Jenzabar CX

Registration

JENZABAR

Implementation Guide

Filename: imregist
Distribution date: 11/15/2001

Contact us at www.jenzabar.com

JENZABAR, INC.
REGISTRATION IMPLEMENTATION GUIDE

TABLE OF CONTENTS

Cross-Functional Issues on the Implementation of the Registrar Module

Introduction

As you implement each application within the Registrar module, various policy issues will arise about which you must make decisions. However, in addition to issues affecting strictly the Registrar's office, there are other issues that involve various offices at an institution. The next few pages list some of these issues, as well as helpful information you can use in deciding how to resolve each issue.

List of Cross-Functional Issues

The following lists each cross-functional issue, as well as a description of each, that should be addressed while implementing the Student Records application.

Use of course, section, and reference numbers

To ensure that the original intent of an institution's numbering system remains intact, answer the following questions:

- Does the meaning of the course number indicate academic year (e.g., freshman)? Does it include the credit value or some other value?
- What are the institution's policy and procedure on reusing course numbers? How often are numbers reused? Why?
- Are section numbers or letters used? If so, is a meaning associated with them? Do they appear on transcripts?
- Are course or section numbers assigned to academic units in a centralized (e.g., Dean's list or Registrar's office, Curriculum Committee) or decentralized manner (e.g., each department assigns its own numbers)?
- Does the current system use a reference number for each course section? How is the reference number generated? How is it used?

Developing the annual processing calendar for catalog and schedule information

The official Academic Calendar table (acad_cal_rec) must reflect the catalog and schedule dates. Therefore, all offices at an institution must understand and participate in developing the calendar used for catalog and schedule information.

Access to student records, catalog and schedule, registration, grading, and academic history information by other offices at an institution

To ensure that individuals outside a particular office are able to access student records, catalog and schedule, registration, grading, and academic history information efficiently and cost-effectively, while guarding against misuse of this ability to access information, answer the following questions:

STUDENT RECORDS

Which office is responsible for creating student ID cards? Does the card include information from admitted students' files? How is the information produced?

CATALOG AND SCHEDULE

- Is an individual other than the registrar responsible for publishing the printed course catalog? How is the master course inventory maintained?
- What information does the bookstore need for ordering books? How does the bookstore get the information it needs?
- Which offices need facility schedule information (e.g., Information Office, Campus Security, Physical Plant)?

REGISTRATION

- Which offices need student schedule information (e.g., Financial Aid office, Dean of Students, Information office, Campus Security, Emergency contact office)?
- Is an office other than the Registrar's office responsible for reports such as enrollment and IPEDS (e.g., Institutional Research office)?
- Is an office other than the Registrar's office responsible for or involved in facility assignment or reporting (e.g., Academic Dean, individual academic departments, and Physical Plant office)?

GRADING

- Do offices other than the Registrar's office maintain copies of grade lists turned in by faculty?
- Which offices receive a copy of student grade reports (e.g., Academic Dean, individual academic departments, Dean of Students, Placement office)? How are the reports organized?
- Which office is responsible for grade reports (e.g., Dean's, Institutional Research office)?
- Which offices are responsible for determining the dean's list? How do these offices get the information they need?
- Which offices determine which individuals are on probation or suspension? How do these offices get the information they need?
- How do the Financial Aid office and the office responsible for merit scholarships get the information by which they assess satisfactory academic progress?
- Where and how is progress determined for veterans and social security students?
- How are requests for term grade information by third-party payees handled?

ACADEMIC HISTORY

- Which offices expect to receive an official or unofficial transcript on a regular basis? On a student-by-student basis? By some other grouping of students? How are such records used?
- Which offices expect to access online academic history information? For what purpose?
- Are transcripts provided to another campus office ever sent off campus? For what purpose?

Who determines whether a degree is to be conferred with honors? What is the basis to determine this information?

Course and section fees

The Registrar's and Bursar's offices usually coordinate course and section fees, but sometimes fees or charges are handled manually by individual academic departments. CARS Solution supports virtually any kind of fee or charge associated with a course or section; Jenzabar, Inc. recommends an institution-wide review of such charges.

All course or section requisites that govern enrollment

To implement the CARS Solution automated requisite checking feature, complete these tasks:

- Build a Requisite record (req_rec) for each course for which requisites exist
- Review the relationships among multiple requisites as they appear in the course catalog (e.g., review "and" and "or" logic)
- Identify non-course requisites (e.g., test scores and grades) that are tracked by CARS Solution

All faculty or administration approvals required for students to enroll and sections

Students occasionally must obtain approval from a faculty member, department, or dean in order to register for a course or section, whether the requirement is course- or student-based. CARS Solution can help apply these requirements consistently. However, every office must completely understand the requirements so that the appropriate records can be created.

Decisions made during Student Records and Catalog and Schedule implementations regarding student, course, and section records used in registration

Review the decisions made and information entered in the CARS Solution Admissions module and the Student Records and Catalog and Schedule applications of the Registrar module regarding the following registration requisites to determine their effects elsewhere during registration:

- The academic status of registering students (e.g., admitted, continuing, or returning)
- The appropriate catalog and term for courses and sections

Responsibility (e.g., office or individual) for each of the registration and add/drop functions occurring in CARS Solution

The Registrar's office usually is responsible for registration and add/drop functions; however, some related registration processes often involve other offices. To improve the efficiency of the entire registration process and of the services provided to students, the offices involved in registration-related functions must do the following:

- Identify each registration-related function
- Identify who is responsible for each function
- Decide whether CARS Solution needs to be modified to accomplish each function

Chronological and physical dimensions of the registration process occurring with CARS Solution

Consider *when* and *where* registration is to occur. For example, if registration now occurs in an arena setting immediately before classes begin, the institution might consider one of these three alternatives:

- Spread the process over time, resulting in fewer disruptions to institutional operations and lower personnel costs
- Decentralized data entry to individual academic departments, in order to link academic advising and course enrollment more closely
- Implement the CARS Solution Interactive Voice Response (IVR) telephone registration, enabling students to register at a time and location of their own convenience

Staffing of the CARS Solution registration process

Staffing considerations are significant when registration is centralized and occurs over a short period of time, whether the institution uses regular staff, temporary staff, or students. During implementation, assess efficient and cost-effective approaches to staffing during registration.

Holds management

CARS Solution enables an institution to replace manual, staff-intensive procedures for tracking academic and administrative holds with automated holds capability. During implementation, the departments at the institution should identify the following information:

- All possible holds
- Who is authorized to place or remove each type of hold
- The intended effect of each type of hold

Using CARS Solution, the institution either can decentralize or centralize hold functions (e.g., placing a hold, clearing a hold). In addition, the institution can label holds either as notation or absolute.

Identification, name, and address management

CARS Solution, because it is fully integrated, is able to provide the following benefits to an institution regarding identification, name, and address management:

- Each individual or entity associated with the institution is assigned a single identification number.

- The social security number, an optional feature in CARS Solution, calls up an individual's or entity's record just as the ID number does.
- CARS Solution can track prior names, salutation names, and names that reflect relationships between individuals or entities, and affords virtually unlimited capability for different addresses.

In resolving this cross-functional issue, the offices at an institution must ensure that the following tasks are completed:

- Appropriate addresses are available when needed (e.g., billing, grade reporting, third-party)
- Offices are assigned responsibility for ensuring that identification, name, and address information is available when needed, and that all offices are working cooperatively

Formatting standards (e.g., address conventions)

For consistent, professional-looking correspondence, reports, and lists, the offices at an institution must agree on the following considerations:

- How names and addresses are to be formatted
- What the valid values are for various data elements in the system (e.g., the names and abbreviations for degrees conferred, names of academic departments, majors)

Use of records

Scroll records, or collections of related information that are used as supplemental data about an individual or an entity associated with an institution, use space in CARS Solution only if data exists in a record. Some of the scroll records available in the Registrar module require coordination by several offices so that the information is usable by all offices that have access to the information contained in them. The following scroll records are examples:

- Relationship scroll records identify the ways in which individuals or entities associated with an institution are related (e.g., father/son, sister/brother, husband/wife, employer/employee). Because different offices might view these relationships differently, and might use the information in unique ways, it is important to meet the needs of all offices.
- The Accomplishment, Education, and Examination scroll records are critical to the Academic History application of the Registrar module because the information contained in these records appears on academic records and transcripts, provided certain key data are entered. The institution must decide whether these records are reserved exclusively for the registrar and, if not reserved for the registrar, what the rules are for use by other offices.

Admissions and academic status coding

A single status table exists in CARS Solution to identify and track prospective students' admissions statuses and enrolled students' academic statuses. However, there must be close coordination among the Admissions office, the Registrar's office, and the individual academic offices in order to identify the possible admission and academic statuses.

Determining completion of degree requirements; certifying degree completion

The offices involved in degree requirements or degree completion should organize the methods for determining when a student has met the requirements for a degree, and for certifying that a degree has been earned and/or conferred; they must understand how these processes work, because CARS Solution can support several ways to organize these processes.

Note: The transcript forms (both unofficial and official) used in the Academic History application are very flexible regarding the information about degrees earned.

Transferring information from the Registrar's office to the Alumni Affairs office when a degree has been conferred (or at other times as appropriate)

CARS Solution electronically transfers information from the Registrar module to the Alumni and Development module for individuals who have earned degrees. However, the Registrar's and Alumni and Development offices must agree on the following related issues:

- The definition of an alumnus
- The timing of the information transfer using CARS Solution
- The content of the information transfer using CARS Solution

Creating the Program Enrollment record from admissions information; accessing the Program Enrollment record for readmitted students

The Program Enrollment record is created when the Admissions office releases admitted students' records to the Registrar's office for registration; however, the Program Enrollment record must be created before a student can enroll in classes.

The Admissions and Registrar's offices must resolve the following issues:

- When to create the records for both offices (because, once created, offices other than the Registrar's office are discouraged from accessing these records)
- Accessing the Program Enrollment records of former students who are being readmitted
- Updating the Program Enrollment record when a new Admission record is created, and the Program Enrollment record already exists

Block Registration

Introduction

Block Registration provides the capability to register students into several common courses (called blocks) without having to enter the courses one-by-one. It does require accessing the students one-by-one in online registration to determine their registration status and to specify the appropriate block of courses.

Setup

The important components of Block Registration are as follows:

- The macro ENABLE_FEAT_BLK_REG, located in the directory path $CARSPATH/macros/custom/student, must be defined (set to "Y" for Yes). After defining this macro, you must reinstall the following:
  - $CARSPATH/menusrc/student/regist/menudesc
  - $CARSPATH/menusrc/student/forms/menudesc
- This process is necessary for the Block Registration options to appear on the Registrar menus. Accordingly, if the institution does not plan to use Block Registration, set the macro to "N" (for No). After the appropriate reinstalls, the menu options for Block Registration will not appear on the Registrar menus.
- Prepare the block tables by selecting Block Registration from the Registrar Main Menu. Next, select the Block Registration Table menu option from the Block Registration menu. This is the option that allows entry of specific courses/ sections into individual blocks

It is relatively common for certain columns of data on the Program Enrollment record to be used to identify the students. Two of the most typical columns used are subprogram and/or student classification.

Note: Before beginning to build these blocks, certain data must be obtained from the Registrar. Typically this includes, but is not limited to, identifiable students who will use this process and the courses that are common for each group of students.

Block Table Explanation

The following list provides a column-by-column explanation of the Block table.

Block Code
This is a user-defined code for a common group of courses (e.g.,T1A). These block entries are relatively short-lived and are required only for registration purposes for a given session/year.

Note: Unless there are justifiable reasons not to do so, these codes can be used again in subsequent sessions.

Block ID
Enter the student's ID only if the block applies to a specific student. Normally a block applies to many students, and the column is set to a value of zero (0).

Block Program
Enter the program for this block of courses (e.g., UNDG).

Block Subprogram
Enter the subprogram if this course applies to a certain group of students. Otherwise, leave blank.

Block Session

Enter the session (e.g., FA) for this block of courses.

Note: The session must correspond to the session in the Section records for these courses.

Block Year
Enter the year (e.g., 1993) for this block of courses.

Note: The year must correspond to the year in the Section records for these courses.

Block Catalog
Enter the catalog (e.g., UG93) for this block of courses.

Note: The catalog must correspond to the catalog in the Section records for these courses.

Block Course
Enter the course number for one of the courses in this block.

Note: The course must have a valid course/section for this session/year and catalog as specified in this block (e.g., HIS303).

Block Section
Enter the section number for this course as contained in the Section record (e.g., A, 01, etc.).

Block Classification
Enter the student classification associated with this block (e.g., FR, SO, etc.). This is used only for reporting purposes.

Block Reference
The reference number for this course/section from the Section record appears, if available.

Note: Add a Block table entry for every course in a given block. For example, if there are six courses in which a given group of students is to register, add six Block table entries with the same block code.

Additional Options

The following lists and explains the five additional options that are available.

