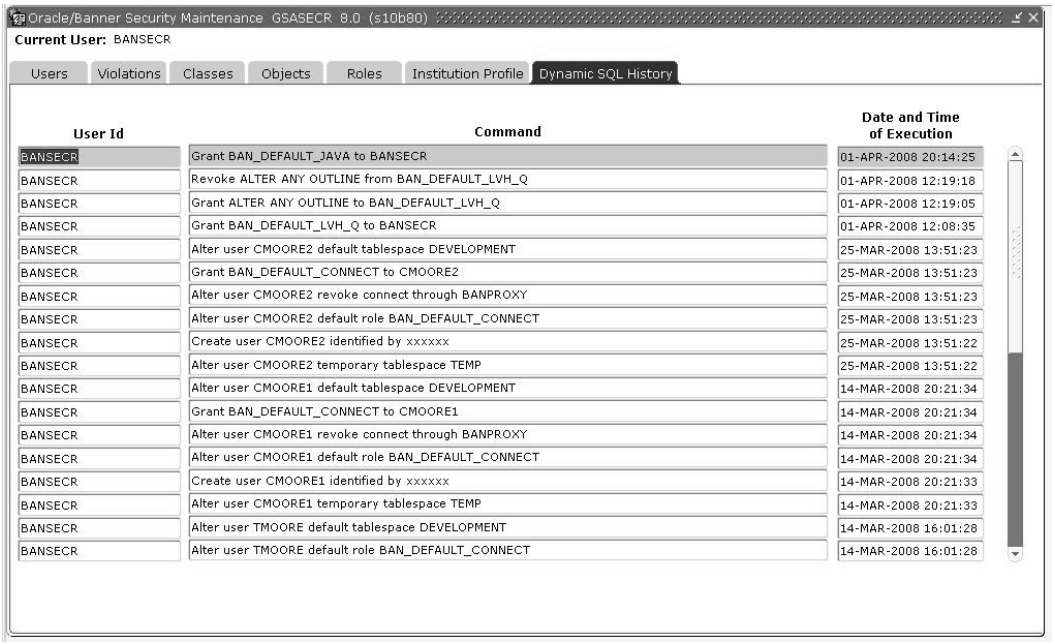
After the compilations are completed, the COBOL source and object code, C header file, and GOQOLIB reference library (GOQOLIB.fmb) must be hidden from other users. These steps will keep the seed numbers secure so that rogue Banner objects cannot be created.

If you change the seed numbers, you must also click the **Encrypt All** button in the Profile tab to redo the encryption of all *BAN_* role passwords.

Dynamic SQL History

This window shows the dynamic SQL statements generated in GSASECR and GSAAUDT and stored in the GURSQLL table. These statements include *ALTER, CREATE, DROP, GRANT,* and *DELETE* SQL statements on the security tables.

The GSAAUDT form, added in Release 8.0, displays an extensive record of security activity. See "Banner Security Table Audits (GSAAUDT)" on page 4-7.



Managing User Accounts

Reviewing Users' Object Access

Object Access by User View (GUVUACC)

This view, delivered in Release 8.1, lets you report on user access for each Banner object.

Because a user can have directly granted object access along with access derived through security classes and security groups, it can be difficult to quickly determine through the GSASECR form exactly what access a given user has for a given object. This view gathers together all forms of access.

In some cases, you might see two or more rows for the same user-object combination. This can happen, for example, when a user has been granted direct access to a form and also has

access to the same form through membership in a security class. In cases like these, the **Rank** column helps you determine which form of access is applied. The entry with the lower rank number takes priority. Rank *1* is the highest priority.

Column	Description
guvuacc_type	How access was defined: via a Class (Class), via a Direct grant (Direct), via a class defined in a group (Group Class), or via a direct grant to a group (Group Direct).
guvuacc_user	The Oracle ID for which access is being defined.
guvuacc_object	The Banner object for which access is being defined.
guvuacc_role	The default role of the object being accessed.
guvuacc_class	The Banner Class where this object access was defined. If access was via a direct grant, this will be NULL.
guvuacc_group	The Security Group where access was defined. If access was defined either directly or via a class this will be NULL.
guvuacc_rank	The order in which the roles will be used. **Note:** In practice, only one role, the one with the lowest rank number, is applied for each user-object combination

Object Access by User Table (GURUACC)

This table, provied in Release 8.1 via the optional `BANNER_HOME/general/plus/guruacc.sql` script, provides exactly the same information as the Object Access by User View (GUVUACC). The table is optionally provided in addition to the view in case you find that its performance is faster. Depending on the number of Banner users and the types of queries that you make, either the view or the table might provide better performance.

To create the optional GURUACC table please review the audit trail details in `BANNER_HOME/general/plus/guruacc.sql` and run the script as the BANSECR user.

```
sqlplus bansecr/password @guruacc.sql
```

Creating a New User

Oracle Options

GSASECR's User window does not allow specification of every possible Oracle option, only the most common ones. If additional information is required at your site you can alter

the user as a follow-up step, using other tools such as SQL*DBA or Oracle server manager.

Warning

Custom scripts for *creating* users are not recommended. ■

User IDs and Passwords

When creating new users, you must follow Oracle's naming conventions for nonquoted identifiers. (Oracle quoted identifiers are not supported for Banner user IDs.) In particular:

- User IDs must not be Oracle reserved words
- User IDs must not begin with numbers

You must also follow the institution's rules for special characters in user IDs and passwords as established on the Institution Profile Tab of GSASECR. See "Initial Password" on page 2-34.

When a new user is created, the characters in the user ID are validated against the *UID* parameter validation rule, and the password will be validated against the *PWD* parameter validation rule. See "Character Validation for User ID, Password, and Job Submission" on page 2-30.

Note

For more details on the limitations that apply to Oracle user IDs, see the section *Naming Objects and Parts* in Oracle's *SQL Language Reference Manual*. Also see SunGard Higher Education's FAQ 10718 for detailed guidance on Banner passwords and user IDs. ■

Case-Sensitive Passwords with Oracle Database 11g

If your institution migrates to Oracle Database 11g, you have the option of using case-sensitive passwords in Banner. This feature allows users to create stronger passwords that mix upper- and lowercase characters. Use of this feature is not required.

If you are migrating to Database 11g and want to take advantage of case-sensitive passwords in Banner, you must make the following settings:

- The initialization parameter `SEC_CASE_SENSITIVE_LOGIN` must be set to *TRUE*.
- You must create an Oracle*Forms environment variable, `FORMS_USERNAME_CASESENSITIVE` and set its value to *1* (the number one).

Note

Environment variable `FORMS_USERNAME_CASESENSITIVE` is available only when using Application Server version 10.1.2.2 or higher. ■

Default Role

You must specify a default role(s) for each account. The user must have, at a minimum, one default role that has the `CREATE SESSION` privilege. The permissions in the role specified here will be active when the user connects and they can be used in third party report writing tools.

It is possible for a user to have more than one default role. Since Release 7.0, it may be necessary for some users to have two default roles, `BAN_DEFAULT_CONNECT` (for example) and the `BAN_DEFAULT_WEBPRIVS` role, when using Value-Based Security through Self-Service. (The script `genrole.sql`, delivered with Release 7.0, created and applied the `BAN_DEFAULT_WEBPRIVS` role.)

Warning

With Oracle Database 11g, password encrypted roles (such as BAN_DEFAULT_CONNECT) can no longer be assigned as default roles. See "Default Roles in Oracle Database 11g" on page 1-5.

Note

The GSASECR form requires a default role. However, if an account is created outside of the GSASECR form it is possible to not define a default role. With no default role, a user will have *all* granted roles enabled when the user logs on to Oracle.

Copying Another User Account

When creating a new user account in Banner, you have the option to copy another user's account when creating a new user. This can save time when you are creating two or more user accounts with identical or similar privileges.

In GSASECR's Alter or Create an ORACLE User ID window, enter an existing Banner User ID in the **Copy User ID** field. The four check boxes below the Copy User ID field give you the option of copying the existing User ID's setup for PII, FGAC, masking, and business profiles.

The following information is copied from the selected user ID and applied to the newly created user ID:

- Oracle System privileges
- Oracle Private synonyms
- Oracle Granted roles
- Oracle Granted table/views privileges
- GURUCLS: class memberships
- GURUOBJ: object permissions
- GURUTAB: tab-level security permissions

- GURUGRP: group memberships
- GURLOGN: logon calendar information (GURLOGN_LOGIN_CALENDAR). A comment is added to the new calendar record, *Copied from user <source user> by <security administrator> on <date>*.
- FGAC VBS restrictions, if the **FGAC** check box is checked
- PII restrictions, if the **PII** check box is checked
- Masking restrictions, if the **Masking** check box is checked
- Business profile memberships, if the **Business Profiles** check box is checked

Deleting a User Account

You can delete a user account on the User window in GSASECR. Select the user and click the **User: Delete** button to delete the account.

Note

If you expect to restore a user's privileges in the future, there are ways to make an account unusable without deleting it. For example, you can lock the account by clicking the **Lock** button on the Alter a User Account window.

If a user account has data in either of the following tables, you will not be able to delete the account. Edit or remove any of the user's records in these tables before deleting the user account.

- NTRPROX: Banner Position Control Proxy Rules Table
- NTRPRXY: Banner Position Control Proxy Rules Table

When a user account is deleted, records associated with that account are deleted from the tables listed below.

Table Owner	Table	Comments
BANSECR	GUBROLE	This table stores the encrypted passwords for the Banner roles. This table is automatically maintained when passwords are generated or regenerated for the Banner roles. Records are deleted WHERE GUBROLE_ROLE = USR_GSASECR_XXXX for distributed security users only
BANSECR	GURLOGN	This table stores information related to logins to Oracle by users of Banner as defined in GURUCLS, for example, calendar, comments, authorization

Table Owner	Table	Comments
BANSECR	GUROWNR	This table defines the objects owned by distributed security users and their access for each object. Records are deleted `WHERE GUROWNR_OBJECT USR_GSASECR_XXX AND GUROWNR_OBJECT_TYPE = 'R';` for distributed security users only
BANSECR	GURUCLS	This table lists the Banner security classes a user is authorized to access.
BANSECR	GURUGRP	This table defines users belonging to a security group.
BANSECR	GURUOBJ	This table defines the type of access, by User ID, for each Banner object.
BANSECR	GURUTAB	This table defines all the forms and their related tabs that are in use for tab based security for a user or a class.
GENERAL	GOBEACC	Enterprise Oracle Access Table.
GENERAL	GURTPRF	This is a preference table that stores toolbar and menu information.
GENERAL	GURUPRF	Personal Preference Table.

If the user has an active GOBTPAC record, a popup message will display, *Disable Banner Self-Service Access for user <User ID>?* At that point you can decide whether to disable the user's Banner Self-Service access or retain it.

When a user account is deleted, data is not deleted from tables that manage FGAC, value-based security, PII, masking, and business profiles. This means that if you drop a user and then recreate the same user account, the previous restrictions established for the user in those tables will still be in place.

After all of the user's specific records are deleted from the tables listed above, the role and user are dropped.

`DROP ROLE` *`USR_GSASECR_XXX`* (if the user was a distributed security user)
`DROP USER` *`XXX`* `CASCADE`

3 Distributed Security

In distributed security, the responsibility for managing security is distributed, or delegated, to a number of security administrators. The most-trusted security administrator (in Banner, the BANSECR user) authorizes other administrators to handle specific security responsibilities, and gives those administrators the appropriate privileges to execute those responsibilities.

Distributed security has been available in Banner for several years via the BANSECR_*xxx* accounts, but the security functions were on an all-or-nothing basis. That is, each new BANSECR_*xxx* user had full access to modify all records when given permission to perform a specific function. For example, if a distributed user was allowed to perform class maintenance, there were no restrictions on which classes the user could maintain.

A new, more structured form of distributed security can be managed through the Banner Distributed Security form (GSADSEC), which was introduced in Release 8.0.

Owners and Privileges

Distributed security is organized around the concept of ownership. Each security role, class, and object has a single *owner*, a user with specific privileges. The owner can designate other users as *proxied owners* and define specific privileges for them.

Note

In addition to any specific ownership that has been established, BANSECR will always have full privileges for all security objects, classes, and roles.

Owners

Owners of roles, classes, and objects are defined on the GSASECR form.

- The owner of an object is defined on GSASECR's Objects window
- The owner of a role is defined on GSASECR's Role window
- The owner of a class is defined on GSASECR's Class window

The privileges of the owner are defined on the GSADSEC form, on the Object Owners, Class Owners, and Role Owners windows.

