SUNGARD SCT HIGHER EDUCATION

SCT Banner Technical Student Technical Training Workbook

January 2005
Release 7

Prepared By: SunGard SCT
4 Country View Road
Malvern, Pennsylvania 19355
United States of America

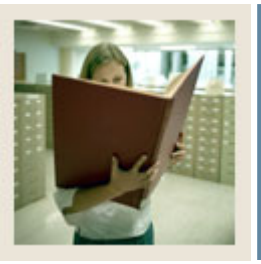


Table of Contents

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Section A: Introduction

Lesson: Overview

Jump to TOC

Workbook goal

This course is designed to provide an overview of the major tables, reports, and processes included in each module of the Student System, as well as providing an introduction to the SCT Banner directory structure and a closer look at some of the database object creation scripts for each of the Student System modules.

This training program will provide the Student technical staff with a basic knowledge of the tables, reports, and processes that make up the SCT Banner Student System.

The training also includes discussion about data conversion, as well as a conversion example exercise.

Intended audience

Programmers, DBA's, and analysts who may teach others about SCT Banner tables and processes will benefit from the training.

Section contents

 Student Technical

Section A: Introduction

Lesson: Introduction

Jump to TOC

Introduction

This course is designed to provide an overview of the major tables, reports, and processes included in each module of the Student System, as well as providing an introduction to the SCT Banner directory structure and a closer look at some of the database object creation scripts for each of the Student System modules.

This training program will provide the Student technical staff with a basic knowledge of the tables, reports, and processes that make up the SCT Banner Student System.

The training also includes discussion about data conversion, as well as a conversion example exercise.

The purpose of this course is to make programmers, analysts, and other technical staff familiar with the tables and other database objects and processes that make up the Student System.

Others areas of interest such as forms modification, advanced database toolkit topics, Object Access, role-based security, etc. are covered completely in other training offered by SCT either at the Ed Center or at the client site.

Section A: Introduction

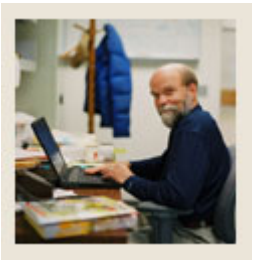
Lesson: Workbook contents

Jump to TOC

Workbook contents

This workbook contains the following sections:

- Section A: Introduction
- Section B: Student Technical Training Overview
- Section C: Course Catalog
- Section D: Referential Integrity
- Section E: General Person
- Section F: Curriculum/Program Rules
- Section G: Recruiting
- Section H: Admissions
- Section I: Overall Forms and Tables
- Section J: Faculty Load
- Section K: Location Management
- Section L: Schedule
- Section M: General Student
- Section N: Accounts Receivable
- Section O: Registration
- Section P: Academic History
- Section Q: Conversion



Section B: Student Technical Training Overview

Lesson: Student Technical Training Overview

Jump to TOC

Intended audience

Programmers, DBA's, and analysts who may teach others about SCT Banner tables and processes will benefit from the training.

Objectives

At the end of this section, you will be able to

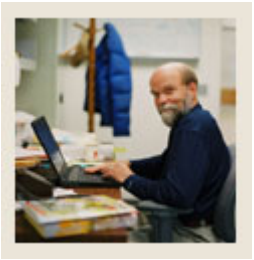
- Describe many of the basic components of the SCT Banner Student System.

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section contents



Section B: Student Technical Training Overview

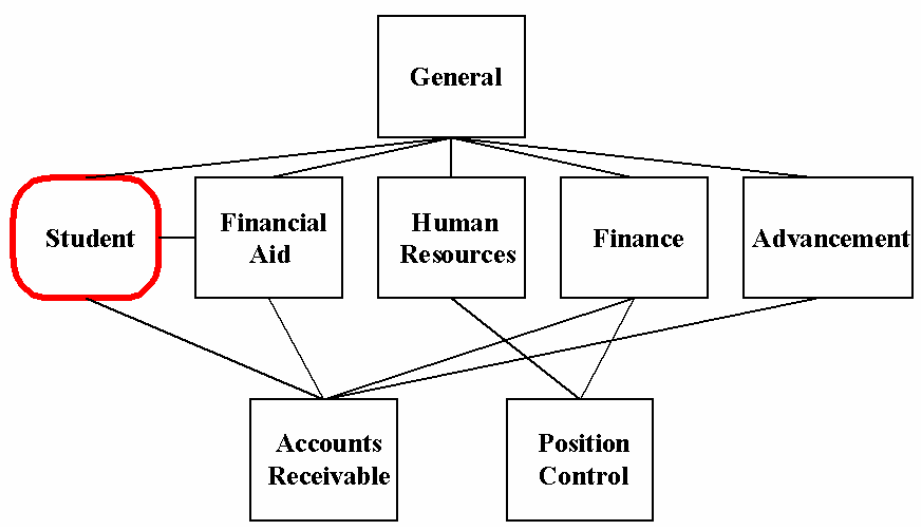
Lesson: The Student System

Jump to TOC

Description

The Student System interacts with Finance through General (GURFEED), but also can be considered to have a direct connection to Finance through Accounts Receivable in the Student System.

Diagram



Student and Advancement

Student to Advancement Interface (APPSTDI)

- Adds records that define individuals as constituents, as well as, updates information on existing constituents

Shared information across SCT Banner

- Identification, Person & Address Information

Information pulled from Student into Advancement

- Academic information is pulled from Admissions, Academic History & Registration
- Student Cooperative information may also be retrieved for employment history
- Student activities will also be retrieved

Student and Finance

Accounts Receivable connects these Systems.

Charges can be posted to an account through the following Student modules:

- Admissions
- Registration
- Location Management
- Academic History
- CAPP

Cashiering sessions would be created for the above transactions

- TGRFEED/FURFEED - processes to move the AR transactions from AR to Finance

  TGRFEED inserts rows into the GURFEED table. FURFEED reads each row and loads the data into the Finance system.

- TSRRFND/FURAPAY - Processes to move AP transactions from AR to Finance

  TSRRFND inserts rows into the GURAPAY table. FURAPAY reads each row and loads data into the Finance system.

Student and Financial Aid

Disbursements

- TSASPAY - Student Payment form. Users can disburse Financial Aid from this form.

  If automatic disbursement flag on TSACTRL is checked then disbursement is done automatically.

  If flag is unchecked the user can disbursement manually. Manual disbursement is performed by entering a 'Y' in the 'Recalculate Financial Aid?' field on the Financial Aid Recalculation window. An AR transaction will be created if disbursement occurred.

- TSASPAY - Student Payment form. Authorized and memoed Financial Aid will display on the student payment form TSASPAY.

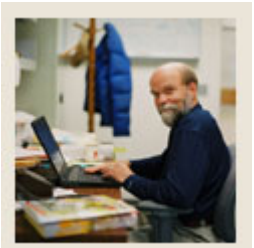
  Authorized Financial Aid can reduce the amount due on this form if the 'Committed/Authorized FA Reduces Amount Due Indicator' on the TSACTRL form is checked. Memos never reduce amount due.

- TSRCBIL - Student Billing Process. Can have authorized FA reduce amount due if flag is set on TSACTRL. Memos can only be printed.

- RPEDISB - allows disbursable aid for a specified term to be credited to a student's account and/or bill in three ways: payments, authorizations and/or memos.

  Students must pass all user-defined edits and any applicable federal requirements.

  Any adjustments made by the Financial Aid office to student awards or due to funds failing disbursement edits may be posted to a student's account and/or bill.



Student and Human Resources

Data entered through either General Person module is shared.

Reports pull information from both the Faculty Load module and the HR system for reports.

- SIRCTAL (Faculty Load Contract Analysis)

  Salary information can be added with a parameter entry.

- PERFACL (Faculty Load Comparison)

  This process identifies where data does not match.

  If the data matches, total and recording of Total Contact Hours and FTE are updated in the HR system.

Field	Table description	Advance-ment	Finance	FinAid	General	HR
STVACCG	Activity Category	A				
STVACTC	Activity	A				
STVACTP	Activity Type	A				
STVACYR	Academic Year			R		
STVADMT	Admission Type			R		
STVAPDC	Admission Application Decision			R		
STVAPST	Admission Application Status			R		
STVASCD	Room Assignment Status			R		
STVASRC	Address Source	A	F	R		H
STVASTD	Academic Standing			R		
STVATYP	Address Type	A	F	R		H
STVBLDG	Building			R	G	
STVCAMP	Campus			R		
STVCIPC	CIP Code					H
STVCLAS	Class			R		
STVCNTY	County	A	F	R		H
STVCOLL	College	A		R	G	
STVDEGC	Degree	A		R		H
STVDEGS	Degree Status			R		
STVDEPT	Department			R	G	
STVDIVS	Division			R		
STVDLEV	Degree Level			R		H
STVDPLM	Diploma Type			R		
STVESTS	Enrollment Status			R		
STVETHN	Ethnic	A	F	R		H
STVETYP	Event Type				G	
STVGEOD	Geographic Region Division	A				
STVGEOR	Geographic Region	A				
STVHAPS	Housing Application Status			R		
STVHLDD	Hold Type			R		

Field	Table description	Advance-ment	Finance	FinAid	General	HR
STVHONR	Institutional Honors	A				
STVINIT	Initials	A				H
STVLANG	Language					H
STVLEAD	Leadership	A				
STVLEVL	Level			R		
STVLGCY	Legacy	A	F	R		H
STVMAJR	Major, Minor, Concentration	A		R		H
STVMDEQ	Medical Equipment					H
STVMEDI	Medical					H
STVMRCD	Meal Rate			R		
STVMRTL	Marital Status	A	F	R		H
STVMSCD	Meal Assignment Status			R		
STVORIG	Originator	A				
STVPENT	Port of Entry			R		H
STVRATE	Student Fee Assessment Code			R		
STVRDEF	Building/Room Attribute			R		
STVRELG	Religion	A	F	R		H
STVRELT	Relation					H
STVRMST	Room Status			R		
STVRRCD	Room Rate			R		
STVRSTS	Course Registration Status			R		H
STVSBGI	Source/Background Institution	A		R		H
STVSITE	Site			R		
STVSPON	International Student Sponsor			R		
STVSTAT	State	A	F	R		H
STVSTST	Student Status			R		
STVTADM	Test Score Administration Type			R		
STVTELE	Telephone Type	A	F	R		H
STVTEPR	Test Purpose			R		

Field	Table description	Advance-ment	Finance	FinAid	General	HR
STVTERM	Term			R		
STVTESC	Test Code			R		
STVTSRC	Admission Test Score Source			R		
STVVETC	Veteran Type			R		
STVVTYP	Visa Type			R		H
TTVBILL	Billing Code			R		
TTVDCAT	Detail Category			R		
TTVPAYT	Payment Type			R		
TTVSRCE	Charge/Payment Detail Source			R		H

SCT Banner views and BANINST1

All SCT Banner views owned by BANINST1:

- General
- General Person
- Finance
- Accounts Receivable
- Position Control
- Payroll
- Student
- Financial Aid
- Advancement
- Security

SCT Banner views

All SCT Banner Views:

- GENERAL
- SATURN
- FIMSMGR
- TAISMGR
- POSNCTL
- PAYROLL
- SATURN
- FAISMGR
- ALUMNI
- BANSECR
- BANINST1

Section B: Student Technical Training Overview

Lesson: Student System Overview

Jump to TOC

Diagram

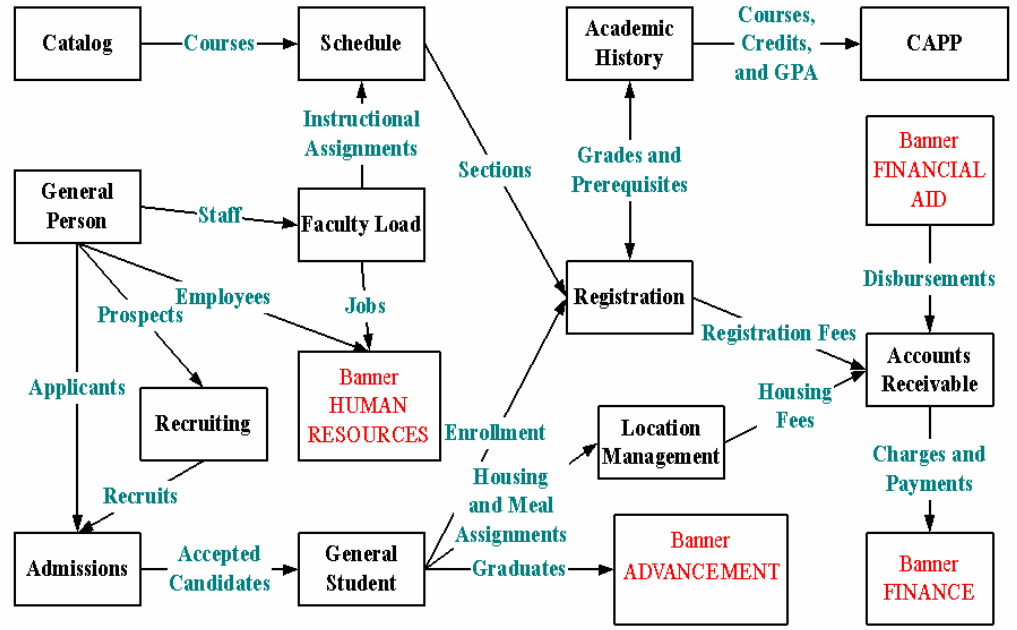


Diagram legend

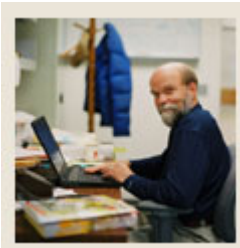
The overview diagram displays how each module interacts with other parts of the Student System.

The diagram flows left-to-right. This does not necessarily reflect the order for conversion and implementation, but does show the logical order of operation in production.

This presentation will follow the logical conversion/implementation order; conversion and implementation order will be discussed along the way.

Color codes (more easily viewable in the course's PowerPoint)

Black = Major Student Modules
Green = Major Content Area of Modules
Red = Other Banner products, which can complement Student



Section B: Student Technical Training Overview

Lesson: Recommended Order for Conversion

Jump to TOC

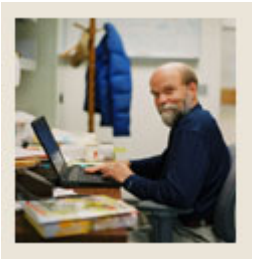
Recommended order

- Catalog
- General Person
- Recruitment
- Admissions
- Location Management & Housing
- Schedule
- Faculty Load
- General Student
- Accounts Receivable
- Registration
- Academic History
- Curriculum, Advising & Program Planning (CAPP)

Course overview

This course will take a forms approach, looking first at the major forms in each module, then at the underlying tables that are required for conversion. We will also examine the validation tables required for conversion. Next, we will look at the tables in the database using SQL*Plus and learn their structure and content, and the relationships among tables in a module. We will look carefully at the delivered reports and processes, looking at the code that produces them. Finally, we will look in the Banner Student directories at some of the scripts that produce database objects such as views, functions and procedures.

Your institution may not choose to use every form or every field on any form, but this class will cover the required forms, tables and fields, reports and processes you will need for conversion in order to use the Student System effectively.

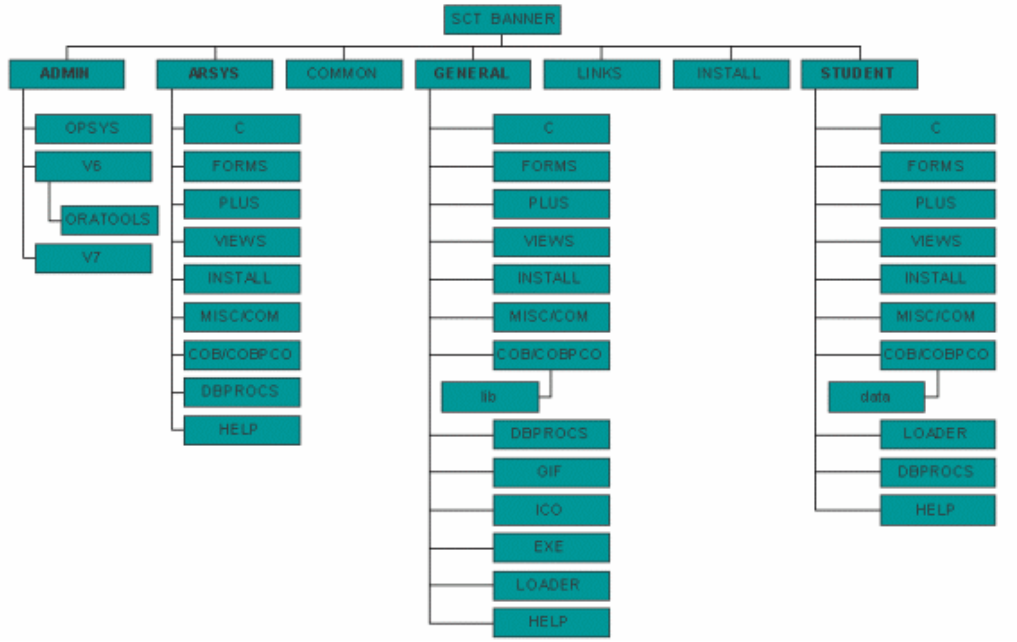


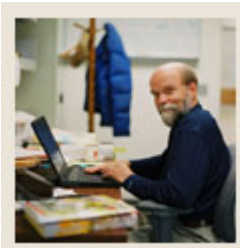
Section B: Student Technical Training Overview

Lesson: SCT Banner Student Directories

Jump to TOC

Diagram





Section B: Student Technical Training Overview

Lesson: SCT Banner Student Directories (Continued)

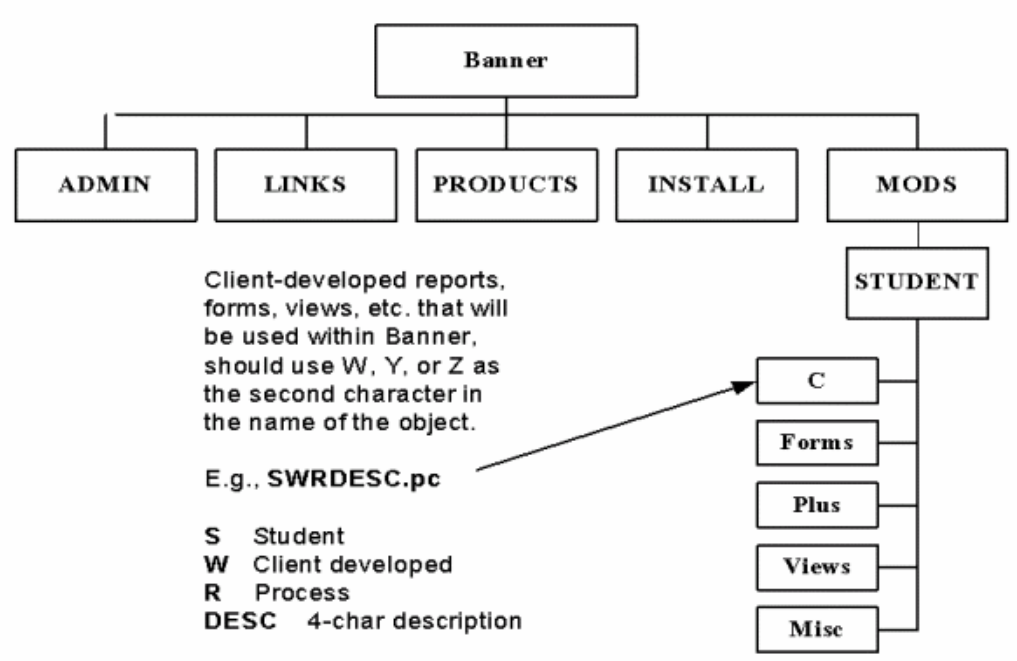
Jump to TOC

Directories

We will be looking at these directories in your operating system in detail for each module as we move through the class. We will look closely at the C, DBPROCS and VIEWS directories.

- C — Pro*C and C source files
- COB — Pro*COBOL files (UNIX only)
- COBPCO — Pro*COBOL files (VAX/VMS only)
- COM — DCL command files (VAX/VMS only)
- DATA — Course request and scheduler input (directory under COB-UNIX only, or COBPCO, VMS only)
- FORMS — ORACLE*FORMS .fmb, .fmx, .pll, and .lib files
- INSTALL — .SCTDMP file used during initial install
- LOADER — ORACLE SQL*LOADER -- ASCII to Oracle Tables
- MISC — Shell scripts (UNIX only; COM -- DCL command files for VAX/VMS)
- PLUS — SQL*PLUS scripts
- VIEWS — SQL*PLUS scripts to recreate views
- DBPROCS — SQL*PLUS scripts to recreate database procedures, packages, functions, triggers

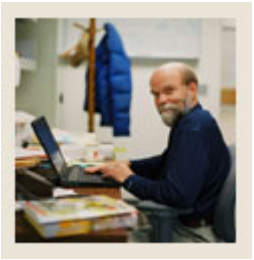
Diagram



MODS directory

The MODS directory is a separate directory for client modifications. Within the MODS directory, you would find another STUDENT directory, etc.

Examples of modifications found here include forms, modifications to C programs, client created views, functions, etc.



Section B: Student Technical Training Overview

Lesson: Review of Database Tools

Jump to TOC

Brief review

This brief review of database tools will cover the data dictionary, which we will use extensively throughout the training, and a brief review of the principle of referential integrity, an understanding of which is essential to a successful conversion.

There are several good ways to see the contents of the database and learn the structure and content of the tables.

Data Dictionary

Built into Oracle is the Data Dictionary, a series of views that give detailed information about the database. This is covered in more detail in the DBA Toolkit class.

GURPDED process from Job Submission

Banner has provided a form interface to the Data Dictionary with the GURPDED utility program, run through job submission.

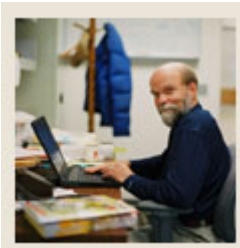
Technical Addendum

Also available is the Technical Addendum, which is a large, all-inclusive, hard-copy version of the output of the GURPDED process.

Even if you have a copy of the Technical Addendum, you should know how to gather table information directly from the database. It is better to use the Data Dictionary or GURPDED to create a customized Technical Addendum for your institution.

Third-party navigator

Finally, your institution may have a third-party navigator, which will be the tool that you will use instead of the data dictionary. These third-party products are NOT SUPPORTED BY SCT.



Section B: Student Technical Training Overview

Lesson: The Data Dictionary

Jump to TOC

Description

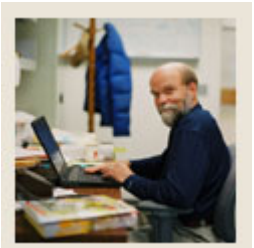
The Data Dictionary is a set of tables and views that are used as a read-only reference about the database.

The Data Dictionary stores information about both the logical and physical structure of the database.

Types of Data Dictionary views

- USER_xxxxx -- shows objects and events owned by user
- ALL_xxxxx -- shows all objects and events to which user has access
- DBA_xxxxx -- restricted; assigned only to those with DBA role
- ALL_TABLES - descriptions of tables
- ALL_COL_COMMENTS - comments on columns of accessible tables
- ALL_TAB_COLUMNS - lists of columns of all tables
- ALL_TAB_COMMENTS - comments on tables
- ALL_USERS - information on all users in database
- ALL_VIEWS - lists text of views accessible to user
- ALL_INDEXES - descriptions of indexes
- ALL_IND_COLUMNS - lists columns of the indexes

For a complete reference, refer to your Oracle documentation.



Section B: Student Technical Training Overview

Lesson: GURPDED Procedure

Jump to TOC

Description

This procedure is run from the Process Control Submission Form (GJAPCTL)

Parameters

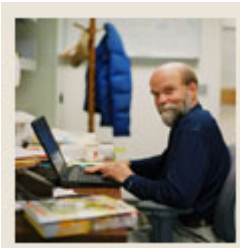
Enter Parameters:

- Table Name
- Table Owner

Output

Output = Technical Addendum

- To DATABASE
- View or Print from GJIREVO

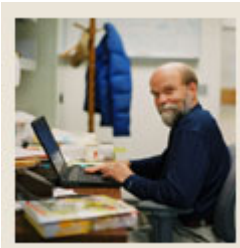


Exercise 1

Find out what indexes exist for the course catalog table (SCBCRSE).

Exercise 2

List the columns in the SCBCRSE indexes that you discovered.



Exercise 1

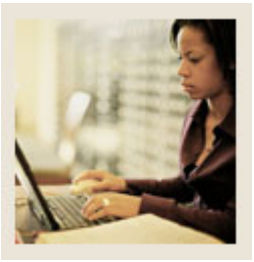
Find out what indexes exist for the course catalog table (SCBCRSE).

```
      desc all_indexes;
   select index_name
     from all_indexes
    where table_name = 'SCBCRSE';

        Result:     scbcrse_key_index
```

Exercise 2

List the columns in the SCBCRSE indexes that you discovered.

```
       select column_name
         from all_ind_columns
        where table_name = 'SCBCRSE'
          and index_name = 'SCBCRSE_KEY_INDEX';

  Result: scbcrse_subj_code
          scbcrse_crse_numb
          scbcrse_eff_term
```



Section C: Course Catalog

Lesson: Overview

Jump to TOC

Intended audience

Programmers, DBA's, and analysts who may teach others about SCT Banner tables and processes will benefit from the training.

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Course Catalog module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

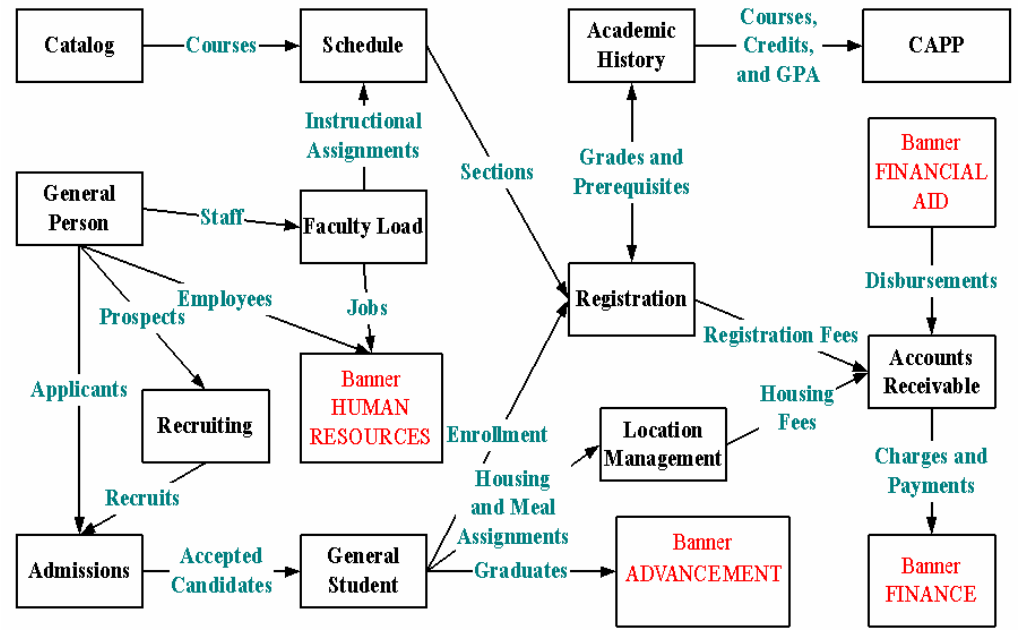
Section contents

Section C: Course Catalog

Lesson: Student System Overview

Jump to TOC

Diagram



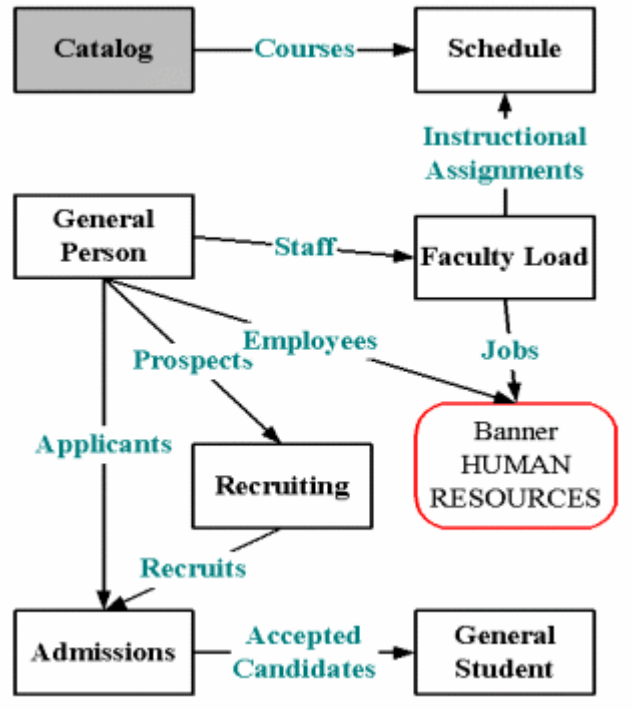
You will begin with the Catalog module, one of the first to be converted. (You could also convert General Person first, depending on the needs of other systems.)

Section C: Course Catalog

Lesson: Course Catalog Module

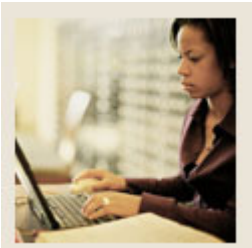
Jump to TOC

Diagram



Legend

This diagram shows a part of the large overview diagram.
The Recruiting Module is OPTIONAL.
The GREEN boxes on the diagram indicate the primary content of the module.



Section C: Course Catalog

Lesson: Course Catalog Module (Continued)

Jump to TOC

Implementation

SCT Banner must be implemented in a logical order.

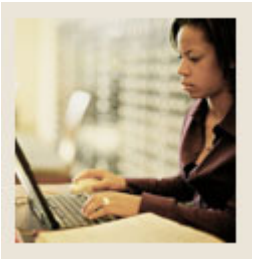
We have organized this training presentation in a conversion order model. It would be recommended to convert either Catalog or General Person first, depending on the needs of other systems.

This order follows a general implementation order that allows clients to build the necessary rules and validation tables in the order that they are needed for successful implementation.

