Banner General Release Guide

Release 8.3
December 2009

SUNGARD HIGHER EDUCATION

Trademark, Publishing Statement and Copyright Notice

SunGard or its subsidiaries in the U.S. and other countries is the owner of numerous marks, including “SunGard,” the SunGard logo, “Banner,” “PowerCAMPUS,” “Advance,” “Luminis,” “fsaATLAS,” “DegreeWorks,” “SEVIS Connection,” “SmartCall,” “PocketRecruiter,” “UDC,” and “Unified Digital Campus.” Other names and marks used in this material are owned by third parties.

© 2009 SunGard. All rights reserved.

Contains confidential and proprietary information of SunGard and its subsidiaries. Use of these materials is limited to SunGard Higher Education licensees, and is subject to the terms and conditions of one or more written license agreements between SunGard Higher Education and the licensee in question.

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.

Prepared by: 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

Documentation Feedback

http://education.sungardhe.com/survey/documentation.html

Distribution Services E-mail Address

distserv@sungardhe.com

Revision History Log

Publication Date	Summary
December 2009	New version that supports Banner General 8.3 software.

Think Before You Print.

Banner General 8.3 Release Guide

Contents

Introduction

This guide documents Release 8.3 of the Banner General System. Release 8.3 includes problem resolutions and the following enhancements:

Payment Processor Connection

The Payment Processor Connection is a new solution for entering payment card information for Banner Self-Service applications.

Advanced Queuing

Banner General 8.3 introduces an optional alternative to `DBMS_PIPE` for session communications. This alternative communication mechanism uses Oracle Advanced Queuing (AQ). These changes will be of interest to Banner customers with database environments that use Oracle's Real Application Clusters (RAC) technology.

Prerequisites for Processing IATs

Banner General 8.3 contains changes for processing IATs directed to U.S. domestic bank accounts in U.S. dollars. These changes were originally delivered with Banner General 8.2.1. Depending on the Banner products installed at your institution, the following upgrades may be required when Banner General 8.3 is installed:

- Banner Finance patch `p1-72jw9z_fin80300` is required for processing Accounts Payable IATs. This patch must be applied, or you must upgrade to Banner Finance 8.4.
- Banner Human Resources patch `p1-72hngx_pay80203` is required for processing Payroll IATs. A second Banner Human Resources patch, `p1-6btvn7_pay80202`, includes a fix for RPE# 1-X4FTP. Both patches must be applied for Banner Human Resources IAT updates, or you must upgrade to Banner Human Resources 8.3.

Note

Upgrades for your installed products should be installed at the same time.

Cumulative Documentation

This document provides detailed information about the Banner General 8.3 release only. Banner General 8.3 is a cumulative release that also includes enhancements, RPEs, and problem resolutions delivered in the General 8.2.1 and 8.2.1.1 releases. For complete documentation about these interim releases, please refer to the interim release guides identified in the table that follows.

Release Number	Contents	Release Date
8.2.1	International ACH Transactions	October 2009

1 Payment Processor Connection - Functional

Overview

The Payment Processor Connection is a new solution for entering payment card information for Banner Self-Service applications.

This document describes the changes in Banner General that support this new solution. For more information about setting up and using this new solution, refer to the following documents:

- *Banner Web General Release Guide 8.3*
- *Banner Web Tailor Release Guide 8.3*
- *Banner General Web Services Handbook*
- Refer to the following FAQs for the latest information about this feature:
  - FAQ 1-6CH32F
  - FAQ 1-8PA0EV
  - FAQ 1-8PW7AF

This enhancement allows institutions to process transactions using payment cards with the new Payment Processor Connection functionality. This replaces the Java payment processing that was used previously. That payment gateway is no longer supported.

Payment card information is no longer stored in Banner for Self-Service transactions. Admissions application fees, registration fee assessment fees, enrollment verification fees, transcript request fees, graduation application fees, and alumni gift payments are now processed by third party vendors. Web technologies (URL redirects and Web services) are used to connect Banner to the third party vendors for payment processing and transaction confirmation. When transactions are processed successfully, the updates can be checked in Banner Accounts Receivable. The merchant ID information in Banner Accounts Receivable and Banner Finance will be populated with the vendor transaction ID number. Only one payment vendor can be active at a time for transactions.

Prerequisites

The Banner Payment Transaction Service Adapter must be installed to Oracle Application Server 10g. The adapter also requires the following Banner products:

Banner 8.X

Product	Minimum Version
Banner General	8.0 with patch `p1-639eqc_gen80200`
Banner Web General	8.0 with patch `p1-639es5_bwg80200`
Banner Web Tailor	8.0 with patch `p1-7549uz_twb80201`

Background

The Payment Card Industry (PCI) Security Standards Council is a forum that develops, enhances, stores, disseminates, and implements security standards for payment card transactions on a global basis. There are two types of security standards:

- The Payment Card Industry Data Security Standard (PCI-DSS) helps organizations proactively protect customer account data.
- The Payment Application Data Security Standard (PA-DSS) helps software vendors develop secure payment applications that comply with the PCI-DSS. The PA-DSS applies to payment applications that are sold, distributed, or licensed to third parties.

Note

A payment application is a software application that stores, transmits, or processes cardholder data as part of the authorization or settlement of a payment card transaction.

SunGard Higher Education does not market or intend for its Banner applications to be payment applications. However, the functionality for entering payment card information in Banner Self-Service and the use of the Java Payment Client (JPC) suggest that Banner would likely be deemed a payment application based on recent changes to the PCI-DSS and the PA-DSS.

For this reason, SunGard Higher Education is replacing the JPC and the ability to enter payment card information in all Banner Self-Service applications. The new solution integrates to payment processing vendors. Your institution can now consolidate the processing of payment card information and regulatory compliance to a single, PCI-DSS compliant application or service provided by the payment processing vendor with whom you already have a relationship.

These changes apply to all institutions that currently use Banner Self-Service and the JPC to enter payment card information. Failure to install this patch can jeopardize your compliance with PCI-DSS. If you previously modified your payment process to redirect access to an alternate payment site, you should review your policies and the changes in this patch to determine the best course of action.

Processing

Previously, payment card processing was split between Banner and the payment processing vendor. A user performed a task (such as pay tuition or make a donation) on a Banner Self-Service page. Banner collected and stored payment card details, then sent a message via a secure link (via the Java Payment Client) to the payment processing vendor. The payment processing vendor processed the information and returned a response to Banner, acknowledging success or failure of the payment.

In the new solution, a user initiates the same task (such as pay tuition or make a donation) on a Banner Self-Service page. However, instead of entering the payment card details in Banner, the user is redirected to the payment processing vendor. This is where the payment card details are captured and validated. The result is transmitted to Banner where a Web service is called to process and store the payment transaction. Then the user is returned to the appropriate location in Banner Self-Service.

Self-Service payment flow

When a user completes a process in Self-Service that requires payment and a payment card is used for payment, the following occurs.

1. The user enters the payment amount in Self-Service if the amount is not pre-determined in Banner.

   Once the amount has been submitted to the payment vendor Web site, it cannot be changed.

2. The user clicks the button on the Self-Service page to submit the transaction.

   The button can be named **Submit**, **Submit Payment**, **Submit Request**, and so on.

3. The user's browser is redirected to the payment vendor site, based on the Web Tailor `PAYVED_URL` parameter.

4. The user may be prompted to sign in to the payment vendor site. This depends on the vendor.

5. The user enters the payment data, such as the payment card information required by the payment vendor.

6. The user follows the instructions on the payment vendor site to process the transaction.

7. A payment vendor wait screen may be displayed while the transaction is processed.

   This warns the user against clicking on the browser **Back** button and accidently resubmitting the transaction.