There is always one owner of each object, class, and role. This owner is initially given access to grant, revoke, delete, and modify, and also the ability to grant these same functions to additional security users.

Proxied Owners

The owner of an object, role, or class can designate one or more proxied owners and grant them specific permissions with respect to the object, role, or class. These permissions can include the ability to grant, revoke, delete, modify, or delegate permissions to other distributed users for the object, class or role.

Proxied owners are identified on the GSADSEC form, on the Object Owners, Class Owners, and Role Owners windows, and the proxied owners' privileges are also defined here.

A proxied owner can be thought of someone who assists the owner in the maintenance of the object, class, or role. You can identify as many proxied owners as needed, and each proxied owner can have different privileges suited to their specific responsibilities.

PUBLIC and Group Owners

An owner (or proxied owner) can be a Distributed Security Group, BANSECR, a distributed security user (such as BANSECR_*xxx*), or *PUBLIC*.

If the owner is a Distributed Security Group, then all users that are members of that group can act as owner of the object, class, or role. The use of *PUBLIC* means that all BANSECR_*xxx* accounts can act as owner. During the Banner 8.0 upgrade process, all objects, classes, and roles were updated to have an owner of *PUBLIC*.

Note

The *PUBLIC* owner here does *not* imply the Oracle definition of *PUBLIC*.

In cases where multiple owners are required, one or more proxied owners can be established.

Privileges

The are two type of privileges that can be assigned: object-related privileges and assignable privileges. Object-related privileges are functions that can be performed by a security owner on a object, class or role for a regular user. The functions include:

- *Grant:* allows the security user the ability to grant access to the object, class, or role
- *Revoke:* allows the security user the ability to revoke access to the object, class, or role

- *Delete:* allows the security user the ability to delete the object, class, or role
- *Modify:* allows the security user the ability to modify the object, class, or role

Assignable privileges define whether or not one security user can grant the specific function to another security user.

Note

These functions can only be assigned to security users, not to regular users.

Assignable privileges include:

- *Grant:* allows the security user the ability to give another security user the ability to grant access to regular users
- *Revoke:* allows the security user the ability to give another security user the ability to revoke access to regular users
- *Delete:* allows the security user the ability to give another security user the ability to delete the object, class, or role
- *Modify:* allows the security user the ability to give another security user the ability to modify the object, class, or role

Planning a Distributed Security Setup

The distributed security feature provided in the GSADSEC form is optional. By definition, the BANSECR account always has full control of all objects and the function that this account performs cannot be restricted. If objects, roles, and classes are owned and managed only by BANSECR, then you will not need to use the Object Owners, Class Owners, and Role Owners windows of GSADSEC. If objects, classes, and roles are owned by *PUBLIC*, distributed security can be used the same way it was used prior to the delivery of Banner 8.0.

Your distributed security setup can be as simple or as detailed as your institution's needs require. Planning distributed security involves identifying individuals who will have specific security administration responsibilities, and then setting up those users in the GSADSEC form so that each has just enough privileges to perform their designated responsibilities.

For a simple example, suppose you have security classes that correspond to job functions in your institution. A person who manages personnel transitions for a specific job function could be made a proxied owner of the corresponding security class, with privileges to add and remove users from the class as needed.

Tab-Level Security for GSADSEC

Tab Security can be enabled for the GSADSEC form. This new form can be made available to Distributed Security Users so that they can establish privileges and assignable privileges for users for classes, objects, and roles.

In order to give a distributed user access to the GSADSEC form, the user must either have direct access to the form, be enrolled in a class that includes the form, or be included in a security group that includes access to the GSADSEC form

In order to limit the use of functionality on this form for distributed users, tab security can be applied. The tab security records for this form are listed below. Use GSASECR to establish tab security for the user and/or class that has access to the GSADSEC form.

Internal Tab Name	External Tab Name	Access Restrictions	System Required
GROUP_TAB	Group Details	F	Y
GTVCALN_TAB	Calendars	Q	Y
GTVCLAS_TAB	Class Owners	Q	Y
GTVOWNG_TAB	Distributed Groups	Q	Y
GTVSGRP_TAB	Security Groups	Q	Y
GTVOWNG_OWNER_TAB	Distributed Group Owners	Q	Y
GUBROLE_TAB	Role Owners	Q	Y
GURAOBJ_TAB	Object Owners	Q	Y
GURCGRP_TAB	Classes	Q	Y
GUROGRP_TAB	Objects	Q	Y
GUROWNR_SECURITY_TAB	Owners	Q	Y
GURUGRP_TAB	Users	Q	Y

Distributed Security User Maintenance (GSADSUM)

This form, introduced with Release 8.1, lets you create a new distributed security user.

Prior to Release 8.1, you could create a new distributed security user only by running a series of scripts. For more information, see "Distributed Security Scripts" on page 3-22.

The GSADSUM form has two windows. The Distributed User Maintenance window lets you create and modify distributed security user accounts. The Distributed User Grants window lets you manage the rules that apply to creating distributed security user accounts.

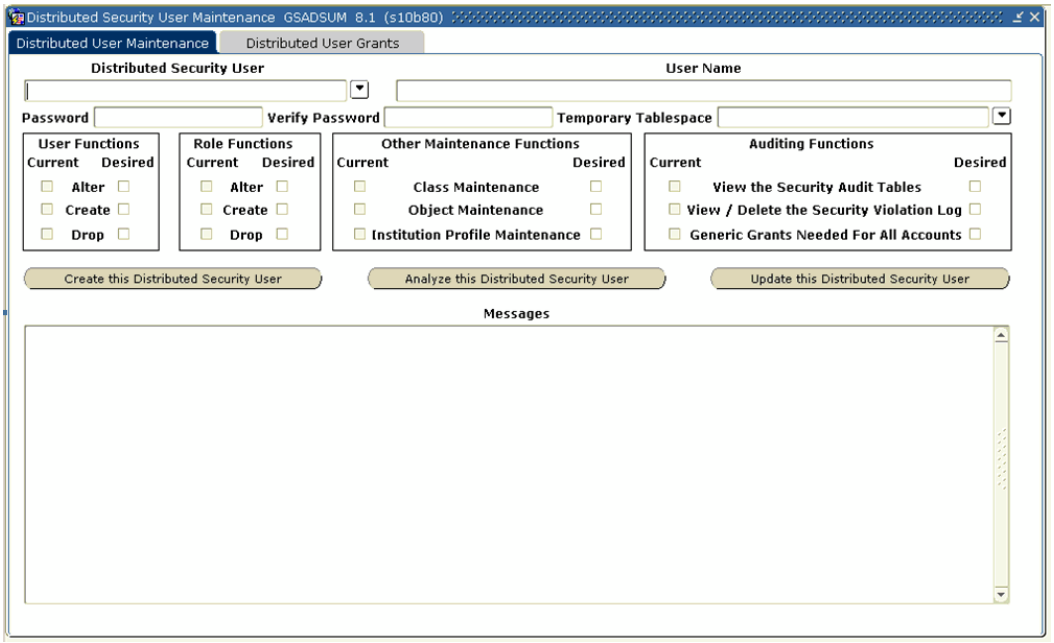
Distributed User Maintenance

Use this window to create and modify distributed security user accounts.

After you create a distributed security user here, you must go to the GSASECR form to establish individual user identity and Banner object access.

Note

You cannot maintain the *BANSECR* user account on this form. BANSECR is the basic system-required security account, defined with full access to Banner Security functions. Its permissions cannot be restricted.



Field	Description
Distributed Security User	The User ID. Must be in the format *BANSECR_xxxxxxxxxx*, where *xxxxxxxxxx* can be any combination of characters identified as valid User ID *(UID)* characters on the GSASECR institution profile tab.
User Name	If the user ID exists, the name established on the GSASECR Banner Rules tab will be displayed in the name field. If this is a new user then *** *NEW USER* *** will be displayed.
Password	The password can be any combination of characters identified as valid password *(PWD)* characters on the GSASECR institution profile tab. Required for new users (or when re-creating an existing user)
Verify Password	When you enter a new password, re-enter the same password here to verify it.
Temporary Tablespace	Must be a valid temporary table spaces identified to Oracle. Required for new users (or when re-creating an existing user).
Current	These 12 check boxes identify the functions that the account can perform. These fields are display-only. When modifying an existing user account, you can click the **Analyze this Distributed Security User** button to populate these check boxes with the account's current data.
Desired	Check or uncheck these 12 check boxes to add or change functions for this account. **Note:** The *Generic Grants* check box is always checked and cannot be changed.
Create this Distributed Security User	Click this button to create a new distributed user account, after making settings in the fields above. **Note:** After you create the account here, you must navigate to the GSASECR form to associate an identity with the account and to establish Banner object permissions for the account.
Analyze this Distributed Security User	For an existing distributed user account, you can click this button to update the 12 **Current** check boxes with information on the functions that the account can perform.

Field	Description
Update this Distributed Security User	Click this button to update a distributed user account, after making changes in the fields above.
Messages	This text area will display Oracle messages that indicate progress in creating or updating the users. Any Oracle errors that occur will display here also.

Distributed User Grants

This window establishes the rules for each of the functions identified on the prior tab. These rules are stored on the Distributed Security Rules Table (GURDSUR) and are essentially the same rules that were delivered in the `gss*.sql` distributed security scripts.

Note

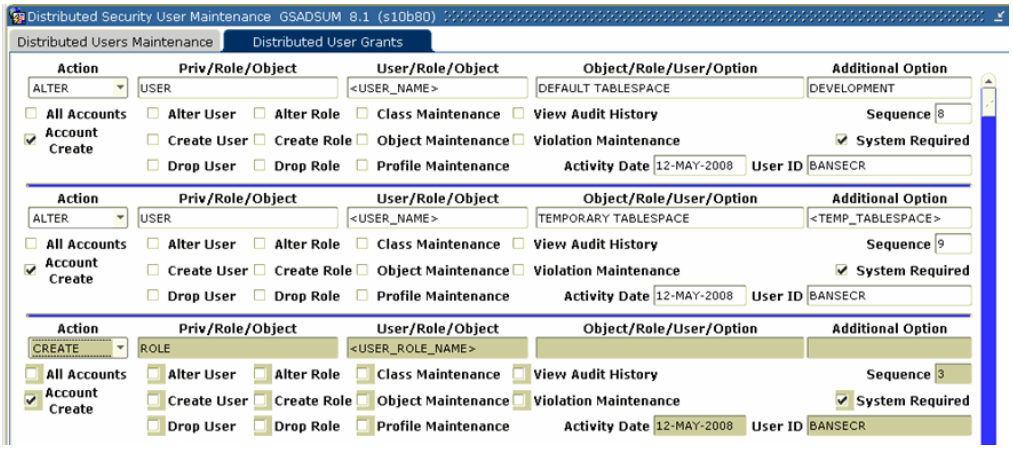
Access to the Oracle/Banner VPD Security Maintenance form (GSAVPDI) cannot be granted through this window. You must use the gssvpdi.sql script to grant access to GSAVPDI. See "Scripts to Create a Distributed Security User" on page 3-25.

In this window the SQL statements are broken into individual fields.

The rules delivered by SunGard are marked as System Required. These rules normally should not be modified.

Note

The options available here may not include all of the possibilities available through Oracle.



The are four dynamic parameters that can be used in this window's text fields:

- *<USER_NAME>* will be replaced by the user name (for example, *BANSECR_xxx*)

- <*USER_ROLE_NAME*> will be replaced by the role name (for example, *USR_GSASECR_BANSECR_xxx*)
- <*USER_PASSWORD*> will be replaced by the password from the form
- <*TEMP_TABLESPACE*> will be replaced by the tablespace from the form

Field	Description
Action	An action for the rule: `ALTER`, `CREATE`, `DROP`, or `GRANT`. Required. `ALTER` and `DROP` are only valid if the **Account Create** box has been checked.
Priv/Role/Object	The privilege, role, or object that the action is being performed on. Required.
User/Role/Object	The user, role, or object that is being modified by the action. Required.
Object/Role/User/ Option	Additional options that may be required with the statement. Optional.
Additional Option	Additional options that may be required with the statement. Optional.
All Accounts	Check this box to apply this rule to all distributed security accounts. Typically, if you check this box, no other function check boxes should be checked.
Account Create	Check this box to apply this rule to all newly created distributed security accounts. If you check this box, no other function check boxes can be checked, and you must enter a value in the **Sequence** field.
(Other functions)	Check one or more functions that apply to this rule. If **All Accounts** or **Account Create** are checked, these normally would be unchecked. These check boxes correspond to the function check boxes on the Distributed User Maintenance tab.
Sequence	A number to indicate the order that rules are applied when a new distributed user account is created. The lowest-numbered rules are applied first. Required when **Account Create** is checked.
System Required	SunGard-delivered records have been marked as *System Required*. As a general rule, these records should not be changed without SunGard input.