Objectives

Examine/Review:

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Review Referential Integrity
- Conversion of Data

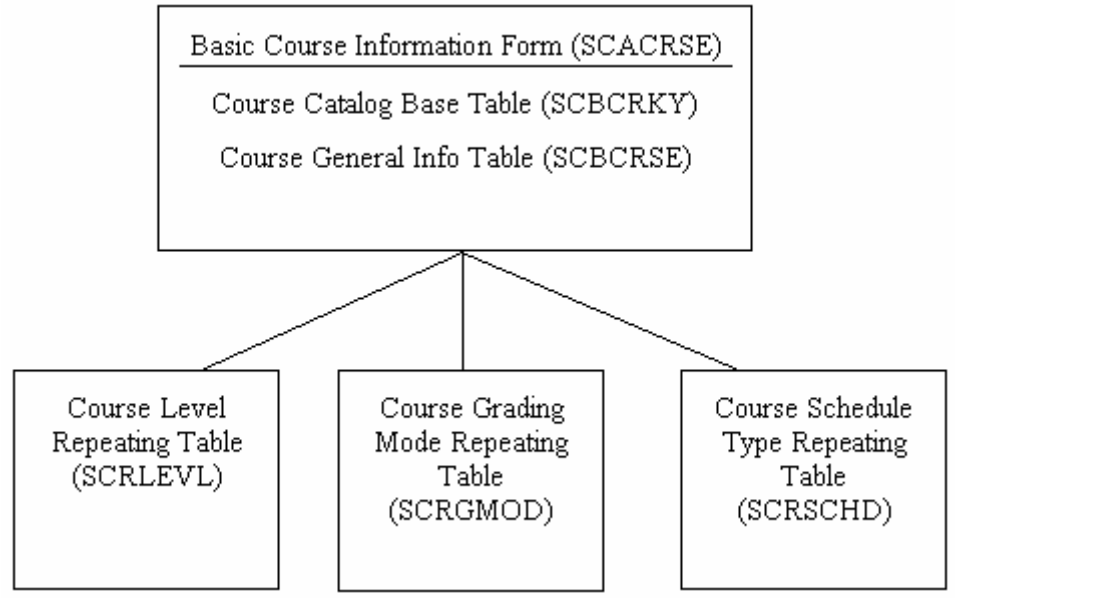


Section C: Course Catalog

Lesson: Course Catalog

Jump to TOC

Diagram



Description

The Course Catalog module holds all general course information. It is used as the foundation for each term's schedule, but does not hold schedule information by term.

You cannot create a schedule for a term (and, therefore, students cannot register for a term) unless all courses are built in Catalog first.

Catalog controls the courses and the TYPE of courses (i.e. labs) that may be included in the Schedule.

Forms and tables

SCACRSE is the main form.

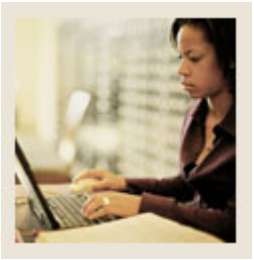
SCBCRSE, SCBCRKY, SCRLEVL, SCRGMOD, SCRSCHD are the main tables. All of these tables are required. (*RED asterisked (*)* text in the diagrams of this manual indicates required elements.)

Major Forms:

- SCABASE
- SCACRSE

Major Tables:

- SCBCRSE
- SCBCRKY
- SCRLEVL
- SCRGMOD
- SCRSCHD



Section C: Course Catalog

Lesson: Naming conventions

Jump to TOC

Naming conventions

All SCT Banner objects adhere to naming conventions.

Objects include forms, tables, processes, etc.

For more information, refer to Chapter 1 of the *Student Technical Reference Manual*.

Form, process and table naming

The names of all SCT Banner forms (except menu forms), reports, processes and tables are seven characters long, with each character representing a position location.

Example:

Character:	S C A C R S E
Position Location:	1 2 3 4 5 6 7

Position 1

Position 1 identifies the primary System that owns the form, report, process or table.

Note: The letters W, Y and Z are reserved for client applications which coexist with SCT Banner.

Letter	System	Letter	System
A	Advancement	**O**	Customer Contact
B	Property Tax	**P**	HR / Payroll / Personnel
C	Courts	**Q**	Electronic Work Queue
D	Cash Drawer	**R**	Financial Aid
F	Finance	**S**	Student
G	General	**T**	Accounts Receivable
I	Information Access	**U**	Utilities
K	Work Management	**V**	Voice Response
L	Occupational Tax/License	**X**	Records Indexing
N	Position Control		

Position 2

Position 2 identifies the module that owns the form, report, process or table. The letter assignments will vary by System.

Position 3

Position 3 identifies the type of form, report, process or table.

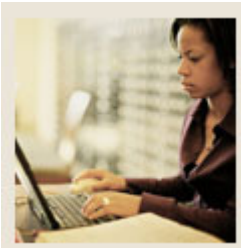
Letter	Type	Letter	Type
A	Application form	**Q**	Query form
B	Base table/Batch COBOL process	**R**	Rule table, repeating table, or report/process
I	Inquiry form	**T**	General maintenance temporary table
O	Online COBOL process	**V**	Validation form/table or view

Positions 4 – 7

The remaining positions identify a unique four-character name for the form, report, process or table.

Example:

- SCACRSE: **S** Student
  C Catalog
  A Application
  CRSE Course Information



Section C: Course Catalog

Lesson: Major Validation Tables/Forms

Jump to TOC

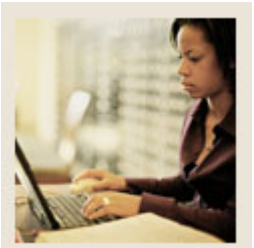
Validation forms

- Subject Code Validation Form (STVSUBJ)
- Term Code Validation Form (STVTERM)
- College Code Validation Form (STVCOLL)
- Course Status Code Validation Form (STVCSTA)
- Level Code Validation Form (STVLEVL)
- Grading Mode Code Validation Form (STVGMOD)
- Schedule Type Code Validation Form (STVSCHD)

Validation tables

Critical to each module are the related validation tables. Validation tables contain the codes that are acceptable to use in a particular field. If a code is not in the validation table, it cannot be used as data in that field.

Validation tables have a parent/child relationship with records. (The principle of referential integrity will be covered more thoroughly in a subsequent lesson.) Validation tables (parents) must be populated with correct codes before converting data which will populate the child records in the tables.



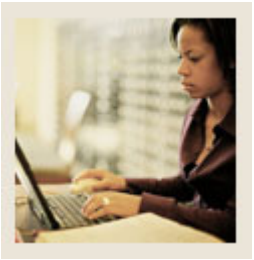
Section C: Course Catalog

Lesson: Basic Course Information Form (SCACRSE)

Jump to TOC

Components

- Key Block
- From and To Terms
- Fields Related to AR
- LOV Fields
- Level, Grade Mode, Schedule Type



Section C: Course Catalog

Lesson: SQL*Plus

Jump to TOC

Questions

- How are the SCBCRSE and the SCBCRKY tables related?
- What data elements are required in SCBCRSE, SCBCRKY and SCRLEVL?
- How are level, grading mode, schedule types connected to a course?

```
SQL> desc scrlevl
SQL> desc scrgmod
SQL> desc scrschd
```

Common fields

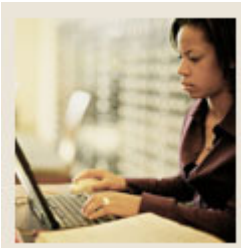
Look at the fields that each has in common with SCBCRSE and SCBCRKY.

Note: Validation tables -- STV + 4-character identifier + _code

SCBCRSE and SCBCRKY

```
SCBCRSE
 Name                          Null?              Type
 ----------------------------  --------           ----
SCBCRSE_SUBJ_CODE              NOT NULL           VARCHAR2(4)
SCBCRSE_CRSE_NUMB              NOT NULL           VARCHAR2(5)
SCBCRSE_EFF_TERM                                  VARCHAR2(6)
…
SCBCRKY_SUBJ_CODE              NOT NULL           VARCHAR2(4)
SCBCRKY_CRSE_NUMB              NOT NULL           VARCHAR2(5)
SCBCRKY_TERM_CODE_START        NOT NULL           VARCHAR2(6)
SCBCRKY_TERM_CODE_END          NOT NULL           VARCHAR2(6)
```

Note: A relationship exists between subj_code and crse_numb. If scbcrse_eff_term is populated, then scbcrky_term_code_start must be >= scbcrse_eff_term.



Section C: Course Catalog

Lesson: Conversion Issues

Jump to TOC

Questions

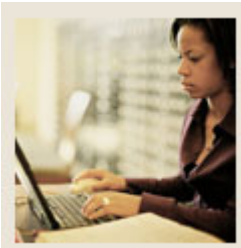
- Will Course Catalog data be converted or entered manually by the users?
- What course catalog data do you have in your legacy system?
- How do you determine where to put it in Banner?

Legacy data

If there is legacy data which does not have an obvious place in the required tables, look at other catalog module tables. To see all tables in the Course Catalog module:

```
select table_name,comments
  from all_tab_comments
 where table_name like 'SC%';
```

- College/Department Table (SCBCDEP)
- Course Catalog Base Table (SCBCRKY)
- Course General Information Base Table (SCBCRSE)
- Supplemental Course Data Table (SCBSUPP)
- Course Attribute Repeating Table (SCRATTR)
- College/Department Text Table (SCRCDTX)
- Course Corequisites Repeating Table (SCRCORQ)
- Equivalent Course Repeating Table (SCREQIV)
- Course Fees Repeating Table (SCRFEES)
- ...etc.



Section C: Course Catalog

Lesson: Reports/Processes

Jump to TOC

SCRBULT

- SCRBULT -- Bulletin Report
- Prints catalog of courses
- Parameter: Academic Year (see STVACYR)
- C program
- Run via Job Submission

Actions

- Go to SCT Banner Job Submission to run this report.
- Send the report output to GJIREVO.
- Type "DATABASE" in the printer field to view the report output within SCT Banner.
- If you are unfamiliar with Job Submission, it is covered in more detail in the General Technical training course.

Section C: Course Catalog

Lesson: Self Check - Course Catalog Exercises

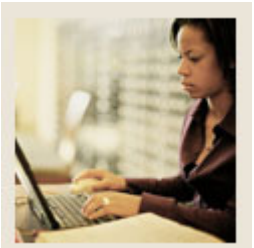
Jump to TOC

Exercise 1

Get the following information about any two of the Course Catalog module tables:

- Table Owner
- Table Name
- Column Name
- Data Type
- Null/Not Null Column

Hint: You will use the data dictionary view “all_tab_columns”

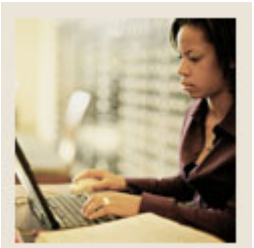


Exercise 2

Get the following information from the course catalog module about all courses with a subject code of 'ENGL':

- subject code
- course number
- course title
- effective term
- start term
- end term

Hint: You will need to use SCBCRSE and one other table.



Exercise 3

Write a select statement that would produce a catalog report which includes the following (no formatting necessary):

- subject code
- course number
- course title
- effective term
- start term
- course level
- grade mode

Hint: You will need to use SCBCRSE and 3 other tables.

Section C: Course Catalog

Lesson: Self Check – Course Catalog Exercises – Answer Key

Jump to TOC

Exercise 1

Get the following information about any two of the Course Catalog module tables:

- Table Owner
- Table Name
- Column Name
- Data Type
- Null/Not Null Column

Hint: You will use the data dictionary view "all_tab_columns"

```
Step 1:
SQL> desc all_tab_columns
Name                 Null?           Type
 OWNER               NOT NULL  VARCHAR2(30)
 TABLE_NAME          NOT NULL  VARCHAR2(30)
 COLUMN_NAME         NOT NULL  VARCHAR2(30)
 DATA_TYPE                     VARCHAR2(9)
 DATA_LENGTH         NOT NULL  NUMBER
 DATA_PRECISION                NUMBER
 DATA_SCALE                    NUMBER
 NULLABLE                      VARCHAR2(1)

Step 2:
SQL>  SELECT owner, table_name, column_name,
             data_type, nullable
        FROM all_tab_columns
       WHERE table_name like 'SC%';
```

Exercise 2

Get the following information from the course catalog module about all courses with a subject code of 'ENGL':

- subject code
- course number
- course title
- effective term
- start term
- end term

Hint: You will need to use SCBCRSE and one other table.

Step 1:

```
SQL> desc scbcrse
SQL> desc scbcrky
```

Step 2:

```
SQL> SELECT scbcrse_subj_code,
  2   scbcrse_crse_numb,
  3   scbcrse_title,
  4   scbcrse_eff_term,
  5   scbcrky_term_code_start,
  6   scbcrky_term_code_end
  7  FROM  scbcrky,
  8   scbcrse
  9  WHERE  scbcrse_subj_code = scbcrky_subj_code
 10  AND   scbcrse_crse_numb = scbcrky_crse_numb
 11  AND  scbcrse_subj_code = 'ENGL'
 12  /
```


Exercise 2, continued

```
SCBC SCBCR SCBCRSE_TITLE                SCBCRS SCBCRK SCBCRK
---- ----- ------------------------------ ------ ------ ------
ENGL 1005  Literature & Composition I     199510 199510 999999
ENGL 1005  Literature & Composition I     199610 199510 999999
ENGL 1005  Literature & Composition I     199620 199510 999999
ENGL 1006  Literature & Composition II    199510 199510 999999
ENGL 101   English Composition            198710 198710 999999
ENGL 101   English Composition            199010 198710 999999
ENGL 101A  Computer Literacy              198710 198710 999999
ENGL 103   20th Century American Lit      199510 199510 999999
ENGL 1050  The Literary Experience        199510 199510 999999
ENGL 107   World Lit                      199010 199010 999999
ENGL 108   World Lit                      199010 199010 999999
ENGL 109   World Lit                      199010 199010 999999
ENGL 1201  Survey of American Lit I       199510 199510 999999
ENGL 201   Topics in English              198710 198710 999999
ENGL 310   African American Prose         199010 199010 999999
ENGL 311   African-American Poetry        199010 199010 999999
ENGL 312   African American Drama         199010 199010 999999
ENGL 408   Topics in English Lit          199520 199520 999999
ENGL 410   Topics in American Lit         199520 199520 999999

19 rows selected.
```

Exercise 3

Write a select statement that would produce a catalog report which includes the following (no formatting necessary):

- subject code
- course number
- course title
- effective term
- start term
- course level
- grade mode

Hint: You will need to use SCBCRSE and 3 other tables.

```
SQL> SELECT scbcrse_subj_code,
  2   scbcrse_crse_numb,
  3   scbcrse_title,
  4   scbcrse_eff_term,
  5   scbcrky_term_code_start,
  6   scrlevl_levl_code,
  7   scrgmod_gmod_code
  8  FROM scrgmod, scrlevl, scbcrky, scbcrse
  9  WHERE scbcrse_subj_code = scrlevl_subj_code
 10  AND scbcrse_crse_numb = scrlevl_crse_numb
 11  AND scbcrse_eff_term =  scrlevl_eff_term
 12  AND scbcrse_subj_code = scrgmod_subj_code
 13  AND scbcrse_crse_numb = scrgmod_crse_numb
 14  AND     scbcrse_eff_term = scrgmod_eff_term
 15  AND scbcrse_subj_code = scbcrky_subj_code
 16  AND scbcrse_crse_numb = scbcrky_crse_numb
 17  /
```

Section D: Referential Integrity

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe referential integrity concepts and how they are implemented in SCT Banner

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

Section D: Referential Integrity

Lesson: Referential Integrity

Jump to TOC

Importance

Referential integrity is reviewed here because of the importance of understanding it in relation to conversion. All required validation tables needed for a module must be populated before populating other data tables within each module.

Ideally, those attending this class should have taken the DBA Toolkit class or have a basic familiarity with SQL.

Types of Data Integrity

- Nulls
- Unique Column Values
- Primary Key Values
- Referential Integrity

* Source: *Oracle 7 Server Concepts*

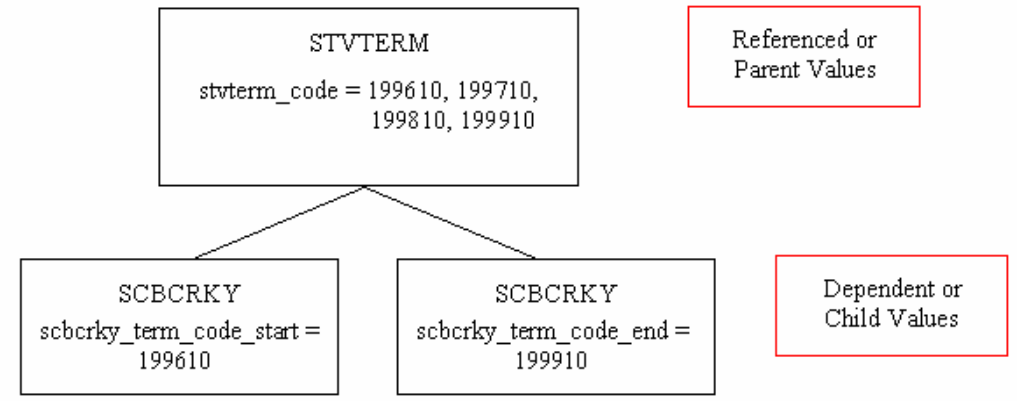
In any discussion of implementation of SCT Banner and conversion to SCT Banner, we must consider referential integrity, which has a direct influence over the order of conversion and implementation.

Section D: Referential Integrity

Lesson: Referential Integrity Illustrated

Jump to TOC

Diagram



Parent and child

STVTERM is the parent; SCBCRKY is the child. Term values cannot appear in SCBCRKY fields unless they are in STVTERM.

A rule defined on a column (or set of columns) in one table that allows the insert or update of a row only if the value for the column or set of columns (the dependent or child value) matches a value in a column of a related table (the referenced or parent value).*

* Source: *Oracle 7 Server Concepts*

Section D: Referential Integrity

Lesson: Referential Integrity Key Types

Jump to TOC

Key types

Referential Integrity relies on two types of keys:

- Primary Keys
- Foreign Keys

These keys are implemented as *constraints* which enforce unique, non-null keys.

Two Integrity Rules

- Entity integrity:
- No attribute participating in the primary key of a base relation is allowed to accept null values.

Referential integrity:

- If a base relation includes a foreign key matching the primary key of some other base relation, then every value of the foreign key in the 1st base relation must either be equal to the value of the primary key in some tuple (row) of the other base relation ***OR*** be wholly null (i.e., each attribute value participating in that foreign key value must be null). The two base tables are not necessarily distinct.
- The basic intent of this rule is that if some tuple t2 references some tuple t1, then t1 must exist. A given foreign key value must have a matching primary key value somewhere in the referenced relation if that foreign key value is **non null**.

Section D: Referential Integrity

Lesson: Primary Key Constraints

Jump to TOC

Constraints

- **Primary Key** - special case of a candidate key -- unique identifier -- absolutely fundamental to the operation of the overall relational model -- to enforce unique, non-null keys
- **Uniqueness** - At any given time, no two distinct rows or records of a relation have the same value for any given attribute
- **Minimality** - None of the attributes can be discarded from the set of attributes without destroying the uniqueness property

Primary key

Every relation has at least one candidate key, because at least the combination of all of its attributes has the uniqueness property. One candidate key is designated at the primary key. The remaining candidate keys (if any) are called alternate keys.

(Example: SPRIDEN: PIDM and LAST_NAME -- ***If*** they were each "unique" then the relation has two candidates, PIDM & LAST_NAME. PIDM could be chosen as the primary key; LAST_NAME then becomes an alternate key.)

Note: PIDM & LAST_NAME are NOT unique.

SCT Banner naming convention

PK_ + primary key table name

Example: PK_STVTERM is defined by:

```
alter table STVTERM
add constraint PK_STVTERM
Primary key (stvterm_code)
```

For Definition of Primary Key Integrity Constraints: Refer to p. 7-10 in ***Oracle 7 Server Concepts***

Section D: Referential Integrity

Lesson: Primary Key Constraints (Continued)

Jump to TOC

Data dictionary views

Oracle 7 Server Reference contains a listing of all data dictionary views, such as all_constraints, etc. The Data Dictionary views all_constraints and all_cons_columns as a way of getting detailed information about all constraints.

```
desc user_constraints  and all_constraints
 select *
   from all_constraints
  where table_name = 'STVTERM';
```

ALTER statements

To see "alter" statements that create primary key constraints for a table (ex: SCBCRKY):

In SQL*PLUS, run GURRDDL script for SCBCRKY:

```
alter table SCBCRKY
add constraint PK_SCBCRKY
Primary key (scbcrky_subj_code, scbcrky_crse_numb)
```


Section D: Referential Integrity

Lesson: Foreign Key Constraints

Jump to TOC

Foreign key

A foreign key is an attribute (or attribute combination) in one relation whose values are required to match those of the primary key of some other relation, to ensure that children are not updated/inserted if parent rows do not exist and to prevent the deletion of parents if children do exist.

Foreign-to-primary-key matches represent *references* from one relation to another; they are the "glue" that holds the database together.

Examples:

- STVTERM Primary Key: STVTERM_CODE
- SGBSTDN Foreign Key: TERM_CODE_EFF
  (which is part of the primary key of SGBSTDN)

GURRDDL

If you run gurrddl you can see the alter statements defining the foreign keys related to STVTERM. This is for SGBSTDN_TERM_CODE_EFF:

```
ALTER TABLE SATURN.SGBSTDN
ADD CONSTRAINT FK3_SGBSTDN_INV_STVTERM_CODE FOREIGN KEY
  (SGBSTDN_TERM_CODE_EFF)
  REFERENCES SATURN.STVTERM
  (STVTERM_CODE) ;
```

Definitions

- **Foreign Key:** The column or set of columns included in the definition of the referential integrity constraint that reference a *referenced key**.
- **Referenced Key:** The unique or primary key of the same or different table that is referenced by a foreign key*.
- **Dependent or Child Table***:* The table that includes the foreign key and is therefore dependent on the values present in the referenced unique or primary key.
- **Referenced or Parent Table***:* The table that is referenced by the child table's foreign key and which determines whether specific inserts or updates are allowed in the child table.

* Source: *Oracle 7 Server Concepts*

Naming convention

FKn_ + foreign table_ + INV_ + Primary table_ + CODE
where

- "n" is a one-up number
- foreign table is the table that contains the constraint
- primary table is the table which contains the primary or referenced key

Example: FK1_SCBCRKY_INV_STVTERM_CODE

Note: The underscore character (_) separates each element of the name.

Section D: Referential Integrity

Lesson: Creating Foreign Key Constraints

Jump to TOC

Alter Statement

FKn_ + foreign table_ + INV_ + Primary table_ + CODE

```
FK1_SCBCRKY_INV_STVTERM_CODE
   alter table SCBCRKY
     add constraint FK1_SCBCRKY_INV_STVTERM_CODE
     foreign key (SCBCRKY_TERM_CODE_START)
     references SATURN.STVTERM (STVTERM_CODE);
```

Before you can enter a term code in SCBCRKY it must exist in STVTERM.
STVTERM codes cannot be deleted if they exist in other tables.

Example 1

select constraint_name
 from all_constraints
where table_name = 'SCBCRKY'

```
   SYS_C002703  NOT NULL
   SYS_C002704  NOT NULL
   SYS_C002705  NOT NULL
   SYS_C002706  NOT NULL
   SYS_C002707  NOT NULL
   PK_SCBCRKY   PRIMARY KEY CONSTRAINT
   FK1_SCBCRKY_INV_STVSUBJ_CODE  Foreign Key for Subject Code
   FK1_SCBCRKY_INV_STVTERM_CODE  Foreign Key for term_code_start
   FK2_SCBCRKY_INV_STVTERM_CODE  Foreign Key for term_code_end
```


Example 2

```
SQL> SELECT  constraint_name,
             constraint_type,
             status
    FROM     all_constraints
   WHERE     table_name =  SCBCRSE';
```

```
CONSTRAINT_NAME                CON.   TYPE   STATUS
-----------------------------------------------
FK1_SCBCRSE_INV_STVAPRV_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVCIPC_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVCOLL_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVCSTA_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVDEPT_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVDIVS_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVPWAV_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVREPS_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVSUBJ_CODE          R   ENABLED
FK1_SCBCRSE_INV_STVTERM_CODE          R   ENABLED
```

Type 'R' = Referential Integrity Constraint;
This illustrates the connection to the Validation tables.

Section D: Referential Integrity

Lesson: Validation Tables/Codes

Jump to TOC

Example

POSITIONS 9th 14th
S C B C R S E _ S U B J _ C O D E

TABLE NAME Validation
Description
|
\/
S T V S U B J

Section D: Referential Integrity

Lesson: Referential Integrity: Summary

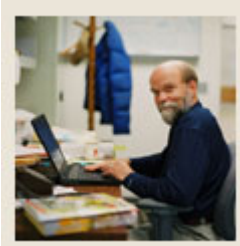
Jump to TOC

Summary

- Enforces unique, non-null columns
- Establishes relationship between parent and child tables
- Parent table row has the Primary Key constraint
- Child table row has the Foreign Key constraint
- Parent row can not be deleted when a child row exists (the child row must be deleted first)

Examples

- 199101 must exist in STVTERM before inserting a record in SCBCRKY with term_code_eff = 199101
- 199101 cannot be deleted from STVTERM if SCBCRKY record exists with 199101 term_code_eff
- Check constraints: to enforce integrity issues specified by the check condition -- prefix would be “cc”
- Unique constraints: designates a column or a combination of columns as a unique key -- prefix is “uk”



Section E: General Person

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the General Person module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

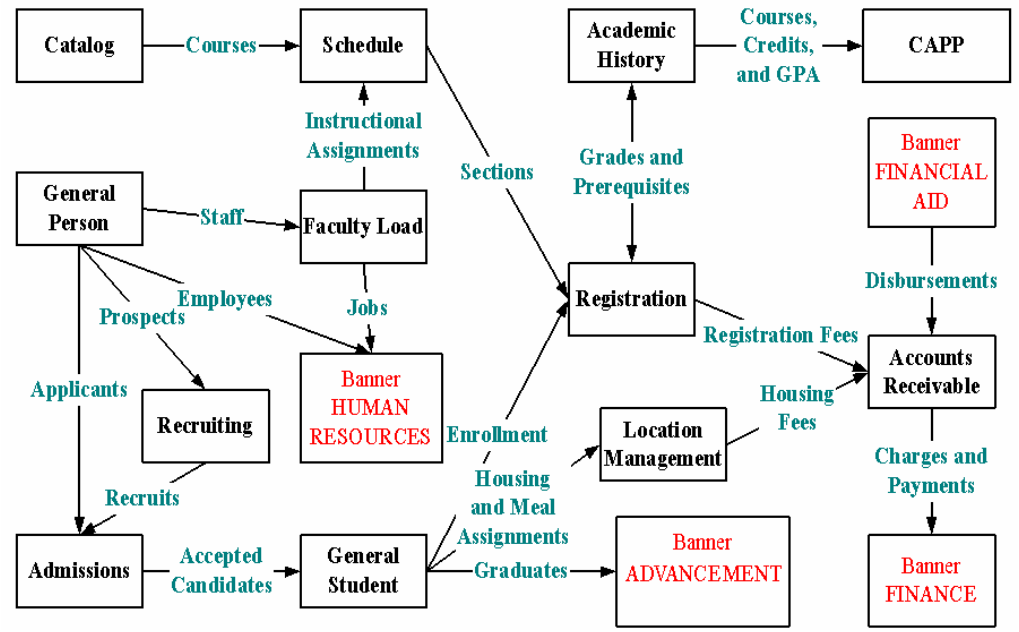
Section Contents

Section E: General Person

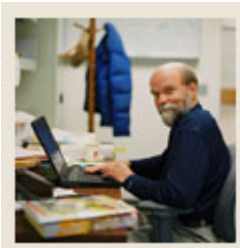
Lesson: Student System overview

Jump to TOC

Diagram



General Person is often the first module to be converted because of its reach into other modules, and into other products as well.

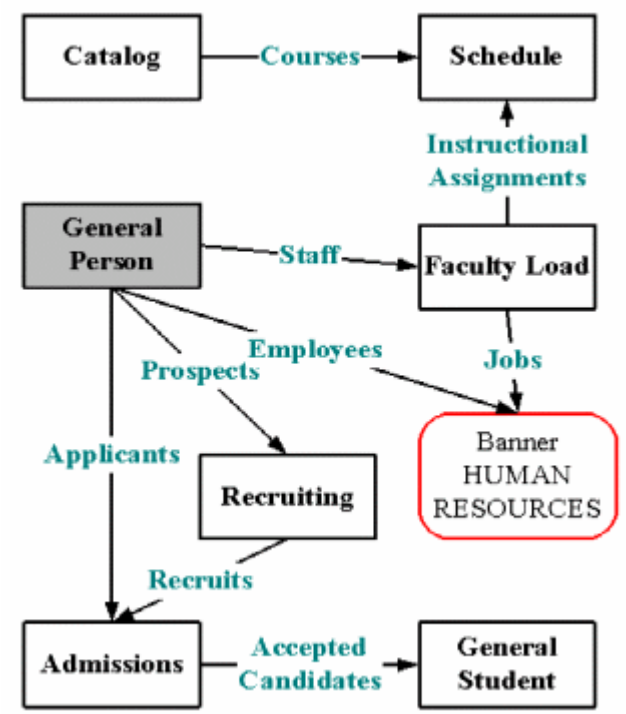


Section E: General Person

Lesson: General Person Module

Jump to TOC

Diagram



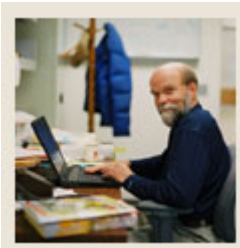
The arrow descriptors on the diagram indicate the primary content of the module.

Overview

Data entry standards are important in all modules, but because General Person allows many opportunities for freeform data entry, it is appropriate to discuss data standards. It is also important to establish institutional data standards before converting legacy data.

Before a person can become a recruit, applicant, student, instructor, advisor, or have an account, the person must first be identified to the system with an identification number and a name.

A person is initially added to the system using the SPAIDEN form, which maintains a person's identification number.