8. The payment vendor site directs processing back to Banner, where the appropriate Banner tables are updated.

9. The confirmation page from the payment vendor is displayed when a successful posting of the transaction is returned from Banner.

10. The payment vendor may send a confirmation email to the payee.

11. The user follows instructions on the payment vendor site to return to Self-Service and view a receipt or a signature page as appropriate.

If a payment vendor failure occurs (the transaction is invalid), the payment vendor site displays a failure confirmation page. Error messages displayed here are defined by the payment vendor.

Depending on the payment vendor, the user then has the opportunity to resubmit the payment information (i.e., use a different payment card). If the user chooses to not complete the transaction, instructions on the payment vendor site are used to cancel the transaction or log out of the site. Processing then returns to Banner, and a message is displayed to the user indicating that a failure occurred on the payment vendor site.

If a processing or posting failure occurs in Banner, a warning message is displayed to the user indicating that a charge may have been posted to their payment card.

Data Transmitted from Banner to Payment Vendor

The following data is transmitted to the vendor through the URL redirect:

- TransactionId (Length = 20)
- TransactionAmount (Length = 15.2)
- TransactionDescription (Length = 30)
- MerchantID (also known as CompanyID, OrderType, etc.) (Length = 30)

Data Transmitted from Payment Vendor to Banner

The following data is transmitted to Banner through the Payment Transaction Web Service:

- Vendor payment transaction ID
- Banner payment transaction ID
- Vendor status/transaction result code, *1* (successful) or *0* (failed)
- Payment amount to vendor
- Vendor merchant (bank) information, payment card used
- Vendor error messages
- Vendor authorization code

If the payment vendor transaction result code is *1*, Banner General updates the transaction in the GORCCAU table as successful and returns a *Success* status to the payment vendor application, which then redirects the user to the Self-Service page from which the transaction originated.

If a vendor transaction result code is *0*, the corresponding GORCCAU record is updated to indicate why the transaction was not successful, and a status of *Failure* is returned to the payment vendor application. The payment vendor application redirects the browser to a failure page, which displays an error message. If no information is available from the payment vendor to indicate why the transaction failed, the status on GORCCAU will remain as *Transaction Started*.

Process Name Codes

Process name codes on GTVPROC are used to identify to the vendor which Banner system is producing the transaction. A merchant ID is included in the outgoing transaction from Banner. The merchant ID can be populated with the process code that defines the Self-Service page used for the transaction, or it can be supplied by the payment vendor. The process code values are system-required and must be used by all institutions.

Process Name Code	Description	Banner System
WEBCCALUGIFT	Web Credit Card Advancement Gift Process	Advancement
WEBCCAPPDEP	Web Credit Card Application Deposit Process	Student
WEBCCAPPFEES	Web Credit Card Application Fees Process	Student
WEBCCENROLLDEP	Web Credit Card Enrollment Deposit	Student

Process Name Code	Description	Banner System
WEBCCEPRTREQ	Web Credit Card Enrollment Verification Charge	Student
WEBCCGRADAPP	Web Credit Card Graduation Application Process	Student
WEBCCREGFEES	Web Credit Card Registration Fees Process	Student
WEBCCTRANSREQ	Web Credit Card Request Process	Student

The descriptions for the process codes are delivered as information text for `gokpven.f_process_payment_updates`. A default description is delivered for use with process codes that do not have information text entries or when the info text entries are null. The available length is 30 characters.

Crosswalk from GTVCCRD to GOAMERC

The return of data to Banner includes a transaction description, which is the key to the crosswalk on GTVCCRD for the payment card type that is used, and the above process name code/merchant ID to determine the accounting information that is inserted into Banner. The **Credit Card Type Code** (`GTVCCRD_CODE`) and the **Process Name Code** (`GTVPROC_CODE`) are used to look up the accounting information on GOAMERC for the cashier ID and detail code. The **Third Party Transaction** information (`MERCHANT_ID`) is used in Accounts Receivable as the merchant ID. Only the credit card type code and the process name code are matched to the GOAMERC record.

Information must be set up on GOAMERC with every combination of process code, credit card code, system code, and merchant ID (third party transaction) that is used by your institution. For each combination, the associated payment detail code and cashier ID must also be defined. When the payment vendor returns a payment transaction, the GOAMERC record that matches on the process code, credit card code, system code, and merchant ID is used to identify the detail code and cashier ID for the payment transaction.

Here are some crosswalk examples.

Example 1:

A user on the Advancement gift page sends a transaction for the amount of $50. The merchant ID is *WEBCCALUGIFT*. The return to Banner includes the transaction description *ALUVISA*. A Banner user can look up *ALUVISA* in the crosswalk to identify the card type. The card type plus the process ID (*WEBCCALUGIFT*) indicates the accounting information that is recorded in Banner for the gift.

Example 2:

A user on the Student admissions application page sends a transaction for the amount of $50. The merchant ID is *WEBCCAPPFEES*. The return to Banner includes the

transaction description *APPVISA*. A Banner user can look up *APPVISA* in the crosswalk to identify the card type. The card type plus the process ID (*WEBCCAPPFEES*) indicates the accounting information that is recorded in Banner for the application fees.

Example 3:

A user on the Advancement page donates $50 with credit. The transaction is for $50. The merchant ID is *WEBCCALUGIFT*. The transaction description returned to Banner is *VISA*.

That user then requests a transcript, and the transaction amount is $45. The merchant ID is *WEBCCTRANSREQ*. The transaction description returned to Banner is *VISA*.

The user can have a single transaction description and card type for both/all Self-Service pages/transactions.

The return to Banner includes the transaction description *ALUVISA*. A Banner user can look up *ALUVISA* in the crosswalk to identify the card type. The card type plus the process ID (*WEBCALUGIFT*) indicates the accounting information that is recorded in Banner for the donation.

Note

Institutions need to confirm the crosswalk values used on GTVCCRD with their vendors and validate each payment card type during testing.

Integration URLs

Four URLs constitute the payment processor connection. Each URL must be configured either in Banner, or in the payment vendor application in order for the connection to function.

URL	Purpose	Provided By	Configured In
Payment Vendor URL	Used to redirect user from Banner to payment vendor application	Payment vendor	Web Tailor
Web Service Endpoint URL	Used by payment vendor application to update Banner GORCCAU table	Payment Transaction Web Service Adapter deployment (`<adapter application server URL>/pci/v1_0`)	Payment vendor application

URL	Purpose	Provided By	Configured In
Banner Success Page URL	Used to redirect user back to Banner when payment transaction is successful	Banner Self-Service deployment (`<Self-Service URL>/gokpurl.p_payment_success_return`)	Payment vendor application
Banner Failure Page URL	Used to redirect user back to Banner when payment transaction fails	Banner Self-Service deployment (`<Self-Service URL>/gokpurl.p_payment_failure_return`)	Payment vendor application

GTVSDAX Rules

Rules on GTVSDAX have been modified for this enhancement. Existing rules have been updated to use a new internal group code, and other rules have been made obsolete.

Renamed Rules

The following rules have been renamed for use with payment card payment. The internal group code of *CRDCARDPMT* has been replaced with a new internal group code of *PAYMENTVENDOR*.