Field	Description
Activity Date	The date of the latest change to the rule.
User ID	The ID of the user that created or most recently changed the rule.

Establishing a New Distributed Security User

To set up a new distributed security user using the GSADSUM form:

1. Decide on a user ID. This will not be the same as the user's regular Banner user ID. For distributed security users, the ID must start with *BANSECR_*. For example, *BANSECR_ABC* is a valid ID for a distributed security user.

2. Use the Distributed User Maintenance window of GSADSUM to create the new distributed user account.

3. (Optional) On the GSASECR form, User tab, select the 'Banner Rules' option to assign the employee name to the new account and any approval information that may be desired.

4. (Optional) In GSADSEC's Distributed Groups window, add the distributed user to a distributed user group.

5. In GSASECR assign a class that gives the distributed user access to security forms (*BAN_FULL_SECURITY_C*, or another similar class).

   You can instead assign the user to a security group, and add *BAN_FULL_SECURITY_C* or a similar class to the security group.

6. On the GOAFPUD form, make the user exempt from PII security. Alternatively, you can build appropriate PII rules for the user. (You can skip this step if PII is not enabled at your institution.)

7. (Optional) Use GSADSEC to establish privileges and assignable privileges for the user for classes, objects, and roles.

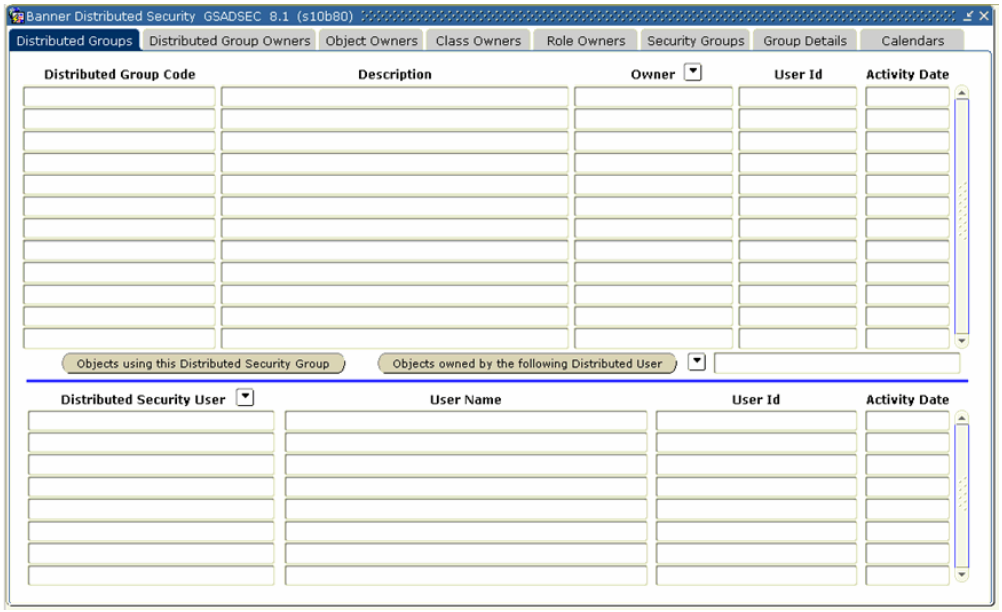
You can also create a distributed security user using scripts. For details, see page 3-24.

Banner Distributed Security (GSADSEC)

The GSADSEC form allows the most-trusted security administrators—the owners of security classes, roles, and objects—to set up specific permissions for other security administrators. You can also use this form to set up security groups and logon calendars.

Distributed Groups

Use this tab to create a group of security users that could include BANSECR, any distributed security user (BANSECR_*xxx*), or *PUBLIC*. The benefit of creating a Distributed Security Group is that maintenance can be done for a group instead of multiple users. After you create a group, this group can be identified as an owner or proxied owner of an object, role, or class.



Field	Description
Distributed Group Code	The unique code that identifies the distributed security group.
Description	A description of the distributed security group.
Owner	The user ID, distributed security group, or *PUBLIC* assigned as the owner of this distributed security group.
User ID	The ID of the user that created or most recently changed the distributed security group.
Activity Date	The date of the latest change.

Group Members

This section shows information about each member of the selected distributed security group.

Field	Description
Distributed Security User	The ID of a distributed security user that is enrolled in the selected distributed security group.
User Name	The name of the distributed security user enrolled in the group.
User ID	The ID of the user that added the distributed security user to the group.
Activity Date	The date the distributed security user was added to the group.

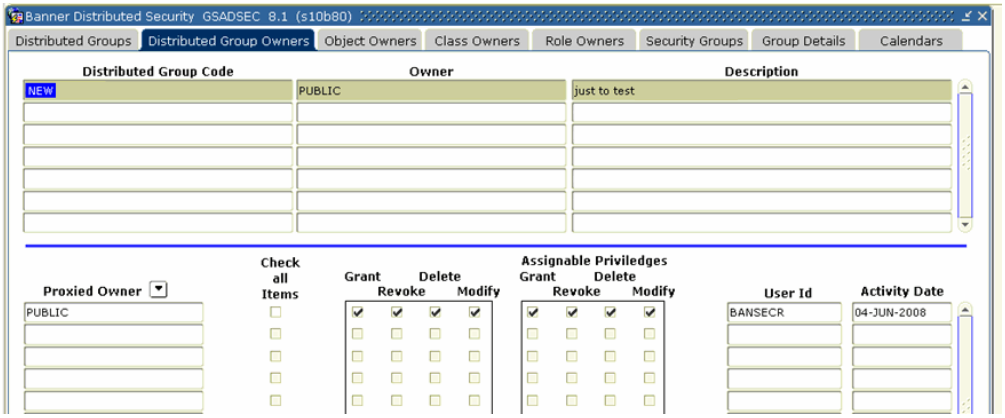
Distributed Group Owners

Use this window to designate proxied owners for distributed security groups, and to assign specific privileges to each proxied owner.

See "Owners and Privileges" on page 3-1 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

For the list of fields in this window, see "Object Owners" on page 3-11.

To change a distributed group's primary owner, you must edit the group in the Distributed Groups window. See "Distributed Groups" on page 3-10.

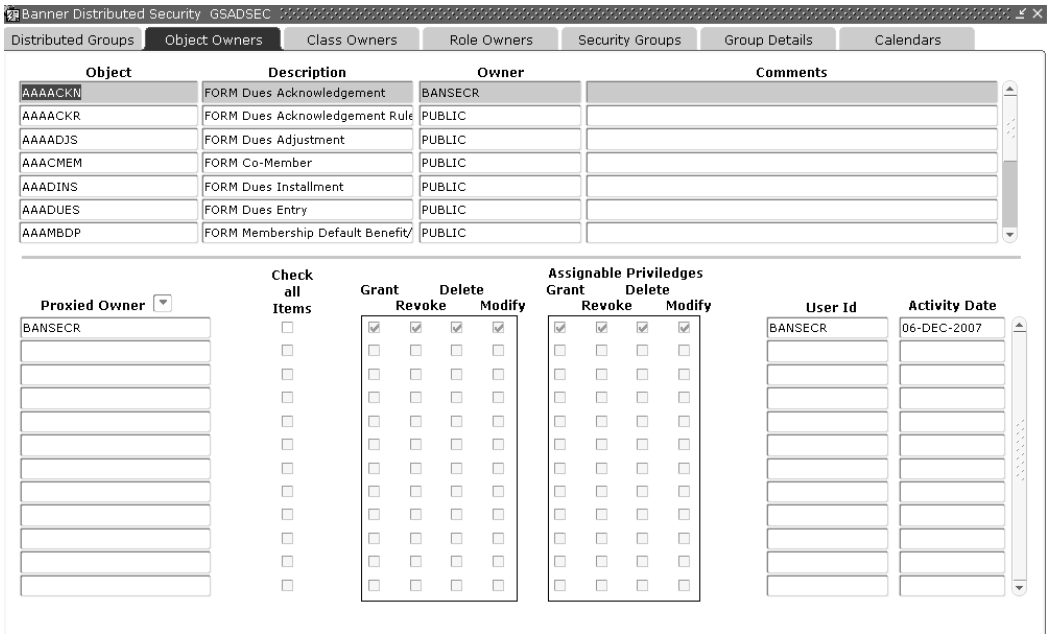


Object Owners

Use this window to designate proxied owners for Banner objects, and to assign specific privileges to each proxied owner.

See "Owners and Privileges" on page 3-1 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

To change an object's primary owner, you must edit the object in the GSASECR form. See "Objects" on page 2-22.



The fields in the Object Owners, Class Owners, Role Owners, and Distributed Group Owners windows are nearly identical, and the Security Group Owners tab is very similar. The list below applies to all of these windows.

Field	Description
Class Code/ Object Code/ Role Code Distributed Group Code	The unique code identifying the Banner object, security class, security role, or distributed group. **Note:** There is no corresponding code field on the Security Group Owners tab.
System	One-letter code for the Banner product associated with the class. **Note:** This field appears only on the Class Owners window.

Field	Description
Owner	The user ID, distributed security group, or *PUBLIC* assigned as the owner of the object, class, or role. This field is read-only in this form. To change the owner, you must edit the object, class, or role in GSASECR, or edit the group in the Distributed Groups or Security Groups window of GSADSEC. **Note:** See "Owners and Privileges" on page 3-1 for an explanation of what ownership means in distributed security.
Description	Comments associated with the object, class, role, or group. This field is read-only in this form. To change the comments, you must edit the object, class, or role in GSASECR, or edit the group in its tab in GSADSEC.

Proxied Owners and Privileges

This section shows the proxied owners for the selected object, class, or role, along with the specific privileges assigned to each proxied owner.

Field	Description
Proxied Owner	The user ID, distributed security group, or *PUBLIC* assigned to administer the security of the object, class, or role.
Check all items	Check this box to select all eight privilege check boxes. Clicking a second time will deselect all eight privilege check boxes. **Note:** If you do not have access to a specific function, then the check box will be protected and no update will be allowed.
Privileges	Click these check boxes to assign the corresponding privileges to the Proxied Owner: *Grant*, *Revoke*, *Delete*, and *Modify*. You can select none or all or any combination of the check boxes.
Assignable Privileges	Click these check boxes to give the Proxied Owner the ability to assign the corresponding privileges to other distributed security users: *Grant*, *Revoke*, *Delete*, and *Modify*. You can select none or all or any combination of the check boxes.
User ID	The ID of the user who created or last changed the Proxied Owner record.
Activity Date	The date of the latest change.

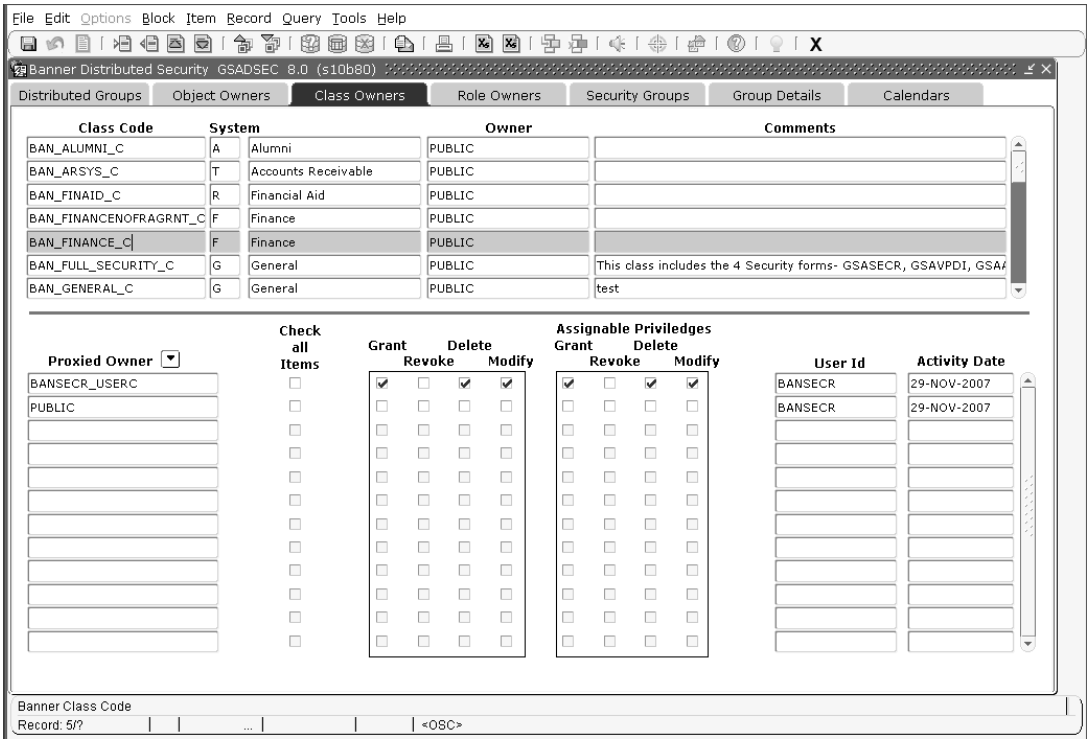
Class Owners

Use this window to designate proxied owners for security classes, and to assign specific privileges to each proxied owner.

