Banner Technical
Human Resources Technical Training Workbook

Release 8.0 - April 2008

Updated 4/30/2008

SUNGARD HIGHER EDUCATION

What can we help you achieve?

SunGard Higher Education
4 Country View Road
Malvern, Pennsylvania 19355
United States of America
(800) 522 - 4827

Customer Support Center website
http://connect.sungardhe.com

Distribution Services e-mail address
distserv@sungardhe.com

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

Trademark
Without limitation, SunGard, the SunGard logo, Banner, Campus Pipeline, Luminis, PowerCAMPUS, Matrix, and Plus are trademarks or registered trademarks of SunGard Data Systems Inc. or its subsidiaries in the U.S. and other countries. Third-party names and marks referenced herein are trademarks or registered trademarks of their respective owners.

Revision History Log

Publication Date	Summary
4/30/2008	New version that supports Banner Human Resources 8.0 software.

Table of Contents

Introduction

Course goal

This course is designed to provide an overview of the major tables, reports and processes included in each module of the Human Resources System, as well as providing an introduction to the Banner directory structure. It will prepare technical staff to support Human Resources in the implementation and operations of the Banner Human Resources product.

The course reviews some basic concepts such as naming conventions, use of the data dictionary, and referential integrity.

Hands-on exercises are used to demonstrate the table/form/process relationships.

Course objectives

Participants in this course will be able to

- identify Banner Human Resources forms and tables
- query the Banner HR tables.
- identify tables and fields for data conversion.
- identify tables and fields for migration to the production database.
- follow key HR processes.
- identify and read reports, processes, procedures and scripts in Banner Human Resources

Intended audience

Programmers, DBAs, and analysts who may teach others about Banner tables and processes, perform programming tasks in the Banner environment, facilitate reporting, utilize any of the Banner Human Resources System features, or perform analysis on any Banner Human Resources module.

Prerequisites

To complete this course, you should have

- Completed Banner Fundamentals
- Completed Banner General
- Familiarity with Relational Databases
- SQL Knowledge
- PL/SQL Knowledge
- Oracle Forms Knowledge

Foundations

Introduction

This section discusses the basic foundations of the Banner Human Resources system.

Objectives

At the conclusion of this section, participants should have a better understanding of:

- Banner object naming conventions
- Banner column naming conventions
- PIDMs
- Common Matching functionality.

Foundation Topics

Topics

- Naming Conventions
- Objects
- Columns
- Constraints
- PIDM
- Common Matching (Banner General)

Banner Objects: Naming Conventions

Naming convention

All Banner objects adhere to a seven-character naming convention for their objects.

Characters identify a particular quality or attribute of the object.

Banner objects

Objects can be:

- Tables
- Views
- Forms
- Processes

Object Naming Conventions: Position 1

Position 1

Position 1 - Identifies the primary system owning the form, report, process, or table.

- The primary system corresponds to a Banner product
- Each product has its own 'schema' in the ORACLE database

Product owners

Product	Owner
General	GENERAL
General Person	SATURN
Finance	FIMSMGR
Accounts Receivable	TAISMGR
Position Control	POSNCTL
Payroll	PAYROLL
Student	SATURN
Financial Aid	FAISMGR
Advancement	ALUMNI
Security	BANSECR

Primary systems

Code	System
A	Advancement
F	Finance
G	General
H	New Products (Web)
N	Position Control
P	HR/Payroll/Personnel
R	Financial Aid
S	Student/Common
T	Accounts Receivable
V	Voice Response
W, X, Z	Client-developed

Object Naming Conventions: Position 2

Position 2

Position 2 - Identifies the component owning the form, report, process, or table.

If Position 1 is P or N:

Code	Owning component
A	Applicant
B	Budget
C	COBRA
D	Benefits/Deductions
E	Employee
H	Time Reporting /History
O	Overall
P	General Person
T	Table (Validation or Rule)
R	Electronic Approvals
U	Utility
X	Tax Administration
W, Y, Z	Client-developed forms

Object Naming Conventions: Position 3

Position 3

Position 3 - Identifies the type or function of the object.

Code	Type/Function
A	Application
B	Base Table, Batch COBOL Process
I	Inquiry Form
P	Process
R	Rule or Repeating Table, Report/Process
V	Validation Table or Form, View
Q	Query Form

Object Naming Conventions: Position 4

Position 4

Positions 4,5,6 &, 7 - A descriptive four-character name for the object.

Example 1

HR Example: PPAIDEN

P	Payroll
P	Person
A	Application Form
IDEN	Identification

Example 2

Another HR Example: PEBEMPL

P	Payroll
E	Employee
B	Base
EMPL	Employee

Example 3

SPRIDEN

S	Common
P	Person
R	Repeating Table
IDEN	Identification

Example 4

GUAIDEN

G	General
U	Utility
A	Application
IDEN	Identification

Banner Columns: Naming Conventions

Table column names

Table column names start with the seven-character table name followed by an underscore and the column name.

If followed by _code, then it validates against a rule or validation table.

Example 1

tablename_pidm

EXAMPLE: SPBPERS_SSN EXAMPLE: SPRIDEN_ID

Example 2

*tablename*_ecls_code

EXAMPLE: SPRADDR_STAT_CODE

- Relates to STVSTAT_CODE

PIDM

Definition

Banner products store people-related records in the database using an internal Key field called a PIDM.

PIDM is used instead of the person's ID number as the key, so that a person can change his or her ID with relative ease.

Acronym

- PIDM is an acronym for ***P***erson ***Id****entification* ***M***aster
- Data type: number

PIDM and tables

- SPRIDEN and all other person related tables are linked together by PIDM
- An Oracle Sequenced Object is used to generate one-up numbers for PIDM creation
- gp_common API (Application Program Interface)
- f_generate_pidm (Function call to the Oracle Sequence)
- pidm_sequence (Oracle Sequence)

Generated IDs

- IDs – Manual, Generated, Previous
- Manual IDs entered by Users
- Generated IDs use Oracle sequence
  - f_generate_id
  - id_sequence
- Prefix for Generated stored in SOBSEQN
- Previous IDs migrated or entered by Users
- spriden_change_ind = 'N' or 'I'

Example

```
select gb_common.f_generate_id() from dual;
select id_sequence.nextval from dual;
```

A00000001 = Generated ID

Common Matching

Definition

A functionality to prevent a single entity (person or non-person) being assigned two or more internal identification records in Banner (PIDMs), a condition known as multiple PIDMs.

Since multiple names and IDs can be associated with a single PIDM in Banner, each entity should have one and only one PIDM.

Benefits

- Helps prevent the accidental creation of multiple PIDMs
- Rule-driven process to determine whether an entity (person or non-person) is truly new
- Centralized algorithm
- Unlimited rules can be created
- Matching can be turned on and off system-wide (GUAINST) or per user
- Defaults can be set for users

Characteristics

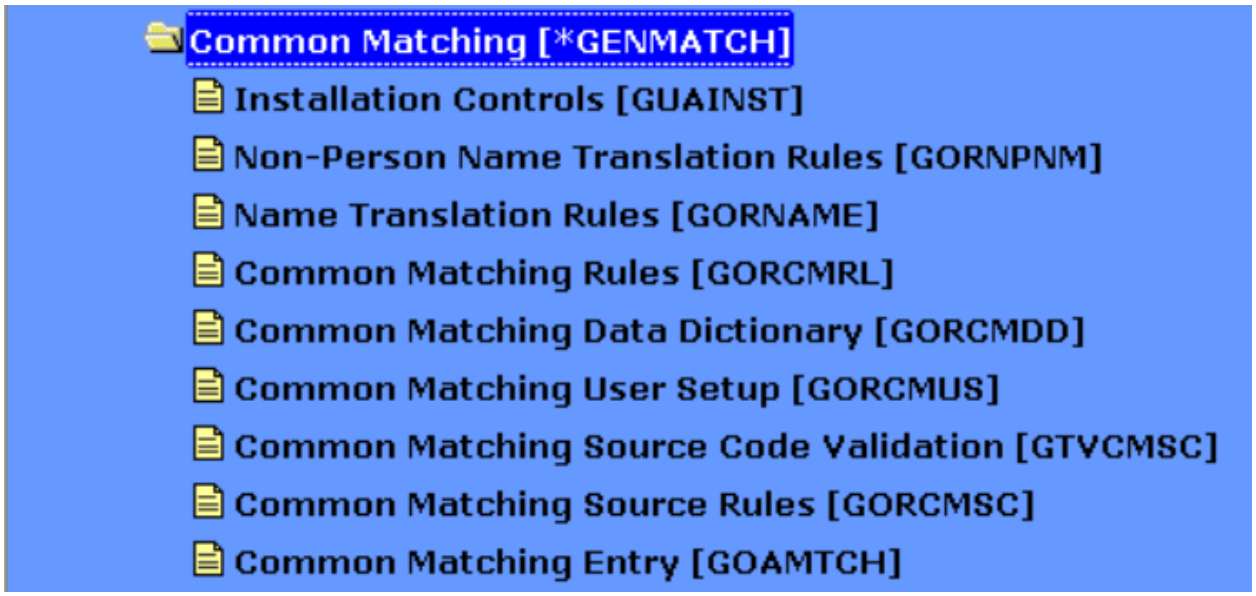
- When a user attempts to generate a new ID, they are taken automatically to the GOAMTCH form where they enter critical data
- The Common Matching process searches the database according to the source rule used to determine if the Person/Non-Person already is in the database
- The user can then review the results and select the ID, update an existing ID or create a new ID for the entity
- Common Matching is used in batch data load processes and online forms that are used to create new person or non-person records (e.g., PPAIDEN)
- Can have separate rules for online vs. batch processing
- Can use matching process for either persons or non-persons

HR Forms that Use Common Matching

- PPAIDEN – Payroll Identification
- PEAHIRE – Quick Hire
- PEA1PAY – One Time Payments

Location

Common Matching Forms are located in the System Functions / Administration Menu under the General Menu.



Further information on Common Matching

- Release Guides
- User Manuals
- Virtual Class
- Common Matching Handbook

Self Check

Directions

Use the information you have learned in this workbook to complete this self check activity.

Exercise 1

Find out information about the table PTRBDCA.

a) Examine the table structure by using the describe command. (List the correct command here.)

b) Who is the owner?

c) What type of table is it?

Exercise 2

Find out information about the table NBBPOSN.

a) Examine the table structure by using the describe command. (List the correct command here.)

b) Who is the owner?

c) What type of table is NBBPOSN?

Exercise 3

What is the next PIDM to be assigned?

Exercise 4

What is the PIDM for the person with the ID# 710000011?

Exercise 5

What is the PIDM and ID# for the person with SSN# 111-22-3333?

Banner 8 Common Enhancements

Introduction

This section provides an overview of the enhancements in Banner General 8.0 and Banner Human Resources 8.0.

Objectives

At the conclusion of this chapter, participants will be able to:

- identify enhancements to PIN security
- identify enhancements related to supplemental data
- identify enhancements made for Internationalization purposes
- identify enhancements made for partial data logging
- identify enhancements made for tab-level security

Internationalization Enhancement

Internationalization

Changes have been made to Banner Human Resources with the 8.0 release that enable international usage. These changes include the expansion of currency amount, address, phone, and name fields. These Internationalization enhancements will reduce the amount of custom modifications necessary to make Banner usable under those circumstances.

Unicode Support

Banner now supports the Unicode international character set through the character standard UTF8. The Oracle database will be converted to UTF8 as part of the Banner 8.x installation process.

Additional IDs

The Additional Identification Table (GORADID) allows storage of unlimited extra IDs for a person in Banner. Each Additional ID must be assigned an ID type, which is set up on the Additional Identification Type Validation (GTVADID) form and table.

Refer to the *Banner General 8.0 Release Guide* for more information on using additional IDs.

Expanded fields

Many fields, including the following, have been expanded on Banner tables and forms so as to accommodate longer data values.

- Name
- Address
- Telephone number
- E-mail address
- ID
- Currency amount
- Currency rate

Refer to the *Banner General 8.0 Release Guide* for more information on internationalization enhancements.

Partial Data Masking

Enhancements

The ability to partially mask a field, which was introduced initially in Banner 7.0, has been extended in Banner 8.x to character fields. You can allow a specified number of characters at one side of a field to remain readable while masking the remainder of a value.

To support partial character masking, two new fields (**Partial Character Mask** and **Partial Unmasked Length**) have been added to the Data Display Mask Rules Form (GOTDMSK).

Please refer to the *Banner General 8.x Release Guide* for more information on these changes.

Supplemental Data Engine

Enhancements

The Supplemental Data Engine allows storage of additional data that are not part of the existing Banner data model. Examples of the types of data affected are comments fields to record miscellaneous notes or data that has been translated into various languages.

No customization of Banner forms or tables is needed to capture and use additional data with SDE. The new data is displayed in a popup window, the Supplemental Data Window, and is stored in a supplemental data table. Because no customization is needed, supplemental data is generally not impacted by Banner upgrades.

Each supplemental data record created through SDE is tied to a specific Banner table, so any forms using that table will be able to access the same data through the Supplemental Data Window. Through SDE you can create additional fields associated with a specific Banner table but stored in a separate table, the Supplemental Data Table.

Although it is natural to think of supplemental data in terms of extra fields on Banner forms, SDE is tied to forms only indirectly.

There are several limitations of SDE, including the following.

- Not all tables, blocks, and forms work with SDE.
- The combination of SDE with Virtual Private Database (VPD) for use in Multi-Enterprise Processing is currently not supported.
- Masking is not currently supported with SDE.

Please refer to the *Banner General 8.x Release Guide* for more information on these changes.

The Data Dictionary

Introduction

This section discusses the Data Dictionary and its components.

The Data Dictionary

Purpose

How do you get more information about the structure and content of tables?

How do you find out about indexes, primary keys, and foreign keys?

How do you find out about table relationships?

