Ellucian Degree Works

Install, 5.1.2

February 21, 2024

Notices and Privacy

In preparing and providing this publication, Ellucian is not rendering legal, accounting, or other similar professional services. Ellucian makes no claims that an institution's use of this publication or the software for which it is provided will guarantee compliance with applicable federal or state laws, rules, or regulations. Each organization should seek legal, accounting, and other similar professional services from competent providers of the organization's own choosing.

Ellucian's Privacy Statement is available at: www.ellucian.com/privacy.

Ellucian shall have the right to (a) use, store, process, modify, reproduce, distribute and display customer data, and to grant sublicenses to third parties, for the sole purposes of providing the software, performing Ellucian's obligations under its agreements with customers and complying with applicable law or legal requirements; (b) use, store, process, modify and reproduce customer data for Ellucian's internal business purposes, including development, diagnostic, forecasting, planning, analysis and corrective purposes in connection with the software, and for otherwise improving and enhancing the software; and (c) use, store, process, modify, reproduce, display, perform, distribute, disclose and otherwise exploit in any manner Aggregated Data for Ellucian's business purposes, including disclosure within its public statements and marketing materials describing or promoting Ellucian or the software. “Aggregated Data” means any data obtained or generated by Ellucian, including data pertaining to the software, Ellucian's systems and software, and the use of any of the foregoing, and includes data derived from customer data, which in all instances (i) does not identify any individual and (ii) is not attributed or attributable to a specific customer. Aggregated Data includes data that has been combined into databases which include third party data.

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

Contents

Degree Works Installation

Updated: September 29, 2023

This release supports a new installation or upgrade for existing customers currently on release 5.1.0 or 5.1.1. System requirements, prerequisites, configuration instructions, and the steps to process Release 5.1.2 are explained in the installation documentation.

Click Attachments on this page to download the installation documentation.

Banner Considerations

Banner Setup Checklist

UCX settings

Updated: March 25, 2022

Review and change UCX settings using Controller.

UCX	Action
UCX-SCR001 STATUS	Change Description to Student Type.
UCX-SCR001 SCHOOL	Change Description to Level.
UCX-SCR001 LEVEL	Change Description to Student Class Level.
UCX-CFG020 BANNER	Be sure the Banner Site flag is Y. You can turn on the Search in Banner flag now but it is suggested that you perform all searches against the bridged data at first while getting everything setup. Leave this flag as N for now but switch it to Y later in your implementation.
UCX-CFG020 REFRESH	Set the flags to appropriately configure the refresh parameters.
UCX-CFG020 SEARCH	Set the Show flags to N for items you are not using. Show Specialization and Show Liberal Learning should be N because these are not used in Banner. Note - School in Degree Works is actually the same as Level in Banner.
UCX-STU352 DISCIPLINE STATUS	Set the Discipline Status flag to I for those disciplines that are Inactive. If a Discipline is Inactive then the class (current, historic and transfer) will be skipped and NOT bridged to the Degree Works rad_class_dtl.
UCX-STU385 IN PROGRESS Flag	The default In-Progress flag is N for historical classes found in SHRTCKN. If an historical

UCX	Action
	grade combination in UCX-STU385 (key = School/GradeType/Grade) is considered In-Progress, then set this In-Progress flag to Y so that the rad_inprog_flag on the rad_class_dtl gets built appropriately with a Y value.
UCX-STU385 OVERRIDE Flags	There is a master Override flag and 8 Override values.
UCX-STU385 Transfer Repeat Flags	There are three Transfer Repeat fields that may be used to override the standard Transfer Repeat Pointer and Repeat Policy used for repeated transfer classes.
UCX-BAN080 Custom Data	For special if statements and for any special data desired on the worksheet setup UCX-BAN080.

Banner data extracts

People selection

Updated: March 24, 2023

Use Controller to modify the listed SQL settings as required, to select the desired users you want to bridge to Degree Works.

A set of sql is delivered in these settings, but you must modify the sql to suit your needs.

```
integration.banner.extract.student.sql.daily
integration.banner.extract.applicant.sql.daily
integration.banner.extract.adv.sql.daily
integration.banner.extract.advx.sql.daily
integration.banner.extract.reg.sql.daily
```

The queries in these files should only select the IDs of the individuals you want to bridge into Degree Works. When the student extract is run, for example, the sql from integration.banner.extract.student.sql.daily is executed and the resulting IDs are then processed by the extract. When running the non-student extract, RAD33, you must specify the user class of the users you want to extract. The job will either use the IDs you specified or will lookup the sql.daily setting based on the user class selected. For example, if you choose REG when running the job the integration.banner.extract.reg.sql.daily setting will be used to select the users. You can create additional sql.daily settings in Controller for the other user classes you use.

You should test the select statements that you use in these settings through a database tool, to verify that the count you get is as expected.

People deletion

Updated: March 25, 2022

Individuals can be deleted from Degree Works by either running RAD40 in Transit or running deleteid through the extract from the command line.

To delete data from Degree Works, use the RAD40 processor in Transit. You can provide the IDs or use selection criteria to identify the individuals that should be deleted, and have the option of either deleting all data or just the data bridged from the student system.

Alternatively, create an ID file of the individuals that should be deleted from Degree Works and run the extract. You must specify the relative or absolute path to the file.

```
$ bannerextract deleteid deleteFile.ids
```

UCX entries selection

Updated: September 30, 2022

You can run RAD36 in Transit to populate the UCX tables, but it will run the extract on all validation tables. There is no way to select a subset of tables.

If selected UCX tables are to be re-extracted from Banner to Degree Works, on the Classic server, create a UCX file containing a list of Degree Works tables. For example, write one table per line:

```
STU356
STU385
STU560
```

Add a .ucx extension to this file in the current directory. The actual file name can be any valid Unix file name. However, it must have the .ucx extension. For example, the file bantables.ucx can reside in your current directory or you can specify the full or absolute path to the file. The banner extract command would be:

```
bannerextract ucx bantables.ucx
```

When the command above is executed, only the Degree Works tables listed in that file will be re-loaded with Banner data. Make sure to check the setting for the UCX-CFG020 RADBRIDGE **Add UCX Entries Only** flag. Make sure it is set to **N** if ALL entries are to be reloaded from Banner. If only NEW entries are to be loaded from Banner, make sure this UCX-CFG020 flag is set to **Y**. It is recommended that this flag always be set to **Y**.

Extract record selection configuration

Updated: September 30, 2022

Modify the integration.banner.extract.config setting as needed to select a different set of records than those Degree Works is selecting by default.

Make sure to review this setting and make all appropriate changes for your site.

If your Banner database has been configured for MEP (multiple-entity processing), see instructions in the Enable multiple-entity processing for Banner topic.

Nightly extracts

Updated: March 24, 2023

It is recommended to run a nightly student extract to ensure your student data is current. Other extracts can be run as needed.

Set up the launchjob script in cron. See The launchjob script for more information on running the launchjob script through cron.

In case launchjob encounters an error executing, be sure to redirect launchjob output to a log. Also be sure that stderr is redirected to stdout, either on the same line or elsewhere. For example:

```
launchjob --wait $ADMIN_HOME/myjobs/rad30.standardSql.json >> $LOGFI
LE 2>&1
launchjob --wait $ADMIN_HOME/myjobs/rad32.standardSql.json >> $LOGFI
LE 2>&1
launchjob --wait $ADMIN_HOME/myjobs/rad31.standardSql.json >> $LOGFI
LE 2>&1
```

It is best to use the **--wait** option when running extracts through cron to ensure that the next job will only start after the previous job is finished. You don't always need to do this but it is safer because some extracts do use data from the previous extract. For example, the student extract uses data from the course, ucx, ets, and equiv extracts. Without the **--wait** option specified the launchjob script will place the request to run the job on the queue and returning immediately. This means that multiple jobs will run in parallel and problems could occur.

It is recommended that you set up only the student, applicant, and advisor extracts to run each night in cron. It is best to run the other extracts on an as-needed basis in Transit, or run one time per month/term through cron. These are the other extracts:

- RAD34 Banner Course extract
- RAD35 Banner Curriculum Rules extract
- RAD36 Banner Validation extract (UCX)
- RAD37 Banner Transfer School extract (ETS)
- RAD38 Banner Equivalencies extract
- RAD39 Banner Transfer Equivalency extract (Mappings)

Post extract

Updated: March 25, 2022

After UCX has been extracted from Banner, you should review each table and field setup as needed.