See "Owners and Privileges" on page 3-1 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

For the list of fields in this window, see "Object Owners" on page 3-11.

To change a class's primary owner, you must edit the class in the GSASECR form. See "Classes" on page 2-18.



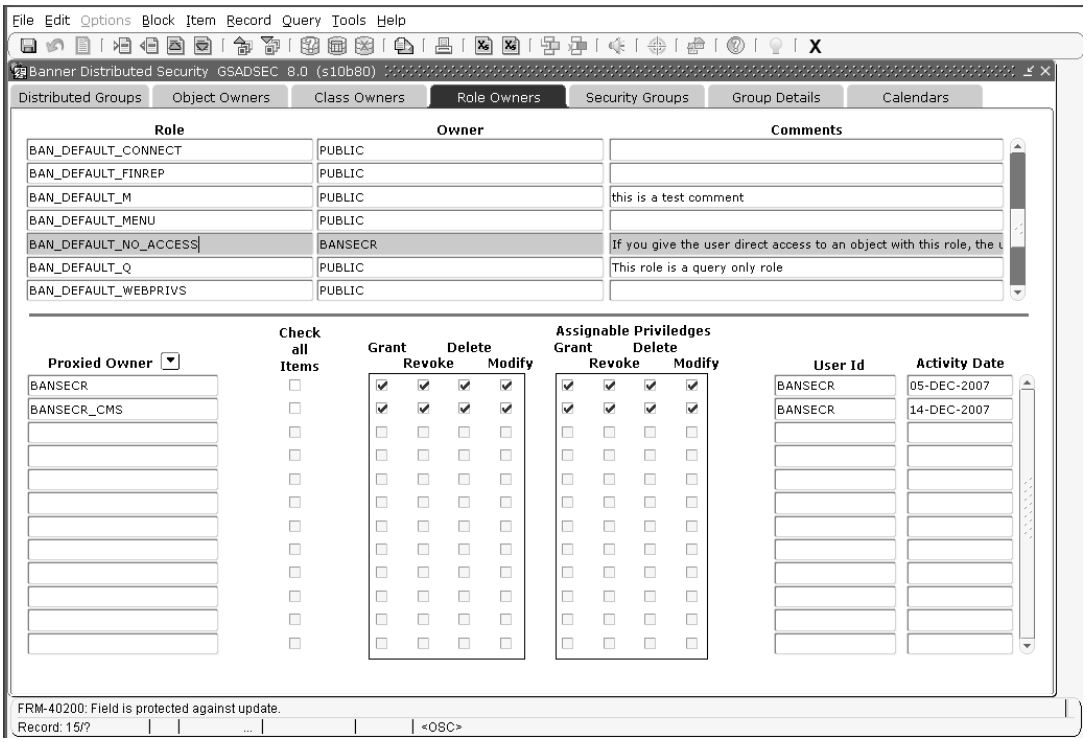
Role Owners

Use this window to designate proxied owners for security roles, and to assign specific privileges to each proxied owner.

See "Owners and Privileges" on page 3-1 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

For the list of fields in this window, see "Object Owners" on page 3-11.

To change a role's primary owner, you must edit the role in the GSASECR form. See "Roles" on page 2-25.

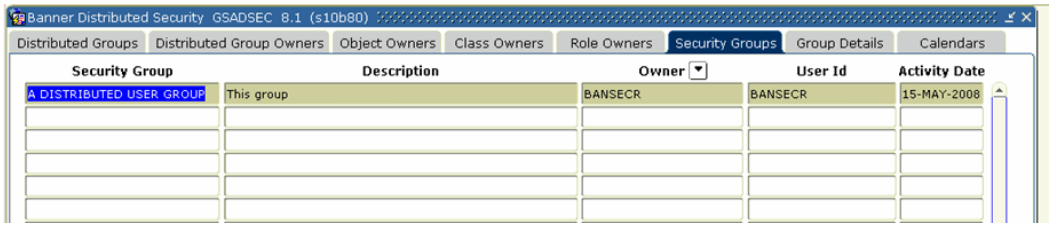


Security Groups

This window lets you define security groups. Security groups give you an additional way to organize user security. Each security group is identified by a group name and can contain security classes, objects, and users. Each security group can be assigned an owner—a distributed security user, distributed security group, or *PUBLIC*—with primary responsibility for the security group.

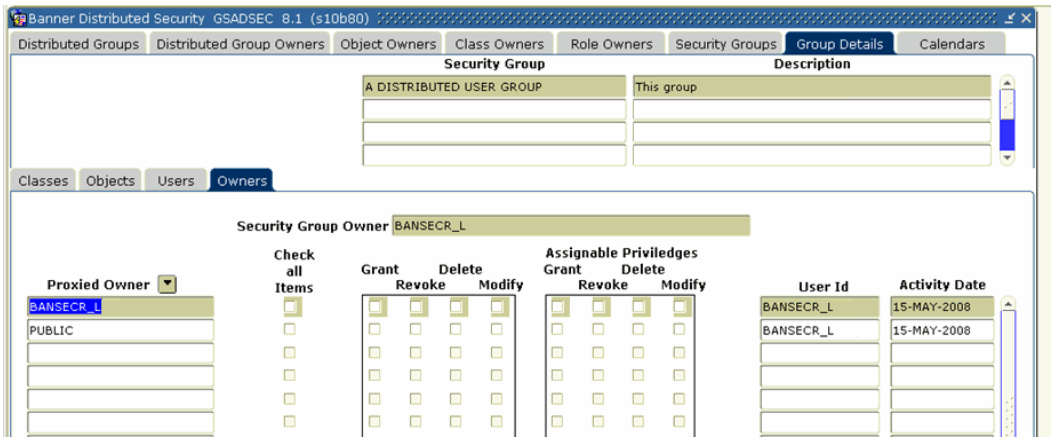
Security groups are optional. The user access to objects through classes and direct object grants will continue to work as before.

The Security Groups window is used to create the name and description of the security group. Additional security group settings are made in the Group Details window.



Group Details

The Group Details window lets you associate security classes, Banner objects, and user IDs with a security group. You can also assign proxied owners for the security group.



Classes

This tab lets you add security classes to the security group. Any object security and any tab-level security privileges that are included in the class will be passed to the users in the group.

Objects

This tab lets you add objects to the security group. Any objects privileges assigned to the group will be passed to the users in the group.

Note

Tab-level security is not available for objects directly assigned to a group. In order to provide tab-level security through a group, the tab security records must be built at the class level. You can then attach the class to the group, and the users will inherit the class's tab privileges through the group. Alternately, tab privileges can be assigned directly to an individual user in GSASECR.

Users

This tab lets you assign users that will gain access to the privileges provided by this group's objects and classes.

Note

Users may also be assigned to a security group through the **Banner Rules** button on GSASECR's User window.

Owners

This tab lets you designate proxied owners for security groups, and to assign specific privileges to each proxied owner.

See "Owners and Privileges" on page 3-1 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

For the list of fields in this tab, see "Object Owners" on page 3-11.

To change a group's primary owner, you must edit the group in the Security Groups window. See "Security Groups" on page 3-15.

Priority of Security Rules

With the addition of security groups, there are now four levels at which security can be applied:

- User direct object grants
- Users enrolled in classes that have object grants
- Groups that have object grants
- Groups that have classes that have object grants.

When security rules applied through different levels contradict each other, the highest priority level is given to user direct object grants. This is an overriding security level. If the user has a direct object grant this will establish the role and any tab security for the object for this user.

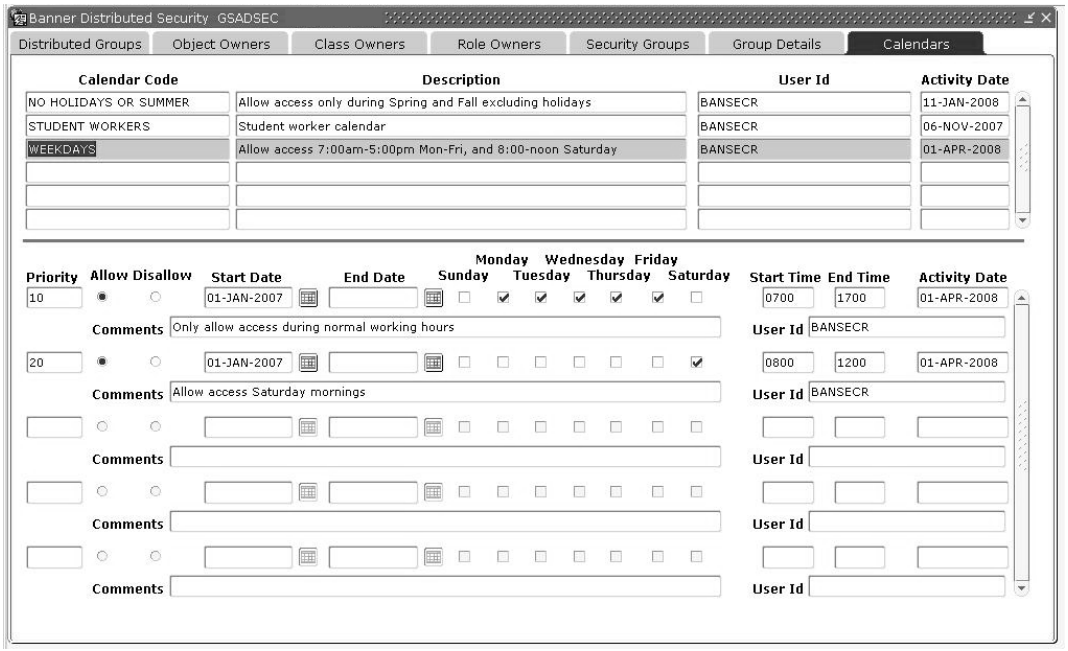
To determine the next highest priority (in the case where there is not direct user grant for an object), the class level and group level privileges are compared.

The order which they are applied is that any privilege with a maintenance role is applied first. If there are multiple records with a maintenance role, they are applied in alphabetical order. Then privileges with a query role are applied, and if there are multiple query roles they are applied in alphabetical order.

Calendars

This window lets you establish logon calendars that can be assigned to user IDs. A logon calendar locks out a user ID during specified days of the week and specified times of day. Use this feature when it is necessary to restrict the time frame of users' access to Banner.

A logon calendar can be assigned to a user in the Setup Logon Rules window, which is accessed through GSASECR's User window. See "Setup Logon Rules for a User" on page 2-8.



Field	Description
Calendar Code	A unique identifier for the calendar.
Description	A description of the calendar.
User ID	The ID of the user who created or last updated the calendar record.
Activity Date	The date of the last update to the calendar record.

Calendar Rules

This section shows the list of rules for the selected calendar. Each calendar will have one or more rules to indicate when login access is allowed and when it is not allowed.

