Table of Contents

Overview.............................................................................................................................................. 3
Step Description and Dependencies Table.......................................................................................... 4
Step 1  Distribute Release Documents ............................................................................................... 5
Step 2  Verify Environment Prerequisites .......................................................................................... 5
Step 3  Upgrade Prerequisites .............................................................................................................. 6
Step 4  Upgrade preparation .................................................................................................................. 11
Step 5  Modify database objects .......................................................................................................... 13
Step 6  Migrate from stage to permanent directories ......................................................................... 16
  UNIX .................................................................................................................................................. 17
  MICROSOFT WINDOWS ................................................................................................................... 18
Step 7  Compile COBOL programs ....................................................................................................... 18
Step 8  Compile C programs.................................................................................................................. 18
Step 9  Apply required data changes .................................................................................................... 18
Step 10 Generate Oracle Forms .......................................................................................................... 18
Step 11 Generate Oracle Reports ....................................................................................................... 18
Step 12 Compile Java Programs .......................................................................................................... 19
Step 13 Update letter generation/population selection tables ............................................................. 19
Step 14 Update referential integrity constraints ................................................................................. 19
Step 15 Restart the gostage process ..................................................................................................... 19
Step 16 Verify the state of the upgraded environment .................................................................... 19
Index...................................................................................................................................................... 21
Overview
This document describes the steps required to upgrade Banner General release 8.11 to release 8.11.1. It is designed to be used in conjunction with the Banner Release Interdependencies and the Banner Media Unload Reference Guide documents, which are available under the 'Banner' tab in the 'Documentation' section under the 'Resources' tab in Ellucian Customer Center.

The Release Interdependencies document outlines the release interdependencies for all products within the Unified Digital Campus (UDC). Before you install or upgrade a product within the UDC, use this document to determine which other UDC products you must install first.

The Banner Media Unload Reference Guide describes the common steps that are required for every Banner upgrade. It also includes reference information about the symbols and conventions used throughout Banner upgrade guides.

NOTES:

1. Any upgrade issues discovered and reported to the Action Line subsequent to the posting of this release will be documented in the "Banner General 8.11.1 Upgrade Issues" Article 000044762 and made available via the Ellucian Customer Center (http://login.ellucian.com). It is necessary that you check this document prior to applying the release by querying for Article 000044762.

2. This upgrade can be installed manually and is also available for installation using the Ellucian Solution Manager (ESM) or the Automated Installer. The minimum version of ESM required is 1.12. See the "Banner Upgrades Support Status" document available within Documentation Libraries on the Ellucian Customer Center, for a complete list of Banner releases that ESM supports for installation.

Additional information regarding the Automated Installer can be found in Step 3 Part B of this upgrade guide.
Step Description and Dependencies Table

The following list of steps is consistent from release to release. Any steps that are not required are marked N/A in the "Applies to this upgrade" column in this table.

<table>
<thead>
<tr>
<th>Step</th>
<th>Applies to this upgrade</th>
<th>Description</th>
<th>Depends on Step</th>
<th>Restart Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Distribute Release Documents</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Verify Environment Prerequisites</td>
<td>Previous</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Upgrade Prerequisites</td>
<td>Previous</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Upgrade Preparation</td>
<td>Previous</td>
<td>A</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Modify database objects (gostage)</td>
<td>Previous</td>
<td>B</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Migrate from stage to permanent directories</td>
<td>Previous</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>N/A</td>
<td>Compile COBOL programs</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>N/A</td>
<td>Compile C programs</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>9</td>
<td>N/A</td>
<td>Apply additional data changes</td>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>N/A</td>
<td>Generate Oracle forms</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>11</td>
<td>N/A</td>
<td>Generate Oracle reports</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>12</td>
<td>N/A</td>
<td>Compile JAVA programs</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>13</td>
<td>N/A</td>
<td>Update letter generation/population selection tables</td>
<td>Previous</td>
<td>A</td>
</tr>
<tr>
<td>14</td>
<td>N/A</td>
<td>Update referential integrity constraints</td>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>15</td>
<td>N/A</td>
<td>Restart the gostage process</td>
<td>Previous</td>
<td>A</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>Verify the state of the upgraded environment</td>
<td>Previous</td>
<td>A</td>
</tr>
</tbody>
</table>

If any errors or problems occur during the upgrade process, login to the Ellucian Customer Center at [http://login.ellucian.com](http://login.ellucian.com) to search for solutions or to submit a case for assistance with the issue. You can also report issues via telephone at 1-844-358-7222

MULTI-ENTITY PROCESSING ALERT: This upgrade delivers steps to support MULTI-ENTITY PROCESSING ( MEP*) in your database and requires no manual modifications to the delivered scripts during the course of installation.