Be sure not to run UCX extract again, unless the UCX-CFG020 RADBRIDGE Add UCX Entries Only is set to Y so that none of the records you changed will be deleted and only new records in Banner will be added to the UCX.

Table	Fields to set up
UCX-AUD027	Filter fields
UCX-STU016	Planner and Show in Transfer Equivalency Self-Service flags and the Financial Aid Year and Term Type fields
UCX-STU035	Planner flag
UCX-STU307	Short degree field
UCX-STU346	Calendar codes used in Transfer Equivalency Self-Service
UCX-STU352	Discipline Status (“I” – Inactive)
UCX-STU385	GPA Calc flag
UCX-STU385	In-Progress flag
UCX-STU385	Override flags
UCX-STU385	Override Transfer Repeat flag/fields

Degree Works Data Extract

Degree Works data extract configuration and installation

Updated: March 24, 2023

The Degree Works Data Extract process for Banner clients uses embedded SQL tools to extract the data

These programs execute on the Degree Works application server – Linux, UNIX, and so on. Ellucian has developed several extract programs to not only extract the required data but to also convert the data to the Bridge Interface Format (BIF) so that it is ready to load into Degree Works.

Verify or perform the following steps to configure your Degree Works server to perform the Banner Degree Works Data Extract.

STEP 1 - Banner Database login

Ensure that the DB_LOGIN_BANNER environment variable is set correctly:

```
$ env | grep DB_LOGIN_BANNER
```

If this does not show the correct Banner database login, you need to edit this variable in `$DGWBASE/dwenv.config`, run **setdbpasswords**, and log back in to your host session. Here are examples of how to set this variable:

```
export DB_LOGIN_BANNER=userid
export DB_LOGIN_BANNER=userid@some.other.machine.edu
```

If the database is on a remote machine, you can use the @ option to specify where the database resides. You must have the remote database server configured in Oracle's tnsnames.ora file.

To set the password for your Banner user run **setdbpasswords**:

```
setdbpasswords --sispassword thebnrpassword
```

Ensure the ORACLE_SID environment variable is set correctly:

```
$ env | grep ORACLE_SID
```

If this is not set correctly, review your setup of `$DGWBASE/dwenv.config`.

If you are connecting to a Banner database that has been configured as MEP (Multi-Entity Processing), you must create a DB_LOGIN_BANNER user for each MEP entity established in Degree Works. For more information, see the Enable multiple-entity processing for Banner topic.

STEP 2 - Set up UCX-CFG020 RADBRIDGE

Review all settings before running the UCX extract. For more information, see Technical Configuration Reference. Specifically, make sure the **Add UCX Entries Only** flag is set to **Y** before running the UCX extract (RAD36).

STEP 3 - Extract UCX and courses

Before attempting to bridge student data from Banner into Degree Works, it is recommended to first extract Validation codes (UCX) and course information. These data are required before Scribe can be used to add or modify requirement blocks in your Degree Works database.

In Transit, launch these jobs:

- RAD34 course extract
- RAD36 UCX extract

STEP 4 - Set up UCX-CFG020 BANNER

Review all settings before running the student extract.

Repeat Policy

Although you can use repeat policies 1-6 telling Degree Works to use the class with the best grade or the most recently taken class, it is recommended that you set the **Repeat Policy** flags to **B** for all Repeat Indicators.

When B is used this behavior will occur:

- Excluded classes will end up in the Insufficient section of the audit but they will not affect the overall GPA or credits.
- Averaged classes will end up in the Insufficient section of the audit and they will affect the overall GPA but will not be counted in the overall credits towards degree.
- Included classes will apply to rules as normal classes affecting the GPA and total credits.

By using the B setting, you are telling Degree Works to handle each class based on the indicator without regard to grades or terms taken as the decision about which classes should be counted and not counted has already been made and recorded using the indicator. When B is in use the normal Degree Works repeat logic is skipped simplifying the auditing process greatly.

The six different **Repeat Policy** fields should be set to the same value. Only in rare scenarios does it make sense for these to be different.

STEP 5 - Set up SQL select settings

Review or modify the sql in the integration.banner.extract.*.sql.daily settings using Controller before launching the RAD3x processors. Make sure that the number of ID codes selected by the sql statement is what is expected and that they are the correct ID codes. In addition, ensure that you use SELECT DISTINCT(SPRIDEN_ID) when selecting users so that the same ID does not appear more than one time in the results.

Note: If a Banner table is used in one of the sql settings identified below that is not found in the Database table access topic, make sure to have your database administrator add the appropriate access. Otherwise the ID codes will not be extracted correctly and the Banner extract will fail.

Review and modify as needed the setting used to select the students you want bridged to Degree Works:

```
integration.banner.extract.student.sql.daily
```

Review and modify as needed the setting used to select the applicants you want bridged to Degree Works:

```
integration.banner.extract.applicant.sql.daily
```

Review and modify as needed the settings used to select the non-student users you want bridged to Degree Works. Additional settings can be created in Controller for each additional user class you use.

```
integration.banner.extract.adv.sql.daily
integration.banner.extract.advx.sql.daily
integration.banner.extract.reg.sql.daily
```

STEP 6 - Set up integration.banner.extract.config setting

Using Controller, review and modify as needed the integration.banner.extract.config setting to select a different set of records based on your particular needs. The SQL FROM/WHERE clauses for every Banner table used by the Banner extract programs are included in this configuration file.

The config should look like the example you see here – the FROM/WHERE text for your school may need to be changed.

```
# SGBSTDN must be a; AND is required at the end of the WHERE
SGBSTDN-from:  FROM SGBSTDN a
SGBSTDN-where: WHERE a.SGBSTDN_TERM_CODE_EFF =
SGBSTDN-where:       (SELECT MAX(b.SGBSTDN_TERM_CODE_EFF)
SGBSTDN-where:        FROM SGBSTDN b WHERE b.SGBSTDN_PIDM = a.SGBSTD
N_PIDM)
AND
#                    a.SGBSTDN_PIDM = <students-pidm>
```

A special set of records with keys of CALCFCN-from: and CALCFCN-where: have been created for the special Banner function: F_CLASS_CALC_FCN. This function call is made by the banner student extract using the PIDM, LEVL_CODE and TERM_CODE specified in this setting to generate the student's Class Standing code that is loaded into the rad_stu_level on the rad_goal_dtl. A default set of special records with a key of CALCFCN are included in this configuration file and are loaded with the FROM and WHERE clauses from the SGBSTDN default entry. Change this set of CALCFCN records as appropriate for your site.

Here is an example of the SORLCUR/SORLFOS entries in this configuration file:

```
########## SORLCUR - LEARNER CURRICULUM (ban40) ##########

# SORLCUR must be a; AND is required at the end of the WHERE
SORLCUR-from:  FROM SORLCUR a
SORLCUR-where: WHERE a.SORLCUR_CACT_CODE = 'ACTIVE'
SORLCUR-where:   AND a.SORLCUR_SEQNO     =
SORLCUR-where:    (SELECT MAX(b.SORLCUR_SEQNO) FROM SORLCUR b
SORLCUR-where:      WHERE b.SORLCUR_PIDM        = a.SORLCUR_PIDM
SORLCUR-where:        AND b.SORLCUR_PRIORITY_NO = a.SORLCUR_PRIORITY
_NO
SORLCUR-where:        AND b.SORLCUR_LMOD_CODE   = 'LEARNER')
SORLCUR-where:   AND
#                    a.SORLCUR_PIDM = <students-pidm>

########## SORLFOS - STUDENT FIELD OF STUDY (ban40) ##########

# SORLFOS must be a; AND is required at the end of the WHERE
SORLFOS-from:  FROM SORLFOS a, SORLCUR b
SORLFOS-where: WHERE b.SORLCUR_CACT_CODE  = 'ACTIVE'
SORLFOS-where:   AND b.SORLCUR_SEQNO =
SORLFOS-where:    (SELECT MAX(c.SORLCUR_SEQNO) FROM SORLCUR c
SORLFOS-where:      WHERE c.SORLCUR_PIDM        = b.SORLCUR_PIDM
SORLFOS-where:        AND c.SORLCUR_PRIORITY_NO = b.SORLCUR_PRIORITY
_NO
SORLFOS-where:        AND c.SORLCUR_LMOD_CODE   = 'LEARNER')
SORLFOS-where:   AND a.SORLFOS_CSTS_CODE  = 'INPROGRESS'
SORLFOS-where:   AND a.SORLFOS_CACT_CODE  = 'ACTIVE'
SORLFOS-where:   AND a.SORLFOS_PIDM       = b.SORLCUR_PIDM
SORLFOS-where:   AND a.SORLFOS_LCUR_SEQNO = b.SORLCUR_SEQNO
SORLFOS-where:   AND
#                    a.SORLFOS_PIDM = <students-pidm>
```

Batch data extract process

Updated: September 30, 2022

The batch extract programs are executed one of two ways: a script named launchjob run through cron, or through Transit's RAD3x family of jobs.

