Banner General

Security Administration

Handbook

March 2020

Notices and Privacy

Ellucian
2003 Edmund Halley Drive
Reston, VA 20191
United States of America

Contents

Banner security

Security controls 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. You must have security enforced not just at the application level, but also at the database level.

Securing the Oracle database

Any user on your network who has a valid Oracle user ID and password can access Banner's Oracle database.

Oracle does not tie the user ID and password to a specific means of access. Therefore, anyone with a user ID to run Banner can use this ID to access the database from some other application (such as Banner Self-Service).

You can easily install many third-party tools on desktop computers that directly access Oracle databases. Users using these third-party tools in conjunction with their Banner IDs and passwords, are 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 permission to run a Banner page, that permission should not extend to the user’s own modified version of that same page.

Ellucian designed the Banner security system 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.

Using roles with passwords solves the issue of users using their Banner-required Oracle privileges in a non-Banner application. 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, known only to the database and valid Banner objects. These encryption keys decipher the password 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 objects. Without these seed numbers, any objects 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 pages, COBOL programs, batch Java programs, and C programs.

Procedure

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:
   a) Direct authorizations to the user are checked first and take precedence over other authorizations.
   b) 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.
   c) When a user has access to an object through two or more security classes, only one of the 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.
    a) 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.

    b) 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.

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

Results

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 object 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.

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

Note: Oracle 12c and later does not support a default role that has a password.

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

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. Ellucian 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 12c and higher

In versions before 10.2.0.5, Oracle allowed Default Roles to be passwords protected, such as the `BAN_DEFAULT_CONNECT` role delivered with Banner. In Oracle database 12c and later, a Default Role can not be password protected and Banner uses the `USR_xxxx` convention for non-password protected roles. For example, `USR_DEFAULT_CONNECT`.

For more information on this issue, refer to FAQ 1-XFTOI5.

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 one time for the class and then assigned to many users. This eliminates the need to define the security for each end user.

Ellucian delivers class definitions for each Banner product. These classes will be maintained by the upgrade process and will allow access to every object (page, C, batch Java, 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.

Banner baseline GSASECR security classes

Banner General delivers baseline GSASECR security classes.

Security class	Security class
BAN_ACCOUNTS	BAN_FULL_SECURITY_C
BAN_ADMIN_C	BAN_GENERAL_API_C
BAN_ADVANCEMENT_C	BAN_GENERAL_C
BAN_AIPADMIN_C	BAN_ILP_API_C
BAN_ALUMNI_C	BAN_INTCOM_C
BAN_ARSYS_API_C	BAN_KIOSK_C
BAN_ARSYS_C	BAN_MDUU_ADMIN
BAN_BDR_C	BAN_MDUU_ADMIN_C
BAN_BRIM_SAMPLE_C	BAN_MDUU_USER
BAN_BSAC_C	BAN_MDUU_USER_C
BAN_CMAIPQUERYEXECUTE_C	BAN_MICRFAID_C
BAN_CMQUERYEXECUTE_M_C	BAN_PAYROLL_API_C
BAN_CMQUERY_C	BAN_PAYROLL_C
BAN_CMSTUDENT_C	BAN_PAYROLL_NO_PPAIDEN2_C
BAN_CRM_API_C	BAN_PAYROLL_NO_PPAIDEN_C
BAN_DEFAULT_C	BAN_PB_PAGES_C
BAN_DISTRUBTED_SECURITY_C	BAN_PORTAL_API_C
BAN_EEAMC_C	BAN_POSNCTL_C
BAN_EEAMC_READONLY_C	BAN_PROXY_ACCESS_C
BAN_ELEVATE_API_C	BAN_SHOWALLMENU_C
BAN_EPRINT_FINANCE_A	BAN_STUDENT_API_C
BAN_EPRINT_FINANCE_R	BAN_STUDENT_C
BAN_EPRINT_HR_A	BAN_STUMB_C
BAN_EPRINT_HR_R	BAN_STUVOICE_C

Security class	Security class
BAN_ETHOS_C	BAN_TAB_LEVEL_SECURITY_TEST_C
BAN_FINAID_API_C	BAN_WSINT_C
BAN_FINAID_C	BAN_XTENDER_C
BAN_FINANCE_API_C	BAN_EXTZ_EDS

A “standard” Banner class

The pages that are typically viewed and minimally required for a Banner user are listed below. You can use this list of pages 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 pages that you consider essential, and some of the pages 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 pages listed below require a default role of BAN_DEFAULT_M.

Minimum pages

- GSQTOFU - Terms of Usage
- GUAABOT - Access to the Help About page
- GUAERRM - Enables display of multiple error messages
- GUAHELP - Allows access to page help
- GUAINIT - Required to be able to log into Banner
- GUIOBJS - Enables searching of menu items

Personalization pages

- 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:

- GJAJOBS - Job Search
- GJAPCTL - Page needed to submit Job
- GJIREVO - Page needed to view results in Database

- GJIREVD - Delete Saved Output
- GJRJPRM - Job Parameter Rules
- GJRRPTS - Example Process being run through GJAPCTL
- GUAUPLP - File Upload Utility
- GUQINTF - Page needed to submit Job
- GURINSO - Process needed to insert results into Database

Person/Non-Person Searching

- GUIALTI - SSN/SIN Alternate ID Search
- SOACOMP - Non-Person Search
- SOAIDEN - Person Search
- SOQMENU - Internal Requirement for S% pages

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

A security group is a collection of individual objects and 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 pages, 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 pages, but also to common utilities such as comments pages.

Security checklist

There are certain practices to follow to have a secure Banner setup.

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 USR_DEFAULT_CONNECT will suffice for most users.

   Note: Oracle 12c and later does not support a default role that has a password.

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 page 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 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 their 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`

Users that do not have a default role pose a security problem. The Banner roles are password protected, but they are granted to the end users. This means that when the users do not have a default role, they get all roles granted to them as the default role and do 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, Ellucian will be delivering only Super Roles.

This role will have access to all Banner database objects either directly or through 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 by 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.

If you attempt to secure object types other than pages, 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 by 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 page 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 page supports sites with multiple BANINST accounts. Before the SQL is passed to the database procedure, the GSASECR page 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.

Database scripts

These are the database scripts called by other database packages and functions.

gurcmpa.sql

This script spools a script, which compiles 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 a not valid 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 did not 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 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, because newuser is a variable.

gurgrta.sql

This script grants EXECUTE privilege on the BANINST1_ss9 (Self Service 9) owned stored packages, passed as the first argument, to the role BAN_DEFAULT_M.

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).

gurgrtn.sql

This script grants EXECUTE privilege on the BANINST1-owned stored packages, passed as the first argument, to NLSUSER.

gurgrts.sql

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

gurtrtsso.sql

This script grants EXECUTE privilege on GOKDBMS stored package to BANSSO, if BANSSO exists.

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 using 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.

Section-level security

Some Banner pages are displayed as two or more windows, with navigation between the windows provided by a set of tabs at the top of the page.

By clicking the Biographical section, for example, you can navigate to the Biographical window of the page. Pages with this navigational structure are called section-enabled pages or tabbed pages, and each window in one of these pages is commonly called a section.

Banner security is based on controlling a user's access to Banner objects, such as pages. Through security setup in the Oracle/Banner Security Maintenance (GSASECR) page, each user is granted access to each of the pages that the user needs. Users are not able to access a Banner page unless they have been granted access in GSASECR.

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

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

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 section, while preventing access to the E-mail section. The following example shows the SPAIDEN page with the E-mail, Biographical, and Additional ID tabs hidden from the current user.

The pages listed below have been enhanced for section-level security.

- General Person Identification (SPAIDEN)
- Identification (PPAIDEN) page
- Person Identification (FOAIDEN)
- Vendor Maintenance (FTMVEND)
- Advancement Identification (APAIDEN)
- Banner Distributed Security (GSADSEC)
- General User Preferences Maintenance (GUAUPRF)
- General Student (SGASTDN)
- Student Course Registration (SFAREGS)

Note: Section-level security cannot be applied to other baseline Banner pages until those pages have been enabled by Ellucian. Adding section security records on GSASECR for a page, without coding changes to that page, will not enable section security on the page.

Resolving section-level permissions

If a security class has permissions for a page, and you don't set up section-level security records, the members of that class will have full access to all of that page's tabs.

There is no need to set up section-level security records for a page, except where you want to create restrictions for individual tabs.

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

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

When a user has access to a page 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 section-level permissions that are applied in this instance. Section-level permissions that exist in the user's other classes are ignored.

Set up section-level security for a page

You can set up section-level security for SPAIDEN, PPAIDEN, FOAIDEN, and APAIDEN, or any other section-enabled page.

Procedure

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

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

   If you wish, you can add restrictions for other tabs. Keep in mind that the section 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 section-level security records for each security class, for each page that you want to restrict. (A security class is a collection of Banner objects grouped together for security setup purposes.)
   a) Navigate to the Classes window of GSASECR. Select a security class for which you want to set up section-level permissions.
   b) Click the Objects button. Select an object (for example, SPAIDEN).
   c) Create the section-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 52.)
3. Add section-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 section-level security setup works as expected.
5. Review your locally-created options menu records on GUAOPTM and make any necessary adjustments.

Set up section-level security for a new section

You can set up section-level security for a new section added to a page.

About this task

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

Procedure

1. In GSASECR's Objects window, select the page and review the section's access restrictions.

2. Add the appropriate section-level security records for each security class that will have restricted access to the new section.
   a) Navigate to the Classes window of GSASECR. Select the security class.
   b) Click the Objects button. Select the page with the new section.
   c) Create a section-level security record for the new section.
3. Add section-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 section-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.

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 112.

ENABLE_GKRRSQL_DML security object

this function enables a non-Select statement on GKRRSQL.

Typically this option should never be granted, but if there is a need to create a GKRRSQL rule for the Mass Data Update Utility only, that performs any type of update (insert, update, delete) then the user will have to be granted access to this object in order for the rule to be created.

ENABLE_GORRSQL_DML security object

This function enables a non-Select statement on GORRSQL.

Typically this option should never be granted, but if there is a need to create a GORRSQL rule that performs any type of update (insert, update, delete), then the user will have to be granted access to this object in order for the rule to be created.

GOASDMD security

With Banner General 8.8 release, there is tighter security surrounding the generation and application of views related to the Supplemental Data Engine (SDE) on the Supplemental Data Attributes Definition (GOASDMD) page. Previously, users with access to the GOASDMD page could create SDE attributes and generate and apply new views.

However, users can only generate and apply views if they are granted the new Banner security object BAN_GOASDMD_DDL and have a maintenance access (i.e. BAN_DEFAULT_M) to GOASDMD. A user with query only access (i.e. BAN_DEFAULT_Q) will not be able to generate or apply views even if they are granted BAN_GOASDMD_DDL.

Typically, the GOASDMD page should only be granted to a small set of power users with a technical background. You can use the following SQL to find out the users that currently have access to GOASDMD:

```
SELECT guvuacc_user,
g$_security.g$_get_username_name(guvuacc_user) "Name",
guvuacc_role, guvuacc_type, guvuacc_class,
guvuacc_group, guvuacc_rank
FROM bansecr.guvuacc
WHERE guvuacc_object = 'GOASDMD'
ORDER BY guvuacc_user, guvuacc_rank;
```

After you have identified the users and verified that they should have the ability to generate and apply the new SDE view, you should grant BAN_GOASDMD_DDL to this group. Because it should be a small number of users, granting this to them directly is recommended instead of through a CLASS or SECURITY GROUP.

Note: The granting of access to this object is done by your local Banner Security Administration using the Security Maintenance (GSASECR) page.

GSASECR_BANNER_RULES security object

If a Direct grant is made for a distributed security user to this object using the `BAN_DEFAULT_NO_ACCESS` role, then the distributed security user will not be permitted to access the Banner Rules button on the Users section on GSASECR.

RESET_PIN security object

This function enables `GB_THIRD_PARTY_ACCESS.F_PROC_PIN` to reset the PIN through Letter Generation. If a user needs to reset the pins using the `GB_THIRD_PARTY_ACCESS.F_PROC_PIN` process, they will have to be granted access to this object in order for the letter generation to be generated / executed without errors.

Supplemental Data Engine security objects

These are the Supplemental Data Engine security objects that exist in the Security Object table (GURAOBJ), but do not exist in the Banner Object Base table (GUBOBJS).

SDE_SQL_VALIDATION

Enables update of the GORRSQL Process / Rule, used for 'free form' dynamic validation of values, on GOASDMD. It also enables testing of those rules.

SDE_SQL_TESTING

This object. or `SDE_SQL_VALIDATION`, enables execution of the GORRSQL procedure as part of the 'Test Data' process.

Note: No special security is needed to execute the procedure during normal LOV validation at SDE value data entry.

SDE_LOV_<table

Allows the user to modify data on GTVSDLV for the specified table.

SDE_TEST_<table

Allows the user to test LOVs for the specified table with 'Test Data'.

The table may be `DEFAULT` to modify the generic tables or `ALL` to indicate they can update any table on GTVSDLV, for example, `SDE_LOV_ALL`, `SDE_TEST_SPRIDEN`, and `SDE_LOV_DEFAULT`.

Note: To allow anyone to update and test all fields of GOASDMD you should create `SDE_SQL_VALIDATION` and `SDE_LOV_ALL` and assign them to the `BAN_GENERAL_C` class. Security can be set up as fine grained as needed, by table, by class, or by user.

For additional information on each of the following security objects, please refer to the “Security Setup Requirements”, “Supplemental Data Engine”, and “Supplemental Data Engine (SDE) Data Validation” sections in the *Banner General User Guide*.

SSN_SEARCH security object

The `SSN_SEARCH` object enables users to enter a value in a Banner ID field (SPRIDEN ID name search) in Banner Admin 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 (GSASECR) page.

You grant users permission for SSN searching by giving users access to the object `SSN_SEARCH`.

As with any Banner object, you can grant 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 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`.

You must meet the following conditions to enable SSN searching:

- Your institution has the SSN Search feature enabled. To enable this feature, select the **Enable SSN Lookup** check box on the Installation Controls (GUAINST) page.
- The ID field has extended ID/name search capability built into the field.