Definition

- A read-only reference of tables and views about the database
- Stores information about both the logical and physical structure of the database

Objects and events displayed

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

Tables

- 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

The Data Dictionary: Examples

Example 1

```
SQL> SELECT table_name
       FROM dict
      WHERE table_name like 'ALL%';

TABLE_NAME
------------------------------
ALL_COL_COMMENTS
ALL_CONSTRAINTS
ALL_SYNONYMS
ALL_TABLES
ALL_TAB_COLUMNS
ALL_TAB_COMMENTS

 …
```

Example 2

```
SQL> SELECT comments
       FROM all_tab_comments
      WHERE table_name = 'PTRECLS';

COMMENTS
--------------------------------
Employee Class Rule Table
```

Example 3

```
SQL> SELECT comments
      FROM all_col_comments
     WHERE column_name = 'PTRECLS_BCAT_CODE';

COMMENTS
----------------------------------------
DEFAULT BENEFIT CATAGORY: A Benefit Cata
gory for which employees in this Employe
e Class will be eligible.  Additional Be
nefit Catagories may be added on Page 2,
 however this category will default to t
he Employee Form (PEAEMPL).
```

Example 4

```
SQL> SELECT column_name
       FROM all_tab_columns
      WHERE owner = 'PAYROLL'
        AND column_name like '%ORGN%'


COLUMN_NAME
------------------------------
PABREQU_ORGN_CODE
PEBEMPL_ORGN_CODE_HOME
PEBEMPL_ORGN_CODE_DIST
PERACTL_ORGN
PERAPPT_ORGN
PERCAPL_ORGN
PERCAPR_ORGN
PEREHIS_HOME_ORGN
. . .                  74 rows selected
```

Example 5

```
SQL> SELECT text
       FROM all_views
      WHERE  view_name = 'PEVLEAV';

TEXT
----------------------------------------
SELECT  PERLEAV_PIDM,
        PERLEAV_LEAV_CODE,
        PTRLEAV_LONG_DESC,
        PTRLEAV_SHORT_DESC,
        PERLEAV_BEGIN_BALANCE,
        PERLEAV_ACCRUED,
        PERLEAV_TAKEN,
        PERLEAV_DATE_AVAIL,
        PERLEAV_HRS_BANKED
FROM    PERLEAV,
        PTRLEAV
WHERE   PTRLEAV_CODE (+)   = PERLEAV_LEAV_CODE
```

Indexes, columns and constraints

- ALL_INDEXES – descriptions of indexes
- ALL_IND_COLUMNS – lists the columns that make up an index
- ALL_CONSTRAINTS – descriptions of constraints
- ALL_CONS_COLUMNS – lists the columns that make up a constraint

Example 6

```
SQL>  SELECT constraint_name,status
        FROM all_constraints
       WHERE table_name = 'PTREARN'
         AND constraint_name not like 'SYS%'

CONSTRAINT_NAME                STATUS
------------------------------      --------
PK_PTREARN                           ENABLED
FK1_PTREARN_INV_PTV1099_CODE         ENABLED
FK1_PTREARN_INV_PTVERGR_KEY          ENABLED
```

Primary key constraints

Primary key constraints are named as follows:

```
PK_tablename
```

Example: *PK_PTREARN*

Foreign key constraints

Foreign key constraints are named as follows:

```
FKn_tablename_INV_primarytablename_CODE (or KEY)
```

Example: *FK1_PTREARN_INV_PTV1099_CODE*

Example 7

```
   SELECT constraint_name, column_name
     FROM all_cons_columns
    WHERE table_name = 'PTREARN'
      AND constraint_name not like 'SYS%'
ORDER BY constraint_name, column_name;

FK1_PTREARN_INV_PTV1099_CODE
PTREARN_1099_CODE

FK1_PTREARN_INV_PTVERGR_KEY
PTREARN_ERGR_CODE

PK_PTREARN
PTREARN_CODE
```

Example 8

```
SQL> SELECT index_name, uniqueness, status
       FROM all_indexes
      WHERE table_name = 'PTREARN';

INDEX_NAME                          UNIQUENES  STATUS
------------------------------      ---------  --------
PK_PTREARN                          UNIQUE     VALID
PTREARN_KEY2_INDEX                  NONUNIQUE  VALID
```

Index naming

The primary index is named as follows:

```
PK_seven-character table name
```

Example: *PK_PTREARN*

Each additional index is numbered numerically starting with 2, after key:

```
Seven-character table name_key2_index
```

Example: *PTREARN_KEY2_INDEX*

```
Seven_character table name_key3_index, etc.
```

Example 9

```
SQL> SELECT index_name, column_name
       FROM all_ind_columns
      WHERE table_name = 'PTREARN'
   ORDER BY index_name, column_name;

INDEX_NAME                COLUMN_NAME
----------------------    ------------------------------
PK_PTREARN                PTREARN_CODE
PTREARN_KEY2_INDEX        PTREARN_BASE_SAL_IND
```

The Data Dictionary: GURPDED

GURPDED Procedure

- Extracts Data Dictionary information into a printable report
- Run from GJAPCTL - the Job Submission Form in the General Product
- Enter parameters:
  - Table name
  - Table owner

GURPDED output

- Output = Technical Addendum
  - To DATABASE
  - View or Print from GJIREVO

The Data Dictionary: ERDs

Entity Relationship Diagrams (ERDs)

Entity Relationship Diagrams for all modules are available for download from the SunGard Higher Education Customer Support Center.

- Log into the Customer Support Center (**www.sungardhe.com**)
- Click on **Documentation and Download Center** (under Self Service)
- Select **Banner – Human Resources** from the Select Product menu and click **List Available Documentation**
- Select the **ERD** checkbox for your version of Banner and click **List Documentation Files From Selected Folders**
- Select and download the appropriate files

Contents

ERDs are created from the following data:

- Data Dictionary
- Forms triggers
- Database procedures

Created by Cast (by Enlighten)

Self Check

Directions

Use the information you have learned in this workbook to complete this self check activity.

Exercise 1

Find the name of a repeating table with employee leave balances.

Exercise 2

Find the name of the validation table for citizenship code.

Exercise 3

Retrieve the table names that your user account owns.
(View: USER_TABLES)

Exercise 4

Retrieve the table names to which your user account has access.

(View: ALL_TABLES)

Exercise 5

Retrieve the comments for the PEBEMPL table.

(View: ALL_TAB_COMMENTS)

Exercise 6

Prompting for a table name, retrieve the associated columns and column comments.

(View: ALL_COL_COMMENTS)

Exercise 7

How many validation tables exist for payroll? How many for position control?

(View: ALL_TABLES)

Exercise 8

How many payroll history tables are there?

(View: ALL_TABLES)

Exercise 9

What are the conditions that must be met before a record can be inserted into the PEBEMPL table (hint: table relationships)?

(View: ALL_CONS_COLUMNS)

Exercise 10

What Indexes exist for PHREARN?

(View: ALL_INDEXES)

Exercise 11

List the columns of the indexes for PHREARN.

(View: ALL_IND_COLUMNS)

Human Resources Objects

Introduction

This section discusses tables, forms and class hierarchies in Banner Human Resources.

Tables

Banner data is stored in ORACLE tables.

Banner tables, like all Banner objects, adhere to the Banner object naming conventions.

Table types

The three basic types of HR tables are:

- Application Tables
- Base Tables
- Repeating Tables
- Temporary Tables
- Validation Tables
- Rules Tables

Application Tables: Base Tables

There can be only one occurrence of the logical key.

Example: PEBEMPL (Employee Base Table)

- The logical key is PEBEMPL_PIDM (person identification master)
  - One record for each person (employee)

Example: NBBPOSN (Position Base Table)

- The logical key is NBBPOSN_POSN
  - One record for each position

Application Tables: Repeating Tables

There can be multiple occurrences of the logical key.

Example: PERLEAV (Leave Balances Repeating Table)

- The logical key is PERLEAV_PIDM
- Multiple records – each employee has a record for each of their leave types

Example: SPRIDEN (Common Identification Repeating Table)

- The logical key is SPRIDEN_PIDM
- Multiple records – each person will have at least one current record
- spriden_change_ind is null
- There may be multiples for any previous name or id

Application Tables: Temporary Tables

- Intermediate internal holding area for Banner reports and processes
- Same naming conventions as application tables
- Example: PHRTDED
  - System Maintained

Rules Tables

Key column

- *tablename*_code
- Example: PTRECLS_CODE

Description column

- *tablename*_desc
- Example: PTRECLS_SHORT_DESC

Fields with a limited number of enterable values

- *tablename_column_name*_ind
- Example: PTRECLS_BUDGET_ROLL_IND

Validation Tables

A validation table and its corresponding form will have the same name.

- Follows the same column naming conventions as rule tables

Validation tables have at minimum these three columns:

- *tablename*_code
- *tablename*_desc
- *tablename*_activity_date

Human Resources Forms

Forms

Users interact with the Banner database through the use of forms.

Banner forms, like all Banner objects, adhere to the Banner objects naming convention.

Form types

There are six types of HR Forms:

- Menu
- Application
- Validation
- Rules
- Query
- Inquiry

Menu forms

- Will not follow BANNER objects naming convention
- List all related forms
- Outline the System
- Example: HRSEMPLOYEE (Employment Administration Menu)

Application Forms

- Enter data
- Update data
- Query the System
- Example: PEAEMPL (Employee Form)

Validation Forms

- List all possible values for a given field
- Data entry allowed
- Table and form names are the same
- Second and third characters are *TV*
- Have code, description, and activity date fields
- Example: PTVESKL (Employee Skills)

Rule Forms

- Define use of variables, objects, and application
- Example: PTRECLS (Employee Class)
- Example: PTRBDCA (Benefits and Deductions)
- Example: NTRPCLS (Position Class)

Query Forms

- Third character is a Q
- Must be called by another form
- Look-up information only
- Information cannot be changed
- Example: PTQECLS (Employee Class Query Form -
  called via clicking the **Search** icon in key block of PTRECLS)

Inquiry Forms

- Third character is an **/**
- Query data and return to another form
- Information cannot be changed
- Form can be accessed from any menu
- Example: PEIETOT (Employee Year To Date Totals Form)
- Example: PEIDTOT (Employee Year to Date Deductions Form)

Human Resources Hierarchy

Hierarchy

Banner Human Resources uses a hierarchy of classes to ease data entry.

By associating an employee with a class or grouping, class information can be automatically entered by the System.

These classes and groupings are defined in the rules and validation tables.

Diagram 1

Diagram 2

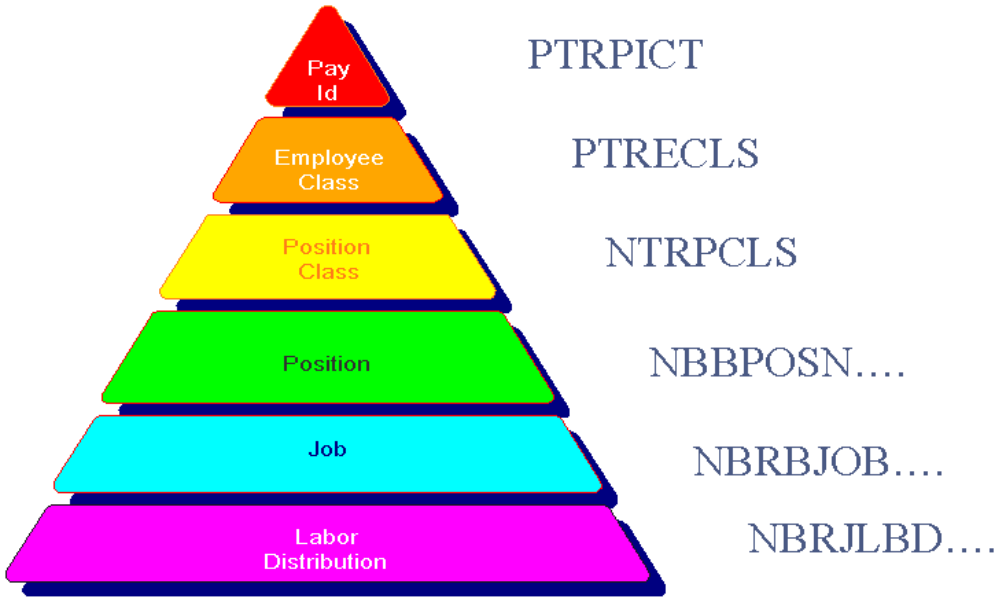
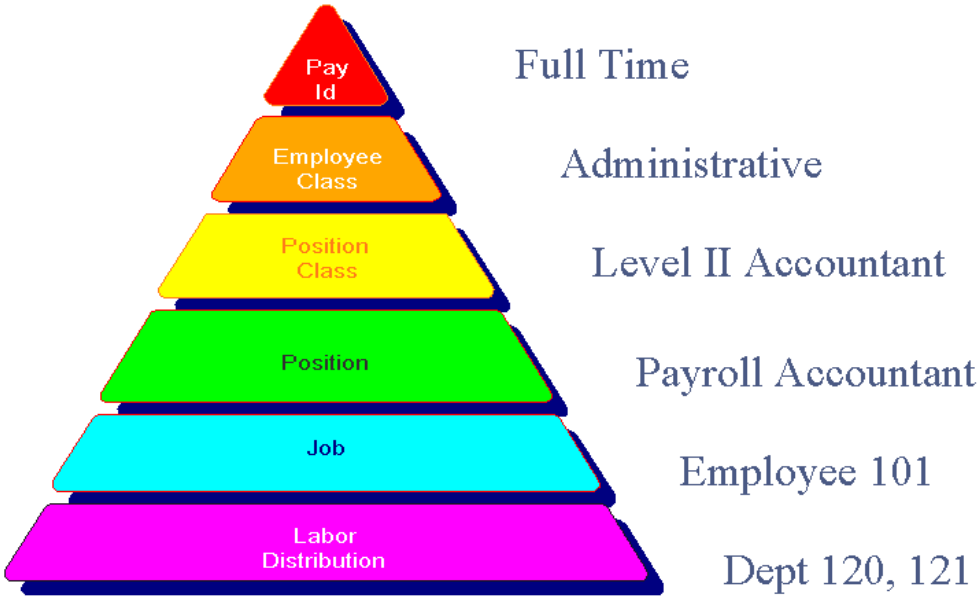


Diagram 3



Data defaults

Data defaults down (NOT up) the pyramid.

Example: A Labor Distribution Override at the Job level will not change the labor distribution on the position

Data defaults once.

Example: A Labor Distribution Change on a position will NOT change the labor distribution of employees already assigned to that position

Banner System Overview

Introduction

This section provides a brief diagram of the connections between the Banner systems.

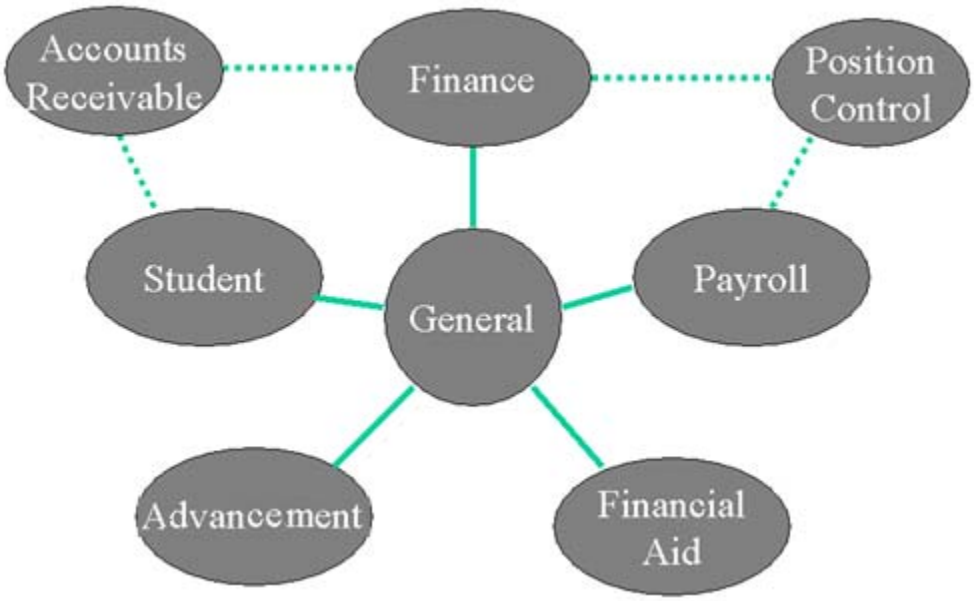
Banner System Overview

Overview

Where does Banner Human Resources fit into the entire Banner System?

Diagram

The Banner Systems:



Human Resources Components

Introduction

This section provides details on the modules that make up the Banner Human Resources system.