When extracting students Degree Works automatically runs a new degree audit for each of the students that has changed data as of the last time they were extracted. In doing this you will be sure that each student's degree audit reflects any changes made to their student record.

Student and applicant academic data are required for Degree Works to generate a Degree Audit. Ellucian extracts student data, applicant data, advisor information, your course catalog and the list of transfer institutions from the Banner database and stores this data in the Repository for Academic Data (RAD) database tables. This procedure is known as the Degree Works Bridge process and consists of the following data extraction processes:

Students - active students' academic data are extracted based on the SQL specified in the integration.banner.extract.configsetting. Student data can also be extracted individually by the SPRIDEN ID.

Applicants - admissions applicants academic data may be extracted based on the SQL specified in the integration.banner.extract.config setting. However, only unique combinations of the Level (School) and Degree may be extracted (this assumes the UCX-CFG020 BANNER configuration flags are set appropriately or the APPLICANT data extract is performed). Applicant data may also be extracted individually by the SPRIDEN ID.

Non-students - access records are created for advisors, deans, and other non-student user classes who require access to Degree Works.

Course Catalog - all current courses from your course catalog.

Course Equivalents - historic courses and their current equivalent courses.

Curriculum Rules - Curriculum Rules used by What-if audits.

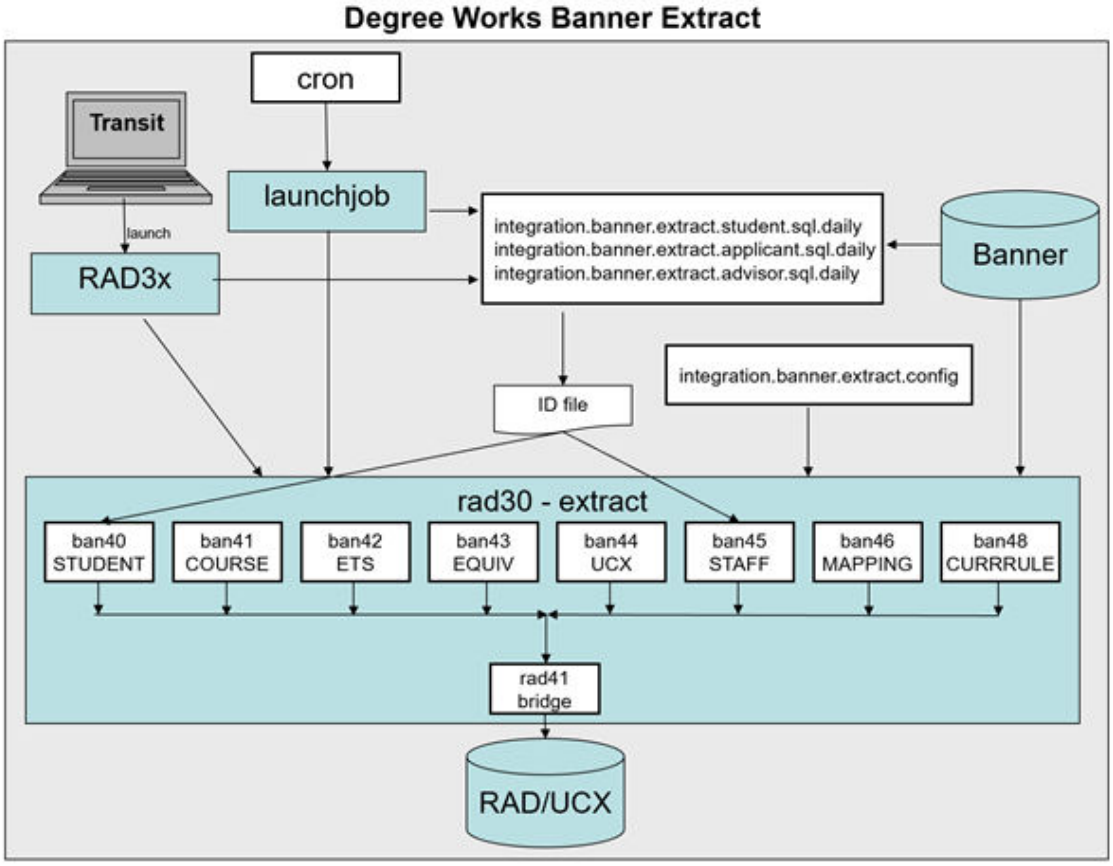
ETS - Transfer institution codes, names, and identification data.

UCX Validation Codes - Validation tables consisting of data from Banner STV tables.

Mappings - Transfer Articulation Mappings for import into Transfer Equivalency Admin and Transfer Equivalency Self-Service. This mapping information can also used by Course Link.

The ETS extract must be run before the mappings extract.

Banner Extract flow diagram



Extract scheduling

Updated: March 24, 2023

The launchjob script can be used to schedule the extracts to run through cron..

The launchjob script can be used to schedule the extract to run using cron, or it can be run directly at any time during the day as needed. However, there are some cases when the bannerextract script may still need to be used. Note that the launchjob script requires a parameter file in JSON format. When running any of the extracts manually you should use Transit; the launchjob script is intended to be used when scheduling jobs in cron.

After running launchjob the resulting log file can be viewed in Transit. After running bannerextract the log file can be viewed in $ADMIN_HOME/dgwspool.

The launchjob script only submits a request for the job, it does not run the job itself. The job goes into the queue and is run by one of the Transit Executors. It does not usually wait for the job to complete. It can be told to wait for the job by using the `--wait` parameter. When this is used, the command polls the Transit API, pausing until the newly launched job completes. Only then will it

exit. This can be used in a cron script to prevent other commands from executing when they depend on the extract. There is also a jobstatus command that can be used in a script to obtain the status of a job. It also has a `--wait` parameter that acts similarly. See Transit Administration for more information on the launchjob and jobstatus commands.

Student

The sample JSON job files in app/samples should be copied to a new $ADMIN_HOME/myjobs directory that you create. You then modify the files in your myjobs directory as needed.

To run the student extract using the default integration.banner.extract.student.sql.daily setting, your JSON file needs to be set up like this:

```
{
        "name": "RAD30",
        "parameters": [{
                        "type": "SELECTDEFAULTQUERY",
                        "usingDefaultQuery": "true"
                },
                {
                        "type": "QUESTIONS",
                        "answers": [{
                                "id": "rad30.forceNewAudits",
                                "value": "false"
                        }]
                }
        ]
}
```

By default new audits are generated only for students with data changes coming from Banner. If you want to generate audits for all students regardless of data changes being detected, you can set the **rad30.forceNewAudits** flag to true. You then specify your JSON file when running launchjob:

```
$ launchjob $ADMIN_HOME/myjobs/rad30.defaultStudentSql.json
```

You can also specify a list of student IDs to be extracted. If you are using a JSON file, it would look like this:

```
{
        "name": "RAD30",
        "parameters": [{
                "type": "SELECTIDLIST",
                "ids": ["studentId1",
                        "studentId2",
                        "studentId3",
                        "etc"
                ]
        }, {
                "type": "QUESTIONS",
                "answers": [{
                        "id": "rad30.forceNewAudits",
                        "value": "false"
```

```
                    }]
            }]
    }
```

However, if you have a file of student IDs, it is probably easier to use Transit to import the IDs instead of creating a JSON file like the one above. You would need to create a JSON file only if, for some reason, you needed to load the IDs in your cron job or if you could not access Transit.

You would then specify your JSON file containing the IDs when running launchjob. For example:

```
$ launchjob $ADMIN_HOME/myjobs/rad30.myids.json
```

To extract a single student ID it is easiest to use Transit or the refresh button on the dashboard.

Applicant

To run the applicant extract using the default integration.banner.extract.applicant.sql.daily setting your JSON file needs to be set up like the one above for students using the default SQL file except change the name to RAD32.