User Preferences for Admin Master security object

The BAN_PERSISTENT_PROFILE_MASTER object provides master users the ability to save user preferences (grid/column, filter and pagination) that apply to all users that do not have their own user settings applied. Users granted the BAN_PERSISTENT_PROFILE_MASTER object have the **Master** check box visible on the section/block header for administrative pages that have User Preferences for Admin functionality enabled on the Object Maintenance (GUAOBJS) page.

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				

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

Object Name	Owner	Object Type	Select	Insert	Update	Delete	Execute
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.

You can grant GSAPASR (Password Reset), to distributed security users to enable password changing. In addition, you can use the User ID Restrictions section on GSADSUM to limit access to specific accounts so that the passwords cannot be changed. Passwords may not be changed on accounts that have a default tablespace of SYSTEM or SYSAUX (typically system type accounts). Password changes are also restricted on Oracle 'predefined administrative or non-administrative' accounts (see http://docs.oracle.com/cd/B28359_01/server.111/b28337/tdpsg_user_accounts.htm).

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 page will pass grant commands to this procedure when a Banner-owned object has to be granted to a role.

Object Name	Type	Description
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$_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$_VPDI_SECURITY	Package	Procedure to set the home context of the user logging in. This is also used when the user has access to more than one institution and chooses one.
G$_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 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.

Object Name	Type	Description
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.
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 page.
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.
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 security routine hex keys.
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.

Object Name	Type	Description
GURADMN	Table	GURADMN: This table stores audit information for the GOBFDMN table.
GURADPI	Table	GURADPI: This table stores audit information for the GORFDPI table.
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.
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.

Object Name	Type	Description
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 pages 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.
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.

Object Name	Type	Description
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.
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 pages and their related tabs that are in use for tab based security for a user or a class.

Securing Single Sign On (SSO) access on GOATPAD and GOAEACC

The Banner Administrative environment requires the use of an external identity provider authentication method for all administrative users including special accounts such as BANSECR, BASELINE, and BANSECR_% Distributed Security accounts.

Before Single Sign On (SSO), the Third-Party Access Audit (GOATPAD) page and the Third-Party Access (GOATPAC) page controlled the PINs only used to access Self-Service. Typically, many users had access to this page to reset the PINs for employees and students. Banner Administrative access was controlled by the Oracle password maintained on GSASECR or GSAPASR which were only accessible by the security BANSECR% users. Additionally, users could reset their own password on the Oracle Password Change (GUAPSWD) page.

With the introduction of external identity provider authentication (e.g. CAS, LDAP, etc.) and SSO, the maintenance of the Third Party ID on GOATPAD and relationship between the Username (Oracle ID) and ID (SPRIDEN_ID) on GOAEACC raise the potential need for additional security efforts.

In the event that the Third Party ID and PIN used for external identity provider authentication are not maintained through Banner events, the access to this information within Banner is not as critical because it is maintained externally. In the event that you are using Banner events for external identity provider account maintenance, it is highly recommended that you refer to the information in this section.

- If the PIN maintained on GOATPAD enables password maintenance through events to your external identity provider authentication system, it may be possible for a non-BANSECR user to change the PIN/password for the BANSECR account, or any account accessed through your external identity provider authentication system.

- The GOAEACC relationships could also be potentially changed to establish a relationship between a non-security user and the BANSECR account, or any account.

Fine-Grained Access Control (FGAC) policies for GOBTPAC and GOBEACC tables

Banner General identifies a template for some Fine-Grained Access Control (FGAC) policies to help mitigate these potential security concerns.

Additional information on the policies is provided below.

In addition to the FGAC policies, you should review some additional steps to secure this information.

- You should review access to the GOATPAD, GOATPAC, and GOAEACC pages and limit access to individuals that really need it.
  - The following SQL, executed as BANSECR, shows which users have access to these pages and how they have been granted access.

    ```
    SELECT * FROM guvuacc
     WHERE guvuacc_object IN ('GOATPAD','GOATPAC','GOAEACC')
     ORDER BY guvuacc_object, guvuacc_user, guvuacc_type;
    ```

  - The following SQL lists if any of the pages are assigned to Security Groups maintained on GSADSEC/Group Details/Objects.

    ```
    SELECT * FROM gurogrp
     WHERE gurogrp_object IN ('GOATPAD','GOATPAC','GOAEACC');
    ```

  - The following SQL shows each class in which the pages are defined on GSASECR/Classes/Objects.

    ```
    SELECT guruobj_object, guruobj_userid FROM guruobj, gtvclas
     WHERE guruobj_object IN ('GOATPAD','GOATPAC','GOAEACC')
       AND guruobj_userid = gtvclas_class_code
     ORDER BY 1,2;
    ```

  - The following SQL shows which pages are granted specifically to a user on GSASECR/Users/Modify.

    ```
    SELECT guruobj_object, guruobj_userid FROM guruobj
     WHERE guruobj_object IN ('GOATPAD','GOATPAC','GOAEACC')
       AND NOT EXISTS (SELECT 1 FROM gtvclas WHERE gtvclas_class_code
     = guruobj_userid)
     ORDER BY 1,2
    ```

  - The following SQL shows which users are exempt from any type of FGAC, not just these new policies. This list should be very short.

    ```
    SELECT * FROM dba_sys_privs
     WHERE PRIVILEGE = 'EXEMPT ACCESS POLICY'
    ```

```
ORDER BY 1;
```

- Because special accounts such as BANSECR% and BASELINE now require a dummy SPRIDEN ID, those SPRIDEN IDs should follow a standard, and that standard could be that they all start with SECURE (e.g. SECURE001, SECURE002, etc.). This enables a quick and easy method to identify SPRIDEN records that are being created solely for the purpose of external identity provider authentication.
- Allow only certain privileged users to access records associated with the SECUREnnn IDs on GOATPAD and GOAEACC (GOBTPAC and GOBEACC tables respectively). This prevents anyone from updating the tables through an Banner page or through SQL directly.
- Allow only certain privileged users to access records on the Enterprise Oracle Access Table (Page=GOAEACC, Table=GOBEACC) associated with BANSECR% and BASELINE. This prevents anyone from updating the tables through a Banner page or through SQL directly.
- Allow only certain privileged users to change the PINs on the Third Party Access Table (Page=GOATPAD, Table=GOBTPAC) for users that are defined on GOBEACC. This allows the vast majority of student PINs to be maintained while preventing external identity provider account PINs from being updated and prevents anyone from updating the tables through a Banner page or through SQL directly.

FGAC policy package

A package named GOKBEIS and two scripts `gokbeis0.sql` and `gokbeis1.sql` have been created to help implement these FGAC policies.

Warning! It is important to remember that the GOKBEIS package only defines the policy predicates. You must add the policies using the `gfbeisaddpol.sql` script to add the policies that use the GOKBEIS package. The `gfbeisdroppol.sql` script enables the dropping of the policies if required. Both of these scripts can be found in the `$BANNER_HOME/general/plus` directory and both should run using BANINST1.

The package and scripts are created separate from the standard Banner FGAC process to remove the Banner FGAC setup pages and potential for exemptions and exclusions to the rules allowed by Banner. These FGAC policy functions on GOBTPAC and GOBEACC ensure that only authorized users are allowed to update the GOBTPAC and GOBEACC tables through Banner or directly through SQL.

The GOBEACC table contains a link between a Username (Oracle ID) and ID (SPRIDEN_ID). This relationship is used by Single Sign-On to determine the type of Object Access the user is entitled to through the standard Banner security processes. This table is updated through the GOAEACC page.

The GOBTPAC table contains a PIN and third party ID. The PIN is used for SSB access and external identity provider authentication (if the messaging is enabled for account maintenance with external identity provider authentication). This table is updated through the GOATPAD page.

Because all external identity provider accounts require a Username linked to an ID (including special accounts like BANSECR) Ellucian recommends a Best Practice to use an ID (SPRIDEN_ID) that begins with SECURE and is suffixed by a number (e.g. SECURE001, SECURE002, etc.).

The following is an example of the data relationships.

```
select spriden_pidm, spriden_id, substr(spriden_first_name,1,20),
 substr(spriden_last_name,1,20)
  from spriden where spriden_id = 'SECURE001';
select gobtpac_pidm, gobtpac_external_user, substr(gobtpac_pin,1,20) from gobtpac where
 gobtpac_pidm = 80374;
select gobeacc_pidm, gobeacc_username from gobeacc where gobeacc_pidm = 80374;

SPRIDEN_PIDM SPRIDEN_ID (SPRIDEN_FIRST_NAME)            (SPRIDEN_LAST_NAME)
------------ ---------- ------------------------------- ------------------------------
       80374 SECURE001  Ban                             Secure

GOBTPAC_PIDM GOBTPAC_EXTERNAL_USER          SUBSTR(GOBTPAC_PIN,1,20)
------------ ------------------------------ ------------------------
       80374 bsecure                        A461AB126A0814C2ADF6

GOBEACC_PIDM GOBEACC_USERNAME
------------ ------------------------------
       80374 BANSECR
```

Policy predicates and variables in the GOKBEIS package (gokbeis0.sql and gokbeis1.sql)

There are several policy predicates and variables defined in the package. This package is intended to be used as a template that you should review and modify for your own requirements.

Note: The `gfbeisaddpol.sql` script will not be executed during installation. It will have to be executed manually when the client is ready to have the policy enforced.

You can review and modify the following values as needed:

lv_exempt_users - List of users that are always allowed access

```
   lv_exempt_users VARCHAR2(500) := q'['BANSECR', 'GENERAL', 'BANINST1', 'OAS_PUBLIC',
 'WWW_USER']';
```

lv_spriden_prefix - SPRIDEN ID prefix to identify users that have a dummy ID for external identity provider authentication (i.e. BANSECR)

```
    lv_spriden_prefix VARCHAR2(09)  := 'SECURE%'
```

lv_debug - Set to 'Y' to add debugging rows to GURDBUG

Refer to the source code for the full and current definitions for each sample set of SQL to be used for the policies, keeping in mind that these are suggested templates and should be reviewed for appropriateness in your environment.

The following predicates are being used:

Field	Description
allow_all_access	Allow access to everything.
deny_all_access	Deny access to everything.

Field	Description
owner_and_non_gobeacc	Allow users to access GOBTPAC records for users who do NOT have any GOBEACC row for the PIDM or for rows with the SPRIDEN_ID that is the same as the reserved lv_spriden_prefix.
special_account_access	Allow a specific list of users (i.e. BANSECR) to create / update / delete rows in GOBEACC for rows when the PIDM is associated with BANSECR_%, BASELINE, or MAINTENANCE Oracle IDs or the SPRIDEN_ID is the reserved lv_spriden_prefix.

The following predicates have been defined as a template in the event that you desire additional rules:

Field	Description
owner_only_access	Allow users to access GOBTPAC records for their own USERNAME if they have a GOBEACC row.
only_access_non_gobeacc	Allow users to access GOBTPAC records for users who do NOT have any GOBEACC row for the PIDM.

Because the majority of the security on the GOBTPAC and GOBEACC tables is the same for each of the Insert, Update, and Delete operations, each function performs the same logic and is centralized in a common logic function.

Note: Due to the design of the GOATPAD and GOAEACC pages, if the SELECT access is different than the update/insert/delete access, the user will get a message indicating `the record was updated by another user` instead of a `security violation`. If the policies are the same, then the records are not displayed to begin with and an insert generates a `security violation` error message.

Enabling special accounts to login in through an external identity provider authentication to Banner

As stated earlier, all accounts must log in through an external identity provider authentication to access Banner Administrative pages.

About this task

Special accounts are those that are typically not related to a person who has only one account. Examples of special accounts are BANSECR, BASELINE, and any of the BANSECR_% accounts used for distributed security.

Following the Best Practice recommendations mentioned earlier, this is the basic process that can be used to establish the minimum data required for a special account. In this example, we will set up external identity provider login credentials for the Username BANSECR using the SPRIDEN ID prefix of 'SECURE'.

Procedure

1. Create a SPRIDEN record with minimum information such as ID, first name and last name. For example, the SPRIDEN ID would be SECURE001, the First Name=Banner, the Last Name=Secure.

   Note: No other information is required, and no additional information really should be entered.

2. Go to GOATPAD and create a new entry with a PIN (and Confirmed PIN). For example, 3Yg8M.s17.

   Note: Ensure the PIN Rules defined on the GUAPPRF page are set for maximum security.

3. Select the **Accepted** check box.

   When you save your changes, a Third Party ID with the value `bsecure` is automatically generated using the first letter of the first name followed by the last name.

   Note: The total length of characters from the last name that will be used is determined by the **Length of Last Name for Generate** on GUAPPRF. The total length of the generated third party ID has a maximum of 30 characters and will be converted to lower case. If a duplicate exists, it will search for a unique ID substituting the last 1, 2, 3 or 4 characters with a number from 1 to 9999 until a unique name is found.

4. Relate the Username and ID on GOAEACC.
5. Insert a new record with a Username of BANSECR and an ID of SECURE001.

   After this relationship has been created, the user bsecure can log on to the external identity provider authentication with a password of 3Yg8M.s17 and access all pages that BANSECR has access to.
6. Open the GSASECR page as BANSECR and confirm that you have the **Authorize BANPROXY** check box selected for BANSECR and BASELINE users.
   a) Enter *USERID* in the **User ID** field where *USERID* is BANSECR or BASELINE.
   b) Select **Alter**.
   c) Select the **Authorize BANPROXY** check box if not already selected.

Unified Digital Campus Identifier (UDC ID) associated with a Banner ID

The Enterprise Identification Controls (GOAEIDE) page displays an Enterprise ID (Unified Digital Campus Identifier (UDC ID)) associated with a Banner ID.

This page enables a query on the Enterprise ID or ID field to return the associated Banner ID, Name and Enterprise ID. It also allows inserting and deleting records but does not allow updating an existing record.

Field	Description
Enterprise ID	The Enterprise ID (UDC ID) associated with the Banner ID.

Field	Description
ID	A user's Banner ID. This must be an existing ID previuosly set up on an *IDEN page. List: Person Search (SOAIDEN) or Alternate ID Search (GUIALTI)
Name	Name associated with the ID on an *IDEN page. Display only.

Ellucian recommends reviewing access to the GOAEIDE page and limiting access to those individuals that require it.

Ellucian delivers GOAEIDE in the BAN_ADMIN_C class.

Maintain user accounts with Banner Access Management

A Banner user account is an Oracle user ID tied to specific Banner permissions. Use the security maintenance pages GSASECR and GSADSEC to create and maintain Banner user accounts.