Section E: General Person

Lesson: General Person Module: Objectives

Jump to TOC

Objectives

Examine:

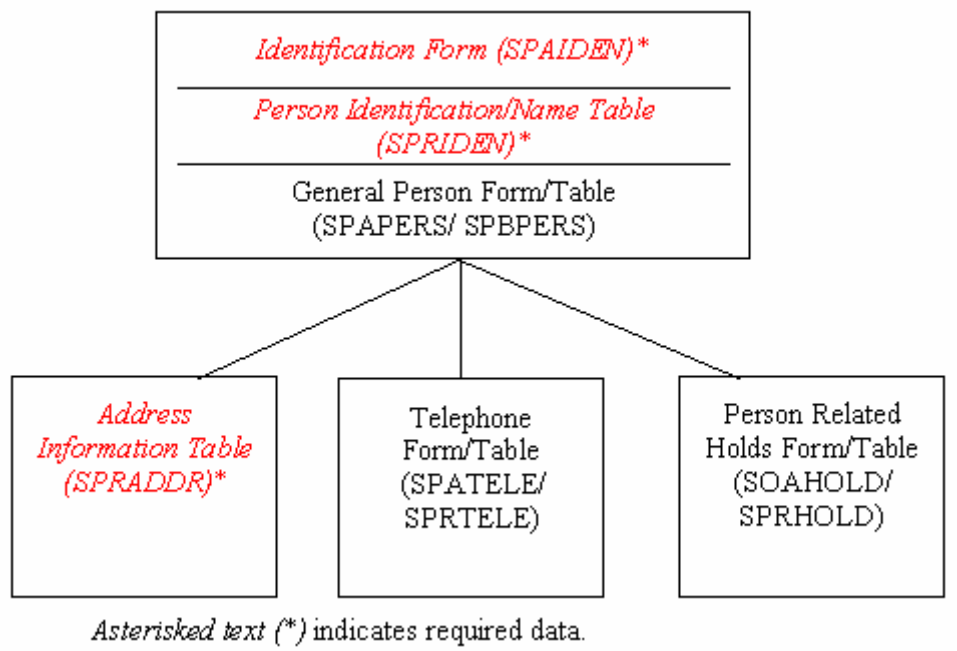
Major & Required Forms and Tables

- SOBSEQN, PIDM

Data standards

- SPRIDEN, SPBPERS indicators
- SPRPDIR process
- Conversion of data
- Stores all biographic and demographic info about an entity in the database.

Diagram



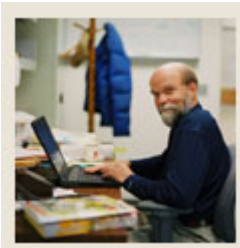
Forms and tables

SPRADDR and SPRTELE data are viewed from SPAIDEN.

Although SPBPERS is not required, it is used as if it is, and it will be discussed in this lesson. The same principle applies to SPRTELE AND SPRHOLD -- most schools use them.

SPBPERS student data is typically maintained by the Registrar's Office for purposes of Federal and State Reporting.

If the institution has the Banner HR product, the HR office may maintain general person data for employees.



Section E: General Person

Lesson: General Person Forms and Tables (Continued)

Jump to TOC

Major forms

- Identification Form (SPAIDEN)
- General Person Form (SPAPERS)
- Telephone Form (SPATELE)
- Hold Information Form (SOAHOLD)

Note: There is no SPAADDR form. Address information data is entered through SPAIDEN.

Major tables

- Identification Table (SPRIDEN)
- General Person Table (SPBPERS)
- Address Information Table (SPRADDR)
- Telephone Table (SPRTELE)
- Person Related Holds Table (SPRHOLD)

Major Validation Tables/Forms

- Address Type Code Validation Form (STVATYP)
- State/Province Code Validation Form (STVSTAT)
- Nation Code Validation Form (STVNATN)
- ZIP/Postal Code Validation Form (GTVZIPC)
- Telephone Type Validation Form (STVTELE)
- Hold Type Code Validation Form (STVHLDD)

These tables have to be populated before converting other general person tables.

Section E: General Person

Lesson: PIDM and SOBSEQN

Jump to TOC

PIDM

A *pidm* is a ***P***erson ***ID***entification ***M***aster, an internal key field used to identify and store records. This is a numeric data type.

Non-Persons, such as vendors, can also have pidms.

Pidms reduce the possibility of creating multiple identification numbers or entries for the same person/entity.

If SCT Banner is used correctly, one person could have multiple IDs or name iterations but only one pidm. Extensive name searches and proper matching procedures further help to eliminate multiple entries for the same person.

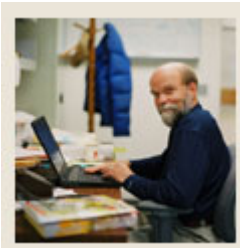
SOBSEQN

- SOBSEQN is a table which stores numbers used to generate pidms and other sequence numbers.
- It is built before Oracle incorporates sequence objects.
- All numbers should be set to zero during production setup.
- Maintenance access should be at highest security level.

Contents of SOBSEQN

```
select * from sobseqn;

SOBSEQN_FUNCTION               S SOBSEQN_MAXSEQNO SOBSEQN_A
------------------------------ - ---------------- ---------
RECEIPT                                       210 15-JUN-98
ID                             @               47 24-JUN-98
PIDM                                          559 26-JUN-98
ALUMNIGIFT                                     43 16-JUN-98
ALUMNIPLEDGE                                   23 07-JUN-98
EDIREQUESTID                                    1 25-APR-95
EDI_DCMT_SEQNO                                  1 08-DEC-95
ALUMNIDUES                                      3 06-MAY-97
ALUMNIRECEIPT                                   1 31-JAN-96
EVENT                          A                4 18-JUN-98
HRREQ                          R                0 31-JAN-96
```

- Maximum Sequence Number = Last number used.
- The setting of *maxseqno* will be discussed in greater detail in the Conversion lesson.



Section E: General Person

Lesson: PIDM and SOBSEQN (Continued)

Jump to TOC

IDM and SOBSEQN

To use the SOBSEQN table in conversion, get the maximum pidm;

```
SELECT sobseqn_maxseqno
  FROM  saturn.sobseqn
 WHERE sobseqn_function = 'PIDM'
```

Increment sobseqn_maxseqno by 1, and update SOBSEQN with the next pidm;

```
UPDATE saturn.sobseqn
   SET sobseqn_maxseqno = sobseqn_maxseqno + 1
 WHERE sobseqn_function = 'PIDM'
```

ID and SOBSEQN

The column *sobseqn_seqno_prefix* allows the client to determine the character which will precede a generated ID.

For example, a sobseqn_seqno_prefix set to "@" precedes the generated ID: *@00000001*.

A user can set the prefix to be any character. The prefix designates that an ID is assigned by the system and not entered manually.

Warning: If your institution is using Voice Response, check for conflicts or compatibility issues with special characters.

Names

Omit spaces within prefixed last names:

- MacArthur O'Connor VanHusen
- St.John deBolt DuShen

Omit spaces within hyphenated last names:

- Smith-Jones Cochram-Ashley

Use the conventional mixed-case format.

Punctuation

Use periods after prefixes and suffixes where applicable:

- Miss Mrs. Mr. Jr. III

Example:

- Prefix Firstname Hyphenated Last name
- Mrs. Joann Robinson-O'Connor

Names should have no spaces -- this makes it easier to use name search in SOAIDEN.

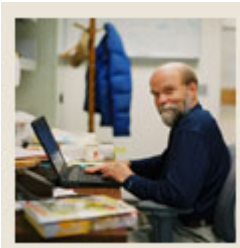
Special characters

Avoid use of the pound sign (#). Banner Letter Generation identifies a pound sign as a formatting command.

Avoid the use of the following special characters (see Student Technical Reference Manual, Chapter 5):

- / * + # & @ $

This will increase the efficiency of SCT Banner, FOCUS, BannerQuest, and any other database accessing tool, helping to minimize confusion for users.



Section E: General Person

Lesson: Data Standards (Continued)

Jump to TOC

Conversion and standard compliance

In conversion, you may need to "massage" the data in order for it to comply to standards.

Your institution may set up a committee to review and set data standards which will be documented. The committee may also make decisions regarding which office has "change control" or "maintenance responsibility" for specific data -- particularly IDs, names, and addresses. If other areas are implemented, such as HR, Finance and/or Advancement, decisions need to be made about maintenance responsibility.

Refer to the Conversion chapter of the Student System Technical Reference Manual.

Date formats (MDY, DMY, YMD)

GUAINST uses radio buttons to determine which date format is used in SCT Banner.

- MDY January 5, 1995 is entered as 01/05/95
- DMY January 5, 1995 is entered as 05/01/95
- YMD January 5, 1995 is entered as 95/01/05

If you enter only part of the date, the rest of the current date defaults

- If you are including a date in query criteria, always include the century
- You can enter a dash (-) instead of a slash (/)

Job Submission uses DD-MON-YYYY format in GJAPCTL. Accounts Receivable uses DD-MON-YY format.

Remember the date formatting that has been chosen and use that format when entering dates in the exercises.

Century

The **Century Pivot** field in GUAINST determines the cutoff year for determining which century a two-digit year belongs to. The value entered in this field will be the earliest year assigned to the 20th century.

For example, if **Century Pivot** is set to 27 and the Date Format record group is set to MDYY, then dates convert in this manner:

- 1-5-19 Converts to 05-JAN-2019
- 1-5-20 Converts to 05-JAN-2020
- 1-5-27 Converts to 05-JAN-1927
- 1-5-28 Converts to 05-JAN-1928
- 1-5-78 Converts to 05-JAN-1978
- 1-5-92 Converts to 05-JAN-1992

If you are querying information and part of the query is a year, you need to enter the century and the year to insure accuracy in your selections.

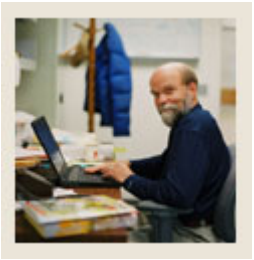
Note: Century can be overwritten when doing data entry.

Oracle format

In writing scripts, reports, etc., the ORACLE 'DD-MON-RR' date format provides additional flexibility:

- 50 - 99 = 20th century
- 00 - 49 = 21st century

See ***Oracle7Server SQL Language Reference Manual***



Section E: General Person

Lesson: General Person Procedures

Jump to TOC

Add a new person

- Navigate to SPAIDEN.
- Add a person to the form.
- Generate an ID.
- Enter Name Information (including suffix or prefix).
- Add Address Information.
- Save this information.
- Rollback to the key block.

Edit a person's ID/address

- Enter SPAIDEN again (Next Block).
- Change the ID.
- Save.
- Change the middle name.
- Save.
- Add another address (different type).
- Save.
- Rollback to the key block.

Query for a person

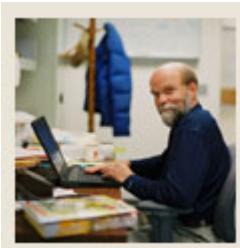
- Use the **LOV** field to access SOAIDEN.
- Perform a query to find the person you just entered.
- Notice the change indicators (I,N).
- Exit SPAIDEN.

Name search consistency

It is important for data entry staff to perform extensive, careful and consistent name searches before entering a new person/entity into the system.

Users need to know that their username is stored in the table along with the data they entered, so their errors are traceable!

Once duplicate records are entered for the same person, it is very difficult and time-consuming to correct the problem, especially if financial transactions have occurred.



Section E: General Person

Lesson: General Person Procedures (Continued)

Jump to TOC

Enter General Person information

- Navigate to SPAPERS.
- Enter SSN (SIN in Canada).
- Enter Birth Date.
- Enter Confidentiality Indicator.
- Save.
- Exit SPAPERS.

Place holds on records

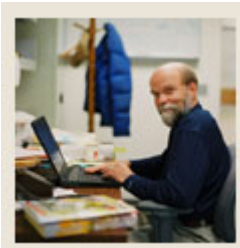
- Navigate to SOAHOLD.
- Use **LOV** field to see list of holds.
- Place 2 different types of holds on your record.
- Save.

Query for your record in SPRIDEN

- Describe SPRIDEN.
- Write a query to retrieve the data that was entered in SPRIDEN today
- where spriden_activity_date like sysdate

Notice the data in:

- spriden_change_ind
- spriden_search_last_name
- spriden_soundex_last_name
- spriden_entity_ind
- spriden_pidm



Section E: General Person

Lesson: General Person Procedures (Continued)

Jump to TOC

Query for your record in SPBPERS

- Describe SPBPERS.
- Write a query to retrieve the data that you entered in SPBPERS
  where spbpers_activity_date like sysdate

Notice the data in:

- spbpers_prefix
- spbpers_suffix
- spbpers_ssn
- spbpers_confidential_ind
- spbpers_activity_date

Note: SSN is NOT required. If institution does not use SSN or SIN (Canada) for ID, yet wishes to keep SSN stored in database for other purposes, SSN must be entered HERE.

Query for your record in SPRADDR

- Describe SPRADDR
- Write a query to retrieve the data that you entered in SPRADDR

```
select *
  from SPRADDR
 where spraddr_activity_date like sysdate;
```

Notice the data in :

- spraddr_atyp_code
- spraddr_seq_no
- spraddr_from_date
- spraddr_to_date

Note: To get information from yesterday instead of today, use
like sysdate – 1 instead of *like sysdate*.

Note: If a student changes addresses for a defined period of time, you would populate the **from** and **to_date** fields. These must be accounted for in reporting.

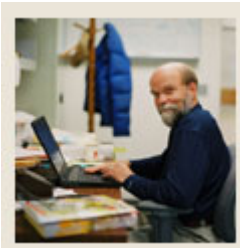
Query for your record in SPRHOLD

- Describe SPRHOLD
- Write a query to retrieve the data that you entered in SPRHOLD

```
select *
  from SPRHOLD
 where sprhold_activity_date like sysdate;
```

Notice the data in:

- sprhold_hldd_code
- sprhold_user
- sprhold_from_date
- sprhold_to_date



Section E: General Person

Lesson: SPRPDIR

Jump to TOC

Person Directory (SPRPDIR)

The Person Directory (SPRPDIR) produces a list of persons, addresses, and primary phone numbers by type of person:

- Recruit (R)
- Applicant (A)
- Student (S)
- Faculty (F)

Tables and views used

Tables used in SPRPDIR.pc:

- SPBPERS - General Person Info Table
- SRBRECR - Recruit Information Table
- SARADAP - Applicant Information Table
- SGBSTDN - Student Information Table
- SIBINST - Faculty Information Table
- SPRCOLR - Address Collector File
- SPRTELE - Telephone Number Table

View used in SPRPDIR.pc:

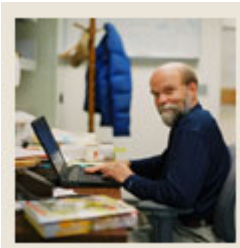
- SPVADDS - Address View

Parameters

- Term, Type, Confidentiality Indicator
- Address Type, Print ID, Faculty type (A,I,B)
- Population Selection Can Be Used
- C program
- Run via Job Submission

Example

To see a view that would be handy to modify for reporting purposes, take a look at **gpvent0.sql** in **$BANNER_HOME/general/views/gpvent0.sql**.



Section E: General Person

Lesson: Conversion Issues

Jump to TOC

Issues

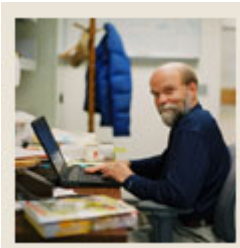
What additional general person data do you have in your legacy system?

- Become familiar with all General Person forms and tables

```
select table_name,comments
  from all_tab_comments
 where table_name like 'SP%';
```

How do you determine where to put it in SCT Banner?

- Consult users about where to put data



Section E: General Person

Lesson: Other Scripts

Jump to TOC

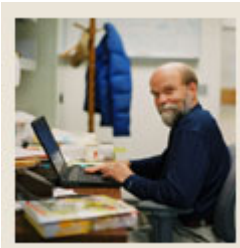
$BANNER_HOME/general/views

views (gpv*)

ag_entity_data: Object:Access view which presents general person data (gpvent0.sql)

Object:Access views are used in conjunction with the Object:Access method of retrieving data from database. This uses the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.



Section E: General Person

Lesson: Self Check – General Person Exercises

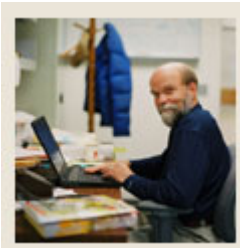
Jump to TOC

Exercise 1

Write a query to return the pidm, id, first name, middle name, last name, and change indicator for persons who have had changes made to their SPRIDEN records today.

Exercise 2

Write a query to return the id, first name, last name, and change indicator for the record that you entered about yourself in the database today. There should be an ID change indicator (‘I’), a name change indicator (‘N’) and a record in which the change indicator IS NULL. Have the query prompt you for the pidm.



Exercise 3

Write a query to select the pidm, id, first name, last name, change indicator, social security number (from SPBPERS) where changes were made to the ID records in SPRIDEN.

Exercise 4

Write a query to extract information that you would use on a mailing label. For this query, select the address type that appears the maximum number of times in the SPRADDR table. You should extract the most current record from the SPRIDEN table. For purposes of simplicity, assume that all SPRADDR records for this address type are current.

Section E: General Person

Lesson: Self Check – General Person Exercises – Answer Key

Jump to TOC

Exercise 1

Write a query to return the pidm, id, first name, middle name, last name, and change indicator for persons who have had changes made to their SPRIDEN records today.

```
SQL> SELECT  spriden_pidm,
  2   spriden_id,
  3   substr(spriden_last_name,1,15)  ||
  4   ', '                            ||
  5   substr(spriden_first_name,1,15) ||
  6   ' '    ||
  7   substr(spriden_mi,1,1),
  8   spriden_change_ind
  9  FROM spriden
 10  WHERE  spriden_entity_ind      = 'P'
 11  AND  spriden_activity_date like sysdate
 12  /
```

Exercise 2

Write a query to return the id, first name, last name, and change indicator for the record that you entered about yourself in the database today. There should be an ID change indicator (‘I’), a name change indicator (‘N’) and a record in which the change indicator IS NULL. Have the query prompt you for the pidm.

```
SQL>  SELECT spriden_id, spriden_last_name,
  spriden_first_name, spriden_change_ind
    FROM spriden
   WHERE spriden_entity_ind      = ‘P’
     AND spriden_activity_date like sysdate
     AND spriden_pidm = ‘&pidm’;
```


Exercise 3

Write a query to select the pidm, id, first name, last name, change indicator, social security number (from SPBPERS) where changes were made to the ID records in SPRIDEN.

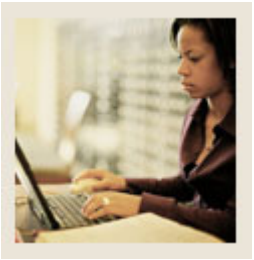
```
SQL>  SELECT spriden_pidm, spriden_id,
             spriden_first_name, spriden_last_name,
             spriden_change_ind, spbpers_ssn
        FROM spbpers, spriden
       WHERE spriden_pidm = spbpers_pidm
         AND spriden_change_ind = 'I'
```

Note: spriden_id is not necessarily the same as spbpers_ssn. That is an institutional decision.

Exercise 4

Write a query to extract information that you would use on a mailing label. For this query, select the address type that appears the maximum number of times in the SPRADDR table. You should extract the most current record from the SPRIDEN table. For purposes of simplicity, assume that all SPRADDR records for this address type are current.

```
Step 1:
SQL>SELECT DISTINCT spraddr_atyp_code, count(*)
      FROM spraddr
  GROUP BY spraddr_atyp_code

RESULT:   List of address types with counts of each type.  Choose
max count.

Step 2:
 SQL> SELECT spriden_first_name || ' ' ||
          spriden_last_name, spraddr_street_line1,
          spraddr_street_line2,
          spraddr_city || ' ' ||
          spraddr_stat_code || '  ' || spraddr_zip
      FROM spriden, spraddr
     WHERE spriden_pidm = spraddr_pidm
       AND spriden_change_ind IS NULL
       AND spraddr_atyp_code =  'PR';
```



Section F: Curriculum/Program Rules

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the forms and tables used in curriculum/program rules functionality

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

Section F: Curriculum/Program Rules

Lesson: Curriculum/Program Rules Overview

Jump to TOC

Introduction

Although Curriculum and Program rules are not a separate module, those tables are introduced now because of their connections to Recruiting, Admissions and General Student.

These rules are not required, but most institutions use them because of CAPP and Web for Students - Admissions.

Objectives

Examine

- Forms used to build rules
- Table relationships

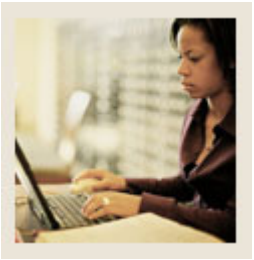
Overview

Your functional consultant will go over the setting up of these rules in detail.

The purpose of this lesson is to examine three forms and six tables which should be set up before implementing Recruiting and Admissions. The lesson will illustrate how to discover the underlying tables and point out the relationships among the tables.

The lesson will also illustrate which tables must be populated during conversion if you are going to use curriculum rules.

For more details on setting up and using program and curriculum rules, CAPP training is appropriate. You can look at Chapter 9 of the CAPP user manual for a detailed discussion of Curriculum rules, as well as the Student User Manual, in Chapters 10(Recruiting), 11(Admissions), 12(General Student), 13(Registration) and 15(Academic History).

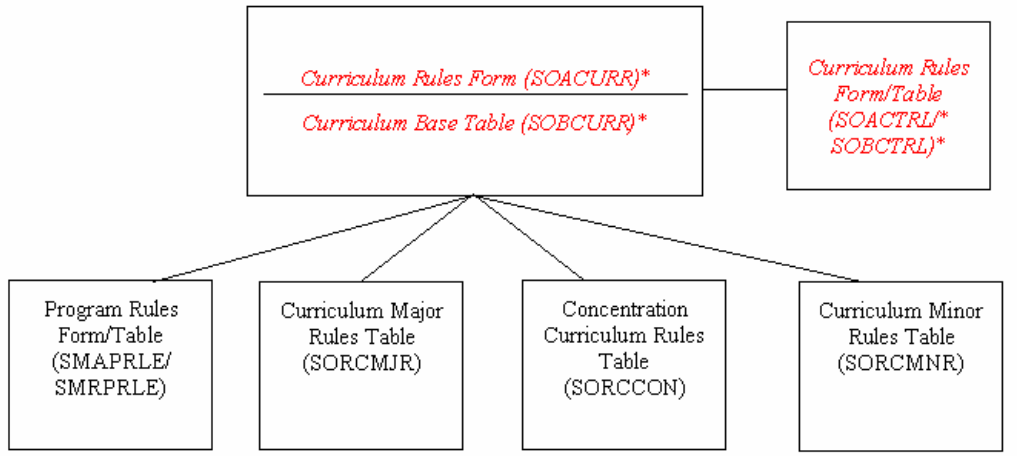


Section F: Curriculum/Program Rules

Lesson: Curriculum/Program Rules Overview (Continued)

Jump to TOC

Diagram



Notice the naming convention:

- SOACURR is an OVERALL form -- used by many modules
- SMAPRLE is part of the CAPP module -- M is the letter for that module

Major forms

- Program Definition Rules Form (SMAPRLE)
- Curriculum Rules Form (SOACURR)
- Curriculum Rules Control Form (SOACTRL)

If your institution is planning to use CAPP, you will be concerned with the SMAPRLE (Program Rules) form. If not, and you do plan to use curriculum rules, you will only need to be concerned with SOACURR and SOACTRL.

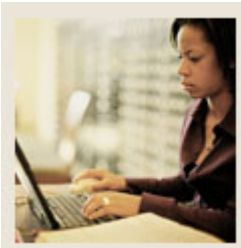
Major tables

- Program Definition Rules Table (SMRPRLE)
- Curriculum Rules Form (SOBCURR)
- Curriculum Rules Control Table (SOBCTRL)
- Curriculum Major Rules Table (SORCMJR)
- Curriculum Minor Rules Table (SORCMNR)
- Curriculum Concentration Rules Table (SORCCON)

Major validation tables

- Term Code Validation Form/Table (STVTERM)
- Level Code Validation Form/Table (STVLEVL)
- College Code Validation Form/Table (STVCOLL)
- Degree Code Validation Form/Table (STVDEGC)
- Campus Code Validation Form/Table (STVCAMP)
- Department Code Validation Form/Table (STVDEPT)
- Major, Minor, Concentration Code Validation Form/Table (STVMAJR)

These validation tables must be set up if you use curriculum rules.



Section F: Curriculum/Program Rules

Lesson: Program Definition Rules Form (SMAPRLE)

Jump to TOC

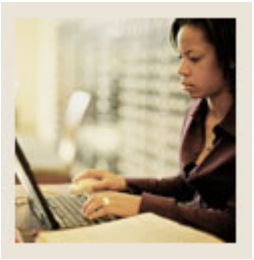
SMAPRLE

SMRPRLE is the underlying table.

- Program is required only if CAPP's Program Planning indicator is set to 'Yes' in SOACTRL.
- sobctrl_program_ind = 'Y'
- Part of CAPP module
- Program is not required unless you are using CAPP and/or unless you set CAPP's Program Planning Indicator to 'Y' on SOACTRL.

When program is used on a curriculum rule, the following must match on SOACURR what is defined in SMAPRLE:

- Level
- Campus
- College
- Degree



Section F: Curriculum/Program Rules

Lesson: Curriculum Rules Form (SOACURR)

Jump to TOC

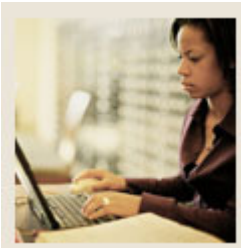
SOACURR

SOBCURR is the underlying table.

- Used to view or create curriculum rules
- Rules are based on Program Definitions if you are using program rules; otherwise, program is not a required field

Note: The key block for SOACURR uses term, which is optional. If you put the term in the key, the form only shows you the rules which are valid for that term; no future term rules are displayed.

Note: When program is used on a rule, the level, campus, college, and degree have to match what has been defined on SMAPRLE. If the campus on SMAPRLE is blank, all campuses are valid for the rule. The information defaults back into SOACURR from the List of Values window for SMAPRLE.



Section F: Curriculum/Program Rules

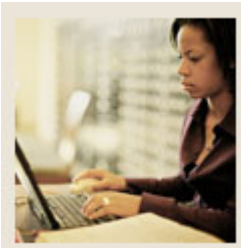
Lesson: Curriculum Rules Control Form (SOACTRL)

Jump to TOC

SOACTRL

SOBCTRL is the underlying table.

- Indicators determine if/how various areas related to curriculum are used
- Can set “Use CAPP’s Program Planning” to ‘Y’ or ‘N’
- Indicators set severity level of error checking by module if curriculum rules are used



Section F: Curriculum/Program Rules

Lesson: Major, Minor, Concentration Rules Forms

Jump to TOC

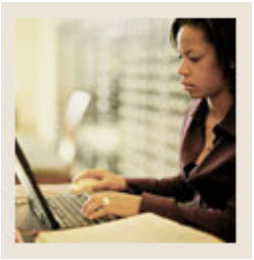
Major, Minor, Concentration

- Curriculum Major Rules Form (SORCMJR)
- Curriculum Minor Rules Form (SORCMNR)
- Curriculum Concentration Rules Form (SORCCON)

Each table contains on/off indicators for each module using curriculum rules.

- e.g. Admissions: sorcmjr_adm_ind = 'Y'

The data from these tables shows up through SOACURR.



Section F: Curriculum/Program Rules

Lesson: Conversion Issues

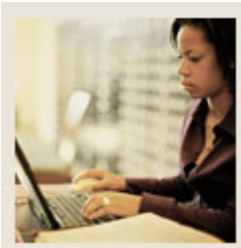
Jump to TOC

Issues

- Will your users build curriculum rules?
- If so, then can you use the rules to your advantage when converting student data?
- Can you use the student's major (on legacy side) to get the valid department and program codes from SOBCURR and SORCMJR?

If you convert the legacy major codes to match Banner major codes in STVMAJR, then you can run a query which will use curriculum rules to give you the valid department and program.

You can use the converse of this to determine which records on your legacy system will have an invalid major when converted to Banner.



Section F: Curriculum/Program Rules

Lesson: Summary

Jump to TOC

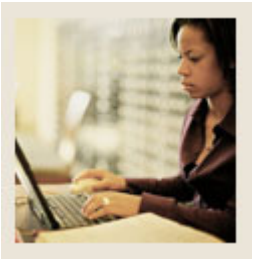
Summary

- Build rules in SOACURR
  - All curriculum rules must be built before setting indicators in SOACTRL

- Build Program Rules on SMAPRLE (if you plan to use CAPP's Program Planning)

- Build control rules in SOACTRL
  - if sobctrl_curr_rule_ind = 'Y', then sobctrl_program_ind must = 'Y'. This means that you are using CAPP's Program Planning.
  - This indicator means that major curriculum rules are "turned on" for STUDENT: sorcmjr_stu_ind = 'Y

- Build control rules in SOACTRL
  - if sobctrl_curr_rule_ind = 'N' in SOACTRL then sobctrl_program_ind can = 'Y'
  - This means that you **are not** using CAPP's Program Planning, but you **are** using curriculum rules



Section F: Curriculum/Program Rules

Lesson: Self Check – Curriculum/Program Rules Exercises

Jump to TOC

Exercise 1

Write a query to retrieve curriculum rules that apply to STUDENT, listing:

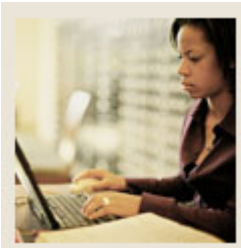
- major and program
- department code
- level code
- college code
- campus code
- degree code

Exercise 2

Write a query to retrieve a list of students who have an invalid major based on the curriculum rules, selecting:

- id
- last name
- college code
- degree code
- major code

(This is an advanced exercise.)