Create Block table
This option automatically creates the block tables explained above if all the courses that make up a given block have unique section numbers (e.g., AZ) for a given catalog (e.g., UG93) for a particular session/year.

Block Registration Report
After adding all the Block tables through the first two options (a and b), this report allows the user to determine the accuracy of input data.

Print Block Schedule
This option allows you to print a block registration schedule for a given session/year.

Block Section Schedule
This option provides a schedule of the block courses in chronological order.

Block Registration Forms
This option provides a registration form for students that lists the required courses of their blocks, space to list elective courses, and instructions.

Using Block Registration

In CARS Solution Release I, Block Registration is an option available in the online registration process. There is no longer a distinction between block and online registration as there was in previous releases.

Block Registration works similar to the online registration process. The primary difference is that Block Registration must be accessed through the Ctrl-n Registration Option window. After a block is specified, all normal registration edits (requisites, class capacity, permission of instructor, etc.) are accomplished with corresponding messages for each course, as appropriate. Ideally, if all edits are passed on all courses, they would be added without any messages.

Block Registration Process

The following list gives the Block Registration process.

Phase 1

Enter online registration from the menu and a Registration Initialization screen appears. Complete this screen routinely, except a default block code should be entered if you plan to register several students in the same block of courses. Exit this screen by pressing F1.

Phase 2

Query on the student desired by entering the ID, social security number, or use Ctrl-t for name query. After locating the student, registration is ready for the first course to be added. You must be in Add (register) mode to access Block Registration.

Phase 3

While in Add mode, press Ctrl-n (register). A series of options appears in a window. From these options, select Block Registration.

Phase 4

- If a default block was entered on the Registration Initialization screen, this Block Add code will appear in a window along with all the courses that were entered into the block. This default block code can be overridden with a Ctrl-c. This allows entry of a different block code.
- If a default code had not been entered on the Registration Initialization screen, selecting the Block Registration option from the window also will allow you to enter a block code.

In either of these last two steps, the cursor moves to Block Add and the desired block code may be entered (use Ctrl-t as necessary to determine the block code). If a valid block code is entered, its courses will appear in the window.

Phase 5

After the courses appear for the desired block, press F1 (Finish) to actually add the courses. Use Ctrl-c to cancel the process if it is decided that the block add is not appropriate.

Phase 6

The courses are edited as if they were entered one at a time in routine online registration, with error messages as necessary appearing a window. Courses are added to the Registration screen as edits are satisfied.

Phase 7

After entering the block courses, additional courses may be added or other changes made as desired.

Phase 8

A final F1 (Finish) actually transmits the transactions to the database.

Maintaining Historical Information on Session Records

Introduction

Institutionally, it is important that a limited amount of Program Enrollment record information be retained for historical purposes. The Program Enrollment record is a single record per student per program. It is ever changing, and if it was important to know information from previous years, it may be impossible to do so. To meet the historical data requirements, a process has been provided to update the student's Student Academic record as part of the registration process. Some basic columns (fields) of data such as the student's classification, academic status, resident state, and subprogram have been defined in the base product. Accordingly, registration will update these four columns with data from the Program Enrollment record when a student registers. This is appropriate since this is when the student’s Academic record is created for each session/year in which they register.

The institution can define additional columns of data that they want to maintain historically from the Program Enrollment record. This process is discussed later in Setup in this section. It should be realized that the data retained is typically the information that was current at the beginning of the term. It is a common practice to keep this information current up to the date the institution determines to be their reporting or statistical date. Since the data initially updated by the registration process may change before the reporting date, it is important that any information changed on the Program Enrollment record be updated on the appropriate Academic record. A process has been provided that can be run from the menu, at the institution's discretion, to update the student’s Academic record and/or provide a report of the inconsistencies between the two records. This is discussed further in *Using the Processes* in this guide.

Although the majority of the changes in historical data are accomplished either by registration or the menu update process, there are situations where the student's classification may be changed by the Transcript Update process. However, Transcript Update does not update the Academic record classification after the beginning date of a current session. A current session is defined as one whose beginning date is past, but the ending is a future date as specified in the Academic Calendar record for the term.

Of all the possible historical data that could be retained, the student's classification is the most important to the system, since it has student billing implications. Jenzabar, Inc. recommends that no changes be made to the standard product on this data field.

CAUTION: There never should be a case where historical information is updated for past terms. If you select a past session and year to update, all of the historical information for the past term will be replaced with current information. In other words, the integrity of the past term's historical data has been destroyed.

The CARS Solution registration process does not permit this type of data corruption. However, external processes not related to CARS Solution could influence this data. Exercise extreme caution when modifying academic session records.

Setup

After determining the data that should be retained institutionally, review and update the macros (includes) in $CARSPATH/include/applic/regent. Follow the instructions precisely if changes are to be made. The two includes that need to be reviewed are:

-
  ```
  #define STUAC_AUD_FIELDS { \
      {"cl", "cl"}, \
      {"subprog", "subprog"}, \
      {"res_st", "res_st"}, \
      {"acst", "acst"} \
  ```

The definition STUAC_AUD_FIELDS is used to link the fields in the Academic record (stu_acad_rec) to the Program Enrollment record (prog_enr_rec) for audit trail purposes. The fields listed will be copied from the prog_enr_rec to the stu_acad_rec when it is created through the Registration program. The first element is the field name from the prog_enr_rec; the second element is the field name from the stu_acad_enr_rec. The definition should appear in the following way:

```
{"prog_enr_field", "stu_acad_field"}, \
```

The braces ({}), Double quotes ("), and the backslash (\) are needed. Also note that the comma is outside the braces on all but the last line.

The CARS Solution standard product provides for the four fields specified. Any changes to this include, whether additions or deletions, should also be made to the menu process that updates the data after registration. The SQL script for this processing is in $CARSPATH/modules/regist/informers/updstuac.

Also, the menu process provides for a report that will identify the Program Enrollment record and Academic record inconsistencies. Changes to the include also should be made to the report in $CARSPATH/modules/regist/reports/lststuac.

- #define STUAC_AUD_LEN   4

The definition STUAC_AUD_LEN is the number of lines that are defined in STUAC_AUD_FIELDS. The number in the include must match the number of data fields in STUAC_AUD_FIELDS. For example, if one additional data field is added to STUAC_AUD_FIELDS, the number in STUAC_AUD_LEN would need to be changed from 4 to 5.

If changes to the includes are made, modify the SQL script accordingly. The SQL script is located in $CARSPATH/modules/regist/informers/updstuac. Also modify the report in $CARSPATH /modules/regist/reports/lststuac as necessary. Ensuring that the informer, the report, and the includes are consistent, cannot be overemphasized.

The institution should be aware that there is a script that actually runs the SQL script. The script, which normally requires no changes, is in $CARSPATH/modules/regist/scripts/updstuac.

Using the Processes

After the setup is accomplished, the processes are ready for use:

- As students register, their Academic records for the session are created. As part of the registration process, the data fields as specified in the include STUAC_AUD_FIELDS are updated to the Academic record with the current values contained in the Program Enrollment record. This is the only time the data will be updated automatically by the CARS Solution software.
- Since the data on the Program Enrollment record may change after the Academic record has been created, run the menu process to update the latter record, if so desired. The institution should determine the cut-off date for updating the historical information on the Term record. Typically, this cut-off date corresponds to the institution's statistical date. Jenzabar, Inc. recommends, at a minimum, that this update process be run on this cut-off date to ensure the historical data will correspond to the data used in the official reports for the term.

- The update process is the Academic Record Updates option on the Registrar: Session Processing Menu. It can be run for any of the following three options:
  - List provides a report of the records that have inconsistencies between the Program Enrollment record and the Academic record.
  - Update will update the Academic records with current Program Enrollment record information.

CAUTION: There should never be a case where this update is run for past terms. If you select a past session and year to update, all of the historical information for the past term will be replaced with current information. In other words, the integrity of the past term's historical data will be destroyed. Both will provide the report of inconsistencies and update the records without having to do them as separate processes.

- It may be desirable to test both the registration and update processes on a few students to ensure that the institution understands them and to validate that the proper data is being updated.
- Transcript Update may update the student's classification when it is run. For example, during the spring term, a student registers early for fall. At the time of registration, the student's classification was freshman, so this classification was put on the student’s Academic record by the registration process. However, as part of grade processing for spring, the student completed enough hours to be a sophomore. Therefore, the Transcript Update process that is run after grades are input will update both the Program Enrollment record and the fall Academic record to sophomore.

Student Billing Custom Record

Introduction

The Student Billing Custom record feature provides flexibility to the CARS Solution product. It allows assessments for non-course related fees interactively or by providing criteria for system-based value judgments on existing CARS Solution data following the registration process.

The Institutional Fee option in the registration process (*regent*) provides the ability to enter values for student billing evaluation upon completion of each student registration. The Student Billing Custom record (sbcust_rec) provides the base for the retention of the data, entered either interactively or in a background process.

Setup

The following list gives the setup process for the Student Billing Custom record.

Phase 1

Make an evaluation to define what non-registration related fees should be assessed at the time of registration. These fees would be separate from standard course assessments (e.g., parking permits).

Phase 2

Modify the Student Billing Custom record (sbcust_rec) to reflect the desired fields identified in phase 1. $CARSPATH/schema/financial defines specific code values in the schema, if appropriate.

Phase 3

Modify the include file $CARSPATH/include/applic
/regent in the following areas:

```
/* -----
   The array below defines those fields in the sbcust record that are to be set to
the specified value when a student     is completely withdrawn or a no show for a
session.  The     first element is the field name and the second element is the value
that the field is to have.
----- */

#define SBCUST_UPD_FIELDS { \
   {"park", " "},\
   {"asb", " "} \}
/* -----
   The definition SBCUST_UPD_LEN is the number of lines that are defined in
SBCUST_UPD_FIELDS.
----- */

#define SBCUST_UPD_LEN     2
```

Phase 4

Modify the Institutional Fees window in the registration process (*regent*) to reflect the new fields in the Student Billing Custom record (sbcust_rec). The path is:

$CARSPATH/modules/regist/progscr/regent/instfee

Note: The Institutional Fees window may be disabled in the *regent* menu opt. If so, the window will not appear until it is enabled.

Values to be used in the Institutional Fees window should be limited by a macro or a table to ensure Student Billing selects appropriate data.

Phase 5

Complete a reinstall of the registration area to install the changes for testing and implementation.

Additional Functions: Libusr

The Student Billing module in CARS Solution may be aided by a custom "C" code function, "Libusr," called by registration processing (*regent*) following completion of the interactive registration process. Each institution can modify the "C" code library or contract through Jenzabar, Inc. consulting to create/test the code.

The function is called from the path $CARSPATH/src/regist/regent/Libusr. It can be used to evaluate any structure of the database and define a value in the Student Billing Custom record (sbcust_rec) for Student Billing to test. The registration process (*regent*) calls this function each time the operator finishes the student add/drop process and automatically performs the "C" code function in the background.

Example: All students are charged a fee for accident and health insurance unless all classes taken meet in an off-campus location.

An example of a function is provided in the system under the pathname mentioned above. It is recommended that the institution carefully defines, develops, and tests these functions prior to live operations on CARS Solution.

Common Tables

Introduction

As part of the implementation process for the Registration application, you must review and modify (if necessary) common tables. The common tables are not exclusive to the Registration application; other applications also might use the common tables. Even though you might not need to modify any of the tables in the list that follows, you should review the files anyway, just to be sure.

CAUTION: Before you begin to modify any common tables, check with other departments at the institution to find out whether any other CARS Solution implementations have occurred, and which common tables were affected by the prior implementation(s). It is very important that you perform this checking so that you do not inadvertently negate the work performed by individuals in other areas of the institution.

Common Tables

The Form Order table (formord_table) is a common table that affects the Registration application. Review and update this table, if necessary.

There might be additional common tables that affect the Registration application. However, those other common tables will have been updated already by the Student Records and Catalog and Schedule application implementations.

Tables Specific to the Registration Application

Introduction

Registration-specific tables are tables that only the Registration application accesses. Any modifications you make to Registration-specific tables should not affect how other CARS Solution applications run.

Registration-Specific Tables

The following lists each Registration-specific table. Review and update (if necessary) these tables in the order listed below. Both the title and the filename of each table are provided below.

reason_table
Add/Drop Reason

grd_table
Grade

progreg_table
Program Registration

sdsform_table
SDS Forms

regperm_table
User/Group Permissions

Tables and Records

Table and Record Information

The following list identifies the tables and records that originate from the Accelerated Degree product. The list includes the filenames, location, purpose, and association of each table and record with programs, products, and other tables and records.

Note: The *Program interrelationships* in the list are included in the Accelerated Degree product. The *Product interrelationships* in the list are not included in the Accelerated Degree product.

Student Report Record
Contains static student enrollment data generated by the Student Reporting program primarily for IPEDS reporting.