*MEP refers to MULTI-ENTITY PROCESSING with the use of Virtual Private Databases (VPD), enabled by the use of VPDI_CODE in MEP enabled tables.

For additional information to determine if the database is MEP enabled, please refer to Article 000037561: How can I determine if MEP (Multi-Entity Processing) is installed in my database instance.

Read All Instructions Before Beginning; Review the Contents of All Scripts (SQL, SHL and PL files).
Step 1  Distribute Release Documents

Distribute the Release Guide available in the Banner tab in the 'Documentation' section under the 'Resources' tab of the Ellucian Customer Center to the appropriate departments. This document explains the modifications that have been made to the system in functional terms and explains those actions that must be taken by the users in preparation for or as part of the release upgrade.

Do not proceed until the responsible users indicate that any current processes or cycles have completed and will not be affected by the upgrade.

Step 2  Verify Environment Prerequisites

Part A  This upgrade is recommended to be applied with Oracle Database Release 12cR2 (12.2.x.x).

Please refer to Article 000006696 for a complete list of support recommendations for Banner Oracle Database Supported Versions.

Be sure all Oracle users are logged off and cannot or will not log on. If you do this by starting the database in Restricted mode, then all user IDs used in the installation will need the Restricted Session system privilege for the duration of this upgrade. (For more information refer to the Oracle Server Administrator's Guide.) The user IDs affected may be found in the ggdvedba.sql script and in the login.sql file delivered with this upgrade where the variables BANNER_OWNERS, ARCHIVE_OWNERS, and UPGRADE_OWNER are defined. Note that all Banner object owners are defined by these variables whether they pertain to your installation or not. Therefore, not all of the BANNER_OWNERS, ARCHIVE_OWNERS, and UPGRADE_OWNER defined will exist at your installation. Under no circumstances should you create these user IDs unless specifically instructed to do so by these installation instructions—if you do, you risk the chance of your upgrade failing.

NOTE:  The user IDs in the ggdvedba.sql script will be given the DBA role as a default role, so a direct grant of Restricted Session is not necessary. The script is mentioned here only for completeness.

COMPLETE BACKUPS OF YOUR EXISTING SYSTEM BEFORE CONTINUING!

Part B  Apply this upgrade to your SEED instance first. Never apply it to production without familiarizing yourself with the process by executing it against a non-critical database. This stage must be applied to all of your Banner General environments.

For all platforms set SQLPATH equal to ORACLE_PATH.

If you are running under UNIX, be sure that the current directory (represented by a ".") is at the front of your ORACLE_PATH, SQLPATH and UNIX path to avoid any problems when starting some of the upgrade SQL scripts and shells.

If you are running under MICROSOFT WINDOWS, be sure that the plus subdirectory of every Banner product you license has been added to the HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\SQLPATH registry entry and/or the SQLPATH environment variable to avoid any problems when starting the upgrade SQL scripts. Please also note that the commands you use to import files and to start SQL*Plus will depend on the version of Oracle Server you are using. To perform an upgrade on the MICROSOFT WINDOWS platform, you must open up an MS-DOS window and execute all commands from the MS-DOS command prompt.
TEMPORARY TABLESPACE ALERT:

Banner uses Oracle’s functionality for creating global temporary tables. Processing which utilizes these tables increases the required amount of temporary tablespace. To avoid the Oracle error: ORA-1652: unable to extend temp segment, it may be necessary to increase the amount of available temporary tablespace. Please refer to Article 000024265 “CMS-12177: Running Banner and the Menu GUAGMNU displays the error FRM-47314 and FRM-47319” for more information. If applicable, global temporary tables can be found listed in the database comparison reports and are identified with “Global Temp” listed in the Sizing Model column.

Step 3 Upgrade Prerequisites

Part A Installation requirements

The installation procedure assumes you are applying these modifications to an environment in which Banner General release 8.11 has been installed.

For more information you may refer to the Release Interdependency Matrix, Banner Release Interdependencies.pdf, available for download from the Ellucian Support Center.

If the Banner ODS, Banner EDW, or one of the Performance applications is installed in your environment, please review the dependency & compatibility information available in the latest version of the “Banner Analytics Resource Guidelines” which can be downloaded from the Ellucian Customer Center.

For clients leveraging Oracle Streams as their replication technology, please refer to Article 000009579 “1-10FYAMQ: Banner/ Advance/ BPRA/ Oracle upgrades on an instance with Oracle Streams installed” prior to applying this

For clients leveraging Oracle Materialized Views as their replication technology, please refer to Article 000038387 “Restaging Materialized View after Banner Upgrade (Best Practices)”

NOTE: Clients interested in switching from Oracle Streams to Materialized Views should review the “Banner ODS Switching Streams to Mviews Supplement.”

