ellucian™

Banner Accounts Receivable
TRM Supplement

Release 8.3
April 2011

Banner®, Colleague®, PowerCAMPUS®, Luminis® and Datatel® are trademarks of Ellucian or its affiliates and are registered in the U.S. and other countries. Ellucian, Advance, DegreeWorks, fsaATLAS, Course Signals, SmartCall, Recruiter, MOX, ILP, and WCMS are trademarks of Ellucian or its affiliates. Other names may be trademarks of their respective owners.

©2000-2011 Ellucian. All rights reserved. The unauthorized possession, use, reproduction, distribution, display or disclosure of this material or the information contained herein is prohibited.

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

In preparing and providing this publication, Ellucian is not rendering legal, accounting, or other similar professional services. Ellucian makes no claims that an institution's use of this publication or the software for which it is provided will 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: Ellucian
4375 Fair Lakes Court
Fairfax, Virginia 22033
United States of America

Revision History

Publication Date	Summary
April 2011	New version that supports Banner Accounts Receivable 8.3 software.

Banner Accounts Receivable 8.3
TRM Supplement

Contents

1 Reports and Processes

The *Banner Accounts Receivable User Guide* contains descriptions and report samples of the Accounts Receivable reports in the General, Student, and Finance modules. Please refer to that manual for functional information about the use of the Accounts Receivable reports.

The following pages contain brief descriptions of reports in the Accounts Receivable module.

Report Identifier	Report Name	Report Description
TFRBILL	Invoice/Bill Process	Processes invoices by invoice number and produces a billing statement for a user-defined time frame.
TFRDETL	Account Detail Report	Shows account transaction ledger by date.
TFRLATE	Assess Penalty/Interest Charges Report	Shows assessment of late charges for outstanding Accounts Receivable accounts.
TFRRFND	Auto Refunding of Credit Amounts	Identifies accounts which have a credit balance identified, flagged and reported.
TGPBILL	Billing Purge Process	Enables you to identify which customer accounts may have their individual charge and payment transactions purged and creates Balance Forward Transactions to identify this summarization based on effective date, due date, and bill dates with user defined time frames.
TGPHOLD	Auto Hold Release Process	Based on the parameters and rules form, automatically adds and releases holds on a student account.
TGRAGES	Aging Analysis Report	Used to analyze all past-due accounts.

Report Identifier	Report Name	Report Description
TGRAPPH	Application of Payments Archive History process	Provides the ability to archive inactive application records from TBRAPPL.
TGRAPPL	Application of Payment Report	Applies payments to charges for accounts based on priority.
TGRCDEL	Cashier Delete Process	Deletes the cashiering records for which all corresponding detail records have been fed to accounting.
TGRCLOS	Cashier Session Close Process	Automates the closing and/or finalizing of cashier sessions.
TGRCOLC	Collection Agency Report	Reviews collection agency performance.
TGRCSHR	Cashier Delete Report	Shows daily bank deposit and user audit.
TGRDELI	Batch Update of Delinquencies	Identifies past due accounts and those with a delinquency code.
TGRDETC	Detail Code Report	An Oracle report that displays detail codes and their associated accounting distributions.
TGRFEED	Accounting Feed Process	Shows audit trail of all cashier transactions.
TGRMISC	Miscellaneous Receipt Report	Shows receipt printouts for miscellaneous transactions.
TGRRCON	Accounts Receivable Reconciliation Report	Enables you to specify various parameters to create a receivables reconciliation report.
TGRRCPT	Account Receipt Report	Shows receipt printout for account related transactions.
TGRUNAP	Unapplication of Payment Report	Unapplies previous application of payment before re-applying with new transaction.
TGRAGES	Aging Analysis Report	Reviews past due status of accounts.

Report Identifier	Report Name	Report Description
TRRAGES	Grant Aging Analysis Report	Provides analysis of the grant receivables. You can print this report for aging of both billed and unbilled receivables.
TRRAPPL	Grants Application of Payment Process	This process enables you apply payments to the charges. This process has no parameters. It applies payments to charges by grant. If a Tpay number or invoice paid number is entered by the user in the payments entry form (FRAAREV), this process first applies the payment to that specific charge transaction number or for that specific bill invoice number. If these two fields are not entered by the user, then the process will apply the payment against a grant to the oldest charge.
TRRCOLL	Grants Collections Report	Prints information about collection efforts that have been entered on the Grant Billing Collections Form (TRACOLL). You can run the report by status of the bill (open or closed), or by user ID, grant, or agency.
TRRRCON	Grant Reconciliation Report	This report reconciles data between the Accounts Receivable module and the Finance module for Grant Billing transactions only. You can reconcile by chart or by grant or by payment management system code. You can also print just the grants that are out of balance or just the Payment management system codes that are out of balance. This report reconciles between billed summary and billed detail, Payments in AR against payments in General ledger and billed charges in AR against charges in the general ledger.

Report Identifier	Report Name	Report Description
TRRUNAP	Unapplication of Payments Process	Enables you to unapply payments that have been incorrectly applied to a charge. After you run this process, you can apply the payment to the appropriate charge.
TRRUNPL	Unapplied Payments Listing	This report prints a list of all payments that have not been applied to charges. It also prints the fund codes the exact payment was posted to. This report is sorted by grant.
TSP1098	1098 Student Tax Notification Process	The TSP1098 process retrieves existing data from the Tax Information Form (TSATAXI), and other fields within Banner and populates the Student Tax Notification Form (TSATAXN).
TSPCPDT	Contract Payment Detail Load Process	The Contract Payment Detail Load is a batch process run from Job submission. This process provides the ability to upload a text file of payment details by student associated with a sponsor payment. The process can be run in audit mode or update mode.
TSPCSTU	Contract Student Load Process	The Contract Student Load Process (TSPCSTU) is a batch process run from Job submission. This process provides the ability to upload a text file of students and associated data to existing third party contracts. The process can be run in audit mode or update mode.
TSPISTA	Assign Installment Plans	This process is used to assign students to an installment plan.
TSPISTT	Create Installment Transaction	This process is used to create the installment plan transactions in the Accounts Receivable Transaction Detail Table (TBRACCD). Rules created on the new Installment Plan Rules Form (TSAISTR) are used for processing.