You then specify your JSON file when running launchjob. For example:

```
$ launchjob $ADMIN_HOME/myjobs/rad32.defaultApplicantSql.json
```

Please see the student example above on creating a JSON file to run a list of IDs. To run a list of applicant IDs you use the same JSON file format except you need to change the name property to RAD32.

Staff

The RAD33 job is used to extract non-students. This includes, advisors, deans, and so on. When running RAD33, you must specify the user class. When usingDefaultQuery is true in the JSON file, the extract will use the sql.daily setting based on the user class you specify. For example, if the user class is defined as ADVX in the JSON file, the extract will use the setting named integration.banner.extract.advx.sql.daily to select this pool of users and assign them the ADVX user class. You can create new sql.daily settings in Controller for each user class from AUD012 that you use. A few non-student sql.daily settings are delivered to you by default, but each needs to be reviewed and changed as needed.

See the sample RAD33 JSON files in the $DGWHOME/samples directory to see how to run RAD33 using either the default SQL or by providing a list of IDs.

Deleting IDs

To delete unwanted IDs from the Degree Works database, you can create an ID file with just those individuals and import the file into the RAD40 Student Data Delete Processor in Transit

When running RAD40, you can choose to delete the audits, plans, notes, and exceptions for a student in addition to all bridged data by checking the **Delete non-bridged data** check box.

Selected UCX tables

To run the ucx extract using a list of Degree Works UCX tables listed in a file with a .ucx extension (do NOT use with cron):

```
$ bannerextract ucx someucxtables.ucx
```

Warning! Do not put the UCX_prefix on the table names even though the actual database table names are UCX_STU352, UCX_STU560, and UCX_STU563.

For example, the someucxtables.ucx file might contain tables:

```
STU352
STU560
STU563
```

In this case, ONLY these 3 UCX tables will be re-extracted from Banner.

Note: The UCX-STU563 table (Concentrations) is re-created from the STVMAJR Banner table, which also is used to recreate major tables (UCX-STU023 and UCX-AUD027) and minor tables (UCX-STU024 and UCX-AUD029). However, only UCX-STU563 will be extracted with this banner extract.

Make sure to check the setting for the UCX-CFG020-RADBRIDGE **Add UCX Entries Only** flag BEFORE running the UCX bannerextract. Make sure it is set to N if ALL entries are to be reloaded from Banner. If only NEW entries are to be loaded from Banner, make sure this UCX-CFG020 flag is set to Y. Most of the time you will want to keep this flag set to Y so existing entries and their values are not lost.

In the above example, UCX-STU352 is being re-extracted. It has a **Discipline Status** flag in it, which is manually input using Controller. These updates will need to be made again if the entire UCX-STU352 is re-extracted.

Other modes

The other extract modes do not have associated sql files. These modes extract ALL records from the Banner database and load them into the associated Degree Works database tables. However, you may modify the relevant sections in the integration.banner.extract.config setting to control exactly which records are bridged into Degree Works.

These other modes do not have parameters so the JSON file you would use with the launchjob script would look like this with the appropriate RAD3x name specified:

```
{
  "name": "RAD3x"
}
```

Each job can be run manually in Transit. The JSON file is required when running these jobs in cron.

- Course – RAD34 - only adds/updates rad_course_mst records, but deletes and re-adds rad_crs_attr_dtl records.

- Curriculum Rules – RAD35 - the old curriculum rules are first deleted and then new curriculum rules are re-added.
- Equivalences – RAD38 - the dap_eqv_crs_mst, UCX-CFG070, and UCX-CFG073 are first deleted and all equivalencies and cross-listings are re-added.
- ETS (transfer schools) – RAD37 - only adds/updates rad_ets_mst records (no deletes).
- Mappings – RAD39 - the old mappings are first deleted and then new mappings are re-added.
- UCX (Validation tables) – RAD36 - if UCX-CFG020 RADBRIDGE Add UCX Entries Only = N, all records are deleted and re-added. If UCX-CFG020 RADBRIDGE Add UCX Entries Only = Y, only new records will be added (no updates).

Transit - RAD30

Updated: March 24, 2023

Transit allows you to run the RAD30 processor by ID or by SQL selection.

For students you can use the default integration.banner.extract.student.sql.daily setting or you can enter individual IDs or use an ID file containing the pool of students that you want to import into Degree Works. There is also a method to submit RAD30 through cron, using the launchjob script. For more information, see the Extract scheduling topic. The launchjob script will submit your job to Transit so that the resulting reports or debug files can be retrieved through the TransitUI. For more information on Transit and launchjob, see Transit Administration.

transit.rad30.workerCount

The RAD30, RAD32, and RAD33 jobs will use the transit.rad30.workerCount setting to split up the file of student, applicant, advisor, or staff IDs into multiple files.

After multiple ID files are created, the scripts are then able to launch multiple processes on each of the files. In theory, with this workerCount set to 4, the task gets done in a quarter of the time.

Use Controller to change your setting. You need to determine what value works best for the resources available on your system.

Bridge audits

Updated: March 25, 2022

When the extract finishes loading student data a list of students with changed data is created.

This list of students is then passed onto DAP22 to have audits run to ensure the latest audits reflect the data changes. You can control how these audits are created by modifying the `integration.bridge.audits` settings in Controller. See integration.bridge.audits for information on these settings. A separate dap22 log file is created for this process and can be accessed through Transit. You can review it to see how many students had data changes and to check the status of those audits.

Dynamic Data Extract process

Updated: March 24, 2023

In addition to extracting student data through the batch extract, you can extract data for a single student using the dynamic method.

There are two ways in which a dynamic extract may take place: the first is available from within the Degree Works web application as a button, and the second occurs whenever a triggering event takes place.

The first way of performing a dynamic extract from Banner is to push a button. Users with the SDREFRES key will be shown the **Data refreshed** field in the student context area at the top of the main web page. This is the last time the student's academic data was copied from Banner into Degree Works. When the user also has the SDREFBTN key, they will be able to click a button to refresh the student.

You can hover over the **Data refreshed** field to see the Data last changed date. This is the last time the student's data was changed, which may be older than the last time it was refreshed. At any time users can click the on-demand refresh button to again pull in this student's data. Users may want to do this after a major or grade change was made in Banner and they don't want to wait for the nightly batch extract. After the extract process completes the user will get a confirmation message and the Last Refresh date/time will be updated. A new audit is automatically created.

The second way of performing a dynamic extract involves a triggering event, which may occur when any user requests an audit from the Worksheets, What If, Planner, Exceptions, or Look Ahead tabs. The UCX-CFG020 REFRESH record in Controller is used to control this behavior.

Audit generation with dynamic refresh

Updated: September 30, 2022

Turning on the Dynamic Refresh with all or some of the other flags turned on will affect performance – both for all users of the system and the particular users attempting to run an audit.

Before Degree Works runs the requested audit it refreshes the student's academic data from Banner – this takes extra time and will consume additional system resources. When a refresh occurs through running an audit the user is not notified – but the Last Refresh date/time is updated.

The **Refresh timeout in minutes** in CFG020 Refresh is a mechanism you may use to inhibit repetitive dynamic refreshes for students that are incidental and not necessary, for instance when multiple What-Ifs are being launched within a short time span. You may specify a length of time during which no additional refreshes of data would be appropriate. This timeout setting is used in conjunction with the refresh date/time stored on the rad-primary-mst to tell Degree Works if it should re-read the student's data in Banner.

As with the batch extract process, the Banner student extract and RAD bridge routines are used to perform the dynamic refresh. The bridge does check for changed data and will skip the insertion of records if no changes exist, but it will update the refresh date/time on the rad-primary-mst and the bridge date/time shown on the dashboard either way.

Audit viewing with dynamic refresh

Updated: September 30, 2022

With the **Refresh before viewing an Audit** flag set to **Y**, the system will check the timeout setting, as with running an audit, and will execute the extract as needed.

If there are data changes, a new audit will be run. In addition, the audit and refresh date and times will be updated in the dashboard, reflecting what has just occurred. When an extract is processed while in View mode, the bridge date and time are not updated if there are no real data changes, which is different from when in Run mode.

Banner Database Considerations

Banner database configuration

Updated: September 30, 2022

Several configuration steps are needed to connect to a Banner database from Degree Works.

Procedure

1. Create a dwmgr user in the new Banner database with select privileges to the list of tables identified in this document. You must run the two grants scripts, bannergrants.sql and bannergrants2.sql in the Banner database (both are in `$DGWHOME/sql`.) (bannergrants2 must be run by sysdba as it grants access to Oracle queues required for pre-requisite checking).