Any modifications you have made to the base product will remain your responsibility. Each object that has been modified contains descriptive text about the purpose of the modification. This text can be found at the beginning of each object, except for forms, for which the comments are found in Form level procedures named AUDIT_TRAIL_“release_number”.

Part B This upgrade may be applied using the Automated Installer. This tool will allow you to apply an upgrade using a menu driven front end that follows each step contained in the upgrade guide. Use of this tool is optional. If you would like to apply this upgrade using the Automated Installer, please see the README.txt file for details regarding installer setup, system requirements and execution.

Use of the Automated Installer will save time doing the upgrade in terms of overall number of keystrokes, but PLEASE NOTE that it does not serve as a substitute for reading each section, step and/or part of the upgrade guide. Failure to read this upgrade guide could cause the upgrade process to be incomplete or unsuccessful.
### Part C

In this part, you will define several SQL*Plus variables that are used at various places in the upgrade process. These variables control where files are written and specify which options are to be used during this upgrade.

Delivered with the stage material is a file called `login.sql`. If you have your own `login.sql` file and need to have it executed, add the following define commands to your `login.sql` file and then remove the `login.sql` file delivered in the stage directory.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLUS_CMD</td>
<td>The <code>plus_cmd</code> variable determines the command to be used to invoke SQL<em>Plus when executing a HOST started SQL</em>Plus task during this upgrade. The default value of this variable is 'sqlplus'.</td>
</tr>
<tr>
<td>SPLPREF</td>
<td>The <code>splpref</code> variable defines the file prefix used by the steps in this installation process that generate listings or intermediate SQL routines. This provides a method to segregate the generated output when the stage has to be applied to more than one instance.</td>
</tr>
<tr>
<td>APPLYMOD</td>
<td>The <code>applymod</code> variable indicates whether or not modifications should be applied automatically. The variable has two valid values: domod or nomod. Choose nomod if you want to review the table alterations before they are applied.</td>
</tr>
</tbody>
</table>

To understand the effect of this option, you must understand the flow of the `gostage` process. The first part of this process is an analysis and comparisons of the modifications that are to be applied by this release with the history of what modification have already been applied to your environment. The modifications that have not been applied to your environment are written to a file. The second phase of this process is to apply these changes and to record each one as it is applied. If the `applymod` variable is set to domod, the second phase is automatically executed.

After the analysis of your environment is completed, the modifications that must be applied to your environment will be found in a file called `domod.sql`. If the `SPLPREF` variable was used to route generated files to another directory, the `domod.sql` file will be found there.

If you run `domod` and it fails:

- you may run `gostage` again to generate a new `domod.sql` file, or
- you must edit `domod.sql` by removing from it the steps that had successfully completed before rerunning it.
| UPGRADE_OWNER/UPGRADE_OWNER PASSWORD | The variable `upgrade_owner` defines the Oracle account which will own the modification tables to be imported in Step 4 of the upgrade (previously, these tables were owned by `GENERAL`). The `login.sql` file is delivered with a default `upgrade_owner` of `upgrade1`.

   - The variable `upgrade_owner_password` is the Oracle password by which the `upgrade_owner` account will be identified. The `login.sql` file is delivered with a default text of “#UPDATEME#” for the `upgrade_owner_password`.
   - In the first part of Step 4 of the upgrade, you will be instructed to execute the General plus script `gupuser.sql` to verify that the `upgrade_owner` account of the name you have specified in `login.sql` exists. If it does not exist, `gupuser.sql` will create the `upgrade_owner` account identified by the `upgrade_owner_password` you have specified and will create under its schema all of the objects it will require to perform the upgrade.

   - When the `gostage` process is started, you will be notified of the particular `upgrade_owner` account being used to perform the upgrade.

| PASSWORDS | The next section of the `login.sql` files defines the passwords to all the Banner product owner accounts. Specifying them here prevents the scripts from prompting you for them each time they need to connect or switch from one account to the other. If you have any other Oracle IDs that own Banner tables, you will have to define their passwords as well.

   - It will be a security problem for you to put your regular passwords into the `login.sql` file. We recommend that you temporarily change all the account passwords for the duration of the upgrade and set them back after the upgrade is completed.

| TABLE AND INDEX SIZING MODELS | The next section of the `login.sql` file defines what tables and indexes are to be used as sizing and placement models for the new tables created during this upgrade. Each product has a definition for small, medium, large, and huge tables and indexes. If you do not feel the value specified reflects your environment, please feel free to change it. Definitions for products that you do not have will be ignored by the upgrade process.

| CLOB/BLOB SIZING MODELS | The next section of the `login.sql` file defines the default clob/blob sizes which are to be used when creating new columns of these types during the upgrade. Again, if you do not feel the values specified for these variables reflect your environment, please feel free to change them. In addition, be sure to verify that the model “???_tablespace_name” of DEVELOPMENT is appropriate for your environment. Definitions that are not used by this upgrade will be ignored.

| SECURITY_AUDIT | The `security_audit` variable is used to turn all Banner BANSEC security auditing on, off or leave them unchanged. Set this variable to “F” to log user date and time records for every change made to Banner...
Security tables during the upgrade process. Set this variable to “N” if you do not wish to audit changes to the Banner Security tables. If you have already defined which BANSEC tables will be audited and determined it should not be changed during the upgrade process, leave the current status set to “U” (default).