Report Identifier	Report Name	Report Description
TSR1098	1098 Student Tax Report Process	Use the TSR1098 Process to print 1098-T paper forms for IRS/ Student Notifications on Moore Business Forms or produce the IRS file.
TSRBTOT	Total Billed Report	The A/R Supervisor uses this report to review the billing amounts by detail code by term.
TSRCBIL	Billing Invoice/Student Billing Statement	Processes deposits, contracts, exemptions, and/or financial aid. You can request either invoices or statements. The A/R Supervisor uses this report on a monthly or user-defined time line.
TSRDETL	Account Detail Report	The A/R Supervisor uses this report to review the account transaction ledger by term.
TSRLATE	Assess Penalty/Interest Charges Report	The A/R Supervisor uses this report to assess penalty and interest charges for outstanding student accounts.
TSRLBOX	Lockbox Tape Unload Process	Posts payments received through electronic funds transfer to a person's account.
TSRPDRV	Lockbox Result Print Program	Prints the results of the action performed by the Lockbox Tape Unload Process (TSRLBOX).
TSRRFND	Auto Refunding of Credit Amounts	Identifies and flags those accounts which have a credit balance with a refund detail code. In addition, it lists the generated refund transactions with an address.
TSRROLL	Contract and Exemption Roll Process	Allows contracts and/or exemptions and associated students to roll from one term to the next term.
TSRSSUM	Student Transaction Summary Report	Displays a student's balance and individual charges and payments.

Report Identifier	Report Name	Report Description
TSRTBIL	Third Party Billing Statement	Processes third party charges and payments in either invoice or statement mode. The A/R supervisor uses this report on a monthly or user-defined time frame.
TSRTRAF	1098 Student Detail Tax Report	Use the TSRTRAF process to produce a flat file of Tax Notification Data to send to third party servicers and/or to print the detail for all tax report codes created for the given tax year. This detail may be included in Student Notification mailings.
TSRTSUM	Transaction Summary Report	Displays billed and unbilled transactions by date range in summary or detail mode for individual students or groups of students.
TVPREQA	Create Title IV Authorizations from Requirements	Use this process to create authorizations from financial aid tracking requirements for students with statuses that have been updated since a date that you specify. You can use this process on an as-needed basis to keep track of students as they complete their financial aid requirements.
TVRCRED	Students with Title IV Credit Report	Use the TVRCRED process to produce a report that displays Title IV students who have a credit account balance. Run the report after Disbursements (RPEDISB) and Application of Payments (TGRAPPL). Students who have authorized the institution to hold Title IV funds are shown separately on this report. You may contact other students with a Title IV credit balance regarding authorization to apply Title IV aid to other amounts due or initiate a refund for them.

Reports and Processes Matrix

Legend

Report or Process	The report/batch process name
Language	Identifies the language for the process - COBOL, C
Update/Query	Does the process update any tables, or is it strictly a query-only report?
Audit	Can you run the update process in Audit Mode, so that you can produce the report without an update taking place?
Job Submission	Can you run the process via job submission?
Sleep/Wake	Is the process used in conjunction with Sleep/Wake?
Off Peak	Does SunGard Higher Education recommend that you defer this program to an off peak processing time (late night, weekends) for performance reasons?
Restart	If the process aborts or is terminated after the process is initiated, can you restart the process without any adverse consequences?

Reports and Processes Attributes

Report or Process	Language	Update/ Query	Audit	Job Submission	Sleep/ Wake	Off Peak	Restart
TFRBILL	C	Update		Yes		Yes	Yes
TFRDETL	C	Query		Yes			
TFRLATE	C	Update	Yes	Yes			Yes
TFRRFND	C	Update	Yes	Yes			Yes
TGPBILL	C	Query		Yes			Yes
TGPHOLD	C	Update	Yes	Yes	Yes	Yes	Yes
TGRAGES	C	Query		Yes			
TGRAPPH	C	Update	Yes	Yes		Yes	Yes
TGRAPPL	C	Update		Yes		Yes	Yes
TGRCDEL	C	Update		Yes		Yes	Yes
TGRCLOS	C	Update	Yes	Yes			
TGRCOLC	C	Query		Yes			Yes
TGRCSHR	C	Query					Yes
TGRDELI	C	Update	Yes				Yes
TGRDETC	Oracle	Query					Yes
TGRFEED	C	Update		Yes		Yes	
TGRMISC	C	Query		Yes	Yes		
TGRRCON	C	Query		Yes			Yes
TGRRCPT	C	Query		Yes	Yes		Yes
TGRUNAP	C	Update		Yes			Yes
TRRAGES	C	Query		Yes			Yes

Report or Process	Language	Update/ Query	Audit	Job Submission	Sleep/ Wake	Off Peak	Restart
TRRAPPL	C	Update		Yes			Yes
TRRCOLL	C	Query		Yes			Yes
TRRRCON	C	Query		Yes			Yes
TRRUNAP	C	Update		Yes			Yes
TRRUNPL	C	Query		Yes			Yes
TSP1098	C	Update	Yes	Yes		Yes	
TSPISTA	C	Update		Yes			Yes
TSPISTT	C	Update		Yes		Yes	Yes
TSR1098	C	Update	Yes	Yes		Yes	
TSPCPDT	C	Update	Yes	Yes			Yes
TSPCSTU	C	Update	Yes	Yes			Yes
TSRBTOT	C	Query		Yes			
TSRCBIL	C	Update	Yes	Yes	Yes	Yes	
TSRDETL	C	Query		Yes		Yes	
TSRLATE	C	Update	Yes	Yes			
TSRLBOX	COBOL	Update		Yes			
TSRPDRV	COBOL	N/A		Yes			
TSRRFND	C	Update	Yes	Yes			
TSRROLL	C	Update	Yes	Yes			
TSRSSUM	C	Query		Yes	Yes		
TSRTBIL	C	Update	Yes	Yes		Yes	
TSRTRAF	C	Update	Yes	Yes			
TSRTSUM	C	Query		Yes			
TVPREQA	C	Update		Yes			
TVRCRED	C	Query		Yes			Yes

For a more detailed description of each report and its parameters, generate a List of Reports and Parameters (GJRRPTS). This report also includes batch processes.

Sleep/Wake-up

Banner provides two different mechanisms for running jobs in a cyclical or *sleep/wake-up* manner.