   Note: Typically the username is dwmgr, but you can use any name you like. If you choose a different name, then use that name in the instructions below instead of dwmgr.

2. On the Degree Works server, update the `$ORACLE_HOME/network/admin/tnsnames.ora` entry and add the new Banner database.
3. On the Degree Works server, edit the `dwenv.config` file and modify the following line: `export DB_LOGIN_BANNER=` and set the value to `export DB_LOGIN_BANNER=dwmgr@service`.
   Where dwmgr is the user defined in the new Banner database and service is the service name from tnsnames.ora. Keep this change by saving the `dwenv.config` file.
   To set the password for your Banner user run `setdbpasswords`:
   `setdbpasswords --sispassword thebnrpassword`
4. Log in again to the Degree Works server so that the new DB_LOGIN_BANNER variable is set. To check, issue the following command: `env | grep DB_LOGIN_BANNER`. You should see the new entry from `dwenv.config`.
5. Test your new Banner connection by typing (as the dwadmin user logged into the Degree Works server): dbb.

Additionally, SQL*Plus should be launched in the Banner database. To verify that you are looking at the correct Banner database issue: `SQL> select * from global_name;`.

6. Restart the web, pre-req and dap daemons using the `webrestart`, `preqrestart` and `daprestart` commands.
7. Determine if the data in the following tables in the new Banner database are different from the Banner database you originally used to populate Degree Works. If so, rerun the associated processes so data from the new Banner database is used to populate Degree Works.

SCBCRSE, SCRATTR, SCRRTST	rerun course extract
SCREQIV, SCBCRKY, STVCSTA	rerun equiv extract
SHRGRDO, SHRGRDE	rerun ucx extract for UCX-STU385 rerun ucx extract for UCX-STU356
STVCLAS	rerun ucx extract for UCX-STU305
STVCOLL	rerun ucx extract for UCX-STU560
STVDEGC	rerun ucx extract for UCX-STU307
STVGMOD	rerun ucx extract for UCX-STU356
STVLEVL	rerun ucx extract for UCX-STU350
STVMAJR	rerun ucx extract for UCX-STU023, UCX-STU024, UCX-AUD027, UCX-AUD029, UCX-STU563
STVSTYP	rerun ucx extract for UCX-STU306
STVTERM	rerun ucx extract for UCX-STU016, UCX-STU035
STVACYR	rerun ucx extract for UCX-STU035

Before running the UCX extract you should review the **Add UCX entries only** flag on UCX-CFG020 RADBRIDGE. If you already have entries in these UCX tables that you do not want to lose you should set this flag to Y so that only new entries that do not exist in the UCX tables are added but existing entries are not touched.

After the ucx extract has been executed, issue a `daprestart` and `webrestart`.

Enable multiple-entity processing for Banner

Updated: September 29, 2023

Banner sites wishing to enable multi-entity processing (MEP) in Degree Works must first configure table sharing in the `dwschema.xml` and `share.xml` files which reside on the Degree Works administrative server.

About this task

After your multi-entity table sharing has been established, you must then ensure that each campus entity in Degree Works has been configured to extract the correct data from Banner. Each of your

Degree Works campus entities will have unique `$ADMIN_HOME` and `$LOCAL_HOME` directories on the Degree Works administrative server. You should be able to use the `newenv` command to switch from one MEP entity to another on the Degree Works server. The following changes must be made to each campus entity environment.

Procedure

1. The integration.banner.extract.student.sql.daily setting should be configured, using Controller, by each campus entity to select the Banner IDs of only their matriculated students. For example, the CAMP_CODE column on the SORLCUR table is a very common way to find active students from a particular campus. Writing SQL to select only those students matriculated at a specific campus entity will assure that degree audits are generated for only these students and not for any student within the multi-campus institution.
2. Each campus entity must also maintain its own setting, using Controller. For each student selected, the SQL in the integration.banner.extract.config setting will be run to select only those records of interest to this college. For example, if a student is matriculated at multiple campuses, the setting should be modified so that only the data for a specific campus are read, which results in the correct degree information for that particular campus.
3. If you are connecting to a Banner database which has been configured as MEP (Multi-Entity Processing), you must create a unique Degree Works user in the Banner database and associate it to a Banner VPDI code. In the Banner database, an individual MEP environment is identified by the VPDI code, which is stored on the VPDI_CODE column of each table. In Degree Works, you must create a unique user in the Banner database for each MEP entity. This is the same user that is stored in the `DB_LOGIN_BANNER` variable in `dwenv.config`.
    a. In Banner, access the GSAVPDI form and click the **User Assignment** tab. Here you need to associate each Oracle user you have created in Banner for Degree Works (DB_LOGIN_BANNER) with its Banner VPDI Institution Code. After your DB_LOGIN_BANNER users are established in the Banner database, they should be available in the User ID picklist in GSAVPDI.
    b. On your Degree Works server, for each of your campus entities, edit the `$ADMIN_HOME/dwenv.config` file and place the following environment variable definition after the DB_LOGIN_BANNER variable: `export BANNER_VPDI=MYINST`, where MYINST is the Banner VPDI Code that you have associated with your Degree Works Oracle user DB_LOGIN_BANNER. Do not place spaces between the equal sign (=) and the VPDI Code value. The Maximum size of the VPDI code is 6-bytes.

       Note: You must log out and back in to your host session after making this change.

       Warning! Make sure the correct VPDI Code is input as NO error checking is done. If a VPDI code is used that is not valid, the database queries will not return any records for tables that are linked to a particular VPDI code, and you will not get the correct data extracted into Degree Works.

Related concepts

- Multi-entity processing in Degree Works
- Degree Works data extract configuration and installation

Banner Workflow integration installation

Updated: March 25, 2022

You can enable Banner Workflow integration with Degree Works for the exception petition and plan approval sample models.

Ellucian provides guidance and suggestions on Banner Workflow configuration. It is assumed you are familiar with setting up Models, Business Events, Business Processes and Business Components in Workflow and you should refer to Banner Workflow documentation for more information.

After the sample Workflow models are installed and set up, you should inspect and customize the models before testing or using them. In particular, you should inspect the activities, notifications, and emails defined in each of these workflow models to customize Roles, Performers, and email addresses. The from address in all notifications and emails used in these models is the email set for the Workflow admin user. These samples also use Workflow Roles: Approver, Academic Advisor, and Academic Dean. You should be aware of the users and email addresses associated with these Roles, because they will be the ones notified to approve Degree Works requests if you do not modify the models. Most likely, you will want to customize the models to change which Roles or Users are responsible for performing the approval actions and receiving Workflow notifications.

Configure Banner Workflow integration in Degree Works

Updated: March 25, 2022

There are several configuration steps in Degree Works for Banner Workflow integration.

Procedure

1. Load Workflow-related packages into the Degree Works database.

   ```
   cd app/sql
   db
   @wf_parameters_pkg.sql
   ```

   This file contains package wf_parameters and procedure F_PetitionParams. This package can be customized to change the parameters that are sent to Banner Workflow model DW_PETITION. The parameters that are extracted from the Degree Works database by this procedure need to match up with the parameters that the Workflow model expects. So, if the context parameters of the model are customized, this procedure will need to be customized, too. For the plan approval workflow, this package is not used because the data parameters are handled in java.

   ```
   @wf_updates_pkg.sql
   ```

   This file contains a package named wf_updates and procedures P_UpdatePetition and P_UpdateSepPlanner. These procedures typically do not need to be customized.
2. Configure Banner Workflow integration for Degree Works using Controller.

Review the `core.workflow.*` Shepherd settings and define your Banner Workflow server configurations.

3. Enable petition approval with Workflow.

   Review the `core.workflow.petition.*` Shepherd settings to enable and configure petition approval with Banner Workflow.

4. Enable plan approval with Workflow.

   Review the `studentPlanner.planner.workflow.*` Shepherd settings to configure plan approval with Banner Workflow. To enable plan approval with Banner Workflow, set `studentPlanner.planner.planApprovalMethod=W`.

   See Shepherd settings for more information.

Configure Banner Workflow integration in Banner

Updated: March 25, 2022

There are several configuration steps in Banner for Banner Workflow integration.

Before you begin

You must be a system administrator who can reconfigure and restart the Workflow server.