<table>
<thead>
<tr>
<th>LOCAL OWNERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>The <code>local_owners</code> variable specifies which non-standard Banner owners should be kept in sync with the modifications being delivered with this upgrade. For example, if you have cloned part of Banner Student under an Oracle ID called MARS and you want these tables to be kept in sync with the SATURN tables, add MARS to the <code>local_owners</code> variable.</td>
</tr>
</tbody>
</table>

The owner IDs defined in this variable must be:
(a) In upper case
(b) Enclosed in single quotes
(c) Separated by commas
(d) The first entry must be preceded by a comma
(e) The list must be enclosed in double quotes
(f) Added to the `PASSWORDS` section mentioned previously
(g) Possibly granted the restricted session privilege - refer to step 3

Examples:
- `define local_owners = ""` (no additional owners)
- `define local_owners = ",'PLUTO','MARS"` (two additional owners)
<table>
<thead>
<tr>
<th>ATTENTION: A NOTE REGARDING DELIVERED ELLUCIAN CODE AND CLEAR TEXT PASSWORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>In order to comply with <em>USA Federal Audit Standard GAO-09-232G</em>, ALL Ellucian releases, effective August, 2011, will no longer deliver code with clear text passwords.</td>
</tr>
<tr>
<td>This affects the delivered file <code>login.sql</code> as well as C and COBOL compile scripts and form generate scripts for the UNIX and Windows environments.</td>
</tr>
<tr>
<td>You will now be responsible for editing the delivered <code>login.sql</code> script for every upgrade and replacing the string “#UPDATEME#” with whatever value the particular schema owner’s password is in your environment. You will be required to do this for all Banner schema owners that exist in your particular environment.</td>
</tr>
<tr>
<td>The compile scripts (C/ COBOL/ form generate scripts/ report generate scripts) in the upgrade, for all supported platforms, have been modified to include environment variables that need to be defined at your site in order to successfully compile/generate the Ellucian delivered objects.</td>
</tr>
<tr>
<td>For the UNIX/LINUX environment, issue the following command:</td>
</tr>
<tr>
<td><code>export DFLT_BANINST1_PASS=the_value_of_baninst1_password;</code></td>
</tr>
<tr>
<td>For the Windows environment issue the following command:</td>
</tr>
<tr>
<td><code>SET DFLT_BANINST1_PASS=the_value_of_baninst1_password</code></td>
</tr>
<tr>
<td>The delivered generate and compile scripts will expect these environment variables to be present in your environment, defined and available to be used at the appropriate times.</td>
</tr>
</tbody>
</table>
Step 4  Upgrade preparation

In this step you will perform several tasks in preparation for applying the upgrade.

**WARNING:** Before beginning this upgrade, please review Step 3 Part C for additional instructions regarding establishment of environment variables and edits to the file login.sql.

Do not proceed until the necessary edits have been finished.

Part A

In this part you will run a script that will:

- Verify that all necessary pre-requisite products have been applied to the environment.
- Verify that the upgrade_owner account specified in login.sql exists. If it does not exist, gupuser.sql will call gcreuser.sql to create the upgrade_owner account and all of the objects it requires to perform the upgrade. If it does exist, gupuser.sql will call gchkuser.sql to make sure it owns all of the objects required to perform the upgrade. If any objects are missing the script will inform you and terminate. At this point you may either drop the user with the cascade option, provided they do not own any objects, or you may create the missing objects by hand. Refer to the gcreuser.sql script for the required objects and their DDL. If you would like to use an upgrade_owner account other than the default owner of upgrade1, refer to Step 3.
- Create a script to be run by gresroled to restore the default roles assigned to the users used in this upgrade.
- Create a script to be run by grestrigs to restore the BANSECR security audit setting back to pre-upgrade status.
- Run gsecaud.sql which will turn all BANSECR security auditing on, off or maintain the pre-upgrade auditing status based on the SECURITY_AUDIT setting in login.sql.
- Alter the default roles assigned to specific accounts used by this upgrade to include the DBA role. Refer to the ggivedba.sql script found in either the current directory or the plus subdirectory for this product for a list of accounts to be modified. These accounts will have their current default roles restored at the end of the upgrade when the gresroled is run.
- Drop the modification tables before importing them. This will assure you have the correct structure for these tables.
- It checks if Multi-Entity Processing has been implemented in your database and if yes, then warns the clients of the additional dependency for MEP aware installation.
- It verifies that user BANINST1 does not have any Fine Grain Access Rules in place that would prevent the inserting of MEP data.