Setting Up Sleep/Wake Processes Using Method One

The first method uses OS command scripts and an SQL*Plus script to cause the job to run in a cyclical fashion. These jobs must be submitted from the operating system prompt and

must be terminated manually. To compile programs to run in this fashion, you must define the `NO_SLEEP_SW` as a pre-compiler directive to exclude the code used by the second technique.

UNIX

The first command procedure, `sleepunx`, prompts for parameters needed by the second procedure and SQL*Plus scripts, `sleepunx.shl` and `sleepunx.sql` respectively. This procedure then starts (or submits) `sleepunx.shl`, which in turn starts `sleepunx.sql`. The SQL*Plus script `sleepunx.sql` will spool OS-specific commands to run the job into a file, provided there is actually work to do as determined by the parameters previously entered. When the SQL*Plus script exits, `sleepunx.shl` executes the *spool* file. The parameters needed by the program are contained in a `XXXXXXX.dat` file which are read via input redirection when the job executes. The second command procedure `sleepunx.shl` then *sleeps* for the specified interval, awakes, and loops back to start the SQL*Plus script again.

VMS

This is essentially the same as for UNIX. The script names are `sleep.com`, `sleepdec.com`, and `sleepdec.sql`. Command input redirection is accomplished by defining `sys$input` as the `.dat` file. The *sleeping* is done with the *wait* command.

CMS

In this environment, there is only one command script, `sleepcms.exec`. Each job run in sleep/wake mode must have its own `.exec` script. The command script `sleepcms.exec` is provided as the model. You also need to set up separate CMS accounts for each job or you may combine the execs to run sequentially on a single account. The `.dat` file is read via input redirection. *Sleeping* is performed using the CP SLEEP command.

Setting Up Sleep/Wake Processes Using Method Two

Note

The Banner systems and processes in the following chart are valid for the Sleep/Wake processing described in this section:

Report/Process	Description
TGRRCPT	Account Receipt
TSRCBIL	Student Billing Statement (Invoices)
TGRMISC	Miscellaneous Receipt
TSRSSUM	Student Transaction Summary Report
TGPHOLD	Auto Hold Release Process

1. Define printer and print command on the Printer Validation Form (GTVPRNT). In the **(Printer) Code** field, enter a name to reference each specific printer that may be used for printing output from sleep/wake processing. In the **Comment (Printer Command)** field, enter the correct operating system print command as it would normally be entered from the command line prompt, substituting an at sign (@) as the place holder for the file name to be printed.

Operating System	Print Command
UNIX example:	`lp -d talaris1 @`
VMS example:	`print/queue=ln01 @`

2. On the appropriate System Distribution Initialization Information Form (SOADEST for Student or TOADEST for Accounts Receivable), enter the printer code from GTVPRNT that should be identified with the collector table rows that will be inserted to the appropriate tables when online application forms create a request for output that can be generated by sleep/wake processing.

The collector tables are as follows:

Process	Collector Table
TGRMISC	TBRCMIS
TGRRCPT	TBRCRCP
TSRCBIL	TBRCBRQ
TSRSSUM	TBRCSUM
TGPHOLD	TBRCHLD

3. On the Process Submission Control Form (GJAPCTL), for the valid sleep/wake jobs listed previously, enter the correct response for the parameter that specifies that the job should be processed for collector table entries. Refer to the documentation for each specific process to determine the appropriate response in each case (correct responses may be *COLLECTOR*, *Y*, *%*, etc.). In addition, each sleep/wake job has a printer code parameter. You must specify exactly the same code for this parameter answer that was entered on either SOADEST or TOADEST. A value of *Y* should be

entered for the run in sleep/wake mode parameter, and a number of seconds should be specified for the sleep/wake interval (cycle) for each process.

Note

Do not enter the printer code in the top section of GJAPCTL; only enter it in the parameter section of the form.

4. The Sleep/Wake Maintenance Form (GJASWPT) should be used to stop the sleep/wake process or to change the sleep interval. A process name and printer code must be entered in the Key Information. A List of Values is available in each field to see the valid list of processes and printer codes that have ever been submitted for sleep/wake processing.

   To stop the process, enter *N* in the **Continue to Run f**ield and do a Save function. The job will not stop immediately, but rather will stop after the next time the process *wakes up* and finishes the next processing cycle. To change the sleep interval, enter the desired interval in the **Next Cycle Time** field and do a Save function.

   The GJASWPT form can also be used to view statistics regarding how many rows were processed for the most recent wakeup cycle and the total number of rows processed since the process was initiated. You can also determine if the processes terminated abnormally. by viewing the **Abnormal Termination** field. If there is a *Y* in **Abnormal Termination**, something caused the process to fail. You should review log files to determine the cause.

Operating Systems Without Sleep/Wake-up Commands

Operating systems that do not have sleep commands, or whose sleep commands may not be executed by user programs, must use the first method.

Collector File Entries

The following forms can be used to generate the collector file entries.

Form	Description/Generate	Report/ Process	Description
TGARCPT	Receipt Form		
	Can generate:	TGRRCPT	A/R Receipt
TFAAREV	Account Detail Query Form		
	Can generate:	TGRRCPT	A/R Receipt
TFADETF	Foreign Currency Detail Form		
	Can generate:	TGRRCPT	A/R Receipt

Form	Description/Generate	Report/ Process	Description
TFADETL	Account Detail Form		
	Can generate:	TGRRCPT	A/R Receipt
TFAMASS	Billing Mass Data Entry Form		
	Can generate:	TGRRCPT	A/R Receipt
TFAMDET	Billing Mass Entry Detail Form		
	Can generate:	TGRRCPT	A/R Receipt
TSAAREV	Account Detail Review Form		
	Can generate:	TGRRCPT	A/R Receipt
		TSRCBIL	Invoice
		TSRSSUM	Student Summary
TSADETF	Student Foreign Currency Detail Form		
	Can generate:	TGRRCPT	A/R Receipt
		TSRCBIL	Invoice
TSADETL	Student Account Detail Form		
	Can generate:	TGRRCPT	A/R Receipt
		TSRCBIL	Invoice
TSAMASS	Billing Mass Data Entry Form		
	Can generate:	TGRRCPT	A/R Receipt
TSASPAY	Student Payment Form		
	Can generate:	TGRRCPT	A/R Receipt
		TSRCBIL	Invoice
TSAMISC	Miscellaneous Transaction Form		
	Can generate:	TGRMISC	Miscellaneous Receipt
TFAMISC	Miscellaneous Transaction Form		
	Can generate:	TGRMISC	Miscellaneous Receipt

Running Oracle Reports with Accounts Receivable

When run through Banner, Reports is not currently able to access the default role defined for the report in the bansecr schema. Therefore, product-specific roles are created with the base privileges needed to run the reports, which must be granted to any user who will run reports for that product using Banner.