Procedure

1. Upload these files from your Degree Works environment to your Workflow server.
   - `dw_petition.zip`
   - `dw_sep_planner.zip`

   They will be located in the `sis_Banner/export/Workflow` directory under the updates (installer root) directory. Locate the updates directory from your Degree Works installation (at `$HOME/updates`) or from your most recent Degree Works update (`$DGWHOME/updates`). Locate the files by issuing this command from within the updates directory:

   ```
   find . -name dw_petition.zip 2>/dev/null
   ```

   Transfer these Workflow .zip files as binary to your Workflow server.

2. Execute these commands to import the sample Degree Works workflow models. Substitute your Workflow username and password (wfroot for example) and the location of the files you uploaded in the previous step. After importing these files, the Workflow Models, and Business Components will be under the “+DegreeWorks” category.
   - `$WFHOME/bin/import username password dw_petition.zip`
   - `$WFHOME/bin/import username password dw_sep_plan.zip`

3. Edit `$WFHOME/config/configuration.xml` to add a DataSource for the Degree Works database to the DataSources section. An example DataSource setup follows in which you should substitute your actual Degree Works server url for `your.degreeworks.server.edu`, the actual port number the Degree Works Oracle listener is running on for 1521, and your actual oracle listener handler for

YOUR_DEGREEWORKS_SID. The username and password here must match what you see in the `$DB_LOGIN` variable; this is the Degree Works user login to the Degree Works schema.

```
<DataSources>
<!-- ... other existing data sources should not be modified... -→
<DataSource name="DegreeWorks">
<Url>jdbc:oracle:thin:@your.degreeworks.server.edu:1521:YOUR_DEGR
EEWORKS_SID↙ Url>
<Username>yourusername↙Username>
<Password>yourpassword↙Password>
↙DataSource>
↙DataSources>
```

a) After editing configuration.xml, run: `$WORKFLOW_HOME/bin/wftool uploadconfig`.

b) Use the `engineconsole` and `startengine` scripts to restart the engine node(s).

c) Restart the OC4J instance(s).

4. In the Workflow console, create the Product Type needed for connecting to the Degree Works database.

   a) As an administrator using Workflow in your web browser, navigate to **Administration** > **Workflow System Administration** > **Product Types**.

   b) Click **Add Product Type**.

   c) Give it the name `DegreeWorks`, Version 1 (one), and choose the Data Source **DegreeWorks** from the drop-down list. This depends on the naming of the DataSource and successful restart of the Workflow server.

5. Create or modify business events for the Degree Works models.

   a) Select **Administration** > **Business Events** > **Business Event Definitions**.

   b) Click **Add Business Event Definition**. For petition approval:

      a. Set **Name** and click **Save**. The default is DW_PETITION and this must match the value in the `core.workflow.petitionEventName` Shepherd setting.

      b. Add Event Parameters for DW_PETITION as follows and set them all to Type=Text and Guaranteed to Yes or No as shown below. For Guaranteed options, if you change any data that you want to be optional in `wf_parameters_pkg.sql`, set those parameters to Guaranteed=No.

Parameter	Type	Guaranteed Option
CREATE_DATE	Text	Yes
NOTE_TEXT	Text	Yes
STUDENT_EMAIL	Text	No
STUDENT_ID	Text	Yes

Parameter	Type	Guaranteed Option
STUDENT_NAME	Text	Yes
UNIQUE_ID	Text	Yes
USER_EMAIL	Text	No
USER_NAME	Text	Yes

c. There are two optional parameters that you can define and modify wf_parameters_pkg.sql to implement. They control whether either or both approvers in the sample model should be ignored. Define and set APPROVAL1_REQUIRED=0 to disable approver1. Define and set APPROVAL2_REQUIRED=0 to disable approver2.

d. Still on Business Event Definition, click **Add Workflow Association**. Select the value from the drop-down list provided for your workflow model and version.

e. After you click **Save**, you will then map all of the business event parameters to the context parameters within that version of the workflow model. You must map all "guaranteed" parameters. When finished click **Save Parameter Mappings**.

6. For plan approval, repeat step 5 for another event named DW_APPROVE_SEP_PLAN, which must match the value in the `studentPlanner.planner.workflow.plannerEventName` Shepherd setting and have the following parameters. Parameters named APPROVAL1_REQUIRED and APPROVAL2_REQUIRED are optional and work the same way.

Parameter	Type	Guaranteed Option
CREATE_DATE	Text	Yes
DESCRIPTION	Text	No
NOTE_TEXT	Text	Yes
STUDENT_EMAIL	Text	No
STUDENT_GOALS	Text	No
STUDENT_ID	Text	Yes
STUDENT_NAME	Text	Yes
UNIQUE_ID	Text	Yes
USER_EMAIL	Text	No
USER_NAME	Text	Yes

7. Create or modify business processes for the Degree Works models.

   a) Select **Administration** > **Enterprise Management**.

   b) Click **Add Business Process**.

c) Name the business processes. Ellucian recommends naming it using the same token as the Business Events: DW_PETITION and DW_APPROVE_SEP_PLAN.

d) Set Status=Active.

e) Click **Add Workflow Association** and select the appropriate workflow model or version.

f) Click **Add Event Association** and choose the Business Event name you defined.

g) Authorized Initiators can be left empty. It is optional.

h) Click **Save Process**.

8. Verify you have the Degree Works models installed by opening them in the Workflow Modeler: DW_PETITION and DW_APPROVE_SEP_PLAN.
9. If you want to customize the models immediately, use the modeler to make a new version or create a copy. Refer to the Banner Workflow documentation for more information on using the modeler.
   - When customizing the models, if you change any context parameters, you must update the `pl/sql` where those parameters are extracted from Degree Works in `app/sql/wf_parameters_pkg.sql` (and reload that package into the Degree Works database).
   - You also must modify the business event parameters you defined later in steps 5 and 6 to match the parameters names given in the model context parameters and in the pl/sql package.

Petition approval management tools

Updated: March 25, 2022

Degree Works provides a native tool to allow your users to approve or reject petitions. However, if you are using Banner Workflow to manage petition approval, it would be best to not use the Degree Works tool because using both mechanisms will most likely lead to confusion on campus about the approval process.

The Exception Management tab in the Responsive Dashboard supports a way to approve or reject petitions. This should be turned off if Banner Workflow is being used to manage petitions approval.

See the Exception Management topic for more information.

Integration with portals

Updated: September 30, 2022

End-user access to Degree Works by Banner customers is typically accommodated through either Banner Self-Service or Student Profile.

Self-Service Banner (Banner 8)

Single sign-on for Degree Works using Self-Service Banner (Banner 8)

Updated: September 30, 2022

You can configure Self-Service Banner and Degree Works for single sign-on integration.

You can install a package that allows passing parameters like student ID to Degree Works. However, it is also possible to simply add a plain link to the Responsive Dashboard app in SSB. Users who click on that will be taken to Degree Works, which will obey the configured SSO. Students will be shown their own audit, and other users will be shown what they have permission to see (their own advisees, ability to search for all students, etc.) Users with access to multiple students will be able to switch between students within Degree Works. If you would prefer to have advisors/staff choose the student in SSB first, and have that student's audit preload when Degree Works is accessed, continue with the instructions below for installation and setting of the pl/sql packages.

The purpose of these installation steps is to make a new menu item available in Self-Service under the tab where it is configured. When that menu item is selected the link will send a single sign-on request to Degree Works then cause the Degree Works application to be displayed if the single sign-on request is validated. If it is not validated then an error will be displayed. The recommended means of accomplishing single sign-on is using Ellucian Identity Services (EIS), which supports SAML and CAS.

Set up menu for student role

Updated: September 30, 2022

You can set up a PL/SQL package to be called when a user selects a menu item.

About this task

The package generates HTML markup, which defines how the page looks and works in the browser. The packages provided with Degree Works are meant to be used as an example, so it would be best to decide on how these links are desired to function and appear before proceeding. The persons who implement this should be familiar with PL/SQL, HTML, Javascript in the browser, and have access to WebTailor Administration. Code modifications will be necessary depending on if you are using native SSB single sign-on, or an external single sign-on provider like CAS.

When modifying these sample SQL files, first copy to a different location and maintain your changes separate from the Degree Works installation package because these files could be updated in future release. Consider using a source control system to maintain your code.

No extra grants are needed for these packages. You already ran the bannergrants.sql script to run the necessary grants.