Invoke SQL*Plus and run the procedure:

sqlplus /nolog @gruready

Review:   gruready listing
After this part is completed, you will have the following files in the directory that the `SPLPREF` variable points to.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Examine</th>
</tr>
</thead>
<tbody>
<tr>
<td>gresrole.sql</td>
<td>Script run by <code>gresroled</code> to restore the default roles assigned to the users.</td>
<td>Yes</td>
</tr>
<tr>
<td>grestrig.sql</td>
<td>Script run by <code>grestrigs</code> which restores BANSECR security audit settings back to pre-upgrade status.</td>
<td>Yes</td>
</tr>
<tr>
<td>ggivrole.sql</td>
<td>Script which altered the users to include DBA as one of their default roles.</td>
<td>Yes</td>
</tr>
<tr>
<td>ggivrole listing</td>
<td>Spooled output from the <code>ggiverole</code> script.</td>
<td>Yes</td>
</tr>
<tr>
<td>gruready listing</td>
<td>Spooled output from the <code>gruready</code> script.</td>
<td>Yes</td>
</tr>
<tr>
<td>gmepeready.sql</td>
<td>Script which invokes other scripts which verify the additional dependencies for a MEP client and also displays messages that are intended for a MEP client. This is spooled output from the <code>gruready</code> script.</td>
<td>Yes, if you are a MEP client.</td>
</tr>
<tr>
<td>gmepeready listing</td>
<td>Spooled output from the <code>gmepeready</code> script. The script verifies the additional dependencies for a MEP client and also displays messages intended for a MEP client.</td>
<td>Yes, if you are a MEP client.</td>
</tr>
</tbody>
</table>

**Part B**

In this part you will import the scripts that modify tables, functions, views, packages and procedures, database triggers and other database objects when `gostage` is run. If you have chosen a different value for `upgrade_owner` other than the default of `upgrade1`, you **MUST** modify the `loadmods.par` file to specify this new value of the `touser` parameter.

```
imp upgrade_owner/password parfile=loadmods.par file=gen81101.dmp
```

The import step will generate a warning stating that the file was exported by `GENERAL`. This is expected and can be ignored.

Verify that the import completed successfully for all tables by comparing the number of rows shown on the terminal with the counts shown in the `gen81101.log` file.

**Part C**

In this part you will perform the tasks necessary that will enable you to run the GUASMOD form (described in restart code D) as `upgrade_owner` should the need arise. Specifically, `upgrade_owner` will be given select privilege with grant option on the `GURDMOD` table, and the `GUVMODS` view will be dropped as `BANINST1` (should it still exist) and recreated under the current `upgrade_owner` account.

Invoke SQL*Plus and run the procedure:

```
sqlplus /nolog @guovmods
```

To run the GUASMOD form, use the userid and password defined by the `upgrade_owner` and `upgrade_owner_password` variables described in Step 3.
Part D  
In this part you will run the `gurutlrp.sql` utility script to compile database objects that are in an invalid state. The `gurutlrp.sql` script runs ORACLE's utlrp utility script (as SYS) and then displays a list of the remaining invalid database objects. This script is run to prevent the upgrade process from failing if it attempts to use one of the invalid objects. All errors should be investigated before continuing to the next step.

Invoke SQL*Plus and run the procedure:

```
sqlplus /nolog @gurutlrp
```

Review: `gurutlrp` listing

You may need to repeat this process several times until all dependencies are validated.

```
NOTE: The following Oracle error message may appear in the gurutlrp listing file. This is a known Oracle bug associated with Oracle's utlrp.sql process and a patch is available (Oracle patch number - 7707103). Applying this patch will resolve this issue. However, the process will complete the validation process and you may continue with the upgrade.

DECLARE
  *
ERROR at line 1:
  ORA-00904: "FALSE": invalid identifier
  ORA-06512: at line 13
```

Step 5  
Modify database objects

Part A  
In this part you will modify existing General tables, create all GENERAL packages, procedures, foreign key constraints and required data as well as create or replace functions, views and database triggers. Changes that you have made may be affected by this process.

```
MULTI-ENTITY PROCESSING ALERT: This upgrade supports MULTI-ENTITY PROCESSING (MEP*) in your database and requires no manual modifications to the delivered scripts during the course of installation.

*MEP refers to MULTI-ENTITY PROCESSING with the use of Virtual Private Databases (VPD), enabled by the use of VPDI_CODE in MEP enabled tables.

For additional information to determine if the database is MEP enabled, please refer to Article 000037561: How can I determine if Multi-Entity Processing (MEP) is installed in my database instance.
```
The `gostage` process may exit and instruct you to execute some other step in this document. In this case, after executing the other step, you must always restart the `gostage` script. The `gostage` script will tell you when all steps of the installation process have completed.