UNIX filename: sturpt

Informix filename: sturpt_rec

Schema location: $CARSPATH/schema/student

Program interrelationships: sturpt

Product interrelationships: Registration

Table/record interrelationships: cw_rec, profile_rec, prog_enr_rec, stu_acad_rec

The following list identifies the tables and records that originate from the Registration product. The list includes the filenames, location, purpose, and association of each table and record with programs, products, and other tables and records.

Note: The *Program interrelationships* in the list are included in the Registration product. The *Product interrelationships* in the list are not included in the Registration product.

Registration Record
Identifies specific information about a registration transaction for a student.

UNIX filename: reg

Informix filename: reg_rec

Schema location: $CARSPATH/schema/student

Program interrelationships: regent, billing, regnoshow, reglist, grading, rollsess, trans, register_aps

Table/record interrelationships: id_rec

Note: The user_id column of this record contains the identification number from the id_rec of the user who inserted or changed this record. It is used with the register_aps.

Overview

Introduction

This section provides reference information about macros and includes used to set up the Accelerated Degree product.

Relationship Among Macros, Includes, and C Programs.

For all elements of the product other than C programs, m4 macros contain the definitions of features that have been designed to be modified for each institution. These macros, located under $CARSPATH/macros, are passed through the m4 processor.

CARS Solution contains an alternative method for the setting of features in C programs. Macros in the source code of C programs are not passed through the m4 processor because the C compiler has its own preprocessor, the cc.

To provide the same apparent functionality to C programs, a section in the include source directory has been allocated for a type of include file that acts as an intermediary between the m4 processor and the cc preprocessor. In operation, m4 macros are defined whose output is a valid cc macro. These m4 macros are placed in the include files. Then the include files are translated and the appropriate cc macro is placed in the include file. The installed include file is included by the C compiler at compile time so that the result of the compilation is influenced by the m4 macro.

General Installation Procedures

See the *CARS Solution Technical Manual - Volume I (Implementation and Maintenance),* for general procedures on setting and installing changes to macros and includes.

Macros

Introduction

CARS Solution contains macros that define specific values used throughout the Accelerated Degree product. The macros and includes enable you to change the available options and functionality of the Accelerated Degree product without having to modify C code. By modifying macros, you can customize your implementation of the Accelerated Degree product and make the product easier to maintain.

Definition and Function

A macro is an instruction that causes the execution of a pre-defined sequence of instructions in the same source language. A macro consists of uppercase letters and underscores, and is used in place of a text string within source files. CARS Solution expands the macro to the longer text during the installation process for a file. CARS Solution uses the following kinds of macros:

- Enable - Allows you to enable a feature of CARS Solution
- DBS_COMMON - Allows you to define database values in screens
- Periodic - Allows you to make changes on a periodic basis

Macros can perform one of the following functions:

- Define defaults on a screen (_DEF)
- Define valid values in a field (_VALID or _INCL)
- Enable system modules (ENABLE_MOD)
- Enable system features (ENABLE_FEAT)
- Establish a valid value for an include

How to Locate Macros

To locate Accelerated Degree macros, access the $CARSPATH/macros/custom/student file. To locate Common macros, access the $CARSPATH/macros/custom/common file.

Applocate Program

You also can locate macros using the *Applocate* program. *Applocate* checks the descriptions of macro files for the product you specify and lists each file that it locates in a separate file in your home directory.

Note: To locate the macros used in Accelerated Degree, using *Applocate*, you must specify the product name.

Follow these steps to run the *Applocate* program.

1. Select Utility Menu from the CARS Solution menu. The Utilities: Main menu appears.
2. Select File Options. The Utilities: File Options menu appears.
3. Select Locate Macro Values. The Locate Macro Values screen appears.
4. Select **Table Lookup** in the Macro Category field. A list of module names appears in a Table Lookup box.
5. Select a module name (e.g., STUDENT/SERVICES), and click **OK**. The Table Lookup box disappears.
6. Select **Finish**. The Output Parameters and Scheduling window appears.
7. Enter the following:
   - In the Time field, enter **NOW**.

- In the Background field, enter **Y**.

8. Select **Finish**.

The system creates the file, *applocate.out*, and sends it to your home directory.

Enable Macros

The following lists the Accelerated Degree enable macros located in the student macro file.

‘ENABLE_FEAT_ACCEL_DEG’, ‘Y’
Defines control of submenu displays and menus and also whether the degree group value is displayed on the grading head screen. The macro default setting, Y, specifies that the Accelerated Degree menu option appears in the Student Management: Registrar Main menu.

Parameters for the Registration List Program (*reglist*)

Introduction

CARS Solution contains parameters and compilation values for executing the Registration List program. You can specify parameters to compile Registration List in a specified manner at the time of execution.

Note: You also can specify compilation values with the includes for the Accelerated Degree product that affect the Registration List program.

The Registration List program provides class lists of all types to the institution. Various menu options may be created to generate the desired output from this program, based on available parameters.

Parameter Syntax

The *reglist* program relies on form screens, which act as templates for the printed copy of the class list. These screens are found in $CARSPATH/modules/regist/forms/reglist. They must be modified to meet the needs of the institution.

Note: The *reglist* program must be used to create NCS scan forms.

The menu options for *reglist* are found in $CARSPATH/menuopt/regist/programs.

You can display *reglist* parameters by entering the following: **reglist -,**

The following is the correct usage for running the Registration List program from the UNIX shell:

reglist [-y year] [-s sess] [-F form] [-L site] [-u subsess] [-C cat] [-D dept] [-N course]

[-S section] [-c class] [-l campus] [-W day_of_week] [-f] [-r] [-X] [-i] [-j] [-k] [-a] [-m] [-w]

[-B] [-I] [-R] [-M] [-O] [-G scantype] [-t special text] [-z] [-g fsbdate] [-h lsbdate]

[-d fsedate] [-e lsedate] [-A fac_id] [-Z deggrp] [-Q]

Parameters that appear in brackets are optional. Parameters that do not appear in brackets are required.

Parameters

The following lists the parameters for running *reglist*.

-y *year*
Optional - Specifies the instructional year.

-s *sess*
Optional - Specifies the instructional session.

-F *form*
Optional - Specifies the list form.

-L *site*
Optional - Specifies the site.

-u *subsess*
Optional - Specifies the subsession of the academic year.

-C *cat*
Optional - Specifies the catalog.

-D *dept*

Optional - Specifies the department.

Note: This parameter provides the ability to select a single department.

-N *course*
Optional - Specifies the course number.

Note: This parameter provides the ability to select a specific course number.

-S *section*
Optional - Specifies the section.

Note: This parameter allows for a single section of a course.

-c *class*
Optional - Specifies the class.

Note: This parameter provides the ability to select only students with a specific classification.

-l *campus*
Optional - Specifies the campus location.

Note: This parameter provides the ability to select sections with at least one meeting on the campus.

-W *day_of_week*
Optional - Specifies the day of the week.

Note: This parameter provides the ability to select sections with at least one meeting on the day.

-f
Optional – Specifies to sort lists by faculty name.

-r
Optional - Specifies to sort students by random number.

-X
Optional - Specifies to sort students by course number (for cross-listed class lists).

-i
Optional - Specifies to ignore cross-listed courses (for cross-listed class lists).

Note: This parameter disregards the “Y” for separate class lists.

-j
Optional - Specifies to include students who dropped a class on or after the first day of class.

Note: This parameter includes all dropped students after the beg_date on the sec_rec.

-k
Optional - Specifies to exclude withdrawn students.

Note: This parameter provides a true list of students who should be attending.

-a
Optional - Specifies to print forms for courses without students.

-m
Optional - Specifies to print only missing grades.

Note: Grading uses this parameter to locate missing grades by section.

-w
Optional - Specifies to print only students on wait lists.

-B
Optional - Specifies to print both confirmed and financially cleared students.

Note: This parameter is valid only if using registration and financial clearance.

-I
Optional - Specifies to print only students with financial clearance.

Note: This parameter is valid only if using financial clearance.

-R
Optional - Specifies to print only confirmed students.

Note: This parameter is valid only if using registration confirmation.

-M
Optional - Specifies to print only matriculated students for evaluation forms.

-O
Optional - Specifies to allow reprinting of class lists for scanning.

Note: This parameter is used in scanning to reprint lost scan forms.

-G *scantype*
Required only if using scanning - Specifies the type of scanning form to use.

-T *special text*
Optional - Specifies text to be printed at the bottom of the class list.

Note: You can specify up to 80 characters per line and up to 5 lines.

-z
Optional - Specifies to output blank student information lines.

Note: This parameter prints lines in place of blank spaces on forms.

-g *fsbdate*
Optional - Specifies the first date in the section beginning date range.

-h *lsbdate*
Optional - Specifies the last date in the section beginning date range.

-d *fsedate*
Optional - Specifies the first date in the section ending date range.

-e *lsedate*
Optional - Specifies the last date in the section ending date range.

Note: The last four date parameters (i.e., -g, -h, -d, -e) can be used to print class lists within a specified date range, and are typically used for alternate calendar sections.

-A *fac_id*
Optional - Specifies the faculty identification number for selecting sections.

-Z *deggrp*
Optional - Specifies the degree group for selecting sections.

-Q
Optional - Specifies to print only sections with a blank degree group.

Program Processing in the Student Reporting Program (***sturpt***)

How the Student Reporting Program Processes Information

The Student Reporting program always runs as batch process that you can schedule. These are the primary phases of the Student Reporting program:

1. The program determines whether rows already exist for the specified reporting session, reporting year, site, program, and reporting type.
2. The program selects the population to process.
3. The program sets column values based on the Profile record (profile_rec), Program Enrollment record (prog_enr_rec), and Academic record (stu_acad_rec).
4. The program sets columns containing the parameter values.
5. The program sets columns based on calculated values.
6. The program inserts rows.
7. The program inserts summary rows.

How the Program Determines Whether Rows Already Exist

The Student Reporting program performs a select against the sturpt_rec using values in the prog, site, rpt_sess, rpt_yr, rpt_type, and deggrp_flag. The deggrp_flag indicates whether it is processing traditional or nontraditional students. If one or more rows exist, and allow_rerun is set to N, it sends mail to the user telling the user that rows with the specified key exist, and the user cannot run the process.

If the allow_rerun parameter is Y, the program removes all rows with the given site, prog, rpt_sess, rpt_yr, rpt_type, and deggrp_flag values.

How the Program Selects Populations to Process

The Student Reporting program always processes accelerated degree course work and traditional course work separately. If the student has both types of course work for a defined period, the program uses separate records for each type of course work the student completed.

For traditional students, if the sel_by_date_only is N, the selected population has the following characteristics:

- Course work associated with a stu_acad_rec where the deggrp column is blank or has course work associated that occurs after a specified deggrp_cmpl_date (i.e., the sec_rec.beg_date > deggrp_cmpl_date).
- Course work that was registered before the given effective date and was not dropped until after the effective date or was never dropped.
- Course work that occurs in the specified reporting session and reporting year for the given program.

For traditional students, if the select_by_date_only is Y, the selected population has the following characteristics:

- Course work associated with a stu_acad_rec where the deggrp is blank or has course work that occurs after a specified deggrp_cmpl_date (i.e., the sec_rec.beg_date > deggrp_cmpl_date).
- Course work that will not be voided or dropped.
- Course work associated with a section beginning within the range of the first_beg_date and last_beg_date as passed to the process.

For accelerated degree students taking accelerated degree courses, the selected population has the following characteristics:

- Course work associated with a stu_acad_rec where the stuacad deggrp is not blank and this course work occurs before the deggrp_cmpl_date in the prog_enr_rec, or the deggrp_cmpl_date is blank.
- Course work associated with a section that is a deggrp section (i.e., the deggrp is not blank).
- Course work that will not be voided or dropped.

For accelerated degree students taking traditional courses, the selected population has the following characteristics:

- Course work associated with a stu_acad_rec where the deggrp is not blank and this course work occurs before the deggrp_cmpl_date in the prog_enr_rec, or the deggrp_cmpl_date is blank.
- Course work associated with a section that is not a deggrp section (i.e., the deggrp is not blank).
- Course work that will not be voided or dropped.

How the Program Sets Column Values

The Student Reporting program sets column values in the sturpt_rec based on values from these tables:

- profile_rec
- prog_enr_rec
- stu_acad_rec
- major_table

For the **profile_rec**, the Student Reporting program copies over columns that are defined in the config_table entry for the STURPT_PROFILE_COPY value. When the program runs the process, it copies these columns from the profile_rec to the sturpt_rec. The standard value is “sex, ethnic_code, age, res_ctry, res_cty” in the config_table entry.

To copy additional columns from the profile_rec to the sturpt_rec, you must change the config_table.value and add the additional columns to the sturpt_rec with the same column names that the profile_rec uses.