External Code	Internal Code	Internal Code Seq Number	Internal Code Group	Description	Activity Date
UPDATE ME	DSPALUDESG	1	PAYMENTVENDOR	Default Designation #1	Sysdate
UPDATE ME	DSPALUDESG	2	PAYMENTVENDOR	Default Designation #2	Sysdate
UPDATE ME	DSPALUDESG	3	PAYMENTVENDOR	Default Designation #3	Sysdate
Y, N	DSPALUGIFT	N/A	PAYMENTVENDOR	Display Adv Gift Number Web CC	Sysdate
Y, N	DSPALUID	N/A	PAYMENTVENDOR	Display Adv Donor ID Web CC	Sysdate
W	PMTSRCE	N/A	PAYMENTVENDOR	Source Code for Web Payment	Sysdate
PRINTERDEF	PRINTERDEF	N/A	PAYMENTVENDOR	Printer Definition for Web CC	Sysdate
Y, N	USEAPPMMID	N/A	PAYMENTVENDOR	Use Multiple Merchant IDs	Sysdate

External Code	Internal Code	Internal Code Seq Number	Internal Code Group	Description	Activity Date
Y, N	USESTUMMID	N/A	PAYMENTVENDOR	Use Multiple Merchant IDs	Sysdate
Y, N	WEBCCHOLDS	N/A	PAYMENTVENDOR	Allow CC Payments if A/R Holds	Sysdate

Obsolete Rules

The following rules used previously with payment card processing are obsolete with this release:

External Code	Internal Code	Internal Code Seq Number	Internal Code Group	Description	Activity Date
D	AUDITMODE	N/A	CRDCARDPMT	Payment Audit Mode	Sysdate
Y, N	AUTHSETTLE	N/A	CRDCARDPMT	Authorize and settle together	Sysdate
Y, N	DSPADMSPER	N/A	CRDCARDPMT	Admissions CC Separator	Sysdate
0 - 0100	DSPADMWIDT	N/A	CRDCARDPMT	Admissions CC Display Width	Sysdate
Y, N	DSPALUSEPR	N/A	CRDCARDPMT	Alumni CC Separator	Sysdate
0 - 0100	DSPALUWIDT	N/A	CRDCARDPMT	Alumni CC Display Width	Sysdate
Y, N	DSPSTUID	N/A	CRDCARDPMT	Display Account ID for Web CC	Sysdate
Y, N	DSPSTUSEPR	N/A	CRDCARDPMT	Student CC Separator	Sysdate
Y, N	DSPSTUTRAN	N/A	CRDCARDPMT	Display Tran Num for Web CC	Sysdate
0 - 0100	DSPSTUWIDT	N/A	CRDCARDPMT	Student CC Display Width	Sysdate
Y, N, M, 4	GFTSTORENO	N/A	CRDCARDPMT	Store Alumni Web Gift Info	Sysdate
tpg, ops, eps	PMTSERVER	N/A	PMTSERVER	Payment Server	Sysdate
Y, N, M, 4	PMTSTORENO	N/A	CRDCARDPMT	Store Web Payment Info	Sysdate
Y, N	SETLCOMMIT	N/A	CRDCARDPMT	Commit when settlement error	Sysdate

Setup Requirements

The following steps must be performed to set up Banner for the Payment Processor Connection:

- Step 1, Define GTVSDAX settings.
- Step 2, Define payment cards.
- Step 3, Define processes that use payment processing.
- Step 4, Define payment combinations.
- Step 5, Define redirects from Banner Self-Service to payment vendor.
- Step 6, Define redirects from payment vendor to Banner Self-Service.

The following section describes each step in detail.

Step 1 Define GTVSDAX settings

Use the following steps to enter rules on the Crosswalk Validation Form (GTVSDAX) for payment transactions.

1. Access the Crosswalk Validation Form (GTVSDAX).

2. Use the following GTVSDAX rule to define the source code that is used for payment transactions made via Banner Self-Service.

Internal Code	*PMTSRCE*
Sequence	none
Group	*PAYMENTVENDOR*
External Code	Source code from TTVSRCE
Description	*Source Code for Web Payment*

3. Use the following GTVSDAX rule to define the accounts receivable holds indicator.

Internal Code	*WEBCCHOLDS*
Sequence	none
Group	*PAYMENTVENDOR*
External Code	*Y* Students cannot make payments via Banner Self-Service if they have holds on their accounts receivable records.
	N Students can make payments via Banner Self-Service if they have holds on their accounts receivable records.
Description	*Allow CC Payments if A/R Holds*

4. Use the following GTVSDAX rule to define the receipt print label. This label is assigned to Banner Self-Service payments and controls the printing of hardcopy receipts.

Internal Code	*PRINTERDEF*
Sequence	none
Group	*PAYMENTVENDOR*
External Code	Printer code from GTVPRNT
Description	*Printer Definition for Web CC*

5. Use the following GTVSDAX rule(s) to define the address hierarchy for payment transactions. This hierarchy provides the order in which address types are used to update address information in the Banner database for payment transactions.

Internal Code	*WEBCCADDR*
Sequence	Sequence numbers used to prioritize records in the hierarchy
Group	*ADDRESS*
External Code	Address type code (for example, *PR* for permanent address)
Description	*Web CC Address Hierarchy*

6. Use the following GTVSDAX rule to define the default term to use when no term is associated with the detail code in TBBDETC.

Internal Code	*DEFAULT*
Sequence	none
Group	*WEBCCDEFTERM*
External Code	Term code from STVTERM
Description	*Default Term for Web CC Insert*

7. Use the following GTVSDAX rules to indicate whether multiple merchant (profile) IDs are used for Banner Student Self-Service payment transactions.

   Your payment vendor provides the merchant IDs, which identify the profiles that are used for payment transactions. You can set up a hierarchy of merchant IDs based on level, campus, or college. Refer to the *Banner Web Credit Card Payments Handbook* for more details on setting up a merchant ID hierarchy.

   The first rule applies to Banner Student Self-Service admissions applications. The second rule applies to all other Banner Student Self-Service payment transactions.

Internal Code	*USEAPPMMID*
Sequence	none

Group	*PAYMENTVENDOR*
External Code	*Y* Multiple merchant IDs are used. *N* Multiple merchant IDs are not used.
Description	*Use Multiple Merchant IDs*
Internal Code	*USESTUMMID*
Sequence	none
Group	*PAYMENTVENDOR*
External Code	*Y* Multiple merchant IDs are used. *N* Multiple merchant IDs are not used.
Description	*Use Multiple Merchant IDs*

8. If multiple merchant (profile) IDs are used for Banner Student Self-Service payment transactions, use the following GTVSDAX rules to define the hierarchy of merchant IDs.

   The first set of rules defines a hierarchy for Banner Student Self-Service admissions applications payment transactions. The second set of rules defines a hierarchy for all other Banner Student Self-Service payment transactions.

Note

If you do not set up a merchant ID hierarchy based on level, campus, or college, then the *DEFAULT* rule determines the merchant ID that is used to process payment transactions. ■

Internal Code	Criteria to be matched *(DEFAULT, LEVEL, CAMPUS,* or *COLLEGE)*
Sequence	Sequence number used to prioritize records in the hierarchy
Group	*WEBAPPCCID*
External Code	Merchant ID provided by the payment vendor to identify the profile for Banner Self-Service admission application payment transactions
Description	*Merchant ID for CC*
Internal Code	Criteria to be matched *(DEFAULT, LEVEL, CAMPUS,* or *COLLEGE)*
Sequence	Sequence numbers used to prioritize records in the hierarchy
Group	*WEBSTUCCID*
External Code	Merchant ID provided by the payment vendor to identify the profile for Banner Student Self-Service payment transactions (except admission applications)
Description	*Merchant ID for CC*

9. (Optional) If Banner Advancement is installed at your institution, enter the following rules on GTVSDAX.