After this step is completed, you will have the following files in the directory that the `SPLPREF` variable points to.

**NOTE:** `VRRFF` represents the release of General for which the files listed below are being generated, (where *V* is the major product version, *RR* is the product release level, and *FF* is the “fix level” or interim release level. For example, 80000 for the 8.0 release). If the upgrade is cumulative, and the initial releases of the upgrade have already been applied, scripts which were run during those releases will not be run again. Thus, there will be no spooled output to examine in the `splpref` directory for some scripts. For example, if you have already applied the 7.0.1 release of a product, you will neither find the generated grant script `gbg70001.sql` nor its spooled output, the `gbl70001` listing, in the `splpref` directory.

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Examine</th>
</tr>
</thead>
<tbody>
<tr>
<td>checksum.sql</td>
<td>Intermediate file created when examining what modifications are currently on your system.</td>
<td></td>
</tr>
<tr>
<td>domod.sql</td>
<td>Script of all the modification SQL and commands that must be applied to your system to apply the upgrade.</td>
<td></td>
</tr>
<tr>
<td>examgrt.sql</td>
<td>Generated script that is executed after generating grants for new database objects.</td>
<td></td>
</tr>
<tr>
<td>listabl listing</td>
<td>Spooled output from the modification process.</td>
<td>Yes</td>
</tr>
<tr>
<td>mhuginx.sql</td>
<td>Sizing definition for a huge index.</td>
<td></td>
</tr>
<tr>
<td>mhugtab.sql</td>
<td>Sizing definition for a huge table.</td>
<td></td>
</tr>
<tr>
<td>mlginx.sql</td>
<td>Sizing definition for a large index.</td>
<td></td>
</tr>
<tr>
<td>mlrgtab.sql</td>
<td>Sizing definition for a large table.</td>
<td></td>
</tr>
<tr>
<td>mmedinx.sql</td>
<td>Sizing definition for a medium index.</td>
<td></td>
</tr>
<tr>
<td>mmedtab.sql</td>
<td>Sizing definition for a medium table.</td>
<td></td>
</tr>
<tr>
<td>msmlinx.sql</td>
<td>Sizing definition for a small index.</td>
<td></td>
</tr>
<tr>
<td>msmltab.sql</td>
<td>Sizing definition for a small table.</td>
<td></td>
</tr>
<tr>
<td>nomod.sql</td>
<td>Script that is executed if the <code>applymod</code> variable is set to <code>nomod</code>.</td>
<td></td>
</tr>
<tr>
<td>gbg\textit{VRRFF}.sql</td>
<td>Generated grant script which gives BANINST1 full privileges to all objects owned by other baseline Banner product owners.</td>
<td></td>
</tr>
<tr>
<td>gbl\textit{VRRFF}.lst</td>
<td>Spooled output from BANINST1 grant script (<code>gbg\textit{VRRFF}.sql</code>).</td>
<td></td>
</tr>
<tr>
<td>gfg\textit{VRRFF}.sql</td>
<td>Generated foreign grant script which gives the owner of the product being upgraded access to new tables and views owned by other products.</td>
<td></td>
</tr>
<tr>
<td>gfl\textit{VRRFF} listing</td>
<td>Spooled output from the foreign grant script (<code>gfg\textit{VRRFF}.sql</code>).</td>
<td></td>
</tr>
<tr>
<td>glisall listing</td>
<td>Spool file built during the extract table allocation process.</td>
<td></td>
</tr>
<tr>
<td>gol\textit{VRRFF} listing</td>
<td>Spooled output from the reapplication of the saved grants for tables script (<code>gog\textit{VRRFF}.sql</code>).</td>
<td></td>
</tr>
<tr>
<td><strong>govgVRRFF.sql</strong></td>
<td>Generated script containing all grants issued for views to be dropped and recreated. The grants are then reissued from the BANINST1 account.</td>
<td></td>
</tr>
<tr>
<td><strong>govlVRRFF listing</strong></td>
<td>Spooled output from the reapplication of the saved grants for views script (govgVRRFF.sql).</td>
<td></td>
</tr>
<tr>
<td><strong>gylVRRFFa listing</strong></td>
<td>Spooled output from the first pass at creating public synonyms (gymVRRFFa.sql).</td>
<td></td>
</tr>
<tr>
<td><strong>gylVRRFFb listing</strong></td>
<td>Spooled output from the second pass at creating public synonyms (gymVRRFFb.sql).</td>
<td></td>
</tr>
<tr>
<td><strong>gymVRRFFa.sql</strong></td>
<td>Generated script which will create public synonyms for any object BANINST1 has select access to, but for which no public synonym exists. This is necessary since views, database procedures, etc., do not prefix database objects with the owner.</td>
<td></td>
</tr>
<tr>
<td><strong>gymVRRFFb.sql</strong></td>
<td>Generated script which will create public synonyms for any object BANINST1 has select access to, but for which no public synonym exists. This second script is needed to create synonyms for new views.</td>
<td></td>
</tr>
</tbody>
</table>