Biographic – Demographic

Biographic/ Demographic Information

- Establish a unique identifier for each individual
- Maintain:
  - Biographic information
  - Educational background
  - Professional qualifications
- Part of the General Module
- Chapter 8 of the Human Resources Using Banner Guide

Core Application Forms

- PPAIDEN
  - Establishes a unique identifier PIDM
  - Maintains biographic and demographic information
- PPAGENL
  - Maintains professional qualifications
- GXADIRD
  - Maintains direct deposit information
- GOAINTL
  - Maintains international data

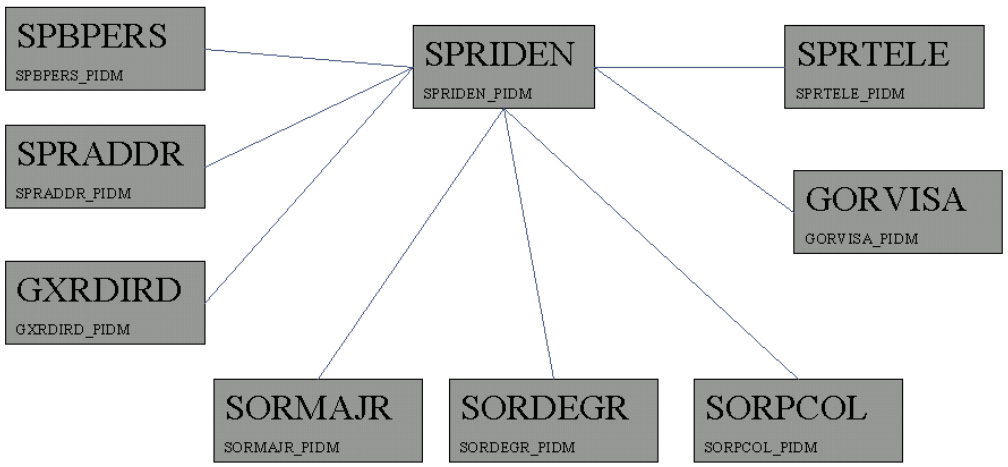
Rule Forms

- PTRCERT
  - Certification Code

Validation Forms

- STVATYP
  - Address Type Code
- STVCITZ
  - Citizen Type Code
- STVETHN
  - Ethnic Code
- STVSBGI
  - College Codes

Diagram



HR Required Fields – SPRIDEN

- SPRIDEN_PIDM
- SPRIDEN_ID
- SPRIDEN_FIRST_NAME
- SPRIDEN_LAST_NAME
- SPRIDEN_CHANGE_IND
    - At least one record with a null
    - PPAIDEN maintains
- SPRIDEN_ENTITY_IND
    - = 'P' for person ('C' for company)
    - PPAIDEN maintains
- SPRIDEN_ACTIVITY_DATE **

HR Required Fields – SPBPERS

- SPBPERS_PIDM
- SPBPERS_SSN
- SPBPERS_BIRTH_DATE
- SPBPERS_ETHN_CODE
- SPBPERS_SEX
- SPBPERS_CITZ_CODE

HR Required Fields – SPRADDR

An employee must have at least one active address record for the address type(s) designated for payroll.

- SPRADDR_PIDM
- SPRADDR_ATYP_CODE
- SPRADDR_SEQ_NO
- SPRADDR_STREET_LINE1
- SPRADDR_CITY
- SPRADDR_STAT_CODE
- SPRADDR_ZIP

Person records

- Each person will have a SPRIDEN record with a SPRIDEN_CHANGE_IND of null
- Other records for the person will have a value in the change_ind indicating the type of change (N)ame or (I)D
- Each person will have one SPBPERS record

Exercises

How many entities are there in SPRIDEN?

```
SELECT count(*) FROM SPRIDEN;
```

How many of those entities are people?

```
SELECT count(*) FROM SPRIDEN
 WHERE SPRIDEN_ENTITY_IND = 'P';
```

How many of these represent current information for those people?

```
   SELECT count(*) FROM SPRIDEN
    WHERE SPRIDEN_ENTITY_IND = 'P'
      AND SPRIDEN_CHANGE_IND is null;
```

Employment Administration

Purpose

Maintain:

- Employee's status
- Hire dates
- Benefit category (BCAT)
- Leave category (LCAT)
- Home Department

Chapter 9 of Human Resources Using Banner Guide

Core Application Forms

- PEAEMPL

  Establishes employee information, status, benefit and leave categories

- PEAREVW

  Maintain performance review information

- PEALEAV

  View and maintain leave balance records (IF leave by employee method chosen on PTRINST – populated by PEAEMPL)

Rule Forms

- PTRLCAT

  Leave Categories

- PTRECLS

  Employee Class

- PTRTREA

  Termination Reason

Validation Forms

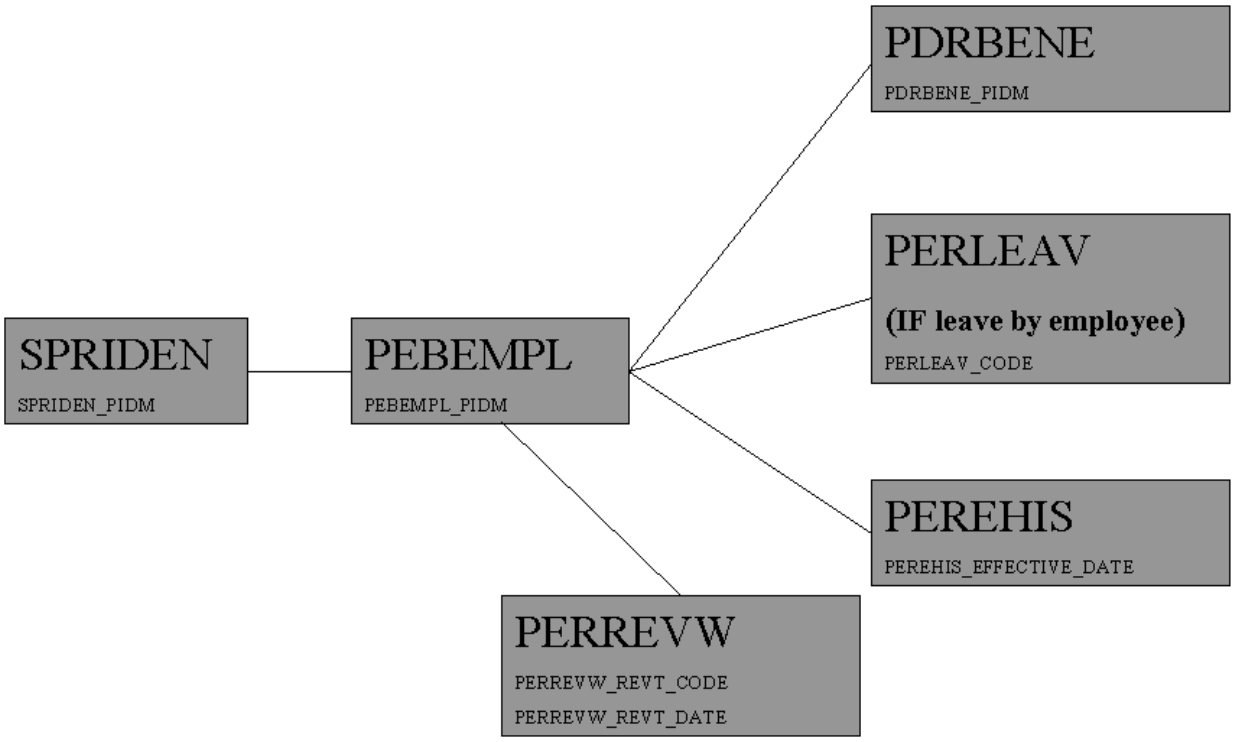
- FTVCOAS

  Chart of Accounts

- FTVORGN (PTVORGN)

  Organization Codes

Diagram



Person records

- A row in PEBEMPL defines a person as an employee
- Each employee will have one PEBEMPL record
- A history of changes made to PEBEMPL through PEAMPLE is stored in PEREHIS
- Must have required Bio-Demo data first
- Much of the required data for the employee record defaults from the HR Hierarchy (Rules and Validation Tables)
- Other required fields are defaulted by the form PEAEMPL, but most can be overridden

HR Required Fields – PEBEMPL

- PEBEMPL_PIDM
- PEBEMPL_EMPL_STATUS
- PEBEMPL_COAS_CODE_HOME
- PEBEMPL_ORGN_CODE_HOME
- PEBEMPL_COAS_CODE_DIST
- PEBEMPL_ORGN_CODE_DIST
- PEBEMPL_ECLS_CODE
- PEBEMPL_LCAT_CODE
- PEBEMPL_BCAT_CODE
- PEBEMPL_FIRST_HIRE_DATE
- PEBEMPL_CURRENT_HIRE_DATE
- PEBEMPL_ADJ_SERVICE_DATE
- PEBEMPL_SENIORITY_DATE
- PEBEMPL_FLSA_IND
- PEBEMPL_INTERNAL_FT_PT_IND

Exercise 1

Which of these columns likely have foreign key constraints?

```
SELECT constraint_name
  FROM all_constraints
 WHERE table_name = 'PEBEMPL'
   AND constraint_name like 'FK%';
```

Exercise 2

What do all those dates mean?

```
SELECT column_name, comments
  FROM all_col_comments
 WHERE table_name = 'PEBEMPL'
   AND column_name like '%DATE';
```

Exercise 3

What do all those dates mean?

```
SELECT comments
  FROM all_col_comments
 WHERE table_name = 'PEBEMPL'
   AND column_name like '%DATE';
```

Exercise 4

How many employees are defined?

```
SELECT count(*) FROM PEBEMPL;
```

How many of those employees are active?

```
SELECT count(*) FROM PEBEMPL
 WHERE PEBEMPL_EMPL_STATUS = 'A';
```

Position Management

Purpose

- Define positions
- Assign positions to budgets
- Assign positions to labor distribution
- Maintain position history

Chapter 14 of Human Resources Using Banner Guide

Core Application forms

- NBAPOSN

  Defines all positions within a position classification and fiscal year

- NBAPBUD

  Assign positions to budgets and FOAPAL

- NBAFISC

  Maintain current fiscal year

Rule forms

- NTRPCLS

  Position Class

- NTRSALA

  Salary rate

- NTRSGRP

  Salary Group

Validation Forms

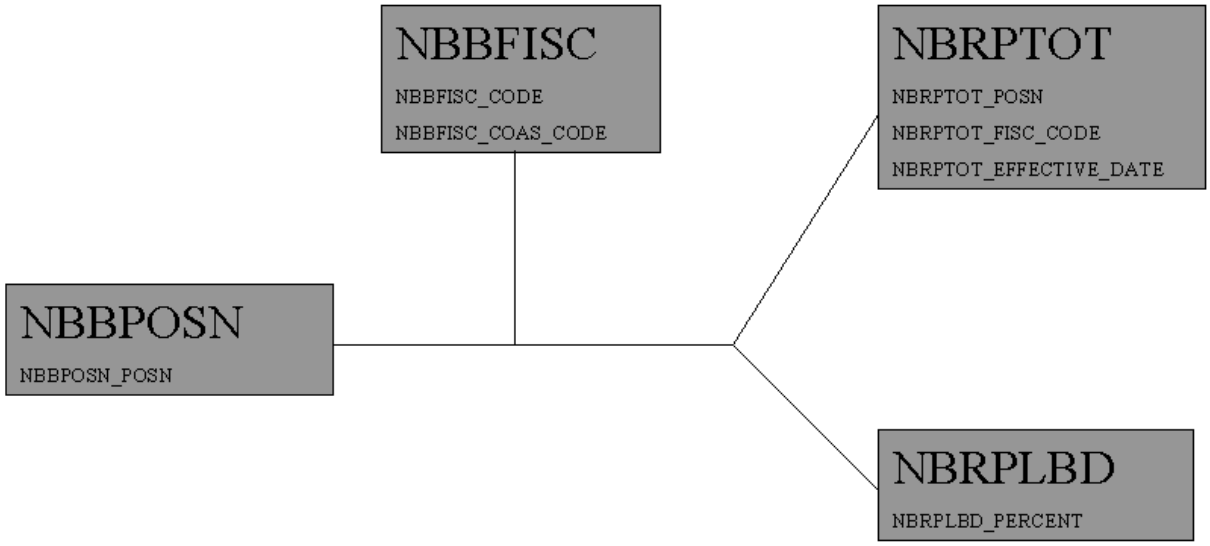
- FTVORGN

  Organization

- FTVCOAS

  Chart of Accounts

Diagram



Position records

- One record in NBBPOSN for each position
- At least one NBRPTOT record for each position/fiscal year combination
- At least one NBRJLBD record for each position/fiscal year combination

Compensation Administration

Purpose

Defaults come from HR hierarchy.

Maintain:

- Employee's Title
- Employee's Salary
- Compensation History

Chapter 10 of Human Resources Using Banner Guide

Forms

- NBAJOBS

  Defines the job for a particular employee with begin and end dates, title, status, labor distribution, and salary information

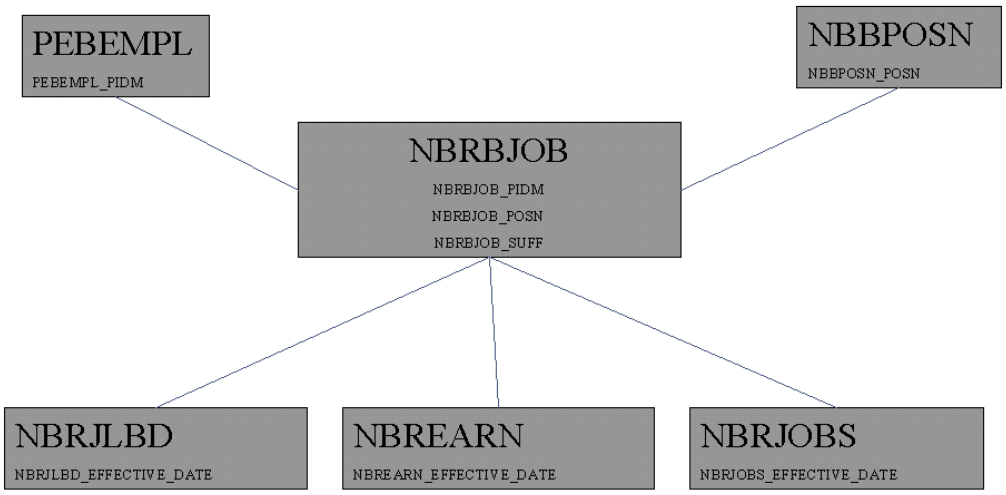
- PEIJHIS

  Employee Job History Form

- PEALEAV

  View and maintain leave balance records (IF leave by job method chosen on PTRINST – populated by NBAJOBS)

Diagram



Position/ employee records

- One record in NBRBJOB for each position/employee combination
- At least one record in NBRJOBS for each position/employee combination
- Additional records for subsequent changes to job information
- Example: Salary, Title

Position/ fiscal year / employee records

- At least one NBRJLBD record for each position/fiscal year/employee combination
- Defaults from NBRPLBD, but can be overridden
- Records in NBREARN are defaulted depending on ECLS set up. It can be overridden

Requirements

- The person must have an active employee record to have an active job.
- The position must be active.
- An employee can have multiple active jobs at any given time.