The Banner General 9.3.15 release, delivered in September 2019, moved the Banner Security Pages (ex. GSASECR) from Banner General into the Banner Access Management (BAM) 9.3.15 application, also delivered in September 2019. The Banner Access Management application allows you to log in with Single Sign-on (SSO) credentials or Direct Oracle Login Credentials.

GSASECR also allows you to set institution-level options for security and review a log of security events and potential security violations.

Menu

The menu provides convenient access to the security administration pages accessible using the menu icon on the upper left corner of the page or using the keyboard shortcut, ALT+CTRL+M.

For use with a screen reader, (i.e. Jaws) you can use keyboard shortcut ALT+CTRL+U to read back the Oracle username that the user is logged in as.

Note: The menu populates only those pages that the user has access to. All users can access any of the non-security pages delivered with the Banner Access Management application, but not all users can access the Banner security pages. Those pages are accessible only to BANSECR and other explicitly assigned `BANSECR_xxx` user IDs. Ellucian recommends that you review security classes and remove any permissions to security pages where necessary to prevent regular users from seeing the security pages on the menu.

Security related pages

Distributed Security Users (i.e. `BANSECR_xxx`) privileges are granted through a password secured role with the name BAN_GSASECR_BANSECR_xxx.

In addition, if the DBA role is password protected, access to any of the GSA% pages by the BANSECR account will be initiated with a prompt for the DBA password. After the password has been entered for a session, it will not need to be entered again.

The new DBA Prompt Password (GSADBAP) page is a quasi-internal page that will be called by the GSA% pages if the user is `BANSECR` and the DBA role has a password. It does not appear on any menu and there is no need to execute it directly.

Security maintenance (GSASECR)

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

The GSASECR page 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 page with seven tabs. You can navigate to any of the seven functions by clicking the corresponding section at the top of the page.

Section	Description
Users	Oracle User Maintenance. In the Oracle user maintenance section, 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 section, object permissions that are common to many users can be defined and shared by the users.
Objects	Banner Object Maintenance. In the object maintenance section, you can define Banner objects to the security system.
Roles	Role Maintenance. In the Oracle role maintenance section, 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 section 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 Ellucian-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 Ellucian-delivered IDs. Remember to change the passwords of these accounts. In addition, the User ID Restrictions section on GSADSUM can be used to limit access to specific accounts so that they will not be changed / deleted.

Main window

The main window is used 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 Delete a user account on page 75.
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 section.

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 Create a new user on page 73.

Note: Global rules for creating account names and passwords may have been established on the Institution section 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 Copy another user account on page 75. 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.
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. There are institutional settings that affect your choice of password. See Create a new user on page 73.
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 74 for an explanation of the default role and why it is necessary.

Field	Description
Profile	Enables the execution of a package for stronger password validation in addition to 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.
Authorize BANJSPROXY	Authorizes this user to be authenticated for Job Submission through BANJSPROXY.
Lock Account (check box)	Check this box to lock the account so the user cannot access it. 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. 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. 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. The Institution Profile section on GSASECR can establish rules that force new accounts to be created in a pre-expired state. 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. 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). This field displays only when altering an existing User ID.
Password Expires	The date on which the user's password expired. 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. This field displays only when altering an existing User ID.
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.

Field	Description
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.
Accepted Date	Date that the user accepted the Terms of Usage.

Field or Button	Description
Administrative Banner Active From	The beginning date of the period when this user will be permitted access to Administrative 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. This date controls only Administrative Banner access. It has no impact on the status of the Oracle account or a Self-Service Banner account.
Administrative Banner Active To	The ending date of the period when this user will be permitted access to Administrative 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. This date controls only Administrative Banner access. It has no impact on the status of the Oracle account or a Self-Service Banner account.
Administrative Banner Login Calendar	You can optionally assign a login calendar (created on the GSADSEC page) 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.
Administrative Banner First Login	The date and time of the user's first login to Administrative Banner.
Administrative Banner Last Login	The date and time of the user's most recent login to Administrative Banner.
Administrative Banner Login Count	The total number of times the user has logged into Administrative Banner as a Banner user.
Self-Service Banner 9.x or Application Navigator First Login	Date and time of first login to Self-Service Banner 9.x or Application Navigator.
Self-Service Banner 9.x or Application Navigator Last Login	Date and time of last login to Self-Service Banner 9.x or Application Navigator.
Self-Service Banner 9.x or Application Navigator Login Count	Number of times the user has logged into Self-Service Banner 9.x or Application Navigator.
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, and rules for data masking.
Security Group	Use this section to assign the user ID to one or more security groups. Security groups are created on the GSADSEC page.

Field or Button	Description
Last Update	The user ID and date of the latest update to this user's security record. This field is only updated when the GURLOGN table is updated by user updatable columns.
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. This button is available only if you are maintaining a user.
Copy Privileges	This button launches the Copy Privileges popup so you can apply another user's full set of privileges to this user. 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.

Section-level security settings

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

The **Copy All Tabs** button adds records for all of the selected page’s tabs. After adding an object to a user or class, you can navigate to the tab section 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 lookup button.

Note: When establishing section privileges for the APAIDEN page, it is recommended that the Household Members section have the same section privileges as the Address section.

Note: If page and section 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 section records set up at the class level are ignored if the user has a direct object grant for the page. If a particular section is not defined at the user level, then the user's access to the section will default to the user's overall page privilege.

Field	Description
Section Name	The section name as referenced internally by the Banner system.
External Section Description	The section name that displays on the Banner page. This value is descriptive information only. It does not control what is actually displayed on the section.

Field	Description
Section Access	Select one of the three option buttons to specify the user/class access privileges for the selected section. • `Full:` The user/class has full access to this section. (Note that access for a section may never exceed access defined at the page level. For example, if a user's access to the page is query-only, then full access for all the tabs will, in practice, be query-only access.) • `Query:` The user/class can view this section but cannot change any data. • `Not Visible:` The user/class will not even be able to see this section. Depending on the settings for this section (as displayed in the section information section of the Objects window), your choice of options here may be limited. For example, if a section is defined on the Objects window with a `Full Access` restriction then you cannot set the section 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 allows you to 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 page 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 and 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 Review users' object access on page 72.

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 page 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, to manage the size of the table.

The 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 section to correct the problem.
1	No records found on GUBIPRF	The security profile record was not found. Create one using the profile maintenance section.
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 page is old or the version number is wrong in the GURAOBJ table.
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.

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 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.

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 59.

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. 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.

Field	Description
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.
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. This function may take a while to complete.
Security Owners	Navigates to the Class Owners window of the Banner Distributed Security (GSADSEC) page.

Creating a new security class

You can use the Class window to create a new security class.

Procedure

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:
4. Select the existing class.
5. Click the Duplicate button. A new class record is created.
6. Change the new class's code (which was generated automatically) to a meaningful code to identify the new class.
7. 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.

Procedure

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, 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 is delivered with permissions for the Banner security objects and other pages needed for security administration. BAN_FULL_SECURITY_C includes permissions for the following objects.

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

Ellucian recommends 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 page, so they will be unable to delete security audit records.

Note: You should use the BAN_FULL_SECURITY_C class 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.

Normally, 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 pages 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 52.

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 section.

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.

Field or Button	Description
Users and Class that have Section Security	List the class and users that have section security defined for this object.
Security Owners	Displays the Object Section on GSADSEC showing the owners and their privileges for this object.

Section-level security settings

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

Note: Because relatively few Banner objects are tabbed pages, this section will be blank for most objects.

For tabbed pages, the tab section 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 section can have one of three values:

- `Full Access Required` means that you cannot restrict access to this section. Any user who is able to access this page will have full access to the section.

  Note: Section access is always limited by a user's level of access to the page containing the section. For example, if a user has query-only access to the page, then a the user will have query-only access to the section even if given `Full Access`. A section may never provide more access than the page that contains it.

- `Query or Full Access Allowed` means that you can give some users query-only access to the section while other users have full access, but you cannot hide the section completely. Any user with access to this page will at least be able to query and view data in the section.
- `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 section, making it completely inaccessible for those users.

You cannot change the system required **Access Restrictions** setting for a section. These settings are determined by Ellucian based on analysis of each page.

Field	Description
Internal Section Name	The section name as referenced internally by the Banner system.
External Section Name	The section name that displays on the Banner page. This value is descriptive information only. It does not control what is actually displayed on the section.
Access Restrictions	Assign access restrictions for the section. 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.

Field	Description
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 and 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 on GUVUACC, see Review users' object access on page 72.

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

The main window is used to create or delete a role, or change a role's permissions.

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 section.)

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 page. It is also implemented in the LOV that displays the roles that can be used with a specific page. The `BAN_DEFAULT` roles can be used with all pages 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 page 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 page. With the implementation of the Banner APIs, pages not being called in query mode could allow the user to perform other DML operations even if the user's role for that page only has `select` privileges. Banner security relies on the `_Q` convention for role names to know when to call a page 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 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

You can use this window to create a 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 and 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 page, 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.

Button	Description
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 Start Over button 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 mode on page 70 for details on the options available.
Initial Password	Settings for password security. See Initial password on page 70 for details on the options available.
Seed Number 1, 2, 3	Seed numbers for encryption. See Seed numbers on page 71 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. **Note:** Banner no longer supports Banner 9 page-level version
Call Query	Call Query is a security feature that can be turned on or off. See Call query on page 70.
Terms of Usage	Displays a terms of usage message that can be edited on this page. The value in the Terms of Usage field is the number of days before an accepted terms of usage must be re-acknowledged by the user.
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.
Proxy Job Submission SID	Database connection identifier that is tied to BANJSPROXY in the Oracle*Wallet.
Enforce Job Submission Proxy	Check box to indicate if the users will be prevented from submitting Jobs if they are not authorized to use BANJSPROXY.
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.

Field	Description
Key1 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key2 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key3 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key4 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.

Note: When you update any of the encryption Key Hex Values, you must rerun the $BANNER_HOME/general/plus/banner_encrypt_bdm_data.sql script in Banner 9x applications that use BDM. This will regenerate the BDM encrypted password using the new ex values.

Audit trigger status for security tables

You can 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 (GSAAUDT) page.

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 . 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. 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 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)`.

In addition, the following were added for additional GURJOBS validation:

- "GJU" GurJobs Unix/Linux
- "GJW" GurJobs Windows

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.

Note: See Create a new user on page 73 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 not valid characters (the listed characters are prohibited, and all other characters are permitted).

Field	Description
Validation Characters	The list of characters considered valid or not valid (depending on the Valid/Invalid selection) for this parameter. 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 not valid characters.

Establish a list of unsupported characters
You can establish a list of unsupported characters for a user ID, password, and print command and store them in Character Validation Table (GJRINVC).

Procedure

1. Select the record for the parameter type you want to edit.
2. Click the **Invalid** option button.
3. Paste or type the unsupported (not valid) characters in the **Validation Characters** field.

Results

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`	!&#*

Establish a list of supported characters
You can establish a list of supported characters for a user ID, password, and print command and store them in Character Validation Table (GJRINVC).

Procedure

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

Results

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`	ABCDEFGHIJKLMNOPQRSTUVWXYZ abcdefghijklmnopqrstuvwxyz1234567890

No Job Submission character restrictions
If you do not have a need to limit your user's 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 permit your users to use any character in their job submission parameters.

Warning! Make sure that you do not delete the seed data records in the GJRINVC table, even if you do not plan to use this feature. Those records must be in place for Job Submission to work correctly.

The HOST and WAIT commands detect unauthorized and unregistered commands (not in GJBJOBS) from executing. However, you can bypass these commands by assigning the special object GURJOBS_BYPASS_ADDL_CHECKS with any role other than BAN_DEFAULT_NO_ACCESS to BANSECR user.

The following two rows validate the GURJOBS command_string. If they detect characters that are not valid, the command aborts.

Button	Description
GJU	GurJobs Unix/Linux

Button	Description
GJW	GurJobs Windows

Security mode

The system provides role-level security.

The role-level security 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 pages. First, it prevents an end-users 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-users from creating their own pages by the same name as Banner pages.

Initial password

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

Option	Description
`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 page security. When the **Call Query** field is set to `Enabled`, any page 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 page'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 page to another page, the second page will also be in query mode regardless of the user's permissions for the second page.

Following is an example of this behavior.

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

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

When **Call Query** is set to `Disabled`, a user's access to each page is determined directly by that user's permissions as set up through GSASECR. In the example above, the user will see page A in query mode, but will be able to edit data in page B, even if the user navigates directly from page A to page 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.

About this task

The three seed numbers are stored in the following places:

Procedure

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. Edit `BANNER_HOME/general/java/BatchSecurity.java` and change the following lines.

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

   Save changes and recompile `gurjbif.jar` (make sure that the Java environment is set correctly).

OS	Command to recompile gurjbif.jar
UNIX	cd $BANNER_HOME/general/miscgupdjar.shl
Windows	cd %BANNER_HOME%\general\miscperl gupdjar.pl

Changing seed numbers

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

After the compilations are complete, you must hide the COBOL source and object code, C header file, and `BatchSecurity.java` file from other users. These steps 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 section 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 page displays an extensive record of security activity. See Banner Security Table Audits (GSAAUDT) on page 108.

Manage user accounts

This section describes how to manage the user accounts. Managing a user's account involves reviewing user's object access, creating a new user, copying another user account, and deleting a user account.

Review users’ object access

You can review a user's object access by using the Object Access by User View (GUVUACC) and Object Access by User Table (GURUACC).