Field	Description
Priority	A number to indicate the relative importance of a rule in the calendar record. *1* is the highest priority, while larger numbers are given lesser importance. Priority numbers do not have to be consecutive. **Tip** Use numbers that are multiples of 10 (10, 20, 30, and so on). This will make it easier to insert numbers in between, if it becomes necessary to add rules later. ■
Allow/Disallow	How to interpret this rule in the calendar record. If *Allow*, this row identifies times when the user is specifically permitted to log in. If *Disallow*, this rule identifies times when the user is specifically prevented from logging in. In cases where two rules conflict, the rule with the lower **Priority** number (the higher-priority rule) takes precedence.
Start Date	A beginning date when this calendar rule takes effect.
End Date	An ending date when this calendar rule ceases to be in effect. If blank, the rule is open-ended.
SUN MON TUE WED THU FRI SAT	Check each day of the week for which this rule applies.
Start Time	A time of day when this rule begins to be in force. Enter a number from 0000 (12:00 AM) to 2399 (11:59 PM). If Start Time and End Time are left blank, the rule is in effect around the clock.
End Time	A time of day when this rule ceases to be in effect. Enter a number from 0000 (12:00 AM) to 2399 (11:59 PM).
Activity Date	The date of the last update to this calendar rule.
Comments	Comments related to this calendar rule.
User ID	The ID of the user who created or last updated the calendar rule record.

Setting Up Logon Calendars

The use of the Logon Calendar is optional for each user. If a user is not assigned to a calendar, no logon restrictions will be applied to that user.

Note

Whether or not a logon calendar is assigned, it is possible to set a start date and end date for each user's INB access. See "Setup Logon Rules for a User" on page 2-8.

Each logon calendar can have a number of rules, which are interpreted in priority order. In cases where two rules contradict each other regarding a specific period of time, the rule with the lower priority number takes precedence and all other rules for that period of time are ignored. If no rule is matched, then access will be denied.

Warning

If a user is assigned to a calendar that has no rules defined, the user will not be permitted to log on to Banner.

A rule can indicate time according to three different levels of time

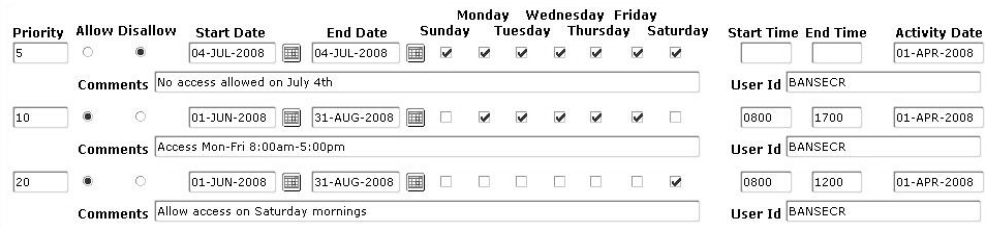
- a daily cycle—you can enforce a beginning and ending clock time
- a weekly cycle—you can select days of the week
- long-term—you can set a beginning date and (optional) ending date

Or you can use any combination of the three.

A rule can be positive (*Allow* this user to logon during this time period) or negative (*Disallow* this user during this time period).

One way to set up a calendar is to start with the broadest rules, assigning them large priority numbers. Then enter the exceptions to those rules, and assign them smaller priority numbers (smaller numbers indicate higher priority). If there are exceptions to the exceptions, add those rules last and give them the smallest priority numbers (highest priority).

The following example shows a simple logon calendar for a summer worker with regular Monday-Friday work hours. This worker also sometimes works a few hours on Saturday morning, and is locked out of the system on July 4.



Reviewing Distributed Security Permissions

Reviewing and Updating Grants with the gchkgrants.sql Script

The gchkgrants.sql script, delivered with Release 8.2, is a tool that helps you maintain distributed security accounts. You can run gchkgrants.sql after each Banner General upgrade to analyze the grants assigned to each distributed security account. The script will identify any missing grants and optionally update the grants for the accounts.

Distributed Security Object Ownership View (GUVOWNR)

This view, delivered with Release 8.1, lets you report on BANSECR distributed security users and the type of ownership that they have for classes, groups, objects, and roles.

The distributed user's ownership is defined as own of four types:

- Owner: The distributed user is the designated owner
- Proxy: The distributed user is a proxied owner
- Owner Group: The distributed user is a member of a distributed group that is the designated owner
- Proxy Group: The distributed user is a member of a distributed group that is a proxied owner

Column	Description
guvownr_object_type	The type of object for which ownership is defined. (C)lass, (G)Distributed Group, (O)bject, (R)ole, or (S)ecurity Group.
guvownr_object	The name of the object for which ownership is defined.
guvownr_owner	The distributed security owner of the object. Values are limited to BANSECR% accounts and PUBLIC.
guvownr_owner_type	The ownership will be defined as Owner or Proxy. If the ownership is granted via a group, then the ownership will be identified as Owner Group or Proxy Group.

Column	Description
guvownr_group	The security group that the user belongs to if the ownership is defined as Group Owner or Group Proxy. If the owner is not part of a group then this will be NULL.
guvownr_grant	Ability to grant or add this object to an end user / class / role / group.
guvownr_revoke	Ability to revoke or remove access to this object to an end user / class / role / group.
guvownr_delete	Ability to delete this object.
guvownr_modify	Ability to modify this object.
guvownr_grant_assign	Ability to assign the grant privilege of this object to another distributed user.
guvownr_grant_revoke	Ability to assign the revoke privilege of this object to another distributed user.
guvownr_grant_delete	Ability to assign the delete privilege of this object to another distributed user.
guvownr_grant_modify	Ability to assign the modify privilege of this object to another distributed user.

Distributed Security Scripts

Prior to Release 8.0, distributed security could be accomplished in Banner only by setting up privileges manually. A set of scripts was provided to help the database administrator manage distributed security accounts by granting the necessary privileges and grants that were needed to access the various BANSECR owned tables.

The Distributed Security User Maintenance form (GSADSUM) now provides an easier way to set up new distributed users, and the Distributed Security Form (GSADSEC) enables a finer level of control of objects that the distributed security users can modify. For more information, see:

- "Distributed Security User Maintenance (GSADSUM)" on page 3-5
- "Banner Distributed Security (GSADSEC)" on page 3-9

The SQL scripts are still available as an alternative method to establish grants and privileges for distributed security users.

Privileges

Banner role security is managed in the GSASECR form. The GSASECR form can be run only from an Oracle ID that begins with *BANSECR*. The BANSECR user owns all the security objects and is considered the master security maintenance account. This account is required to have DBA privilege; therefore, it can perform all security maintenance functions.

Logic in the GSASECR pre-form trigger checks to see if the user running the form is the master account named BANSECR. If it is being run from the BANSECR account, the trigger checks to see if BANSECR has been granted the DBA role. If it was granted, a `SET ROLE DBA` command is issued. If not, an error condition is raised and form execution stops. Additionally, if the user is BANSECR, a Dynamic SQL History tab is made visible and navigable. This tab shows the list of all commands executed by the various BANSECR users from within the form sorted in descending order by date and time.

If the form is being run from any other legitimate account (a BANSECR_*xxx* account), the form checks for the following specific privileges to determine if the user is to be allowed to perform security maintenance tasks. If the user does not have the required permission, the link or button to perform that function is grayed out so that it cannot be selected.

This Privilege . . .	Activates the Function . . .
`ALTER ANY ROLE` and `GRANT ANY ROLE`	Role Maintenance
`ALTER USER`	Alter User
`CREATE USER` and `ALTER USER`	Create User
`DELETE` on BANSECR.GURUCLS	Class Maintenance
`DROP ANY ROLE`	Delete Role
`DROP USER`	Delete User
`GRANT ANY PRIVILEGE`	System Privilege Maintenance for Roles
`GRANT ANY ROLE`	User Maintenance
`UPDATE` on BANSECR.GTVCLAS	Class Maintenance
`UPDATE` on BANSECR.GUBIPRF	Profile Maintenance
`UPDATE` on BANSECR.GURALOG	Review Security Violations
`UPDATE` on BANSECR.GURAOBJ	Object Maintenance
`UPDATE` on BANSECR.GURUCLS	User Maintenance

This Privilege . . .	Activates the Function . . .
UPDATE on BANSECR.GURUOBJ	Class Maintenance
UPDATE on BANSECR.GURUOBJ	User Maintenance

Establishing a New Distributed Security User

To set up a new distributed security user using the distributed security scripts:

1. Decide on a user ID. This will not be the same as the user's regular Banner user ID. For distributed security users, the ID must start with *BANSECR_*. For example, *BANSECR_ABC* is a valid ID for a distributed security user.

2. First, run the gssacct.sql script. This script will execute three other scripts: gsssysg.sql, gsspriv.sql, and gssbasg.sql. You will need to provide

   - the user ID
   - the role, which must be USR_GSASECR_*<user id>*. For example: *USR_GSASECR_BANSECR_ABC*
   - the passwords for BANSECR, BANINST1, SYS, SYSTEM, and the newly created account

   Note

   Running this script will drop the user, if it exists, before recreating the user.

   Refer to the gss*.sql scripts on the following pages for more details about each individual script.

3. Choose one or more additional scripts to run, from the list of distributed security scripts below, to provide the appropriate permissions to the user. based upon the function that this account will need to perform, for example, class, object, role, or user maintenance; profile maintenance; access to audit records. Do not grant more access than is actually required.

4. (Optional) On the GSASECR form, User tab, select the 'Banner Rules' option to assign the employee name to the new account and any approval information that may be desired.

5. (Optional) In GSADSEC's Distributed Groups window, add the distributed user to a distributed user group.

6. In GSASECR assign a class that gives the distributed user access to security forms (*BAN_FULL_SECURITY_C*, or another similar class).

You can instead assign the user to a security group, and add *BAN_FULL_SECURITY_C* or a similar class to the security group.

7. On the GOAFPUD form, make the user exempt from PII security. Alternatively, you can build appropriate PII rules for the user. (You can skip this step if PII is not enabled at your institution.)

8. (Optional) Use GSADSEC to establish privileges and assignable privileges for the user for classes, objects, and roles.

You can also create a new distributed security user using the GSADSUM form. For details, see page 3-9.

Scripts to Create a Distributed Security User

To help you create distributed security accounts, several scripts have been delivered in the GENERAL directory tree. The scripts will prompt for the password for any or all of these users: BANSECR, SYSTEM, SYS, BANINST1, and the BANSECR_*xxx* user ID.

The gssacct.sql script is the basic script that you will run to create every new distributed security user. It in turn calls the gsssysg.sql, gsspriv.sql and gssbasg.sql scripts.

Script	Description
gssacct.sql	Sample BANSECR_*xxx* account creation script. This script creates the distributed user, assigns the appropriate role, password and temporary tablespace. After this is complete, it executes the gsssysg.sql, gsspriv.sql and gssbasg.sql scripts.
gsssysg.sql	SYS grants required by all BANSECR_*xxx* accounts.
gsspriv.sql	Synonyms required for all BANSECR_*xxx* accounts.
gssbasg.sql	Minimal grants for a BANSECR_*xxx* account.

Each of the following optional scripts applies specific privileges to the newly created BANSECR_*xxx* account.

Script	Description
gssaudt.sql	Grants and privileges needed for BANSECR_*xxx* access to GSAAUDT.
gssclsm.sql	Privileges required to do class maintenance.
gssobjm.sql	Privileges required to do object maintenance.

Script	Description
gssprfm.sql	Privileges required to do profile maintenance. **Note:** Grant this function with care. The Institution Profile on GSASECR controls security for the entire Banner system. Access to this functionality should be given only to those who understand the consequences of changes to Institution Profile settings.
gssrolc.sql	Privileges required to create new roles.
gssrold.sql	Privileges required to drop roles.
gssrolm.sql	Privileges required to do role maintenance.
gsssels.sql	Sample row level security for security maintenance.
gsssysp.sql	Privileges required to grant system privileges to roles.
gssusra.sql	Privileges required to alter Oracle user IDs.
gssusrc.sql	Privileges required to create new Oracle IDs.
gssusrd.sql	Privileges required to drop an Oracle ID.
gssusrm.sql	Privileges required to alter user object authorizations. This script is a prerequisite for any of the user maintenance scripts (gssusra.sql, gssusrc.sql and gssusrd.sql).
gssviol.sql	Privileges required to view and clear security violations.
gssvpdi.sql	Access to the Oracle/Banner VPD Security Maintenance form (GSAVPDI).