In the Accounts Receivable product *plus* sub-directory, you will find `tasorep.sql`. This script will create the necessary role and assign the privileges needed to run the reports. Please review this script before applying it to your production database.

1. Invoke SQL*Plus and run the procedure:
   ```
   sql_cmd system/password [ENTER]
   start tasorep  [ENTER]
   ```

2. Next, grant the role to the users who will be running the reports and set it as one of their default roles.
   ```
   sql_cmd system/password [ENTER]
   grant ban_arsys_rep to “your_user” [ENTER]

alter user “your_user”
default role ban_default_connect, ban_arsys_rep
   ```

Note

The alter user statement should include all of the user’s default roles. Refer to `sys.dba_role_privs` for existing roles where `default_role=`*`YES`* for `your_user` as grantee.

Additional documentation on Oracle reports is available in the *Banner General Technical Reference Manual*.

2 Interfaces

This chapter identifies Banner products that interface with Banner Accounts Receivable and tables used by these interfaces. This chapter also provides detailed information about the Lockbox Tape Unload Process (TSRLBOX).

Accounts Receivable Interfaces

The Banner Accounts Receivable system interfaces with the following Banner products:

- Banner Finance System
- Banner Student System
- Banner General System
- Banner Financial Aid

Interfaces within Banner

Table Name	Description
GURFEED Table	This table contains financial transactions from other Banner applications or client-developed applications which are to be processed into Banner Finance using the FURFEED and FGRTRNI processes.
GURAPAY Table	This table contains single line invoices from other Banner applications or client-developed applications, which are to be processed into Banner using the FURAPAY process.

Lockbox Payment Processing

The Lockbox Tape Unload Process (TSRLBOX) posts payments received through electronic funds and transferred to a person's account.

Note

The tape format must include at least one detail record and one batch summary record in order for the tape to be processed.

If the data received from the bank via the tape is in error, use the Lockbox Correction Form (TSALBOX) to correct errors. ID transaction date and invoice number paid can be modified. After corrections have been made, rerun TSRLBOX using the Lockbox Intermediate Table (TBRLBOX) as input.

Processing Details

This process unloads the Lockbox electronic funds transfer tape data to a database table (TBRLBOX) and/or attempts to post the information from the Lockbox Intermediate Table to the Account Detail Table (TBRACCD).

If the input to the process is TAPE, the Lockbox tape is read. File header, batch header, and file summary records are not loaded to the TBRLBOX table. The TBRLBOX table contains only batch detail and batch summary records. Each batch detail record read from the tape is loaded to TBRLBOX. When a batch summary record is encountered, information about the batch is loaded to an internal table (which will be printed on the control report), the record is loaded to TBRLBOX, the batch is committed to the database and the `TBRLBOX_BATCH_VERIFY_IND` is set. A batch is marked as verified (`TBRLBOX_BATCH_VERIFY_IND` = *Y*) when the computed total of the batch detail amounts is the same as the total on the batch summary record. An entire batch is either verified on unverified (that is, the indicator is set on the batch summary record and on all the batch detail records).

Once the data have been loaded to the database, or the input to the process is TABLE, the Lockbox Intermediate Table (TBRLBOX) is processed. The ID must be a valid ID on SPRIDEN. The invoice number paid (if present) must be a valid invoice number on TBRACCD for the ID. The amount to be paid against an invoice must also be valid. When it is determined that at least one payment will be posted to the Account Detail Table (TBRACCD), a cashiering session is created. For each payment to be posted to TBRACCD, a tran number will be generated, the payment will be inserted into TBRACCD, and the TBRLBOX record processed will be deleted. When all batch details are deleted from TBRLBOX, the batch summary will be deleted from TBRLBOX.

A control report will be generated.

Note

The Lock Box process must be tailored to your specific input format, including the check digit algorithm if you choose to use one. The tape layout provides for check digits for further verification of data. Check digit verification should occur when validating ID, invoice number paid, etc. Add logic to use your check digit algorithm. Technical staff should refer to the Cobol source code `TSRLBOX.pco`. ■

The program inserts a year 2000 compliant date into TBRLBOX when reading from TAPE. There is a SUBSTR function used and the format for the SUBSTR function is YYMMDD. The date being read from the tape should be in YYMMDD format. This gets inserted into `TBRLBOX_BATCH_TRANS_DATE` with appropriate century. If the format is different on the tape, then the code for the SUBSTR function needs to be changed.

Tailoring the Lock Box Process to your Institution

The Lock Box process must be tailored to your specific input format. The record length of the Lock Box input file is 80 characters. The basis of this layout is that each record contains a record type and a batch number. If this basic premise cannot be kept intact, the BANNER AR Lock Box process should not be used since this is inherent in the design. The record length may be changed, but the record must contain a record type and a batch number. The length of record type and batch number may be modified.

There are five record types:

1. File Header Record; denoted by a record type of *1*
2. Batch Header Record; denoted by a record type of *2*
3. Batch Detail Record; denoted by a record type of *3*
4. Batch Summary Record; denoted by a record type of *4*
5. File Summary Record; denoted by a record type of *5*

To modify the record types, change the value of the following 05-level variables under the 01 level `DECLARE-SPECIFIC`:

```
05  DECL-FILE-HEADER     PICTURE X(01)  VALUE '1'
05  DECL-BATCH-HEADER    PICTURE X(01)  VALUE '2'
05  DECL-BATCH-DETAIL    PICTURE X(01)  VALUE '3'
05  DECL-BATCH-SUMMARY   PICTURE X(01)  VALUE '4'
05  DECL-FILE-SUMMARY    PICTURE X(01)  VALUE '5'
```

To modify the lock box record layout, perform the following tasks:

1. Modify the LBOXFILE File Definition (FD); change the `RECORD CONTAINS` clause and the picture clause on the 01 level `LBOX-RECORD`. This is only necessary if record length is being modified.