Section F: Curriculum/Program Rules

Lesson: Self Check – Curriculum/Program Rules Exercises – Answer Key

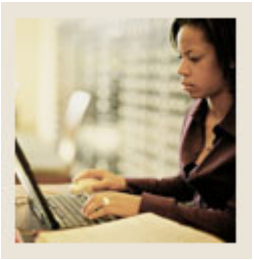
Jump to TOC

Exercise 1

Write a query to retrieve curriculum rules that apply to STUDENT, listing:

- major and program
- department code
- level code
- college code
- campus code
- degree code

```
select sorcmjr_majr_code, sobcurr_program,
       sorcmjr_dept_code, sobcurr_levl_code,
       sobcurr_coll_code, sobcurr_camp_code,
       sobcurr_degc_code
  from sobcurr, sorcmjr
 where sobcurr_curr_rule = sorcmjr_curr_rule
   and sorcmjr_stu_ind = 'Y';
```



Section F: Curriculum/Program Rules

Lesson: Self Check – Curriculum/Program Rules Exercises – Answer Key (Continued)

Jump to TOC

Exercise 2

Write a query to retrieve a list of students who have an invalid major based on the curriculum rules, selecting:

- id
- last name
- college code
- degree code
- major code

(This is an advanced exercise.)

```
select spriden_id, spriden_last_name,
       sgbstdn_coll_code_1, sgbstdn_degc_code_1,
       sgbstdn_majr_code_1
  from spriden, sgbstdn
 where spriden_pidm = sgbstdn_pidm
   and not exists (select 'x'
                     from sorcmjr, sobcurr
                    where sobcurr_curr_rule =
                          sorcmjr_curr_rule
                      and sorcmjr_majr_code =
                          sgbstdn_majr_code_1
                      and sobcurr_coll_code =
                          sgbstdn_coll_code_1
                      and sobcurr_degc_code =
                          sgbstdn_degc_code_1);
```

Section G: Recruiting

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Recruiting module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

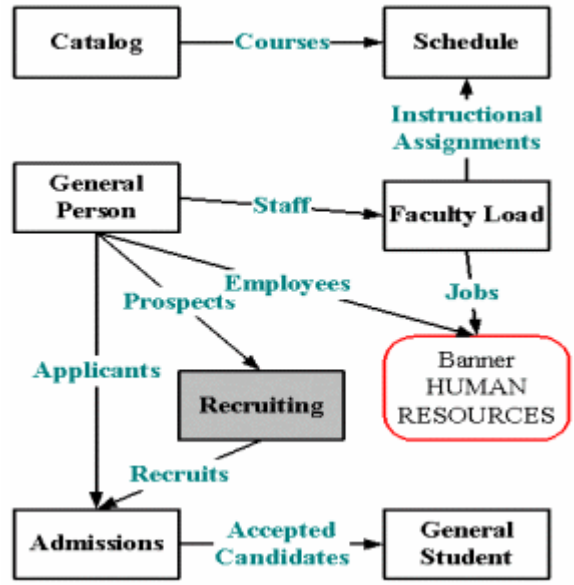
Section Contents

Section G: Recruiting

Lesson: SCT Banner Student Recruiting Module

Jump to TOC

Diagram



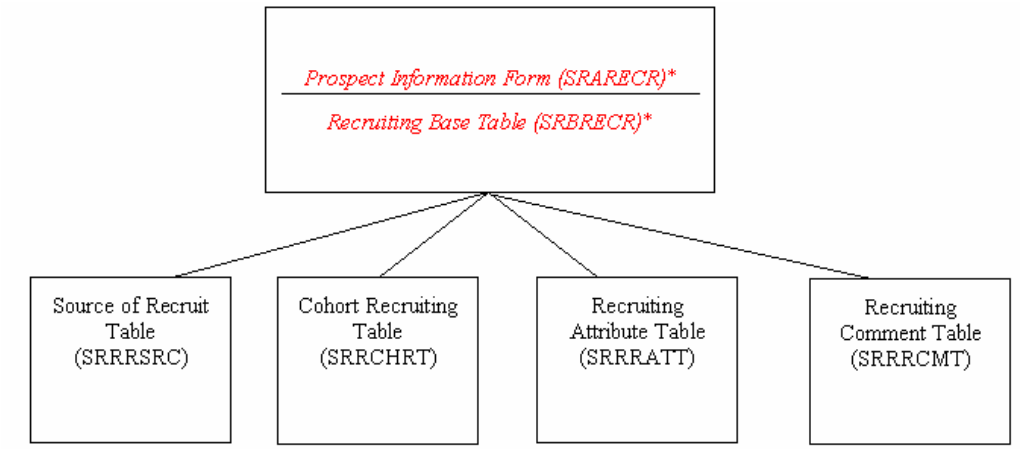
This section covers the Recruiting module. If it is implemented, it would be best to follow this order, working on Recruitment after Catalog, General Person and Curriculum Rules.

Objectives

Examine

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Diagram



Only one table is required for conversion, if you choose to convert (if you do, you must also have appropriate validation tables set up).

Major forms and tables

Major Form:

- Prospect Application Form (SRARECR)

Major Tables:

- Recruiting Base Table (SRBRECR)
- Source of Recruit Table (SRRRSRC)
- Cohort Recruiting Table (SRRCHRT)
- Recruiting Attribute Table (SRRRATT)
- Recruiting Comment Table (SRRRCMT)
- Curriculum Rules Tables

Notice patterns in the names of forms and tables on this page.

Major Validation Tables/Forms:

- Term Code Validation Form/Table (STVTERM)
- Level Code Validation Form/Table (STVLEVL)
- College Code Validation Form/Table (STVCOLL)
- Degree Code Validation Form/Table (STVDEGC)
- Major, Minor, Concentration Code Validation Form/Table (STVMAJR)
- Student Type Validation Form/Table (STVSTYP)
- Residence Code Validation Form/Table (STVRESD)

Section G: Recruiting

Lesson: Prospect Information Form (SRARECR)

Jump to TOC

SRARECR

This form provides information necessary for all recruitment related activities, and is the basis for all related recruiting forms.

- Can go to SPAIDEN form to create a person record from this form
- Notice connections to Curriculum

Section G: Recruiting

Lesson: Quick Recruit Form (SRAQUIK)

Jump to TOC

SRAQUIK

This form allows entry of new prospective students.

General Person information is created via this form (populating tables: SPRIDEN, SPBPERS, SPRADDR, etc.), along with other information (populating tables: SORHSCH, SORPCOL, SORINTS, SRRRSRC, SORCONT etc.)

Note: This form allows direct creation of a person record (you don't go to SPAIDEN--the form does that in the background).

Section G: Recruiting

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of Recruiting Module?

```
select table_name
  from all_tables
 where table_name like 'SR%'
```

- What data elements are required?

```
desc srbrecr
```

  - Notice the "NOT NULL" columns.

- What are the key fields in srbrecr?

```
SQL> select column_name
       from all_cons_columns
      where table_name = 'SRBRECR'
        and constraint_name = 'PK_SRBRECR';
```

- Describe each table: SRBRECR, SRRRSRC, SRRCHRT, SRRRATT, SRRRCMT

```
SQL> desc srbrecr
SQL> desc srrrsrc
SQL> desc srrchrt
SQL> desc srrratt
SQL> desc srrrcmt
```

Section G: Recruiting

Lesson: Reports

Jump to TOC

Reports

- Recruiting Enrollment Analysis (SRRENRL)
- Recruits Never Applied to Institution Report (SRRINQR)

$BANNER_HOME/student/dbprocs

functions (srf*)

$BANNER_HOME/student/views

views (srv*): srvrecr0.sql creates view called as_recruiting_data

Some views are used in conjunction with the Object:Access method of retrieving data from database. This method uses the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

f_gurmail_rowid in sofmail in student views

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.

Section G: Recruiting

Lesson: Conversion Issues

Jump to TOC

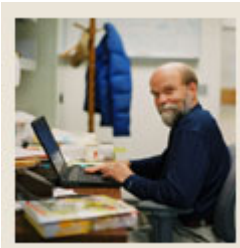
Questions

- Will Recruiting data be converted or entered manually by the users?
- What Recruiting data do you have in your legacy system?

```
select table_name,comments
from all_tab_comments
where table_name like 'SR%';
```

- How do you determine where to put it in SCT Banner?
- Will you use curriculum rules?

Recruiting Module, if used, is usually brought up first -- and is usually a manual process of setting up validation tables and curriculum rules (if they are to be used with Recruiting). ***Conversion of legacy data is not common.***



Section H: Admissions

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Admissions module

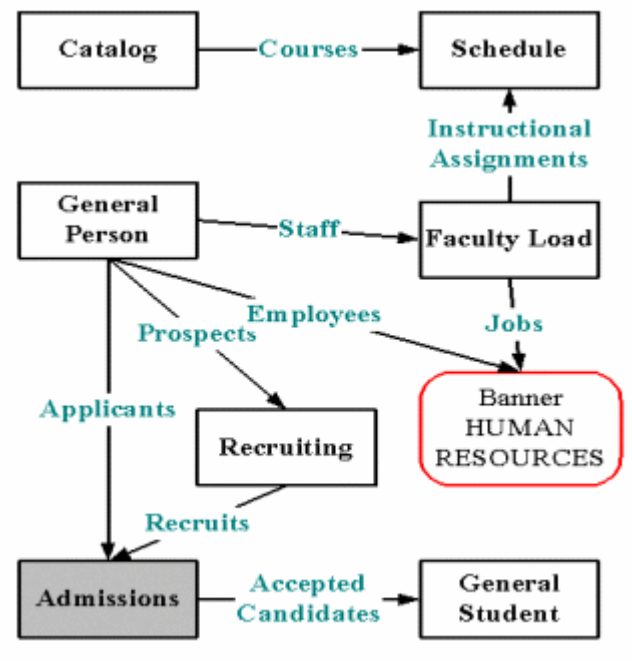
Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

Diagram

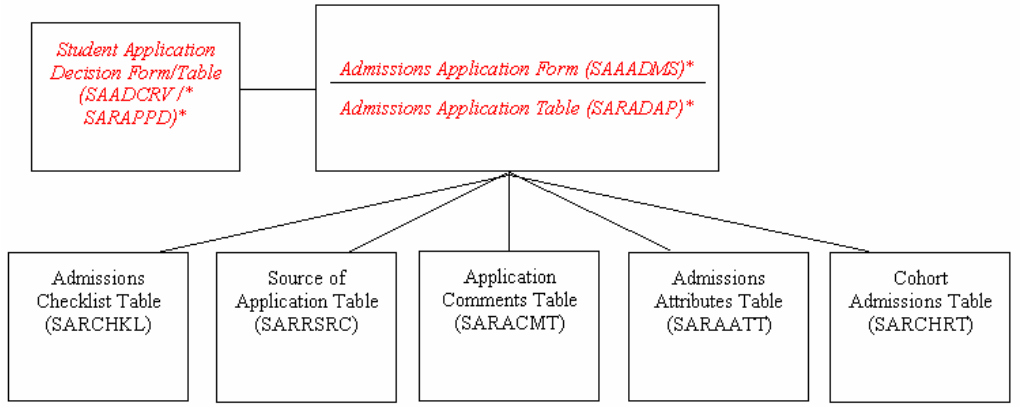


Objectives

Examine/Review

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Diagram



Required tables

The required tables are SARADAP for Admissions application data and SARAPPD for Application decision data.

Major Validation Tables/Forms

- Term Code Validation Form/Table (STVTERM)
- Level Code Validation Form/Table (STVLEVL)
- College Code Validation Form/Table (STVCOLL)
- Degree Code Validation Form/Table (STVDEGC)
- Major, Minor, Concentration Code Validation Form/Table (STVMAJR)
- Student Type Code Validation Form/Table (STVSTYP)
- Residence Code Validation Form/Table (STVRESD)
- Admission Application Status Code Validation Form/Table (STVAPST)
- Admission Application Decision Code Validation Form/Table (STVAPDC)
- Test Code Validation Form/Table (STVTESC)
- Degree Level Code Validation Form/Table (STVDLEV)
- Degree Award Category Code Validation Form/Table (STVACAT)
- State/Province Code Validation Form/Table (STVSTAT)
- Letter Code Validation Form/Table (GTVLETR)
- Paragraph Code Validation Form/Table (GTVPARA)

All of these Validation Tables are necessary for the Admissions Module.

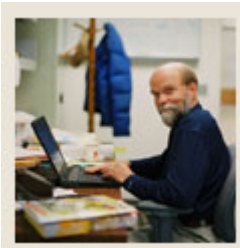
Major Validation Tables/Forms (cont.)

Notice that many of the tables were previously mentioned in preceding modules, particularly in Recruiting.

GTVLETR and GTVPARA are used in Letter Generation. You may want to set these up before bringing up Recruiting if you plan to do that before Admissions and if you plan to send mail to prospects.

New validation tables:

- STVAPST (Admission Application Status Codes)
- STVAPCD (Admission Application Decision Codes)



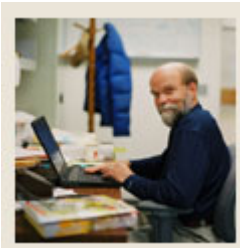
Section H: Admissions

Lesson: Admissions Application Form (SAAADMS)

Jump to TOC

SAAADMS

This form is used for maintaining applications submitted to the institution. It can maintain an unlimited number of applications for any given term (saradap_term_code_entry, saradap_appl_no).



Section H: Admissions

Lesson: Quick Admit Form (SAAQUIK)

Jump to TOC

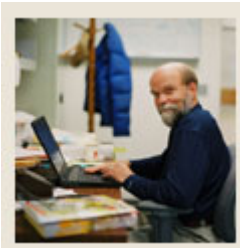
SAAQUIK

This form allows entry and registration of new students with minimal effort.

General Person information is created via this form (populating tables: SPRIDEN, SPBPERS, SPRADDR, SPRTELE, etc.).

Admissions and/or Recruitment records may be created through this form.

Other information can be accessed via this form (tables: SORHSCH, SORPCOL, SPRHOLD, SORTEST, SPRINTL, etc.).



Section H: Admissions

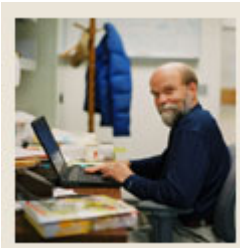
Lesson: Admissions Decision Form (SAADCRV)

Jump to TOC

SAADCRV

The underlying table is SARAPPD.

Once an applicant is accepted through SAADCRV, a student record is created (SGASTDN form/SGBSTDN table).



Section H: Admissions

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Admissions Module?

```
select table_name
  from all_tables
 where table_name like 'SA%'
```

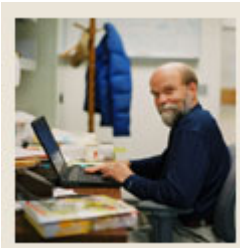
- What data elements are required?

```
desc saradap
```

  - Notice the "NOT NULL" columns.

- What are the key fields in SARADAP?

```
select column_name
  from all_cons_columns
 where table_name = 'SARADAP'
   and constraint_name = 'PK_SARADAP';

select table_name, comments
  from all_tab_comments
 where table_name like 'SA%';
```



Section H: Admissions

Lesson: Reports

Jump to TOC

Reports

- Admissions Count by College/Major Report (SARACTM)
  This report prints admission application count by college/major.
  - C program run from Job Submission
- Admissions Application Report (SARADMS)
- Admission Decision Criteria Report (SARDCSN)

Other reports and purge processes are also available for the Admissions module. Refer to Chapter 10 of the Student System Technical Reference Manual for a complete list of reports/processes.

$BANNER_HOME/student/dbprocs

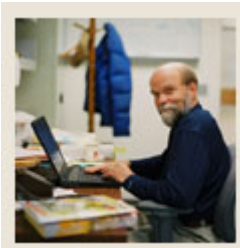
functions (saf*)

$BANNER_HOME/student/views

views (sav*): savadm0.sql creates as_admissions_applicant

Some views are used in conjunction with the Object:Access method of retrieving data from database. This method uses the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



Section H: Admissions

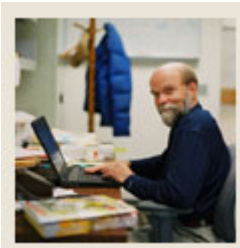
Lesson: Conversion Issues

Jump to TOC

Questions

- Will Admissions data be converted or entered manually by the users?
- What Admissions data do you have in your legacy system?

Admissions is typically not converted for going live but may be converted later for Institutional Research purposes.



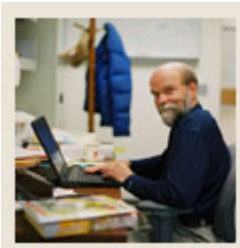
Section H: Admissions

Lesson: Self Check – Admissions Exercise

Jump to TOC

Exercise

Write a query to get the id, last name, term of entry and student type for applicants for a specific future term (prompt user for term code). The records returned should be for the most current application for that term and the decision should be the most recent decision made that matches that application.



Section H: Admissions

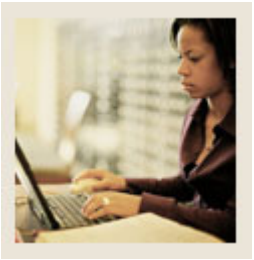
Lesson: Self Check – Admissions Exercise – Answer Key

Jump to TOC

Exercise

Write a query to get the id, last name, term of entry and student type for applicants for a specific future term (prompt user for term code). The records returned should be for the most current application for that term and the decision should be the most recent decision made that matches that application.

```
select spriden_id, substr(spriden_last_name, 1,15),
       saradap_term_code_entry, saradap_styp_code
  from spriden, saradap
 where saradap_term_code_entry = &TERM
   and saradap_pidm in
        (select sarappd_pidm
           from sarappd x
          where sarappd_term_code_entry =
                saradap_term_code_entry
            and sarappd_pidm = saradap_pidm
            and saradap_appl_no = sarappd_appl_no
            and sarappd_seq_no =
                 (select max(sarappd_seq_no)
                    from sarappd
                   where x.sarappd_pidm = sarappd_pidm
                     and x.sarappd_term_code_entry =
                         sarappd_term_code_entry
                     and x.sarappd_appl_no =
                         sarappd_appl_no))
   and spriden_pidm = saradap_pidm
   and spriden_change_ind is null
          order by spriden_last_name
```



Section I: Overall Forms and Tables

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe major forms and tables referenced through many areas of SCT Banner

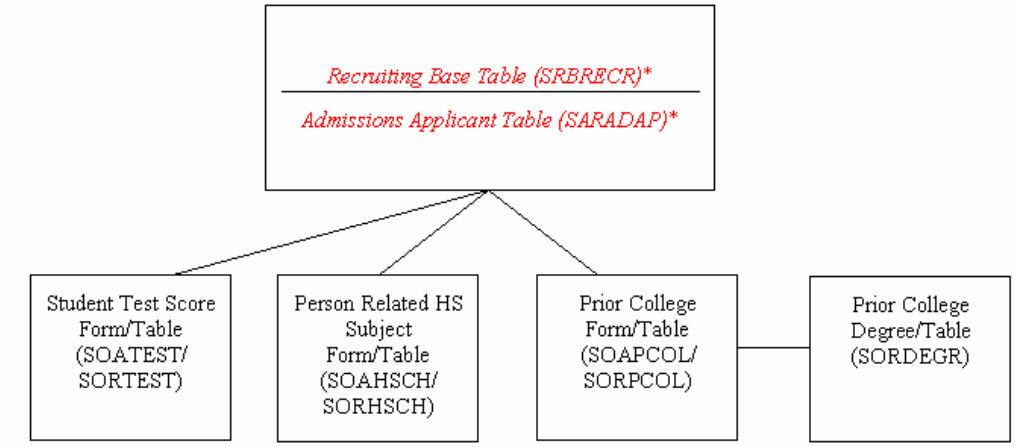
Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

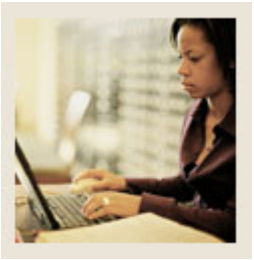
Diagram



Overall tables are not restricted to connections with Recruitment or Admissions, but are referenced through many areas of SCT Banner.

Overall tables are referenced in reporting.

Many other overall forms may be accessed from the General Student Module, through the Educational Background Menu and other menus.



Section I: Overall Forms and Tables

Lesson: Overall Forms and Tables (Continued)

Jump to TOC

Major forms

- Test Score Information Form (SOATEST)
    - Used to maintain test score information.
- High School Information Form (SOAHSCH)
    - Used with Recruiting and Admissions for high school information
- Prior College Form (SOAPCOL)
    - Used with Recruiting, Admissions, and Faculty Load for prior college information

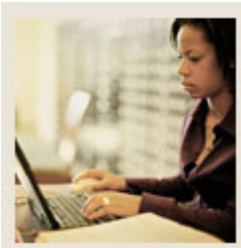
Major tables

- Student Test Score Table (SORTEST)
- Person Related HS Subject Table (SORHSCH)
- Prior College Table (SORPCOL)
- Prior College Degree Table (SORDEGR)

Major Validation Tables/Forms:

- Test Code Validation Form/Table (STVTESC)
- Source/Background Institution Code Validation Form/Table (STVSBGI)
- Degree Code Validation Form (STVDEGC)

You will use STVTESC in the exercise at the end of the lesson.



Section I: Overall Forms and Tables

Lesson: SQL*Plus

Jump to TOC

Questions

- What are the tables that are used for multiple modules?

```
select table_name, table_type, comments
  from all_tab_comments
 where table_name like 'SO%'
   and table_type = 'TABLE';
```

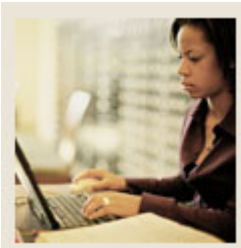
- Are any data elements required in SORTEST, SORHSCH, SORPCOL?
  - Examine GUROPTM;

```
select *
  from guroptm
 where guroptm_form_name = 'SOATEST';
```

  This will show what forms are related to 'SOATEST' and how they are related, (triggers, etc).

- How many overall tables are there, and what is a description of their content?

```
select table_name, table_type, comments
  from all_tab_comments
 where table_name like 'SO%'
   and table_type = 'TABLE';
```



Section I: Overall Forms and Tables

Lesson: Conversion Issues

Jump to TOC

Questions

- Will Overall data be converted or entered manually by the users?
- What Overall data do you have in your legacy system?
- How do you determine where to put it in SCT Banner?

Interface Tape Load Process (SORTAPE)

- Run the SORTAPE process from Job Submission.
- This process uses the conversion and default values from SOBCNVT. If the process fails it is possible that a code does not have the appropriate conversion value. One of the parameters of this process allows you to delete the record(s) from the temporary tables as it gets inserted into the SCT Banner table. If you need to rerun SORTAPE because a conversion value is not in SOBCNVT, you may add the value to the form SOTCNVT, then rerun SORTAPE for the remainder of the records still in the temporary tables.
- For more information about SORTAPE, refer to Chapter 14 of the Student Technical Reference Manual.

Tape Interface Rules Form (SOAINFR)

- Establish rules for each tape type.

Tape Code Conversion Form (SOTCNVT)

- Establish converted values on the Tape Code Conversion Form.
- Load the tape to a flat file
- Clean out the temporary tables (run $BH/student/plus/ sostdel.sql
- Run the SQL*Loader from UNIX prompt
  (sqlldr userid=username/password, control=sat9495.ctl, data=sat9798.dat)

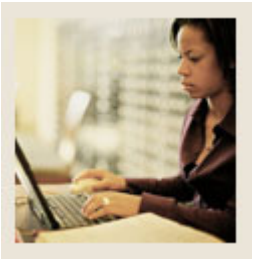
Note: For a good discussion of SOTCNVT, refer to Chapter 16 of the Student Users Manual.

Note: sat9495.ctl should not need to be modified unless the tape layout changes.

Suspended Records Maintenance Form (SOASUSP)

Tape Comparison Processing Report (SORINFR)

- Run SORINFR from UNIX prompt to compare data in temporary tables to existing SCT Banner information. SORINFR identifies Matches, New records, or Errors (The source code is located in /u01/banner/test/student/c).
- If this process produces any records that have an indicator other than M - Match or N - New, then the records can be viewed and corrected on the Suspended Records Maintenance Form (SOASUSP).



Section I: Overall Forms and Tables

Lesson: Self Check – Overall Forms and Tables Exercise

Jump to TOC

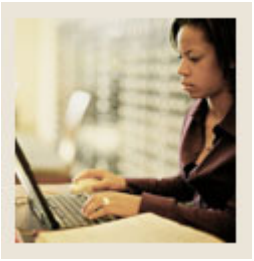
Exercise

Get the following information about all applicants for a term (prompt for term):

- Full Name
- Entry Term
- Test Code
- Test Score
- High School GPA

for Students who took either the ACT English or the SAT Verbal tests.

Your query should return only records with values in all the above areas.



Exercise

Get the following information about all applicants for a term (prompt for term):

- Full Name
- Entry Term
- Test Code
- Test Score
- High School GPA

for Students who took either the ACT English or the SAT Verbal tests.

Your query should return only records with values in all the above areas.

```
Step 1
desc stvtesc
-find SAT Verbal and ACT English code
Step 2
desc sortest
-get proper column names
Step 3
desc sorhsch
-find column name for hs gpa
Step 4

select  spriden_id, substr(spriden_last_name, 1,15)
       || ', ' ||  substr(spriden_first_name,1.15),
        saradap_term_code_entry, sortest_tesc_code,
        sortest_test_score, sorhsch_gpa
  from    spriden, saradap, sortest, sorhsch
 where    sorhsch_pidm = saradap_pidm
   and  saradap_term_code_entry = '&term'
   and  sorhsch_pidm = sortest_pidm
   and  sortest_tesc_code IN ('S01', 'A01')
   and  sorhsch_pidm = spriden_pidm
   and  spriden_change_ind is null
   and    sorhsch_gpa is not null
 order by spriden_last_name;
```

Section J: Faculty Load

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Faculty Load module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

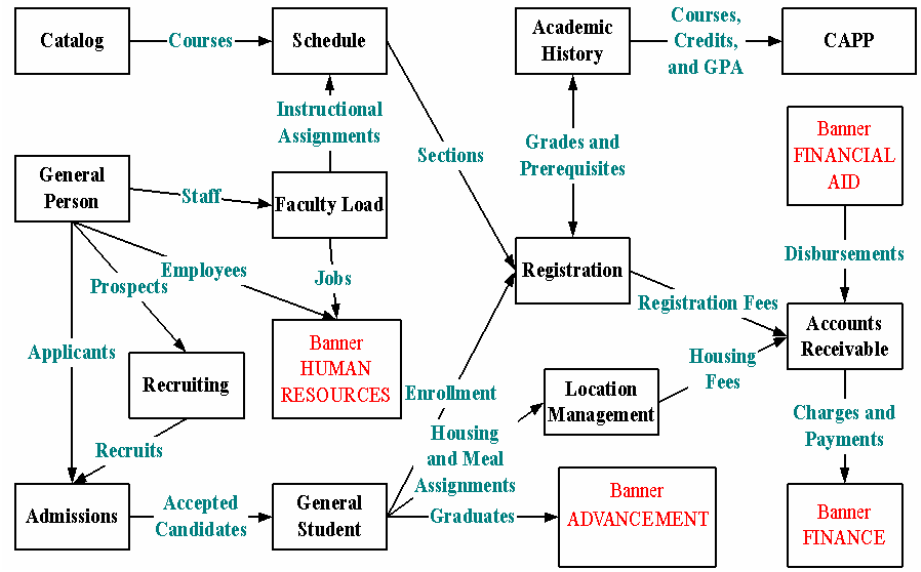
Section Contents

Section J: Faculty Load

Lesson: Student System Overview

Jump to TOC

Diagram

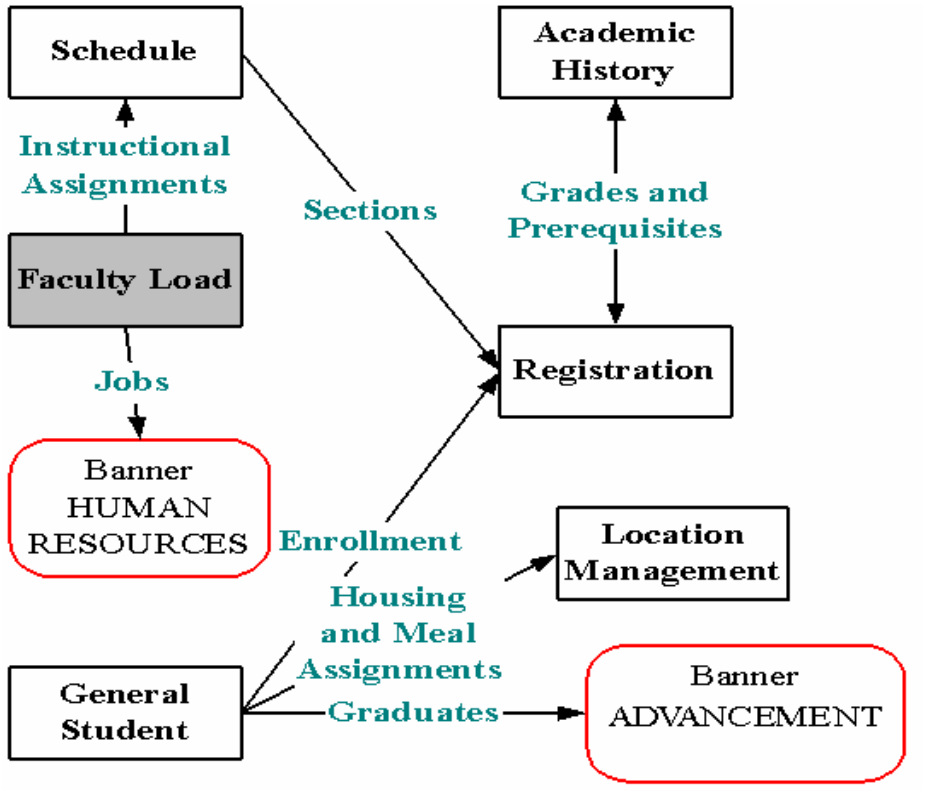


Section J: Faculty Load

Lesson: Faculty Load Module

Jump to TOC

Diagram



Overview

Faculty Load enables users to enter and maintain information including instructional and non-instructional assignments for a faculty member or advisor.

Personnel information, such as tenure status and sabbatical dates, is maintained in this module along with workload and contract information.

Faculty MUST have a SPRIDEN record (etc.) before they can be designated as Faculty (Advisor or Instructor).

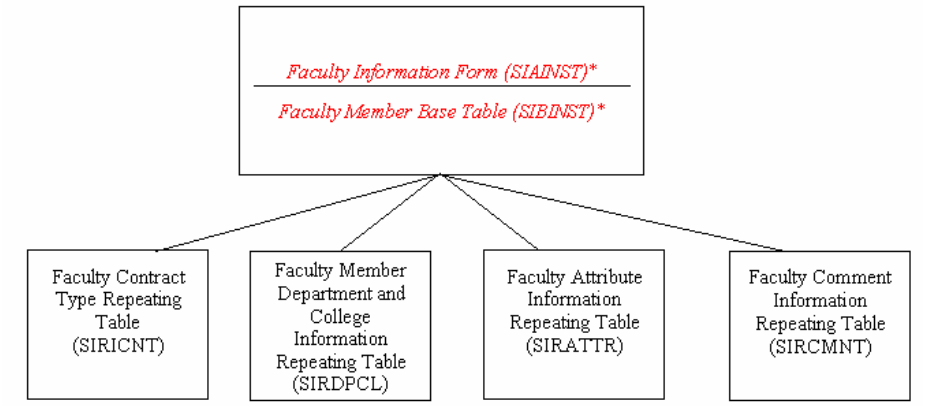
A person must be flagged as an “instructor” in the SIBINST table (SIAINST form) before he/she can be assigned an instructional section in SCHEDULE.

Section J: Faculty Load

Lesson: Faculty Load

Jump to TOC

Forms and tables



Faculty Load is a "swing" module, which may be shared between HR and Student. Decisions need to be made about ownership/responsibility of maintenance.

- SIBINST is the only required table.
- SIRASGN is populated when the schedule is done

Major Validation Tables/Forms

- Faculty Status Code Validation Form/Table (STVFCST)
  - STVFCST is the Faculty Status validation table; it is required because it is used in SIBINST.
- Faculty Category Code Validation Form/Table (STVFCTG)
- Faculty Staff Type Code Validation Form/Table (STVFSTP)
- Faculty Contract Type Code Validation Form/Table (STVFCNT)
- Term Workload Rules Code Validation Form/Table (STVWKLD)
- Contract Rules Validation Form/Table (STVCNTR)

Section J: Faculty Load

Lesson: Faculty Information Form (SIAINST) / Faculty Member Base Table (SIBINST)

Jump to TOC

SIAINST / SIBINST

This form and table are used to maintain Faculty Information.