Internal Code	*DEFAULT*
Sequence	none
Group	*WEBALUCCID*
External Code	ID provided by payment vendor to identify the profile used for gifts
Description	*Default for Adv CC Merch ID*

Internal Code	*DSPALUDESG*
Sequence	1, 2, 3
Group	*PAYMENTVENDOR*
External Code	Default campaign code (five characters) plus designation code
Description	*Default designation #1, #2, #3*

Internal Code	*DSPALUGIFT*	
Sequence	none	
Group	*PAYMENTVENDOR*	
External Code	*Y*	Display gift number on online receipt.
	N	Do not display gift number on online receipt.
Description	*Display Adv Gift Number Web CC*	

Internal Code	*DSPALUID*	
Sequence	none	
Group	*PAYMENTVENDOR*	
External Code	*Y*	Display donor ID on online receipt.
	N	Do not display donor ID on online receipt.
Description	*Display Adv Donor ID Web CC*	

Step 2 Define payment cards

Various payment cards can be used within Banner Self-Service. Examples include American Express, Visa, and MasterCard. Payment card codes stored within Banner must be cross-referenced to the payment card codes sent by your payment vendor. Use the following steps to identify the payment cards that can be used for payments at your institution.

1. Access the Credit Card Validation Form (GTVCCRDD).

2. Enter the following information for each payment card that can be used at your institution:

Code	Payment card code that is stored in Banner
External Merchant ID	Payment card code sent by the payment vendor
Description	Description of the payment card

3. Save.

Step 3 Define processes that use payment processing

Various Banner Self-Service processes can use payment processing. Examples include processes for gifts, application fees, graduation applications, registration fees, and transcript requests. A process code tells the payment vendor which Banner process produced a payment transaction. Use the following steps to identify the processes that use payment processing at your institution.

1. Access the Process Name Validation Form (GTVPROC).

2. Enter the process code in the **Code** field.

   This code identifies the Banner Self-Service process that uses Web payment processing. For example, *WEBCCREGFEES* is the Banner Student Self-Service process that handles payments within registration.

3. Enter the process description in the **Description** field.

4. Save.

Step 4 Define payment combinations

The Credit Card Merchant ID Form (GOAMERC) is used to identify each combination of process code, payment card code, system code, and merchant ID (third party transaction ID) that your institution uses. For each combination, you must define the associated payment detail code and cashier ID.

When your payment vendor returns a payment transaction, the GOAMERC record that matches on process code, payment card code, system code, and merchant ID is used to determine which detail code and cashier ID are used for the payment transaction.

Use the following steps to set up records on GOAMERC.

1. Access the Credit Card Merchant ID Form (GOAMERC).

Note

The key block is used to enter search criteria for displaying information in the next block. Any or all fields in the key block can be left blank. ■

2. Go to the Credit Card Merchant ID block.

3. For every combination of process code, payment card code, system code, and merchant ID, insert a new record with the following information:

Process	Banner process that uses payment processing (for example, *WEBCCREGFEES*).
Credit Card	Payment card that is used with payment processing (for example, *VISA*).
System	Banner system (for example, *S* for Banner Student).
Third Party Transaction	Merchant ID provided by the payment vendor to identify the profile that is used for payment transactions.
Detail	Code used to identify payment detail. This code depends on the process (for example, a detail code in Banner Student Self-Service or a gift type in Banner Advancement Self-Service).
Voice Response Message	Message number that is spoken to a Banner Voice Response user to identify the payment card type. Optional.
Active	Check box that indicates if the record is active.
Cashier ID	User ID for the cashiering session when payments for the process, system, payment card type, and merchant ID are inserted into Banner.

4. Save.

Step 5 Define redirects from Banner Self-Service to payment vendor

When a Banner Self-Service user makes a payment, the Banner Self-Service application redirects the browser to an external payment vendor. Use the following steps to define the URL for the payment vendor.

1. Enter the Secure Area of Banner Self-Service.
2. Navigate to Web Tailor Administration.
3. From the Web Tailor menu, select Web Tailor Parameters.
4. Enter the following parameters:

Parameter	Description
PAYVEND_TRANS_TIMEOUT	Number of minutes after which the payment vendor's Web site times out if there is no activity. An update received from the vendor after this time limit is treated as a "transaction not found." An error message indicates the current date/time, transaction date/time, and expiration date/time.
PAYVEND_URL	Payment vendor's URL. The user's browser is redirected from Banner Self-Service to this URL to complete a payment transaction. **Example:** `http://m039087.sungardhe.com:6018/pci/gateway` **Note:** Only one vendor can be active at a time.

Step 6 Define redirects from payment vendor to Banner Self-Service

The payment vendor processes a payment and then redirects the browser to a success or failure URL in Banner Self-Service, depending on whether the transaction update via the PaymentTransaction Web service was a success or failure.

You must provide the success and failure URLs to your payment vendor. These URLs must be consistent with your Banner Self-Service login URLs. For example, if the redirect from the payment vendor to Banner must be via SSL, then Banner Self-Service sessions must be started via SSL. Conversely, if Banner Self-Service sessions are started without SSL (http rather than https), then the redirect from the payment vendor to Banner Self-Service must be http. This consistency ensures that the SESSID cookie is found and that the user is taken directly to the proper Banner Self-Service page without requiring the user to log in.

Use the following structures to define your success and failure URLs:

Success URL	`<http or https>://<Banner Self-Service server>/gokpurl.p_payment_success_return`
Failure URL	`<http or https>://<Banner Self-Service server>/gokpurl.p_payment_failure_return`

New Form

The following form is new for this enhancement.

Vendor Payment Transaction Audit Form (GOICCAU)

This form is used to query on payment card transactions. Each record shows transaction and vendor detail.

Main Window

This window contains the Key Block and the Data block.

Key Block

Use the Key Block to query by ID, transaction ID, transaction reference number, and transaction date. This block can also be left blank to query on all transactions.

Field	Description
ID	ID and name of person for whom payment card transactions are queried. List - Person Search Form (SOAIDEN) Count Query Hits - Non-Person Search Form (SOACOMP) Duplicate Item - SSN/SIN Alternate ID Search Form (GUIALTI)
Transaction ID	Unique ID number associated with the external vendor payment. If blank, all transactions will be displayed for the ID.
Ref No	Reference number returned from the payment vendor that can be tied back to the payment card transaction.
Date	Date of the payment card transaction.

Data Block

Use the Data Block to view the payment card transaction and vendor detail records for the ID in the Key Block.

Field	Description
Transaction ID	Unique ID number associated with the external vendor payment.
Ref No	Reference number returned from the payment vendor that can be tied back to the payment card transaction.
Auth Code	Authorization code returned from the payment vendor that can be tied back to the payment card transaction.
Amount	Amount of the payment card transaction.
Pay Card	Payment card associated with the transaction, such as VISA, Mastercard, and so on.
Merchant	Third party transaction code of merchant processing the transaction.
ID	ID of the user who requested the payment card transaction.
System	System code for Banner product associated with the transaction, such as *A* (Advancement), *S* (Student), and so on.
Date	Date of the payment card transaction.
Term	Term code associated with the payment card transaction.
Application	Admissions application sequence number associated with the transaction.
Gift No	Advancement gift number associated with the transaction.
Process	Process name code (GTVSDAX rule) associated with the transaction.
Location	Point in the transaction when the record was created.
Sub Code	Merchant ID based upon calling application.
Status	Status of the payment card transaction.
Banner Status	Last status of the transaction from a Banner perspective.
Vendor Status	Last status of the transaction from a payment vendor perspective.
Vendor Message	Error message from the payment vendor related to a failed transaction, if applicable.

Field	Description
Appl Data	Application specific data for the transaction. This data depends on the application used to create the transaction, such as Banner Student. This field is not used for all transactions.
Update Function	Name of the function called when a successful pre-authorization is returned from the payment vendor.
Success URL	Banner Self-Service URL for the Web page displayed when the transaction is processed successfully.
Failure URL	Banner Self-Service URL for the Web page displayed when the transaction is not processed successfully.