Object Access by User View (GUVUACC)

This view allows you to 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 page exactly what access a given user has for a given object. This view gathers together all pages 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 page and also has access to the same page through membership in a security class. In cases like these, the **Rank** column helps you determine which page 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: through a Class (Class), through a Direct grant (Direct), through a class defined in a group (Group Class), or through 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 through a direct grant, this will be NULL.
guvuacc_group	The Security Group where access was defined. If access was defined either directly or through a class this will be NULL.
guvuacc_rank	The order in which the roles will be used. 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 provides 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
```

Create a new user

You can create a new user account on the User window in GSASECR.

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*Plus, SQL*Developer, etc.

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 Section of GSASECR. See Initial password on page 70.

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 67.

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 Ellucian's FAQ 10718 for detailed guidance on Banner passwords and user IDs.

Case-sensitive passwords with Oracle database 12c

Starting with Oracle Database 12c, Oracle supports case-sensitive passwords with the default setting value of on.

For backwards compatibility, if your database continues to use case-insensitive passwords, you must set the initialization parameter SEC_CASE_SENSITIVE_LOGIN to false. Oracle has deprecated this initialization parameter and the use of case-insensitive passwords.

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. 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.

Warning! With Oracle Database 12c and later, password encrypted roles (such as BAN_DEFAULT_CONNECT) can no longer be assigned as default roles. See Default roles in Oracle database 12c and higher on page 11.

Note: The GSASECR page requires a default role. However, if an account is created outside of the GSASECR page 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.

Copy 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: section-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.

Delete a user account

You can delete a user account on the User window in GSASECR by selecting the user and clicking the **User: Delete** button.

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.

- FOBPROF: User Profile Table
- 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.
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 pages and their related tabs that are in use for section 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, business profiles, masking rules, PII rules, and FGAC rules that they were associated with the user are also deleted.

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 through 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 page of distributed security can be managed through the Banner Distributed Security (GSADSEC) page.

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 page.

- 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 page, 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 page, 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.

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 page 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.

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 page 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.

Section-level security for GSADSEC

Section Security can be enabled for the GSADSEC page. This new page 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 page, the user must either have direct access to the page, be enrolled in a class that includes the page, or be included in a security group that includes access to the GSADSEC page.

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

Internal Section Name	External Section 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

Internal Section Name	External Section Name	Access Restrictions	System Required
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 page lets you create a new distributed security user.

For more information, see Distributed security scripts on page 93.

The GSADSUM page has four tabs. The Distributed User Maintenance section lets you create and modify distributed security user accounts. The Distributed User Grants section lets you manage the rules that apply to creating distributed security user accounts. The Assignable Privileges section allows you to determine which system privileges are assignable to specific security users and groups. The User ID Restrictions section allows you to specify whether dropping, modifying, or querying specific users is restricted.

Distributed User Maintenance

This section is used to create and modify distributed security user accounts.

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

Note: You cannot maintain the `BANSECR` user account on this page. 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 section.
User Name	If the user ID exists, the name established on the GSASECR Banner Rules section will be displayed in the name field. If this is a new user then `*** NEW USER ***` will be displayed.

Field	Description
Password	The password can be any combination of characters identified as valid password `(PWD)` characters on the GSASECR institution profile section. 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 clear these 12 check boxes to add or change functions for this account. 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. After you create the account here, you must navigate to the GSASECR page 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.
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 section establishes the rules for each of the functions identified on the prior section. 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: You cannot grant access to the Oracle/Banner VPD Security Maintenance page (GSAVPDI) through this window. You must use the `gssvpdi.sql` script to grant access to GSAVPDI. See Scripts to manage a distributed security user on page 95.

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

The rules delivered by Ellucian are marked as System Required. Do not modify these rules.

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

You can use the following four dynamic parameters in this window's text fields:

- *<USER_NAME>* - replaced by the user name (for example, `BANSECR_xxx`)
- *<USER_ROLE_NAME>* - replaced by the role name (for example, *BAN_GSASECR_BANSECR_xxx*).
- *<USER_PASSWORD>* - replaced by the password from the page
- *<TEMP_TABLESPACE>* - replaced by the tablespace from the page

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 section.
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	Ellucian-delivered records have been marked as `System Required`. As a general rule, these records should not be changed without Ellucian 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.

Assignable Privileges

This section allows you to determine which system privileges are assignable to specific security users and groups.

Note: If a user has a specific privilege, regardless of any groups, then that specific user privilege value takes precedence. If a user has no specific privilege then, any group that the user is a member of that is assignable will prevent them from performing that function. If no rules exist at all, then the function may be performed.

Field	Description
System Privilege	The system privilege that may be specified as assignable to a specific security user or group.
Security User	The user or group to which the system privilege is assignable.
Assignable	Select the Assignable check box to mark this system privilege as assignable to a to the specified security user or group.
Comments	Comments on the system privilege. Optional.

User ID Restrictions

The User ID Restrictions section allows you to specify whether dropping, modifying, or querying specific users is restricted.

Note: When Banner checks for a restriction, specific Security User ID entries are checked first, then Security Groups in alphabetical order, then PUBLIC until a match is found. Only one entry is used to determine access.

Field	Description
Protected User	The user whose that may be protected by drop, modification, and query restrictions.
Security User	The user or group to which the protected user belongs.
Disallow Drop	Select the Disallow Drop check box to prevent the specified Protected User from being dropped.
Disallow Modify	Select the Disallow Modify check box to prevent the specified Protected User from being modified.
Disallow Query	Select the Disallow Query to prevent the specified Protected User from being queried.

Field	Description
Comments	Comments on the User ID. Optional.

Establish a new distributed security user

You can set up a new distributed security user using the GSADSUM page.

Procedure

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 page, User section, select the 'Banner Rules' option to assign the employee name to the new account and any approval information that may be desired.
4. 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 pages (`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 page, 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. Use GSADSEC to establish privileges and assignable privileges for the user for classes, objects, and roles.

Results

You can also create a distributed security user using scripts. For details, see page Scripts to manage a distributed security user on page 95.

Banner Distributed Security (GSADSEC)

The GSADSEC page 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 page to set up security groups and logon calendars.

Distributed Groups

You can use this section 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

This window is used to designate proxied owners for distributed security groups, and to assign specific privileges to each proxied owner.

See Owners and privileges on page 77 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 86.

To change a distributed group's primary owner, you must edit the group in the Distributed Groups window. See Distributed Groups on page 85.

Object Owners

This window is used to designate proxied owners for Banner objects, and to assign specific privileges to each proxied owner.

See Owners and privileges on page 77 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 page. See Objects on page 60.

The fields in the Object Owners, Class Owners, Role Owners, and Distributed Group Owners windows are nearly identical, and the Security Group Owners section 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. There is no corresponding code field on the Security Group Owners section.
System	One-letter code for the Banner product associated with the class. This field appears only on the Class Owners window.
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 page. 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. See Owners and privileges on page 77 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 page. To change the comments, you must edit the object, class, or role in GSASECR, or edit the group in its section 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. 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

This window is used to designate proxied owners for security classes, and to assign specific privileges to each proxied owner.

See Owners and privileges on page 77 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 86.

To change a class's primary owner, you must edit the class in the GSASECR page. See Classes on page 57.

Role Owners

This window is used to designate proxied owners for security roles, and to assign specific privileges to each proxied owner.

See Owners and privileges on page 77 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 86.

To change a role’s primary owner, you must edit the role in the GSASECR page. See Roles on page 62.

Security Groups

This window allows you to 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 allows you to associate security classes, Banner objects, and user IDs with a security group. You can also assign proxied owners for the security group.

Classes

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

Objects

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

Note: Section-level security is not available for objects directly assigned to a group. In order to provide section-level security through a group, the section 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 section privileges through the group. Alternately, section privileges can be assigned directly to an individual user in GSASECR.

Users

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

See Owners and privileges on page 77 for an explanation of owners, proxied owners, and each of the privileges that can be assigned.

For the list of fields in this section, see Object Owners on page 86.

To change a group’s primary owner, you must edit the group in the Security Groups window. See Security Groups on page 88.

Priority of security rules

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

The following are the 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 section 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 allows you to 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 50.

Field	Description
Calendar Code	A unique identifier for the calendar.
Description	A description of the calendar.

Field	Description
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. **Note:** 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.

Set 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 Banner Administrative access. See Setup logon rules for a user on page 50.

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

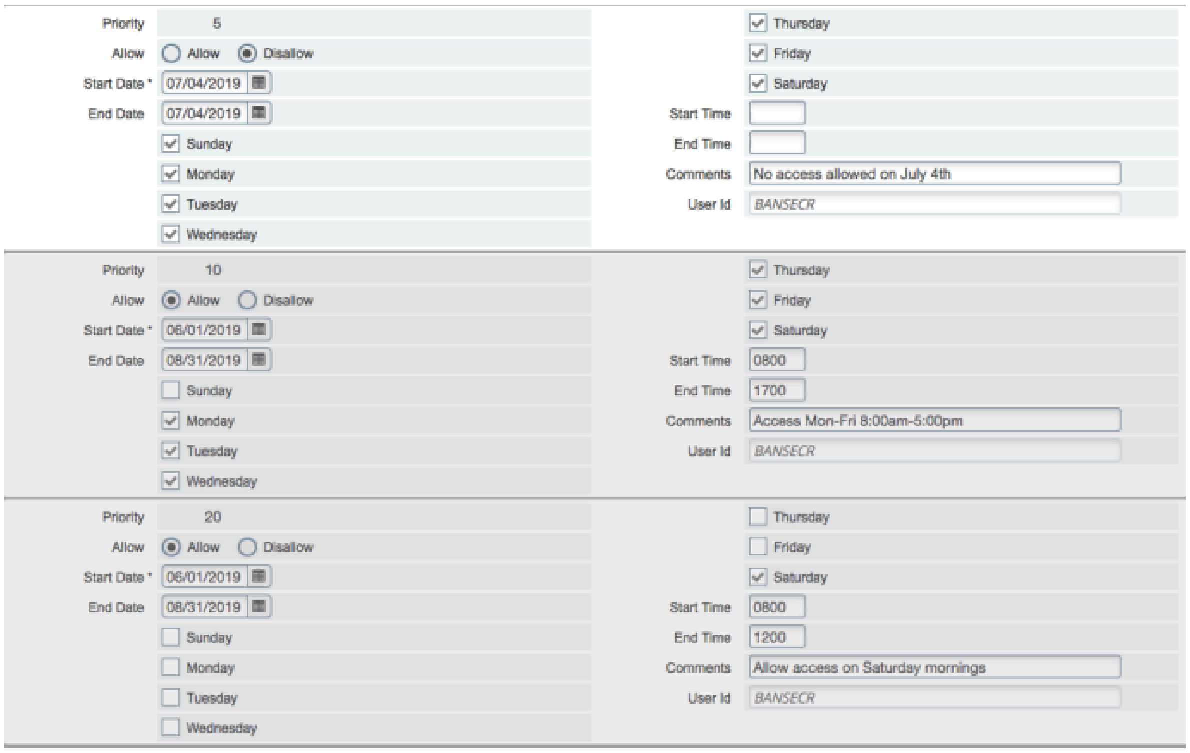
- a daily cycle—you can enforce a beginning and ending clock time
- a weekly cycle—you can select days of the week
- long-term—you can set a beginning date and (optional) ending date

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.



Priority	5		✓ Thursday
Allow	Allow [✓ Disallow]		✓ Friday
Start Date *	07/04/2019		✓ Saturday
End Date	07/04/2019	Start Time	
	✓ Sunday	End Time	
	✓ Monday	Comments	No access allowed on July 4th
	✓ Tuesday	User Id	BANSECR
	✓ Wednesday		

Priority	10		✓ Thursday
Allow	[✓ Allow] Disallow		✓ Friday
Start Date *	06/01/2019		✓ Saturday
End Date	08/31/2019	Start Time	0800
	Sunday	End Time	1700
	✓ Monday	Comments	Access Mon-Fri 8:00am-5:00pm
	✓ Tuesday	User Id	BANSECR
	✓ Wednesday		

Priority	20		Thursday
Allow	[✓ Allow] Disallow		Friday
Start Date *	06/01/2019		✓ Saturday
End Date	08/31/2019	Start Time	0800
	Sunday	End Time	1200
	Monday	Comments	Allow access on Saturday mornings
	Tuesday	User Id	BANSECR
	Wednesday		

Review distributed security permissions

You can review the distributed security permissions by using the `gchkgrants.sql` script and the Distributed Security Object Ownership View (GUVOWNR).

Review and update grants with the gchkgrants.sql script

The `gchkgrants.sql` script 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 allows you to 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 through a group, then the ownership will be identified as Owner Group or Proxy Group.
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

The Distributed Security User Maintenance (GSADSUM) page provides an easy way to set up new distributed users, and the Distributed Security (GSADSEC) page enables a finer level of control of objects that the distributed security users can modify.

See the following for more information:

- Distributed Security User Maintenance (GSADSUM) on page 80
- Banner Distributed Security (GSADSEC) on page 85

Privileges

Banner role security is managed in the GSASECR page. The GSASECR page 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 Gsasecr_TaskStartedPre() method checks to see if the user running the page 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 page execution stops. Additionally, if the user is BANSECR, a Dynamic SQL History section is made visible and navigable. This section shows the list of all commands executed by the various BANSECR users from within the page sorted in descending order by date and time.

If the page is being run from any other legitimate account (a BANSECR`_xxx` account), the page 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
UPDATE on BANSECR.GURUOBJ	Class Maintenance
UPDATE on BANSECR.GURUOBJ	User Maintenance

Scripts to manage a distributed security user

To help you manage distributed security accounts, several scripts have been delivered in the GENERAL directory tree.

Script	Description
gsssels.sql	Sample row level security for security maintenance.
gssvpdi.sql	Access to the Oracle/Banner VPD Security Maintenance (GSAVPDI) page.

Oracle Distributed Security Password Reset (GSAPASR)

The Oracle Distributed Security Password Reset page (GSAPASR) enables a distributed security user the ability to reset a user's password. If the distributed security user also has ALTER USER, then the user can expire, lock, and unlock the account as well.

This page enables a distributed security user to reset passwords without requiring access to GSASECR.

Field	Description
Oracle User ID	The unique code that identifies the distributed security group.
Database	A description of the distributed security group.
New Oracle Password	The user ID, distributed security group, or PUBLIC assigned as the owner of this distributed security group.
Verify Password	The ID of the user that created or most recently changed the distributed security group.
Password Expires	The date of the latest change.
Oracle Account Status	The ID of the user that created or most recently changed the distributed security group.
Locked Date	The date of the latest change.
Expire Password	The user ID, distributed security group, or PUBLIC assigned as the owner of this distributed security group.
Lock	The ID of the user that created or most recently changed the distributed security group.
Unlock	The date of the latest change.
Save Changes	The ID of the user that created or most recently changed the distributed security group.
Exit Without Changing Password	The date of the latest change.

Note: In addition, you can use the User ID Restrictions section on GSADSUM to limit access to specific accounts so that the passwords may not be changed. You cannot change passwords for accounts that have a default tablespace of SYSTEM or SYSAUX (typically system type accounts). Password changes are also restricted on Oracle predefined administrative or non-administrative accounts.

Related information

https://docs.oracle.com/database/121/TDPSG/GUID-3EC7A894-D620-4497-AFB1-64EB8C33D854.htm#TDPSG20030

G$_SECURITY.g$_change_password

G$_SECURITY.g$_change_password was intended to be called from a security page, such as GSASECR, GSAPASR, and GUAPSWD. The user must have either DBA, ALTER USER, or access to the GSAPASR page to execute this routine.

However, if the routine is called directly, the following checks and error messages are in place:

- Username and password must be supplied.

  ERROR The username and password cannot be null.
- The user being modified has a valid username using DBMS_ASSERT.

  ERROR The user that you are trying to modify does not exist.
- There are no Banner restrictions on modifying the user based on GURUSRP entries on GSADSUM.

  ERROR You are not authorized to modify this user per GSADSUM User ID Restrictions.
- There are no $ or any other not valid character in GJRINVC, as defined on the Institution Profile section on GSASECR, for the password. \

  ERROR Unsupported character detected in the password.
- The password of a system account cannot be changed as determined by default tablespace = SYSTEM or SYSAUX.

  ERROR You are not authorized to modify this user since default tablespace is SYSTEM or SYSAUX.

- The password of an Oracle predefined administrative or non-administrative account cannot be changed.

  ERROR You are not authorized to modify this user since account is Oracle installed account name.

  Note: For more information, please refer to the following: https://docs.oracle.com/en/database/oracle/oracle-database/12.2/tdpsg/securing-the-database-installation-and-configuration.html#GUID-4F79E765-AF6C-4B39-A2DD-CA78A8892DE1

  Note: You may add any accounts that should have restricted access on the User ID Restrictions section on GSADSUM.

  These accounts include the following:

  – ANONYMOUS
  – CTXSYS
  – DBSNMP
  – EXFSYS
  – LBACSYS
  – MDSYS
  – MGMT_VIEW
  – OLAPSYS
  – OWBSYS
  – ORDPLUGINS
  – ORDSYS
  – OUTLN
  – SI_INFORMTN_SCHEMA
  – SYS
  – SYSMAN
  – SYSTEM
  – TSMSYS
  – WK_TEST
  – WKSYS
  – WKPROXY
  – WMSYS
  – XDB
  – APEX_PUBLIC_USER
  – DIP
  – FLOWS_30000
  – FLOWS_FILES

- MDDATA
- ORACLE_OCM
- SPATIAL_CSW_ADMIN_USR
- SPATIAL_WFS_ADMIN_USR
- XS$NULL

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 (GSAAUDT) page.

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 65 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.
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.

Audit Table	Base Table	Trigger	Type of Information
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)	Pages and tabs that can be used in section 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.
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.

Audit Table	Base Table	Trigger	Type of Information
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 section 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 General 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.

Audit Table	Base Table	Navigation Path in GSAAUDT	Type of Information
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 > Section Security	User section security access.
GURAUOB	GURUOBJ	User Security > Object Access	Type of access, by user ID, for each Banner object.
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 > Section Security	Pages and tabs that can be used in section 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.

Audit Table	Base Table	Navigation Path in GSAAUDT	Type of Information
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.
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

The GURALGN table records all Banner Administrative access. In addition, if you have the `GT_LOGIN_AUDIT_ACCESS` or `GT_LOGOFF_AUDIT_ACCESS` triggers enabled, then standard Oracle logins by Banner users will also be logged in the table.

To determine which logins belong to Banner General users (as distinguished from Oracle user IDs that do not have Banner General security access) the user ID validates against the following tables.

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

If the user ID exists in any of these tables, the application records the Oracle login in the GURALGN table and displays it on the Logon/Logoff Activity section of the GSAAUDT page.

You can enable or disable tracking, or set tracking to audit selective pages and jobs by setting an option on the Institution Profile tab.

Page and job activity audits

GSASECR can log successful object access in addition to any violations. This allows you to track objects used on a daily basis.

You can enable or disable tracking, or set tracking to audit selective pages and jobs by setting an option on the Institution Profile tab.

The BAN_AUDIT_USAGE object, if associated with a user (through direct, class, or group) tracks all usage for that user, regardless of any other settings that enable auditing on a user by user basis.

Selecting the **Track Access** check box on the Objects tab enables you to select specific objects that you want to track.

Initial delivery of this indicator sets all pages and jobs, as identified on the Object Maintenance (GUAOBJS) page to Y. If an entry does not exist on GUAOBJS, the indicator is set to N.

Note: You can find the logic for form and job tracking in `g$_security_pkg.g$_verify_password2_prd` and `g$_verify_password2_prd_grails` (`gspsecr.sql`), which means that to track a form/job, it must go through the standard security checking. If it does not, tracking will not occur unless you modify the object to directly call the new tracking process `g$_security_pkg.p_track_access`.

The following pages/jobs are not audited regardless of system or object settings:

- %OQMENU (application menus)
- %OADEST (printer destination settings)
- GUAGMNU (main menu)
- GJAPCTL (job submission setup)
- GUQINTF (internal form associated with job submission)
- GURINSO (job to copy an output file to the database for use by GJIREVO)

- GUAERRM (form to display error messages)
- GUACALN (calendar pop-up)

Banner Self-Service page and login audit

Banner provides ways to track and capture page access and login/logout activity across all Banner Self-Service applications.

Page audit

Banner captures key information in the Banner 9 page Access (GURASSA) table.

Banner captures the following information in the GURASSA table:

- Audit Time
- SSB Login ID
- Application ID
- Page URL
- IP Address
- Activity date
- User ID
- PIDM

Note: The following items describe some details of key information captured.

- Access to an unsecured page of a Self-Service application updates the audit record with ANONYMOUS as the SSB Login ID, BANNER as the User ID, and null for PIDM.
- Guest user access to Self-Service applications sets the Login ID and User ID to the email address and PIDM to null.
- The Page URL format does not include the http/https protocol and hostname or IP address.
- The Page URL contains the URL parameters.

Configurations to enable page audit

The default configuration initially delivered by Ellucian for page level auditing is at the GLOBAL application level.

The Banner Applications Configurations (GUACONF) page displays this parameter in the configurations section. You can configure the EnablePageAudit parameter at the application level by copying it to individual Self-Service applications.

Table 1: Configuration details

App Id	GUACONF Key	Value	Result
GLOBAL	EnablePageAudit	N or null	No pages audited
		%	Audit all pages across all Self-Service applications.
		String or page name	Audit pages that contain a string. For example, specify a value of `hrDashboard` to audit all pages that contain hrDrashboard in the URL.
		Comma separated string or page name	Audit pages containing multiple strings. For example, classRegistration,studentProfile

Page audit configurations made at the application level override any GLOBAL page audit configurations.

Login audit

Banner captures the following key information in the Banner 9 Self Service Logins (GURASSL) table.

- Audit Time
- SSB Login ID
- Application ID
- User Agent
- Activity date
- User ID
- PIDM
- Logon comment

Note: The following items describe some behavior of audit tracking.

- The application audits successful login/logout in the GURASSL table.
- The application audits Single Sign-On (SSO) login/logout. The Login ID is the Spriden ID for Single Sign-On.
- Guest user access to Self-Service applications, sets the Login ID and User ID to the email address.

Configurations to enable login audit

The default configuration initially delivered by Ellucian for login level auditing is at the GLOBAL application level.

The Banner Applications Configurations (GUACONF) page displays this parameter in the configurations section. You can configure the EnableLoginAudit parameter at the application level by copying it to individual Self-Service applications.

Table 2: Configuration details

App Id	GUACONF Key	Value	Result
GLOBAL	EnableLoginAudit	N or null	No login/logout information audited.
		Y	Audit all login/logout information across all Self-Service applications.

Page audit configurations made at the application level override any GLOBAL page audit configurations.

Additional privacy configuration

To add an additional level of privacy, you can configure the AuditIPAddress parameter which allows you to mask, or disable tracking the IP address of the user.

The Banner Applications Configurations (GUACONF) page displays this parameter in the configurations section.

The configuration shown in the following table applies for both Page level auditing (GURASSA) and Login auditing (GURASSL).

Table 3: Configuration details

App Id	GUACONF Key	Value	Result
GLOBAL	AuditIPAddress	Y	IP address audited.
		N	IP address not audited.
		M	Mask the digits after the last period (.) of the IP address.

Banner Security Table Audits (GSAAUDT)

The Banner Security Table Audits page (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 page uses a two-level section structure to make room for the large number of audit tables. See Banner Security Table Audits (GSAAUDT) on page 108 for a list of the security audit tables and the GSAAUDT navigation for each.

You can use Banner's standard filter 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.

Turn auditing on or off

Each of GSAAUDT's tabs displays audit records from a specific table. A message at the bottom of each section 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 66.

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.

Delete 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 section.

About this task

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:

Procedure

1. Navigate to the appropriate section 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 page will typically have inquiry-only access.

3. Select a time period.

   - `Before today`: All records for this table with a date before the system date will be deleted from the history table.
   - `Older than one month`: All records for this table with a date before one month ago will be deleted from the history table.
   - `Older than one year`: All records for this table with a date before one year ago will be deleted from the history table.
   - `Prior to a specific date`: All records before 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 72.

   Warning! If you initiate the delete process after doing a filter, then only records that were included in the filter results (and that match the date selection that you made) will be deleted. Records that were excluded from the filter 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 filters in a section before you begin the delete process.

Mask rules in Banner 9.x

The following fields are available for use with masking in Banner 9.x.

Field	Description
Banner9 Object Name	Banner9 object name for the selected section or table. The primary use of this field was for Banner 9x ZK applications. With the support of Banner 9x ZK applications ending, Ellucian Ethos is now making use of this field for the following: Model name. For example, persons.
Banner9 Block ID	Banner9 Block ID for the selected section or table. This information can be attained through Banner9 Item Properties. The primary use of this field was for Banner 9x ZK applications. With the support of Banner 9x ZK applications ending, Ellucian Ethos is now making use of this field for the following: Full path of field name in JSON. For example, grade or credentials.value.

Field	Description
Banner9 Item ID	Banner9 Item ID for the selected section or table. This information can be attained through Banner9 Item Properties. The primary use of this field was for Banner 9x ZK applications. With the support of Banner 9x ZK applications ending, Ellucian Ethos is now making use of this field for the following: Specify with an asterick (*) or any non-null value. This field is form required, but not used for filtering.

Note: These fields are only for use with Banner 9.x.

Security with shared connections

Connections to the Banner database can use either the end-user authenticated connections used by Oracle Forms and C programs, or the shared connections used by Self Service, channels, and messaging.

Because 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

Ellucian provides support for Lightweight Directory Access Protocol (LDAP) to perform user authentication for Banner Self-Service. Ellucian 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.

The Self-Service products 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:

- The DAD without the user ID and password is the existing DAD. It is still used for standard Self-Service logins.
- The DAD with the user ID and password 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 by 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 users to enter their Oracle user ID and password. The end users do not need to enter it again during the session unless they log 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.

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;
```