2. Modify the 05-level and 07-level variables under the 03 level `WS-LBOX-RECORD` in the `WORKING-STORAGE SECTION`. Keep in mind that the different record types were defined using a `REDEFINES` clause. When the length of the 05 level `LBOX-DATA` (length 76) is added to the length of the 05 levels `LBOX-BATCH-NUMBER` (length 3) and `LBOX-RECORD-TYPE` (length 1), the total is 80. Therefore, any change in record length must be accounted for when redefining.

   Modify the 07-level variables under the 05-level variables (`WS-LBOX-FILE-HEADER`, `WS-LBOX-BATCH-HEADER`, `WS-LBOX-DETAIL`, `WS-LBOX-BATCH-SUMMARY`, `WS-LBOX-FILE-SUMMARY`) to define the layout of your input file. Remember that the total of the lengths of 07-level variables under the 05-level variables (`WS-LBOX-FILE-HEADER`,

WS-LBOX-BATCH-HEADER, WS-LBOX-DETAIL, WS-LBOX-BATCH-SUMMARY, WS-LBOX-FILE-SUMMARY) must be the same length as WS-LBOX-DATA.

3. The 01 level DECL-LBOX-RECORD is a combination of the record type, batch number, and the detail record (WS-LBOX-DETAIL in the WORKING-STORAGE SECTION); make the appropriate changes.

4. Keep in mind that any references, whether direct (that is, the field itself) or implied (for example, hold fields), to fields that have been modified or deleted, must be modified or removed

Once the record layout has been modified, perform the applicable tasks from the following list.

- If the file header record has been modified or dropped, modify or delete paragraph MBB-FILE-HEADER.
- If the batch header record has been modified or dropped, modify or delete paragraph MBC-BATCH-HEADER.
- If the batch detail record has been modified or dropped, modify or delete paragraph MBD-BATCH-DETAIL.
- If the batch summary record has been modified or dropped, modify or delete paragraph MBE-BATCH-SUMMARY.
- If the file summary record has been modified or dropped, modify or delete paragraph MBD-FILE-SUMMARY.
- If the total batch amount has been removed from the batch summary record, remove logic to verify the batch (that is, delete paragraphs MBG-VERIFY-BATCH and MBH-UPDATE-BATCH-IND).
- All references to any deleted paragraph should be removed.
- If check digits are being used, add paragraph(s) to further verify data using check digits. These paragraph(s) should be performed prior to inserting into the Lock Box Intermediate Table.

An internal table, used to display information about what was contained on the lock box file, is loaded in paragraph MBI-LOAD-BATCH-TABLE. The table can hold information for a maximum of 100 batches. If your input file contains more than 100 batches, modify the OCCURS clause on the 03 level BATCH-INFO and change the reference to 100 in MBI-LOAD-BATCH-TABLE.

The record layout for the extract record, which is passed to a sort program and a print program, resides in the include module TSRLBOXO. If any changes are made to the extract record, the sort program (TSRSORT) and the print program (TSRPDRV) must be modified to accommodate the changes.

Note

The program inserts a year 2000 compliant date into TBRLBOX when reading from TAPE. There is a SUBSTR function used and the format for

the SUBSTR is YYMMDD. So the date being read from the tape should be in YYMMDD format. This gets inserted into TBRLBOX_BATCH_TRANS_DATE with the appropriate century. If the format is different on the tape, then the code for the SUBSTR function needs to be changed.

If the value of record type has been changed in TSRLBOX, the Lock Box Correction Form (TSALBOX) must also be modified. Specifically:

- Modify KEY-STARTUP to assign your values to TBRLBOX_REC_TYPE.
- Modify KEY-CLRFRM to assign your values to TBRLBOX_REC_TYPE.

3 Conversion

To successfully convert your data, you should have a basic understanding of the Banner Data Element Dictionary (DED), and a report called *Tables and Views*. You should print hard copies of the General DED and the Accounts Receivable DED when SunGard Higher Education delivers each new tape.

The DED displays the following information:

Conversion Tables for AR

Column Name	Column Description
Field Name	Field name
Type	Character or date field indicator
Width	Field length
Scale	Number of digits to the right of the decimal point. For example, a scale of *2* indicates one-hundredths of a unit, as in *.99*.
Nulls	Null/Not Null Indicator
Column Comments	Description of the column, abbreviation of the validation table, and so on

On the following pages are the conversion tables you need to convert your data for Banner Accounts Receivable.

Step 1 Add IDs using the Identification Form.

Step 2 Populate the Detail Charge/Payment Code Definition Table (TBBDETC) and the Detail Code Account Definition Table (TBRACCT) using the Detail Code Control Form.

If you will be loading transactions in detail, you must build an entry for each unique detail code contained in your existing file. If you will be loading only summary information, you only need to build one or two codes. You would create two if you will identify credits with a detail code different from charges. You would build one if you use the same detail code for credits or charges.

For example: *BFWP* - Balance forward payment, *BFWC* - Balance forward charge, or *BFWD* - Balance forward.

Step 3 Load the Account Detail Table (TBRACCD) according to the information in the following chart.

Column Name	Column Description
TBRACCD_PIDM	Internal identification number as calculated by the Identification Form.
TBRACCD_TRAN_NUMBER	A one-up number, beginning with one, to identify each transaction on the account. All accounts will start with *1*.
TBRACCD_TERM_CODE	Value is *ARTERM*'.
TBRACCD_DETAIL_CODE	Valid detail code as entered on the Detail Code Control Form.
TBRACCD_USER	Userid of the cashier posting the transaction to the account.
TBRACCD_ENTRY_DATE	System date including hours, minutes, and seconds.
TBRACCD_AMOUNT	Dollar amount of transaction in base currency.
TBRACCD_BALANCE	If the amount represents a credit to the person, this will be the amount multiplied by -1. If the amount represents a charge, this must be equal to the amount.
TBRACCD_EFFECTIVE_DATE	System date excluding hours, minutes, and seconds.
TBRACCD_BILL_DATE	If you want the transactions you create to appear in detail when statements are produced, this field must be null. If you want transactions to appear in summary as a previous balance, you should include the system date excluding hours, minutes, and seconds.
TBRACCD_DUE_DATE	If you want the transactions you create to appear in detail when statements are produced, this field must be null. If you want transactions to appear in summary as a previous balance, you should include the system date excluding hours, minutes, and seconds.