Changed Form

The following form has been modified for this enhancement.

Credit Card Validation Form (GTVCCRD)

The **External Merchant ID** field has been added to the form. This field is used to create a crosswalk for the vendor merchant ID that the payment is applied to.

When the vendor calls the process payment update function, the credit card code and the merchant ID from the vendor are read from the **Code** field and the **External Merchant ID** field on GTVCCRD. The process then updates the GORCCAU table. If the value for the external merchant ID cannot be found, a Banner update failure message will be returned to the vendor. This new field is delivered with null values so you can enter the appropriate values.

An edit has been added to the form to prevent the entry of duplicate external merchant IDs.

Field	Description
External Merchant ID	Credit card code sent by the payment vendor that creates a crosswalk for the vendor merchant ID that the payment is applied to.

Obsolete Form

The following form is obsolete.

Credit Card Rules Form (GORCCRD)

This form is no longer used with payment vendor processing.

Changed Menu

The following menu has been updated for this enhancement.

Miscellaneous General Forms Menu [*GENMISC]

The new Vendor Payment Transaction Audit Form (GOICCAU) has been added to this menu.

2 Payment Processor Connection - Technical

Changed Tables

The following tables have been modified for this enhancement.

Banner Credit Card Audit Table (GORCCAU)

This table has been modified for use with payment card processing. When the payment vendor site is called, transaction details are inserted into the table and the payment transaction ID is created.

The following columns have been added:

GORCCAU_PAY_TRANS_ID	NUMBER
GORCCAU_VENDOR_STATUS	VARCHAR2(40)
GORCCAU_BANNER_STATUS	VARCHAR2(40)
GORCCAU_VENDOR_ERROR_MSG	VARCHAR2(4000)
GORCCAU_APPLICATION_DATA	VARCHAR2(4000)
GORCCAU_UPDATE_FUNCTION	VARCHAR2(100)
GORCCAU_SUCCESS_URL	VARCHAR2(1020)
GORCCAU_FAILURE_URL	VARCHAR2(1020)
GORCCAU_VENDOR_AUTH_CODE	VARCHAR2(50)
GORCCAU_LAST_NAME	VARCHAR2(60)
GORCCAU_FIRST_NAME	VARCHAR2(60)
GORCCAU_MIDDLE_NAME	VARCHAR2(60)
GORCCAU_NAME_PREFIX	VARCHAR2(20)
GORCCAU_NAME_SUFFIX	VARCHAR2(20)

GORCCAU_SUB_CODE	VARCHAR2(20)
GORCCAU_PROC_CODE	VARCHAR2(30)
GORCCAU_SYSI_CODE	VARCHAR2(02)
GORCCAU_APPL_NO	NUMBER(3)
GORCCAU_CITY	VARCHAR2(50)
GORCCAU_AIDM	NUMBER(8)
GORCCAU_GIFT_NO	VARCHAR2(07)
GORCCAU_VENDOR_REFER_NO	VARCHAR2(30)
GORCCAU_CART_ID	VARCHAR2(60)

The GORCCAU_PAY_TRANS_ID column is set equal to BAN_PAYMENT_SEQUENCE.nextval.

The existing GORCCAU_AMOUNT column has been changed from type VARCHAR2 to NUMBER(12,2).

The following columns are no longer used and have been dropped:

- GORCCAU_CCRD_NUM
- GORCCAU_EXP_MO
- GORCCAU_EXP_YR
- GORCCAU_SENT_STRING
- GORCCAU_RETURN_STRING

The following Primary Key Constraint has been added for use with the transaction ID:

```
PK_GORCCAU PRIMARY KEY (GORCCAU_PAY_TRANS_ID)
```

The entire table is described below for reference.

Column Name	Nullable	Type	Comments
GORCCAU_ID	No	VARCHAR2(9)	ID: ID of user requesting credit card transaction.
GORCCAU_ACTIVITY_DATE	No	DATE	ACTIVITY DATE: Timestamp of audit record.
GORCCAU_PIDM	Yes	NUMBER(8)	PIDM: PIDM of user requesting credit card transaction.
GORCCAU_TERM_CODE	Yes	VARCHAR2(6)	TERM: Term Code associated with credit card transaction.
GORCCAU_LOCATION	Yes	VARCHAR2(120)	LOCATION: Point in transaction when record was created.

Column Name	Nullable	Type	Comments
GORCCAU_DETAIL	Yes	VARCHAR2(20)	DETAIL: Detail/Gift Code associated with transaction.
GORCCAU_STREET_LINE1	Yes	VARCHAR2(75)	LINE 1: Street line 1 of address data for transaction.
GORCCAU_STREET_LINE2	Yes	VARCHAR2(75)	LINE 2:Street line 2 of address data for transaction.
GORCCAU_STAT_CODE	Yes	VARCHAR2(3)	STATE: State/Province of address data for transaction.
GORCCAU_ZIP	Yes	VARCHAR2(30)	ZIP: Zip/Postal Code of address data for transaction.
GORCCAU_NATN_CODE	Yes	VARCHAR2(5)	NATION: Nation code of address data for transaction.
GORCCAU_MERCHANT_ID	Yes	VARCHAR2(20)	MERCHANT ID: Third party trans code for transaction.
GORCCAU_DEBUG_MSG	Yes	VARCHAR2(100)	DEBUG MESSAGE: Debug message for troubleshooting process.
GORCCAU_STATUS	Yes	VARCHAR2(40)	STATUS: Status of transaction.
GORCCAU_STREET_LINE3	Yes	VARCHAR2(75)	LINE 3:Street line 3 of address data for transaction.
GORCCAU_STREET_LINE4	Yes	VARCHAR2(75)	LINE 4:Street line 4 of address data for transaction.
GORCCAU_PAY_TRANS_ID	Yes	NUMBER	TRANSACTION ID: The unique transaction ID associated with the external vendor payment.
GORCCAU_VENDOR_STATUS	Yes	VARCHAR2(40)	VENDOR STATUS: The last status associated with the transaction from a payment vendor perspective.
GORCCAU_BANNER_STATUS	Yes	VARCHAR2(40)	BANNER STATUS: The last status associated with the transaction from a Banner perspective.
GORCCAU_VENDOR_ERROR_MSG	Yes	VARCHAR2(4000)	VENDOR ERROR MESSAGE: The error message from the payment vendor, if applicable, related to a failed transaction.
GORCCAU_APPLICATION_DATA	Yes	VARCHAR2(4000)	APPLICATION DATA: Any extra information that may be associated with this specific transaction.
GORCCAU_UPDATE_FUNCTION	Yes	VARCHAR2(100)	UPDATE FUNCTION: The name of the function that will be called upon a successful pre-authorization return from the payment vendor.
GORCCAU_SUCCESS_URL	Yes	VARCHAR2(1020)	SUCCESS URL: The URL that will be used upon a successful transaction.
GORCCAU_FAILURE_URL	Yes	VARCHAR2(1020)	FAILURE URL: The URL that will be used upon an unsuccessful transaction.