After 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`, and 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.

Because 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.

Table 4: GSPPRXY input parameters

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, For example, 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.

GSPPRXY output parameters

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 clients 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 pages and tables), the password for the role, and the internal Oracle 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.

After 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

During security setup, all users or classes that need access to channels must have the CHANNEL object added to their privileges.

Set up a user for a proxy connection

You can use the GSASECR security setup page, to enable a user to use a proxy connection.

Procedure

1. Access the GSASESCR security setup page.
2. On the Users section, select **Authorize BANPROXY**.
3. Alter the user with this command: `ALTER USER` *username* `GRANT CONNECT THROUGH BANPROXY;`

Map 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 by 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

You can use Banner classes to map a default Oracle user ID for particular connection types.

When 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 seven 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 the "ID Mappings Between Systems" topic in the "Implement Single Sign-On (SSO)," section of the *Middle Tier Implementation Guide*.

Oracle*Wallet Proxy for Job Submission

This supports communicating with Job Submission through a proxy user with an Oracle*Wallet supporting configuration on the Job Submission server.

Essentially, this modification supports running batch jobs without requiring to pass the user's password value to the Job Submission server.

When a user is connected to the transformed Banner application, that user is commonly connected by a proxy connection. The user's password will not be known to pass it to the Job Submission server.

Establish proxy wallet for job submission

You need to have certain minimum requirements to be able to use Proxy Wallet for Job Submission.

- Assign a database connection identifier for the Proxy Job Submission SID

  Note: See the Security Maintenance (GSASECR) institution Profile window. This value needs to match the new alias created in the tnsnames.ora described in Modify tnsnames.ora on page 117.

- Assign user/users of Job Submission to be authorized for BANJSPROXY.

  Note: See the Security Maintenance (GSASECR) User Section/Alter or Create an Oracle User ID window.

- Establish the Oracle*Wallet supporting configuration on the Job Submission server.

Oracle*wallet supporting configuration on the Job submission server

You must setup a Secure External Password Store using Oracle*Wallet with a proxy user's credentials so that a password is not needed to log in to the database when running batch jobs.

Note: These steps are a basic example of getting this solution running in a Linux environment. The solution does not rely on Linux specific utilities, so it can be used to implement this solution in other operating systems using the analogous command.

Single instance setup

You can set up Oracle*Wallet in a single instance environment. This process involves creating a wallet, setting up tnsnames.ora and sqlnet.ora setup, and storing the credentials in the wallet.

Create a wallet

This process creates an empty wallet with the necessary file permissions. Only the Operating System (OS) user that is used to run Job Submission process (gurjobs) has access to them.
Create directory to store the wallet files
You have to create a directory to store the wallet files. For example, assume the Job Submission OS user is banjobs.

Procedure

In the `banjobs` home directory, enter the command: `[ ~ ]$ Dirk proxy_wallet`.

Create a wallet in the new directory
You can now create the wallet in the new directory. You can use the `mkstore` command to interact with the wallet.

Procedure

1. To create the wallet, call the command with the `-create` option as `mkstore -wrl <wallet directory> -create`.
2. Enter a password when prompted.

   You will use this password for all future interactions with the wallet. The commands look like the following if you continue with the banjobs example.

   ```
   [ ~ ]$ cd proxy_wallet
   [ ~/proxy_wallet ]$ mkstore -wrl . -create
   <enter and confirm your new password>
   ```

Verify that the wallet files were created correctly
You have to check and verify that the wallet files were created correctly. The wallet files should only be accessible to the Job Submission user, who is banjobs in this example.