Column Name	Column Description
TBRACCD_DESC	Description of the detail code as found in the Detail Charge/ Payment Code Definition Table for TBRACCD_DETAIL_CODE.
TBRACCD_RECEIPT_NUMBER	Null
TBRACCD_TRAN_NUMBER_ PAID	Null
TBRACCD_CROSSREF_PIDM	Null
TBRACCD_CROSSREF_ NUMBER	Null
TBRACCD_CROSSREF_ DETAIL_CODE	Null
TBRACCD_SRCE_CODE	Must be *T*.
TBRACCD_ACCT_FEED_IND	Refer to "Accounting Feed Options" on page 3-4.
TBRACCD_ACTIVITY_DATE	System date including hours, minutes, and seconds.
TBRACCD_SESSION_NUMBER	Refer to "Accounting Feed Options" on page 3-4.
TBRACCD_CSHR_END_DATE	Refer to "Accounting Feed Options" on page 3-4.
TBRACCD_CRN	Null
TBRACCD_CROSSREF_SRCE_ CODE	Null
TBRACCD_LOC_MDT	Null
TBRACCD_LOC_MDT_SEQ	Null
TBRACCD_RATE	Null
TBRACCD_UNITS	Null
TBRACCD_DOCUMENT_ NUMBER	Null
TBRACCD_TRANS_DATE	System date excluding hours, minutes, and seconds
TBRACCD_PAYMENT_ID	Null
TBRACCD_INVOICE_NUMBER	Null

Column Name	Column Description
TBRACCD_STATEMENT_DATE	If you want the transactions you create to appear in detail when statements are produced, this field must be null. If you want transactions to appear in summary as a previous balance, you should include the system date excluding hours, minutes, and seconds.
TBRACCD_INV_NUMBER_PAID	Null
TBRACCD_CURR_CODE	Null
TBRACCD_EXCHANGE_DIFF	Null
TBRACCD_FOREIGN_AMOUNT	Null
TBRACCD_LATE_DCAT_CODE	Null
TBRACCD_FEED_DATE	Null
TBRACCD_FEED_DOC_CODE	Null

Accounting Feed Options

If you do not process these transactions through the Accounting Feed Process (TGRFEED), supply the following values for the following columns:

Column Name	Column Description
TBRACCD_ENTRY_DATE	System date including hours, minutes, and seconds.
TBRACCD_ACCT_FEED_IND	Must be *F*.
TBRACCD_SESSION_NUMBER	001
TBRACCD_CSHR_END_DATE	Same value as TBRACCD_ENTRY_DATE

If you run these transactions through the Accounting Feed Process (TGRFEED), supply the following values for these columns:

Column Name	Column Description
TBRACCD_ENTRY_DATE	System date including hours, minutes, and seconds
TBRACCD_ACCT_FEED_IND	Must be *Y*.
TBRACCD_SESSION_NUMBER	001
TBRACCD_CSHR_END_DATE	Same value as TBRACCD_ENTRY_DATE

All transactions you create during a given conversion run must have the same value for the TBRACCD_ENTRY_DATE and TBRACCD_CSHR_END_DATE. You also need to create an entry in the Cashier Session Table with the following values:

Column Name	Column Description
TBBCSHR_USER	Same value as TBRACCD_USER
TBBCSHR_SESSION_NUMBER	Same value as TBRACCD_SESSION_NUMBER
TBBCSHR_START_DATE	Same value as TBRACCD_CSHR_END_DATE
TBBCSHR_END_DATE	Same value as TBRACCD_CSHR_END_DATE
TBBCSHR_SESSION_IND	Must be *R* or *F*'
TBBCSHR_ACTIVITY_DATE	System date including hours, minutes, and seconds
TBBCSHR_BANK_DEPOSIT_NUMBER	Null
TBBCSHR_BANK_CODE	Null

Run the Cashiering Report (TGRCSHR) to verify your work. Once satisfied, proceed by running the Accounting Feed Process (TGRFEED).

Title IV Setup Considerations

Three data elements in the Detail Code Table (TBBDETC) process Title IV credits and returns. These elements must be populated to use the features related to handling Title IV credits in application of payments and refunding.

In addition, one data element in the Accounts Receivable Detail Table (TBRACCD) can be used in calculating original charges for Title IV returns. This data element needs to be updated for any term(s) for which calculations are likely to be made.

4 APIs

Application Programming Interfaces (APIs) enhance processing and simplify code. This chapter contains information about the APIs and supporting packages that process Accounts Receivable information.

For additional information, refer to the API Technical Documentation available via the Customer Support Center.

Overview

The API is actually a program comprised of four different packages: a primary package containing query functions and create, update, and delete procedures, a Rules package containing subprograms that support the primary package, a Strings package containing error messages, and an associated DML package that performs the actual database maintenance of the edited data.

In addition to the API packages, there are three supporting package types.

- *Utility* - Provides utility functions and procedures for the Banner APIs.
- *Validation* - Contains standard functions and procedures for a specific validation table.
- *Support* - Contains supporting programs that may be called by other APIs and error messages related to these edits.

The APIs are called by programs within Banner (for example, forms, C processes, self-service packages), and may be called by external systems that need to create, update, or delete information stored in Banner tables.

Various institutions requested that SunGard Higher Education support school-specific validation logic with the APIs. The Banner APIs include connection points that you can use to interface with your own code to enable your validation logic without having to change baseline code.

Note

In addition to standard User Exits for Create and Update functionality throughout the APIs, the `tb_misc_detail_rules` package includes a User Exit to enable editing of the Accounting String (Banner Finance not installed) or FOAPAL information (Banner Finance installed) when ledger numbers are enterable for miscellaneous detail.

APIs and Supporting Packages used in Banner

The following charts identify the API packages and supporting packages used to process Accounts Receivable data in Banner. The APIs replace the corresponding code in the Banner forms.