Benefits and Deductions

Purpose

- Maintain Employee's benefits and deductions
- Eligibility administration
- Maintain Beneficiary/Dependent Information
- Chapter 13 of Human Resources Using Banner Guide

Core Application Forms

- PDADEDN

  Establish/Maintain deductions for benefits, taxes and other withholdings
- PDABENE

  Beneficiary Form
- PDABCOV

  Beneficiary coverage Form

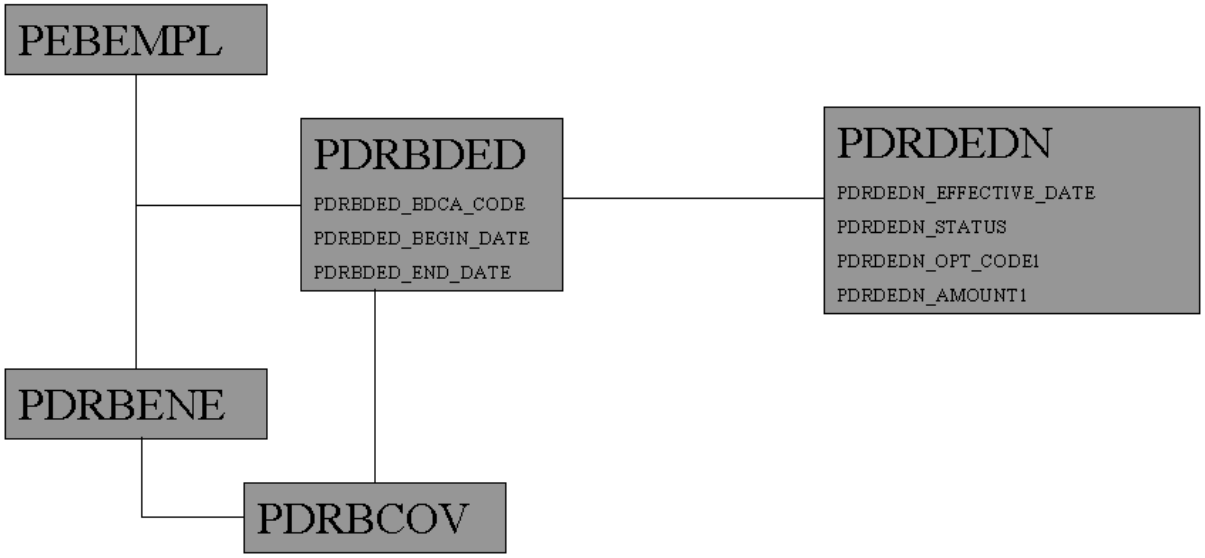
Rule Forms

- PTRBCAT

  Benefit Category
- PTRBDCA

  Benefit/Deduction Code

Diagram



Deduction/ employee records

- One record in PDRBDED for each deduction/employee combination
- At least one record in PDRDEDN for each deduction/employee combination
- Additional records for subsequent changes to deduction
- Example: Plans, options, amounts

Requirements

- Must be an active employee to set up deductions.
- Eligibility is driven by benefit categories (BCAT) in PTRBCAT.
- 'Self' beneficiary records are created when employee record is created with PEAEMPL

Establishing Combined Deduction Limits

Banner Human Resources provided the ability to check and ensure that combinations of employee and employer contributions towards employee retirement plans were within the elective deferral maximum limit set by the IRS. This functionality has now been extended to check for the employee's

- Annual contribution limit
- Annual compensation limit
- Catch-up contributions
- 15-year rule plan.

Application forms

- PTRBDCL

  Specify annual dollar limits for combinations of employee retirement plans, 50+ catch-up contributions, and the 15-year rule plan.

Altered forms

New fields related to establishing combined deduction limits have been added to:

- PTRBDCA
- PDADEDN
- PEIDHIS
- PTRBDMC

For more details on this enhancement, please refer to the *Banner 8 Human Resources Release Guide*.

Leave Administration

Purpose

- Leave can be tracked by employee or by job.
- Choose the method of leave on PTRINST form
- Different set of leave tables for employee or job tracking
- Chapter 9 of Human Resources Using Banner Guide

Forms

- PEALEAV – View, track, and update balances for each leave code
- PEILHIS – View leave balance history
- PHIACCR – View leave accrual history

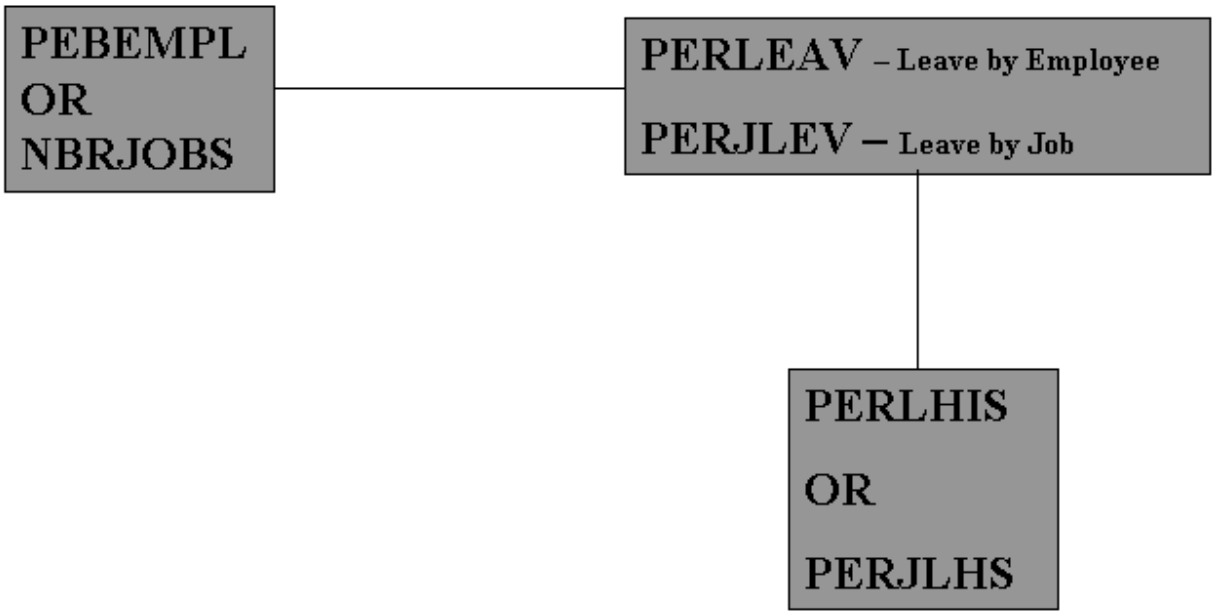
Leave by Employee

PEAEMPL – Leave balance records are created for those leave types for which their leave category (LCAT) on PEAEMPL dictates

Leave by Job

NBAJOBS – Leave balance records are created for those leave types for which their leave category (LCAT) on NBAJOBS dictates

Diagram


PEBEMPL
OR
NBRJOBS
PERLEAV – Leave by Employee
PERJLEV – Leave by Job
PERLHIS
OR
PERJLHS

Time Entry and Payroll Processing

Time Entry

- Collect time sheet information
- Validate earnings codes, hours and special rates to ensure eligibility of earnings type by employee group
- Payroll Processing will be addressed in more detail later
- Chapter 16 of Human Resources Using Banner Guide

Application Forms

- PHAHOUR - Online Time Entry
  - Directly updates the PHR% tables when successfully saved
- PHATIME - Time entry with approvals
  - Requires set up of users and routing queues
  - Updates to PHR% tables once transaction successfully completes the routing queue
- PHAMTIM - Mass time entry
  - Uses temporary table PHRMTIM
  - Updates to PHR% tables after successful completion of PHPMTIM process

Applicant Tracking

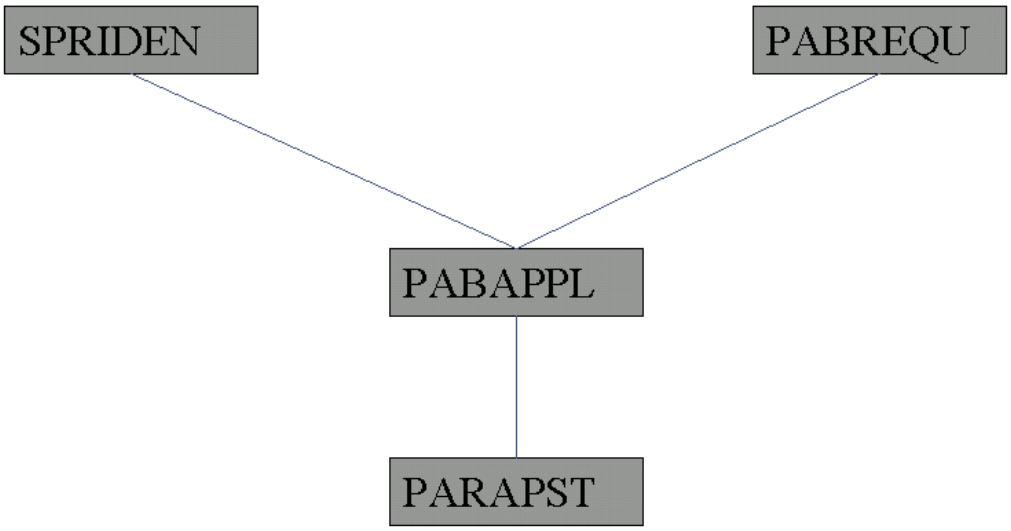
Purpose

- Create and maintain detailed requisitions for vacant positions
- Create and maintain applicant records for existing positions
- A PIDM is required in order to be an applicant

Application Forms

- PAAAPPL – Applicant Information Form
- PAAREQU – Requisition Form
- Chapter 7 of Human Resources Using Banner Guide

Diagram



Employee Relations

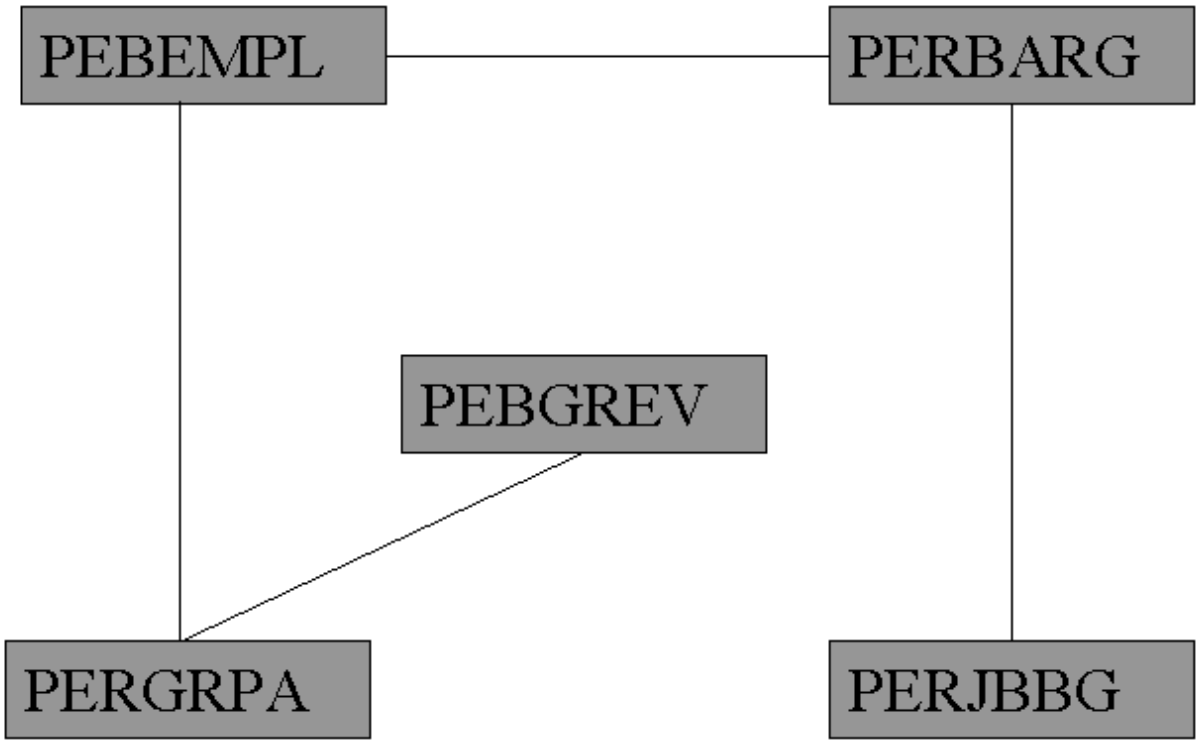
Purpose

- Bargaining Unit Membership
- Bargaining unit relations
- Seniority Tracking
- Grievance tracking

Application forms

- PEABARG – Employee/Job Labor Relations
- PEAGREV – Employee Relations Grievance
- Chapter 11 of Human Resources Using Banner Guide

Diagram



Health and Safety

Purpose

- Record and report employee health information
- Satisfy OSHA requirements

Application Forms

- PEAHSIN – Health and Safety Incident Form
- Driving Table - PEBHSIN
- Chapter 12 of Human Resources Using Banner Guide

Electronic Approvals (EPAFs)

Purpose

- Efficient approval signature process
- Support the movement towards a paperless office
- Chapter 18 of Human Resources Using Banner Guide

Application Forms

- NOAEPAF – Personnel actions form
- Driving Table - NOBTRAN

Process

- NOPEAMA – Mass Apply Process

Race and ethnicity

In Human Resources Release 7.1.2, the functionality of the Identification Form (PPAIDEN) was enhanced to associate a person with multiple race categories. In Human Resources 8.0, this functionality has been extended to support data entry of race and ethnicity information through EPAFs.

For more information on this, please refer to the *Banner 8 Human Resources Release Guide*.

Interior

Introduction

This section discusses effective dating and APIs in Banner Human Resources.

Interior

Components

- Effective Dating
- HR APIs

Effective Dating

Purpose

- Maintains history
- Allows for future dated personnel actions

Example

```
     select nbrjobs_pidm, nbrjobs_posn, nbrjobs_suff,
        nbrjobs_effective_date, nbrjobs_desc,
        nbrjobs_status
      from nbrjobs
     where nbrjobs_pidm = 408
   order by nbrjobs_effective_date;

   408 S00001 00 31-AUG-96 Federal Work Study (Pooled)    A
   408 S00001 00 30-SEP-96 History Dept. Office Clerk-CWS A
   408 S00001 00 11-MAY-97 History Dept. Office Clerk-CWS T
   408 S00001 00 31-AUG-97 History Dept. Office Clerk-CWS A
```

Exercise 1

What job record is active as of today?

```
select max(nbrjobs_effective_date)
  from nbrjobs
 where nbrjobs_effective_date <= SYSDATE
   and nbrjobs_pidm = 408;
```

31-AUG-97

Exercise 2

What job record is active as of Oct. 1, 1996?

```
select max(nbrjobs_effective_date)
  from nbrjobs
 where trunc(nbrjobs_effective_date) <=  '01-OCT-96'
   and nbrjobs_pidm = 408
   and nbrjobs_status = 'A';
```

30-SEP-96

A Nested Select

```
SELECT nbrjobs_pidm, nbrjobs_posn, nbrjobs_suff,
       nbrjobs_desc, nbrjobs_status
  FROM nbrjobs a
 WHERE nbrjobs_pidm = 408
   AND a.nbrjobs_status = 'A'
   AND a.nbrjobs_effective_date =
   (SELECT max(nbrjobs_effective_date)
      FROM nbrjobs b
     WHERE trunc(nbrjobs_effective_date) <= '01-OCT-96'
       AND b.nbrjobs_pidm = a.nbrjobs_pidm
       AND b.nbrjobs_posn = a.nbrjobs_posn
       AND b.nbrjobs_suff = a.nbrjobs_suff);
```

408 S00001 00 30-SEP-96 History Dept. Office Clerk-CWS A

HR tables

Some HR tables with effective dating logic:

- NBREARN
- NBRJLBD
- NBRJOBS
- PDRDEDN

Self Check

Directions

Use the information you have learned in this workbook to complete this self check activity.

Exercise 1

What is the current effective date of the active job for Lawrence Mitchell? (NBRJOBS)

Exercise 2

What was the effective date of the same job as of Jun 1, 1994? (NBRJOBS)

Exercise 3

List name, position and annual salary as of Oct. 23, 1996 for all employees with jobs. (NBRJOBS, SPRIDEN).

Human Resources APIs

Purpose

Application programming interfaces (APIs) facilitate the integration of Banner with other applications on a campus.

Human Resources related APIs

- PPAIDEN form
  - Populates tables
    - SPRIDEN
    - SPBPERS
    - SPRADDR
    - SPRTELE
- PEAEMPL form
  - Populates table PEBEMPL

Human Resources / Banner System Interfaces

Introduction

This section discusses the touch points between the Banner Human Resources system and other Banner systems.