Column Name	Nullable	Type	Comments
GORCCAU_VENDOR_AUTH_CODE	Yes	VARCHAR2(50)	VENDOR AUTHORIZATION CODE: The authorization code returned from the payment vendor that can be tied back to the payment transaction.
GORCCAU_LAST_NAME	Yes	VARCHAR2(60)	LAST NAME: The last name associated with this transaction.
GORCCAU_FIRST_NAME	Yes	VARCHAR2(60)	FIRST NAME: The first name associated with this transaction.
GORCCAU_MIDDLE_NAME	Yes	VARCHAR2(60)	MIDDLE NAME: The middle name associated with this transaction.
GORCCAU_SURNAME_PREFIX	Yes	VARCHAR2(60)	SURNAME PREFIX: The surname prefix associated with this transaction.
GORCCAU_NAME_PREFIX	Yes	VARCHAR2(20)	NAME PREFIX: The name prefix associated with this transaction.
GORCCAU_NAME_SUFFIX	Yes	VARCHAR2(20)	NAME SUFFIX: The name suffix associated with this transaction.
GORCCAU_SUB_CODE	Yes	VARCHAR2(20)	SUB CODE: Merchant ID based upon calling application.
GORCCAU_PROC_CODE	Yes	VARCHAR2(30)	PROCESS NAME CODE: Process Name Code associated with the transaction.
GORCCAU_SYSI_CODE	Yes	VARCHAR2(2)	SYSTEM CODE: System Code associated with the transaction.
GORCCAU_APPL_NO	Yes	NUMBER(3)	APPLICATION NUMBER: Person Application Sequence Number.
GORCCAU_CITY	Yes	VARCHAR2(50)	CITY: City of address data for transaction.
GORCCAU_HOUSE_NUMBER	Yes	VARCHAR2(10)	HOUSE NUMBER: House number of address data for transaction.
GORCCAU_AIDM	Yes	NUMBER(8)	AIDM: AIDM of user requesting credit card transaction.
GORCCAU_GIFT_NO	Yes	VARCHAR2(7)	GIFT NUMBER: Advancement gift number associated with the transaction.
GORCCAU_VENDOR_REFER_NO	Yes	VARCHAR2(30)	VENDOR REFERENCE NUMBER: The reference number returned from the payment vendor that can be tied back to the payment transaction.
GORCCAU_AMOUNT	Yes	NUMBER(12,2)	AMOUNT: The amount of the transaction.
GORCCAU_CART_ID	Yes	VARCHAR2(60)	SHOPPING CART ID: The shopping cart ID associated with the external vendor payment.

Credit Card Type Validation Table (GTVCCRD)

The following column has been added for use with external vendor payment processes:

```
GTVCCRD_EXTERNAL_MERCHANT_ID                    VARCHAR2(30)
```

The comment on column is: MERCHANT ID: Third party trans code for transaction.

New Packages

The following packages have been added for this enhancement.

GOKPVEN/GOKPVE1

This package has been added for this enhancement. It is used for payment vendor processing and contains the following procedures and functions.

- `p_redirecturl`

  Calls Web Tailor function if Web Tailor is installed.

- `p_payment_url_return`

  Verifies transaction ID, updates vendor status (GORCCAU), performs redirect to application-specific success or failure URL.

- `f_process_payment_updates`

  Called from `f_add_transaction`, which calls the functions needed to update the Banner tables.

- `f_fetchwtparam`

  Calls Web Tailor function if Web Tailor is installed.

- `f_encode`

  Calls Web Tailor function if Web Tailor is installed.

- `f_get_info`

  Reads Web Tailor TWGRINFO table and retrieves information text for the transaction description.

- `f_add_transaction`

  Called by the Web Service (via a request from the payment vendor) to call the update process (`f_process_payment_updates`).

- `f_vendor_failure`

  Called by the Web Service (via a request from the payment vendor) when there is a failure from the vendor.

- `f_process_payment_updates`

  Called by the `f_add_transaction` function, which calls the application-specific update procedure.

- `f_collect_payment_info`

  Calls payment vendor and is called by application processes.

The `f_collect_payment_info` function is used to call the payment vendor. If a non-zero status is returned from the call to the `f_collect_payment_info` function, a message is displayed indicating the vendor could not be called and to try the transaction later.

The package will send the `PROC_CODE` as part of the call to the `f_collect_payment_info` function for the merchant ID to the payment vendor.

The `f_collect_payment_info` routine will perform the following tasks:

1. Use the passed address or retrieve an address based upon the prior rules in effect (such as the *WEBCCADDR* rule on GTVSDAX).
2. Retrieve the current name passed in if a valid PIDM is supplied and `LAST_NAME` is *Null*.
3. Verify that either the AIDM or PIDM is *Not Null*.
4. Validate that the amount passed in is a valid amount.
5. Use the transaction ID passed in unless it is *Null*, in which case the value is retrieved from `BAN_PAYMENT_SEQUENCE`.
6. Update GORCCAU with the information passed in using the new `gb_payment` API.
7. Perform a redirect to the payment vendor where the user will enter the credit card information.

The `f_process_payment_updates` function performs the Banner updates. The `f_process_payment_updates` routine will perform the following:

1. Verify that the transaction ID passed in is valid.
2. Verify that the vendor amount and the Banner amounts are the same.
3. Verify that the vendor pre-authorization is valid and then update GORCCAU using the `gb_payment` API.
4. Call the application-specific `update_function`.
5. Verify that the result of the `update_function` is *0* and then return a success update status, otherwise return a failure update status.

The p_payment_url_return routine will do the following:

1. Verify that the transaction ID passed in is valid.
2. Update GORCCAU with the vendor status.
3. Perform a redirect to the application-specific success or failure URL.

GOKPURL

This package has been added for this enhancement. It redirects payment success or failure processes back to Self-Service Banner. It contains the following procedures.

- p_payment_cancel_return

  Called by the vendor after a user has intentionally cancelled the transaction.

- p_payment_failure_return

  Called by the vendor after a failed process.

- p_payment_success_return

  Called by the vendor after a successful process.

Changed Packages

The following packages have been modified for this enhancement.

BWGKCCRD/BWCKCCR1

The p_validation_display, p_collect_credit_card_info, and p_verify_credit_card_info procedures have been removed.

The p_acknowledgement_page procedure has been modified to no longer use credit card number and expiration date information.

GOKFUNC/GOKFUN1

The following obsolete procedures have been removed from the package and package body:

- p_insert_gorccau
- p_format_credit_card_data
- p_validate_cc_expdate

The following obsolete functions have been removed from the package and package body:

- f_valid_cc_number
- f_mask_credit_card_number
- f_format_cc_expdate

The p_validate_cc_amount procedure has been updated for numeric checking, as the GORCCAU_AMOUNT column is now a numeric type instead of a VARCHAR2 type.

Obsolete Packages

The following packages are no longer used:

- BWGKCPAY/BWGKCPA1 - Script to support credit card payments through Self-Service
- BWGKEPAY/BWGKEPA1 - EPOS payment server
- BWGKIPAY/BWGKIPA1 - Oracle payment server
- BWGKJPAY/BWGKJPA1 - Java process
- BWGKTPAY/BWGKTPA1 - TouchNet payment gateway

New API

The following API is new for this enhancement.

Payment API (gb_payment)

This package provides the Common Business interface for the Payment API (gb_payment). This API is used to standardize the GORCCAU CRUD functions.

Table	Objects	API Object Name	API Entity Name	Task Performed
GORCCAU	GOICCAU	gb_payment	PAYMENT	Used to process payment card transactions in Self-Service

The following packages are in this API:

- gokb_payment0.sql
- gokb_payment1.sql
- gokb_payment_r0.sql
- gokb_payment_r1.sql
- gokb_payment_s0.sql
- gokb_payment_s1.sql

- gokb_payment_build.sql
- gokb_payment_message.sql
- gokb_gorccau0.sql
- gokb_gorccau1.sql
- gos_payment_seq.sql

New Scripts

The following scripts are delivered with this enhancement.