Procedure

1. Use the `ls` command to check that the wallet files were created and that they have the correct file permissions.

   ```
   [ ~/proxy_wallet ]$ ls -l
   Total 16
   -rw------- 1 banjobs users 3000 Feb 27 07:46 cwallet.sso
   -rw------- 1 banjobs users 3000 Feb 27 07:46 ewallet.p12
   ```

2. If the `ls` command indicated that other users had access to the wallet files, then use `chmod` to set them appropriately.

   ```
   [ ~/proxy_wallet ]$ chmod 600 ./*
   ```

   You can now set up the `tnsnames.ora` and `sqlnet.ora` files.

Set up tnsnames.ora and sqlnet.ora

Setting up the `tnsnames.ora` and `sqlnet.ora` involves three steps.
Copy tnsnames.ora and sqlnet.ora to a new location
The `tnsnames.ora` and `sqlnet.ora` files need to be copied to a different location so that the Proxy Wallet will only be used with Job Submission and the current functionality is preserved. Having the `WALLET_OVERRIDE in sqlnet.ora` set to TRUE can interfere with other Oracle tools.

Procedure

1. Create a new directory to store the `tnsnames.ora` and `sqlnet.ora` files that will be modified.
2. Copy the `tnsnames.ora` and `sqlnet.ora` files that are currently being used to this new directory.

   The `TNS_ADMIN` environment variable will need to be set to this location for the shell that will execute GURJOBS.

   ```
   [ ~ ]$ cd $ORACLE_HOME/network/admin
   [ /u01/app/oracle/ORDBMS/11.2.0.2/network/admin ]$ mkdir
   proxy_setup
   [ /u01/app/oracle/ORDBMS/11.2.0.2/network/admin ]$cp
   tnsnames.ora proxy_setup/tnsnames.ora
   [ /u01/app/oracle/ORDBMS/11.2.0.2/network/admin ]$cp
   sqlnet.ora proxy_setup/sqlnet.ora
   ```

Modify tnsnames.ora
You have to modify the `tnsnames.ora` file on the Job Submission server so the Oracle Net client knows where to look for the wallet. The default location for these files is `$ORACLE_HOME/network/admin`.

Procedure

1. Open the new `tnsnames.ora` file.

   This file contains entries that describe connections to your database. The original TNS entries should remain in both `tnsnames.ora` files.
2. Add a new entry as a copy of the one that is normally used to connect to the database and rename it.

   This alias will be used to create the wallet credentials and must be specified in the Proxy Job Submission SID on the GSASECR page. In the following `tnsnames.ora` example, SMPL is what is normally used to connect to the database, and JSUB was added as an exact copy:

   ```
   SMPL =
          (DESCRIPTION =
              (ADDRESS_LIST =
                (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL) (PORT =
   ```

```
1521))
    )
    (CONNECT_DATA =
      (SID = SMPL)
      (SERVICE_NAME = WWW.YOURINSTITUION.COM)
      (SERVER = DEDICATED)
   )
 )
JSUB =
  (DESCRIPTION =
    (ADDRESS_LIST =
     (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL) (PORT =
1521))
    )
    (CONNECT_DATA =
    (SID = SMPL)
    (SERVICE_NAME = WWW.YOURINSTITUION.COM)
    (SERVER = DEDICATED)
  )
 )
```

Modify sqlnet.ora
You have to modify the `sqlnet.ora` file on the Job Submission server so the Oracle Net client knows where to look for the wallet. The default location for these files is `$ORACLE_HOME/network/admin`.

Procedure

1. Open the new `sqlnet.ora` file.
2. Add entries to this file to specify the wallet location and force the wallet to be used to authenticate users to the database.

   The directory entry below should be the directory where the wallet was created. In this example,`/u03/banjobs/proxy_wallet`.

```
WALLET_LOCATION =
 (SOURCE = (METHOD = FILE)
  (METHOD_DATA =
   (DIRECTORY=/u03/banjobs/proxy_wallet)))
SQLNET.WALLET_OVERRIDE = TRUE
SSL_CLIENT_AUTHENTICATION = FALSE
```

Store credentials in wallet

This process creates a user that all Banner Job Submission users will use as a proxy to connect to the database. This user's credentials will be stored in the wallet. The banjsproxy user will already have been created during the Banner General 8.8 upgrade process by running the `gcreate_banjsproxy.sql` script.

About this task

Note: The wallet files will only be accessible to the Job Submission users.

Store the proxy user's credentials
After creating the proxy user, you have to store the proxy user's credentials in the wallet.

Procedure

1. Use the `mkstore` command to store the proxy user's credentials in the wallet.

   The `mkstore` command format must be:

   `mks tore -wrl` *`<wallet directory>`* `-createCredential` *`<alias from tsnnames.ora> <username> <password>`*
2. Enter the password when prompted.

```
[ ~/proxy_wallet ]$ mkstore -wrl . -createCredential JSUB
BANJSPROXY <password>
Enter password:
Create credential oracle.security.client.connect_string1
```

Test wallet credentials
After creating the proxy user and storing the proxy user's credentials in the wallet, you have to test and verify that the wallet files were created correctly.

Procedure

1. When testing, make sure the *TNS_ADMIN* variable is set to the location of the new `tnsnames.ora` and `sqlnet.ora` files.
2. Use the `tnsping` command to test that the alias created in the `tnsnames.ora` file is able to connect to the database.
3. Try to connect to sqlplus using the wallet syntax, `sqlplus [<authorized user>] / @<alias from tnsnames.ora>`.

   You will need to be logged in to the Job Submission OS user to make use of the wallet, and a user will need to be given permission to connect through BANJSPROXY.

```
[ ~/proxy_wallet ]$ export
TNS_ADMIN=/u01/app/oracle/ORDBMS/11.2.0.2/network/admin/proxy_setup/
[ ~/proxy_wallet ]$ tnsping jsub


TNS Ping Utility for Linux: Version 11.2.0.2.0 - Production on 02-
APR-2015 10:48:46

Copyright (c) 1997, 2010, Oracle.  All rights reserved.

Used parameter files:
/u01/app/oracle/ORDBMS/11.2.0.2/network/admin/proxy_setup/sqlnet.ora


Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS =
 (PROTOCOL = TCP)(HOST = DBSMPL) (PORT = 1521))) (CONNECT_DATA =
 (SID = SMPL) (SERVICE_NAME = WWW.YOURINSTITUTION.COM) (SERVER =
 DEDICATED)))
OK (10 msec)
[ ~/proxy_wallet ]$ sqlplus baninst1/<password>
```

```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Apr 2 10:56:03 2015

Copyright (c) 1982, 2010, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit
 Production
With the Partitioning, OLAP, Data Mining and Real Application
 Testing options

SQL> alter user saisusr grant connect through banjsproxy;

User altered.

SQL> exit
Disconnected from Oracle Database 11g Enterprise Edition Release
 11.2.0.2.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application
 Testing
options

[ ~/proxy_wallet ]$ sqlplus [saisusr]/@jsub
SQL*Plus: Release 11.2.0.2.0 Production on Thu Apr 2 10:56:03 2015

Copyright (c) 1982, 2010, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit
 Production
With the Partitioning, OLAP, Data Mining and Real Application
 Testing options

SQL> show user
USER is "SAISUSR"
```

RAC setup

There are a couple of different approaches for setting up the wallet in a RAC environment with multiple job submission servers. The one best suited for your environment depends on how that RAC environment is used and implemented.

This step will set the wallet up in a shared location that all nodes will use. An alternative is to install the wallet on each node. This can be done by repeating the Single Instance steps for each node and making sure to use TNS alias/service associated with the RAC, as opposed to the instance, for the wallet's connection string.

The wallet and SQL*Net files should be set up in a shared location so all the instances can see the wallet. This shared location could be a sub-directory of the Grid home or some other networked file system.

Create a wallet

Create an empty wallet with the necessary file permissions. Only the OS users that run Job Submission process (gurjobs) should have access to them. These users should belong to their own group so that all other users cannot access the wallet files.
Create directory to store wallet files
You have to create a directory to store the wallet files. For example, assume the Job Submission OS user is banjobs and the user group jobsub has been created for all the Job Submission OS users.

Procedure

1. As banjobs, navigate to the shared location.
2. In this directory, enter the command: `[ /s01/app/jsub ]$ mkdir proxy_wallet`.

Create wallet in the new directory
You can now create the wallet in the new directory. You can use the `mkstore` command to interact with the wallet.

Procedure

1. To create the wallet, call the command with the `-create` option as :`mkstore -wrl <wallet directory> -create`.
2. Enter a password when prompted.

   This password will be used for all future interactions with the wallet.

```
[ ~ ]$ cd proxy_wallet
[ ~/proxy_wallet ]$ mkstore -wrl . -create
<enter and confirm your new password>
```

Verify that the wallet files were created correctly
You have to check and verify that the wallet files were created correctly. The wallet files should only be accessible to the Job Submission user, who is banjobs in this example.

Procedure

1. Use the `ls` command to check that the wallet files were created and that they have the correct file permissions.

```
[ /s01/app/jsub/proxy_wallet ]$ ls -l
Total 16
-rw-r----- 1 banjobs jobsub 3000 Feb 27 07:46 cwallet.sso
Banner General Security Administration Handbook | Security with
 Shared Connections 127
-rw-r----- 1 banjobs jobsub 3000 Feb 27 07:46 ewallet.p12
```

2. If the `ls` command indicated that other users had access to the wallet files, then use `chmod` to set them appropriately.

   `[[ /s01/app/jsub/proxy_wallet ]$ chmod 640 ./*`

   You can now set up the `tnsnames.ora` and `sqlnet.ora` files.

Set up tnsnames.ora and sqlnet.ora

Setting up the `tnsnames.ora` and `sqlnet.ora` involves three steps.
Copy tnsnames.ora and sqlnet.ora to a new location
The `tnsnames.ora` and `sqlnet.ora` files need to be copied to a different location so that the Proxy Wallet will only be used with Job Submission and the current functionality is preserved. Having the `WALLET_OVERRIDE in sqlnet.ora` set to TRUE can interfere with other Oracle tools.

Procedure

1. Create a new directory to store the `tnsnames.ora` and `sqlnet.ora` files that will be modified.
2. Copy the `tnsnames.ora` and `sqlnet.ora` files that are currently being used to this new directory.

   The `TNS_ADMIN` environment variable will need to be set to this location for the shell that will execute GURJOBS.

   ```
   [ ~ ]$ cd /s01/app/jsub
   [ /s01/app/jsub ]$ mkdir proxy_setup
   [ /s01/app/jsub ]$ cp $ORA_CRS_HOME/network/admin/
   tnsnames.ora proxy_setup/tnsnames.ora
   [ /s01/app/jsub ]$ cp $ORA_CRS_HOME/network/admin/
   sqlnet.ora proxy_setup/sqlnet.ora
   ```

Modify tnsnames.ora
You have to modify the `tnsnames.ora` file on the Job Submission server so the Oracle Net client knows where to look for the wallet. The default location for these files is `$ORACLE_HOME/network/admin`.

Procedure

1. Open the new `tnsnames.ora` file.

   This file contains entries that describe connections to your database. The original TNS entries should remain in both `tnsnames.ora` files.
2. Add a new entry as a copy of the one that is normally used to connect to the database and rename it.

   This alias will be used to create the wallet credentials and must be specified in the Proxy Job Submission SID on the GSASECR page. In the following `tnsnames.ora` example, SMPLRAC is what is normally used to connect to the RAC, and SMPLRAC_JSUB was added as an exact copy:

   ```
   LISTENERS_DBSMPL =
     (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL1)(PORT =
   1521))
      (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL2)(PORT =
   1521))
     )
    LISTENER_DBSMPL1 =
      (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL1)(PORT =
   1521))
   ```

```
 LISTENER_DBSMPL2 =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL2)(PORT =
1521))
SMPLRAC2 =
  (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL2)(PORT =
1521))
    (CONNECT_DATA =
     (SERVER = DEDICATED)
     (SERVICE_NAME = www.yourinstitution.com)
     (INSTANCE_NAME = SMPLRAC2)
    )
  )
SMPLRAC1 =
  (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL1)(PORT =
1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = www.yourinstitution.com)
      (INSTANCE_NAME = SMPLRAC1)
  )
 )
SMPLRAC =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL)(PORT =
1521))
    (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = www.yourinstitution.com)
)
)
SMPLRAC_JSUB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = DBSMPL)(PORT =
1521))
    (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = www.yourinstitution.com)
  )
 )
```

Modify sqlnet.ora
You have to modify the `sqlnet.ora` file on the Job Submission server so the Oracle Net client knows where to look for the wallet. The default location for these files is `$ORACLE_HOME/network/admin`.

Procedure

1. Open the new `sqlnet.ora` file.
2. Add entries to this file to specify the wallet location and force the wallet to be used to authenticate users to the database.

   The directory entry below should be the directory where the wallet was created. In this example,`/u03/banjobs/proxy_wallet`.

```
WALLET_LOCATION =
    (SOURCE = (METHOD = FILE)
     (METHOD_DATA =
      (DIRECTORY=/s01/app/jsub/proxy_wallet)))
SQLNET.WALLET_OVERRIDE = TRUE
SSL_CLIENT_AUTHENTICATION = FALSE
```

Store credentials in wallet

This process creates a user that all Banner Job Submission users will use as a proxy to connect to the database. This user's credentials will be stored in the wallet. The banjsproxy user will already have been created during the Banner General 8.8 upgrade process by running the `gcreate_banjsproxy.sql` script.

About this task

Note: The wallet files will only be accessible to the Job Submission user.

Store the proxy user's credentials
After creating the proxy user, you have to store the proxy user's credentials in the wallet.

Procedure

1. Use the `mkstore` command to store the proxy user's credentials in the wallet.

   The `mkstore` command format must be:

   `mks tore -wrl` *`<wallet directory>`* `-createCredential` *`<alias from tsnnames.ora> <username> <password>`*
2. Enter the password when prompted.

   ```
   [ /s01/app/jsub/proxy_wallet ]$ mkstore -wrl . -
   createCredential SMPLRAC_JSUB BANJSPROXY <password>
   Enter password:
   Create credential oracle.security.client.connect_string1
   ```

Test wallet credentials
After creating the proxy user and storing the proxy user's credentials in the wallet, you have to test and verify that the wallet files were created correctly. This can be tested in the same way as for the single instance demonstrated previously.