Procedure

1. Locate the dwssbstudent.sql file in the app/sql directory on the DW host server (cd app/sql). If your institution uses something other than WWW_USER for Banner Self-Service connections,

edit the grant statement in dwssbstudent.sql and dwssbfaculty accordingly. Use the shortcut script dbb to connect to your Banner database in sqlplus, and then run the file by issuing @dwssbstudent.sql. This will create the package called DW_Student in the database.

2. Log in to SSB with a user who has access to WebTailor Administration. Select the WebTailor tab, then Web Menus and Procedures from the menu.
3. Click **Create** to add a new web menu or procedure. Enter data for the following required fields. Enter data for the other fields according to your preference.
   a. Page Name: DW_Student.P_SignOn.
   b. Description: Degree Works. Or, enter whatever text you would like to appear as the description.
   c. Module: Student Self-Service. Or, select the appropriate module for your site.
   d. Enter page title, header text, and header graphic as you prefer.
   e. Enter back link settings as you prefer. If you are adding the Degree Works menu item to the student main menu then the back link url would be:

      `twbkwbis.P_GenMenu?name=bmenu.P_StuMainMnu`
   f. Choose Associated Roles according to your preference.
4. Steps 5 and 6 are options dependent on how you want to display or call the DW package from within Self-Service. Display the Degree Works link on a main menu by calling the new web procedure from your main menu package: DW_Student.P_SignOn (term, pidm, 0). The first two arguments, term and pidm, can be sent as null. The third argument, show_headers, should equal 0 (zero) when you are calling this procedure from another package. If you choose to do this step, you can skip step 6 below.
5. Display the Degree Works package link on a main menu. Do this if you skipped step 5 above. This option will allow you to display a full page containing the DW link and any additional information you want to provide.
   a. Go back to WebTailor > Web menus and Procedures.
   b. Select the menu where you want to add Degree Works (for example the main menu bmenu.P_GenMnu or the student main menu bmenu.P_StuMainMnu).
   c. Click **Customize Menu Items**.
   d. Click **Add a New Menu Item**. If you do not see that button, click **Copy Baseline to Local** first.
   e. Enter the URL: DW_Student.P_SignOn.
   f. Enter the link text, description, and sequence number according to your preference.
6. Configure WebTailor Parameters, as follows:
   a. DWLINKTEXT: The text you want to display as the link to Degree Works.
   b. DWURL: The URL to Degree Works.

      You should enter the URL + the context path of the dashboard. This would be the same composite URL that you would see in your browser window if you just type in the URL and access the dashboard that way.

      For instance, if your dashboard URL is https://your.server.com:8471, and you have your dashboard deployed at /Dashboard, your DWURL WebTailor parameter should be set to:

      https://your.server.com:8471/Dashboard/

c. DWDISPLAYBUTTON: Set to 1 to have a button displayed instead of an automatic redirect.

Menu setup for faculty role

Updated: March 25, 2022

You can set up a menu item for a faculty role similar to the student role with some differences listed.

Procedure

1. The sql file app/sql/dwssbfaculty.sql contains another package DW_Faculty with a procedure called P_SignOn. The default LOGONUSERCLASS in this package is “ADV”. If you are providing this link to users who are ADVX or REG instead of ADV, then modify the hardcoded value for LOGONUSERCLASS in this file. Insert the package into the Banner database using dbb.
2. The “Page Name” in step 4.a. above will be: DW_Faculty.P_SignOn.
3. Choose a module and associated roles that appropriate for faculty members.
4. Add the menu item for faculty members to a menu that is appropriate for your site. One option is to add it to the bmenu.P_FacStuMnu menu.

Configure logout

Updated: March 24, 2023

You can redirect users to the SSB page when they log out.

Procedure

1. To redirect users to the SSB page when they click **Logout**, configure the core.security.logoutUrl setting in Controller for the **dashboard** specification.
2. Change the example URL to your Banner Self-Service URL.

Student ID pass-along to Degree Works: Self-Service Banner (Banner 8)

Updated: September 30, 2022

An advisor working in SSB must select a student before clicking on Degree Works. When the user does choose Degree Works the student being reviewed is passed to the student context area within Degree Works.

As soon as the student appears the Worksheet tab is automatically selected and the student's most recent degree audit is displayed.

Allowing an advisor user coming from SSB to switch to another student in Degree Works would cause much confusion when the user then switched back to SSB because the new student ID is not then pushed back to SSB - they would be surprised to see the old student ID still sitting there in SSB.

To prevent such confusion it is best to take away the ability for these users to switch to another student within Degree Works. To remove this ability you should remove the SDSTUANY and SDFIND keys from the ADV or ADVX groups in Controller or both.

In the Responsive Dashboard, the student ID needs to be passed in the URL when connecting to Degree Works using this format:

https://<your server name>/<path to Degree Works>?studentId=<someid>

Only users who have been given permission to access this student's records will be successful in seeing this student loaded when they connect to Degree Works. If the user is an advisor with a fixed set of advisees and the advisees have been loaded, this specific student will appear as the selected student. If the user does not have the SDFIND or SDSTUANY keys they will not be able to switch to a new student; switching students must occur in SSB.

User role pass-along to Degree Works: Self-Service Banner (Banner 8)

Updated: September 30, 2022

An advisor or staff member that is also a student may want to access Degree Works as their student persona.

The as-student parameter should be used to log the user on with a STU user class. Note that the access granted to the user will be that which is defined in core.security.rules.shpcfg where DGWUSERCLASS = STU. However, if the user has also been assigned Shepherd keys or groups on their shp_user_mst through Controller, they will also be granted access to that functionality.

In Responsive Dashboard, the as-student parameter needs to be passed in the URL when connecting to Degree Works using this format:

https://<responsive-dashboard-url>/<context path>?as-student=true

Banner Student Profile (Banner 9)

Student ID pass-along to Degree Works: Banner Student Profile (Banner 9)

Updated: September 30, 2022

The student ID needs to be passed in the URL when connecting to Degree Works using a specific format.

https://<your server name>/<path to Degree Works>?studentId=<someid>

Only users who have been given permission to access this student's records will be successful in seeing this student loaded when they connect to Degree Works. If the user is an advisor with a fixed set of advisees and the advisees have been loaded, this specific student will appear as the selected student. If the user does not have the SDFIND or SDSTUANY keys they will not be able to switch to a new student; switching students must occur in SSB.

Link to the Worksheets tab in Responsive Dashboard:

https://{server}/{deployment-path}?studentId=<someid>

Link to the Exceptions tab in Responsive Dashboard:

https://{server}/{deployment-path}/exceptions?studentId=<someid>

Link to the Plans tab in Responsive Dashboard:

https://{server}/{deployment-path}/plans?studentId=<someid>

User role pass-along to Degree Works Banner Student Profile (Banner 9)

Updated: September 30, 2022

Advisors or staff members who are also students may want to access Degree Works as their student persona.

The as-student parameter should be used to log the user on with a STU user class. Note that the access granted to the user will be that which is defined in core.security.rules.shpcfg where DGWUSERCLASS = STU. However, if the user has also been assigned Shepherd keys or groups on their shp_user_mst through Controller, they will also be granted access to that functionality.

In Responsive Dashboard, the as-student parameter needs to be passed in the URL when connecting to Degree Works using this format:

https://<responsive-dashboard-url>/<context path>?as-student=true

RabbitMQ

Updated: March 25, 2022

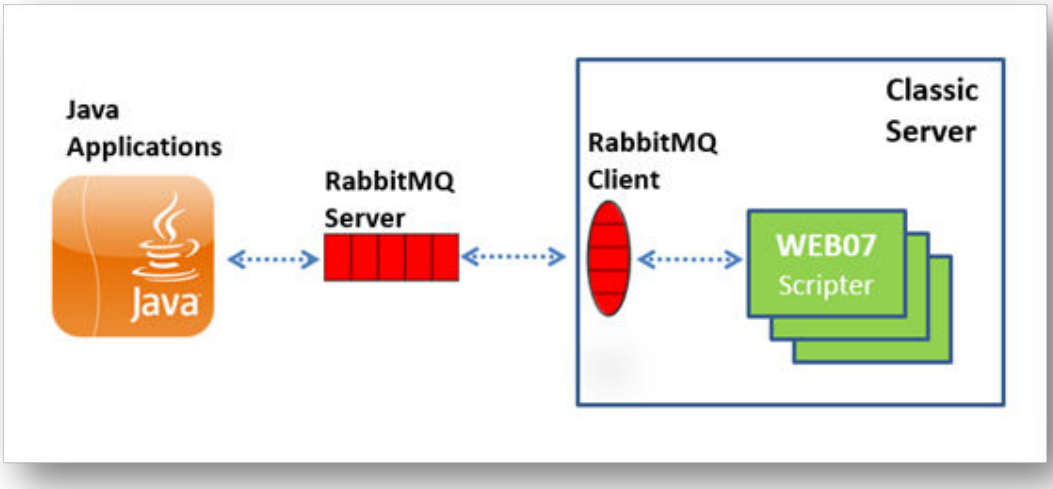
Degree Works uses RabbitMQ, a third-party open source message brokering software that implements the Advanced Message Queuing Protocol (AMQP), to communicate between processes running on the same or on different servers.