**CAUTION:**

If you have made database changes at your site, this script may abort when it attempts to drop a database object that is not there or create a new database object that already exists.

Make sure ALL prerequisites specified in the Overview and Installation Requirements sections have been met before proceeding.

Anytime `gostage` fails, you should review the files in the `splpref` directory in order of date/time created. Typically, the most recent file will contain the error.

Please refer to the Banner_General_Object_List_8.11.1.pdf object list report to review the list of objects changed that are specific to the General product.

Invoke SQL*Plus and run the procedure:

```
sqlplus /nolog @gostage
```

If this step fails review the `listab1` listing file and refer to restart code A, otherwise, continue to part B.

Review: `listab1 listing`
Part B  In this part you will run the gurutlrp.sql utility script to compile database objects that are in an invalid state. The gurutlrp.sql script runs ORACLE’s utlrp utility script (as SYS) and then displays a list of the remaining invalid database objects. This script is run to shorten the time required to perform subsequent steps of this upgrade in which the rdbms would otherwise have to recompile invalid database objects as they are referenced. This will also enable the successful generation of all Oracle Forms modules which may not have successfully generated in the past because the rdbms considered the identifiers for invalid database objects to be insufficiently declared, but when the same modules were regenerated, the executables for those modules were created without error. All errors should be investigated before continuing to the next step.

To compile objects which are currently in an invalid state perform the following.

    sqlplus /nolog @gurutlrp

Review:  gurutlrp  listing

All errors should be resolved before continuing to the next step. You may need to repeat this process several times until all dependencies are validated.

NOTE:  The following Oracle error message may appear in the gurutlrp listing file. This is a known Oracle bug associated with Oracle’s utlrp.sql process and a patch is available (Oracle patch number - 7707103). Applying this patch will resolve this issue. However, the process will complete the validation process and you may continue with the upgrade.

    DECLARE
    *
    ERROR at line 1:
    ORA-00904: "FALSE": invalid identifier
    ORA-06512: at line 13

Step 6  Migrate from stage to permanent directories

Before executing any of the migration scripts make sure you are signed on to an operating system account that has write permission into the target Banner directories.

The migration scripts provided for the UNIX and MICROSOFT WINDOWS platforms expect your directory structure to match the one created by the Banner install process. You will have to modify the scripts if you chose a different directory structure. Migration scripts for other platforms are not provided due to their highly customized structures but you may use the GENMIGR.TXT file as a starting point for writing your own migration script.
In this step you will migrate the staged files to your permanent directories.

The file `GENMIGR.TXT` lists all files that need to be deleted from your permanent directories, and all files which should be copied from the staging directory to your permanent directories. The destination is indicated in UNIX format, and will be different on other platforms.

**UNIX**

The file `genmigr.shl` will do the appropriate removes, copies, and links. Review for correct directory path names and make sure that the environment variable `$BANNER_HOME` is set to the appropriate directory before executing.

```
NOTE: The genmigr.shl file defines a local variable, LN, at the top of the file which determines the type of links which should be used in the migration. This change enables clients who wish to use symbolic links, for example, to set LN='ln -s' instead of the default value of 'ln' so that the command ${LN} file $BANNER_HOME/links is translated to ln -s file $BANNER_HOME/links. Similarly, clients who wish to force the removal of any existing targets before linking files can set LN='ln -f'.
```

Note that even if your directory structure matches the baseline perfectly, some of the link commands will fail (that is, where the link currently exists). Other link errors may indicate that you had two copies of an object when the migration script was executed. This condition must be corrected. The duplication is probably between links and the product subdirectory.

You may wish to run the migration shell in background so that you may review any errors when it is complete. To submit into background and produce an error log, do the following:

1. If your operating system prompt is a percent sign, you are a cshell user. Enter the Bourne shell by typing:
   
   sh

2. Position to the staging directory for this product.

3. Run the migration script by typing:
   
   sh genmigr.shl >genmigr.log 2>&1 &

4. If you were a cshell user and want to return to that mode, press CTRL-D or type:
   
   exit

   Review: genmigr.log

   This file contains the results of the migration.
MICROSOFT WINDOWS

The file `genmigr.pl` will do the appropriate deletes and copies. Before running the migration script you must check the BANENV environment variable. This value may be determined by executing the SET command from the DOS prompt.