Procedure

1. When testing, make sure the *TNS_ADMIN* variable is set to the location of the new `tnsnames.ora` and `sqlnet.ora` files.
2. Use the `tnsping` command to test that the alias created in the `tnsnames.ora` file is able to connect to the database.
3. Try to connect to sqlplus using the wallet syntax, `sqlplus [<authorized user>] /@<alias from tnsnames.ora>`.

   You will need to be logged in as one of Job Submission OS users to make use of the wallet, and a user will need to be given permission to connect through BANJSPROXY.

Enable Job Submission proxy wallet

To enable users to use the wallet, the alias that was added to `tnsnames.ora` needs input into Banner.

Procedure

1. Navigate to the GSASECR page.
2. Under the Institution Profile section, enter the alias in the **Proxy Job Submission SID** text field.

   Each user who wants to use Job Submission Proxy should have the **Authorize BANJSPROXY** check box selected.

   The `gbanjsproxy_users.sql` script creates a script that authorizes all users except the schema owners to use the proxy. You can edit the script before running it to add or remove additional users.

Confirm Job Submission allows brackets
You have to confirm that Job submission allows the "[" and "]" characters.

Procedure

1. Run the `gchkjobsubproxy.sql` script to confirm JOBSUB is allowing the passing of the "[" and "]" characters.

   ```
   cd $BANNER_HOME/general/plus
   sqlplus system/manager @gchkjobsubproxy.sql
   ```

   It should state the following:

   ```
   *********************************************************
   *****
   SUCCESS: JOB SUBMISSION 'VALIDATION CHARACTERS' SET IS
            COMPATIBLE WITH JOB SUBMISSION USER PROXY.
   *********************************************************
   ****
   ```

   If it states the following, you need to fix the Validation Characters on the GSASECR page.

   ```
   WARNING: THE CHARACTERS '[' AND ']' ARE INCOMPATIBLE WITH
   THE JOB SUBMISSION
           USER PROXY. IF PRESENT IN THE JOB SUBMISSION
   'VALIDATION CHARACTERS'
           SET, THEY WILL PREVENT THE PROXY FROM WORKING.
           ONE OR MORE OF THESE WERE FOUND IN THE CURRENT
   VALIDATION CHARACTERS
           SET FOR THE 'GJW' PARAMETER. CURRENT SET
   CONTAINS: &<>|%;[
            PLEASE REMOVE THE INCOMPATIBLE CHARACTER(S) IN
   ORDER FOR THE PROXY
           TO FUNCTION PROPERLY.
   *********************************************************
   *****
   FAILED: JOB SUBMISSION 'VALIDATION CHARACTERS' SET IS NOT
           COMPATIBLE WITH JOB SUBMISSION USER PROXY.
   ```

```
PLEASE
        SEE ABOVE FOR INFORMATION ON RESOLVING THIS.
*********************************************************
*****
```

2. Navigate to the GSASECR page and on the Institution Profile section, confirm that the GJU and GJW entries do not have "[" or "]" as not valid.

Edit GURJOBS driver shell
Article CMS-2317 in Ellucian Support Center contains information on how to implement GURJOBS as a background process.

Procedure

1. Set the *TNS_ADMIN* variable of the driver shell to the location of the new `tnsnames.ora` and `sqlnet.ora`.
2. Modify the example `banjobs_driver.shl` from CMS-2317 as follows:

```
:
# banjobs_driver.shl
# This script is run as banjobs from from the Unix prompt.
LOGFILE1=/u01/banner/banjobs/gurjobs.log;export LOGFILE1
LOGFILE2=/u01/banner/banjobs/sfrpipe.log;export LOGFILE2
export TNS_ADMIN=/u01/app/oracle/ORDBMS/11.2.0.2/network/
admin/proxy_setup
/u01/banner/banjobs/gurjobs.shl >>${LOGFILE1} 2>&1 &
/u01/banner/banjobs/sfrpipe.shl >>${LOGFILE2} 2>&1 &
```

Notes about Batch Java

The Proxy Wallet can only be used with Oracle OCI JDBC driver. The classpath argument that is passed to the batch program must contain the jar file that is specific to the version of the Oracle Client that is installed.

The classpath variable is set up in `banjavaenv.shl` for Unix/Linux systems and `banjavaenv.pm` for Windows systems.

Notes about Windows file outputs

On Windows, the .lis and .log file names now contain the user ID with brackets as this [saisusr].

- m038216_ban8_[saisusr]_gjrrpts_21947.log
- m038216_ban8_[saisusr]_gjrrpts_21947.lis

Before this change, they were named as the following:

- m038216_ban8_saisusr_gjrrpts_21947.log
- m038216_ban8_saisusr_gjrrpts_21947.lis

Notes when Job Submission server resides on the database server

When using Job Submission Proxy on the database server, unexpected behavior can occur when running jobs. Typically, the result is `ORA-12534: TNS operation not supported`.

Procedure

1. To avoid these errors, add the Oracle OS user to the Job Submission user group.
2. Then, change the permissions of the wallet so that the Job Submission user group also has read access to the wallet.
3. Add the Oracle user to the jobsub group:

```
File - /etc/group
jobsub::507:oracle

[ ~ ]$ cd proxy_wallet/
[ ~/proxy_wallet ]$ chmod 640 *
bash-3.00$ ls -ltr
total 32
-rw-r-----   1 banjobs   jobsub      7696 Jun 24 03:32 ewallet.p12
-rw-r-----   1 banjobs   jobsub      7724 Jun 24 03:32 cwallet.sso
```

Multi-Entity Processing

Multi-Entity Processing enables a user to switch between Institution Codes while logged into Banner. By using Multi-Entity Processing, 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).

For additional information, refer to the *Banner Mult-Entity Processing Implementation Guide*.

Home and process contexts

At institutions that use Multi-Entity Processing, all pages that are Multi-Entity Processing-enabled display two contexts in the title bar: the *home* context and the *process* context.

Home context

The home context is the **Institution Code** that is an Oracle Application Context; the value is your default VPDI code. 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 (GSAVPDI) page.

- If there are multiple **Institution Codes** set up for your User ID on the User Assignment section of GSAVPDI, then you will have multiple codes from which to choose the home context for your Banner session during the login process. 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 section, then that code will be used as your home context, and GUQSETI will not appear when you log in. The Institution Code assigned to you is your default Institution Code.
- If no codes are set up for you on the User Assignment section, 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 some Multi-Entity Processing implementations, the process context can be changed to query different institutions.

If your process context is different than your home context, 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 (BDM) interface. The process context is passed as a parameter in the URL handed from Banner to BDM.

How to switch between institution codes

If your institution has determined to allow cross institution viewing of data on specific pages, you can switch institutions (process context) without leaving the page. This feature only works for data that has been designed to be queried by all institutions.

About this task

The **Institution Selector** button is enabled only for pages that use data which is designed to be queried by all institutions.

Procedure

1. Click the icon on the toolbar of the page.

   --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 page. You can then view the data related to those particular values elsewhere on the page.
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

Changing institution codes impacts your Banner session.

Banner processes always use the user's home context, the institution chosen on the Set Institution Code (GUQSETI) page during Banner login. Banner pages might use the new institution code (your process context) if the Multi-Entity Processing implementation allows the data on the page to be queried across institutions; processes will continue to use your home context.

When you change **Institution Codes** during a session, the Banner page might use the code that appears second in your title bar if the Multi-Entity Processing implementation allows for querying across institutions for the data the page uses. 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 might be used by Banner pages if the Multi-Entity Processing implementation allows for querying across institutions for the data behind the Banner pages.

Job Submission

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

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.

Before 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 (GJAPCTL) page, 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)

Jasper Reports

Changing the institution codes impacts the Jasper Reports.

Running Jasper Reports from the page

Jasper Reports run using the process context setting, as long as the page from which the report is being called is listed on GORVPDI.

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

Running Jasper Reports from Job Submission

Use the home context is if you have Jasper Reports that can only run through GJAPCTL, or if you initiate the Jasper Report through GJAPCTL.

If users need to use the process context for Jasper Reports initiated through GJAPCTL, you may want to add 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 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 Jasper Reports.

Pages

There are certain pages that are used for Multi-Entity Processing.

VPDI Included Objects (GORVPDI)

This page is used to list the pages that are enabled for Multi-Entity Processing at your institution.

When a page is added to this list, then both the home and process contexts appear in the title bar of the page. You should only add pages to GORVPDI where the data behind the page is set up to allow querying across institutions (for example, Query By All).

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

Note: Do not add pages that do not re-query data and use the `Start Over` function. If the data cannot be re-queried, then the user should not be changing **Institution Codes** while on that page.

This page appears on the Miscellaneous General Pages Menu (*GENMISC).

Field	Description
Object Name	Name of the Banner object. Choices come from the Object Maintenance (GUAOBJS) page 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 page allows you to maintain VPD security options. This page is accessed through the *SECURITY menu.

Only the BANSECR user and authorized distributed security users can access this page. Distributed security users can be granted access to this page through the `gssvpdi.sql` script.

Institution Codes section

This section 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
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.
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. This field is not currently used by any Baseline objects. If you want to use it, you must modify Baseline objects. 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.

Field	Description
Voice Response Message Number	This field is used to assign a number to the recorded message.
Activity Date	Date on which this data was created or modified.

User Assignment section

This section allows you to 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.
Code	Code created on the Institution Code Maintenance window.
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 page 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 page.

The valid codes are listed with the default at the top. To change the default institution code, select a different code in the GSAVPDI page and click the **Select** button. You will be logged in to the selected institution, and the new home institution will be displayed in the title bar of GUQSETI page. The new institution code also becomes the default code on the GUQSETI page the next time you login.

The fields on this page are display-only.

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

MEP Rules page - GUAMEPR

This page allows the creation, maintenance, and application of policies for Multi-Entity Processing. The page enables an institution to define the tables for which the data needs to be segregated.

Segregation can be done by defining a template of tables manually or using the delivered templates. The page then generates a scripts that can be executed to apply the necessary changes to columns, indices, foreign keys, triggers, and policies to implement Multi-Entity Processing.

Key block

The Key block is used to search by Owner, Name, and Category. You can enter only one search criteria at a time. Searches are case sensitive.

Fields	Descriptions
Owner	The Oracle ID that owns the table. You can perform a query from this field.
Name	A name of a specific table that exists in GUBMEPO table.
Category	A name of a category that exists in GUBMEPO table.

Templates section

This section is used to load multiple tables in the Objects section. Rows that are displayed in this section are defined in the Business Rules (GORRSQL) page.

Fields

Fields	Descriptions
Category	Displays the Category code associated with the policies.
Status	The column shows the number of rules that match from the **Templates** section with the rules on the **Objects** section. The following statuses can be displayed: • `All <number of tables> match` - All rules from the **Templates** and **Objects** tabs match for all tables. • `<number of tables>/<number of tables> Different` - There are no tables on the **Objects** section that are defined by this template. • `<number of different tables>/<number of tables> Different` - Some rules on the **Templates** section match with some rules on the **Objects** section.
User ID	The User ID of the last user to update the rule in the GORRSQL page that associate with the row displayed.
Activity Date	The date when the rule was last updated in the GORRSQL page.

Fields	Descriptions
Default Access	The header only for **Owner**, **All**, and **User** fields.
Owner	The level of access associated with a user who has the same VPDI code as the data in the table. The access level is controlled by the BY_OWNER policy.
All	The level of access associated with a user who has the same VPDI code as the data in the table. The access level is controlled by the BY_ALL policy.
User	The level of access associated with a user who has the same VPDI code as the data in the table. The access level is controlled by the BY_USER policy.

Buttons

Button	Description
View Tables	Lists the tables defined by the GORRSQL rule along with the currently defined rules on the **Objects** section.
Add Table	Adds the tables with the respective default access to the **Objects** section for any table in that rule that does not exist.
Replace Table	Adds the tables with the respective default access to the **Objects** section for any table in a rule that does not exist and replaces the rule for any entry that exists.
Delete Tables	Deletes the tables from the **Objects** section for any table in a rule that currently exists but does not have the **In Keys** option checked. If the **In Keys** option is checked, VPDI code has been added to at least one of the keys or indices of that table.

Objects section

This section is used to refine the rules established from the **Templates** section as needed. This information is used by the scripts (defined in the next section) that enable Multi-Entity Processing functionality.

Fields

Fields	Descriptions
Table Owner	The schema owner of the table.
Table/View Name	The name of the table.
Category	The category that groups several tables together.
System Code	The GTVSYSI code that indicates the product to which this table belongs.

Fields	Descriptions
System Name	A translation of the system code.
User Id	The User ID of the last user to update this record.
Activity Date	The date when this record was last inserted or updated.
Owner	The level of access associated with a user who has the same VPDI code as the data in the table, which is controlled by the BY_OWNER policy.
All	The level of access associated with a user who has the same VPDI code as the data in the table, which is controlled by the BY_ALL policy.
User	The level of access associated with a user who has the same VPDI code as the data in the table, which is controlled by the BY_USER policy.
VPDI Column	If this option is checked, a column with the name of <table_name>_VPID_CODE is defined.
In Keys	If this option is checked, a column with the name of <table_name>_VPID_CODE is defined and is included in at least one of the keys or indices for the table.
Policy Name	If a VPDI policy exists for the table, the policy associated with the table if a policy exists is displayed. The VPDI policies names are formatted as `<table>_VPDI_<type of access>_POLICY`. <type of access> represents these values: S for select/query, I for insert, U for update, and D for delete. For example, RARAAWD_VPDI_SIUD_POLICY.
Policy Group	If a policy exists for the table, the group to which that policy belongs is displayed.
Select	If this option is checked, a policy exists for the table, and the policy is associate with Query or Select.
Insert	If this option is checked, a policy exists for the table, and the policy is associate with Insert.
Update	If this option is checked, a policy exists for the table, and the policy is associate with Update.
Delete	If this option is checked, a policy exists for the table and the policy is associate with Delete.
Static	If this option is checked, a policy exists for the table, and the policy is defined as STATIC.
Enabled	If this option is checked, a policy exists for the table, and the policy is enabled.
Check Option	If this option is checked, a policy exists for the table, and the policy has the CHECK OPTION enabled.

Fields	Descriptions
Policy Owner	If a policy exists for the table, the schema owner of that policy is displayed.
Package	If a policy exists for the table, the name of the package associated with that policy is displayed.
Function	If a policy exists for the table, the name of the function in the package that is associated with that policy is displayed.