Codes and Indicators for:

- Active/Inactive (sibinst_fcst_code)
- Instructor (sibinst_schd_ind)
- Advisor (sibinst_advr_ind)

SIAINST also utilizes these tables (but data is not required in SIBINST):

- Faculty Contract Type Repeating Table (SIRICNT)
- Faculty Member Department and College Information Repeating Table (SIRDPCL)
- Faculty Attribute Information Repeating Table (SIRATTR)
- Faculty Comment Information Repeating Table (SIRCMNT)

The Prior College Degree Table (SORDEGR) is used in the Faculty Degree Information Form (SIAFDEG) form to maintain faculty degree information.

Section J: Faculty Load

Lesson: Faculty Assignment Form (SIAASGN) / Faculty Assignment Table (SIRASGN)

Jump to TOC

SIAASGN / SIRASGN

This form and table contain faculty teaching assignments for a particular term. It is populated automatically when a faculty member is entered on the SSASECT form in the schedule module, if records exist in SIBINST (faculty status).

Parts of SIAASGN are updated automatically when faculty information is entered in the SSASECT form. The **Assignment Type** field IS NOT updated and must be updated manually.

After building sections in SSASECT/SSBSECT, assignment information will be present in SIRASGN.

SIRASGN is a good table to use for faculty load reports. You will need to include SIRASGN in schedule reports to get faculty pidms in connection with class assignments.

Section J: Faculty Load

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Faculty Load Module?

```
select table_name
  from all_tables
 where table_name like 'SI%'
```

- What data elements are required?

```
desc sibinst
```

  - Notice the "NOT NULL" columns.

- What are the key fields in sibinst?

```
select column_name
  from all_cons_columns
 where table_name = 'SIBINST'
   and constraint_name = 'PK_SIBINST';
```

Warning: SIBINST_ADVR_IND MUST be filled in if you want to use faculty as advisors when you get to General Student, even though it is not required by the table.

Section J: Faculty Load

Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Faculty Load Purge (SIPASGN)
- Faculty Schedule Report (SIRASGQ)
- Faculty Load Contract Analysis Report (SIRCTAL)
- Faculty Load Term Analysis Report (SIRTRAL)

Refer to the Student Technical Reference Manual, Chapter 10, for additional information on Student Module reports and processes.

Don't forget that the Person Directory Process (SPRPDIR) can generate the directory information, including for faculty.

$BANNER_HOME/student/dbprocs

functions (sif*)

$BANNER_HOME/student/views

views (siv*)

Some views are used in conjunction with Object:Access method of retrieving data from database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.

Section J: Faculty Load

Lesson: Conversion Issues

Jump to TOC

Questions

- Will Faculty Load data be converted or entered manually by the users?
- What Faculty Load data do you have in your legacy system?
- How do you determine where to put it in SCT Banner?

When we talk about SGBSTDN (Student record), we will see that the advisor pidm is stored in this table. To ensure that the student's advisor will appear in the form, the advisor must be active for that term and "flagged" as an adviser (sibinst_advr_ind).

Section J: Faculty Load

Lesson: Self Check – Faculty Load Exercise

Jump to TOC

Exercise

Write a query which would return the full name, id, faculty status and effective term for that status for an instructor.

Section J: Faculty Load

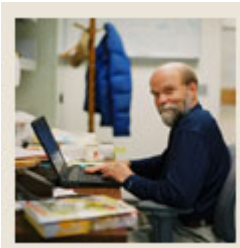
Lesson: Self Check – Faculty Load Exercise – Answer Key

Jump to TOC

Exercise

Write a query which would return the full name, id, faculty status and effective term for that status for an instructor.

```
select substr(spriden_last_name,1,15) || ', ' ||
       substr(spriden_first_name,1,15), spriden_id,
       sibinst_term_code_eff, sibinst_fcst_code
  from spriden, sibinst
 where sibinst_pidm = spriden_pidm
   and spriden_change_ind is null
 order by spriden_last_name
```



Section K: Location Management

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Location Management module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

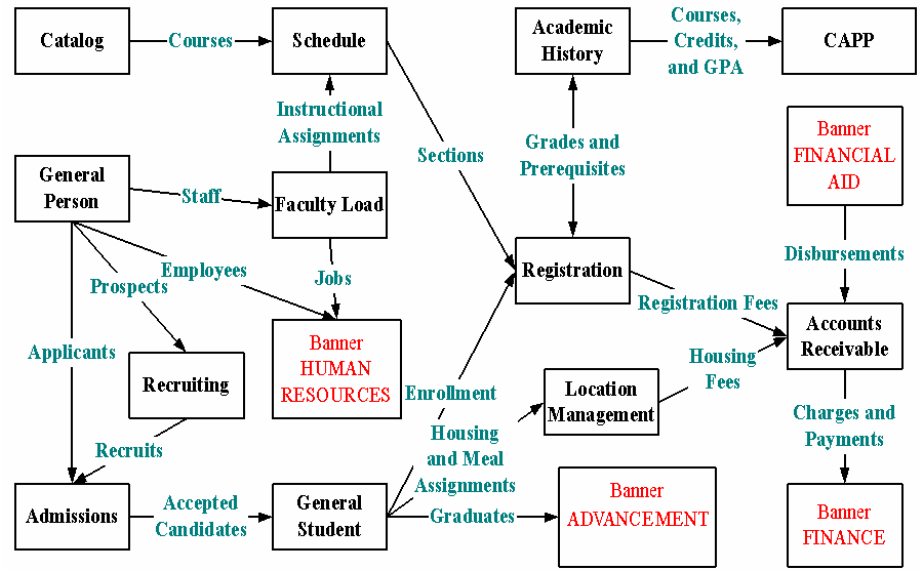
Section Content

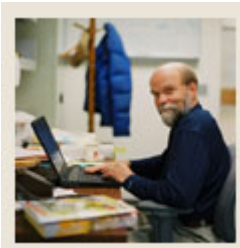
Section K: Location Management

Lesson: Student System Overview

Jump to TOC

Diagram



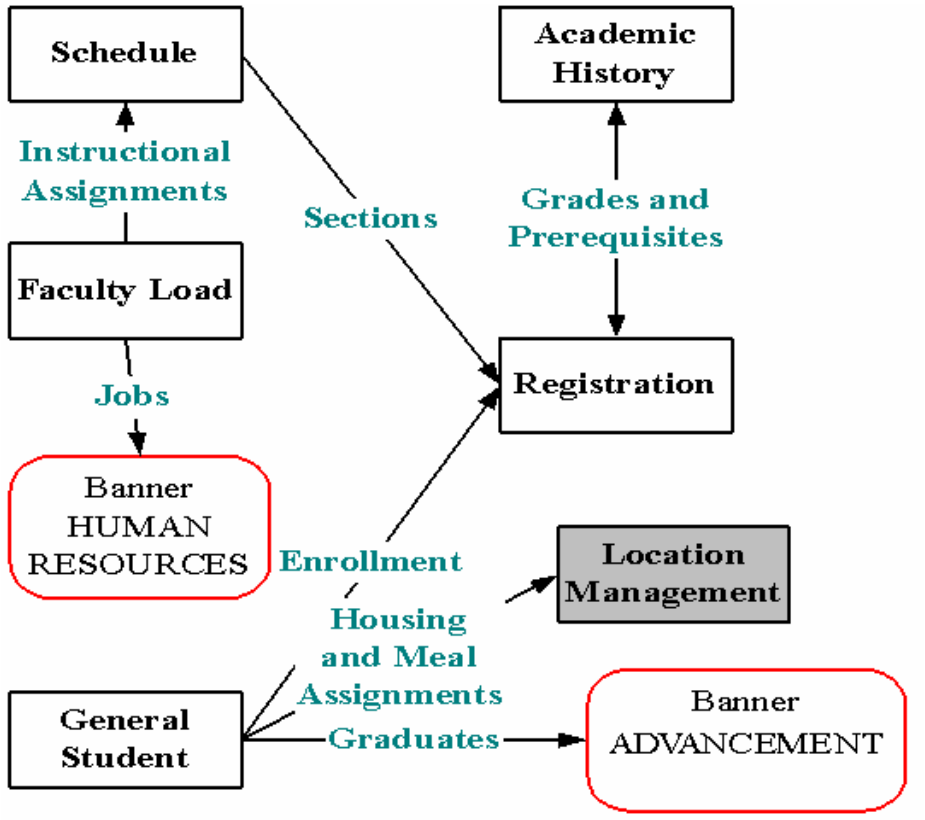


Section K: Location Management

Lesson: Location Management Module

Jump to TOC

Diagram



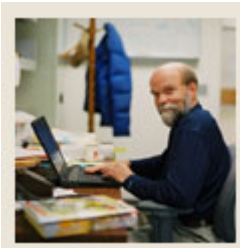
Objectives

Examine/Review

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Overview

- Allows for the definition of the institution's building and room facilities
- Provides a means of assigning rooms for special events
- Provides a listing of available rooms with attributes
- Maintains dormitory, meal plan, phone assignments and assessments

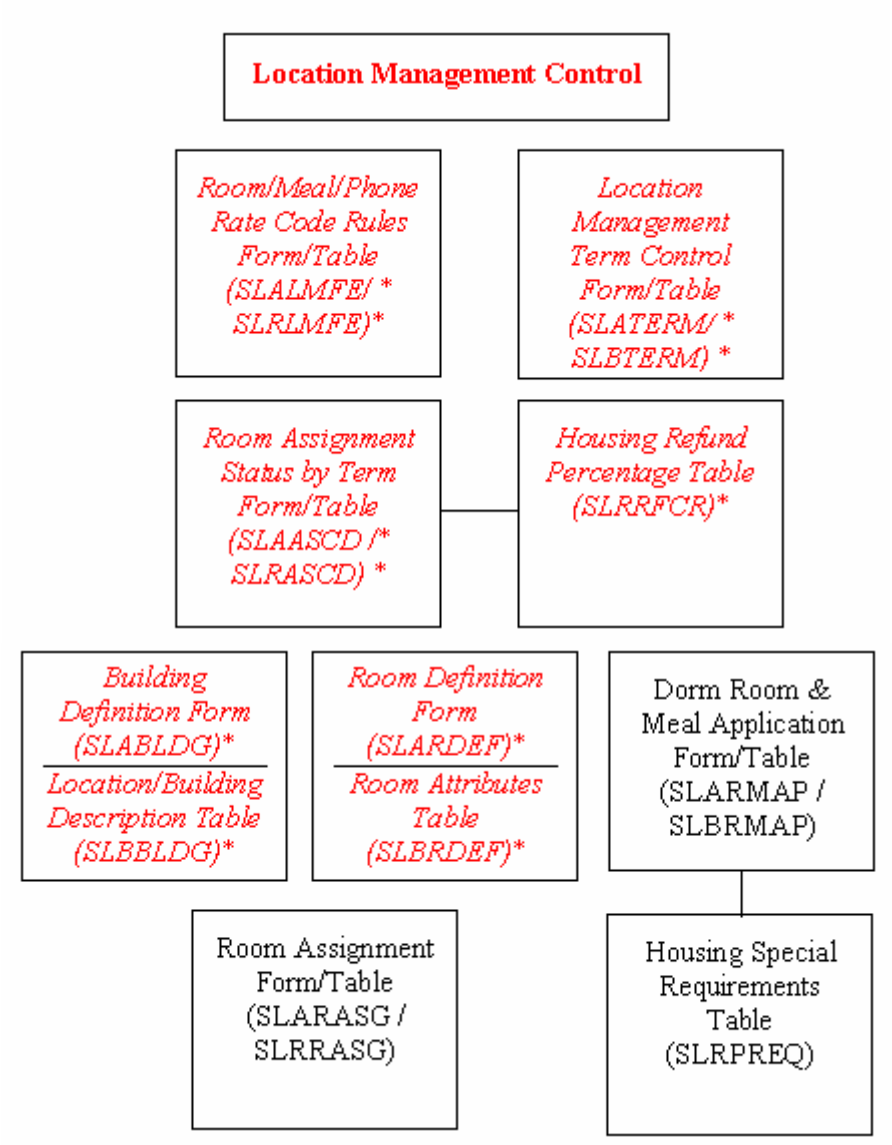


Section K: Location Management

Lesson: Location Management Module (Continued)

Jump to TOC

Overview, continued



SLBBLDG and SLRRDEF are required for scheduling. If you are a residential institution, you will need to populate SLBRMAP and SLRRASG as well.

Major forms and tables

- Building Definition Form (SLABLDG)
  - Table = SLBBLDG
- Room Definition Form (SLARDEF)
  - Table = SLBRDEF

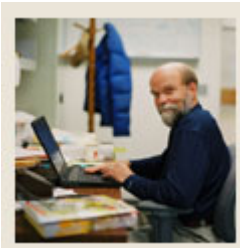
If your institution is residential:

- Dorm Room & Meal Application Form/Table (SLARMAP/SLBRMAP)
  - Room Application Form/Table (SLARASG/SLRRASG)

Major Validation Tables/Forms

STVBLDG STVCAMP
STVRRCD STVHAPS
STVARTP STVTERM
STVASCD TTVDCAT
TTVTAXT (for Canadian use only)
TBBDETC must also be populated for those making residence hall assignments.

AR table TBBDETC must be set up for residential institutions before they can assign residence hall rooms.



Section K: Location Management

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Location Management Module?

```
select table_name
  from all_tables
 where table_name like 'SL%'
```

- What data elements are required?

```
desc slbrdef
```

  - Notice the "NOT NULL" columns.

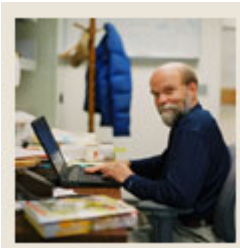
- What are the key fields in SLBRDEF?

```
select column_name
  from all_cons_columns
 where table_name = 'SLBRDEF'
   and constraint_name = 'PK_SLBRDEF';
```

Fields of note

- slrrasg_assess_needed
  This field identifies whether fee assessment is needed for the room assignment. SLRFASM FEE ASSESSMENT PROCESS looks at that field to determine which records should be assessed fees.

- slrrasg_ar_ind
  This field identifies whether the room assignment charges have been processed. SLFRASM updates this field when fees have been assessed.



Section K: Location Management

Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Batch Room, Meal and Phone Assessment Process (SLRFASM)
    - selects records based on slrrasg_assess_needed
    - updates slrrasg_ar_ind in records that were assessed
- Active Housing Assignments Report (SLRHLST)
- Room Assignment Roll Process (SLRROLL) -- Roll like terms -- fall to fall, etc.

Section K: Location Management

Lesson: Other Scripts

Jump to TOC

$BANNER_HOME/student/dbprocs

functions (slf*)

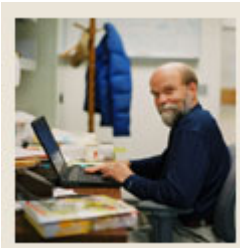
$BANNER_HOME/student/views

views (slv*): slvres0.sql creates view as_residential_life

Some views are used in conjunction with Object:Access method of retrieving data from database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



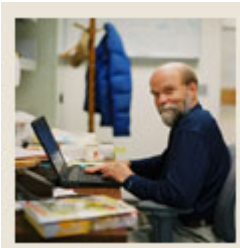
Section K: Location Management

Lesson: Conversion Issues

Jump to TOC

Questions

- Will your institution convert or manually enter Location Management information?
- What Location Management data do you have in your legacy system?
- How do you determine where to put it in SCT Banner?



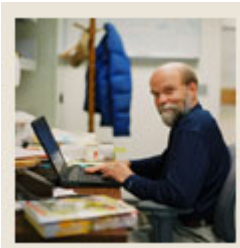
Section K: Location Management

Lesson: Self Check – Location Management Exercise

Jump to TOC

Exercise

Write a simple report that will show the residence hall assignments for a term (prompt the user for the term). On the report, show last name, id, term, building description and room.



Section K: Location Management

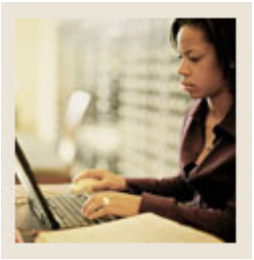
Lesson: Self Check – Location Management Exercise – Answer Key

Jump to TOC

Exercise

Write a simple report that will show the residence hall assignments for a term (prompt the user for the term). On the report, show last name, id, term, building description and room.

```
SELECT  substr(spriden_last_name,1,10),
        spriden_id, stvbldg_desc,
        slrrasg_room_number, slrrasg_term_code
  FROM  spriden, stvbldg, slrrasg
 WHERE  spriden_pidm = slrrasg_pidm
   AND  spriden_change_ind IS NULL
   AND  slrrasg_bldg_code = stvbldg_code
   AND  slrrasg_term_code = '&term';
```



Section L: Schedule

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

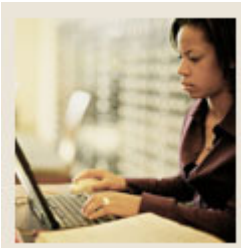
- Describe the role and functions of the Schedule module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

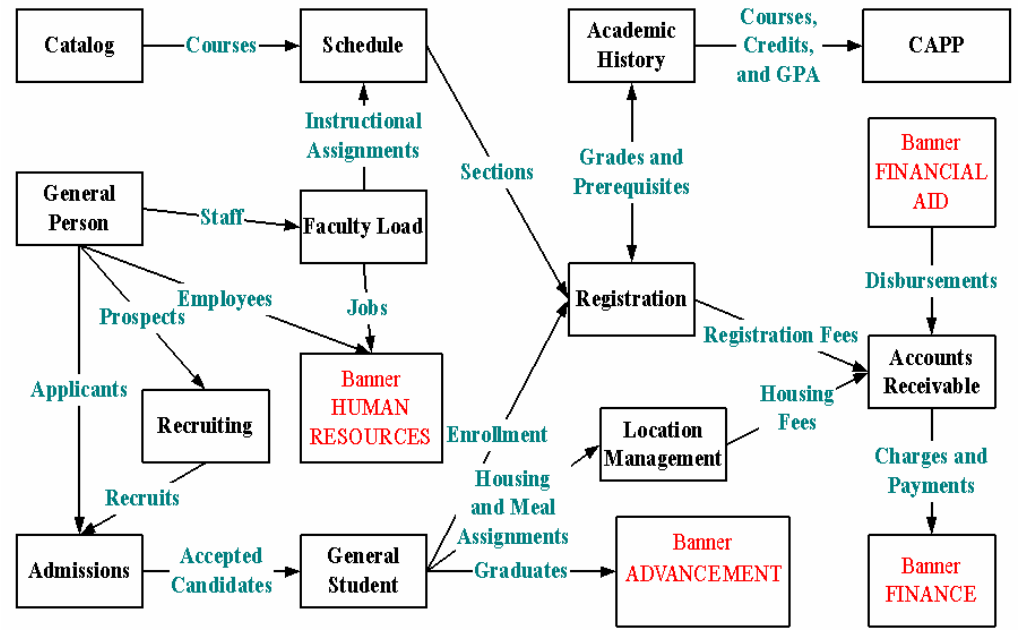


Section L: Schedule

Lesson: Student System Overview

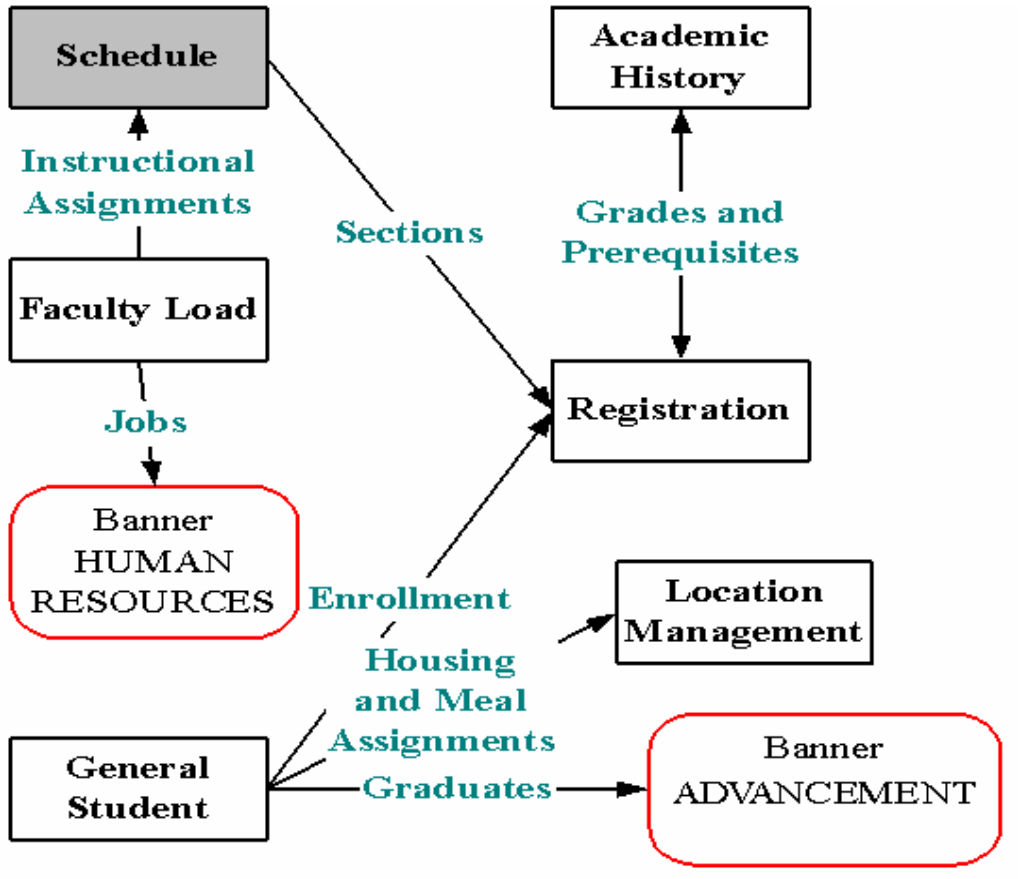
Jump to TOC

Diagram



Note the relationship of the Schedule module to the entire SCT Banner System.
Catalog, General Person, Faculty Load, Location Management and validation tables must be set up before using Schedule.

Diagram



At this point, Catalog has been built, buildings and rooms have been defined, and Faculty are active and available for scheduling in sections. Also, terms (SOATERM) have been defined.

Section L: Schedule

Lesson: Schedule Module (Continued)

Jump to TOC

Objectives

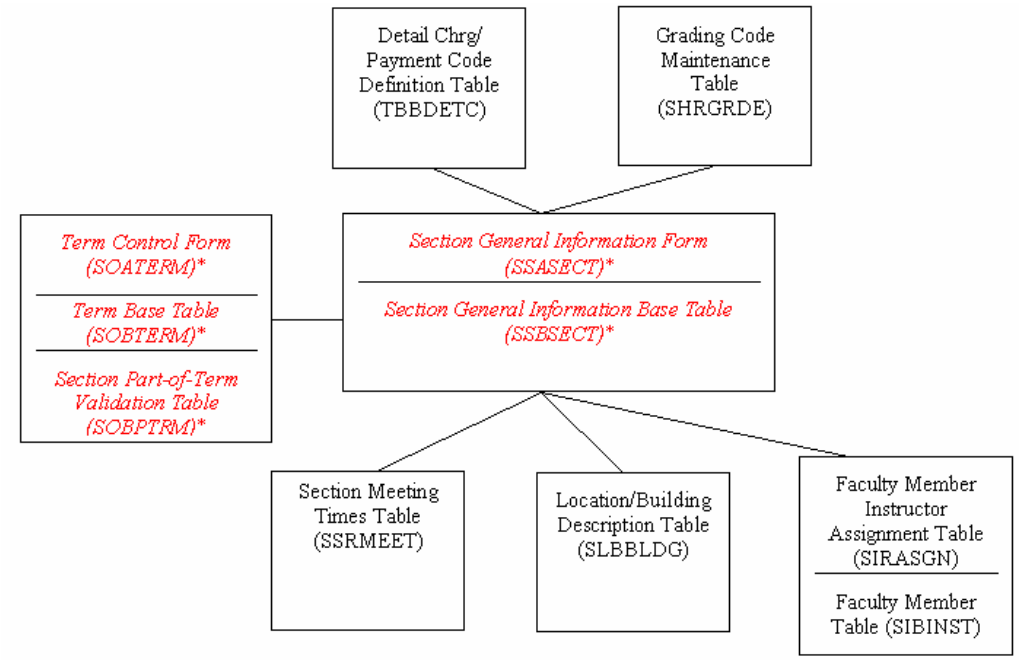
Examine/Review

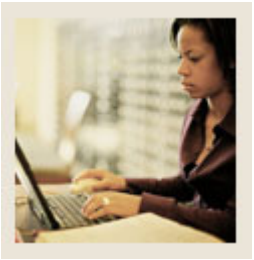
- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Overview

- Build and print a schedule of classes, including term attributes such as date, for each session within a term
- Establish course reference numbers
- Assign instructors to classes
- Schedule classes in rooms
- Allow users to roll the schedule forward to next applicable term to decrease the data entry process

Diagram





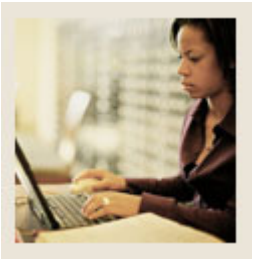
Section L: Schedule

Lesson: Section General Information Form (SSASECT) /Section General Information Base Table (SSBSECT)

Jump to TOC

Purpose

- Used to build and maintain schedule of classes
- Much of the data defaults from Course Catalog (SCBCRSE, etc)
- Connections with AR module through billing hours, tuition waivers



Section L: Schedule

Lesson: Term Control Form (SOATERM)

Jump to TOC

Purpose

- Related tables: Term Control Table (SOBTERM) and Section Part-of-Term Validation Table (SOBPTRM)
- Used to set up controls for each term's schedule, registration, and fee assessment

CRN oneup

Before building the Schedule for a term, a beginning CRN must be set (CRN Oneup).

There are more details about SOATERM and the underlying tables (SOBTERM and SOBPTRM) in Lesson 14 (Registration).

Major Validation Tables/Forms

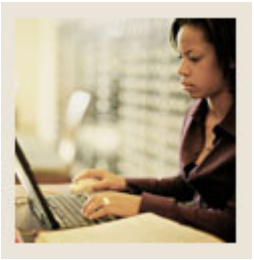
- Term Code Validation Form/Table (STVTERM)
- Level Code Validation Form/Table (STVLEVL)
- Part of Term Code Validation Form/Table (STVPTRM)
- Campus Code Validation Form/Table (STVCAMP)
- Section Status Code Validation Form/Table (STVSSTS)
- Schedule Type Code Validation Form/Table (STVSCHD)
- Subject Code Validation Form/Table (STVSUBJ)
- Grading Mode Code Validation Form/Table (STVGMOD)
- Day of Week Code Validation Form/Table (STVDAYS)
- Detail Charge/Payment Code Definition Table (TBBDETC)

TBBDETC must be set up if you populate SSRFEES table through the SSADETL form, which is not required.

Other Forms/Tables

- Section Meeting Times Table (SSRMEET)
- Location/Building Description Table (SLBBLDG)
- Faculty Information Form/Table (SIAINST/SIBINST)
- Faculty Assignment Form/Table (SIAASGN/SIRASGN)

SSASECT uses SSRMEET table to store meeting times and buildings (SSRMEET_BLDG_CODE). Building information is built in the SLABLDG form, and faculty information is built in SIAINST and SIAASGN forms.



Section L: Schedule

Lesson: SLQMEET and SSAMATX

Jump to TOC

SLQMEET

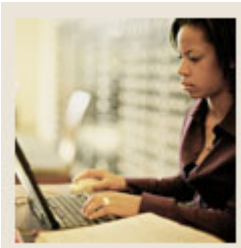
Available Classroom Query Form (SLQMEET)

- Only accessible through SSASECT

SSAMATX

Building/Room Schedule Form (SSAMATX)

- Accessible through menu, direct access, other form (SSASECT)



Section L: Schedule

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Schedule Module?

```
select table_name
  from all_tables
 where table_name like 'SS%'
```

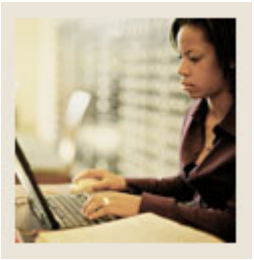
- What data elements are required?

```
desc ssbsect
```

  - Notice the "NOT NULL" columns.

- What are the key fields in ssbsect?

```
select column_name
  from all_cons_columns
 where table_name = 'SSBSECT'
   and constraint_name = 'PK_SSBSECT';
```



Section L: Schedule

Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Schedule Purge Process (SSPSCHD)
- Term Roll Process (SSRROLL) -- Roll like terms -- Fall to Fall
- Class Schedule Report (SSRSECT)
- Scheduled Section Tally (SSRTALY)

Rolling of Room Assignments should be of “like” terms -- fall to fall, spring to spring, etc.

See Student Technical Reference Manual, Chapter 10, for additional information on Student Module reports and processes.

$BANNER_HOME/student/dbprocs

functions (ssf*)

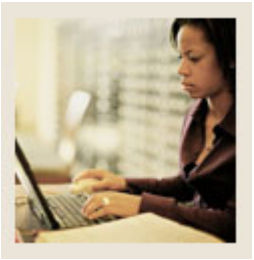
$BANNER_HOME/student/views

views (ssv*): ssvsec0.sql creates view as_catalog_schedule

Some views are used in conjunction with Object:Access method of retrieving data from database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



Section L: Schedule

Lesson: Conversion Issues

Jump to TOC

Questions

- Will Schedule data be converted or entered manually by the users?
- What Schedule data do you have in your legacy system?
- How do you determine where to put it in SCT Banner?

Note: It is generally not recommended to convert a schedule, but to enter it manually instead.

Schedule creation

After Catalog data is entered or converted, the Schedule can be created.

If it is decided that conversion of legacy data is required, it may be advisable to manually enter several courses into SSASECT to ensure that all of the Catalog data that should "default" to the sections does so properly.

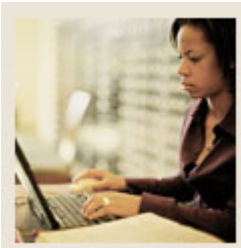
Section L: Schedule

Lesson: Self Check – Schedule Exercise

Jump to TOC

Exercise

Write a query that returns full name, id, crn, subject code, course number, section number, course title and term code for all faculty members teaching any English course. Prompt the user for the term.