Variables Used in Distributed Security Scripts

Variable	Description
`&&temp_tablespace`	The new BANSECR_*xxx* account's temporary tablespace
`&&user_name`	The new BANSECR_*xxx* account's name
`&&user_role_name`	User role that defines this BANSECR_*xxx* account's privileges. This role must be named *USR_GSASECR_[username]*.
`&&user_password`	The new BANSECR_*xxx* account's initial password

Script Example

For example, the following sequence could be used to create a new security account called `bansecr_example`.

```
start gssacct (also runs gsssysg, gsspriv & gssbasg)
start gssusrm
start gssusra
start gssrolm
```

When prompted, respond with:

- *bansecr_example* for the `user_name`
- *u_pick_it* for the `user_password`
- *temp* for the `temp_tablespace`
- *usr_gsasecr_bansecr_example* for the `user_role_name`

4 Security Auditing

Banner's security auditing capability is designed to save a detailed history of security events affecting the Banner security tables. If auditing is enabled for a security table, any change to that table triggers the creation of an audit record. The audit records are stored in security audit tables and are viewed through the Security Table Audits form (GSAAUDT).

Audit Triggers

When a change is made to a record on a Banner security table, a trigger on that table saves a record of that change in a separate security audit table. There are also triggers for database logon and logoff events.

Each of these audit triggers can be turned on or off in the Institution Profile window of GSASECR. See "Institution Profile" on page 2-28 for details.

Audit Table	Base Table	Trigger	Type of Information
GURAINV	GJRINVC	gt_gjrinvc_audit_row (gutinvc0.sql)	Job Submission Character Validation
GURAEAC	GOBEACC	gt_gobeacc_audit_row (guteacc0.sql)	Enterprise Oracle Access
GURADMN	GOBFDMN	gt_gobfdmn_audit_row (gutfdmn0.sql)	FGAC Domain Driver
GURAEOB	GOBFEOB	gt_gobfeob_audit_row (gutfeob0.sql)	FGAC objects excluded from FGAC processing rules
GURAGAC	GOBFGAC	gt_gobfgac_audit_row (gutfgac0.sql)	FGAC Group Access Rules
GURAPUD	GOBFPUD	gt_gobfpud_audit_row (gutfpud0.sql)	FGAC Personal User Defaults
GURAMSK	GORDMSK	gt_gordmsk_audit_row (gutdmsk0.sql)	Display Mask Column Rules
GURABPI	GORFBPI	gt_gorfbpr_audit_row (gutfbpr0.sql)	FGAC PII domain business profile assignments

Audit Table	Base Table	Trigger	Type of Information
GURABPR	GORFBPR	gt_gorfdpi_audit_row (gutfbpi0.sql)	FGAC business profile assignments
GURADPI	GORFDPI	gt_gorfdpi_audit_row (gutfdpi0.sql)	FGAC PII Policy
GURADPL	GORFDPL	gt_gorfdpl_audit_row (gutfdpl0.sql)	FGAC Domain Policy
GURAGBP	GORFGBP	gt_gorfgbp_audit_row (gutfgbp0.sql)	FGAC Profiles per Predicate and Domain
GURAGUS	GORFGUS	gt_gorfgus_audit_row (gutfgus0.sql)	FGAC Users defined for predicate and domain
GURAPRD	GORFPRD	gt_gorfprd_audit_row (gutfprd0.sql)	FGAC Predicate per Domain
GURAVCL	GTVCLAS	gt_gtvclas_audit_row (gutclas0.sql)	Validation table of user classes
GURAVOG	GTVOWNG	gt_gtvowng_audit_row (gutowng0.sql)	Validation table of security owner groups used in distributed security
GURASGR	GTVSGRP	gt_gtvsgrp_audit_row (gutsgrp0.sql)	Validation table of security groups
GURAIPF	GUBIPRF	gt_gubiprf_audit_row (gutiprf0.sql)	Site profile record
GUBAROL	GUBROLE	gt_gubrole_audit_row (gutrole0.sql)	Definitions of Banner roles
GURAAOB	GURAOBJ	gt_guraobj_audit_row (gutaobj0.sql)	All valid Banner objects
GURAATB	GURATAB	gt_guratab_audit_row (gutatab0.sql)	Forms and tabs that can be used in tab security
GURABGP	GURBGRP	gt_gurbgrp_audit_row (gutbgrp0.sql)	Business profiles belonging to a security group
GURACAL	GURCALN	gt_gurcaln_audit_row (gutcaln0.sql)	Calendars used for logon verification
GURACGP	GURCGRP	gt_gurcgrp_audit_row (gutcgrp0.sql)	Classes belonging to a security group

Audit Table	Base Table	Trigger	Type of Information
GURADSU	GURDSUR	gt_gurdsur_audit_row (gutdsur0.sql)	Rules used in creating new distributed security users
GURAULG	GURLOGN	gt_gurlogn_audit_row (gutlogn0.sql)	Banner Logon rules
GURAOGP	GUROGRP	gt_gurogrp_audit_row (gutogrp0.sql)	Objects belonging to a security group
GURAOWG	GUROWNG	gt_gurowng_audit_row (gutowng0.sql)	Distributed security groups
GURAOWN	GUROWNR	gt_gurownr_audit_row (gutownr0.sql)	Object access for distributed security users
GURACLS	GURUCLS	gt_gurucls_audit_row (gutucls1.sql)	Security classes a user is authorized to access
GURAUGP	GURUGRP	gt_gurugrp_audit_row (gutugrp0.sql)	Users belonging to a security group
GURAUOB	GURUOBJ	gt_guruobj_audit_row (gutuobj0.sql)	Type of access, by user ID, for each Banner object
GURAUSI	GURUSRI	gt_gurusri_audit_row (gutusri0.sql)	VPD Institution/Banner User
GURAUTB	GURUTAB	gt_gurutab_audit_row (gututab0.sql)	User tab security access
GURALGN	(None)	gt_login_audit_access (gutalgn0.sql), gt_logoff_audit_access (gutalgn2.sql)	Login/logoff audit information

Audit Tables

Each Banner security audit table stores a history of changes to a corresponding Banner security table. Each audit table contains all of the columns found in its base security table, plus two additional columns.

Column	Data Type	Description
*TABLENAME*_AUDIT_TIME	TIMESTAMP	The date and time the audit record was created
*TABLENAME*_AUDIT_ACTION	VARCHAR2(01)	Action taken on row: (U)pdate, (I)nsert, (D)elete

There are a few security audit tables that store information derived from sources other than Banner tables. For example, the GURALGN table's information is drawn from Oracle logon activity.

The security audit tables are listed below, along with their base tables.

Audit Table	Base Table	Navigation Path in GSAAUDT	Type of Information
GURAIPF	GUBIPRF	Institution > Profile	Site profile record
GURAINV	GJRINVC	Institution > User/ Password Validation	Job Submission Character Validation Table
GURAVCL	GTVCLAS	Institution > Security Classes	Validation table of user classes
GUBAROL	GUBROLE	Institution > Roles	Definitions of Banner roles
(None)	GURSQLL	Institution > Dynamic Security SQL	Dynamic SQL executed on GSASECR
GURADSU	GURDSUR	Institution > Distributed User Rules	Rules used in creating new distributed security users
GURAUTB	GURUTAB	User Security > Tab Security	User tab security access
GURAUOB	GURUOBJ	User Security > Object Access	Type of access, by user ID, for each Banner object

Audit Table	Base Table	Navigation Path in GSAAUDT	Type of Information
GURACLS	GURUCLS	User Security > Assigned Classes	Security classes a user is authorized to access
GURAEAC	GOBEACC	User Security > Enterprise Access	Enterprise Oracle Access Table
(None)	GURALGN	Logon Audits > Logon/ Logoff Activity	Logon/logoff audit information
GURACAL	GURCALN	Logon Audits > Logon Calendars	Calendars used for logon verification
GURAULG	GURLOGN	Logon Audits > User Logon Rules	Banner Logon rules
GURAUSI	GURUSRI	Logon Audits > VPD Access Rules	VPD Institution/Banner User Table
GURAATB	GURATAB	Object Security > Tab Security	Forms and tabs that can be used in tab security
GURAOWG	GUROWNG	Object Security > Distributed User Groups	Distributed security groups
GURAOWN	GUROWNR	Object Security > Object Owners	Object access for distributed security users
GURAAOB	GURAOBJ	Object Security > Object Definitions	All valid Banner objects
GURAEOB	GOBFEOB	Object Security > FGAC Exclusions	FGAC objects excluded from FGAC processing rules
GURAMSK	GORDMSK	Business Profiles > Masking Rules	Display Mask Column Rules
GURABPI	GORFBPI	Business Profiles > PII Assigned to Business Profile	FGAC PII domain business profile assignments
GURABPR	GORFBPR	Business Profiles > Users Assigned to Business Profiles	FGAC business profile assignments
GURADMN	GOBFDMN	FGAC Rules > Domain Driver	FGAC Domain Driver Table

Audit Table	Base Table	Navigation Path in GSAAUDT	Type of Information
GURADPL	GORFDPL	FGAC Rules > Domain Policy	FGAC Domain Policy
GURAPRD	GORFPRD	FGAC Rules > Predicate per Domain	FGAC Predicate per Domain
GURADPI	GORFDPI	FGAC > PII Policy	FGAC PII Policy
GURAGAC	GOBFGAC	FGAC > Group Access Rules	FGAC Group Access Rules
GURAGBP	GORFGBP	FGAC > Profiles per Predicate	FGAC Profiles per Predicate and Domain
GURAGUS	GORFGUS	FGAC > Users Defined for Predicate	FGAC Users defined for predicate and domain
GURAPUD	GOBFPUD	FGAC > User Defaults	FGAC Personal User Defaults
GURABGP	GURBGRP	Group Security > Business Profiles	Business profiles belonging to a security group
GURACGP	GURCGRP	Group Security > Class	Classes belonging to a security group
GURAOGP	GUROGRP	Group Security > Object	Objects belonging to a security group
GURAUGP	GURUGRP	Group Security > Users in Group	Users belonging to a security group
GURASGR	GTVSGRP	Group Security > Security Groups	Validation table of security groups
GURAVOG	GTVOWNG	Group Security > Distributed Groups	Validation table of security owner groups used in distributed security

Logon/Logoff Activity Audits

All Banner INB access is recorded in the GURALGN table. In addition, if the `GT_LOGIN_AUDIT_ACCESS` or `GT_LOGOFF_AUDIT_ACCESS` triggers are enabled, then standard Oracle logins by Banner users will also be logged in the table.

In order to determine which logins are Banner users (as distinguished from Oracle user IDs that don't have Banner security access) the user ID is checked against the following tables.

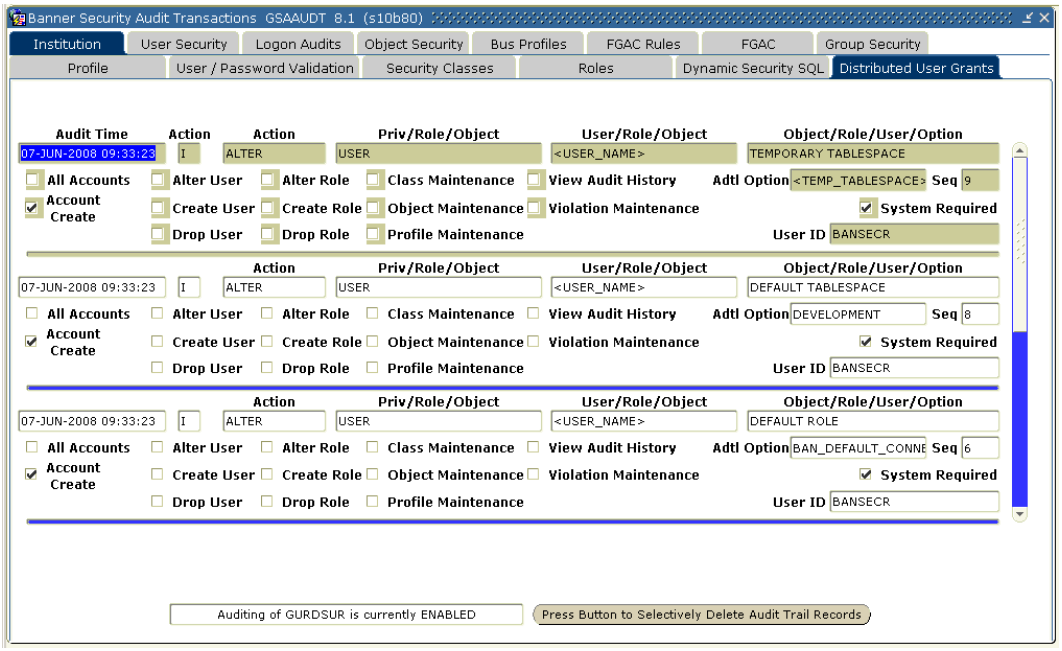
- GURUCLS: There will be a record if the user is a member of any class
- GURUOBJ: There will be a record if the user has direct grants to any object
- GURUGRP: There will be a record if the user is part of any security group

If the user ID is found in any of these tables, the Oracle login is recorded in the GURALGN table and displayed on the Logon/Logoff Activity tab of the GSAAUDT form.