For the **prog_enr_rec**, the Student Reporting program copies over columns that are defined in the config_table entry for the STURPT_PROGENR_COPY value. When the program runs the process, it copies these columns from the prog_enr_rec to the sturpt_rec. The standard value is “major1, major2” in the config_table entry.

Note: The Student Reporting program may copy values from the stu_acad_rec, and major1 and major2 may be in stuacad in the standard product.

To copy additional values from the prog_enr_rec to the sturpt_rec, you must change the config_table.value and add the additional columns to the sturpt_rec with the same column names that the prog_enr_rec uses.

For the **stu_acad_rec**, the Student Reporting program copies over columns that are defined in the config_table entry for the STURPT_STUAC_COPY value. When the program runs the process, it copies these columns from the stu_acad_rec to the sturpt_rec. The standard value is “cl, wd_date, deggrp” in the config_table entry.

The stu_acad_rec that the program selects to copy over is determined by the following criteria:

- For accelerated degree students taking accelerated degree courses, the stu_acad_rec the program selects is the one associated with the last course work for a deggrp section for this student and date range. The sec_rec.beg_date determines the last course work.

- For accelerated degree students taking traditional courses, the stu_acad_rec the program selects is the one associated with the last course work for a non-deggrp section for this student and date range. The sec_rec.beg_date determines the last course work.
- For traditional students, if the program selects by session (i.e., the sel_by_date_only is N), the stu_acad_rec the program selects is the one associated with the rpt_sess and rpt_yr.
- For traditional students, if the program selects by date (i.e., the sel_by_date_only is Y), the stu_acad_rec the program selects is the one associated with the last course the student took in the date range.

For the **major_table**, the Student Reporting program copies over the cip_no value from the major_table entry corresponding to the major1 value it copies into the sturpt_rec.

How the Program Sets Column Values Based on Passed Parameters

The Student Reporting program stores all parameters to the process except allow_rerun and group to process in the resulting rows, which are then inserted.

How the Program Sets Column Values Based on Calculated Values

The Student Reporting program must determine the values for each of the columns in the following list. Descriptions in the schema explain how to determine the values.

- reg_hrs
- reg_au_hrs
- reg_hrs_deg
- reg_hrs_wd
- reg_hrs_rem_nondeg
- reg_au_hrs_conf
- reg_zero_hr_crs
- hs_grad_date
- hs_id
- trnsfr_id
- deg_seek
- first_time
- first_time_trnsfr
- first_time_fresh
- excl_out_ctry
- excl_out_st
- excl_in_st

For many of the values, the program will perform a select against the cw_rec, and the various totals will accumulate as the program loops through the rows.

If the course work the program processes is related to accelerated degree (i.e., the sec_rec.deggrp is not blank), the deggrp_flag column in the sturpt_rec is set to Y. If the course work the program processes is not related to accelerated degree (i.e., the sec_rec.deggrp is blank), the deggrp_flag column in the sturpt_rec is set to N.

How the Program Inserts the Rows

Once the Student Reporting program processes each student, it inserts a row into the sturpt_rec with all values set. The row the program inserts is specific to the id, prog, site, rpt_sess, rpt_yr, rpt_type, sum flag, and deggrp_flag values. The only time the deggrp_flag is Y is for deggrp students taking deggrp courses.

How the Program Inserts the Summary Rows

The Student Reporting program inserts summary rows only if the group to process parameter value is S for summary. CARS Solution assumes that summary records are used primarily for IPEDS reporting. You should create summary rows only after you run the Student Reporting program for all programs for both traditional and nontraditional students to add non-summary rows.

Creating summary records results in a single sturpt_rec for a given id, site, rpt_sess, rpt_yr, and rpt_type. This sturpt_rec reflects hour totals that cross programs and traditional/degree group characteristics.

The priority and tot flags in the prog_table determine how Student Reporting records are summed. The priority value indicates which program has the highest priority (0 is the highest value). The tot flag indicates whether statistics for this program are totaled into the highest priority program.

The sturpt_rec having the highest priority program for the id being processed contains the data that copies to the sturpt_rec summary row. However, the hours values of the summary row reflects hours not only for its program, but also hours values for any program where the associated prog_table.tot value is Y.

IPEDS reporting uses the summary rows from the sturpt_rec. Each student the program processes has only one summary row for a given site, rpt_sess, rpt_yr, and rpt_type.

Parameters for the Student Reporting Program (*sturpt*)

Introduction

CARS Solution contains parameters and compilation values for executing the Student Reporting program. You can specify parameters to compile Student Reporting in a specified manner at the time of execution.

Note: You also can specify compilation values with the includes for the Accelerated Degree product that affect the Student Reporting program.

The Student Reporting program determines student enrollment data and populates the Student Report record (sturpt_rec) with this information. Various menu options may be created to generate the desired output from this program, based on available parameters.

Parameter Syntax

The *sturpt* program relies on program screens. These screens are found in $CARSPATH/modules/. They must be modified to meet the needs of the institution.

The menu options for *sturpt* are found in $CARSPATH/menuopt/regist/programs.

You can display *sturpt* parameters by entering the following: **sturpt -,**

The following is the correct usage for running the Student Reporting program from the UNIX shell:

sturpt -s session -y year [-p prog] -L site -t report type [-d effective date]

[-b first begin date] [-e last begin date] [-f first time student date] [-r] [-D]

-g group to process - trad, deggrp, both, summary

Parameters that appear in brackets are optional. Parameters that do not appear in brackets are required.

Parameters

The following lists the parameters for running *sturpt*.

-s *session*
Required - Specifies the reporting session associated with the Student Report record. The default is the value of SESS_CUR.

-y *year*
Required - Specifies the reporting year associated with the reporting session. The default is the value of ACAD_YR_CURR.

-p *prog*
Optional - Specifies the program of students to process. The default is the value of PROG_UNDG.

-L *site*
Required - Specifies the site of students to process. The default is the value of CARSSITE.

-t *report type*
Required - Specifies the type of report to assign to records this program creates.

-d *effective date*

Optional - Specifies the effective date against which this process is to run. The default is today's date.

-b *first begin date*
Optional - Specifies the beginning date in the range of dates against which this program evaluates courses in accelerated degree programs.

-e *last begin date*
Optional - Specifies the ending date in the range of dates against which this program evaluates courses in accelerated degree programs.

-f *first time student date*
Optional - Specifies the date against which this program evaluates a student to determine first time student status for IPEDS reporting.

-r
Optional - Specifies to allow a rerun of this process although Student Report records already exist. The default is N.

-D
Optional - Specifies to select course work by date, not session and year, even for traditional students. The default is N.

-g *group to process*
Required - Specifies the population of students to process.

Program Screens and Windows

Introduction

The Student Reporting program (*sturpt*) has screens and windows for performing the following interactive functions:

- Determining student enrollment data
- Populating the Student Report record (sturpt_rec)

Access

The screen and window files for the Student Reporting program are located in the following directory paths:

- $CARSPATH/modules/Lib/progscr/libentry

Notes:

- You can access windows from each program screen in Student Reporting.
- See the *CARS Solution Technical Manual - Volume II (System Reference)* for information about common windows that appear in Accelerated Degree.

Screen Files and Table/Record Usage

The Student Reporting screens and windows appear in the following files and use the indicated tables and records.

sturpt

Contains the Student Reporting record.

Access: $CARSPATH/modules/Lib/progscr/libentry

Tables/Records:

- cw_rec
- profile_rec
- prog_enr_rec
- sec_rec
- stu_acad_rec
- sturpt_rec

Menus, Screens, Scripts, and Reports

Introduction

This section provides reference information on the following features of the Accelerated Degree product:

- Menu source files
- Menu option files
- PERFORM screens
- SQL scripts
- Csh scripts
- ACE reports
- Letters

Directory Locations

The features detailed in this section are located in the following directory paths:

Menu source files
$CARSPATH/menusrc/student/regist/accdeg
$CARSPATH/menusrc/student/regist/accdeg/classlist
$CARSPATH/menusrc/student/regist/accdeg/regist
$CARSPATH/menusrc/student/regist/accdeg/reports
$CARSPATH/menusrc/student/regist/accdeg/session

Menu option files
$CARSPATH/menuopt/regist

PERFORM screens
None

SQL scripts
$CARSPATH/modules/regist/informers

Csh scripts
$CARSPATH/modules/regist/scripts

ACE reports
$CARSPATH/modules/regist/reports

Letters
None

Menu Structure

Introduction

CARS Solution menus provide access to a functionally related group of menu options. For example, an CARS Solution user in the Admissions office can access all the options necessary for processing prospects, recruits, and applicants from his/her menu, but typically cannot access options for processing accounting information. Depending on the work responsibilities of the CARS Solution users at your institution, you can customize their menus to offer access to as many or as few menu options as desired.

The selections that appear on the CARS Solution menus at your institution are controlled by a variety of interrelated directories and files. This section explains how the directories and files define CARS Solution menus, and it also provides information about the standard CARS Solution menu options.

Directories and Files that Define Menu Structures

The directories and files that control the standard CARS Solution menus are:

- Menu source directories
- Menu description files
- Menu option files

Menu Source Directories

Menu source (menusrc) directories define the branches of menus and submenus that offer access to CARS Solution features. The highest level of the menu source directory, located at $CARSPATH/menusrc, contains files and subdirectories similar to the following:

```
admit/                      student/
fiscal/                     system/
instdev/                    utility/
menudesc
```

Note: In this example, six subdirectories (designated with a slash [/] after their names) and one file appear in the menusrc directory.

The Menu Description File: Example 1

Menusrc directories always contain a menu description (menudesc) file. The menudesc file defines the options available at the current menu level. To continue the example above, the menudesc file in the $CARSPATH/menusrc directory contains the following type of information:

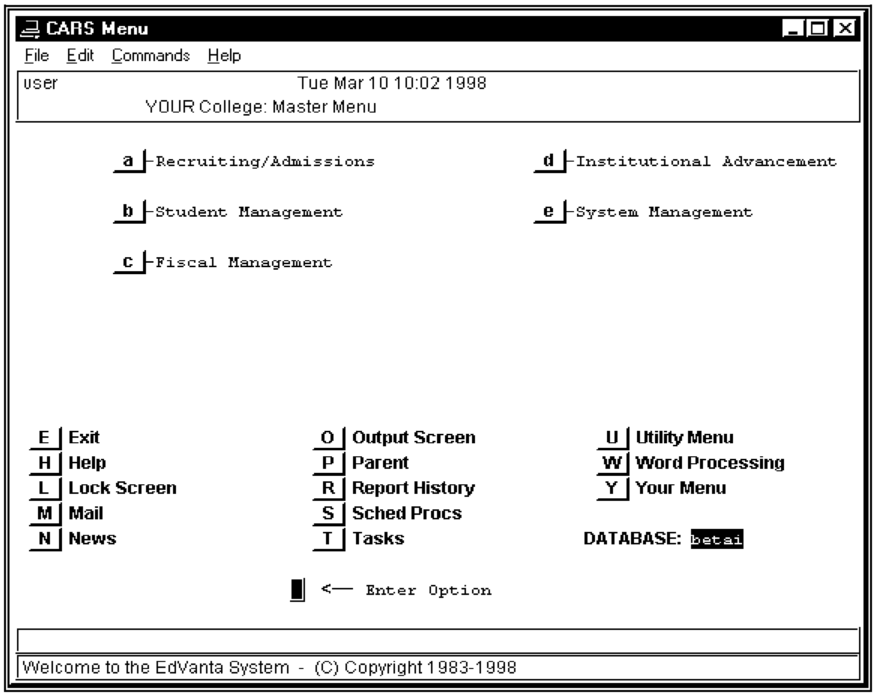
```
#
#Revision Information (Automatically maintained by 'make' - DON'T CHANGE)
#-----------------------------------------------------------------------
#$Header:
#$Log:
#
#-----------------------------------------------------------------------
#
#
TI=INST_NAME: Master Menu
SD=CARS Solution Master Menu
LD=Master Menu For INST_NAME
m4_keepif(ENABLE_MOD_ADMISSIONS, `Y')
MNU_SUB(admit)
m4_keepend
MNU_SUB(student)
MNU_SUB(fiscal)
m4_keepif(ENABLE_MOD_DEVELOP, `Y')
MNU_SUB(instdev)
m4_keepend
MNU_SUB(system)
```

Interpreting the menudesc File in Example 1

In this example, the menudesc file has the following information:

- Comment lines (lines beginning with a pound sign [#])
- Title and descriptive information (lines beginning with TI [title], SD [short description], or LD [long description])
- Keepif lines (lines indicating that a particular submenu is available if a corresponding ENABLE macro is set to Y)
- Keepend lines (lines indicating the end of a macro-controlled option)
- MNU_SUB lines (lines showing the name of the submenu)

Specifically, this example defines a menu called <institution name>: Master Menu. Assuming your institution has purchased the Admissions and Development applications and has enabled the ENABLE_MOD_ADMISSIONS and ENABLE_MOD_DEVELOP macros, your Master Menu would contain the following menu options:



```
user                         Tue Mar 10 10:02 1998
                 YOUR College: Master Menu

     a Recruiting/Admissions              d Institutional Advancement
     b Student Management                 e System Management
     c Fiscal Management

E Exit              O Output Screen        U Utility Menu
H Help              P Parent               W Word Processing
L Lock Screen       R Report History       Y Your Menu
M Mail              S Sched Procs
N News              T Tasks                DATABASE: betai

                    <— Enter Option

Welcome to the EdVanta System - (C) Copyright 1983-1998
```

Note: The names used for each of the submenus are controlled in their respective submenus.