Section L: Schedule

Lesson: Self Check – Schedule Exercise – Answer Key

Jump to TOC

Exercise

Write a query that returns full name, id, crn, subject code, course number, section number, course title and term code for all faculty members teaching any English course. Prompt the user for the term.

```
select  substr(spriden_last_name, 1,15) ||  ', ' ||
          substr(spriden_first_name,1,15),
        spriden_id, ssbsect_crn, ssbsect_subj_code,
        ssbsect_crse_numb, a.scbcrse_title,
        ssbsect_seq_numb, a.scbcrse_eff_term,
        ssbsect_term_code
 from   spriden, ssbsect, scbcrse a, sirasgn
where   sirasgn_pidm = spriden_pidm
  and   spriden_change_ind is null
  and   sirasgn_crn = ssbsect_crn
  and   sirasgn_term_code = ssbsect_term_code
  and   ssbsect_subj_code = 'ENGL'
  and   ssbsect_term_code = '&term'
  and   ssbsect_subj_code = a.scbcrse_subj_code
  and   ssbsect_crse_numb = a.scbcrse_crse_numb
  and   a.scbcrse_eff_term =
       (select    max(b.scbcrse_eff_term)
                from    scbcrse b
             where      b.scbcrse_subj_code =
                        ssbsect_subj_code
            and     b.scbcrse_crse_numb =
                        ssbsect_crse_numb
               and      b.scbcrse_eff_term <=
                        ssbsect_term_code);
```


Section M: General Student

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the General Student module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

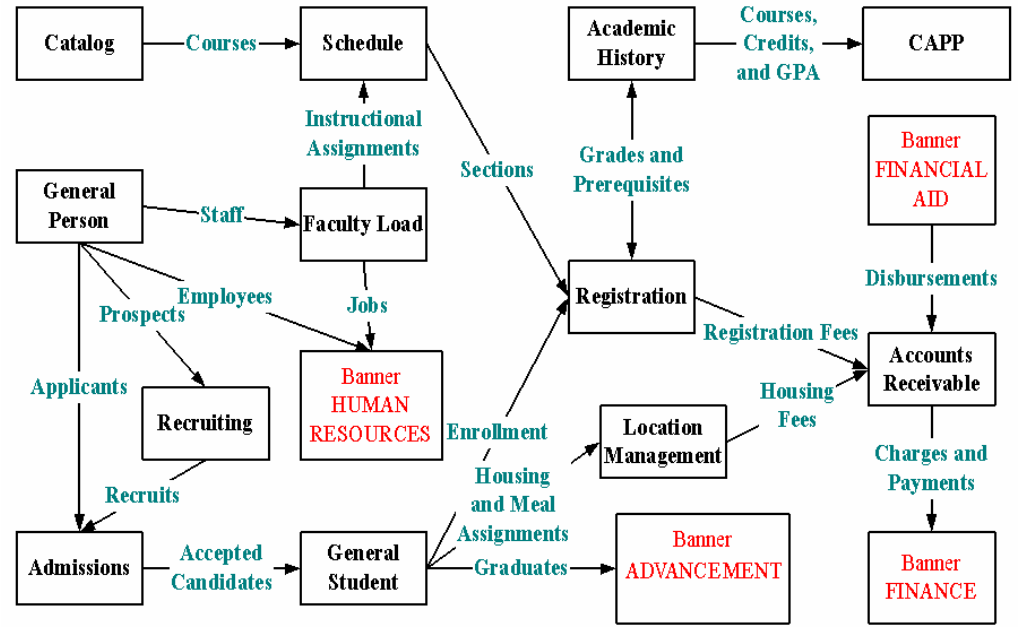
Section Contents

Section M: General Student

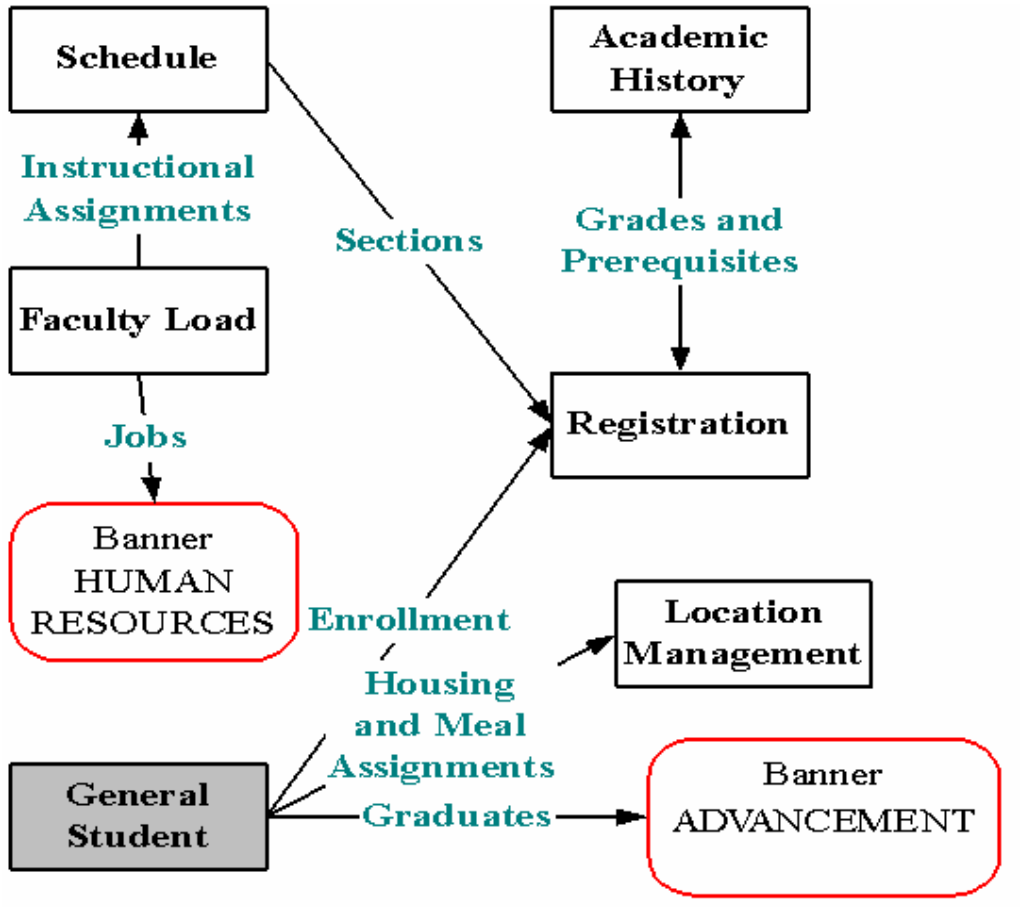
Lesson: Student System Overview

Jump to TOC

Diagram



Diagram



General student records have been created via the traditional method through Admissions (using SAAADMS) or by quick admitting (via SAAQUIK).

SIBINST table is populated with faculty data and advisor flag.

Objectives

Examine/Review

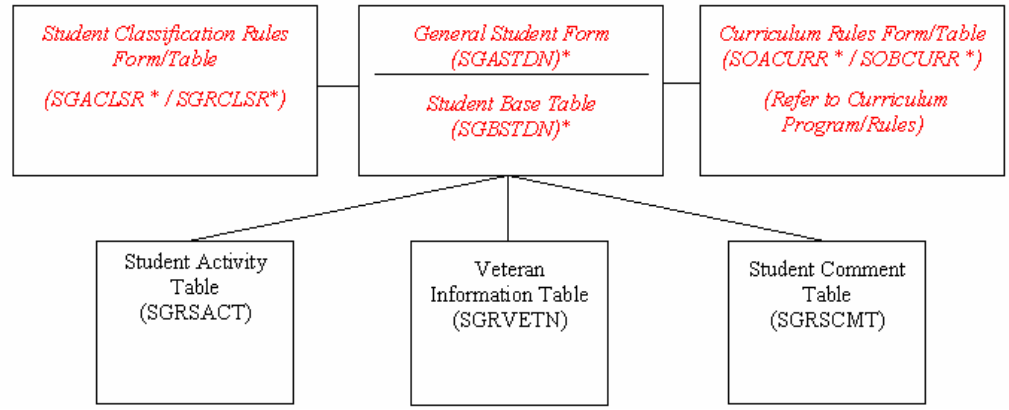
- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Section M: General Student

Lesson: General Student Module (Continued)

Jump to TOC

Overview



Major Form/Table

General Student Form (SGASTDN) / Student Base Table (SGBSTDN)

- Used to maintain current and historical information about a student

SGASTDN form also utilizes the following tables:

- Student Activity Table (SGRSACT)
- Veteran Information Table (SGRVETN)
- Student Comment Table (SGRSCMT)

Notes

- Data in form is similar to Recruiting and Admissions
- No change can be made in SGASTDN for a term if the student has registered for that term. Changes would need to originate in SFAREGS
- Must have a SPRIDEN record
- Must know how far back to go with academic history
- A student must have a SGBSTDN record before the registration process is allowed on that student's record
- Must have correct flags on stvmajr (major, minor, concentration)
- Must have SGBSTDN for records if you are converting Academic History

Rule Forms/Tables

Student Classification Rules Form/Table (SGACLSR/SGRCLSR)

- Used to establish classification rules based on range of credit hours entered and student attributes

Curriculum Rules Form/Table (SOACURR/SOBCURR)

- Refer to Lesson 5 (Curriculum/Program Rules)
- If rules are to be used, indicator will be 'ON' for General Student

Major Validation Tables/Forms

- Term Code Validation Form/Table (STVTERM)
- Residence Code Validation Form/Table (STVRESD)
- Level Code Validation Form/Table (STVLEVL)
- Student Status Code Validation Form/Table (STVSTST)
- Campus Code Validation Form/Table (STVCAMP)
- Class Code Validation Form/Table (STVCLAS)
- College Code Validation Form/Table (STVCOLL)
- Degree Level Code Validation Form/Table (STVDLEV)
- Degree Code Validation Form/Table (STVDEGC)
- Degree Award Category Code Validation Form/Table (STVACAT)
- Major, Minor, Concentration Code Validation Form/Table (STVMAJR)
- Student Type Code Validation Form/Table (STVSTYP)

Most of the validation tables have been used in Recruiting and Admissions. When the student record is created, the data from Admissions (SARADAP) will default into SGBSTDN.

Additional Information

- SGRADVR - Multiple advisors
- SGRSPRT - Sports
- SGRCHRT - Cohorts
- SGRSATT- Attributes
- SGRDISA - Disability Services

Rules

SOBCURR -- major, program, department, etc.
SGRCLSR -- Student classification rules

Questions

- What tables are part of the General Student Module?

```
select table_name
  from all_tables
 where table_name like 'SG%'
```

- What data elements are required?

```
desc sgbstdn
```

  - Notice the "NOT NULL" columns.

- What are the key fields in sgbstdn?

```
select column_name
  from all_cons_columns
 where table_name = 'SGBSTDN'
   and constraint_name = 'PK_SGBSTDN';
```

Section M: General Student

Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Hold Purge (SGPHOLD)
- General Student Purge (SGPSTDN)
- Student Report (SGRSTDN)

See Student Technical Reference Manual, Chapter 10, for additional information on Student Module reports and processes

$BANNER_HOME/student/dbprocs

functions (sgf*)

$BANNER_HOME/student/views

views (sgv*): sgvstd0.sql creates view as_student_data

Some views are used in conjunction with Object:Access method of retrieving data from database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.

Section M: General Student

Lesson: Conversion Issues

Jump to TOC

Questions

- What General Student data do you have in your legacy system?
- How far back do you wish to go with your data conversion?
- How do you determine where to put legacy data in Banner?
- Must have a student record with sgbstdn_term_code_eff = first term of history

Section M: General Student

Lesson: Self Check – General Student Exercise

Jump to TOC

Exercise

Write a query that returns the student's full name, id, advisor's name, major code, and residency code from the current student record.

Section M: General Student

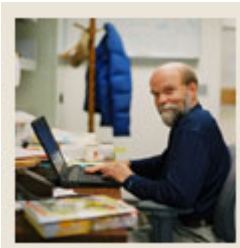
Lesson: Self Check – General Student Exercise – Answer Key

Jump to TOC

Exercise

Write a query that returns the student's full name, id, advisor's name, major code, and residency code from the current student record.

```
select substr(S.spriden_last_name, 1,15) || ', ' ||
          substr(S.spriden_first_name,1,15),
       S.spriden_id,  sgbstdn_majr_code_1,
       sgbstdn_resd_code,
       substr(A.spriden_last_name, 1,15)
  from spriden S, spriden A, sgbstdn, sgradvr
 where sgradvr_pidm = S.spriden_pidm
   and S.spriden_change_ind is null
   and sgradvr_term_code_eff =
         (SELECT MAX(I.SGRADVR_TERM_CODE_EFF)
            FROM SGRADVR I
           WHERE I.SGRADVR_TERM_CODE_EFF <= '&term'
             AND SGRADVR_PIDM = I.SGRADVR_PIDM)
   and sgradvr_pidm = sgbstdn_pidm
   and sgbstdn_term_code_eff =
         (SELECT MAX(B.SGBSTDN_TERM_CODE_EFF)
            FROM SGBSTDN B
           WHERE B.SGBSTDN_TERM_CODE_EFF <= '&term'
             AND SGBSTDN_PIDM = B.SGBSTDN_PIDM)
   and sgradvr_advr_pidm = A.spriden_pidm
   and A.spriden_change_ind is null
```



Section N: Accounts Receivable

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

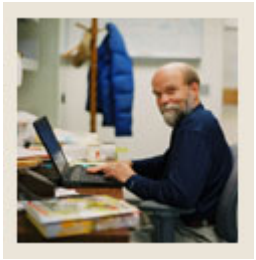
- Describe the role and functions of the Accounts Receivable module

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

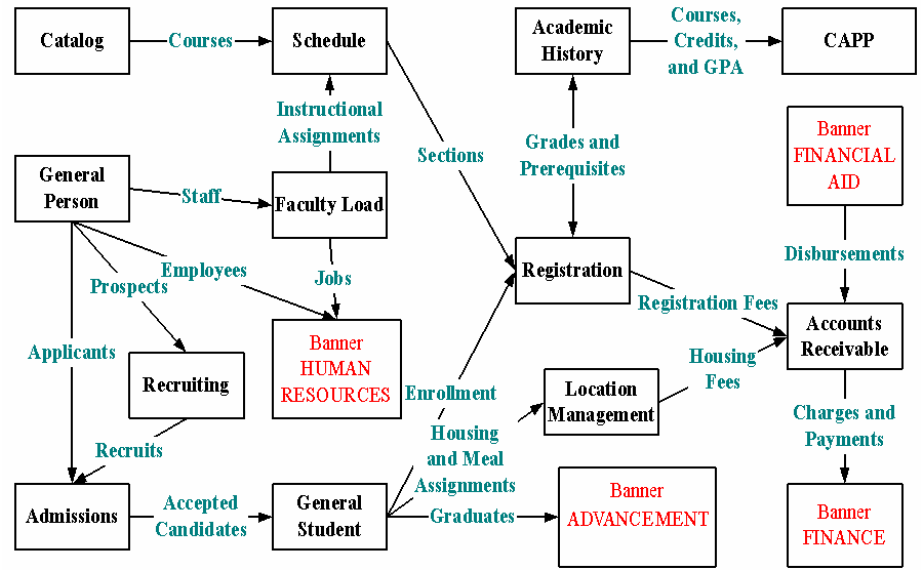


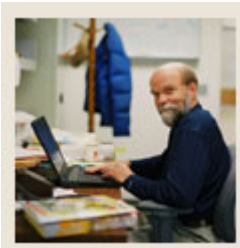
Section N: Accounts Receivable

Lesson: Student System Overview

Jump to TOC

Diagram



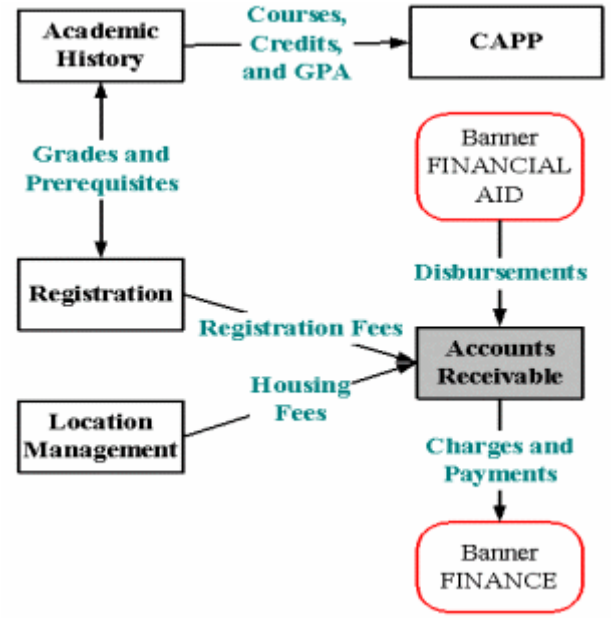


Section N: Accounts Receivable

Lesson: Accounts Receivable Module

Jump to TOC

Diagram



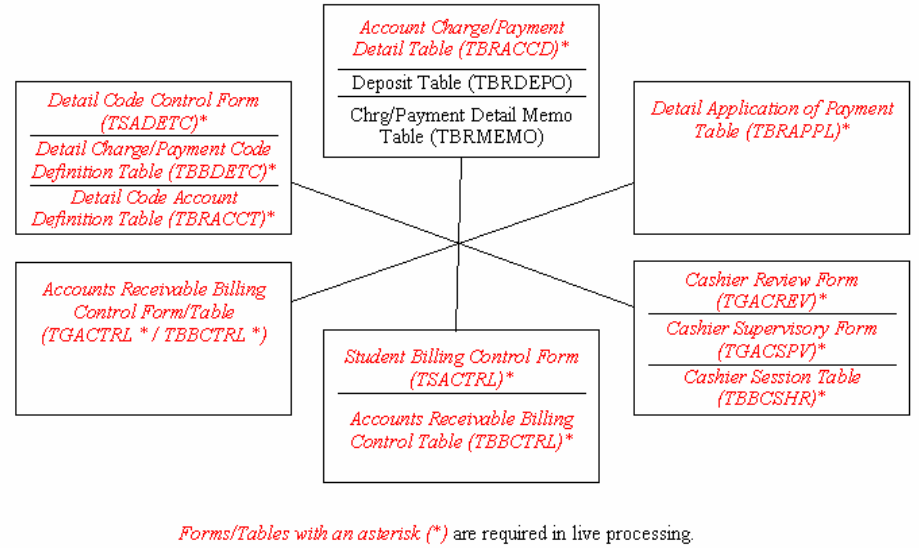
For additional process flow and database schematics, refer to the Student Technical Reference Manual, Chapters 8 and 9.

Objectives

Examine/Review

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

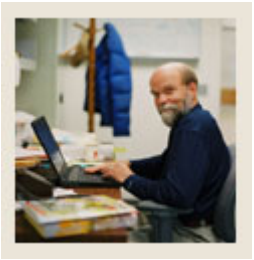
Diagram



Note: For conversion, only TBRACCD needs to be populated. You will need to set up TBBDETC and TBRACCT first, along with other validation tables.

Major Validation Tables/Forms

- Bill Code Validation Form/Table (TTVBILL)
- Detail Category Code Validation Form/Table (TTVDCAT)
- Delinquency Code Validation Form/Table (TTVDELI)
- Deposit Type Code Validation Form/Table (TTVDTYP)
- Payment Type Code Validation Form/Table (TTVPAYT)
- Charge/Payment Source Code Validation Form/Table (TTVSRCE)
- Term-Based Designator Validation Form/Table (TTVTBDS)
- Tax Type Code Validation Form/Table (TTVTAXT)
  (for Canadian Inst. only)



Section N: Accounts Receivable

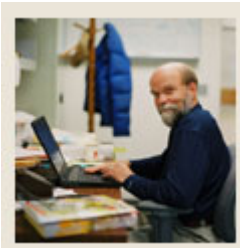
Lesson: Accounts Receivable Billing Control Form (TGACTRL) / Student Billing Control Form (TSACTRL)

Jump to TOC

GACTRL / TBBCTRL

The Billing Control Table (TBBCTRL) is used with each of these forms.

These forms are required for conversion.



Section N: Accounts Receivable

Lesson: Detail Code Control Form (TSADETC)

Jump to TOC

TSADETC

- Enter detail code information
- Establish payment priorities used in the Application of Payment Process (TGRAPPL)

Priority

- Priority is an algorithm-determined 3-digit code that determines a payment hierarchy
- `select distinct tbbdetc_priority from tbbdetc` will display which priorities have been set
- "0" is a wildcard in priority codes -- it will pay anything
    - TBBDETC, TBRACCT tables
    - Establishes interface with Finance package (SCT Banner Finance or a third-party package)
    - Set up fund codes, account numbers
    - Required for conversion

TSASBRL

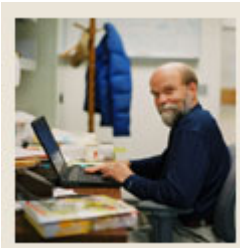
Schedule/Bill Rules Form (TSASBRL)

This form sets up parameters used in the Student Billing Statement Process (TSRCBIL). TBBSBRL is its related table.

TSATBDS

Term-based Designator Rules Form (TSATBDS)

This form allows users to establish relationships between term codes and term-based AR designators. TBBTBDS is its related table.



Section N: Accounts Receivable

Lesson: TGACREV/TGACSPV

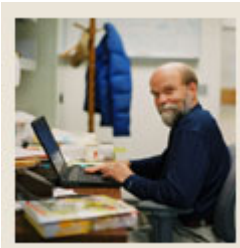
Jump to TOC

TGACREV

Cashier Session Review Form (TGACREV)
This form is used to review all charge or payment activity for a specific session. The Cashier Session Table (TBBCSHR) is its related table.

TGACSPV

Cashier Supervisory Form (TGACSPV)
This form is used to display all active and inactive cashiering sessions on the system. The Cashier Session Table (TBBCSHR) is its related table.



Section N: Accounts Receivable

Lesson: Student Account Detail Form (TSADETL)

Jump to TOC

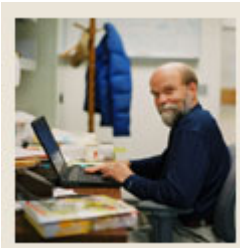
TSADETL

This form holds account detail by detail code.

- Major Table = Student Account Detail Review Table (TBRACCD)

The form also displays deposits, memos and comments.

- Tables = Deposit Table (TBRDEPO), Chrg/Payment Detail Memo Table (TBRMEMO), Comment Table (TBRCMNT)



Section N: Accounts Receivable

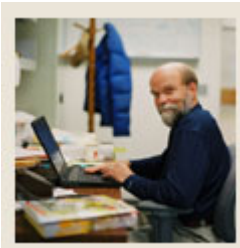
Lesson: Student Account Detail Review Form (TSAAREV)

Jump to TOC

TSAAREV

This form is used to review and enter information about an account. It presents an online view of each transaction by term.

The Student Account Detail Review Table (TBRACCD) is its related table, which is also accessed from SFAREGS.



Section N: Accounts Receivable

Lesson: Student Payment Form (TSASPAY)

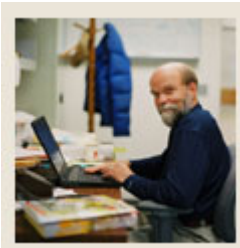
Jump to TOC

TSASPAY

This form is used to determine status of student's account for a term. It can be used to accept charges and disburse Financial Aid.

This form is affected by changes in TSADETL, SFAREGS, SLAMASG and other forms.

The Student Account Detail Review Table (TBRACCD) is its related table.



Section N: Accounts Receivable

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Accounts Receivable Module?

```
select table_name
  from all_tables
 where table_name like 'T%'
```

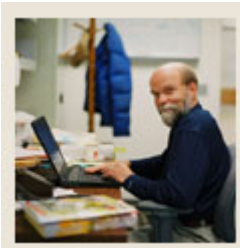
- What data elements are required?

```
desc tbraccd
```

  - Notice the "NOT NULL" columns.

- What are the key fields in tbraccd?

```
select column_name
  from all_cons_columns
 where table_name = 'TBRACCD'
   and constraint_name = 'PK_TBRACCD';
```



Section N: Accounts Receivable

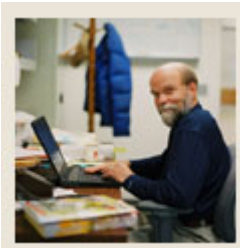
Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Application of Payment Process (TGRAPPL)
- Accounting Feed Process (TGRFEED)
- Student Billing Statement Process (TSRCBIL)
- Account Receipt Process (TGRRCPT)
- Miscellaneous Receipt Process (TGRMISC)

Refer to the Student Technical Reference Manual, Chapter 10, for a complete list of Accounts Receivable reports and processes.



Section N: Accounts Receivable

Lesson: Application of Payments Process (TGRAPPL)

Jump to TOC

Overview

This process:

- Applies payments to charges for accounts based on priority (tbbdetc_priority)
- Creates correct accounting entries to be fed by TGRFEED process
- Gets other rules from TBBCTRL table

The process is a C program run from Job Submission. Results are visible on the Application of Payment Review Form (TSIAPPL), and it populates the Detail Application of Payment Table (TBRAPPL).

Refer to the matrix in the Student Technical Manual, p. 10-18, for stats about this process.

Application of payments

This chart shows how the process TGRAPPL applies payments, using a series of transactions in the Student Account Detail Review Table (TBRACCD).

- Detail codes:
  - ACTF = activity fee
  - T101 = tuition
  - CHEK = Check
  - AMEX = American Express Payment
- Detail codes are defined in TBBDETC as charge or payment codes.

TBRACCD table	**TGRAPPL applies:**	**BALANCES**	**TBRAPPL table**
$35 ACTF [tran num 1]	**$7000 AMEX pmt [*tbracct tran num 3*]**	**chg. = $0 pmt = $6965**	**tbrappl_chg_tran_number 1 tbrappl_pay_tran_number 3 tbrappl_amount = $35**
$7500 T101 [*tran num 2*]	**$6965 pmt bal [bal of AMEX pmt tbraccd tran num 3)**	**chg. = $535 pmt. = $0**	**tbrappl_chg_tran_number 2 tbrappl_pay_tran_number 3 tbrappl_amount = $6965**
$7000 AMEX [*tran num 3*]	**applied to charges in trans 1 and 2**		
$535 CHEK [*tran num 4*]	**$535 CHEK pmt**	**chg. = $0 pmt. = $0**	**tbrappl_chg_tran_number 2 tbrappl_pay_tran_number 4 tbrappl_amount = $535**

Section N: Accounts Receivable

Lesson: Accounting Feed Process (TGRFEED)

Jump to TOC

Overview

This process takes all applications of payment, deposits, miscellaneous transactions and account detail transactions from finalized cashiering sessions. Based on the accounts built, it creates a file of accounting detail records (GURFEED) that interface the Accounts Receivable module with the institution's financial accounting system, along with refund and check information (GURAPAY).

Source tables are updated to show that those records have been fed into the General Ledger.

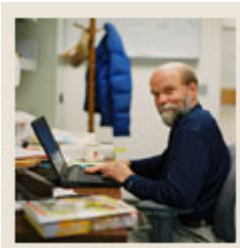
Output

The process produces a report that details debit and credit entries by account number.

TGRFEED uses data from TBRAPPL, TBRDEPO, TBRMISD and TBRACCD, and refers to the TBRACCT, TBBDETC and TBBCTRL tables for distribution and detail information.

TGRFEED goes to TSADETC (TBRACCT) to get accounting distribution codes and rules.

TGRFEED is a C program run from Job Submission.



Section N: Accounts Receivable

Lesson: Student Billing Statement Process (TSRCBIL)

Jump to TOC

Overview

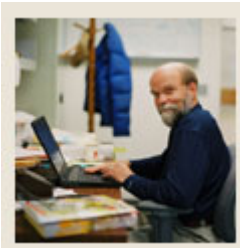
- In Invoicing Mode, TSRCBIL prints invoices and estimates credits based on current charges.
- In Statement Mode, TSRCBIL calculates credits, prints bills, updates accounts with billed and due dates, applies credits and begins the aging process.

Schedules may also be printed via this job.

Rule parameters for TSRCBIL are set on the Bill Selection Parameters Window of the Schedule/Bill Rules Form (TSASBRL) (TBBSBRL table).

The process updates AR indicators in SLRMASG, SLRPASG, SLRRASG and SFBETRM.

TSRCBIL is a C program run from Job Submission, which can be run in sleep/wake mode.



$BANNER_HOME/student/arsys

functions (t*f*) ex: tofbala.sql

$BANNER_HOME/student/views

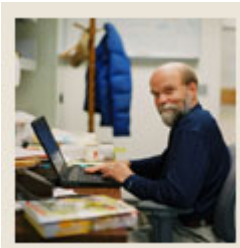
views (t*v*): tovbal0.sql creates view at_ar_history_by_balance

Some views are used in conjunction with theObject:Access method of retrieving data from the database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

This one does not use the GTVSDAX table; not all layered views refer to GTVSDAX. It is used as a crosswalk table to “spread out” repeating table row values into columns for easier reporting.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



Section N: Accounts Receivable

Lesson: Conversion Issues

Jump to TOC

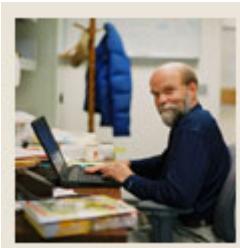
Conversion issues

Balance Forward

- Set up controls on TSACTRL
- Populate TBBDETC -- Detail Charge/Payment Code Definition Table
- Populate TBRACCT -- Detail Code Account Definition Table (fund and account codes)

TBBDETC and TBRACCT must be populated before fee assessment can take place.

Refer to Student Technical Ref. Manual, pp. 5-12 -- 5-15.



Section N: Accounts Receivable

Lesson: Self Check – Accounts Receivable Exercises

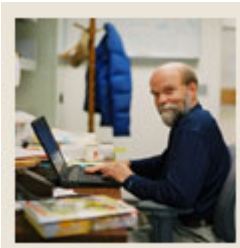
Jump to TOC

Exercise 1

Find all columns in the Accounts Receivable module associated with detail codes.

Exercise 2

Write a simple report that will show full name, id, term and balance from the student account detail table for a given term for those students with a balance > 0. Prompt user for term.