Buttons

Button	Description
View Alter Table Script	Displays the script that will be created by the gmep_genvpid and gmep_genindex scripts.
View Policy Create Script	Displays the script that will be created by the gmep_genpolicy script.
View Trigger Create Script	Displays the script that will be generated by the gmep_gentrigger script.
View Source Code	Displays the source code of the policy.

Defaults section

This section identifies the functions that will be used for generating policies.

Fields	Descriptions
Owner	The owner of the package used by the By Owner Policy, By All Policy, and By User Policy.
Package Name	The package name used by the By Owner Policy, By All Policy, and By User Policy.
Function Name	The function name used by By Owner Policy, By All Policy, and By User Policy.
User ID	User ID of the last user to insert or update this record.
Activity Date	The date when this record was last inserted or updated.

Multi-Entity Processing scripts

There are several scripts that are used to add Multi-Entity Processing to a table. The following table describes the scripts.

Script	Description
gmep_gendisable.sql	Generates two scripts that disables all Banner tables before adding Mult-Entity Processing and enables the triggers after the process is complete.
gmep_genvpdi.sql	Generates the scripts that add the VPDI column to tables.
gmep_genindex.sql	Generates the scripts that add the VPDI column to the indices, foreign keys, unique indices, and primary keys as needed.
gmep_genpolicy.sql	Generates the scripts that add the policies to the tables.
gmep_gentrigger.sql	Generates the scripts that add the triggers for populating the VPDI code for tables.
gmep_gendata.sql	Replicates data from one VPDI code to all the others defined on the GTVVPDI page.
gmep_applyseed.sql	Generates a script to apply another script multiple times to a table for each GTVVPDI code.

Tables

There are certain tables that are used for Multi-Entity Processing.

- Multi-Entity Processing Pages Table (GOBVPDI)
- Multi-Entity Processing Objects Table (GUBMEPO)
- Multi-Entity Processing Policy Default Table (GUBMEPD)
- Multi-Entity Processing Privileges Validation Table (GTVMEPP)
- VPD Institution Code Validation Table (GTVVPDI)
- VPD Institution/Banner User Table (GURURSI)

Database logging and cache control

The Banner Logging and Monitoring Data (GUAMNTR) page provides a convenient way to troubleshoot and diagnose performance issues with database Web Service (WS) requests.

The views and controls on this page provide an administrator the ability to self-diagnose and monitor specific data fields, user interface (UI) page behavior, and page load times during heavy stress cycles with respect to performance and overall health of the application. The Banner Logging and Monitoring Data (GURMNTR) table stores all of the monitoring information for administrative pages in near real-time. Change the logging level of specific loggers, including those that log the request details to the database at run time, without needing to redeploy and restart the application. In addition, users can also view and clear the server-side cache.

This feature provides flexible and scalable database monitoring functionality across deployment options. For example, if your institution has Banner Administrative applications deployed on a different application server from Banner Access Management, statistics for both deployments are available through this new administrative page.

Warning! Logs might contain sensitive information and only authorized users should have access to these logs. It is the responsibility of the institution to delete logs when no longer needed to protect this information. Please ensure that your institution restricts access to the Banner Logging and Monitoring Data (GUAMNTR) page to only those administrators that should have access.

Security and configuration

Ellucian recommends leaving the Banner Logging and Monitoring Data (GUAMNTR) page assigned to the BAN_ADMIN_C security class allowing users with the BAN_DEFAULT_M role access.

The `config.properties` file must contain the following entries to allow the GUAMNTR page to function. Add these entries if they do not exist in your file.

```
log4j.appender.LOG_REQUEST_START_DB = morphis.foundations.core.
appsupportlib.logging.DBRequestStartAppender
log4j.appender.LOG_REQUEST_START_DB.filter.ID  =
 morphis.foundations.core.appsupportlib.logging.MonitorFilter
log4j.logger.morphis.foundations.core.appsupportlib.runtime.monitor.
RequestMonitorStart=INFO, LOG_REQUEST_START_DB

log4j.appender.LOG_REQUEST_INFO_DB = morphis.foundations.core.
appsupportlib.logging.DBRequestInfoAppender
log4j.appender.LOG_REQUEST_INFO_DB.filter.ID  =
 morphis.foundations.core.
appsupportlib.logging.MonitorFilter
log4j.logger.morphis.foundations.core.appsupportlib.runtime.monitor.
RequestMonitorInfo=INFO,LOG_REQUEST_INFO_DB

log4j.appender.LOG_REQUEST_START_DB.appenderName =
 GOKMNTR.log_request_start
```

```
log4j.appender.LOG_REQUEST_INFO_DB.appenderName  =
 GOKMNTR.log_request_info
```

INFO is the minimum verbosity logging level for the `log4j.rootLogger`. For example, `log4j.rootLogger=INFO, stdout`. A logging level set to something less verbose than INFO (OFF, FATAL, ERROR, WARN) results in no data writing to the Banner Logging and Monitoring Data (GURMNTR) table and no data displaying on the Monitors tab.

Monitors tab

Displays Web Service (WS) requests details and performance statistics stored in the Banner Logging and Monitoring Data (GURMNTR) table.

The Monitors tab consists of the following three sections:

- Actions: Includes the **Show error** button, **Purge (filtered) records older than (days)** field, and **Purge** button.
- Request summary: Includes a list of requests and performance indicators for each request record.
- Request details: Displays request details in a single or multiple record view by clicking the **Single Record** or **Multiple Record** icon.

Actions

The **Show error** button, if enabled, displays the error of the selected request. The **Show error** button becomes enabled when the selected request contains an error indicated by the selection of the **Error Description Stored** check box.

The **Purge** button allows the deletion of monitor data. You can purge records older than *n* days by entering a number (*n*) into the **Purge (filtered) records older than (days)** and click **Purge**. or you can purge records containing specific criteria by clicking the **Filter** button and applying filter criteria. The purge function purges only rows matching the filter criteria and older than *n* number of days.

Request summary

Each record listed in the request summary section contains information that identifies each request such as Request Timestamp, Principal user, Username, Page Name, etc, followed by timing performance statistics. This view is also known as the multiple record view, beneficial if you need to scroll through a long list of requests to find those that have errors (**Error Description Stored** check box enabled).

Request details

Multiple record view is the default view when you first access the Banner Logging and Monitoring Data (GUAMNTR) page. You can switch between single and multiple record view by selecting the **Single Record** or **Multiple Record** icons.

Selecting the **Single Record** icon displays the details of one record. This view displays the same details and timing statistics as displayed in the multiple record view, however this view organizes the details into the following four categories:

- Detail: Request identification information
- Server: Server-side performance indicator details.
- Database: Database performance indicators
- Client: Browser performance indicators.

Detail

Displays the request/response details.

Field	**Description**
Request Timestamp	Request timestamp
Principal	Name of user logged in (Single sign-on (SSO) user)
Username	Name of user at database level
Page Name	Name of the page
Action Name	Action executed
Response Page	Name of the current page when request completes
Page ID	Unique ID of the current page when request arrives
Parent Page ID	Name of the parent page
Interaction ID	User interaction state ID
Request ID	Request ID
Previous Request ID	Previous request ID
Request Status	Request status: • "OK" = response sent. • Empty field = Processing
Server Name	Name of the application server where Banner Administrative Pages and Banner Access Management runs.
WebApp Name	Name of Banner General application. Example: Banner GeneralAdmin.ws or Banner GeneralAccessMgmt.ws
Page Module Name	Name of the Banner General Module For example, General, Student, Security among others.
Response Page Module Name	Name of the Banner General Module For example, General, Student, Security among others.
Client Address	User's browser IP address

Field	Description
Client Name	User's browser name and version

Server

Displays the server-side performance indicator details.

All time values displayed in the fields below display in milliseconds (ms).

Server field	Description
Request Exec Time	Overall request time
Action Exec Time	Time spent executing the action
Config Load Time	Time spent loading the configuration from the filesystem
DB ValueSet Load	Time spent loading ValueSets from database
WS Read Time	Time spent reading from database
WS Write Time	Time spent writing to database
Bind From Time	Time to bind the request message (server side)
Bind To Time	Time to bind the response message (server side)
Pre Invoke Action Exec Time	Time before executing the action on the server
Invoke Action Exec Time	Time before executing the action on the server
Pos Invoke Action Exec Time	Time after executing the action on the server
Logger Exec Time	Time it takes to execute the logging action
Used Memory	Estimated memory used by task

Database

Displays the database performance indicators

All time values displayed in the fields below display in milliseconds (ms).

Database field	Description
DAO Load Time	Time spent loading Data Access Objects
DAO Query Time	Time spent querying data from the database
DAO Save Time	Time spent saving Data Access Objects
DAO Update Time	Time spent updating Data Access Objects
DB Read Hits	Number of database reads
DB Read Exec Time	Time spent reading from database

Database field	Description
DB Write Hits	Number of database writes
DB Write Exec Time	Time spent writing to database
DBWrp Marshall Time	Time spent marshalling queries to database
DBWrp Unmarshall Time	Time spent unmarshalling results from queries to database

Client

Displays the browser performance indicators.

Field	Description
Client Service	Time that client side waits for the service (message from the server)
Client Message	Time it takes to parse the message
Client Task	Global time it takes to process all monitors and being ready for user input
Client Json Find	Time it takes for specific data (block/item values) to bind to Widgets
Client Json Find Count	Number of times that data was search in the model
Client View	Time consumed from the time of the view request until the view is ready. Specifically, the time it takes to get the HTML and initialize the widgets and controls
Client View Get	The time it takes to get the HTML file (this should happen the first time the view downloads from the server)
Client Init Controls	Time spent on controls and widgets initialization (this should happen the first time the view downloads from the server and initialized)
Client Reinit Controls	Time spent on controls and widgets refresh. This could happen when using accordions, collapsible panels, and tabpanels when initializing the first visible container (i.e: tabpage) or when opening new containers (i.e: other tabpage)
Client Reinit Controls Count	Number of repainted widgets and controls
Client Pre Control Info	Properties that need to be set before MODEL
Client Pre Control Info Count	Number of properties set before MODEL
Client Window Info	Time spent setting window control properties
Client Control Info	Time spent setting block and item properties
Client Control Info Count	Number of blocks receiving CONTROL_INFO
Client Model	Time consumed when setting data (binding)
Client Model Count	Number of tasks where MODEL was applied

Views tab

The views tab provides an easy way to view and delete the server-side cache for a page.

Refresh the page to see the latest cached view updates.

Server-side cache consists of the following data:

Field	Description
Name	Name of the view
Created	View creation date and time
Accessed	Last accessed date and time of the view
Hits	Number of times the view loaded from cache

You can perform the following actions on this tab:

Button	Description
Show content	Click to display the cached information of the selected or highlighted record.
Select All	Click to select all records to delete
Unselect All	Click to clear the **Selected** check box for all of the selected records
Delete Selected	Click to delete all selected records
Selected	Select one or more views that you want to delete.

Loggers tab

The Loggers tab provides a way to configure the logging level without the need to edit the `config.properties` file, redeploy and restart the application server.

Warning! Logs might contain sensitive information and only authorized users should have access to these logs. It is the responsibility of the institution to delete logs when no longer needed to protect this information. Please ensure that your institution restricts access to the Banner Logging and Monitoring Data (GUAMNTR) page to only those administrators that should have access.

This tab displays all known logger names, appenders, logging level, inheritance, and filter.

Field	Description
Name	Name of the logger (read only)
Appenders	Log message destination (read only)

Field	Description
Level	Select one of the available logging levels if not using the level of the parent/root logger: • ALL: Logs all information. • TRACE: Logs all TRACE, DEBUG, INFO, WARN, ERROR and FATAL information. • DEBUG: Logs DEBUG, INFO, ERROR and FATAL information • INFO: Logs INFO, ERROR and FATAL information. • WARN: WARN, ERROR and FATAL information • ERROR: Logs only ERROR and FATAL information. • FATAL: Logs only FATAL information. • OFF: Turns off logging. No information logged.
Inherited	If selected, inherits the logging level from the parent / root, otherwise uses the level selected in the **Level** field.
Inherits From	Parent / root logging level to inherit from.
Filtered	If selected, uses the conditions specified on the **Logger Script** tab.

An application restart clears any changes entered on this page.

Logger Script tab

The Logger Script tab provides a way to edit the JavaScript function bodies evaluated by the framework to choose specific requests and details to log and display on the Monitors tab.

Use of these customizable functions apply only to loggers with the **Filter** check box selected on the Loggers tab.

There are two JavaScript functions available to edit on this tab.

Section	Description
FILTER	Specify criteria that you want the log event to return and display on the Monitors tab. The filter function has access to the following set of request properties: • logStage: START (of request) or INFO (end of request) • messageRequest: Request message • messageResponse: Response message (only available if logStage = INFO) • requestId: Single request ID • interactionStateId: User interaction ID • timestamp: Request timestamp (only available if logStage = START) • client_CLIENT: Browser name (only available if logStage = START) • pageName: Page name when request started • blockName: Block name when request started • blockItem: Item name when request started • actionName: Request action name • remoteAddress: Client address • serverName: serverAddress • principal: Single sign-on (SSO) username • username: Database username
LOG TASK MEMORY	Enable to compute and record memory usage that displays on the Monitors tab. • `return true;` to enable. • `return false;` to turn off. You cannot filter on task memory.

Edit the function bodies monitor script

Edit the JavaScript function bodies to enable and define specific criteria to capture and display on the Monitors tab.

Procedure

1. Access the **Banner Logging and Monitoring Data (GUAMNTR)** page.
2. Go to the Logger Script tab.

3. Edit the filter function to define the specific criteria that you want to display on the Monitors tab.

   See the following examples:

Description	Script example
Monitor specific pages	`return (['GSADSUM, SPAIDEN'].indexOf(e.log('taskName'))>-1);`
Monitor a specific block	`return (['CLASS_BLOCK'].indexOf(e.log(blockName)) > -1 );`
To monitor NEXT_ITEM on a specific page	`return ((['GUAABOT'].indexOf(e.log('taskName')) > -1) && (['NEXT_ITEM'].indexOf(e.log('actionName')) > -1 );`

4. Set the `logTaskMemory` function return value to `true` in the LOG TASK MEMORY section to enable computing and recording memory usage.
5. Click **Save**.

   An application restart clears any changes entered on this page.