Subdirectories in the menusrc Directory

Typically, the menusrc directory for a particular menu also contains subdirectories that relate to that menu's submenu options. In the example of the Master Menu, the following subdirectories are in the main $CARSPATH/menusrc directory, and they relate directly to the menu options that appear on the preceding menu screen example.

admit/
Recruiting/Admissions submenu

fiscal/
Fiscal Management submenu

instdev/
Institutional Advancement submenu

student/
Student Management submenu

system/
System Management submenu

utility/
Utility submenu (accessed from the lower part of the CARS Solution menu screen)

Contents of menusrc Subdirectories

The menusrc subdirectories have the same components as the main menusrc directory used in the preceding examples. Each contains subdirectories for more submenus, as well as a menudesc file that defines the appearance and contents of the submenu itself.

Navigating the menusrc Subdirectories

By using UNIX commands to move into any of the menusrc subdirectories and viewing its contents (i.e., its own subdirectories, if any, and its related menudesc file), you can view and map the menus, submenus, and menu options available at your institution. To continue the preceding example, if you enter **cd utility** at the UNIX prompt while in the $CARSPATH/menusrc directory, you can then enter **lsf** to display the contents of the utility subdirectory, as in the following:

```
Makefile            data/           iq/             ltrlblrep/       spooler/
RCS/                document/       login/          menudesc
adr/                files/          ltrlbl/         sbscr/
```

If you then enter **cd data** at the UNIX prompt and execute the **lsf** command again, you display the contents of the data subdirectory, as in the following:

```
Makefile            RCS             menudesc
```

In this example, only a menudesc file appears, indicating that no submenus exist at the $CARSPATH/menusrc/utility/data menu source level. The menudesc file in this case contains the actual menu options you can run from the Utilities/Data Entry menu. If desired, you can view the menudesc file using *vi*, *more*, or other UNIX commands.

The Menu Description File: Example 2

The menudesc file in Example 1 contains references only to submenus, that is, each line in the menudesc file that begins with MNU_SUB refers to a submenu. Another type of line can appear in the menudesc file: the MNU_OPT line. MNU_OPT lines do not refer to submenus; instead, they provide the link from the menu to the actual process (e.g., a program, script, report, or screen) you can run. For example, the menudesc file in the $CARSPATH/menusrc/utility/data directory contains the following type of information:

```
#
#Revision Information (Automatically maintained by 'make' - DON'T CHANGE)
#-------------------------------------------------------------------------
#$Header: menudesc,v 8.1 97/10/03 15:34:36 jdoe Released $
#-------------------------------------------------------------------------
#
TI=Utilities: Data Entry Menu
SD=Data Entry
LD=Data Entry
PW=@STD
MNU_OPT(common/programs/ide)
```

Interpreting the menudesc File in Example 2

In this example, the menudesc file has the following information:

- Comment lines (lines beginning with a pound sign [#])
- Title and descriptive information (lines beginning with TI [title], SD [short description], or LD [long description])
- Password information (the line beginning with PW)
- MNU_OPT line (the line showing the location of the menu option itself)

Notes:

- As with the menudesc file in Example 1, this file also can contain Keepif and Keepend lines to control the availability of optional processes.
- Although the menudesc file in Example 2 contains only one menu option (common/programs/ide), menudesc files can contain as many options as you care to display on a single menu.

Menu Option Files

The menudesc file for a particular menu or submenu points to the specific menu options you can run (e.g., Example 2 referred to the single menu option common/programs/ide, indicating that a common program is the only option available from the selected submenu).

To locate the menu option files for a particular menu, use the following command syntax:

cd menuopt/<directory on menudesc line>

For Example 2, the appropriate command is:

cd menuopt/common/programs

When you list the contents of the directory, the list resembles the following:

```
Makefile            ctcbatchr        ide              tuserid
RCS/                evnte            ide,fa
adrtest             fo.op            perm.a
ctcbatch            fo.stu           perm.c
```

This directory example contains several menu option files that are linked to other menudesc files. By their names, you usually can determine the process the menu option executes. For example, ide and ide.fa run versions of *identry*, and fo.op and fo.stu run versions of *forment*. More than one version of these processes exist because, depending on the needs of each menu user, you may want to use different titles or parameters.

You can view the specific menu option file using *vi*, *more*, or other UNIX commands.

Interpreting the Menu Option File

The menu option file contains three primary types of information:

- A definition of the parameter screen that appears when a user selects the option from the menu, including information for comment lines
- The name of the screen, program, script, or report the menu option executes
- The parameters, if any, that determine how the menu option should execute

An excerpt from the ide file demonstrates these three types of information. The horizontal lines on the excerpt divide the types of information and have been added to help you interpret the file's contents.

```
-------------------------------- Parameter screen ---------------------------------------
}
screen
{
                    m4_center_clipped(ID DATA ENTRY,40)
                    m4_center_clipped(PP_OFFICE[PA5], 40)
                    m4_center_clipped(PP_TICK[PA7], 40)
}
end
attributes
SD: optional,
    default = "ID Data Entry";
PW: optional,
    default = "@STD";
RD1: optional,
    default = "`Functions:'                          ";
RD2: optional,
    default = "";
RD3: optional,
    default = "`Enter ID Records for Individuals'    ";
RD4: optional,
    default = "Enter ID Records for Non-individuals";
PR: optional,
---------------------------------Executed Process---------------------------------------
    default = "BIN_PATH/identry";
-------------------------------Processing Parameters------------------------------------
m4_keepif(ENABLE_FEAT_AUTOMODE, `Y')
PA1: optional,
    default = "-a";
m4_keepelse
m4_keepif(ENABLE_FEAT_FORCEQUERY, `Y')
PA1: optional,
    default = "-F";
m4_keepend
m4_keepend
PA2: optional,
    default = "-D";
PA3: optional,
    default = "3";
PA4: optional,
    default = "-o";
LU5 = ofc_table.txt, optional;
PA5: optional,
    comments = "COMMENT_OFC_TBCODE  COMMENT_TBL",
    length = 4,
    lookup LU5 joining *ofc_table.ofc,
    upshift;
PA6: optional,
    default = "-T";
LU7 = tick_table.txt, optional;
PA7: optional,
    comments = "COMMENT_TICK  COMMENT_TBL",
    length = 4,
    lookup LU7 joining *tick_table.tick,
    upshift;
end
```

Determining the Type of Executed Process

In the preceding example, the location of the menu option ($CARSPATH/menuopt/common/programs) indicates that the menu option executes a program. The reference to BIN_PATH within the menu option file also indicates the execution of a program.

However, many menu options execute other types of processes (e.g., screens, scripts, or reports).

The following list shows the type of process executed, the _PATH reference from the menu option file, and the type of menu option directory path in which the process resides.

Process type	_PATH reference	Menu option directory path
Program	BIN_PATH	/programs
C shell script	SCP_PATH	/scripts
Report	ARC_PATH OTH_PATH	/reports /others
SQL statement	INF_PATH	/informers
PERFORM screen	SCR_PATH FRM_PATH	/screens
Utility	UTL_PATH	/utilities

Summary of Menu Source, Menu Description, and Menu Option Information

When you need to modify any menu options in use at your institution, remember the following:

- Menu source directories reflect your CARS Solution menus and submenus.
- Menu source directories contain other menu source subdirectories and menu description files.
- Menu description files define the menu options that appear on each menu.
- Menu description files contain lists of the menu options and submenus.
- Menu option files contain the instructions needed to execute each CARS Solution process on every menu.

Menu Options

Programs, Screens, and Scripts

The following list associates each Accelerated Degree program, screen, and script menu option and corresponding menuopt file and identifies the menuopt locations and what the menu option accesses.

Note: The following menus and options are listed in the order in which they appear on the standard CARS Solution menu structure. Italicized parameters indicate those that a user can enter or change.

Registrar: Reports Menu

Student Locator Report

Accesses: $CARSPATH/modules/regist/reports/stulocat (ACE report)

File: $CARSPATH/menuopt/regist/reports/stulocat

Parameters Passed:
- *Session*
- *Year*
- *ID#*

Registrar: Session Processing Menu

Unmet Requisites

Accesses: $CARSPATH/modules/regist/scripts/insprereq (Csh script)

File: $CARSPATH/menuopt/regist/scripts/insprereq

Parameters Passed:
- *Catalog*
- *Session*
- *Academic Year*
- *Program*
- *ID#*
- *Course Number*
- *Section*
- *Requisites Category*
- *List Met Requisites*

Registrar: Holds Processing Menu

Holds by Student

Accesses: $CARSPATH/modules/regist/reports/holdstu (ACE report)

File: $CARSPATH/menuopt/regist/reports/holdstu

Parameters Passed:
- *ID#*

Holds by Office

Accesses: $CARSPATH/modules/regist/reports/holdofc (ACE report)

File: $CARSPATH/menuopt/regist/reports/holdofc

Parameters Passed:
- *Office*

Registrar: Accelerated Degree Menu

Note: This submenu is available if the macro ENABLE_FEAT_ACCEL_DEG is set to Y.

Accelerated Degree: Registration Menu

Degree Group Registration

Accesses: $CARSPATH/modules/regist/scripts/grpreg (Csh script)

File: $CARSPATH/menuopt/regist/scripts/grpreg

Parameters Passed:
- *ADD/DROP*
- *Degree Group*
- *Program*
- *ID#*
- *Course Number*
- *Section*
- *Catalog*
- *Session*
- *Academic Year*
- *Effective Date*

Select SDS – Acc Deg

Accesses: $CARSPATH/modules/regist/reports/sdsctc (ACE report)

File: $CARSPATH/menuopt/regist/reports/adsdsctc

Parameters Passed:
- *Session*
- *Academic Year*
- *Program*
- *Subprogram*
- *Status*
- *Resource*
- *Degree Group*
- *Update*

Accelerated Degree: Class Lists Menu

Class Lists – Acc Deg

Accesses: $CARSPATH/src/regist/reglist (Program)

File: $CARSPATH/menuopt/regist/programs/regl.cZ

Parameters Passed:
- -y *year* (Academic year)
- -s *session* (Academic session)
- -C *catalog* (Course catalog of students to select)
- -Z *degree group* (Degree group of students to select)
- -f (Sort class lists by faculty names)
- -a (Print courses without students)

- -F (Class list form code)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -X (Sort students by course number for cross-listed class lists)
- -B (Print both confirmed and financially cleared students)
- -I (Print only students with financial clearance)
- -R (Print only confirmed students)

Final Grades – Acc Deg

Accesses: $CARSPATH/src/regist/reglist (Program)

File: $CARSPATH/menuopt/regist/programs/regl.fZ

Parameters Passed:
- -y *year* (Academic year)
- -s *session* (Academic session)
- -C *catalog* (Course catalog of students to select)
- -Z *degree group* (Degree group of students to select)
- -f (Sort class lists by faculty names)
- -a (Print courses without students)
- -F (Class list form code)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -X (Sort students by course number for cross-listed class lists)
- -B (Print both confirmed and financially cleared students)
- -I (Print only students with financial clearance)
- -R (Print only confirmed students)

Classes By Date – Acc Deg

Accesses: $CARSPATH/src/regist/reglist (Program)

File: $CARSPATH/menuopt/regist/programs/regl.cghZ

Parameters Passed:
- -y *year* (Academic year)
- -s *session* (Academic session)
- -C *catalog* (Course catalog of students to select)
- -Z *degree group* (Degree group of students to select)
- -f (Sort class lists by faculty names)
- -a (Print courses without students)
- -F (Class list form code)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -X (Sort students by course number for cross-listed class lists)

- -g *begin date range* (First date in the section begin date range)
- -h *end date range* (Last date in the section begin date range)
- -B (Print both confirmed and financially cleared students)
- -I (Print only students with financial clearance)
- -R (Print only confirmed students)

Grade Lists – Acc Deg

Accesses: $CARSPATH/src/regist/reglist (Program)

File: $CARSPATH/menuopt/regist/programs/regl.gdeZ

Parameters Passed:
- -y *year* (Academic year)
- -s *session* (Academic session)
- -C *catalog* (Course catalog of students to select)
- -f (Sort class lists by faculty names)
- -a (Print courses without students)
- -F (Class list form code)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -t *special text* (Special text to appear on the class list)
- -X (Sort students by course number for cross-listed class lists)
- -Z *degree group* (Degree group of students to select)
- -d *begin date range* (First date in section end date range)
- -e *end date range* (Last date in section end date range)
- -B (Print both confirmed and financially cleared students)
- -I (Print only students with financial clearance)
- -R (Print only confirmed students)