Section N: Accounts Receivable

Lesson: Self Check – Accounts Receivable Exercises – Answer Key

Jump to TOC

Exercise 1

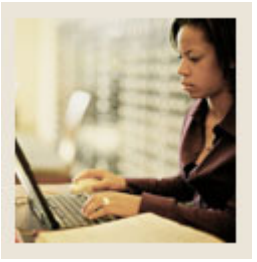
Find all columns in the Accounts Receivable module associated with detail codes.

```
select owner,
       table_name,
       column_name,
       comments
  from all_col_comments
 where owner = 'TAISMGR'
   and column_name like '%DET%_CODE'
```

Exercise 2

Write a simple report that will show full name, id, term and balance from the student account detail table for a given term for those students with a balance > 0. Prompt user for term.

```
select substr(spriden_last_name,1,12),
       substr(spriden_first_name,1,10),
       spriden_id,
       sum(tbraccd_balance)
 from  spriden, tbraccd
where  spriden_pidm = tbraccd_pidm
  and  spriden_change_ind is null
  and  tbraccd_balance > 0
  and  tbraccd_term_code= '&term'
group  by spriden_last_name,
          spriden_first_name,
          spriden_id
order  by spriden_last_name
```



Section O: Registration

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Registration module

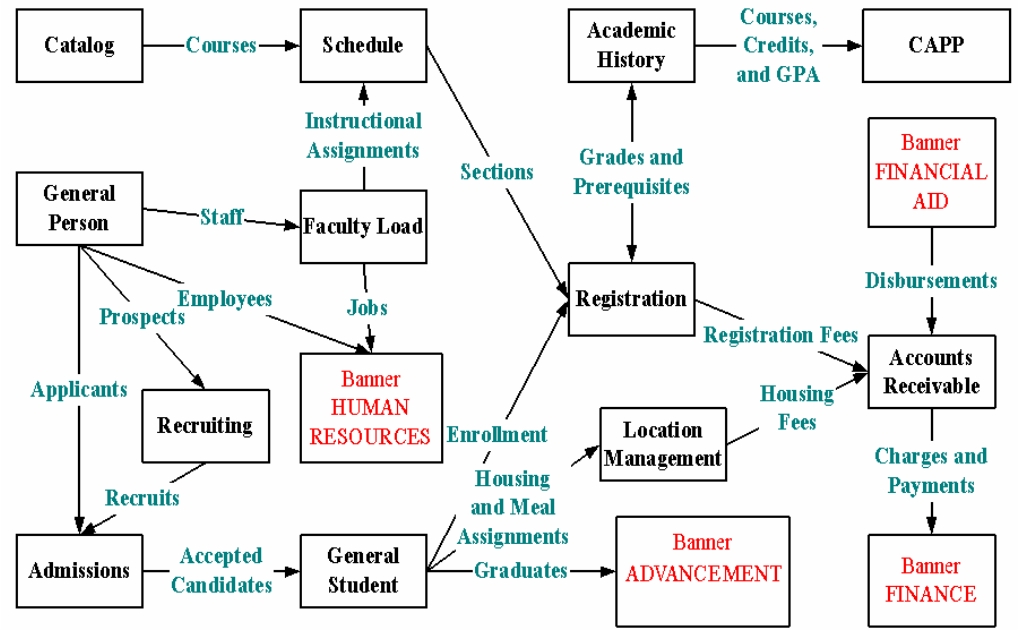
Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Contents

Diagram

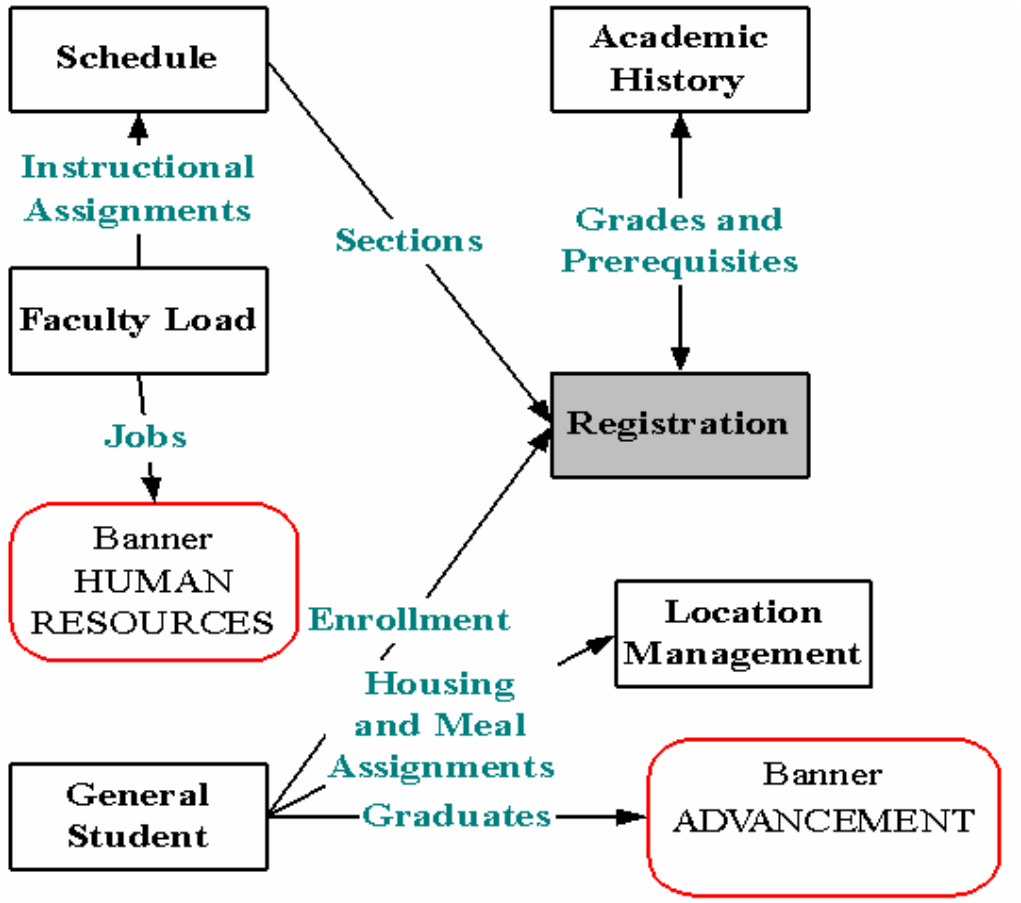


Notice the placement of the Registration module.

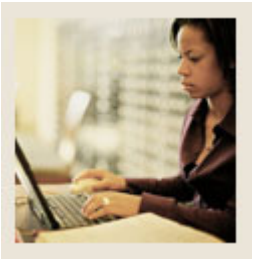
Catalog, Schedule, General Person, Admissions, General Student, Faculty Load, Location Management and AR have all been implemented. Registration is connected to all of these modules.

Accounts Receivable appears “after” Registration in this chart to show the flow of activity; however, the implementation order must have AR in place before Registration can occur, so that fees from Registration may be assessed.

Diagram



- Catalog is built
- Buildings and rooms are defined
- Schedule of classes is built
- Faculty Information is loaded
- Student records are active
- Accounts Receivable has been set up
- Current students may now register for classes



Section O: Registration

Lesson: Registration Module (Continued)

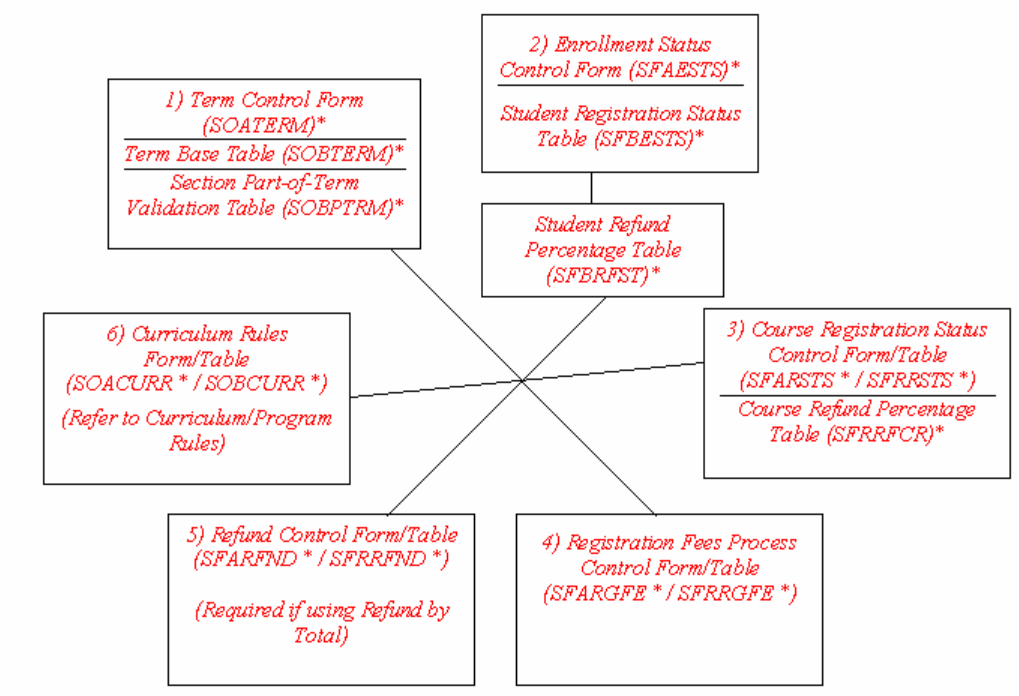
Jump to TOC

Objectives

Examine/Review

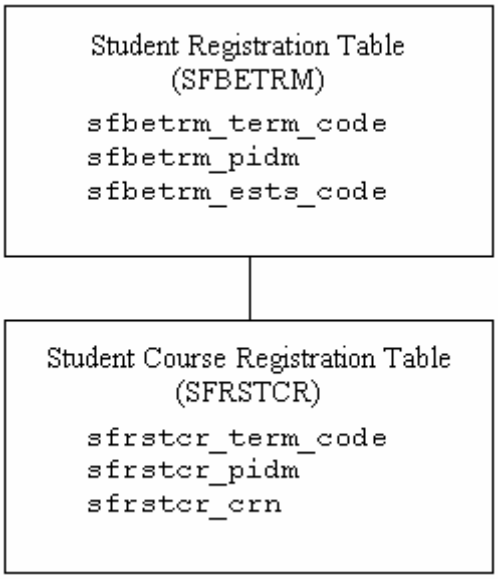
- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Diagram

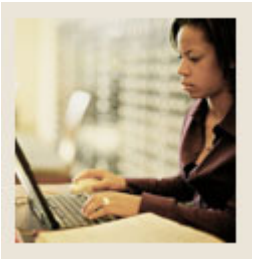


Note: Registration is connected directly to Student AR through fee assessment.

Registration tables



For reporting purposes, these two tables are most important in Registration Module.



Forms and tables

Curriculum Rules Form (SOACURR)

- SOBCURR is the related table

Term Control Form (SOATERM)

- SOBTERM & SOBPTRM are the related tables
- Set Online Fee Assessment
- Error Checking & Severity Level, etc.
- Rules forms must be set up before registration can occur

Enrollment Status Control Form (SFAESTS)

- SFBESTS & SFBRFST are the related tables
- Enrollment status codes are maintained on the Enrollment Status Code Validation Form/Table (STVESTS)
- Also related: Student Refund Percentage Table (SFBRFST)

Course Registration Status Control Form (SFARSTS)

- SFRRSTS & SFRRFCR are the related tables
- Course registration status codes are maintained on the Course Registration Status Code Validation Form/Table (STVRSTS)
- Also related: Student Refund Percentage Table (SFBRFST)

Registration Fees Process Control Form (SFARGFE)

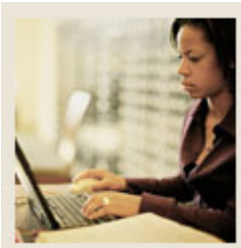
- SFRRGFE is the related table

Refund Control Form (SFARFND)

- SFRRFND is the related table
- Connection to AR

Major Validation Tables/Forms

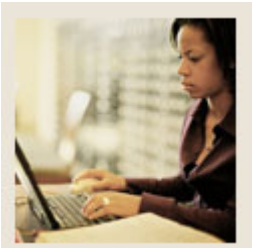
- **All** that were mentioned in the previous modules
- For a complete list, refer to the Student User Manual.



Major Form/Tables

Student Course Registration Form (SFAREGS)

- Mechanism for registering students
- Upon opening SFAREGS for the first time in a Banner session, you go directly to SOADEST (printer choice form for sleep/wake processes)
    - SFBETRM - Table containing Registration Status
    - SFRSTCR - Table containing Course Registrations
- A registration record (SFBETRM) may be created without courses. Be aware of this when writing reports using SFBETRM. (The Registered, Not Paid Process (SFRRNOP), when run in update mode, will "clean out" these records. Alternatively, a script written to delete unwanted SFBETRM records may be necessary.)



Section O: Registration

Lesson: Fee Assessment

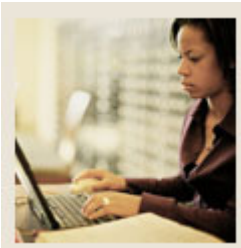
Jump to TOC

Fee Assessment

Fee Assessment may be performed online (via SOATERM), or in batch via the Batch Fee Assessment Process (SFRFASM).

Fee Assessment uses rules built in Catalog, Schedule and Registration Modules. It always writes a record to the Registration Fee Assessment View Collector Table (SFRCOLR), which should be cleaned out periodically.

There is a good discussion of Fee Assessment options in the Registration chapter of the Student User Manual.



Section O: Registration

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Registration Module?

```
select table_name
  from all_tables
 where table_name like 'SF%'
```

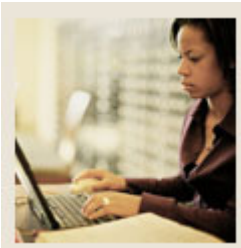
- What data elements are required?

```
desc sfrstcr
```

  - Notice the "NOT NULL" columns.

- What are the key fields in sfrstcr?

```
select column_name
  from all_cons_columns
 where table_name = 'SFRSTCR'
   and constraint_name = 'PK_SFRSTCR';
```



Section O: Registration

Lesson: Reports and Processes

Jump to TOC

Reports and processes

- Student Schedule Process (SFRSCHD)
    - Can be run in sleep/wake mode
- Class Roster Process(SFRSLST)
- Batch Fee Assessment Process (SFRFASM)
- Registered, Not Paid Process (SFRRNOP)
- Registration Purge Process (SFPREGS)

Student Schedule Process (SFRSCHD)

This process prints a student schedule for a term. It may be run in sleep/wake mode.

SFRSCHD is a C program run from Job Submission.

Batch Fee Assessment Process (SFRFASM)

This process is run if an institution decides not to do online fee assessment. Registration charges are posted to the student's account in the Accounts Receivable module.

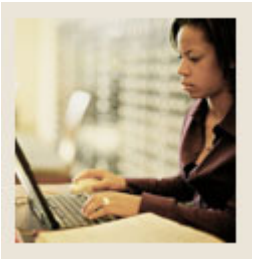
SFRFASM is a C program run from Job Submission.

There is a good discussion of Fee Assessment options in the Registration chapter of the Student User Manual.

Registered, Not Paid Process (SFRRNOP)

This process prints/purges all students who have registered but not yet paid for a term. It may be run in query or update mode.

SFRRNOP is a C program run from Job Submission.



Section O: Registration

Lesson: Sleep/Wake Mode

Jump to TOC

Purpose

Sleep/Wake mode allows you to run jobs in cyclical or "sleep/wake-up" manner.

There are two methods of doing this:

- Submit from Operating System and terminate manually
  (scripts are in $BANNER_HOME/general/misc and $BANNER_HOME/general/plus)
- Submit through SCT Banner Job Submission (GJAPCTL form)

Method two (through forms) is the most commonly used method.

Submitting Sleep/Wake processes via Job Submission

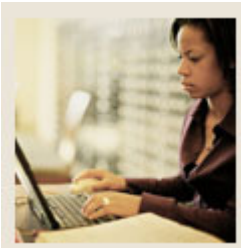
- Define Printer and print command on GTVPRNT
- On the SOADEST or TOADEST form, enter the correct printer code from GTVPRNT
- On GJAPCTL, for the valid sleep/wake jobs, enter the parameters that specify sleep/wake processing
- Stop sleep/wake process on GJASWPT form

Refer to Chapter 10 of the Technical Reference Manual for specific directions for setting up printer commands, etc.

Sleep/Wake jobs

Jobs that can be run in sleep/wake mode:

- Student Schedule Process (SFRSCHD)
- Academic Transcript Process (SHRTRTC)
- Account Receipt Process (TGRRCPT)
- Student Billing Statement Process (TSRCBIL)
- Miscellaneous Receipt Process (TGRMISC)



$BANNER_HOME/student/dbprocs

functions (sff*) ex: sffrgfe1.sql

$BANNER_HOME/student/views

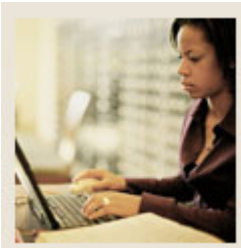
views (sfv*): sfvstc0.sql creates view as_student_registration_detail

Some views are used in conjunction with theObject:Access method of retrieving data from the database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

This one does not use the GTVSDAX table; not all layered views refer to GTVSDAX. It is used as a crosswalk table to “spread out” repeating table row values into columns for easier reporting.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



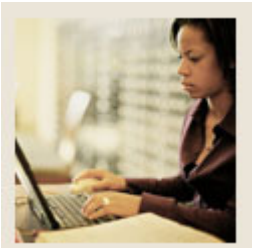
Section O: Registration

Lesson: Conversion Issues

Jump to TOC

Issues

- Conversion is not recommended for Registration.
- Possibly could run parallel
- Legacy and SCT Banner



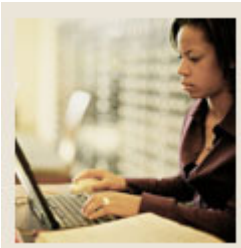
Section O: Registration

Lesson: Self Check – Registration Exercise

Jump to TOC

Exercise

Write a query that returns a student's full name and a list of courses for which he or she is registered for a given term, including: subject and course number, crn, and credit hours. Prompt user for term.



Section O: Registration

Lesson: Self Check – Registration Exercise – Answer Key

Jump to TOC

Exercise

Write a query that returns a student's full name and a list of courses for which he or she is registered for a given term, including: subject and course number, crn, and credit hours. Prompt user for term.

```
col stunam   for a30    hea   'STUDENT NAME'
col subj     for a8     hea   'SUBJECT'
col crse     for a6     hea   'COURSE|NUMBER'
col crn for a7     hea   'CRN'
col hrs for 99999  hea   'HOURS'

break on stunam noduplicates skip1

select  substr(spriden_last_name,1,15) || ', ' ||
          substr(spriden_first_name,1,15) stunam,
        ssbsect_subj_code  subj,
          ssbsect_crse_numb crse,
          sfrstcr_crn   crn,
          sfrstcr_credit_hr  hrs
 from   spriden, ssbsect, sfrstcr
where   spriden_pidm = sfrstcr_pidm
  and   spriden_change_ind is null
  and   sfrstcr_term_code = '&term'
  and   sfrstcr_rsts_code IN ('RE', 'RW')
  and   sfrstcr_crn = ssbsect_crn
  and   sfrstcr_term_code = ssbsect_term_code
order by  stunam;
```


Section P: Academic History

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the role and functions of the Academic History module

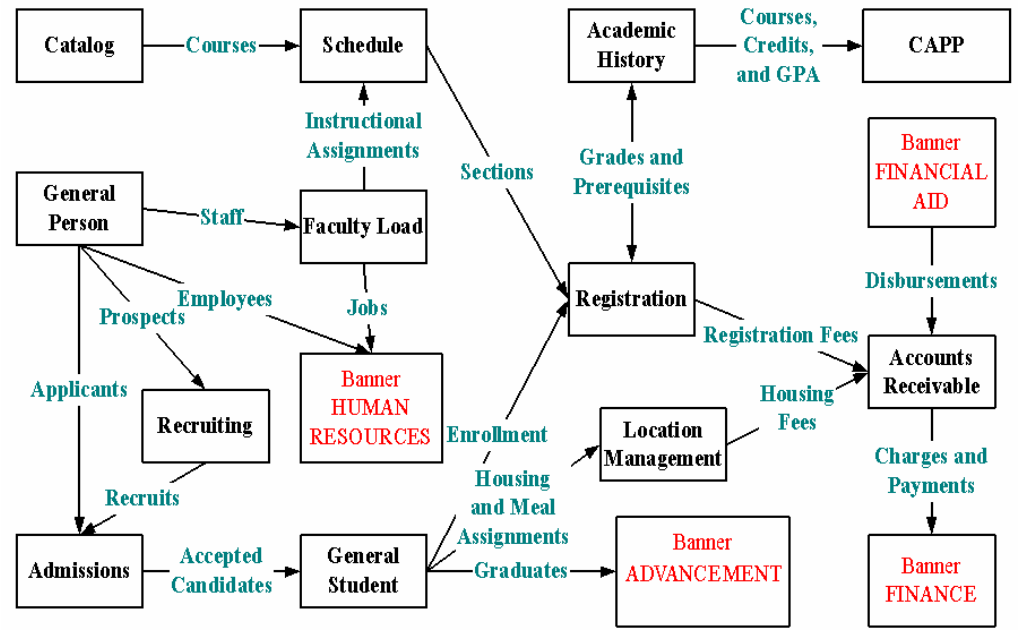
Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Section Content

Diagram



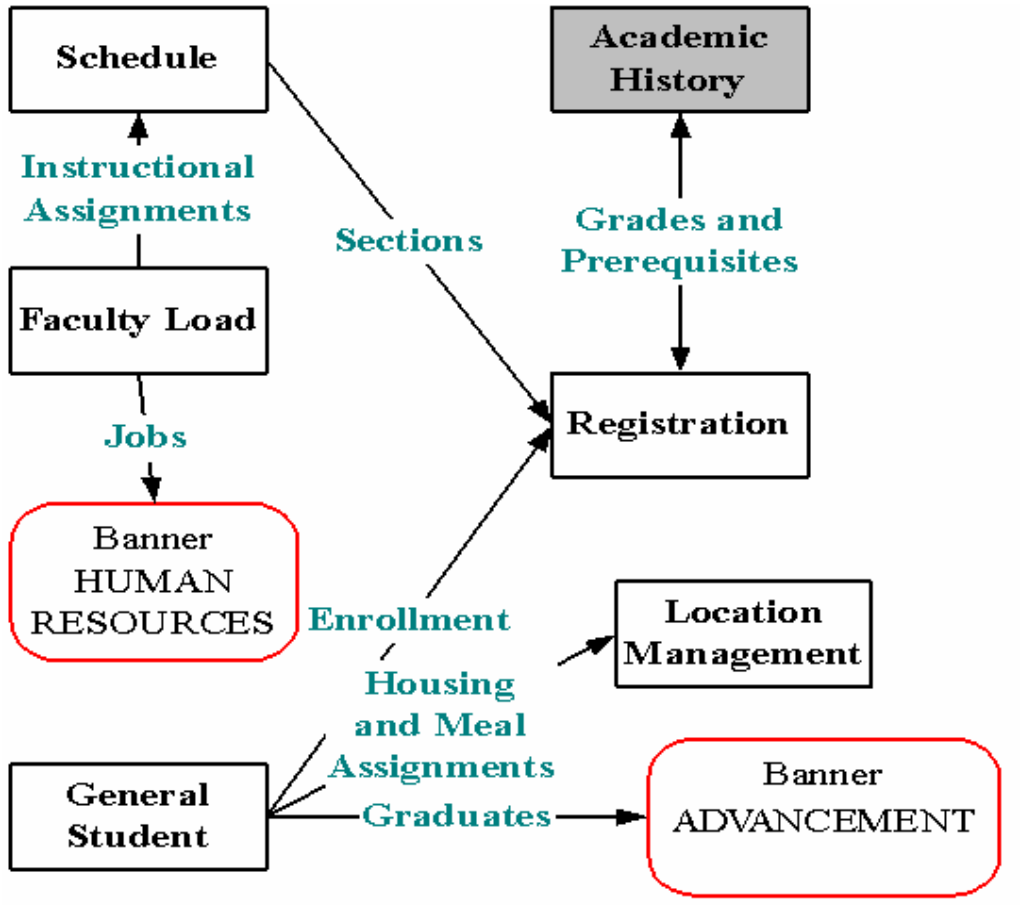
This overview diagram shows Academic History and its relationship with all other modules of the SCT Banner Student System.

Section P: Academic History

Lesson: Academic History Module

Jump to TOC

Diagram



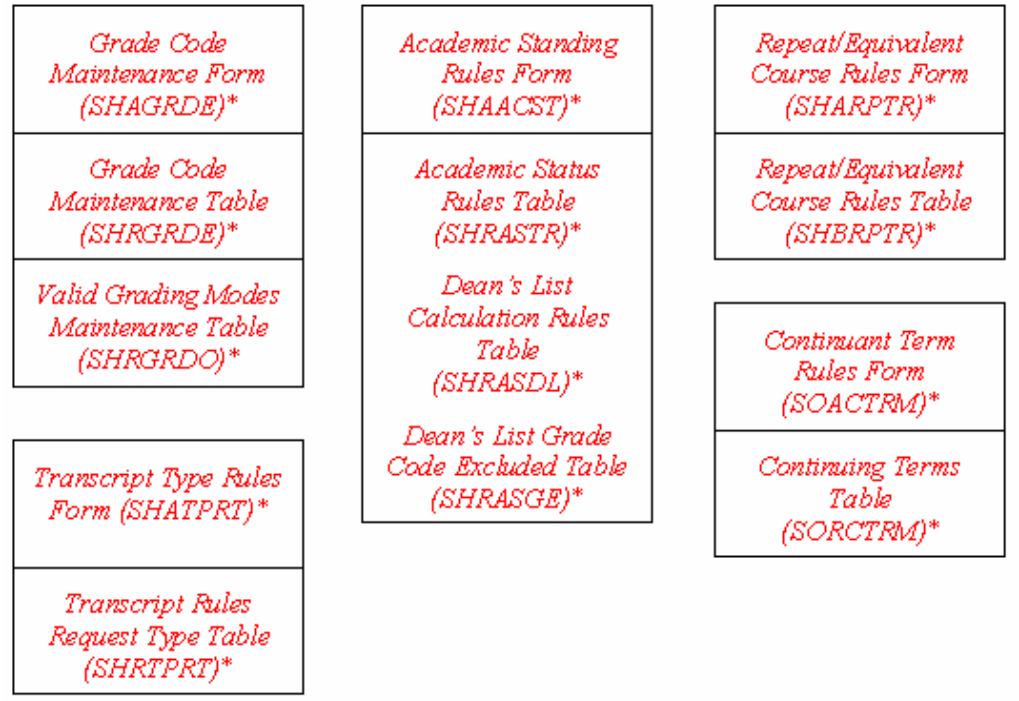
Academic History should be implemented prior to Registration if pre- and/or co-requisite checking is used. Otherwise, Academic History information can be implemented as soon as possible after Registration in order for current students to have statistics (and courses) in Academic History prior to the end of the first "live" term.

Objectives

Examine/Review

- Major & Required Forms and Tables
- Reports, Processes and Procedures
- Conversion of Data

Rules

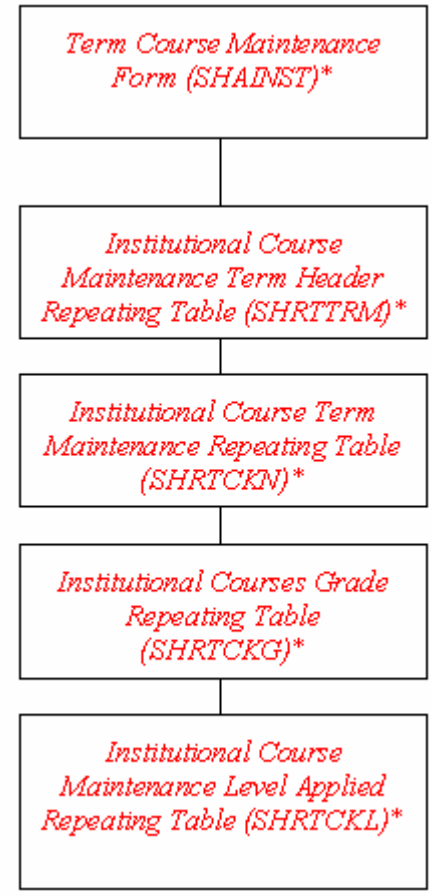


All of these rule tables are required. (*Asterisked (*)* text indicates required.)

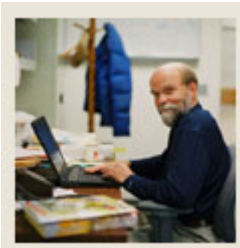
Major Validation Tables/Forms

- Grade Change Code Validation Form/Table (STVGCHG)
- Grading Mode Code Validation Form/Table (STVGMOD)
- Academic Standing Code Validation Form/Table (STVASTD)
- Many of the validation tables that have been referenced in previous modules

Diagram



If detailed institutional course information is to be converted, then all of the tables listed will be required.



Section P: Academic History

Lesson: Institutional Courses (Continued)

Jump to TOC

Term Course Maintenance Form (SHAINST)

Term Header Information Table (SHRTTRM)

- Academic Status
- Dean's List

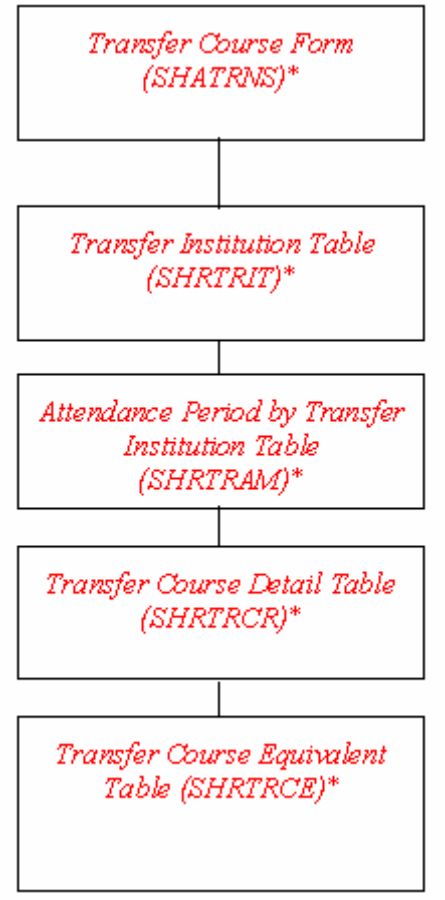
For each institutional course taken:

- Institutional Course Term Maintenance Table (SHRTCKN)
  - subjects, course numbers, titles, etc.
- Institutional Course Grade Repeating Table (SHRTCKG)
  - credit hours, final grade, etc.
- Course Level Applied Repeating Table (SHRTCKL)
  - course level applied

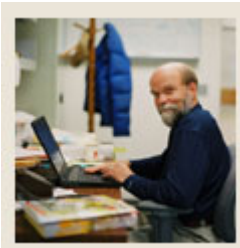
Records are associated by term and by SHRTCKN sequence numbers.