API Packages

Object Name/ Entity Name	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_customer_ profile	CUSTOMER_ PROFILE	Contains the Common Business interface for the Customer Profile API (CUSTOMER_ PROFILE).	TBBCPRF	TGACPRF	tb_customer_ profile_rules Provides supporting subprograms for CUSTOMER_PROFILE. tb_customer_ profile_str Provides error messages and error message functions for CUSTOMER_ PROFILE.
tb_contract_ auth	CONTRACT_ AUTH	Used to maintain contract student authorizations in TBBCSTU	TBBCSTU	TSPCSTU	tb_contract_ auth_rules Provides supporting subprograms for CONTRACT_ AUTH tb_contract_ auth Provides error messages and error message functions for CONTRACT_ AUTH.
tp_ application	Not Applicable	Used to perform application and unapplication of payments	TBRACCD TBRAPPL	TSAUNAP TFAUNAP TGRAPPL TGRUNAP	

Object Name/ Entity Name	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_contract_ payment	CONTRACT_ PAYMENT	Used to maintain contract payment detail in TBRCPDT	TBRCPDT	TSACPDT TSPCPDT	tb_contract_ payment_rules Provides supporting subprograms for CONTRACT_ PAYMENT tb_contract_ payment Provides error messages and error message functions for CONTRACT_ PAYMENT.
tb_deposit	DEPOSIT	Contains the Common Business interface for the DEPOSIT API (DEPOSIT).	TBRDEPO	TFADETF TFADETL TFAMASS TGACREV TGPBILL TGRCLOS TGRFEED TSADETF TSADETL TSAMASS TSASPAY TSRCBIL TSRTBIL	tb_deposit_rules Provides supporting subprograms for DEPOSIT. tb_deposit_str Provides error messages and error message functions for DEPOSIT.
tb_memo	MEMO	Contains the Common Business interface for the MEMO API (MEMO).	TBRMEMO	TFADETL TGPBILL TSAACCT TSACONP TSACONT TSADETL TSASPAY TSRCBIL TB_DEPOSIT	tb_memo_rules Provides supporting subprograms for MEMO. tb_memo_str Provides error messages and error message functions for MEMO.

Object Name/ Entity Name	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_misc_ detail	MISC_ DETAIL	Contains the Common Business interface for the Miscellaneous Receipt Detail API (MISC_ DETAIL).	TBRMISD	TFAMISC, TGACREV TGRCLOS TGRFEED TSAMISC TSKMODS BWSKAPMT BWSKWCCP	tb_misc_detail_ rules Provides supporting subprograms for MISC_DETAIL and includes a User Exit to edit Accounting Information when ledger numbers are enterable. tb_misc_detail_ str Provides error messages and error message functions for MISC_DETAIL.
tb_misc_ header	MISC_ HEADER	Contains the Common Business interface for the Miscellaneous Receipt Header API (MISC_ HEADER).	TBBMISC	TFAMISC, TGACREV, TGRCLOS TGRFEED TSAMISC TSKMODS BWSKAPMT BWSKWCCP	tb_misc_header_ rules Provides supporting subprograms for MISC_HEADER. tb_misc_header_ str Provides error messages and error message functions for MISC_HEADER.

Object Name/ Entity Name	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_receivable	RECEIVABLE	Contains the Common Business interface for the Student and Non-Student Accounts Receivable API (RECEIVABLE).	TBRACCD	TFAADSP, TFADETF TFADETL TFAISTL TFAMASS TFAMDET TFAUNAP TFRBILL TFRLATE TFRRFND TGACREV TGPBILL TGRAPPL TGRCLOS TGRFEED TGRUNAP TSAADSP TSAAREV TSADETF TSADETL TSAISTL TSAMASS TSASPAY TSAUNAP TSKFUNC TSKMODS TSP1098 TSPISTT TSRLATE TSRLBOX TSRRFND	tb_receivable_rules Provides supporting subprograms for RECEIVABLE. tb_receivable_str Provides error messages and error message functions for RECEIVABLE.

Object Name/	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_receivable (continued)				TSRTBIL TTKMODS RPEDISB ISRKBIL ISRKTRN SAAADMS SFAWDRL SFKFEES SFKMODS SFRRNOP SHACATT SHADEGR SHADIPL SHAMCAT SHAMDEG SHAMDIP SHREDIY SHRTRTC SLAMASG SLAPASG SPARASG SLRFASM SMACACT SMARQCM	
tb_statement	STATEMENT	Contains the Common Business interface for the Statement API (STATEMENT).	TBBSTMT	TSRCBIL TGPSTMT	tb_statement_str Provides error message and error message functions for STATEMENT. tb_statement_ rules Provides supporting subprograms for STATEMENT.

Note

This API will interact with the LARGE_OBJECT API (gb_large_object) to store a binary file. Please refer to the API Technical documentation of LARGE_OBJECT for details regarding the parameters handling file storage.

Object Name/	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_term_control_ar	TERM_CONTROL_AR	Contains the Common Business interface for the Term Control API (TERM_CONTROL_AR).	TBBTERM	TSATERM	tb_term_control_ar_rules Provides supporting subprograms for TERM_CONTROL_AR. tb_term_control_ar_str Provides error messages and error message functions for TERM_CONTROL_AR.
tb_aid_year_desig	AID_YEAR_DESIG	Performs all validations related to the maintenance of aid year-based designator rules and performs DML operations on TBBABDS table.	TTVAIDY		tgkb_aidydesg0.sql tgkb_aidydesg1.sql tgkb_aidydesg_r0.sql tgkb_aidydesg_r1.sql tgkb_aidydesg_s0.sql tgkb_aidydesg_s1.sql tokd_tbbabds0.sql

DML Packages

The DML packages in the following chart provide the procedures to insert, update, and delete from various Banner Accounts Receivable tables.

Package Name	Associated Tables
dml_tbbabds	Aid year-Based Designator Rules Table (TBBABDS)
dml_tbbcprf	Customer Profile Attributes Table (TBBCPRF)
dml_tbbmisc	Miscellaneous Transaction Receipt Header Table (TBBMISC)
dml_tbbterm	Accounts Receivable Term Control Table (TBBTERM)
dml_tbrcpdt	Third Party Contract Payment Detail (TBRCPDT)
dml_tbrcstu	Contract Student Authorization Table (TBBCSTU)
dml_tbrmemo	Charge/Payment Detail Memo Table (TBRMEMO)

Package Name	Associated Tables
dml_tbrmisd	Miscellaneous Transaction Charge/Payment Detail Table (TBRMISD)
dml_tbraccd	Account Charge/Payment Detail Table (TBRACCD)
dml_tbrdepo	Deposit Table (TBRDEPO)

Support Packages

The following chart identifies API utility packages that contain supporting programs that may be called by other APIs and error messages related to these edits.