Accelerated Degree: Session Processing Menu

Academic Record Updates

Accesses: $CARSPATH/modules/regist/scripts/updstuac (Csh script)

File: $CARSPATH/menuopt/regist/scripts/updstuac.D

Parameters Passed:
- *Session*
- *Year*
- *Option*
- *Degree Group*

Accomp – Dean - ACC

Accesses: $CARSPATH/modules/regist/informers/addeanaccomp (SQL script)

File: $CARSPATH/menuopt/regist/informers/addeanaccomp

Parameters Passed:
- *Session*
- *Academic Year*
- *Date*
- *GPA*
- *Degree Group*

Accelerated Degree: Reports Menu

Counts and Lists

Accesses: $CARSPATH/modules/regist/others/adcountrpt (ACE report)

File: $CARSPATH/menuopt/regist/others/adcountrpt

Parameters Passed:
- *Primary Sort Field*
- *Secondary Sort Field*
- *Session*
- *Academic Year*
- *Program*
- *Subprogram*
- *Status*
- *Classification*
- *Major*
- *ID#*
- *Interest*
- *School Type*
- *New Page*
- *Summary*
- *Report Type*
- *Include summary records*

Student Schedule

Accesses: $CARSPATH/modules/regist/reports/adschd (ACE report)

File: $CARSPATH/menuopt/regist/reports/adschd

Parameters Passed:
- *Degree Group*

Acc Deg Roster

Accesses: $CARSPATH/modules/regist/reports/adroster (ACE report)

File: $CARSPATH/menuopt/regist/reports/adroster

Parameters Passed:
- *Campus*
- *Degree Group*
- *Session*
- *Academic Year*
- *Include Deggrp*

Grades by Dept

Accesses: $CARSPATH/modules/regist/reports/addeptgrd (ACE report)

File: $CARSPATH/menuopt/regist/reports/addeptgrd

Parameters Passed:
- *Catalog*
- *Percentage*
- *Report*
- *Hour Counts*

- *Grade Result*
- *Degree Group*
- *Beginning Date*
- *Ending Date*

NSLC Using Sturpt Data

Accesses: $CARSPATH/modules/regist/scripts/adnslc (ACE report)

File: $CARSPATH/menuopt/regist/scripts/adnslc

Parameters Passed:
- *School Number*
- *Branch Code*
- *Session*
- *Year*
- *Program*
- *Type*
- *Begin date for accel degree*
- *Standard Reporting Flag*
- *Credit Hours*
- *Verification Report*

Degree Group Report

Accesses: $CARSPATH/modules/regist/reports/deggrp (ACE report)

File: $CARSPATH/menuopt/regist/reports/deggrp

Parameters Passed:
- *Campus*
- *Session*
- *Academic Year*
- *Program*
- *Begin Date Range*
- *End Date Range*

Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree

Create Enr Data-IPEDS

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt.bet

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)
- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students who are processed)
- -t (Report type to assign to records this process creates)
- -d *effective date* (Effective date for this run)
- -f *first time student date* (Date against which to determine first time student status)
- -b *first begin date* (Beginning date for selecting accelerated courses)
- -e *last begin date* (Ending date for selecting accelerated courses)
- -g *group to process* (Code of the group to process)
- -r (Allow rerun of process although records exist)

Create Enr Data-IPEDS Sum

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt.gt

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)
- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students to process)
- -t (Report type to assign to records this process creates)
- -g (Code of the group to process)
- -r (Allow rerun of process although records exist)

Student Detail Report

Accesses: $CARSPATH/modules/regist/reports/sturpt (ACE report)

File: $CARSPATH/menuopt/regist/reports/sturpt

Parameters Passed:
- *Session*
- *Academic Year*
- *Program*
- *Type*
- *Summary*
- *Code*

First Time Flags Report

Accesses: $CARSPATH/modules/regist/reports/ipedchkft (ACE report)

File: $CARSPATH/menuopt/regist/reports/ipedchkft

Parameters Passed:
- *Session*
- *Academic Year*
- *Program*
- *Summary*

IPEDS Part A-Undergrad Enr

Accesses: $CARSPATH/modules/regist/scripts/ipedenru (Csh script)

File: $CARSPATH/menuopt/regist/scripts/ipedenru

Parameters Passed:
- *Session*
- *Academic Year*
- *Detail*

IPEDS Part A-Graduate Enr

Accesses: $CARSPATH/modules/regist/scripts/ipedenrg (Csh script)

File: $CARSPATH/menuopt/regist/scripts/ipedenrg

Parameters Passed:
- *Session*
- *Academic Year*

- *Program*
- *Detail*

IPEDS Part B-Age

Accesses: $CARSPATH/modules/regist/scripts/ipedageall (Csh script)

File: $CARSPATH/menuopt/regist/scripts/ipedageall

Parameters Passed:

- *Session*
- *Academic Year*
- *Program*
- *Detail*

IPEDS Part C-First-Time Fr

Accesses: $CARSPATH/modules/regist/reports/ipedresff (ACE report)

File: $CARSPATH/menuopt/regist/reports/ipedresff

Parameters Passed:

- *Session*
- *Academic Year*
- *Detail*

IPEDS Part D. Create Data – IPEDS Yr

Accesses: $CARSPATH/src/regist/sturpt

File: $CARSPATH/menuopt/regist/programs/srpt.bety

Parameters Passed:

- Session
- Academic Year
- Program
- Effective Date
- Date to Determine First Time
- Beginning Date
- Ending Date
- Group to Process

IPEDS Part D. Create Data – IPEDS Yr Sum

Accesses: $CARSPATH/src/regist/sturpt

File: $CARSPATH/menuopt/regist/programs/srpt.gty

Parameters Passed:

- Session
- Year

IPEDS Part D. IPEDS Undg Yearly Enr

Accesses: $CARSPATH/modules/regist/reports/ipedenru.y

File: $CARSPATH/menuopt/regist/scripts/ipedenru.y

Parameters Passed:

- Session
- Academic Year
- Detail

IPEDS Part D. IPEDS Grad Yearly Enr

Accesses: $CARSPATH/modules/regist/reports/ipedenrg.y

File: $CARSPATH/menuopt/regist/scripts/ipedenrg.y

Parameters Passed:
- Session
- Academic Year
- Program
- Detail

IPEDS Part D. IPEDS Credit and Clock Hrs

Accesses: $CARSPATH/modules/regist/reports/ipeddhrs

File: $CARSPATH/menuopt/regist/reports/ipeddhrs

Parameters Passed:
- Program
- Beginning Date
- Ending Date

IPEDS Degree Completion

Accesses: $CARSPATH/modules/regist/reports/ipeddeg (ACE report)

File: $CARSPATH/menuopt/regist/reports/ipeddeg

Parameters Passed:
- *Beginning Date*
- *Ending Date*
- *Site*
- *Program*
- *Undecided major code*
- *Detail*

Print Occupational Majors

Accesses: $CARSPATH/modules/regist/reports/ipedocc (ACE report)

File: $CARSPATH/menuopt/regist/reports/ipedocc

Parameters Passed: None

IPEDS Occ Specific Program

Accesses: $CARSPATH/modules/regist/reports/ipedoccmaj (ACE report)

File: $CARSPATH/menuopt/regist/reports/ipedoccmaj

Parameters Passed:
- *Session*
- *Academic Year*
- *Detail*

Registrar: Static Enrollment Data

Create Enroll Data - Trad

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)
- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students who are processed)
- -t *report type* (Report type to assign to records this process creates)
- -d *effective date* (Effective date for this run)
- -f *first time student date* (Date against which to determine first time student status)
- -g (Code of the group to process)
- -r (Allow rerun of process although records exist)

Create Enroll Data

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt.be

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)
- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students who are processed)
- -t *report type* (Report type to assign to records this process creates)
- -d *effective date* (Effective date for this run)
- -f *first time student date* (Date against which to determine first time student status)
- -b *first begin date* (Beginning date for selecting accelerated courses)
- -e *last begin date* (Ending date for selecting accelerated courses)
- -g *group to process* (Code of the group to process)
- -r (Allow rerun of process although records exist)

Create Enroll Data-by Date

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt.beD

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)
- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students who are processed)
- -t *report type* (Report type to assign to records this process creates)
- -D (Select students by date, not by session)
- -f *first time student date* (Date against which to determine first time student status)
- -b *first begin date* (Beginning date for selecting accelerated courses)
- -e *last begin date* (Ending date for selecting accelerated courses)
- -g *group to process* (Code of the group to process)
- -r (Allow rerun of process although records exist)

Create Enroll Data - Sum

Accesses: $CARSPATH/src/regist/sturpt (Program)

File: $CARSPATH/menuopt/regist/programs/srpt.g

Parameters Passed:
- -s *session* (Beginning date for selecting accelerated courses)

- -y *year* (Reporting year for this data snapshot)
- -p *program* (Program code)
- -L (Site of students who are processed)
- -t *report type* (Report type to assign to records this process creates)
- -g (Code of the group to process)
- -r (Allow rerun of process although records exist)

Data Entry

Accesses: $CARSPATH/src/regist/stuentry (Program)

File: $CARSPATH/menuopt/regist/programs/stue

Parameters Passed:
- -a (Pass parameter to enter Query mode automatically)
- -F (Pass parameter to force query attempt before allowing Insert mode)
- -L (Site code to be used)
- -D (Debug_level 3)

Student Detail Report

Accesses: $CARSPATH/modules/regist/reports/sturpt (ACE report)

File: $CARSPATH/menuopt/regist/reports/sturpt

Parameters Passed:
- *Session*
- *Academic Year*
- *Program*
- *Type*
- *Summary*
- *Code*

SQL Scripts

Introduction

The Accelerated Degree product contains SQL scripts that perform queries and produce reports from database records. The scripts are located in the following directory path: $CARSPATH/modules/regist/informers

Note: Csh scripts can call ACE reports and SQL scripts. Such ACE reports and SQL scripts do not reside on the CARS Solution menu system.

SQL Scripts

The following lists the SQL scripts provided with Accelerated Degree.

addeanaccomp
Initiates a process that adds Accomplishment records for those students in a specified degree group who qualify for the dean's list.

Menu Access: Accelerated Degree: Session Processing menu: Accomp – Dean – ACC menu option

Tables/Records Used:
- stu_acad_rec

Csh Scripts

Introduction

Accelerated Degree contains Csh scripts to automate the processing of information. Csh scripts are UNIX-based program statements that can execute a series of UNIX commands, such as SQL scripts and ACE reports. The Accelerated Degree Csh scripts are located in the following directory path: $CARSPATH/modules/regist/scripts

Csh Scripts

The following list associates an Accelerated Degree menu option with the corresponding Csh script and provides a description of the script.

Note: In the following list, descriptions of Csh scripts include:

- Purpose of the script
- Menu access option, if applicable
- A list of ACE reports used, if applicable
- A list of SQL scripts used, if applicable
- A list of tables used, if applicable

updstuac

Initiates a process that updates and/or prints rows where selected information in the Academic record and Program Enrollment record differ.

Menu Access: Accelerated Degree: Session Processing menu: Academic Record Updates menu option

Tables/Records Used:

- prog_enr_rec
- stu_acad_rec

Perl Scripts

Introduction

Registration uses Perl scripts to manage processing. The Perl scripts are located in the following directory path:

- $CARSPATH/modules/regist/scripts

Perl Scripts

The following list provides a description of the Perl script.

Note: In the following list, descriptions of Perl scripts include:

- Purpose of the script
- A list of tables used, if applicable

batchreg

This script enables the batch registration of a set of students for a set of courses for a given session, year, and program.

Tables/Records Used:

- id_rec
- prog_enr_rec
- sec_rec

insprereq

This script enables the reporting of unmet prerequisites for students. This perl script will add prereqrpt_recs, which then will be used in an ACE report to provide actual output.

Tables/Records Used:

- prereqrpt_rec

ACE Reports

Introduction

CARS Solution contains ACE reports for easy reporting of Accelerated Degree database information.

ACE Reports

The following lists the ACE reports provided with Accelerated Degree. Some reports listed do not appear on the CARS Solution menu system because they are used only in Csh scripts.

Contacts for SDS
Provides a list of persons who have Contact records to receive student data sheets.

`Menu Access:` Accelerated Degree: Registration menu: Select SDS – Acc Deg menu option

File: $CARSPATH/modules/regist/reports/sdsctc

Data Consistency Report - sturpt
Provides the contents of the Student Reporting record (sturpt_rec) in a detail report, including an option to run a data validation check.

`Menu Access:` Registrar: Static Enrollment Data menu: Student Detail Report menu option. Also, Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: Student Detail Report menu option.