Banner Security Table Audits (GSAAUDT)

The Banner Security Table Audits form (GSAAUDT) provides a convenient way to view and search the security audit tables. The BANSECR account can also delete old records from each of the security audit windows.

The GSAAUDT form uses a two-level tab structure in order to make room for the large number of audit tables. See "Audit Tables" on page 4-4 for a list of the security audit tables and the GSAAUDT navigation for each.



You can use Banner's standard query functionality in any of GSAAUDT's tabs to search for records that meet specific criteria. For example you can search for records associated with specific Banner objects or specific user IDs.

Turning Auditing On or Off

Each of GSAAUDT's tabs displays audit records from a specific table. A message at the bottom of each tab identifies the table that stores the audit records and tells you if auditing for that table is currently turned on or off.

In GSASECR's Institution Profile window, you can turn auditing on or off for each of the security audit tables. See "Audit Trigger Status for Security Tables" on page 2-29.

Note

Turning auditing off for a security audit table does not delete any existing records. Turning off auditing for a table stops the recording of new events in the audit table, but existing records are retained and can still be browsed in GSAAUDT.

Deleting Audit Records

Because large numbers of security audit records can accumulate, you might eventually choose to delete older records. Each security audit table's records can be deleted from the table's corresponding GSAAUDT tab.

Note

The ability to delete security audit records should be restricted to the most trusted security administrators. Therefore, the BANSECR account is the only account that can delete records from any of the audit tables. The gssaudt.sql script can be run to grant query access to BANSECR_*xxx* accounts, but if delete access is required the accounts must be granted delete access specifically for each individual table. ■

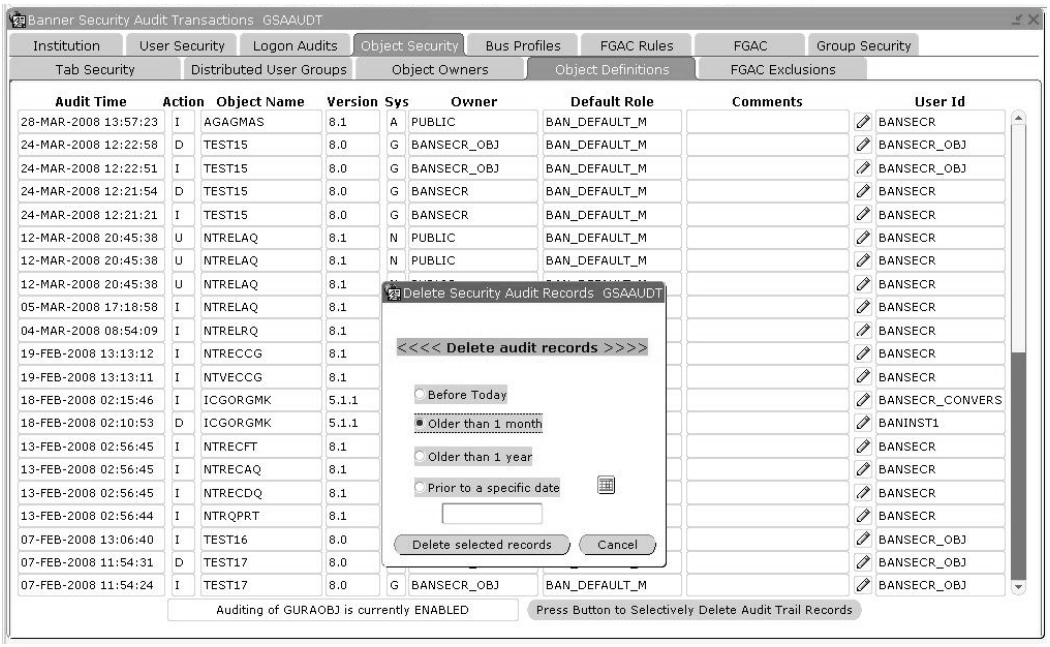
To delete old records from one of the security audit tables, log in as BANSECR and perform the following steps:

1. Navigate to the appropriate tab in GSAAUDT.

2. Click **Press Button to Selectively Delete Audit Trail Records**. The Delete Security Audit Records popup appears.

Note

If the user does not have delete access to the table, the button will not be enabled. BANSECR will always be able to delete, but a distributed user that has access to this form will typically have inquiry-only access. ■



3. Select a time period.

   - *Before today:* All records for this table with a date prior to the system date will be deleted from the history table.
   - *Older than one month:* All records for this table with a date prior to one month ago will be deleted from the history table.
   - *Older than one year:* All records for this table with a date prior to one year ago will be deleted from the history table.
   - *Prior to a specific date:* All records prior to the date entered in the date field will be deleted from the history table.

4. Click **Delete Selected Records**.

Note

An audit of the `delete` statement is captured in the GURSQLL table and can be viewed on GSASECR's Dynamic Security SQL window. See "Dynamic SQL History" on page 2-37.

Warning

If you initiate the delete process after doing a query, then only records that were included in the query results (and that match the date selection that you made) will be deleted. Records that were excluded from the query results will not be deleted, even if they are old enough to qualify for the date selection in the delete process. To ensure consistent results, avoid executing any queries in a tab before you begin the delete process.

5 Security with Shared Connections

Connections to the Banner database take one of two forms: either the end-user authenticated connections used by Oracle Forms and C programs, or the shared connections used by Self Service, channels, and messaging.

Since many areas of Banner depend on a known Oracle user ID for security and auditing purposes, Banner's basic security setup (described in Section 1) was designed with the assumption that the end user's Oracle ID is known. For Banner extensions that rely on shared connections, two different approaches have been adopted for user authentication.

- Banner Self-Service uses DADs (Database Access Descriptors) for database connection information, and rely on Lightweight Directory Access Protocol (LDAP) to handle user authentication
- New extensions to Banner, such as messaging and channels, use JDBC connections to the database, with user authentication managed by proxy connections.

LDAP Authentication for Banner Self-Service

SunGard provides support for Lightweight Directory Access Protocol (LDAP) to perform user authentication for Banner Self-Service. SunGard is supporting all v3-compliant LDAP servers.

You can use the LDAP authentication process to authenticate all your users' IDs and passwords. They can use their LDAP user IDs and passwords to logon to the self-service applications they need to use. The mapping between the LDAP user and the Banner user can be stored on the LDAP server as an attribute, or it can be stored on the Third Party Access Table (GOBTPAC) in Banner General.

Note

The programming logic in Web Tailor that authenticates user credentials in GOBTPAC is bypassed if your institution uses LDAP to authenticate Banner Self-Service.

Note

If your institution is using an LDAP server to authenticate user logons, you cannot modify PINs in Banner General. They must be changed in LDAP.

For additional information, please refer to the *Web Tailor User Guide* and Chapter 4, "Implement Single Sign-On (SSO)," of the *Middle Tier Implementation Guide*.

VBS in Self-Service

Banner Self-Service uses Database Access Descriptors (DADs) for database connection information. For Release 7.0, the self-service products were changed to recognize two different DADs, one with a user ID and password, and one without. This was necessary to support the enhanced VBS with PII using Oracle's FGAC:

1. The DAD without the user ID and password is the existing DAD. It is still used for standard Self-Service logins.

2. The DAD with the user ID and password is new for Release 7.0 and identifies users that will be restricted under the enhanced VBS with PII using Oracle's FGAC. Administrators at your site can specify that certain pages are subject to VBS restrictions via the **Secured Access** indicator on the new Create/Customize a Channel page (`twbkchnl.P_ModifyChannel`) in Web Tailor.

   If this indicator is selected, the new DAD is used instead of the old one, it prompts the end user to enter their Oracle user ID and password. The end user does not need to enter it again during the session unless he or she logs out. The user will be subject to VBS restrictions.

For more information, please refer to the *Web Tailor User Guide* and the *Channel Developer Guide*.

Proxy Authentication for Channels and Other JDBC-Based Connections

Proxy connections are used for Messaging, Channels, and other Banner extensions that rely on Java Database Connectivity (JDBC) for connection to the Banner database. User authentication is handled on a proxy server which mediates between the user and the database server using a method called middle-tier authentication. The proxy server connects to the database as a proxy for (acting on behalf of) the end user.

Middle-Tier Authentication

Banner middle-tier authentication uses the new Oracle user ID BANPROXY. In order to establish a proxy connection to a particular end user's Oracle user ID, it must be authorized by an ALTER USER command, as in this example:

```
ALTER USER oracleuser GRANT CONNECT THROUGH banproxy;
```

Once this is done, the middle tier only needs to know the password for the proxy user — in this example, the password for BANPROXY. It is then up to the middle tier to authenticate the end user, and then establish a connection to the database. The session then appears as if the proxied user is logged in, with the exception that the USERENV attribute `PROXY_USER` is now set to *BANPROXY*. The USERENV attribute `SESSION_USER`, as well as the SQL pseudo-column USER will be set to the proxied user (the end user's ID), and all FGAC processing operates on the proxied user.

Since an Oracle user ID may only be allowed proxy access through one proxy user at a time, it was decided by the architects to create a single Oracle user ID, BANPROXY, to handle all proxy connections.

GSPPRXY Package

This package has been added to BANSECR to map user IDs from external sources to Oracle user IDs that Banner recognizes so Banner can apply the appropriate security.

External users who do not have a one-to-one mapping to an Oracle user ID will use the generic user ID.

The BANPROXY user has only two privileges, CREATE SESSION and an EXECUTE grant to the new package GSPPRXY. The GSPPRXY package is owned by BANSECR, the Oracle user ID that manages Banner security setup.

The input parameters for GSPPRXY are:

Parameter Name	Description
`p_access_method`	The mechanism by which the user is connecting to Banner Self-Service. Valid values are *CHANNELS* and *MESSAGING*.
`p_external_system`	The system from which the user is connecting, e.g., the name of the channel, the external source of the message, etc.
`p_external_user_id`	The user ID of the person in the external system.

The output parameters for GSPPRXY are:

Parameter Name	Description
`p_internal_user_id`	The Oracle user ID of the person. It will either be BANPROXY or the generic role.
`p_oracle_role`	The Oracle role.
`p_oracle_role_pwd`	The password that corresponds to the Oracle role.

The g$_get_proxy_info Procedure

The g$_get_proxy_info procedure insulates the various clients (messaging, channels, etc.) from being aware of the issues surrounding Oracle roles and user mapping. Instead, the client need only connect as BANPROXY (without a proxy user ID), supply the IN parameters of P_ACCESS_METHOD (currently *CHANNELS* or *MESSAGING*), P_EXTERNAL_SYSTEM (which could be the channel name or the external source of the message), and P_EXTERNAL_USER_ID (the end user's ID), and the procedure will return the appropriate Oracle role (managed using existing Banner security forms and tables), the password for the role, and the internal OracIe user ID.

In the case where there is no mapping for the user in either LDAP or GOBEACC, a default Oracle user ID will be used, based on the combination of P_ACCESS_METHOD and P_EXTERNAL_SYSTEM.

The g$_get_proxy_info procedure checks that the PROXY_USER attribute of the USERENV context is *BANPROXY*. If any other value is found for PROXY_USER, the connection will not be allowed, and a security violation will be logged.

Once a proper connection attempt is identified, and the role name retrieved, then a private function in GSPPRXY will be used to decrypt the password stored in the security tables for that role.

CHANNEL Object

A new object named CHANNEL was added in BANSECR for Release 7.0. During security setup, all users or classes that need access to channels must have this object added to their privileges.

Setting Up Users for Proxy Connections

Users which will have proxy connections will need to be altered with the following command:

```
ALTER USER username GRANT CONNECT THROUGH banproxy;
```

This can be accomplished through the Users tab in the GSASECR security setup form, by selecting **Authorize BANPROXY**.

Mapping External Users

When external user IDs are mapped to Oracle user IDs for proxy authentication, multiple sources of data are considered.

If the user is coming from Luminis via a channel, the Banner SSO CPIP adapter's mapping logic is used. This mapping logic uses the LDAP directory to do the mapping.

There is a DN in the directory for users whose Luminis credentials do not match the Oracle ones. If there is an entry there, the mapped user should be used as the Oracle user.