If BANENV has a value of REG, the value used for BANNER_HOME will be taken from the registry entry:

```
HKEY_LOCAL_MACHINE\SOFTWARE\BANNER\BANNER_HOME
```

If BANENV has a value of ENV, the value for BANNER_HOME will be taken from the environment variable BANNER_HOME.

Review the script for correct directory path names.

To run the migration script and produce an error log for the migration, do the following:

1. Position to the staging directory for this product.
2. Run the migration script by typing:

   ```
   perl genmigr.pl >genmigr.log 2>\&1
   ```

   Review: `genmigr.log`
   
   This file contains the results of the migration.

---

**Step 7 Compile COBOL programs**

This step does not apply to this upgrade.

**Step 8 Compile C programs**

This step does not apply to this upgrade.

**Step 9 Apply required data changes**

This step does not apply to this upgrade.

**Step 10 Generate Oracle Forms**

This step does not apply to this upgrade.

**Step 11 Generate Oracle Reports**

This step does not apply to this upgrade.
Step 12  Compile Java Programs

This step does not apply to this upgrade.

Step 13  Update letter generation/population selection tables

This step does not apply to this upgrade.

Step 14  Update referential integrity constraints

This step does not apply to this upgrade.

Step 15  Restart the gostage process

This step does not apply to this upgrade.

Step 16  Verify the state of the upgraded environment

Part A  In this part you will run the gurscls.sql utility script to synchronize the objects with their associated classes. All objects belong to a Banner class (as defined in the BANSEC.R.GTVCLAS table) and all objects have an associated role. This routine checks every person enrolled in the class to make sure they have been given execute privilege to every role used by any object in the class.

Invoke SQL*Plus and run the procedure:

```
sqlplus bansecr/password <Enter>
start gurscls <Enter>
```

Part B  In this part you will run the grudone.sql utility script to verify that all of the loaded modifications for this release have been applied. Gostage did not execute any of the rows that are displayed by this script.

Invoke SQL*Plus and run the procedure:

```
sqlplus upgrade_owner/password <Enter>
start grudone <Enter>
```

Review: grudone listing

Part C  In this part you will run the gudv81101.sql utility script to insert the release version into the Version History Table.

Invoke SQL*Plus and run the procedure:

```
sqlplus general/password <Enter>
start gudv81101 <Enter>
```

Review: gudv81101 listing
Part D In this part you will run the `grestrigs.sql` which restores BANSECR security audit settings back to pre-upgrade status which were altered by the `gruready` script run at the beginning of this upgrade.

**WARNING:** If you are running upgrades in parallel, do NOT run this part until you have finished all of your upgrades. The reason is that the actual pre-upgrade Banner Security Audit trigger status is saved in the upgrade where Step 4, Part A was applied first. You will have to position yourself in the stage directory for the product upgrade that was applied first and then execute this Part to restore the audit triggers to their pre-upgrade status. Not following this procedure could have adverse effects on restoring the actual pre-upgrade audit trigger status.

Invoke SQL*Plus and run the procedure:

```
sqlplus /nolog @grestrigs
```

Review: `grestrig` listing

Part E In this part you will run the `gresroled.sql` to restore the original roles for those accounts modified by the `gruready` script run at the beginning of this upgrade.

**WARNING:** If you are running upgrades in parallel, do NOT run this part until you have finished all of your upgrades. The reason is that several accounts are used by all upgrades. If they do not have the DBA role as a default role, this script will reset their privileges so DBA is not included. This could have adverse effects on the other upgrades.

Invoke SQL*Plus and run the procedure:

```
sqlplus /nolog @gresroled
```

Review: `gresrole` listing

This Banner General 8.11.1 upgrade is now complete.
Index

B
backup 5

C
C compiles 18
COBOL compiles 18
compile
  C 18
  COBOL 18
Compile
  Java 19
Compile Java Programs 19
  constraints 19

F
Forms
  generation 18

I
installation
  requirements 6

L
letter generation/population selection 19
login.sql 7

M
migration

O
Oracle Forms 18
Oracle Reports 18
  overview 3

P
population selection/letter generation 19

R
referential integrity 19
Release Guide 5
Reports
  generation 18
  requirements
    installation 6

S
splpref 7
spool prefix 7

T
Temporary Tablespace Alert 6

V
VPD 13, 16
Ellucian
2003 Edmund Halley Dr
Reston, VA 20191
United States of America