File: $CARSPATH/modules/regist/reports/sturpt

Degree Group Summary Report
Provides total student enrollments by degree group, program, and campus. You can specify an optional date range, session, and year to restrict the selection of degree groups.

`Menu Access:` Accelerated Degree: Reports menu: Degree Group Report menu option

File: $CARSPATH/modules/regist/reports/deggrp

Final Grade Dist. by Department
Provides the grade distribution for each course each department teaches for a specified session and year.

`Menu Access:` Accelerated Degree: Reports menu: Grades by Dept menu option.

File: $CARSPATH/modules/regist/reports/addeptgrd

First Time Flag Check Report - sturpt
Provides a detail report listing students who have an inconsistency in their first time flags (e.g., first time freshman, first time transfer).

`Menu Access:` Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: First Time Flags Report menu option.

File: $CARSPATH/modules/regist/reports/ipedchkft

IPEDS Completions – Including Degrees in Education Record
Provides counts of degrees awarded sorted by major over any specified date range. Totals are given by gender within ethnic groups.

`Menu Access:` Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Degree Completion menu option.

File: $CARSPATH/modules/regist/reports/ipeddeg

IPEDS Enrollment in Occupationally Specific Programs

Provides the counts of students enrolled in occupationally specific programs, sorted by CIP codes, gender, and ethnic group.

Menu Access: Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Occ Specific Program menu option.

File: $CARSPATH/modules/regist/reports/ipedoccmaj

IPEDS Enrollment Report – All Undergraduate Programs

Provides the counts of full-time and part-time students sorted by classification for all undergraduate programs. The totals are given by gender within ethnic groups.

Menu Access: Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Part A – Undergrad Enr menu option.

File: $CARSPATH/modules/regist/reports/ipedenru

IPEDS Enrollment Report by Age

Provides the counts of full-time and part-time students sorted by age. The categories for each age bracket are undergraduate, first professional, and graduate program students.

Menu Access: Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Part B – Age menu option.

File: $CARSPATH/modules/regist/reports/ipedageall

File: $CARSPATH/modules/regist/reports/ipedageft

File: $CARSPATH/modules/regist/reports/ipedagept

IPEDS Graduate Enrollment Report

Provides the counts of full-time and part-time students for specified graduate programs. The totals are given by gender within ethnic groups.

Menu Access: Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Part A – Graduate Enr menu option.

File: $CARSPATH/modules/regist/reports/ipedenrg

IPEDS Part D – Create Enrollment Data Yearly

Creates student report records (sturpt_rec) for all students who have taken coursework over a specified date range.

Menu Access: Registrar Reporting/External Agencies: IPEDS Reports Menu: IPEDS Part D – Create Data - IPEDS Yr.

File: $CARSPATH/menuopt/regist/programs/srpt.bety

IPEDS Part D – Create Enrollment Data Yearly Summary

Creates summary student report records (sturpt_rec) using the student report records that were created in the Create Data – IPEDS Yr.

Menu Access: Registrar Reporting/External Agencies: IPEDS Reports Menu: IPEDS Part D – Create Data - IPEDS Yr Sum.

File: $CARSPATH/menuopt/regist/programs/srpt.gty

IPEDS Part D - Undergraduate Yearly Enrollment

Provides results of the Create Enrollment Data Yearly Summary option for the undergraduate program.

Menu Access: Registrar Reporting/External Agencies: IPEDS Reports Menu: IPEDS Part D – IPEDS Undg Yearly Enr.

File: $CARSPATH/menuopt/regist/scripts/ipedenru.y

IPEDS Part D – Graduate Yearly Enrollment

Provides results of the Create Enrollment Data Yearly Summary option for the programs specified in the parameters.

`Menu Access:` Registrar Reporting/External Agencies: IPEDS Reports Menu: IPEDS Part D – IPEDS Grad Yearly Enr.

File: $CARSPATH/menuopt/regist/scripts/ipedenrg.y

IPEDS Part D – Credit and Clock Hours

Provides the total number of credit and clock hours for registered and withdrawn coursework taken in the date range specified.

`Menu Access:` Registrar Reporting/External Agencies: IPEDS Reports Menu: IPEDS Part D – IPEDS Credit and Clock Hours.

File: $CARSPATH/menuopt/regist/reports/ipeddhrs

Parameters for Student Count Report

Provides the counts and lists of students. This report uses the Student Reporting record (sturpt_rec) as its base.

`Menu Access:` Accelerated Degree: Reports menu: Counts and Lists menu option.

File: $CARSPATH/modules/regist/others/adcountrpt

Report for NSLC – Using sturpt_rec

Provides the student enrollment verification to the National Student Loan Clearinghouse using the sturpt_rec as its database, not the stu_acad_rec.

`Menu Access:` Accelerated Degree: Reports menu: NSLC Using Sturpt Data menu option.

File: $CARSPATH/modules/regist/scripts/adnslc

Report of Occupationally Specific Majors Currently in the Major Table

Provides the majors in the Major table that are considered to be occupationally specific according to the NCES. This report is sorted by CIP number.

`Menu Access:` Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: Print Occupational Majors menu option.

File: $CARSPATH/modules/regist/reports/ipedocc

Residence of First-Time Freshmen

Provides the counts of undergraduate first-time freshmen by state of their residence. There is another column showing how many graduated from high school within one year of the beginning of the session (acad_cal_rec.beg_date).

`Menu Access:` Registrar Reporting: IPEDS Reports Menu – Incl. Accel. Degree: IPEDS Part C-First-Time Fr menu option.

File: $CARSPATH/modules/regist/reports/ipedresff

Roster of Accelerated Students

Provides a roster of accelerated degree students sorted by campus and degree group. A page break separates each degree group. The report lists students in each degree group sorted by last name.

`Menu Access:` Accelerated Degree: Reports menu: Acc Deg Roster menu option.

File: $CARSPATH/modules/regist/reports/adroster

Schedule of Classes by Degree Group

Provides a listing of all courses students in a degree group must take in their accelerated degree program, including instructor and meeting time (based on mtgdtl_rec) information. Accelerated Degree sorts the schedule by course number in the order in which students must take the courses.

`Menu Access:` Accelerated Degree: Reports menu: Student Schedule menu option.

File: $CARSPATH/modules/regist/reports/adschd

Producing Student Data Sheet Forms

Introduction

The Student Data Sheet (SDS) is available in two formats:

- One format is a one part form matching the size of the paper, typically 66 lines in length, containing registration information with or without billing and financial aid information.

  Default forms provided with the standard CARS Solution system in $CARSPATH/modules/regist/forms/regent are:
    - s80bill
    - sdsbill
    - sds80
    - ws80bill

  The “s80bill” form is a SDS with billing information.

  The "sds80" form is a SDS without billing information that is formatted for 80 columns. The “ws80bill” form includes billing information and is formatted for Web output.

- The other format is a multiple part form, which is combined from individual forms to create a composite SDS. It contains registration information, with or without billing and financial aid information, and cash receipt information.

  Default forms provided with the standard CARS Solution system in $CARSPATH/modules/regist/forms/regent are:
    - header
    - classes
    - remarks
    - sdsstmt

  The default form provided with the standard CARS Solution system in $CARSPATH/modules/accounting/forms/cashier is:

  cash_rcpt

Macros Used in SDS Production

The macros in custom/table must be set appropriately based on the names used in the SDS Form table (sdsform_table).

```
{>>REGISTRATION }
{*** Default regist form for student data sheets ***}
m4_define(`SDS_FORM_DEF', `SDS')
m4_define(`SDS_FORM_INCL',
    `include=(SDS,SDSB,SDS8,SD8B), upshift')

m4_define(`SDSBILL_FORM_DEF', `SDSB')
```

The macros in custom/student must be set appropriately based on the needs of the institution.

```
{>>REGISTRATION }
{*** Set to Y to get SDS billing address from subscription records      ***}
m4_define(`ENABLE_FEAT_GET_SBSCR', `Y')
```

```
{>>REGISTRATION }
{*** Set this to Y to include wait listed courses on a SDS form.  ***}
m4_define(`SDS_INCLUDE_WAIT_LIST', `N')

{>>REGISTRATION }
{*** Defines the replacement value of the beg/end time that appears on  ***}
{*** SDS forms for "meeting record begin time" when that time is zero.  ***}
m4_define(`MEETING_TBA_VALUE', `TBA')

{>>REGISTRATION }
{*** Set this to Y to print the section title before the course title   ***}
{*** on SDS forms.                                                      ***}
m4_define(`SDS_SECTION_TITLE_FIRST', `N')

{>>REGISTRATION }
{*** Specify the number of lines of title to be printed on the SDS      ***}
{*** form.  If all lines are to be printed (3 lines of course title     ***}
{*** and one line of section title), this value would be set to 4.      ***}
{*** Only non-blank title lines are printed and included in this total. ***}
m4_define(`LINES_OF_SDS_CRS_TITLE', 4)

{>>REGISTRATION }
{*** Defines the valid values that can be passed to enable the -F       ***}
{*** (fee collection) parameter in the regent program.  The use of the F         ***}
{*** in the registration menuopt enables regent without fee collection. ***}
{*** DO NOT modify this macro.                     ***}
m4_define(`FINISH_ONLY', `F')
m4_define(`FINISH_SDS', `S')
m4_define(`FINISH_POST_SDS', `P')
m4_define(`FINISH_FEES_POST_SDS', `C')
```

SDS Form Design Options

All SDS forms must be located in one of two directories:

- $CARSPATH/modules/regist/forms/regent/...
- $CARSPATH/modules/accounting/forms/cashier/...

Note: The only section of the form that can reside in the accounting path is one section of a multiple part form for printing a cash receipt.

To add a new form, in the CARS_PATH/modules/regist/forms/regist directory, copy one of the existing SDS forms (i.e., sds, sdsbill) to a new form name. Then do the following:

- Remove the "RCS" information from the copied form
- Modify form as desired
- Add and test the form

Note: The SDS form names are limited to 4 characters in length. When the form is installed, the Registration program allows the form to be selected, as long as the form is set up correctly according to the CARS Solution screen package requirements and the SDS Form table (sdsform_table) contains the valid required data for the form.

The SDS form prints the notation "NF," in the faculty abbreviated name field, if no faculty is assigned for this section. The "NF" represents, "No Faculty Assigned."

Single File SDS Form

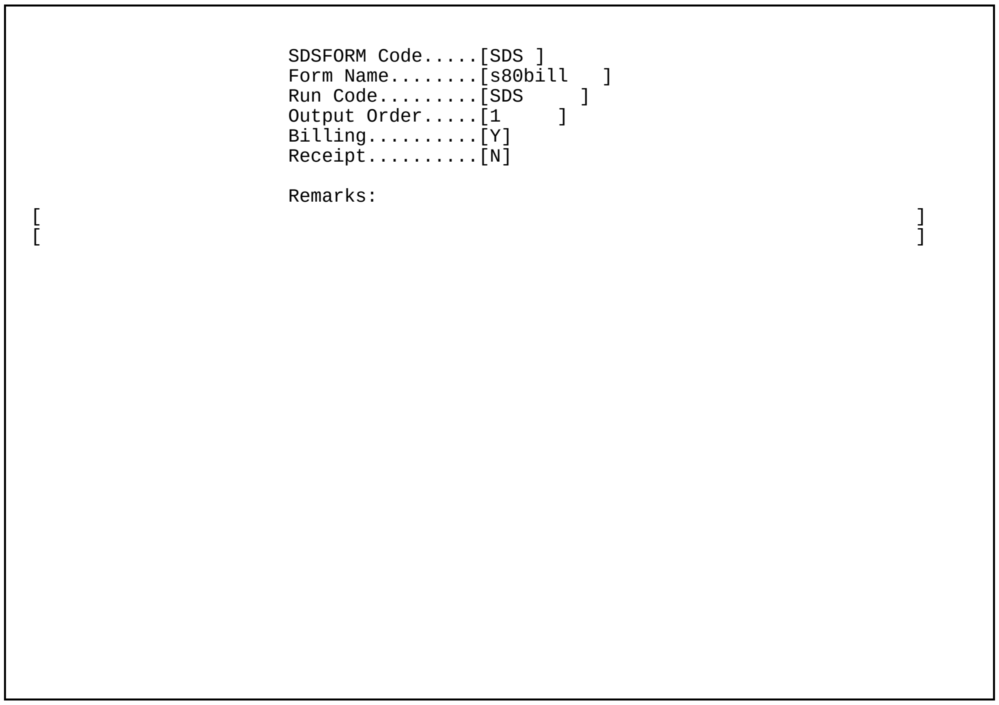
The following rules apply for a single file SDS form:

- In modules/regist/forms there should be a single form screen for the SDS type desired (e.g., s80bill).
- All rules for design and syntax of the form must be considered.
- All sections of the form should be included on one screen file.
- The name of the form must be at the bottom of the form file, linking this single screen to the merged information for production in the CARS Solution Forms Production System (FPS).