Script	Result
gtvccrd_639EOR_01.sql	Adds column
gtvccrd_639EOR_02.sql	Adds comment on column
ggtvsdaxi_639EOR.sql	Updates rules, removes obsolete rules
gorccau_639EQC_01.sql	Adds 8.X only columns to GORCCAU
gorccau_639EQC_02.sql	Adds comments on column for 8.X only columns
gorccau_639EOR_01.sql	Adds new columns
gorccau_639EOR_02.sql	Drops obsolete columns
gorccau_639EOR_03.sql	Adds comments on column
gorccau_639EOR_04.sql	• Updates transaction ID of existing transactions so the GORCCAU_KEY_INDEX3 will be unique • Sets GORCCAU_PAY_TRANS_ID to BAN_PAYMENT_SEQUENCE.nextval
gorccau_639EOR_05.sql	Adds Primary Key Constraint for GORCCAU_PAY_TRANS_ID

The following scripts are used to change the GORCCAU_AMOUNT column from type VARCHAR2 to NUMBER(12,2):

gorccau_639EOR_06.sql	Renames GORCCAU_AMOUNT to OLD_AMT
gorccau_639EOR_07.sql	Adds GORCCAU_AMOUNT NUMBER(12,2))
gorccau_639EOR_08.sql	Sets GORCCAU_AMOUNT equal to OLD_AMT
gorccau_639EOR_09.sql	Drops OLD_AMT

Note

Unless otherwise noted, new scripts are run as part of the upgrade process for a release.

Obsolete Java Payment Gateway Objects

The following objects are no longer used:

- creditcard.properties
- paymentclient.properties
- sctpaymentclient.jar
- scttransactor.jar
- paymentclient.bat
- paymentclient.shl

3 Advanced Queuing - Technical

Overview

Banner General 8.3 introduces an optional alternative to DBMS_PIPE for session communications. This alternative communication mechanism uses Oracle Advanced Queuing (AQ). These changes will be of interest to Banner customers with database environments that use Oracle's Real Application Clusters (RAC) technology.

Background

Before this release, a majority of Banner session communication was done exclusively with DBMS_PIPE. DBMS_PIPE communication between database sessions is supported only when those sessions are on the same database instance. When an Oracle database is set up using RAC, the database runs across multiple instances, which means that DBMS_PIPE communication can fail. The failure occurs when the RAC database session initiating the communication is on a different instance from the database session receiving the communication.

Advanced Queuing Option

Banner General 8.3 is introducing the option of using Advanced Queuing (AQ) instead of DBMS_PIPE for session communication. You can now selectively enable AQ for four application areas:

- Data extract
- Review output
- Job submission
- Luminis Single sign-on (SSO) into Internet Native Banner or into Self Service Banner

GTVSDAX rules exist for each of these four application areas. There is one GTVSDAX rule (or row) for application areas, Data Extract, Review output, and Luminis SSO. There are two GTVSDAX rules (or two rows) for the Job submission application area. You can use AQ instead of DBMS_PIPE for any of these application areas by making the appropriate

settings in the GTVSDAX form. For each GTVSDAX row that exists for the application area, set the rule's external code to a value of *Y*. The GTVSDAX rule rows are all those where the column GTVSDAX_INTERNAL_CODE has a value of *AQ4PIPES*.

```
WHERE GTVSDAX_INTERNAL_CODE = 'AQ4PIPES'
```

The General 8.3 AQ modifications are being made available as an optional communication mechanism to that of DBMS_PIPE, but it is not required. The GTVSDAX rules are being delivered with a value of *N*, with Banner General continuing to use DBMS_PIPE just as it did before this release.

Note

The changes delivered in this release affect Banner General source objects only. There are other Banner products that continue to rely on DBMS_PIPE for certain process-level session communication.

Encryption for AQ

The alternative communication mechanism that uses Advanced Queuing works by briefly storing message fragments in a database table (the queue). Because certain message fragments will include sensitive data, these sensitive message fragments are encrypted when they are stored and decrypted when they are retrieved. This is the case with certain message fragments central to the Job submission application area.

The encryption method for the AQ-based alternative communication mechanism relies one of four hexadecimal keys, which are managed through the GSASECR form. Randomly-generated default values for these keys are delivered with this release.

Warning

If you plan to use the AQ based alternative communication mechanism, you should change the encryption keys after installing this release. Using the AQ-based alternative communication mechanism with the default keys cannot be considered secure.

New Forms

Single Consumer Queue Purge Form (GUAQPRG)

At institutions using Advanced Queuing, a queue administrator can use this form to purge queues when necessary.

To purge a queue, select the record for that queue and then perform a COMMIT operation. After prompting you to confirm the action, the system will delete the items in the queue.

Field	Description
Queue Name	The short name of the queue.
Description	The Banner application area that the queue serves: Data Extract, Review Output, Job Submission, or Luminis Single Sign On. **Note:** There are two queues needed to support the Job Submission application area. To understand the roles of each queue, consider this example of processing flow for Job Submission. First, an Internet-Native Banner session submits a job for processing. A message is enqueued to queue `GURJOBS_Q` (each message carries a unique token). The GURJOBS process listens to the queue `GURJOBS_Q`, dequeuing and processing messages. The final message processing by the GURJOBS process is to enqueue a return message to the queue `GURJOBS_RTN_Q` (with corresponding unique token). The internet native Banner listens to the queue `GURJOBS_RTN_Q` conditionally dequeuing (for unique token) and processing the return message.
Queue Count	The number of items (messages) currently in the queue

Changed Forms

Security Maintenance Form (GSASECR)

In GSASECR's Institution Profile window, four fields were added to maintain the encryption keys for the AQ-based alternative communication mechanism.

The AQ encryption keys are 32-character hexadecimal values. When changing the encryption keys, be sure to use valid hexadecimal characters (the digits 0-9 and the letters A-F).

Field	Description
Key1 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key2 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key3 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.
Key4 Hex Value	A 32-character hexadecimal key used with encryption of Advanced Queuing messages.

GUQINTF

The GUQINTF form works in the background to support job submission for jobs submitted from application forms. This form was changed to support AQ processing as an alternative to DBMS_PIPE for job submission.

Changed Processes

GURJOBS

This process, which runs background jobs for Banner, was changed to support AQ processing as an alternative to DBMS_PIPE.

Changed Tables

Institutional Preference Table (GUBIPRF)

This table has four new columns that store the hexadecimal encryption keys for AQ. Only one of these 32-byte HEXVALUE keys has actually been implemented for the 256-bit data encryption and decryption procedures with General 8.3. The three additional, similar columns were created to allow for potential future implementation.

Column	Data Type	Comment
GUBIPRF_EKEY1_HEXVALUE	VARCHAR2(32)	Institution KEY Value *1* may be used with g_msg_fragments encrypt/ decrypt logic
GUBIPRF_EKEY2_HEXVALUE	VARCHAR2(32)	Institution KEY Value *2* may be used with g_msg_fragments encrypt/ decrypt logic
GUBIPRF_EKEY3_HEXVALUE	VARCHAR2(32)	Institution KEY Value *3* may be used with g_msg_fragments encrypt/ decrypt logic
GUBIPRF_EKEY4_HEXVALUE	VARCHAR2(32)	Institution KEY Value *4* may be used with g_msg_fragments encrypt/ decrypt logic

Changed Libraries

GOQRPLS

This library, which provides common functions for Banner forms, was changed to support AQ processing as an alternative to DBMS_PIPE. Support for publishing to single consumer queue(s) with a specific payload datatype of g_msg_fragments (message fragments). The specification and body G$_HANDSHAKE_AQ was established for enqueue from form and dequeue to form.

New Database objects

The following were added to support AQ processing as an alternative to DBMS_PIPE:

- G_MSG_FRAGMENTS - Object Type Declaration for message fragments for AQ. MF_MISC_01 is valued with the unique token, which can then be used for conditional dequeue operations. MF_NN individual message fragments.