Two RabbitMQ software components must be installed for this communication to function:

RabbitMQ Server: installed on any server. For example, you can install it on the classic server or on one of your application servers or on a new server. It must be running at all times. For information on installing and using a RabbitMQ Server, please visit https://www.rabbitmq.com.

RabbitMQ Client: installed on the classic server; the Degree Works software makes calls into the RabbitMQ C library to communicate with the server.

The RabbitMQ software allows applications to communicate as shown in this diagram. The Java applications communicate with each other through the RabbitMQ server. They communicate with the classic daemons using both the RabbitMQ server and the RabbitMQ client to read from and write to the server.



By default, traffic between the RabbitMQ broker and our applications are not encrypted. If the traffic will travel on an unsecured network, you should consider encrypting it. See the Enable SSL/TLS for Degree Works and RabbitMQ section.

The following settings are required for configuring RabbitMQ in an environment.

- classicConnector.amqp.channelCacheSize
- classicConnector.amqp.timeout
- core.amqp.broadcast.heartbeatSeconds
- core.amqp.broker.host
- core.amqp.broker.port
- core.amqp.password
- core.amqp.request.heartbeatSeconds
- core.amqp.request.timeoutSeconds
- core.amqp.useSsl
- core.amqp.username
- core.amqp.virtualHost

The following settings are required for configuring RabbitMQ in an environment and must be unique for each environment.

- classicConnector.amqp.exchange
- core.amqp.exchange.shpSettings
- core.amqp.exchange.transit
- core.amqp.exchange.ucx

Add a user to allow access to RabbitMQ

Updated: September 30, 2022

You must add the Degree Works RabbitMQ user with the rabbitmqctl script. This is the user and password defined in the core.amqp.username and core.amqp.password Shepherd settings, respectively.

About this task

From a command prompt on the server on which your RabbitMQ server is installed, you can run the commands listed in this task on all platforms, including Windows.

For other useful commands, see the Manual Pages (man pages) at https://www.rabbitmq.com.

Procedure

1. Enter the command for the user and password. This user is only to allow access to RabbitMQ and is independent of all other Oracle or Unix users you need for Degree Works.
   ```
   # rabbitmqctl add_user <someuser> <somepwd>
   ```

2. Set permissions for your user.
   ```
   # rabbitmqctl set_permissions -p / <someuser> ".*" ".*" ".*"
   ```
3. Optional: Verify that your user and password are set up and working.
   ```
   # rabbitmqctl authenticate_user <someuser> <somepwd>
   ```

Enable SSL/TLS for Degree Works and RabbitMQ

Updated: March 24, 2023

The default installation of RabbitMQ has SSL disabled and only TCP connections are used. However, you can configure Degree Works to communicate with RabbitMQ over SSL.

About this task

The RabbitMQ server must be configured to support SSL/TLS. Information about setting up RabbitMQ with SSL/TLS can be found on the RabbitMQ website.

After the RabbitMQ server has been configured for SSL, enable SSL in RabbitMQ for Degree Works.

Procedure

1. Change the core.amqp.broker.port Shepherd setting to the SSL port specified in rabbitmq.config – the default SSL port is 5671.
2. Change the core.amqp.useSsl Shepherd setting to **true**.
3. Make sure any firewalls between the applications and the broker are configured for the SSL port.
4. Optional: Enable peer verification for the RabbitMQ server. This is typically not needed, but if it is enabled, you must copy the CA certificate that is on your RabbitMQ server to your classic server (if on a different server), and make the following changes in dwenv.config:
   a. Set AMQP_VALIDATE_PEER_CERTIFICATE to 1.
   b. Set AMQP_SERVER_CA_CERTIFICATE_PATH with the path to the CA certificate. Make sure the two lines for this variable are not commented out.
   c. Log out of and in to your classic session to set the variables.
5. Restart all daemons on the classic server and all applications on the Java application server.

Install the RabbitMQ Client for Degree Works

Updated: March 25, 2022

The RabbitMQ Client is a required component on the Degree Works classic server.

Before you begin

Before installing the RabbitMQ Client, you must first install CMake, if it is not already on your server. To check for CMake on your server, use the `# which cmake` command:

If you do not find it, then install it following the instructions in Install CMake on the classic server. If CMake is already installed, then proceed to Install the RabbitMQ Client on the classic server.

Install CMake on the classic server

Updated: March 25, 2022

If CMake is not already on your server, you must install it.

Before you begin

For information about where to download CMake, see https://github.com/alanxz/rabbitmq-c.

Procedure

1. Download the appropriate tarball file for your server.
2. As the root user, create a cmake directory under /etc and cd into it.
   ```
   # cd /etc
   # mkdir cmake && cd cmake
   ```
3. Transfer the cmake tarball file you downloaded above to the /etc/cmake directory and run gunzip and then untar it. For example,
   ```
   # gunzip cmake-3.7.2-Linux-x86_64.tar.gz
   # tar -xvf cmake-3.7.2-Linux-x86_64.tar
   # rm cmake-3.7.2-Linux-x86_64.tar
   ```
4. Create a link to the cmake binary in /usr/bin to where CMake was installed. For example, `# ln -s /etc/cmake/cmake-3.7.2/bin/cmake /usr/bin/cmake`

Install the RabbitMQ Client on the classic server

Updated: September 29, 2023

After you install or verify installation of CMake on your server, you can install the RabbitMQ Client.

Before you begin

For information about downloading the RabbitMQ Client, see https://github.com/alanxz/rabbitmq-c. Do not download the master branch version. Be sure to select the version of RabbitMQ Client that your release of Degree Works supports. See Degree Works Installation.

Procedure

1. As the root user on your classic server, transfer the `rabbitmq-c` zip file into the `/opt` directory.
   When the file is unzipped, you will have a `/opt/rabbitmq-c-x.x.x` directory, where x.x.x is the version of the RabbitMQ Client. This is where the Degree Works $LIBS and $INCLUDES environment variables will point.
2. Unzip the zip file in the `/opt` directory. For example,
   ```
   cd /opt
   unzip rabbitmq-c-0.13.0.zip
   rm rabbitmq-c-0.13.0.zip
   ```
3. In the `rabbitmq-c-x.x.x` directory, create a build directory and navigate to it.
   ```
   cd rabbitmq-c-0.13.0
   mkdir build && cd build
   ```
4. Run `cmake ..` pointing to the parent directory. This will take several minutes.
5. Run `cmake --build .`, and do not forget the space and dot at the end. This will take several minutes to complete.
6. For versions of the client higher than 0.11.0 only, run the command `ls` and look for a folder called `include`. If it is present, move the contents of its subdirectory to the same location in the `include` folder of the current folder's parent directory with the following command:
   ```
   cp ./include/rabbitmq-c/* ../include/rabbitmq-c/
   ```
7. Create or modify a link from `/usr/include` to the `rabbitmq include` directory.
   For versions of the client up to and including 0.11.0, the include directory is called `librabbitmq`.
   ```
   ln -s /opt/rabbitmq-c-0.11.0/librabbitmq /usr/local/include/rabbitmq
   ```
   For versions of the client higher than 0.11.0, the include directory is called `include`.
   ```
   ln -s /opt/rabbitmq-c-0.13.0/include /usr/local/include/rabbitmq
   ```
8. Create a symbolic link to the `librabbitmq.so*` library file from `/usr/local/lib`. For example,
   ```
   ln -s $PWD/librabbitmq/librabbitmq.so.4 /usr/local/lib/librabbitmq.so.4
   ```
9. Create a symbolic link for the `librabbitmq.a` file. For example,
   ```
   ln -s $PWD/librabbitmq/librabbitmq.a /usr/local/lib/librabbitmq.a
   ```