CARS Solution prints the form exactly as designed in this form screen. The form prints the exact number of lines from the form on the hard copy SDS. If any one of the sections of the form contains more information than can be expressed in the individual section, an entire second form is produced to express the data.

The "remarks" lines in the SDS Form table allow for two comment lines per form. If this feature is used, the remarks entered here display on the two remarks lines in the SDS form (these lines must appear in the attributes section of the file).

If the institution wishes to produce a single SDS, as past features of CARS Solution provided, you need only a single entry in the sdsform_table. The following is an example of a completed SDS Form table entry.



```
                       SDSFORM Code.....[SDS ]
                       Form Name........[s80bill   ]
                       Run Code.........[SDS     ]
                       Output Order.....[1     ]
                       Billing..........[Y]
                       Receipt..........[N]

                       Remarks:
[                                                                              ]
[                                                                              ]
```

Multiple File SDS form

The following rules apply for a multiple file SDS form:

- In modules/regist/forms there should be a multiple form screens for the SDS type desired (e.g., s80bill).
- All rules for design and syntax of the form must be considered.
- Each section of the form should have a single screen file.
- The same name of the form must be at the bottom of each form file, linking this form screen to the merged information for production in the CARS Solution Forms Production System (FPS).

CARS Solution links and prints the forms as one SDS based on the design of the screens and the order in the sdsformn_table. It is important to note that the combination of the individual screen files must be equal to the size of the printed form. In addition, if any one of the sections of the form contains more information than can be expressed in the individual section, only one section of the form is duplicated to express the data.

The "Remarks:" lines in the SDS Form table allow for two comment lines per section of the form. If this feature is used, the remarks entered in the table require the form spacing design to allow for the insertion of the desired text per form. It is possible to have text at each level of the form, however, the space must be allocated for each set of comments.

If the institution wishes to produce a multiple part SDS, there must be multiple entries in the sdsform_table. The following is an example of a completed SDS Form table entry for a multiple file SDS form.

Notes:

- You can produce only one SDS using this method (i.e., you cannot use batch).
- It is recommended the institution use individual screens that are 22 lines in length, and purchase paper that is perforated in 22 line increments.

```
Entry 1
                        SDSFORM Code.....[SDS ]
                        Form Name........[header    ]
                        Run Code.........[SDS     ]
                        Output Order.....[1     ]
                        Billing..........[Y]
                        Receipt..........[N]

                        Remarks:
 [                                                                               ]
 [                                                                               ]


 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -   Entry 2
                        SDSFORM Code.....[SDS ]
                        Form Name........[classes   ]
                        Run Code.........[SDS     ]
                        Output Order.....[2     ]
                        Billing..........[Y]
                        Receipt..........[N]

                       Remarks:
 [                                                                               ]
 [                                                                               ]
```

Producing an Interactive SDS in Registration

The Output option in the *regent* ring menu produces an interactive SDS with biographical, academic, and housing data; a record of all courses for which the student is waiting or registered; and billing and financial aid information.

Note: The Output option is designed to provide a complete SDS for the students during advising or registration.

The SDS is spooled to the printer specified on the parameter screen with the formtype "sds" or "sdsbill" indicating that the printer formtype must be the same before the SDS will be printed.

Producing SDS Forms in Batch

The production of SDS forms in batch is managed by a "C" program, *sdsbatch*. The process requires the institution to create a series of Contact records for the *sdsbatch* program to use in the identification of students who should receive the forms. The CARS Solution menu, Registrar: Registration Menu contains all of the functions for this process.

If only a few SDS's are to be run, Contact records can be created either through a PERFORM data entry screen or with an SQL statement containing the specified conditions students must meet.

Note: The Contact Table Tickler field (ctc_tick) must be: the tickler for the Registrar's office; the Contact Table Resource field (ctc_resrc) must match the form code (in uppercase, maximum of four characters) used in the SDS Form table (sdsform_table).

Once the SDS's are created, they should be printed using the Forms Production System (FPS). The Contact records created to select the students who are to receive the SDS's, are updated so that the contact status will be "C" for contacted.

Note: A menu option for printing SDS's with billing data in batch also appears on the Student Billing menu.

The use of multiple part forms in the *sdsbatch* program requires the production of SDS forms immediately to the printer (the -X option).

Parameters for the *sdsbatch* Program

The following are the parameters for the *sdsbatch* program.

```
Usage: sdsbatch [-c sched] -f resource -P printer -y year -s sess -p prog [-t post]
       -d efdate -L site  [-i ID] [-r CorrID] [-X]

where:   sched:       Comments are included on the form (Y/N)
         resource:    Resource to use from the contact table
         printer:     Printer to be selected
         year:        Year to be selected for processing SDS forms
         sess:        Session to be selected for processing SDS forms
         prog:        Program to selected for processing SDS forms
         post:        Post transactions (Y/N)
         efdate:      Date printed on bills if any
         site:        Site to be selected for processing  SDS forms
         ID:          ID to be used for single SDS production
         CorrID:      Correspondent ID to use instead
         -X:          Direct output immediately to printer
```

Printing a SDS

The SDS is formatted in a file for printing, using the CARS Solution Forms Production System (FPS). The printer displays a formtype of "sds" or "sdsbill," depending on the form selected for printing.

Refer to documentation on FPS for further information on the actual printing of the forms.

Optional Financial Aid Message on Sdsbill

A message concerning financial aid can be printed on SDS forms that include billing information (i.e., sdsbill). This message is printed only when a student has financial aid.

To have this financial aid message printed, the FA_ macros in the $CARSPATH/include/applic/regent file need to be modified. The FA_MSG macro defines the message to be printed.

- Each line of the message needs to be enclosed in double quotes.
- Every line of the message except the last one should have a comma, a space, and a backslash following the ending double quote.
- Only a blank and then a backslash should follow the last line.

A sample message in the regist file is provided as a guide in creating the financial aid message. The macro FA_HRS can be used within the definition of FA_MSG to specify where the number of financial aid hours will be displayed, if the message is to contain the number of hours.

After the FA_MSG macro is defined, the FA_MSG_LEN macro needs to be modified to be the number of lines that are in FA_MSG.

If the macro FA_HRS is part of the definition of FA_MSG and the financial aid hours are stored in the stufa_rec, the macro USE_STUFA_HRS should be uncommented. Otherwise, USE_STUFA_HRS should be left commented. If the USE_STUFA_HRS macro is uncommented, then there should be a field added to the stufa_rec for each defined session. The data in these fields needs to be entered manually. The FA_MSG_SESS macro is used to link the session to the field in the stufa_rec that contains the number of financial aid hours. The definition for this macro contains the session (from the sess_table) and the stufa_rec field. There is an example in the $CARSPATH/include/applic/regent file that can be used as a guide for defining this macro.

If the FA_MSG macro definition contains the FA_HRS macro and the USE_STUFA_HRS macro is commented, then the DEF_HRS should be defined as the default number of financial aid hours.

After the modifications to the include file are completed and installed, the *regent* program needs to be reinstalled and the SDS form modified. The $CARSPATH/modules/regist/forms/<form name (i.e., sdsbill)> needs to have fields added to contain the financial aid message. These fields need to be of the form famsg0, famsg1, famsg2, etc. The number of fields to add should be the same number as defined by FA_MSG_LEN.

Registration Reports

Introduction

A report is a paper copy of the information contained in one or more tables. After you review or update tables in CARS Solution, it is good practice to print the corresponding report. The printed copy of the report allows you to verify that the information you added or updated in the table was changed in the database and that the information is set up correctly on the report.

Registration Reports

Access the reports corresponding to the Registration application by using the following procedure.

Type menu **at the prompt and press** <CR>.
The CARS College: Master Menu appears.

Select Student Management.
The Student Management: Main Menu appears.

Select Registrar.
The Student Management: Registrar Main Menu appears.

Select Reports.
The Registrar: Reports Menu appears.

Select the category of reports that you would like to have printed (e.g., Enrollment).
The main menu for the option you selected appears.

Select the specific report that you would like to have printed (e.g., Faculty).
A window containing information related to the report you selected appears.

Complete the required fields in the window, then select Finish.
The Output Parameters window appears

Complete the fields in the Output Parameters window, then select Finish.
The report you selected is printed.

Modifying the Registration Program (*regent*) Menu

Introduction

The CARS Solution Registration program, *regent*, provides the ability to define which options are available to the institution in three internal menus.

Menu options

The following lists the available menu options for the three internal menus in the Registration program.

Note: Enter only those options to be *excluded*.

Main regent ring menu (-m program parameter)

CAPACITY_OPT,	'P'
CONFIRM_OPT,	'C'
EXIT_OPT,	'E'
FEE_OPT,	'F'
HOLDS_OPT,	'H'
INIT_OPT,	'I'
OUTPUT_OPT,	'O'
QUERY_OPT,	'Q'
REGISTER_OPT	'R'
SESSION_OPT,	'S'

TAB enroll options window (-e program parameter)

ALTCAL_OPT,	'A'
CRSDTL_OPT,	'C'
DROP_OPT,	'D'
ENROLDTL_OPT,	'E'
LOCSEC_OPT,	'L'
MTGDTL_OPT,	'M'
VOIDSEC_OPT,	'V'

Ctrl N options window (-r program parameter)

BLKADD_OPT,	'B'
DISPMTGS_OPT	'M'
DISPREFNO_OPT	'R'
DROPALL_OPT,	'D'
EDATE_OPT,	'C'
FINAID_OPT,	'F'
LOCPRMS_OPT,	'S'
REGFEES_OPT,	'I'
STUINFO_OPT,	'E'
VOIDALL_OPT,	'V'

Modifying the Registration Program

To create a new menuopt in the $CARSPATH/menuopt/regist/programs directory, make a local copy of the *regent* menuopt. The new menuopt should include one or more of the desired parameters to be removed.

Example: Removing the Institutional Fees and Block Add options in the Cntrl N window:

- PA9: optional, default = "-r";
- PA10: optional, default = "I B";

Troubleshooting

Introduction

This section deals with undesirable or unexpected conditions that may occur in *regent*, along with their possible solutions. These conditions are addressed to minimize an operator’s research and/or need to seek assistance from Jenzabar, Inc.

Issues

The following lists some problems you may encounter and possible solutions for them.

As the user attempts to accomplish some transaction in Registration (*regent*), a message appears such as "You do not have permission to .... program at .... site."
If the user is supposed to have permissions to accomplish transactions in Registration, a User/Group Permissions Table entry must be present and completed properly. This table can be accessed through the User/Group Permissions option on the Table Maintenance: Registrar (R-Z) Menu. Add or update the table as necessary, either for the user's ID or group ID, which will permit the user to perform required duties. Refer to the Registrar Tables and Table Reports document for instructions on completing this table. If this does not correct the problem, run dbadmin to check on additional permission issues for the user.

The number of students on a class list is different from the registered count on the Section record for the same course.

Example: The class list has 25 students in ENG101, section A, but upon entering the Section record for this course, the registered count shows 23.

Note: A similar condition could exist for wait listed students where a class list shows different values from the Wait List count on the Section record. In general, the class lists are correct and the counters on the Section record are incorrect. To determine whether there are any incorrect section counters, run the Audit Registration Totals option on the Registrar: Session Processing Menu. This process evaluates the Course Work records for all courses, both for registered and wait list counts, updates the Section records if necessary, and produces a report for all courses that had differences between section and actual counts. The report lists both the Section Registered and Section Waiting counts, both of which reflect the counts on the Section record before any corrections. Also listed are the Actual Registered and Actual Waiting, both of which reflect the counts from an evaluation of the Course Work records. The Section record counts are reset to the actual counts. Jenzabar, Inc. recommends that you run this process, commonly referred to as “Registration Audit,” on a scheduled basis to ensure record agreement.

After students have registered for courses, it is learned that some of the courses have incorrect tuition, fee, and/or bill codes, which result in incorrect billing. When the codes are corrected on the Course and/or Section records, it will ensure the proper assignment of the codes on the Registration records for any students who register after the changes were made. But, how are the Registration records changed for those students who registered before the changes?

Run the Update Registration Record option on the Registration: Course/Class Schedule Menu after reading the comments that explain specifically what will occur if the process is completed. Enter ctrl-w (help) after selecting the Update Registration Record menu option to access these comments. If billing does not reflect the correct totals, it may be necessary to analyze individual registration records to determine whether they contain the proper tuition, fee, and/or bill codes.

A class list is run for a course that has registered students; however, no students appear on the list.

After students have registered for the course, someone changed the subsession code on the Section record. Since the students who have already registered have Course Work records with the previous subsession code, the Class List program cannot find matches of students to courses, thereby resulting in the blank lists. There is presently no process in the system to correct such situations. Jenzabar, Inc. recommends that subsession codes, if they are to be used, must be determined before students register and not changed thereafter. However, if they are changed after students register, the institution must ensure that the appropriate Course Work records also are corrected.

INDEX