Interfaces and Integration with Other Banner Systems

Interfaces

- Advancement Interface
- Finance Interface
- Student Integration

Chapter 6 of Human Resources Using Banner Guide

Alumni Pledge Payments

Advancement processing

Deductions can be gifts to the institution

- Deduction code set up by HR (PTRBDCA)
- Advancement sets up pledge (AGAPLDG) and installments (AGAPINS)
- Pledge information appears in AGCFDED and PEAFDED
- HR uses form PEAFDED in HR to activate the deduction
- Check the **Received Signature** box
- Assign appropriate BDCA code
- During payroll processing, PHPUPDT creates pledge payment records, (GURALMP), to be processed in Advancement
- Advancement runs Automatic Deduction Process (AGPALMP) to create pledge payments

Faculty Load Data

Faculty load data

- The General information form, PPAGENL, shares faculty history data with SIAFPER, and faculty academic history data with SIAFDEG
- The Employee form, PEAEMPL, shares personnel data with SIAFPER
- Faculty Load Analysis Report – displays salary info from HR
- The HR Faculty Load process, PEPFACL, updates HR with contract hours and FTE from Faculty Load

Automated Faculty Load and Compensation

Introduction

The new Automated Faculty Load and Compensation module merges faculty information in Banner Student and Human Resources systems to capitalize and deliver a robust, contiguous, and a comprehensive business process that gives institutions the power to automate the derivation and calculation of appropriate compensation packages for their full-time or part-time employed faculty members based on their individual work loads.

Using this module, you can now define rules for calculation of compensation packages in Banner Human Resources and Banner Student, and evaluate work loads and actual compensation packages for full-time as well as part-time faculty members in the module's new web interface on Employee Self-Service.

Requirements

At a minimum, the implementation of Automated Faculty Load and Compensation module requires the following Banner products:

- Banner Student
- Banner Human Resources
- Employee Self-Service
- Banner WebTailor

Changes

The translation of the module's business objective resulted in the following changes within Banner Human Resources:

- New validation forms and rule forms have been added to provide for changes in the calculation of compensation rates by this module. These include:
  - Faculty Compensation Level Code Validation Form (PTVFLCL)
  - Incremental Compensation Code Validation Form (PTVFLIC)
  - Faculty Load Incremental Compensation Rules Form (PTRFLIC)
  - Faculty Load and Compensation Institution Rules Form (PTRFLAC)
  - Faculty Load and Compensation Non-Instructional Rules Form (PTRNIST)
  - Faculty Load Contract Type Control Rules Form (PTRFLCT)
- Existing rule forms in Banner HR as well as Banner Student have been modified.

  These include:
  - Installation Rules Form Form (PTRINST)
  - User Codes Rules Form Form (PTRUSER)
  - Roll-up Security (NSASPSC) Form (Banner® Finance Installed)
  - Faculty Action Tracking Form assigning Faculty Level (PEAFACT)
  - Faculty Assignment Form (SIAASGN).

  For more details refer to the *Banner Student Release Guide, Release 8.0*.
- Appropriate HR and user security rules have to be established for using this module. Consequently, the following security forms have been modified:
  - User Code Rules Form (PTRUSER)
  - Employee Class Rules Form (PSAECLS)
  - Organization Rules Form (PSAORGN)
  - Organizational Hierarchy Security Form (NSASPSC)
  - Employer Rules Form (PSAEMPR).

For more information regarding this new module and related changes, please refer to the *Banner 8 Human Resources Release Guide*.

Finance Interface Table Setup

Tables

- NTRFINI - HR/Finance Setup
- FTVCOAS - Chart of Accounts
- FTVFUND - Fund
- FTVORGN - Organization
- FTVACCT - Account
- FTVPROG - Program

Note: Finance items to check

- NTRFINI - Be sure that all Rule codes and Net Distribution FOAPAL elements are in place
- PTREARN - Be sure that all Earn codes have Labor Distribution Overrides (optional)
- PTRBDCA - Be sure that all Deduction codes have Labor Distribution Overrides (mandatory)
- FTMRUCL - Be sure that all 'H***' Rule codes with 'G026' Process codes have Payroll Clearing Account in Posting modifier

Finance Interface Processes

Processes

- NBAPOSN and NBAPBUD - build position budgets
- NBPBUDM and NHPFIN1 – run to build budget and encumbrance transactions
- PHPFEXP – run to build payroll expense transactions
- NHPFIN2 - insert transactions into GURFEED

New Hire Process

Introduction

This section contains an overview of the New Hire process in Banner Human Resources.

PPAIDEN

PPAIDEN form loads tables

- SPRIDEN PIDM, ID, Name...
- SPRADDR Address
- SPBPERS Birth Date, SSN, Ethnicity....
- SPRTELE Telephone
- SPREMRG Emergency Contact

PEAEMPL

PEAEMPL form loads tables

- PEBEMPL Employee Status, Hire Dates, Employee Classification, ...
- PERLEAV Leave Balances (if using leave by employee)
- PEREHIS Employee History
- PDRBENE Beneficiary

NBAJOBS

NBAJOBS form loads tables

- NBRBJOB — Position Number, Job Begin and End Date…
- NBRJOBS — Position Number, Effective Date, Title, Salary….
- NBREARN — Default Earnings Code, Default Hours…
- PERJHIS — Job History
- PERJLEV — Leave Balances (if using leave by job)

PDADEDN

PDADEDN form loads tables

- PDRBDED — Deduction Code (BDCA), Deduction Begin Date and End Date….
- PDRDEDN — Deduction Code (BDCA), Effective Date, Deduction Plan, Deduction Amounts…
- PERDHIS — Deduction History

New Hire Quick Set Up

- PEAHIRE
  - Allows you to move through a series of blocks that will load tables behind PPAIDEN, PEAEMPL, and NBAJOBS
  - Initial Set up only
- PDABDSU
  - Loads the tables behind PDADEDN
  - Initial Set up only

Self Check

Directions

Use the information you have learned in this workbook to complete this self check activity.

Exercise 1

Log in to Banner (TRNG), and access the Identification Form (PPAIDEN).

Exercise 2

Enter a new person into the System. (PPAIDEN)

Exercise 3

Identify the tables behind the form. (Select *Help -> Dynamic Help Query* from the Menu Bar)

Exercise 4

Make the new person an employee. (PEAEMPL)

Exercise 5

Give the new person a job. (NBAJOBS)

Exercise 6

Assign the person benefits. (PDABDSU, PDADEDN)

Payroll Process

Introduction

This section discusses the many steps involved in the Payroll Process in Banner Human Resources.

Key Concepts

Dispositions

- Status identifiers
- Every employee is assigned a disposition at every step of the Payroll process
- Disposition of *70* indicates successful Payroll cycle completion

Original Payroll dispositions

- 05 Awaiting Re-Extract
- 10 Awaiting Time Entry
- 15 Awaiting Correction
- 20 Awaiting Proof
- 22 Hours Correction
- 25 Awaiting Leave Process
- 30 Awaiting Calc
- 40 Awaiting Document
- 42 Awaiting Check/Direct Deposit Run
- 43 Awaiting Direct Deposit Run
- 44 Awaiting Check Run
- 50 Awaiting Update
- 60 Finance Extract
- 70 Complete

Payroll cycle errors

- Can be fixed as they appear, or
- Print the disposition report, continue with the other employees, then return to correct the error record
    - PHIDERR – Form to view payroll errors
    - PHRDERR – Payroll Errors report
    - PHRDCON – Disposition report

Chapter 5 of the Human Resources Using Banner Guide, pages 115-122 contains a flow diagram.

PDPLIFE

PDPLIFE – Life Insurance Calculations

- This step is optional and may not be necessary for every payroll

NOPEAMA – Electronic Approvals Mass Apply Process

- This step is also optional

PHPTIME

PHPTIME - Time Sheet Generation

- Sets up Payroll tables
- Initializes disposition at *05 (Awaiting Re-extract), 10 (Awaiting Time Entry)*, or *20 (Awaiting Proof)*
- Open the payroll process for time entry

Time entry options

PHAHOUR- Application form for exception hours entry

- Updates disposition from 10 to 20

PHAMTIM – Application form for mass time entry

PHPMTIM - If hours are entered on PHANTIM, then this process must be run to 'pull' the hours into the payroll process

- Updates disposition from 10 to 20

PHPPROF

PHPPROF - Pay Period Proof Batch Process

- Validates Payroll entries
- Updates disposition to *25* (*when correct*)

PHPLEAV

PHPLEAV - Leave Accruals and Taken Process

- Accrued and taken leaves
- Updates disposition to *30* (when correct)

PHPCALC

PHPCALC - Payroll Calculation Report

- Gross to net Payroll calculation process (COBOL)
- Updates disposition to *40*

PHPDOCM

PHPDOCM - Check/Direct Deposit Amount Calculation

- Creates document numbers for the Check process to follow
- Updates disposition to *42*

PHPCHKL

PHPCHKL - Check/Direct Deposit Notice Process

- Check Print process
- Produces 8 1/2” x 11” check and stub
- Updates disposition to *43/44*, and then *50* (after checks and direct deposit)

PHPDIRD

PHPDIRD – Create Direct Deposit File

- Does not update disposition

PHPUPDT

PHPUPDT - Pay Period Update Batch Process

- Updates Payroll history
- Updates disposition to *60* after PHPUPDT process

NBPBUDM

NBPBUDM - Budget Maintenance Process

- Computes encumbrances and budget amounts
- Assigns encumbrance numbers

NHPFIN1

NHPFIN1 - Finance Interface Extract

- Extracts new and changed encumbrances and/or budgets amounts

NHPFIN2

NHPFIN2 - Finance Interface Report

- Passes encumbrances and/or budgets amounts to Finance System
- FURFEED, FGRTRNI, FGRTRNR, FGRACTG

PHPFEXP

PHPFEXP - Expenditures Finance Extract

- Extracts Payroll Expense Finance Data
- Updates disposition to *62*

NHPFIN2

NHPFIN2 - Finance Interface Report

- Prints Finance report and interfaces with Finance
- Updates disposition to *70*

End of Payroll Process

- FURFEED, FGRTRNI, FGRTRNR, FGRACTG
- Payroll interface to Finance

PHPTIME - Time Processing Report

Purpose

Initializes Disposition to 05, 10, or 20

References data from the following tables to process the particular Year, Pay ID, and Pay Number combination:

- PTRCALN - Payroll Calendar Rule Table
- PTREARN - Earnings Code Rule Table
- PTRECLD - Earnings Code Labor Dist Rule Table
- PTRECLS - Employee Class Rule Table
- PTREERN - Employee Class Earn Code Rule Table
- PTREHOL - Employee Holidays Rule Table
- SPRIDEN - Identification/Name Repeating Table

Tables referenced

More tables referenced by PHPTIME:

- NBREARN - Employee Default Earnings Code Table
- NBRJLBD - Assignment Labor Dist Repeating Table
- NBRJOBS - Assignment Repeating Table
- NBRBJOB - Assignment Repeating Base Table
- PDRDEDN - Employee Deduction repeating Table
- PDRBDED - Employee Deduction Repeating Table
- NBBFISC - Fiscal Year Base Table
- PEBEMPL - Employee Base Table

Tables initialized

Tables 'initialized' by PHPTIME:

- Insert into PHRHIST: Pay History Repeating Table
- Insert into PHRJOBS: Pay History Jobs Repeating Table
- Insert into PHREARN: Pay History Earnings Repeating Table
- Insert into PHRHOUR: Hour Validation Table
- Insert into PHRELBD: Pay History L/D Override Repeating Table
- Insert into PHRDEDN: Pay History Deduction Repeating Table
- Insert into PHRERRL: Pay History Error Log Repeating Table

PHPPROF - Pay Period Proof Process

Preparing for PHPPROF

- Run PHRDERR - Payroll Errors Display Report
- Run PHRDCON - Disposition Control Report
- Go to PHAHOUR - Online Time Entry Form
  - Correct any errors
  - Check default hours
  - Add exception hours
  - Change Labor Distributions
  - Re-extract, if necessary
- Run PHPPROF - Pay Period Proof Process

Purpose

- Incoming Disposition is *20*
- Outgoing Disposition is *25* (success), *15* or *22* (failure)
- Tables Processed by PHPPROF
  - Update PHRHIST - Payroll History table
  - Update PHRJOBS - Payroll History Jobs table
  - Update PHRERRL with errors - Pay History Error Log table

PHPLEAV - Leave Accruals/Taken Process

Preparing for PHPLEAV

- Run PHRDERR - Payroll Errors Display Report
- Run PHRDCON - Disposition Control Report
- Time Entry
  - PHAHOUR
  - PHATIME
- Web Time Entry
  - Correct any errors
  - Re-extract if necessary
- Run PHPLEAV - Leave Accruals/Taken Process

Purpose

- Incoming Disposition 25
- Outgoing Disposition 30
- Tables referenced by PHPLEAV
  - PERLEAV - Leave Balances
  - PTRLEAV - Leave Code Rule Form
  - PTRLVAS - Leave Assignment Rule Form
  - PTRLVAC - Leave Accrual Rule Table
  - PTRLVPR - Leave Priority Code table

Tables processed

- Tables processed by PHPLEAV
  - PHRHIST – Update Disposition to 30
  - PHRJOBS – Update Disposition to 30
  - PHREARN – Insert any Dock Pay Records
  - PHRERRL – Insert any Errors/Warnings
  - PHRACCR – Insert leave Accrual Records

PHPCALC - Payroll Calculation Process

Preparing for PHPCALC

- Run PHRDERR - Payroll Errors Display Report
- Run PHRDCON - Disposition Control Report
- Check any Dock Pay entries
- Correct any errors
- Re-extract if necessary
- Run PHPCALC - Payroll Calculation Process

Purpose

- Calculates Gross to Net
- Calculated Benefit/Deduction Amounts
- Incoming Disposition 30
- Outgoing Disposition 40

Updated reports

- Updates PHRDEDN - Deduction Calculation Report
  - Calculated Deduction amounts
- Updates PHREARN - Payroll Earnings Report
  - Calculated Earnings amounts
- Updates PHRJOBS - Payroll Jobs Report
  - Updates Disposition to *40*
- Updates PHRHIST - Payroll History Report
  - Updates Disposition to *40* and records Gross and Net amounts
- Updates PHRHOUR - Payroll Time Entry Report
  - With calculated earnings amounts by Data Entry period
- Updates PHRACCR - Payroll History Accruals Report
  - Accrual amounts (if applicable)

PHPDOCM - Check/Direct Deposit Amounts Process

Purpose

- Incoming Disposition 40
- Outgoing Disposition 42
- Creates Check and Direct Deposit Document Records
- Checks GXRDIRD - Employee Payroll Direct Deposit Record for direct deposit information

Functions

- Updates PHRHIST- Payroll History table
- Updates PHRJOBS - Jobs History table
- Inserts Document numbers into PHRDOCM - Disposition Control Report (starting with *1*)
- Inserts records into temporary table PHRTDED - Temporary Payroll Deduction Record for better performance

PHPCHKL/PHPCHEK Printing Process

PHPCHKL/ PHPCHEK

The PHPCHKL/PHPCHEK printing process is run twice, once for each document type.

- Once for Checks
- Once for Direct Deposits
- Updates disposition to 43 or 44 the first run, depending on which document type is run first
- Updates PHRHIST – Payroll History table
- Updates PHRJOBS – Jobs History table
- Updates PHRDOCM – Updates document numbers
- Updates disposition to 50 after both document types have been run

Status

Now we are ready to update year to date totals and perform cleanup.