- Term Header Information Table (SHRTTRM)
- Course Section Attribute Table (SHRATTR)
- Transcript Comment Table by Level (SHRTMCM)
- Transcript Comment Table by Term (SHRTTCM)

Diagram



If detailed transfer course information is to be converted, then all of the tables listed will be required.



Section P: Academic History

Lesson: Transfer Courses (Continued)

Jump to TOC

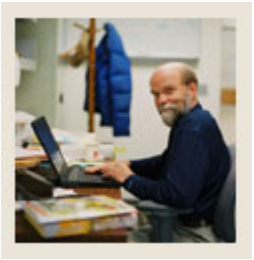
Academic History: Transfer Course Form (SHATRNS)

Required only if detail of transfer courses is to be converted.

For each course transferred:

- Transfer Institution Repeating Table (SHRTRIT)
- Attendance Period by Transfer Institution Repeating Table (SHRTRAM)
- Transfer Course Detail Repeating Table (SHRTRCR)
- Transfer Course Equivalent Repeating Table (SHRTRCE)

Records are associated by term, and by SHRTRAM, SHRTRIT and SHRTRCR sequence numbers.

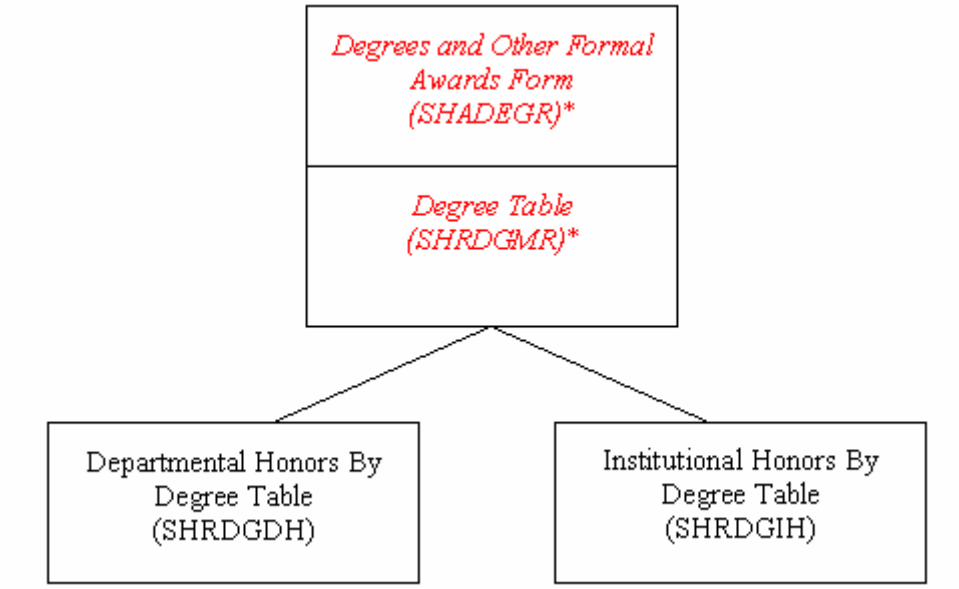


Section P: Academic History

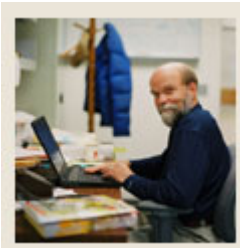
Lesson: Degrees

Jump to TOC

Diagram



Each student must have at least one degree record with the status 'SO'. Those students who have graduated will have a second sequence number with a status of 'AW'.



Section P: Academic History

Lesson: Degrees (Continued)

Jump to TOC

Degree Information - SHADEGR

Degree Repeating Table (SHRDGMR)

- Required even if a student does not have a degree

SHRDGMR_ DEGC_CODE = 'SO' for "seeking"
SHRDGMR_ DEGC_CODE = 'DA' or 'AW' for "degree awarded" if student has degree

- Contains major and term awarded

Institutional Honors by Degree Table (SHRDGIH)

- Used if student had institutional honors associated with the degree.

Departmental Honors by Degree Table (SHRDGDH)

- Used if student had departmental honors associated with the degree

Records in SHRDGIH and SHRDGDH are associated by SHRDGMR sequence numbers.

Note: There are other academic history tables that can be populated during the conversion based on the legacy data (e.g., SHRQPND -- Qualifying Papers)

Section P: Academic History

Lesson: GPA

Jump to TOC

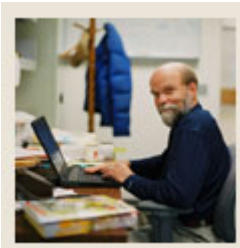
Tables

Level GPA Table (SHRLGPA)

- Cumulative institutional courses (I)
- Cumulative transfer courses (T)
- Overall GPA (O) (Includes both institutional and transfer courses)

Term GPA Table (SHRTGPA)

- Term statistics for institutional courses (I)
- Term statistics for transfer courses (T)

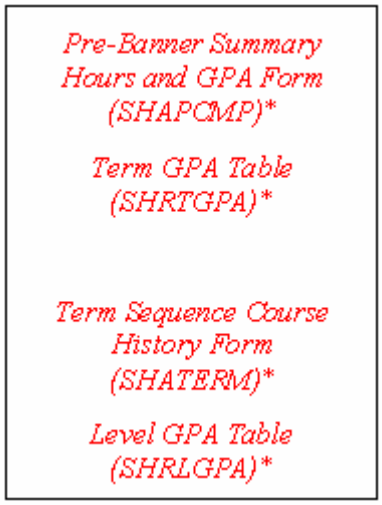


Section P: Academic History

Lesson: Pre-Banner Summary

Jump to TOC

Diagram

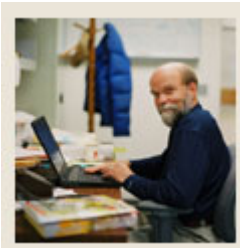


If detailed academic history -- institutional courses/transfer courses -- are not going to be converted, then SHRTGPA should be populated with the students' GPA information -- (I)nstutional and (T)ransfer.

Refer to the Student User Manual, Chapter 15.

SHAPCMP

The Pre-Banner Summary Hours and GPA Form (SHAPCMP) captures and maintains summary GPA in lieu of the actual converted term's course work. This is helpful if an institution does not intend to convert transcript data, or has chosen to defer the conversion to a later date. The ability to add pre-Banner hours and GPA means that more accurate assessments can be made when determining class level, and in calculating the institutional or transfer GPA.



Section P: Academic History

Lesson: Term GPA Table (SHRTGPA)

Jump to TOC

Term GPA Table (SHRTGPA)

Results are displayed in the Pre Banner Summary Hours and GPA Form (SHAPCMP).

There will be at *least* one record per student in SHRTGPA:

- shrtgpa_type_ind = 'I'
    - would reflect total cumulative statistics
- Use '000000' as the term code

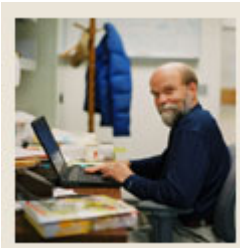
It is possible to have two records in SHRTGPA ('I' and 'T' GPA types) for a student who has transferred.

Conversion

For summary conversion, determine whether separate brought-forward data will be maintained for institutional and transfer data.

If only a total cumulative strip is to be converted, load only one SHRTGPA record per level per student, using "000000" as the SHRTGPA_TERM_CODE. For this record, set the _GPA_TYPE_IND to 'I'.

If separate institutional and transfer brought-forward data is converted, load one or two SHRTGPA records per level per student. Load one record per level, representing the institutional GPA, if the person has no brought-forward transfer work, using "000000" for the SHRLGPA_TERM_CODE and "I" for the SHRTGPA_GPA_TYPE_IND. If the person also has brought-forward transfer work, load a second record, using "000000" for the _TERM_CODE and "T" for the _GPA_TYPE_IND.



Section P: Academic History

Lesson: Level GPA Table (SHRLGPA)

Jump to TOC

Level GPA Table (SHRLGPA)

SHRLGPA contains two records per student per level, with a third record for students who have transferred:

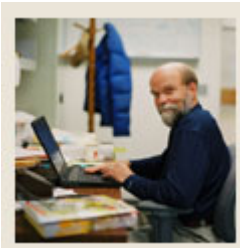
- (I)nstitutional GPA
- (T)ransfer GPA (not always present, depending on the student)
- (O)verall GPA (a combination of Institutional and Transfer GPAs)

Conversion

For summary conversion, determine whether separate brought-forward data will be maintained for institutional and transfer data.

If only a total cumulative strip is to be converted, load two SHRLGPA records per level per student. For one record, set the SHRLGPA_GPA_TYPE_IND to "I" (Institutional). For the second record, set the SHRLGPA_GPA_TYPE_IND to "O" (Overall).

If separate institutional and transfer brought-forward data is converted, load two or three SHRLGPA records per level per student, depending upon whether the student has transfer work at the level. For the institutional record, set the _GPA_TYPE_IND to "I". For the transfer record, set the _GPA_TYPE to "T". For the cumulative record, set the _GPA_TYPE_IND to "O".



Section P: Academic History

Lesson: SQL*Plus

Jump to TOC

Questions

- What tables are part of the Academic History Module?

```
select table_name
  from all_tables
 where table_name like 'SH%'
```

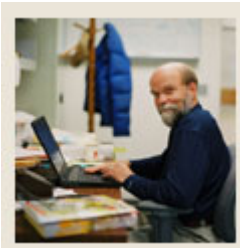
- What data elements are required?

```
desc shrdgmr
```

  - Notice the "NOT NULL" columns.

- What are the key fields in shrdgmr?

```
select column_name
  from all_cons_columns
 where table_name = 'SHRDGMR'
   and constraint_name = 'PK_SHRDGMR';
```



$BANNER_HOME/student/dbprocs

functions (shf*) ex: shfttrm.sql

$BANNER_HOME/student/views

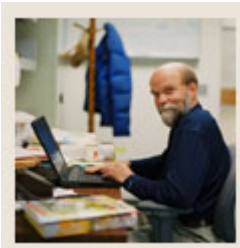
views (shv*): shvsum0.sql creates view as_academic_history_summary

Some views are used in conjunction with theObject:Access method of retrieving data from the database, using the concept of “layered” views; you must have the GTVSDAX form/table populated with crosswalk values.

This one does not use the GTVSDAX table; not all layered views refer to GTVSDAX. It is used as a crosswalk table to “spread out” repeating table row values into columns for easier reporting.

Clients should know the naming conventions and the locations of these other database object creation scripts. To use them as models, create your own MODS directory (as discussed earlier in this course) and put any modifications in there.

For more information about Object:Access views and the GTVSDAX translation form/table, contact your account manager to request the manuals and/or appropriate training.



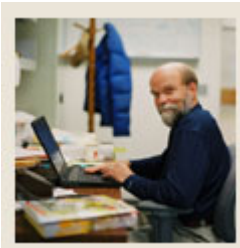
Section P: Academic History

Lesson: Conversion Issues

Jump to TOC

Questions

- Will detailed academic history data be converted?
- Will you convert ALL academic history records or only a certain number of years?
- What academic history data do you have in your legacy system?
- How do you determine where to put it in Banner?



Section P: Academic History

Lesson: Reports/Processes - End of Term

Jump to TOC

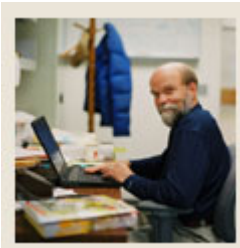
End of Term processes

- Grade Roll to Academic History Process (SHRROLL)
- Repeat/Equivalent Course Check Process (SHRRPTS)
- Calculate GPA Process (SHRCGPA)
- Calculate Academic Standing Process (SHRASTD)
- Grade Mailer Process (SHRGRDE)
- Student Type Update (SHRTYPE)

Refer to the Student Technical Reference Manual, Chapter 10, for additional information on Student Module reports and processes.

Additional processes

- Academic Transcript Process (SHRTRTC)
    - Can be run in sleep/wake mode
- Degree Status Update Process (SHRDEGS)



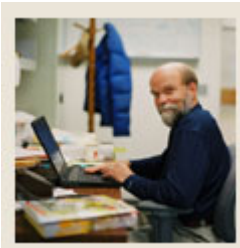
Section P: Academic History

Lesson: Self Check – Academic History Exercises

Jump to TOC

Exercise 1

Write a query which returns full name, id, level (the level code associated with gpa hours and calculations), and term gpa (for institutional work) for a given term. Prompt user for term.

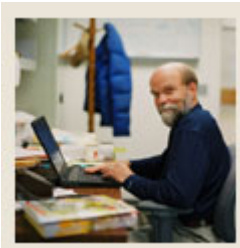


Exercise 2

Write a query which returns full name, id, course level, crn, subject code, course number, and grades for a given term. Prompt user for term.

Exercise 3

Write a query which returns full name, id, level, all transfer courses, and grades for all students who have transfer work. Order by student last name.



Section P: Academic History

Lesson: Self Check – Academic History Exercises – Answer Key

Jump to TOC

Exercise 1

Write a query which returns full name, id, level (the level code associated with gpa hours and calculations), and term gpa (for institutional work) for a given term. Prompt user for term.

```
select  substr(spriden_last_name,1,15) || ', '  ||
          substr(spriden_first_name,1,15),
        spriden_id, shrtgpa_term_code,
          shrtgpa_levl_code, shrtgpa_gpa
  from  spriden, shrtgpa
 where  shrtgpa_pidm = spriden_pidm
   and  spriden_change_ind is null
   and  shrtgpa_term_code = '&term'
   and  shrtgpa_gpa_type_ind = 'I'
 order by spriden_last_name;
```

Exercise 2

Write a query which returns full name, id, course level, crn, subject code, course number, and grades for a given term. Prompt user for term.

The solution to this exercise contains some formatting as well as the select statement. You may introduce the formatting if there's time.

```
col     stunam      for a25     hea 'STUDENT NAME'
col     id          for a10     hea 'ID'
col     levl        for a5      hea 'LEVEL'
col     crn         for a6      hea 'CRN'
col     subj        for a6      hea 'SUBJ'
col     crse        for a6      hea ' CRSE |NUMBER'
col     grde        for a6      hea 'GRADE'
col     today       new_value   xtoday
col     term        new_value   xterm

ttitle -
              LE  xtoday -
              CE 'Student Grade Report for ' xterm -
              RI 'Page:  ' format 999 sql.pno -
              SKIP2

break -
              on stunam  nodup skip1 -
              on id      nodup -
              on id      nodup -
              on levl    nodup
```

Exercise 2, continued

```
select  substr(spriden_last_name,1,15) || ', ' ||
        substr(spriden_first_name,1,15) stunam,
        spriden_id id,
           shrtckl_levl_code levl,
        shrtckn_crn crn,
        shrtckn_subj_code subj,
        shrtckn_crse_numb  crse,
        shrtckg_grde_code_final grde,
        sysdate today,
        shrtckn_term_code term
 from   spriden, shrtckl, shrtckn, shrtckg
where   shrtckg_pidm = spriden_pidm
  and   spriden_change_ind is null
  and   shrtckg_term_code = '&term'
  and   shrtckg_pidm = shrtckn_pidm
  and   shrtckg_term_code = shrtckn_term_code
  and   shrtckg_tckn_seq_no = shrtckn_seq_no
  and   shrtckg_pidm = shrtckl_pidm
  and   shrtckg_term_code = shrtckl_term_code
  and   shrtckg_tckn_seq_no = shrtckl_tckn_seq_no
order   by 1,5,6;
```

Exercise 3

Write a query which returns full name, id, level, all transfer courses, and grades for all students who have transfer work. Order by student last name.

```
set pagesize 47
set linesize 130

col     stunam for a25   hea 'STUDENT NAME'
col     id     for a10   hea 'ID'
col     term   for a6    hea 'TERM'
col     subj   for a20   hea 'TRAN CRSE NAME'  TRUNC
col     crse   for a6    hea ' TRAN|CRSE |NUMBER'
col     titl   for a25   hea 'TITLE' TRUNC
col     grde   for a6    hea 'GRADE'

select  substr(spriden_last_name,1,15) || ', ' ||
          substr(spriden_first_name,1,15) stunam,
          spriden_id id,
             shrtrcr_term_code term,
             shrtrcr_trans_course_name subj,
        shrtrcr_trans_course_numbers crse,
        shrtrcr_tcrse_title titl,
             shrtrcr_trans_grade grde
  from  spriden, shrtrcr
 where  spriden_pidm = shrtrcr_pidm
   and  spriden_change_ind is null
 order by spriden_last_name,
             shrtrcr_term_code;
```

Section Q: Conversion

Lesson: Overview

Jump to TOC

Objectives

At the end of this section, you will be able to

- Describe the conversion process

Prerequisites

To complete this section, you should have

- completed OR101 (Introduction to Oracle)
- completed SCT Banner Navigation

Secton Contents

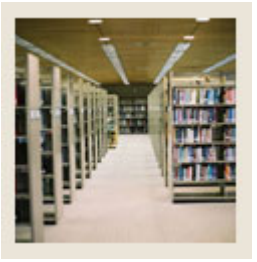
Section Q: Conversion

Lesson: Conversion Considerations

Jump to TOC

Considerations

- Keeping track of PIDM on legacy system
- Generated ID or SSN?
- Name/Address formatting
    - Avoid “#” if using letter generation
    - Additional data standards if using BannerQuest
- Address types
    - Do you have Multiple ID’s on legacy system?



Section Q: Conversion

Lesson: Conversion Steps

Jump to TOC

Steps

- Document steps as you proceed
- Review current data
- Determine scope:
    - What will you convert?
- Map legacy data to Banner tables
- Write a detail plan of:
    - Data to be converted
    - Banner tables to be populated
    - Deadlines/timelines
- Review plan & get approval from users
- Develop procedures & programs
- Test conversion in TEST or PPRD database
- Users verify data
- TEST again and make corrections to procedures and programs
- Do conversion in production
- Users verify data

Resources

Users can populate the validation tables, and can/will use some of the seed data.

Functional consultants may assist with the conversion process (mapping).

Refer to the Student Technical Reference Manual for more information on conversion topics:

- Chapter 5: Conversion
  Has good information regarding Accounts Receivable Data Conversion
- Chapter 6: Migration to Production
  Includes info about seed data that must be kept
- Chapter 7 : Integration
  Includes a list of Shared Tables, and has information about ethnicity codes and non-resident aliens
- Chapter 8: Process Flow Diagrams

Section Q: Conversion

Lesson: Conversion Steps (Continued)

Jump to TOC

Notes

Conversions can be automatic, manual or a combination of both.

Validation information can be keyed in by end-users. If RI is turned on, much error handling does not have to be built into scripts or programs.

When doing a conversion, keep in mind that both form-based and table-based rules must be met.

Section Q: Conversion

Lesson: Conversion Strategies

Jump to TOC

Strategies

- Create data standards, especially for names and addresses
  All offices need to agree to and document data standards

- Determine whether you will enter the data electronically or manually
  e.g., Some validation tables/forms can be entered manually in both the preproduction and production databases *if* the number of records is small (unlike STVSBGI!!)

- Determine which tables you will be using
  May be helpful to look at the forms with the users, then you will be able to determine the tables used

- Mapping legacy data to SCT Banner
- Review the legacy to SCT Banner mapping with the users
- Create a document programmers can use that tells how to convert the data
- Create a Conversion Plan document
- Review the steps that are involved to get to your “go live” dates
- Create a time line
- Determine the processes that need to be written
    - Will data need to be translated?
    - Will data need to be cleaned up on legacy side?

If error handling is built into programs, then RI can be turned off.

Values

Chapter 6 of the Student Technical Reference Manual lists all validation table values that must be kept in production.

If the **System required indicator** = 'Y', this does not mean one must use this information. Most of this information is needed by external reports to third parties (e.g. IPEDS).

Section Q: Conversion

Lesson: Conversion Examples

Jump to TOC

Examples

The following examples will demonstrate how to:

- Convert data to three Banner tables
- Create, drop, and alter temporary tables
- Assign a pidm
- Use SQL*LOADER to load temporary tables
- Use Update statement and Decode function to do cross-walk (translation)
- Use Insert statement
- Use a shell script or command procedure
- Check the data when complete
- Clean up data if it is incorrect

Flat file

The examples will use a flat file containing

- Person's (student's) SSN
- Last name
- First name
- Street
- City
- State
- Zip
- Sex
- Birth date

Information converted

We will convert basic general person information:

- Person Identification/Name Table (SPRIDEN)
- General Person Table (SPBPERS)
- Address Information Table (SPRADDR)

Section Q: Conversion

Lesson: Conversion Example: Flat File Layout

Jump to TOC

Layout

```
210009506Abbe     Anthony      PO Box 21049      Malvern       PA19355226-MAR-77
610009711Abbot    James        PO Box 27         Malvern       PA19355217-NOV-79
210009101Adams    Andrew       803 King Street   Malvern       PA19355210-DEC-72
610009101Adams    Anthony      20789 Lancaster Ln Clarksville  PA15122210-DEC-74
710000011Adams    Eugene       3400 Wendrow Way  University ParkPA16802201-JAN-01
210009619Barker   Clementine   83 Park Avenue    New York      NY10013128-APR-72
210009613Barker   James        854 Charlestown Pk King of PrussiaPA19401201-DEC-77
```

This layout is in columns, although comma delimited or quotation mark delimited can be used with SQL*LOADER (see manual for more information.)

Section Q: Conversion

Lesson: Conversion Example: Create Statement

Jump to TOC

Code

```
Create temporary tables   (create_temp.sql):
spool create_tables
drop table sytiden;
drop table sytaddr;
drop table sytpers;
create table sytiden as select * from spriden where 1 = 2;
create table sytaddr as select * from spraddr where 1 = 2;
create table sytpers as select * from spbpers where 1 = 2;
spool off
```

Usage

First, create temporary tables based on the actual layout of the SCT Banner table, then alter the table so that the pidm column could be null (not absolutely necessary – depends on method for creating the pidm). This example will not need a null pidm, but in later temp tables when the pidm already exists in SCT Banner tables, it may be a good method for handling pidms.

Pidms may be created during the load process or after the data is loaded into the temp tables. The main thing here is to know how to create, alter, and drop tables. Drop is in the script so that if a .shl or .com is used, the process can be rerun. (i.e., drop all tables, then recreate them) -- Not mandatory.

Remember when a table is dropped, all of the data is lost. In a production situation, a backup of the completed temp table should be made before manipulating the data in the temp table and running the process.

Section Q: Conversion

Lesson: Conversion Example: Alter Statement

Jump to TOC

Code

```
Alter temporary tables   (alter_temp.sql):
spool alter_tables
alter table sytiden modify spriden_pidm null;
alter table sytaddr modify spraddr_pidm null;
alter table sytpers modify spbpers_pidm null;
spool off
```

Usage

Alter the table so that the pidm column could be null (not absolutely necessary – depends on method for creating the pidm). This example will not need a null pidm, but in later temp tables when the pidm already exists in SCT Banner tables, it may be a good method for handling pidms.

For example, if HR has been converted, but you know some HR general person records are students, then you can get the pidm from SCT Banner before proceeding.

General student

If general person records have been created for students and you are now doing general student information:

```
update sytiden y
   set y.spriden_pidm =
         (select distinct x.spriden_pidm
            from spriden x
           where y.spriden_id = x.spriden_id);
```

You can then alter the temporary table used for general student information to include an id:

```
update sytstdn
   set sytstdn_pidm =
        (select distinct spriden_pidm
           from spriden
          where spriden_id = sytstdn_id);
```

Section Q: Conversion

Lesson: Conversion Example: SQL*LOADER

Jump to TOC

Code

SQL*LOADER (load.ctl):

```
load data
infile 'data_file.dat'
badfile 'bad_data.txt'
discardfile 'discard_file.txt'
append
into table sytiden (
spriden_pidm       sequence(77777777,1),
spriden_id              position(1:9),
spriden_last_name  position(10:23),
spriden_first_name position(24:39),
--  spriden_change_ind  null,
spriden_entity_ind constant 'P',
spriden_activity_date   constant '25-DEC-98',
spriden_user       constant 'CONVERSION',
spriden_origin          constant 'CONVERSION')
into table sytaddr (
spraddr_pidm       sequence(77777777,1),
spraddr_atyp_code    constant 'MA',
spraddr_seqno      constant '1',
spraddr_street_line1    position(40:58),
spraddr_city       position(59:73),
spraddr_stat_code  position(74:75),
spraddr_zip        position(76:80),
spraddr_activity_date   constant '25-DEC-98',
spraddr_user       constant 'CONVERSION')
into table sytpers (
spbpers_pidm       sequence(77777777,1),
spbpers_ssn          position(1:9),
spbpers_sex        position(81:81),
spbpers_birth_date position(82:90),
spbpers_activity_date   constant '25-DEC-98')
```


Section Q: Conversion

Lesson: Conversion Example: SQL*LOADER (Continued)

Jump to TOC

Usage

This example of SQL*LOADER includes a method for assigning a pidm.

The starting integer is determined after checking the SCT Banner database for the max (and min) spriden_pidm. 77777777 is used in this example.

The data in the flat file is in columns; however, the loader can be written to use comma or quotation delimited files. The default is APPEND (so it is optional) and the discardfile is optional.

The activity date could be sysdate rather than a constant, but using a date like 25-DEC-97 stands out as a “conversion” date.

Spriden_change_ind will be null by default (note that it is commented out).

Notes

If one record does not load for some reason -- i.e., the data in the column is out of alignment, then none of the data will load into the temp table for that record, but the sequence number (pidm) will be preserved.

The log from the loader will indicate which record didn’t load. The bad data file will also show which records didn’t load and could be edited to use with the next load of data.

Verification

Verify that the number of records loaded match the number in the data file.

- Review the log file.
- Check that pidms were assigned properly.
- Were all NOT NULL columns filled?

Section Q: Conversion

Lesson: Conversion Example: Decode Statement

Jump to TOC

Code

Decode SPBPERS_SEX (decode_sex.sql):

```
spool decode
update sytpers
set spbpers_sex = decode (spbpers_sex,
  ‘1','F','2','M','N');
spool off
```

Usage

This is an example of using the update statement to decode a value in the sytpers temp table. This script is run after the temp table is loaded. A simple example of how cross-walking can be done within the temp table rather than on the legacy side. (Whichever is easier for you!)

Section Q: Conversion

Lesson: Conversion Example: Check data in the temp tables

Jump to TOC

Code

```
   select     spriden_id,substr(spriden_last_name,1,15)||
         ',                          '||
              spriden_first_name, spriden_change_ind IND,
           spriden_entity_ind ENT,
           spriden_activity_date, spriden_pidm,
           spraddr_pidm, spbpers_pidm,
           spraddr_street_line1, spraddr_city,
           spraddr_stat_code, spraddr_zip, spbpers_sex,
spbpers_birth_date
     from      sytiden, sytaddr, sytpers
   where      spriden_pidm = spraddr_pidm
     and      spriden_pidm = spbpers_pidm
    order by   spriden_pidm;
```

Usage

Check the data in the temporary tables. This is just one example of a simple script to use for checking.

Section Q: Conversion

Lesson: Conversion Example: Insert Statement

Jump to TOC

Code

```
Insert into SATURN tables  (insert_real.sql):
spool insert_real
insert into spriden select * from sytiden;
insert into spraddr select * from sytaddr;
insert into spbpers select * from sytpers;
spool off
```

Usage

Now insert the data into the “real” SCT Banner tables.
Review the .lst file and verify that all records were inserted.

Section Q: Conversion

Lesson: Conversion Example: Check the data in SCT Banner

Jump to TOC

Code

```
   select    spriden_pidm, substr(spriden_last_name||',’
                   ||spriden_first_name,1,25),
                    spriden_entity_ind, spraddr_atyp_code,
                    spraddr_seqno, spraddr_street_line1,
                    spraddr_city, spraddr_stat_code,
                    spraddr_zip, spbpers_sex,
                    spbpers_birth_date
        from    spraddr, spbpers, spriden
       where    spriden_pidm > 77777776
         and    spriden_pidm = spraddr_pidm
         and    spriden_pidm = spbpers_pidm
    order by    spriden_pidm;
```

Usage

Data is now in the SCT Banner tables and should be reviewed before proceeding.

Notice the search and soundex columns that are now in SPRIDEN.

Write some scripts to look at the data in SCT Banner.

Section Q: Conversion

Lesson: Conversion Example: Update SOBSEQN

Jump to TOC

Code

```
update sobseqn
   set sobseqn_maxseqno = 77777783,
       sobseqn_activity_date = sysdate
  where sobseqn_function = 'PIDM';
```

Usage

Provided all is well with the insert and the data looks good, then SOBSEQN needs to be updated with the appropriate pidm that was last used and the activity date.

There may be cases where the pidm range for student records is lower than a product that was previously converted (e.g., human resources may have a higher pidm range).

Section Q: Conversion

Lesson: Conversion Example: Clean the data in SCT Banner

Jump to TOC

Code

Clean SATURN tables (clean_tables.sql):

```
spool clean_tables
delete from spriden where spriden_pidm > 77777776;
delete from spraddr where spraddr_pidm > 77777776;
delete from spbpers where spbpers_pidm > 77777776;
spool off
```

Usage

Clean the "real" SCT Banner tables if the data was not inserted correctly or you wish to rerun the process. (clean_tables.sql)

For this example, we will clean out the data we inserted into Banner.
In a test and production environment, you may need to delete all or some records if they are inaccurate or invalid. By looking at the activity date in each table or the pidm range, records can be deleted.

Notes

Some versions of sql*loader will permit the use of a sequence. In this case, instead of using sequence(77777777,1) use myseq.nextval and myseq.currval. A sequence would need to be created.

```
create sequence for PIDM   (create_seq.sql):
spool sequence
drop sequence myseq;
create sequence myseq
  increment by 1
  start with 77777777;
spool off
exit
```


Section Q: Conversion

Lesson: Conversion Example: Shell script

Jump to TOC

Code

Shell Script (convert.shl):

```
export ORAENV_ASK=NO
export ORACLE_SID=YOURSID
. oraenv

sqlplus saturn/u_pick_it @create_temp
sqlplus saturn/u_pick_it @alter_temp
sqlldr saturn/u_pick_it control=load.ctl
sqlplus saturn/u_pick_it @decode_sex
sqlplus saturn/u_pick_it @insert_real
```

Usage

After verifying that each script works properly, the scripts can be combined into a shell script (for UNIX) or a command file (for VMS).

Your shell script may look differently depending on your operating system. Check with your system administrator.

Notes

Each script (i.e., create_temp) will need to have 'exit' at end:

```
spool create_tables
drop table sytiden;
drop table sytaddr;
drop table sytpers;
create table sytiden as select *
                          from spriden
                         where 1 = 2;
create table sytaddr as select *
                          from spraddr
                         where 1 = 2;
create table sytpers as select *
                          from spbpers
                         where 1 = 2;
spool off
exit
```


Release Date

Jump to TOC

This workbook was last updated on 09/19/2005.