Name	Null?	Type
MF_MISC_01		VARCHAR2(512 CHAR)
MF_01		SYS.ANYDATA
MF_02		SYS.ANYDATA
MF_03		SYS.ANYDATA
MF_04		SYS.ANYDATA
MF_05		SYS.ANYDATA
MF_06		SYS.ANYDATA
MF_07		SYS.ANYDATA
MF_08		SYS.ANYDATA
MF_09		SYS.ANYDATA
MF_10		SYS.ANYDATA

- GSPCRPU - Decryption package for AQ alternative communication mechanism to that of DBMS_PIPE. Supports publishing and consuming to single consumer queue(s) with a specific payload datatype of g_msg_fragments (message fragments). This package created specifically for decryption ONLY of certain

message fragments. Control access to this package will control those that can decrypt these encrypted message fragments. These routines are central within the gurjobs process just after dequeuing from GURJOBS_Q.

Changed Database objects

The following packages were changed to support AQ processing as an alternative to DBMS_PIPE:

- gokb_advq_util0.sql/gokb_advq_util1.sql - Oracle AQ utility package
- GOKCSSO - package supporting single sign-on
- GOKOUTP - package supporting report output
- GOKOUTD - package supporting data extract package
- GOKSSSO - package supporting single sign-on
- GSPCRPT - package supporting encryption

New GTVSDAX Rules

Five new GTVSDAX rules (or rows) allow you to turn Advanced Queuing on and off for four different application areas. In each case, setting the external code to a value of *Y* enables Advanced Queuing for that area, while a value of *N* (the default) disables Advanced Queuing in favor of DBMS_PIPE.

Note

The Job Submission and Jobs Submission Return rows are either both set to *N* or are both set to *Y*. If either of these rows is not *Y* then Job Submission will assume to use DBMS_PIPE.

Code Description	Internal Group	Internal Code	External Code
Job Submission	GURJOBS	AQ4PIPES	*Y* to enable AQ, *N* to disable AQ
Job Submission Return	GURJOBS_RTN	AQ4PIPES	*Y* to enable AQ, *N* to disable AQ
Luminis Single Sign On	SSO	AQ4PIPES	*Y* to enable AQ, *N* to disable AQ
Review Output	GOKOUTP	AQ4PIPES	*Y* to enable AQ, *N* to disable AQ
Data Extract	GOKOUTD	AQ4PIPES	*Y* to enable AQ, *N* to disable AQ

New Scripts

Script to create administrative queues

The `gqueqtabc_080300_01.sql` script, which is run during the 8.3 upgrade, establishes the administrative queues and queue tables for Advanced Queuing.

GUBIPRF scripts

- The `gubiprf_080300_01.sql` script adds new columns for the four AQ encryption keys.
- The `gubiprf_080300_02.sql` script defines column comments for the new columns.
- The `gubiprf_080300_03.sql` script adds seed data values for the four AQ encryption keys.

Note

After this release is installed, these initial values should be changed in the GSASECR form to unique, secret values for your institution. These values, after changing, should remain relatively static for an instance. Specifically, should you have encrypted data sitting in the `GURJOBS_Q`, the same KEY value used to encrypt these messages is needed to decrypt the message properly.

GTVSDAX script

The `gtvsdaxi_080300_01.sql` script inserts seed data for the new GTVSDAX rules that control the use of AQ. These are delivered with External Code of *N*.

Changed Scripts

Script to stop Job Submission

The gurstop.sql script, which stops job submission, was changed to support AQ processing as an alternative to `DBMS_PIPE`. Logic control is based upon the GTVSDAX rows that control Job Submission. If the External Code on these rows is *N*, `gurstop.sql` will use the `DBMS_PIPE` to stop job submission. If the External Code on these rows is `Y`, `gurstop.sql` will use the queues to stop job submission.

4 Problem Resolutions

This section lists a summary of the problem resolutions included in Release 8.3. For details about a problem resolution, refer to the general80300resolutions.txt file that accompanies this release.

Process/Object	Defect Number	Summary
*IDEN	1-4UKBR1	Banner 8 *IDEN
GJAPCTL	1-7EJZSF	GJAPCTL 8.2.0.1 no longer functions correctly with Dynamic Parameter auto-fill for dates
gjapctl.fmb	CMS-DFCT102635	Problem with mouse double click in printer field when going from FAABATC to GJAPCTL.
glbdata.pco	1-47D0L1	Access violation encountered in GLBDATA
GLBLSEL	CMS-DFCT58950	GLBLSEL Process throws an SQL error when the total length of combined letter variables added in a letter has exceeded the 1500 Characters.
goadisc.fmb	1-69ZVWZ	Typo in description hint text on the GOADISC form
gokb_address1.sql	1-67EC5O	gokb_address1.sql - local user exit - p_street_line4 is missing comma
gokb_emerg_cont1.sql	1-63ARCC	gb_emergency_contact - local user exit doesn't have rowid (gokb_emerg_cont1.sql)
gokb_ident1.sql	1-67IE33	gokb_ident1.sql - local user exit - p_data_origin is missing a comma
gokb_third_party_a_r1.sql	1-7QR9W7	SSB Pin Change or API call to gb_third_party_access_rules.p_validate_pinrules incorrectly displays
goktpt1.sql,goktpty.sql	1-7CL25E	External ID generation stripping off special characters in Banner 8
goqrpls.pll	1-143HWB	Navigational problem on IDEN forms when entering a zip that is tied to several cities on GTVZIPC
goqrpls.pll	1-1V1USZ	Error using CTRL+TAB in forms called through quickflow directly from GUAGMNU
goqrpls.pll, gtvsdaxi_080300.sql	1-5J85OF	Extended Search failing with FRM-91021 error.

Process/Object	Defect Number	Summary
goradms.fmb	1-H7WLU	Minor UI issues
GUAGMNU.fmb	1-3CXWU9	'Exit Banner' from File -> Menu does not work correctly
GUAOPTM	1-8J508S	Form GUAOPTM error FRM-40831: Truncation occurred: value too long for field FORM_DESC
guastdf.cpp	1-49R5LP	Corrupted data from leftpad function
guainst.fmb	1-82PYHP	There is no version populating in the Installation Information block for the general product.
guqwutl	1-3Y804A	An unhandled exception occurred if local directory is invalid on guauprf
see comments	1-639EQC	PCI Compliance
twbkwbis.P_Logout	1-8R6NO9	Sign Out usability issue - button displays Sign Out after the user has logged out
find a page	1-8RSQAI	Using the Find a page function renders an unreadable list.
bwgkoad1.sql	1-2YFWSI	Self-Service allows only a space character to be entered for the City field
twbkwbis	1-3A9OPF	There is an issue with VBS and self service - after exiting and trying to re-enter self-service.
bwgkopr1.sql	1-50WOYW	Directory Profile page does not display confirmation message for changes.
twbkwbis.P_ValLogin	1-5RYVP9	Incorrect error message when user has no security question defined in GOBANSR
bwpktetm.P_TimeSheetButtonsDriver	1-8X7AGH	Left/Right Scroll bar needed to display Timesheet Preview
DesktopTools.xla	CMS-DFCT58238	Desktop Tools requires Visual Basic 6.0 which is no longer supported by Microsoft
bannerui.jar	1-5T7PU5	`bannerui.jar` - Hardcoded Java font “verdana” causing an issue in Arabic option list of INB session and Alignment of text was not correctly displayed in a Arabic INB session

Process/Object	Defect Number	Summary
	1-8K2MM8	Finance and Finance>Encumbrance Query
	1-8YF0YF	IE7 and Firefox- Click on Finance Menu - menu items do not appear-Systest both URLS