PHPUPDT – Pay Period Update Process

Tables updated by PHPUPDT

- PERETOT - Insert/update monthly earnings information
- PERJTOT - Insert/update monthly earnings by position
- PERDTOT - Insert/update monthly deduction information
- PEREHIS - Insert when PEBEMPL changes
- PERLHIS - Insert when Leave balances are updated
- PERPADV - Insert/update with Pay Advance amounts
- PHRHIST - Updates Disposition to *60*
- PHRJOBS - Updates Disposition to *60*
- PDRBDED - Updates to delete add/replace information
- PDRDEDN - Updates for bonds purchased
- GXRDIRD - Updates for pre-note employees
- NBRBJOB - Updates with deferred pay balances information
- PERLEAV/PERJLEV- Updates Leave balance amounts

Feed to Finance – Budget Maintenance

NBPBUDM

NBPBUDM - Budget Maintenance Process

- Computes encumbrances
- Computes budget amounts
- Assigns encumbrance numbers

Feed to Finance - Budgets and Encumbrances

NHPFIN1

- Extracts data from Payroll and Position Control tables
- Inserts into NHRFINC (temporary)
- Inserts into NHRDIST if option checked on NTRINST

NHRFIN2

- Loads Budget and Encumbrance interface data into GURFEED Finance Interface table
- Looks to FOBSEQN table to get the next Document Number, and it will start with *F*

Feed to Finance - Finance Processes Take Over

FURFEED

- Loads Budget and Encumbrance data into the FGBTRNI table from GURFEED

FGRTRNI

- Loads Budget and Encumbrance data into FGBJVCD and FGBJVCH tables
- Loads any error records into the FGRTRNR error table
- Run the FGRTRNR report for errors

FGRACTG

- Usually set up to run on Sleep/Wake interval
- All Budget and Encumbrance data from the current Payroll will now be posted to FGBOPAL - the Operating Ledger

Feed to Finance – Payroll Expenses

PHPFEXP

- Extracts Actual data from Payroll and Position Control tables
- Inserts into NHRFINC (temporary)
- Inserts into NHRDIST if option checked on NTRINST
- Updates Disposition to *62*

NHRFIN2

- Loads Actual interface data into GURFEED - the Finance Interface table
- Looks to FOBSEQN table to get the next Document number, and it will start with *F*

Feed to Finance – Finance Processes Take Over

FURFEED

- Loads Budget and Encumbrance data into the FGBTRNI table from GURFEED

FGRTRNI

- Loads Budget and Encumbrance data into the FGBJVCD and FGBJVCH tables
- Loads any error records into the FGRTRNR error table
- Run the FGRTRNR report for errors

FGRACTG

- Usually set up to run on Sleep/Wake interval
- All Actual Payroll data from the current Payroll will now be posted to FGBOPAL - the Operating Ledger

Self Check

Directions

Use the information you have learned in this workbook to complete this self check activity.

Exercise 1

Retrieve the IDs for all employees whose records are confidential.

(SPRIDEN, PEBEMPL, SPBPERS)

Exercise 2

Retrieve the first name, last name, and ID for all active employees. Order the rows by last name, then first name.

(SPRIDEN, PEBEMPL)

Exercise 3

Retrieve the first name, last name, and position for all active employees.

(SPRIDEN, NBRBJOB, PEBEMPL)

Exercise 4

Allowing the parameters of pay year, pay id, and pay number, retrieve the begin and end dates for a payroll.

(PTRCALN)

Exercise 5

Combining exercises 3 and 4, retrieve the first name, last name, and position for all active employees whose job was active during the specified payroll. Allow the parameters of pay year, pay id, and pay number.

(SPRIDEN, PEBEMPL, NBRBJOB, PTRCALN)

Exercise 6

Retrieve the first name, last name, home department and current annual salary for all active employees.

(SPRIDEN, PEBEMPL, NBRJOBS)

Exercise 7

Retrieve the payroll records which did not successfully complete the payroll process.

(PHRRHIST)

Exercise 8

Retrieve all active positions and the fiscal year 1995 labor distribution accounts associated with them. Order by position.

(NBBPOSN, NBRPLBD)

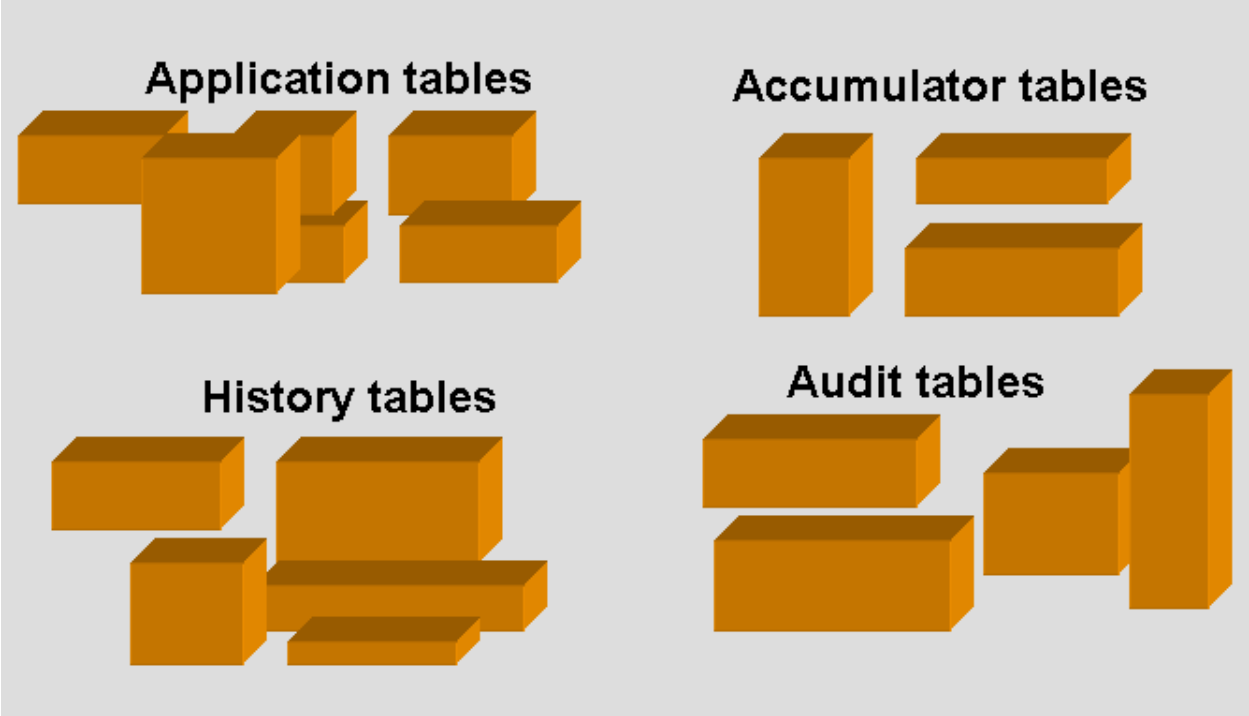
Exercise 9

Accepting the parameter of fiscal year (fisc_code), retrieve each position and the total budget allocated for it.

(NBRPTOT)

What is History and What is Not?

Diagram



Application tables vs. History tables

Application tables:

- PEBEMPL
- PERLEAV
- NBRBJOB/NBRJOBS
- PDRBDED/PDRDEDN

History tables:

- PEREHIS
- PERLHIS
- PERJHIS
- PERDHIS

Note that the above history tables are P% tables.

Accumulator tables vs. History tables

Accumulator tables:

- PERETOT
- PERJTOT
- PERDTOT

History tables:

- PHRHIST
- PHRJOBS
- PHRDEDN
- Etc.

Again, note that the history tables are P% tables, even those related to jobs.

Human Resources Security

Introduction

This section discusses security in Banner Human Resources.

Human Resources Security

Overview

- In addition to Banner security
- Four types of security:
  - Employer
  - Organization
  - Employee Class
  - Salary Level
- The types can be used in any combination

Application Forms

Forms

PTRINST - Installation Rule Form

- Turn on Security

PTRUSER - User Codes Rule Form

- Set up HR Users

PSAEMPR - Banner EMPR Security Form

- Set up Employer Security

PSAORGN - Banner ORGN Security Form

- Set up Organization Security

PSAECLS - Banner ELCS Security Form

- Set up Employee Class Security

PTRUSER

Purpose

- Set up HR/Payroll User ID and Name
- **If ANY security types are turned on, ALL HR users must be defined here**
- Grant Master Employer security or not
- Grant Master Organization security or not
- Grant Master Employee Class security or not
- Grant limit on Salary Level access

Step one

Get the maximum salary for this employee:

```
SELECT MAX(NBRJOBS_ANN_SALARY)
  INTO  :SECURITY_SALARY
  FROM NBRJOBS
 WHERE NBRJOBS_PIDM = :PIDM
   AND NBRJOBS_POSN = :POSN
   AND NBRJOBS_SUFF = :SUFF
```

Step two

Check to see if user has Specific Organization:

```
SELECT 'X'
  FROM PTRUSER
 WHERE PTRUSER_CODE = USER
   AND NVL(:SECURITY_SALARY,0) <= PTRUSER_SALA_LEVEL
```

PSAEMPR

Purpose

- Form inserts/updates/deletes rows in PSREMPR table
- For Each HR/Payroll user, enter the employer(s) that the user can view
- Used less than Organization and Employee Class security
- Works with same logic as ORGN and ECLS security

PSAORGN

Purpose

- Form inserts/updates/deletes rows in PSRORGN table
- Form allows copying of information from one user to another
- Form allows for granting access to a range of organizations - low to high organizations

Step one

Check to see if user has Master Authority:

```
SELECT 'X'
  FROM PTRUSER
 WHERE PTRUSER_MASTER_ORGN_IND = 'Y'
   AND PTRUSER_CODE            = USER
```

Step two

If Step One fails, check to see if user has Specific Organization:

```
SELECT 'X'
  FROM PSRORGN, PTRINST
 WHERE PSRORGN_USER_CODE           = USER
   AND PTRINST_CODE                = 'PAYROLL'
   AND NVL(PTRINST_COAS_CODE,*') =
    NVL(PSRORGN_COAS_CODE,'*')
   AND :SECURITY_ORGN_CODE        >= PSRORGN_ORGN_LOW
   AND :SECURITY_ORGN_CODE        <= PSRORGN_ORGN_HIGH
```

PSAECLS

Purpose

- Inserts/updates/deletes rows in PSRECLS table
- Allows copying of information from one user to another
- Allows for granting access to individual Employee classes

Step one

Check to see if user has Master Authority:

```
SELECT 'X'
  FROM PTRUSER
 WHERE PTRUSER_MASTER_ECLS_IND = 'Y'
   AND PTRUSER_CODE = USER
```

Step two

If Step One fails, check to see if user has Specific ECLS:

```
SELECT 'X'
  FROM PSRECLS
 WHERE PSRECLS_USER_CODE = USER
   AND PSRECLS_ECLS_CODE = :SECURITY_ECLS_CODE
```

Maintenance Tools and Tips

Introduction

This section discusses tips and tricks for maintaining Banner Human Resources.

Maintenance

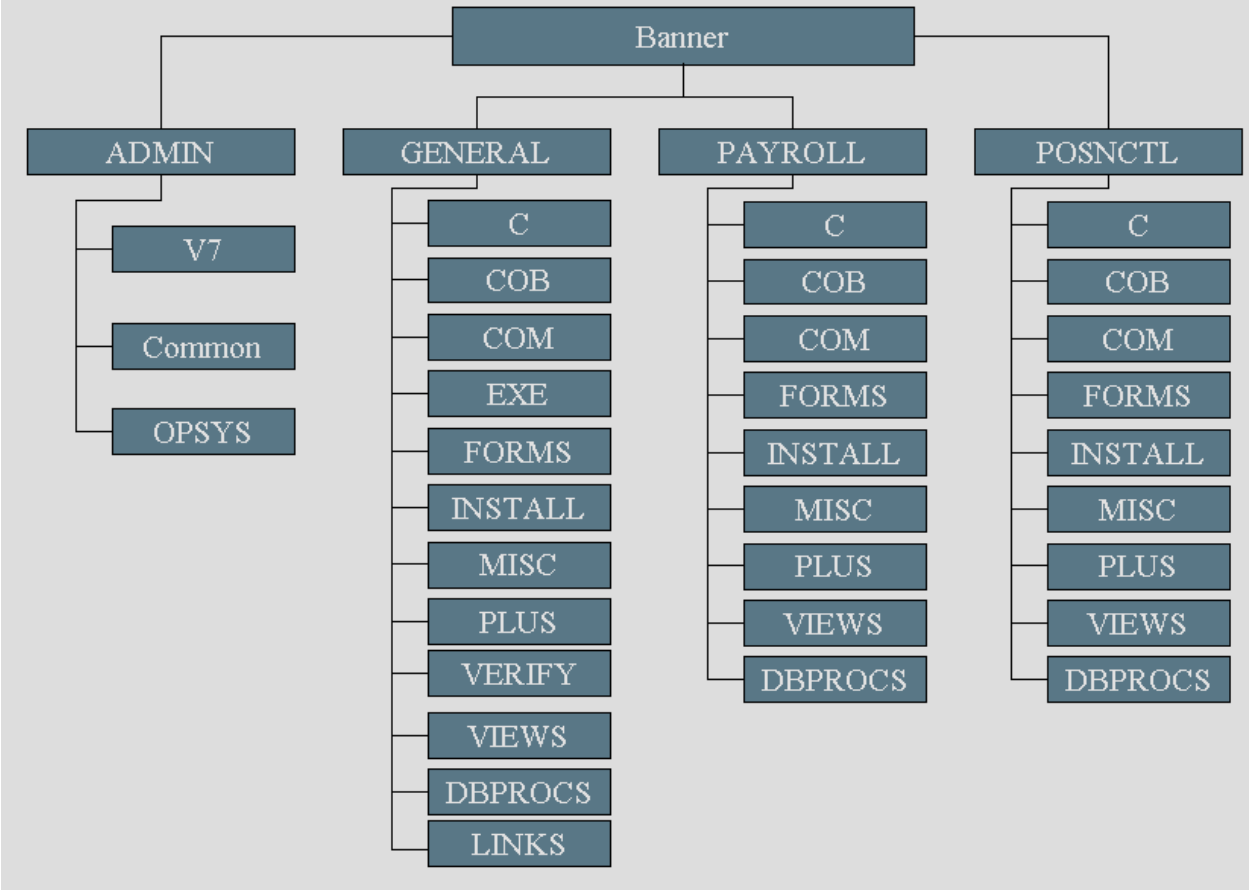
Components

- Directory Structures
- Customizing Banner
- Reporting
- Migration
- Troubleshooting

Directory structures

Diagram

Where are all these forms and processes stored?



ADMIN directories

V7

- Scripts to create an Oracle Banner database

COMMON

- Common objects shared by all products

OPSYS

- Contains COBOL make files for platform (UNIX only)

GENERAL directories

C

- Pro*C and C source files, C compile procedures, EXEC INCLUDE files

COB/COBPCO

- Pro*COBOL files (VAX/VMS only)

COB/LIB

- Links to copybooks with .cob extension and lowercase names (UNIX only)

COM

- DCL command files (VAX/VMS only)

EXE

- Compiled PRO*COBOL executables for all products

FORMS

- Oracle*Forms .fmb, .fmx, .mmb (menu), .mmx, .pll (libraries)

INSTALL

- .SCTDMP file used during initial install (renamed to .DMP during install)

MISC

- Shell scripts (UNIX only)

PLUS

- SQL*Plus scripts

VERIFY

- Files used by the verification step of upgrades

VIEWS

- SQL*Plus scripts to recreate views

DBPROCS

- SQL*Plus scripts to recreate database procedures, packages, functions, and triggers

INSTALL

- All Banner installation scripts

LINKS

- Composite directory for local access of Banner products

PAYROLL and/or POSNCTL directories

C

- Pro*C and C source files, C compile procedures, EXEC INCLUDE files

COB/COBPCO

- COBOL copybooks for all products
- Pro*COBOL files (VAX/VMS only)

COM

- DCL command files (VAX/VMS only)

FORMS

- Oracle*Forms - .fmb, .fmx, .mmb (menu), .mmx, .pll (libraries)

INSTALL

- .SCTDMP file used during initial install (renamed to .DMP during install)

MISC

- Shell scripts (UNIX only)

PLUS

- SQL*Plus scripts

VIEWS

- SQL*Plus scripts to recreate views

DBPROCS

- SQL*Plus scripts to recreate database procedures, packages, functions, and triggers

Standards

Creating standards

- Create data standards, especially for names and addresses
- All offices need to agree
- Document data standards and distribute to all offices
- Offices need to agree on common validation table codes (e.g. STVATYP). Subsequent additions and changes to these should be agreed upon

Date format

A setting on the Installation Control Form (GUAINST) determines the format for dates and a pivot year:

- MDY Month, Day, Year
- DMY Day, Month, Year
- YMD Year, Month, Day

Create descriptive and meaningful codes

Establish a common method of abbreviation before values are assigned.