If the Luminis user ID matches the Banner user ID, mapping in LDAP is not required. In these cases GOBEACC is checked to see if a USERNAME mapping exists.

If the user either from the Luminis credential or the mapped user does not exist in Oracle, the default user and role should be used.

Default User Mapping

Banner classes are used to map a default Oracle user ID for particular connection types. This is accomplished by creating new classes named using the following formula:

`pxy_`*accessmethod_externalsystem*

where *accessmethod* is the first 7 characters of the `p_access_method` parameter, and *externalsystem* is the first 16 characters of the `p_external_system` parameter, yielding a class name with a total of up to 30 characters. For example, the class created for proxy users connecting through a channel from Luminis is `pxy_channel_luminis`.

For these special classes beginning with *pxy_*, only one user ID is permitted per class. After the class is determined based on the access method and external system values, the class's user ID is assigned as the default user ID for proxy connections when no known user ID has been determined.

In the case where no appropriate proxy class exists, or no user ID is associated with the class, GSPPRXY will not allow the proxy connection and will return an error message indicating that the class was not set up correctly.

For information on user ID mappings in other scenarios, please refer to "ID Mappings Between Systems" in Chapter 4, "Implement Single Sign-On (SSO)," of the *Middle Tier Implementation Guide*.

6 Multi-Entity Processing (MEP)

Multi-Entity Processing (MEP) enables a user to switch between Institution Codes while logged into Banner. By using MEP, data can be segmented by different organizational entities, such as campuses, within the same physical Banner database environment.

Note

This functionality was previously called Multi-Institution Processing (MIP).

Virtual Private Database

Multi-Entity Processing (MEP) is only available at schools that have implemented Virtual Private Database (VPD) functionality. If your institution is interested in using MEP, it is strongly recommended that SunGard Higher Education Application Practices perform the implementation. Do not attempt to make VPD changes to the database on your own.

You can only set up MEP after VPD has been implemented. Additionally, several Web Tailor components are required for setup if MEP is used in Self-Service Banner or Luminis Channels for Banner.

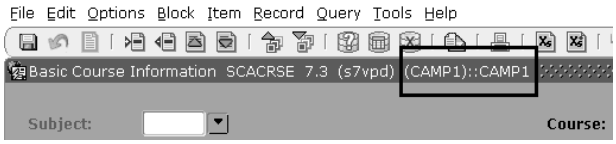
Warning

If your institution does not use MEP, do *not* place any data in the following tables: GTVVPDI and GURUSRI. Placing data in these tables in a non-MEP environment can seriously impact the integrity of your system.

Additionally, your DBA should ensure that the `gt_login_set_vpdi_context` trigger is disabled if your institution does not use MEP.

Home and Process Contexts

At institutions that use MEP, all forms that are MEP-enabled display two contexts in the title bar: the *home* context and the *process* context. In the following example, the home context appears first, and is enclosed in parentheses. The process context appears second, and has no parentheses.



Home Context

The home context is the **Institution Code** that you choose on the Set Institution Form (GUQSETI) when you log in to your Banner session. Your options for choosing the home context vary according to how **Institution Codes** are set up for you on the Oracle/Banner VPD Security Maintenance Form (GSAVPDI):

- If there are multiple **Institution Codes** set up for your User ID on the User Assignment tab of GSAVPDI, then you will have multiple codes from which to choose the home context for your Banner session. If you exit GUQSETI without manually selecting a code, then the code designated as your default **Institution Code** on GSAVPDI will be used as the home context. Your default code always appears first in the GUQSETI list, and is highlighted.
- If there is only one **Institution Code** set up on the User Assignment tab, then that code will be used as your home context, and GUQSETI will not appear when you log in.
- If no codes are set up for you on the User Assignment tab, then the Banner system default **Institution Code** will be used as your home context.

Process Context

The process context is the **Institution Code** to which you switch *during* your Banner session. In a typical MEP implementation, the process context is effectively query-only; you cannot save any changes to the database for the Institution Code of your process context.

Note

If you need to switch to a different Institution Code and make changes to that institution's data, you must restart your Banner session and select the new Institution Code during login. The new Institution Code is then your home context. ■

If you do not change codes during the session, then your process context will be the same as your home context.

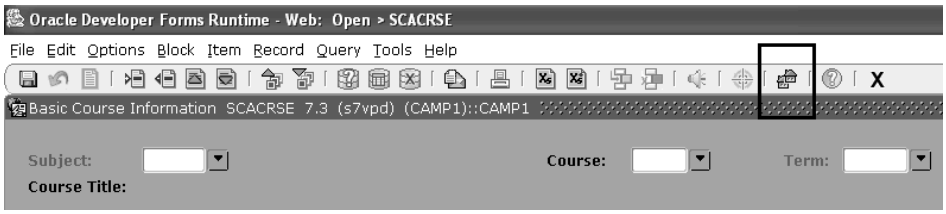
Note

The process context (and not the home context) is used by the Banner Document Management Suite (BDMS) interface. The process context is passed as a parameter in the URL handed from Banner to BDMS. ■

How to Switch Between Institution Codes

If you have permission to access two different institution codes, you can switch from one institution code to another without leaving a form.

1. Click the icon on the toolbar.



   --OR--

   Press the Ctrl-Shift-F10 keys simultaneously.

2. Enter an **Institution Code**. You can use the **Search** button to select a code from the Institution Code Validation (GTVVPDI) list, or, if you want to view further information about the available codes, click the **View Existing Institutions Values** link. If you select a row from the Existing Institution Values list, the values are brought back to the key block of the form. You can then view the data related to those particular values elsewhere on the form.

3. Click **OK**.

   This **Institution Code** (the "process context") now appears in the title bar after the code under which you logged in (the "home context"). Depending on the security policies established by the institution, you may have the ability to insert, update, or delete information using this selected institution code, or you may be limited to only viewing data with this selected institution.

How Changing Institution Codes Impacts your Banner Session

Banner *processes* always use the user's home context, the institution chosen on the Set Institution Code Form (GUQSETI) during Banner login. If you switch to another institution code during a Banner session, Banner forms will use the new institution code (your *process context*), but processes will continue to use your home context.

When you change **Institution Codes** during a session, the code that the forms use is the code that appears second in your title bar. The code you chose on GUQSETI upon login appears first in the title bar.

Example of title bar after you have switched Institution Codes:
`(CAMP1) : CAMP2`

(CAMP1) is the Home Context you chose upon login. It is always used by Banner *processes*, regardless of whether you change **Institution Codes**.

CAMP2 is the Process Context, the code to which you changed. It is used by Banner *forms*.

Job Submission

Your jobs will run under the home institution that you set for the session at login, regardless of the default institution code that is set up for you on GSAVPDI.

For Jobs Submitted Through Job Submission (Not on Hold)

The institution code that Banner uses for processing in a particular session is always the one you chose on login to that Banner session. Prior to this enhancement, if you had several sessions open at once, Banner used the code from the most recently-opened session for jobs that you processed in any of your open sessions, regardless of the code originally chosen for that session. With this enhancement, the institution code used for processing jobs is specific to the session you are in, regardless of other sessions you may have open.

For Jobs Put on Hold for Future Processing

When you put a job on hold on the Job Submission Form (GJAPCTL), Banner saves the institution code that you chose for the session, along with the job sequence number, to a new table. Later, when you submit the job for processing, you must manually set the "one up" environment variable to the job sequence number, so that Banner can retrieve the stored institution code for the job.

The environment variable you must set is:

Platform	Variable	Example
UNIX	ONE_UP	export ONE_UP=123456 (where *123456* is the jobseqno)
WIN NT	SCTBAN_ONE_UP_NUMBER	set SCTBAN_ONE_UP_NUMBER = 123456 (where *123456* is the jobseqno)
VMS	SEQ	ONE_UP_NO :== 123456 (where *123456* is the jobseqno)

Oracle Reports

Running Oracle Reports From the Form

Oracle Reports are run using the process context setting in the preceding example), as long as the form from which the report is being called is listed on GORVPDI.

Note

If the form that calls the Oracle Report is not listed on GORVPDI, then the report can only be run under the user's home context.

Running Oracle Reports From Job Submission

If you have Oracle Reports that can only be run via GJAPCTL, or if the Oracle Report is initiated through GJAPCTL, then the home context is used.

If users need to use the process context for Oracle Reports that are initiated through GJAPCTL, then you may consider adding GJAPCTL to GORVPDI. However, this causes the process context to appear in the title bar of GJAPCTL, which then allows the user to change **Institution Codes**. A user who is executing a C process or other Banner report might assume that the process or report is being executed under the process context institution, which is not the case. The change in the **Institution Code** is effective only for Oracle Reports.

Forms

VPDI Included Objects (GORVPDI)

Use this form to list the forms that are enabled for MEP at your institution. When a form is added to this list, then both the home and process contexts appear in the title bar of the form.

If you add a form that has had VPD changes applied to it, then the MEP toolbar button becomes active and users can access the Institution Code Validation list of values (LOV) from that form. For forms that have not had VPD applied to them, the button is not active.

Note

Do not add forms that do not re-query data and use the *Rollback* function. If the data cannot be re-queried, then the user should not be changing **Institution Codes** while on that form.

This form appears on the Miscellaneous General Forms Menu (*GENMISC).

Field	Description
Object Name	Name of the Banner object. Choices come from the Object Maintenance Form (GUAOBJS) list.
Description	Description of the Banner object. This value comes from the description associated with the object on GUAOBJS.
Activity Date	Date on which the record was created or last modified.
User ID	User ID that created or modified the record.

Oracle/Banner VPD Security Maintenance (GSAVPDI)

The GSAVPDI form allows you to maintain VPD security options. This form is accessed through the *SECURITY menu.

Only the BANSECR user and authorized distributed security users can access this form. Distributed security users can be granted access to this form through the gssvpdi.sql script. See "Scripts to Create a Distributed Security User" on page 3-25.

Selection Window

This window appears when you first access this form. It allows you to specify the next window you will see.

Field	Description
Selection	The option code. Enter a valid value and select Next Block to proceed to the next window. Valid values are: *I* - Institution Code Maintenance *U* - User/Institution Maintenance You can also select a button to proceed to the next window.

Institution Code Maintenance Window

This block allows you to define all the institution codes within Banner. Each code must be created before it can be assigned to a user.

Field	Description
Institution Code	Identifies all valid institution codes. These codes are the key piece of information used to separate data in the database. Each can be up to six characters long.
Institution Description	The description that corresponds to the institution code. It can be up to 30 characters long.
Institution Type	This two-character code allows you to group institution codes by certain criteria. It can also be used to group users. **Note:** This field is not currently used by any Baseline objects. If you want to use it, you must modify Baseline objects. **Note:** You can either modify the `pred_fnc` functions, or add new functions in the `G$_VPDI_SECURITY` package to use the **Type Code** field. If you create different functions, you must change the functions associated with the policy functions.
System Default	Indicates which institution code is the default for your all campuses. It will be the default for all Banner, Oracle, or third-party sessions. Valid values are: *Selected* - this institution code is the default for all users *Cleared* - this code is not the default code You must have one, and only one, default code.
User ID	The user ID for the person who created the data or modified it last.
Activity Date	Date on which this data was created or modified.

User/Institution Maintenance Window

This window lets you associate a user with a particular institution code. Users can have more than one institution code, but the can have only one as the default.

The User ID and institution code combinations determine what data the user can access.

Field	Description
User ID	The user's Oracle ID (created on GSASECR). This must already be set up in Banner.
Institution Code	Code created on the Institution Code Maintenance window.

Field	Description
Institution Description	The description of the institution code. It is populated automatically when you enter the institution code.
User Default	Check box that indicates if the institution code is the default for this user. Valid values are: *Selected* - this code is the default *Cleared* - another code is the default
Activity Date	Date on which this data was created or modified.

Set Institution Code (GUQSETI)

This form allows users with multiple institution codes to choose which one they want to access for this session. It appears only when they log on. Users who have only one institution code will not see this form.

The valid codes are listed with the default at the top. To select a different code, you must clear the check box for the default and select the check box for the new institution code. Save the record.

The fields on this form are display-only.

Field	Description
Institution Code	The valid institution codes for the user. They are pulled from the GSAVPDI form.
Institution Description	The corresponding description.

Tables

The following tables are used for MEP:

- MEP Forms Table (GOBVPDI)
- VPD Institution Code Validation Table (GTVVPDI)
- VPD Institution/Banner User Table (GURURSI)

Index

Symbols

A

B

C

D

L

M

N

O

P

Q

R

S

T

U

V

W