Example:

- Posn.
- Posn (no period)

Avoid descriptions that have abbreviated, non-English language values.

Avoid special characters:

- hyphen (-)
- slash (/)
- asterisk (*)
- plus (+)
- pound (#)
- ampersand (&)
- at (@)
- dollar sign ($)

Additional notes

- Use alphabetic codes (rather than numeric) when possible
- Avoid embedded spaces within a Rule or Validation code
- Avoid using words that have specific meaning to the product

Customizing Banner

Initial suggestion

Avoid customizations for a designated amount of time. When users adjust to the new system, they will see if the change is absolutely necessary or not.

Customization necessity

- Customize delivered Banner objects only when absolutely necessary
- Upgrades that include the modified object need to be checked against the delivered one
- The changes will need to be repeated
- Adding in-house objects (forms and processes) will be easier to maintain than customizing delivered objects

Create site forms

To create site forms:

- Clone one of the Banner forms
- Start with GUASKEL.fmb to gain access to the global variables and common triggers
- Keep your Source code separate from SunGard Higher Education's form directories
- Recommended: Oracle Forms Training

Reporting Tool Options

Delivered tools

- C
- COBOL
- Developer 2000

Other options

- MS Access
- Crystal Reports
- Brio Query
- SAS
- Business Objects

Methodologies

- Web
- Object:Access

Purpose

- Comply with the Coding Standards in the General TRM
- Includes Banner standards for Forms, C, and COBOL programming
- Reports and Processes Grid
- Database Schematics

Utilities

Utility	Purpose
GURRDDL	Script to PL/SQL script which generates DDL syntax for a specified table(s).
GURPDED	The Data Element Dictionary (DED) can be generated by running the GURPDED Pro*C program that is delivered in Banner General.
GURLSID	Lists of all tables and columns in which a person exists.
GURDLID	Script used to delete all information about a PIDM in the database.
indexes.sql	
GJRRPTS	The List of Reports and Parameters can be generated by running this Pro*C program that is delivered in Banner General.

Seed Data

Usage

Use of all seed data is not mandatory; however, some is required.

Most of this information is needed for external reports to third parties:

NTRAUBK	NTRAUFD	NTRAUFM
PTREEOS	PTV1099	PTVEEOC
PTVEFUN	PTVESKL	PTVHSMT
PXRADDS	PXRCALC	PXREXEM
PXRFSTA	PXRGRAD	PXRTVFY
PXRTXCD	PTRGTAX	PTVBDTY
PTVEEOG	PTVRANK	PTVREPT
PXRRBOX		

PIDM- or POSN-related seed data should be deleted.

All other data should be examined to determine if it should be deleted.

Troubleshooting

Introduction

This section discusses troubleshooting tips for Banner Human Resources.

Troubleshooting Tips

Helping "stuck" users

If a user gets 'stuck' within Banner, have him/her go into the pull-down menu and perform a **Remove Record** or **Clear Record** function.

This occurs frequently when a user gets to the last record, and a record is automatically inserted. The user needs to enter all required fields, or remove the record.

Other sources

- Encourage users to read the known issues report - this will relieve you from much of the burden
- Encourage users to subscribe to listservs related to them

Tell your users what you need

- Have your users capture a screenshot of the error, when possible.
- With the Banner window active, hold down the ALT key and hit the PRINT SCREEN key. This captures the screen on the clipboard. This can be pasted into a document or email
- Ask for specifics:
  - What form or process?
  - What does the hint line say?

Display Error

- This will return actual ORACLE errors
- Common error is a constraint violation, this will display the constraint name

Dynamic Help Query

- To get table/field names involved.
- This will only work when the block is named the same as the table it is linked to.
- What id?
- Compare the 'bad' record with ones that do not produce the errors and examine differences between them

Referenced libraries and forms

- GOQOLIB form and GOQRPLS library
- Contains procedures and objects used in multiple forms across Banner products
- Is referenced by at least every form which uses Banner Security, because it contains the security trigger G$_VERIFY_ACCESS
- Errors can occur at runtime if not all the references are in the Forms path

Conversion

Introduction

This section discusses conversion strategies, steps and examples.

Conversion

Overview

- Conversion strategies
- Conversion steps
- Conversion example

Conversion – Strategies

Considerations

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

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

Preparation

- Create a Conversion Plan document
- Review the steps that are involved to get to your “go live” dates
- Create a timeline
- Determine the processes that need to be written
- Will data need to be translated?
- Will data need to be cleaned up on legacy side?

Legacy system considerations

- Keep track of PIDM on legacy system
- Generate ID or SSN?
- Name/Address format
    - Avoid using “#” with Letter Generation
- Address types
    - Multiple ID's on legacy system?

Inserting non-Oracle data

To insert non-Oracle data into Banner tables:

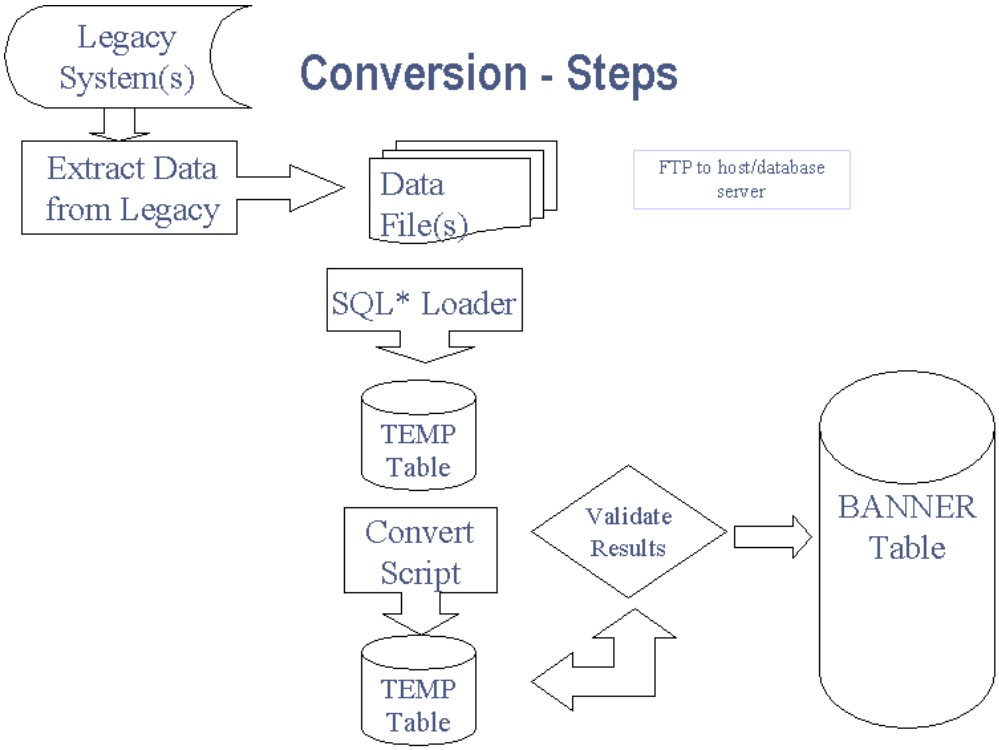
- Create flat files which contain the relevant data
- Read the files by SQL*Loader into intermediate Oracle tables (don't load them into Banner directly)
- Validate the data
- Load data into the Banner tables

Conversion – Steps

Steps

1. Document steps as you proceed
2. Review current data
3. Determine scope:
    - What will you convert?
    - Which tables will be populated?
4. Map legacy data to Banner tables
    - Create a mapping document working with users and consultants
5. Write a detail plan of:
    - Data to be converted
    - Banner tables to be populated
    - Deadlines/timelines
6. Review plan and get approval from users
7. Develop procedures and programs
8. Test conversion in TEST or PPRD database
9. Users verify data
10. Test again and make corrections to procedures and programs
11. Do conversion in production
12. Users verify data

Diagram


Legacy System(s)
Conversion - Steps
Extract Data from Legacy
Data File(s)
FTP to host/database server
SQL* Loader
TEMP Table
Convert Script
TEMP Table
Validate Results
BANNER Table

Conversion – Example

Example purpose

The following example shows how to:

- 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 example uses a flat file containing:

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

Flat file layout

Example of a Flat File 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
```

Create temporary tables

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
```

Alter temporary tables

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
```

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')
```

Decode SPBPERS_SEX

Decode SPBPERS_SEX (decode_sex.sql):

```
SPOOL decode
UPDATE sytpers
SET spbpers_sex = decode (spbpers_sex,'1','F','2','M','N');
SPOOL OFF
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;
```

Insert into SATURN tables

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

  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;

UPDATE  sobseqn
   SET  sobseqn_maxseqno = 77777783,
             sobseqn_activity_date = sysdate
 WHERE  sobseqn_function = 'PIDM';
```

Clean SATURN tables

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
```

Shell script

```
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
```

SQL*LOADER and Import Tip

Tip

Constraint checking uses resources.

Constraints are checked as each row is inserted into the database.

To speed up large data loads or imports:

- Consider disabling constraints first
- Consider creating the indexes after process has completed

Conversion – Summary of Steps

Summary

- Determine flat file layout
- Create temporary tables
- Alter temporary tables, if necessary
- Create loader control file
- Load flat file data into temporary tables
- Translate codes using DECODE, if necessary
- Check temporary table data
- Insert temporary table data into Banner tables
- Check data in Banner (in forms and tables)
- Update sobseqn
- Use Shell Script

APIs

Introduction

This section lists the APIs used in Banner Human Resources.

Human Resources/Position Control APIs

Introduction

A *Banner API* is essentially a database package that encapsulates the business logic surrounding a Banner business entity.

A *Banner business entity* is the fundamental unit of information that a Banner application can modify (for example, a course, address, e-mail, and telephone number).

APIs leverage Banner messaging support so that Banner becomes "messaging enabled", i.e., be able to stimulate the production of messages based on changes to data they own. Programs within Banner as well as external systems can then manipulate a business entity or table by calling APIs.

The foundation of all Banner APIs is the ability to perform CRUD (create, retrieve, update, and delete) actions on the database. APIs contain the same validation logic and edits that are executed when performing these actions (CRUD) from a Banner Internet-Native form, self-service page, or batch process.

For example, calling the API for the "employee" business entity allows you to:

- Create employee information (insert rows into the Employee table).
- Retrieve data from the Employee table.
- Update employee information (modify rows in the Employee table).
- Delete employee information (physically remove data from the Employee table).

The advantages of using APIs include:

- APIs expose business objects that are useful to users of other applications comprising an institution's digital campus.
- APIs are bi-directional. That is, they support read and write functions.
- APIs ensure consistent edit and validation business logic (create, retrieve, update, and delete).
- APIs eliminate duplicate code that would need to be maintained in each application.

Employee API

The Employee API is a database package that contains the business logic (validation rules and other related processing) surrounding the table associated with Employee data. The Employee API can be called by programs within Banner (like Forms, C processes, Web packages, etc.) as well as by external systems when there is a need to insert/update/delete information into the Employee table (PEBEMPL).

The API provides for consistent processing and utilizes the same business logic whether a program in Banner calls it or an external system calls it. The table that contains Employee information is PEBEMPL.

The primary form associated with the Employee table is the Employee Form (PEAEMPL). This form retains a number of immediate edits over the data, such as the validations of Employee Class, against the Benefit Category and Leave Category codes. When inconsistent data is detected, the error messages will continue to be displayed at the bottom of the window on the Auto-help line. However, when you save the data and one or more error conditions continue to exist, a new window will open, detailing all error messages. Otherwise, you will not notice any other functional differences from prior versions of Banner Human Resources.

Other Human Resources forms and processes affect data associated with the employee table, and these have been modified to process through the Employee API. These include:

- New Hire Form (PEAHIRE) - This form creates Employee table data and validates using the Employee API;
- One-time Payment Form (PEA1PAY) - This form processes quick payments for employees and also impacts the Employee table (PEBEMPL);
- Employee Status Change Form (PEAESCH) - This form updates the Employee table with Leave or Termination data;
- Termination Verification Form (PEATMVF) - This form is designed to review employee records where there has been a Termination Date and Termination Reason established on PEAEMPL, but the Employee Status has not been set to 'Terminated'.
- Pay Period Update Process (PHPUPDT) - This process is the final payroll program that may affect employee records where the Employee Status is changed because Leave or Termination Dates have been future effective-dated consistent with the payroll.

Electronic Approvals, EPAF package (NOKPLIB) - Since employee records can be created or updated through the use of EPAF, this package was modified to call the Employee API.

Please refer to the API Reference Guide for documentation related to all edits associated with the Employee API.

The PB_EMPLOYEE package provides the common business interface for the Employee API. The Employee API establishes information about an employee's terms of employment.

The following table provides a brief overview of the Employee API package. For more details, refer to the HR *API Reference Guide*.

API Object Name	**API Entity Name**	**Associated API Packages**	**Associated scripts**	**Associated Table**	**Affected Objects (Forms, Processes, etc.)**
PB_EMPLOYEE	EMPLOYEE	PB_EMPLOYEE_RULES PB_EMPLOYEE_STRINGS	pokb_employee0.sql pokb_employee1.sql pokb_employee_r0.sql pokb_employee_r1.sq pokb_employee_s0.sql pokb_employee_s1.sql	PEBEMPL	PEAEMPL PEAESCH PEAHIRE PEA1PAY PHPUPDT NOKPLIB

Important API Notes

Employee History Table Enhancement

The Employee History table (PEREHIS) has been expanded to include all data items associated with the Employee table (PEBEMPL). Therefore, any insert or change to any existing field on the Employee Form (PEAEMPL) will result in an audit record being posted to the Employee History table, on a going-forward basis. No changes were made to the Employee History Form (PEIEHIS) to display these new items. However, the audited elements are present on the table.

Payroll Processing Notes

An Employee table update function in the Pay Period Update Process (PHPUPDT) was modified to also utilize the Employee API. This API is called when the process determines that the Employee Status should be updated to reflect that the employee on Leave or Terminated. Through this evaluation, the Start and End Dates are evaluated against the job record in order to determine whether this update should occur.

For example, on December 1st, you indicate that the job is to go on Leave on December 31st and re-activates the job on March 1st. Next, in order to indicate the Leave information on the Employee record you enter that the Leave Start Date is January 1st and the Expected Return Date is February 28th. During the month-end payroll, since all records are in synch, the Pay Period Update Process (PHPUPDT) changes the Employee Status to *Leave*.

There is a situation within Leave and Termination processing where the Pay Period Update Process (PHPUPDT) may error, the employee will not be updated and the process will stop. This will affect the finalization of the payroll. This situation involves records where you have further modified the job status and dates without reviewing the Employee record controls. Following the above example, if the job record has been reactivated on January 15th and is extracted and attempted to be paid, the Pay Period Update Process (PHPUPDT) will error out because the employee Leave dates are inconsistent with the job dates.

In order to ensure that the Pay Period Update Process (PHPUPDT) does not error and terminate, an edit has been placed in the Pay Period Proof Process (PHPPROF). If the job dates and status are not consistent with the employee record criteria, either on the Leave or Termination windows, the employee will receive an error in the payroll and the disposition will be set to '05'. The employee must be deleted in the payroll, their records must be corrected on the Employee Form (PEAEMPL), and they must be re-extracted through the Time Processing Report (PHPTIME) to be paid.

Security

It is important to note that Oracle and Banner HR security will continue to execute when data is processed through the API against the employee table, whether the processing source is within Banner or external to it.

Biographic/Demographic Information APIs

The following table provides a brief overview of the Biographic/Demographic APIs. For more details, refer to the Banner Human Resources API Reference Guide, Release 7.2. These APIs replace the code in the corresponding Banner form.

All APIs packages used in Banner Human Resources will begin with a 'p,' followed by an 'b,' and an underscore '_,' followed by a verb/noun describing what the API does. If the 2nd letter is a 'p' instead of a 'b,' then that database package is a Business Process API defining a 'unit of work'.

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Forms and Processes
PB_CERTIFICATION	CERTIFICATION	pb_certification pb_certification_rules pb_certification_strings	ppkb_certification0.sql ppkb_certification1.sql ppkb_certification_r0.sql ppkb_certification_r1.sql ppkb_certification_s0.sql ppkb_certification_s1.sql ppkd_pprcert0.sql ppkd_pprcert1.sql	PPRCERT	PPACERT
PB_ENDORSEMENT	ENDORSEMENT	pb_endorsement pb_endorsement_rules pb_endorsement_strings	ppkb_endorsement0.sql ppkb_endorsement1.sql ppkb_endorsement_r0.sql ppkb_endorsement_r1.sql ppkb_endorsement_s0.sql ppkb_endorsement_s1.sql ppkd_pprends0.sql ppkd_pprends1.sql	PPRENDS	PPACERT
PB_EXPERIENCE	EXPERIENCE	pb_experience pb_experience_rules pb_experience_strings	ppkb_experience0.sql ppkb_experience1.sql ppkb_experience_r0.sql ppkb_experience_r1.sql ppkb_experience_s0.sql ppkb_experience_s1.sql ppkd_pprexpe0.sql ppkd_pprexpe1.sql	PPREXPE	PPAEXPR

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Forms and Processes
PB_DRIVER_ LICENSE	DRIVER_ LICENSE	pb_driver_license pb_driver_license_rules pb_driver_license_strings	ppkb_driver_lic0.sql ppkb_driver_lic1.sql ppkb_driver_lic_r0.sql ppkb_driver_lic_r1.sql ppkb_driver_lic_s0.sql ppkb_driver_lic_s1.sql ppkd_pprdlic0.sql ppkd_pprdlic1.sql	PPRDLIC	PPAGENL
PB_HONOR_ AWARD	HONOR_ AWARD	pb_honor_award pb_honor_award_rules pb_honor_award_strings	ppkb_honor_award0.sql ppkb_honor_award1.sql ppkb_honor_award_r0.sql ppkb_honor_award_r1.sql ppkb_honor_award_s0.sql ppkb_honor_award_s1.sql ppkd_pprhnaw0.sql ppkd_pprhnaw1.sql	PPRHNAW	PPAGENL
PB_ PUBLICATION	PUBLICATION	pb_publication pb_publication_rules pb_publication_strings	ppkb_publication0.sql ppkb_publication1.sql ppkb_publication_r0.sql ppkb_publication_r1.sql ppkb_publication_s0.sql ppkb_publication_s1.sql ppkd_pprpubl0.sql ppkd_pprpubl1.sql	PPRPUBL	PPAGENL
PB_SKILL	SKILL	pb_skill pb_skill_rules pb_skill_strings	ppkb_skill0.sql ppkb_skill1.sql ppkb_skill_r0.sql ppkb_skill_r1.sql ppkb_skill_s0.sql ppkb_skill_s1.sql ppkd_pprskil0.sql ppkd_pprskil1.sql	PPRSKIL	PPASKIL

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Forms and Processes
PB_PTVENDS	n/a	pb_ptvends	pvkb_ptvends0.sql pvkb_ptvends1.sql	PTVENDS	PB_ ENDORSEMENT
PB_PTVLCSV	n/a	pb_ptvlcsv	pvkb_ptvlcsv0.sql pvkb_ptvlcsv1.sql	PTVLCSV	PB_CERTIFICATION
PB_PTVPUBT	n/a	pb_ptvpubt	pvkb_ptvpubt0.sql pvkb_ptvpubt1.sql	PTVPUBT	PB_PUBLICATION

Student-Employee FICA APIs

The following table provides a brief overview of the student FICA APIs. For more details, refer to the Banner Human Resources API Reference Guide, Release 7.2. These APIs replace the code in the corresponding Banner Human Resources forms.

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Objects (Forms, Processes, etc.)
PB_ASSOC_ TERM	ASSOC_ TERM	pb_assoc_term pb_assoc_term_rules pb_assoc_term_ strings	phkb_assoc_term0.sql phkb_assoc_term1.sql phkb_assoc_term_r0.sql phkb_assoc_term_r1.sql phkb_assoc_term_s0.sql phkb_assoc_term_s1.sql phkd_ptratrm0.sql phkd_ptratrm1.sql	PTRATRM	PTRATRM
PB_STU_ DEDUCTION	STU_ DEDUCTION	pb_stu_deduction pb_stu_deduction_ rules pb_stu_deduction_ strings	phkb_stu_deduction0.sql phkb_stu_deduction1.sql phkb_stu_deduction_r0.sql phkb_stu_deduction_r1.sql phkb_stu_deduction_s0.sql phkb_stu_deduction_s1.sql phkd_ptrstde0.sql phkd_ptrstde1.sql	PTRSTDE	PTRSTDE
PB_STU_ EMP_CRED	STU_EMP_ CRED	pb_stu_emp_cred pb_stu_emp_cred_ rules pb_stu_emp_cred_ strings	phkb_stu_emp_cred0.sql phkb_stu_emp_cred1.sql phkb_stu_emp_cred_r0.sql phkb_stu_emp_cred_r1.sql phkb_stu_emp_cred_s0.sql phkb_stu_emp_cred_s1.sql phkd_ptrscrd0.sql phkd_ptrscrd1.sql	PTRSCRD	PTRSCRD
PB_ PTVATRM	n/a	pb_ptvatrm	pvkb_ptvatrm0.sql pvkb_ptvatrm1.sql	PTVATRM	PB_ASSOC_ TERM

Electronic Approvals APIs

The following table provides a brief overview of the APIs for Electronic Approvals. For more details, refer to the Position Control *API Reference Guide*.

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Objects (Forms, Processes, etc.)
NB_EA_ COMMENTS	EA_ COMMENTS	NB_EA_ COMMENTS NB_EA_ COMMENTS_ RULES NB_EA_ COMMENTS_ STRINGS	nokb_ea_comments0.sql nokb_ea_comments1.sql nokb_ea_comments_r0.sql nokb_ea_comments_r1.sql nokb_ea_comments_s0.sql nokb_ea_comments_s1.sql nokd_norcmnt0.sql nokd_norcmnt1.sql	NORCMNT	NOAAPSM NOAEPAF
NB_EA_ GROUP_ CATEGORY	EA_GROUP_ CATEGORY	NB_EA_GROUP_ CATEGORY NB_EA_GROUP_ CATEGORY_ RULES NB_EA_GROUP_ CATEGORY_ STRINGS	ntkb_ea_group_cat0.sql ntkb_ea_group_cat1.sql ntkb_ea_group_cat_r0.sql ntkb_ea_group_cat_r1.sql ntkb_ea_group_cat_s0.sql ntkb_ea_group_cat_s1.sql ntkd_ntragrp0.sql ntkd_ntragrp1.sql	NTRAGRP	NTRAGRP
NB_EA_ GROUP_ USER	EA_GROUP_ USER	NB_EA_GROUP_ USER NB_EA_GROUP_ USER_RULES NB_EA_GROUP_ USER_STRINGS	ntkb_ea_group_user0.sql ntkb_ea_group_user1.sql ntkb_ea_group_user_r0.sql ntkb_ea_group_user_r1.sql ntkb_ea_group_user_s0.sql ntkb_ea_group_user_s1.sql ntkd_ntragid0.sql ntkd_ntragid1.sql	NTRAGID	NTRAGRP NOAOGRP

Jobs Information APIs

Four APIs have been created for the following tables associated with the Employee Jobs Form (NBAJOBS), namely:

- The Base Job API contains the business logic for processing base job data (NBRBJOB) on the Employee Jobs Form (NBAJOBS).
- The Job Detail API contains the business logic for processing job detail data (NBRJOBS) on the Employee Jobs Form (NBAJOBS).
- The Job Earnings API contains the business logic for processing job default earnings data (NBREARN) on the Employee Jobs Form (NBAJOBS).
- The Job Labor Distribution API contains the business logic for processing job labor distribution data (NBRJLBD) on the Employee Jobs Form (NBAJOBS).

Three BPIs have been created to maintain the employee jobs information. Each BPI, in turn, invokes one or more of the new APIs.

Note**:** It is important to note that the APIs alone cannot be used to maintain the jobs data. Applications must invoke the BPIs to ensure data integrity.

The new BPIs are:

- Job Labor BPI
- Job Assignment BPI
- Job Earnings BPI

The following table provides a brief overview of the Jobs APIs associated with the Employee Jobs Form (NBAJOBS). For more details, refer to the Position Control API Reference Guide, Release 7.3.

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Objects (Forms, Processes, etc.)
NB_JOB_ BASE	JOB_BASE	nb_job_base nb_job_base_rules nb_job_base_strings	nbkb_job_base0.sql nbkb_job_base1.sql nbkb_job_base_r0.sql nbkb_job_base_r1.sql nbkb_job_base_s0.sql nbkb_job_base_s1.sql nbkd_nbrbjob0.sql nbkd_nbrbjob1.sql	NBRBJOB	NBAJOBS PEA1PAY PEAHIRE PEAESCH
NB_JOB_ DETAIL	JOB_DETAIL	nb_job_detail nb_job_detail_rules nb_job_detail_strings	nbkb_job_detail0.sql nbkb_job_detail1.sql nbkb_job_detail_r0.sql nbkb_job_detail_r1.sql nbkb_job_detail_s0.sql nbkb_job_detail_s1.sql nbkd_nbrjobs0.sql nbkd_nbrjobs1.sql	NBRJOBS	NBAJOBS PEA1PAY PEAHIRE PEAESCH
NB_JOB_ EARNINGS	JOB_ EARNINGS	nb_job_earnings nb_job_earnings_ rules nb_job_earnings_ strings	nbkb_job_earnings0.sql nbkb_job_earnings1.sql nbkb_job_earnings_r0.sql nbkb_job_earnings_r1.sql nbkb_job_earnings_s0.sql nbkb_job_earnings_s1.sql nbkd_nbrearn0.sql nbkd_nbrearn1.sql	NBREARN	NBAJOBS PEA1PAY PEAHIRE PEAESCH
NB_JOB_ LABOR	JOB_LABOR	nb_job_labor nb_job_labor_rules nb_job_labor_strings	nbkb_job_labor0.sql nbkb_job_labor1.sql nbkb_job_labor_r0.sql nbkb_job_labor_r1.sql nbkb_job_labor_s0.sql nbkb_job_labor_s1.sql nbkd_nbrjlbd0.sql nbkd_nbrjlbd1.sql	NBRJLBD	NBAJOBS PEA1PAY PEAHIRE PEAESCH

Business Process Interface (BPIs) Packages

Three BPIs have been created to maintain the employee jobs information. Each BPI, in turn, invokes one or more of the Jobs APIs. It is important to note that the APIs alone cannot be used to maintain the jobs data. Applications must invoke the BPIs to ensure data integrity.

The following table provides a brief overview of the Jobs APIs associated with the Employee Jobs Form (NBAJOBS). For more details, refer to the Position Control API Reference Guide, Release 7.3.

BPI Object Name	BPI Entity Name	Associated BPI Package	Associated API Packages	Associated BPI scripts	Associated Table	Affected Objects (Forms, Processes, etc.)
JOB_ LABOR	JOB_LABOR	np_job_labor	nb_job_labor nb_job_labor_rules nb_job_labor_strings	nbkp_job_ labor0.sql nbkp_job_ labor1.sql	NBRJLBD	NBAJOBS PEA1PAY PEAHIRE PEAESCH
JOB_ ASSIGNMENT	JOB_ ASSIGNMENT	np_job_ assignment	nb_job_detail nb_job_detail_ rules nb_job_detail_ strings nb_job_base nb_job_base_rules nb_job_base_ strings	nbkp_job_ assignment0.sql nbkp_job_ assignment1.sql	NBRJOBS NBRBJOB	NBAJOBS PEA1PAY PEAHIRE PEAESCH
JOB_EARNINGS	JOB_EARNINGS	np_job_ earnings	nb_job_earnings nb_job_earnings_ rules nb_job_earnings_ strings	nbkp_job_ earnings0.sql nbkp_job_ earnings1.sql	NBREARN	NBAJOBS PEA1PAY PEAHIRE PEAESCH

Combined Deduction Limits APIs

The following table provides a brief overview of the APIs for combined deduction limits, which are new to Banner 8.0. For more details, refer to the *Banner 8 Human Resources Release Guide*.

API Object Name	API Entity Name	Associated API Packages	Associated scripts	Associated Table	Affected Objects (Forms, Processes, etc.)
PB_DEDN_COMB_LMT	DEDN_COMB_LMT	pb_dedn_comb_lmt pb_dedn_comb_lmt_rules pb_dedn_comb_lmt_strings		PTRBDCL	PTRBDCL

SunGard Higher Education Help Resources

Introduction

This section discusses the various help resources provided by SunGard Higher Education.

Sources of Help

Sources

- Discussion lists
  - http://lists.sungardhe.com
- Customer Support Center
  - Log on from http://www.sungardhe.com
- Action Mail
  - ambanhr@sungardhe.com
- Action Line
  - 800.522.4TCP

SunGard Higher Education Discussion lists

Discussion lists

SunGard Higher Education supports a number of useful Discussion lists:

- boracle - ORACLE issues
- bstudent - Banner Student Product issues
- bgeneral - Banner General Product issues
- bhumres - Banner Human Resources Product issues
- bar - Banner AR Product Issues
- breport - Banner Reporting Issues
- btrain - Banner Training Issues
- bsmall - Small organization Issues
- blarge - Large organization Issues
- and more...

Obtaining a listing

For a complete listing of SunGard Higher Education's discussion lists, send a message to:

- listserv@sungardhe.com
  - No subject
  - Message body: `LISTS`

Or go to http://lists.sungardhe.com

Subscribing

To subscribe to a list, send mail to:

- listserv@sungardhe.com
  - Leave subject field blank
  - In message body:
    ```
    Subscribe listname1 YourFirstName YourLastName
    Subscribe listname2 YourFirstName YourLastName
    ```

Note: To subscribe to multiple lists in the same email, put each subscribe command on a separate line.

Unsubscribing

To unsubscribe from a list, send mail to:

- listserv@sungardhe.com
  - Leave subject field blank
  - Message body:
    ```
    Unsubscribe listname
    ```

SunGard Higher Education Support

Customer Support Center

- Browse open and resolved contacts of your organization in detail
- Browse all product defects reported by your organization
- Browse the Known Issues Reports for defect descriptions, corrections, and workarounds
- Request Electronic Distribution Download of software modifications and other updates
- Browse the frequently asked questions (FAQ)

Accessing ActionWeb

To request access to SunGard Higher Education's “secured” Customer Support Center pages, send an email request to:

- csr@sungardhe.com
- Include your name, phone number, organization name

SunGard Higher Education Action Mail

- Send Action Line questions through email
- Action Mail requests/questions are given the same priority as phone requests
- Use for less time-critical issues

Contacting Action Mail

To contact Action Mail, send an email to:

- ambanhr@sungardhe.com

Send the following information:

- Your name
- Your organization
- Your product and release number
- Severity of problem (high, medium, low)
- Your e-mail address and phone number
- Your question (detailed description)

SunGard Higher Education Action Line

Phone/Fax help for time-critical issues.

To contact the Action Line:

Phone: 1-800-522-4TCP

FAX: 1-610-725-7430