Object Name	Entity Name	Task Performed	Table	Forms & Processes	Associated Packages
tb_appl_ detail	APPL_DETAIL	Provides functional support for the tp_application process level API	TBRAPPL	Not Applicable	tb_appl_ detail_rules Provides supporting subprograms for APPL_DETAIL tb_appl_ detail Provides error messages and error message functions for APPL_DETAIL.
tb_cashier _ session	CASHIER_ SESSION	Provides functional support for the Cashiering API (CASHIER_ SESSION). In a future release, this package may provide the Common Business interface for the Cashiering API.	TBBCSHR	Not Applicable	tb_cashier_ session_rules Provides supporting subprograms for cashiering. tb_cashier_ session_str Provides error messages and error message functions for cashiering.

Object Name	Entity Name	Task Performed	Table	Forms & Processes	Associated Packages
tb_depo_control	DEPO_CONTROL	Provides the Common Business interface for the Deposit Control API (DEPO_CONTROL).	TBBDEPC	Not Applicable	tb_deposit_control_rules Provides supporting subprograms for DEPOSIT_CONTROL. tb_deposit_control_str Provides error messages and error message functions for DEPOSIT_CONTROL.
tb_detail_code	DETAIL_CODE	Provides functional support for the Detail Code API (DETAIL_CODE). In a future release, this package may provide the Common Business interface for the Detail Code API.	TBBDETC	Not Applicable	tb_detail_code_str Provides error messages and error message functions for DETAIL_CODE.

Utility Packages

The following chart identifies API utility packages that provide utility functions and procedures for the Banner Accounts Receivable APIs.

Object Name	Entity Name	Task Performed	Tables	Forms & Processes	Associated Packages
tb_common	COMMON	Provides utility functions and procedures for Banner Accounts Receivable APIs.	Not Applicable	TSPCSTU TSPCPDT	tb_common_str Provides error messages and error message functions for COMMON.
tgkfoap	Not Applicable	Provides interface to Finance packages for FOAPAL and security validations.	Not Applicable	TFAMISC	ffkfoap fakrdef

Validation Packages

The following chart identifies API utility packages that provide standard functions and procedures for the specified Accounts Receivable validation tables.

Object Name	Task Performed	Associated Table
tb_ttvabds	Contains standard functions and procedures for the Aid Year-Based Designator Code Validation	TTVABDS
tb_ttvaidy	Contains standard functions and procedures for the Aid year Code	TTVAIDY
tb_ttvdcat	Contains standard functions and procedures for the Detail Charge/Payment Category Code Validation Table.	TTVDCAT
tb_ttvecat	Contains standard functions and procedures for the Entity Category Code Validation Table.	TTVECAT
tb_ttvsrce	Contains standard functions and procedures for the Billing Detail Source Validation Table.	TTVSRCE

5 System-Required Data

Banner is a comprehensive system with many parts that work together to manage your institution's data and to interact with users. When any one of the components of the system is missing, some of the system's functions may fail or may not work as intended.

In some cases, data itself can be considered an essential component of the system. The complete contents of certain tables, and specific rows in other tables, must be present for the system to work correctly. This special data is called *system-required data*. System-required data is a subset of the seed data delivered with a new Banner installation. Banner software releases often include seed data scripts that deliver additional system-required data.

Generally, Banner forms and processes will prevent you from deleting system-required data. But when you are using database tools or scripts to delete rows from the database—for example, during database cleanup to remove sample data before migrating into production—there is nothing to prevent system-required data from being accidentally deleted. In those situations, you should take care not to delete any system-required data.

This chapter lists system-required data for Banner Accounts Receivable. For more information on system-required data, please see the *Banner General Technical Reference Manual*.

Charts

This chapter summarizes system-required data in the following charts:

- "Accounts Receivable Tables" on page 5-1 - This chart identifies tables for which seed data rows should not be deleted.
- "System-Required Rows for Validation Tables" on page 5-2 - This chart identifies specific values in system-required rows for the listed tables.

Accounts Receivable Tables

This section identifies tables for which data must be created before the system can be used. In some tables the delivered seed data rows can be modified or deleted as necessary for the business model of the institution, while in others (flagged with an asterisk (*) in the following chart) the delivered seed data rows should not be deleted. You can add your own specific data and can also modify delivered data—but exercise caution when doing so. It is recommended that you save a data export of these tables before making any changes.

TBBABDS	Aid Year-Based Designator Rules Table
TBBCTRL	Accounts Receivable Billing Control Table
TBBDEPC	Deposit Detail Definition Table
TBBDETC	Detail Charge/Payment Code Definition Table
TBBEABD	Effective Dated Aid Year-Based Designator Definition Base Table
TBBEACT	Detail Code Effective Date Table
TBBETBD	Term-Based Designator Effective Date Table
TBBTBDS	Term-Based Designator Rules Table
TBRACCT	Detail Code Account Definition Table
TTVBILL*	Bill Code Validation Table
TTVDCAT*	Detail Charge/Payment Category Code Validation Table
TTVDTYP*	Deposit Type Validation Table
TTVPAYT*	Detail Code Payment Type Validation Table
TTVSRCE*	Billing Detail Source Validation Table
TURVERS	Accounts Receivable Version Tracking Table
TVVAUTH*	Title IV Authorization Validation Table

System-Required Rows for Validation Tables

System-required data rows in Banner Accounts Receivable are listed below. The list is organized alphabetically by validation table name. These tables may contain additional values.

- Bill Code Validation Form (TTVBILL)
- Detail Category Code Validation Form (TTVDCAT)
- Deposit Type Code Validation Form (TTVDTYP)
- Payment Type Code Validation Form (TTVPAYT)
- Charge/Payment Source Code Validation Form (TTVSRCE)
- Title IV Authorization Validation Table (TVVAUTH)

TTVBILL — Bill Code Validation Form

NO	No Bill Will Be Generated

TTVDCAT — Detail Category Code Validation Form

APF	Admissions Application Charges
DEP	Deposit
FEE	Registration Fees
HOU	Housing Charges
MEA	Meal Plan
PHO	Phone Plan
PPL	ParentPlus Title IVPayment
TRN	Transcript Charges
TUI	Registration Tuition

TTVDTYP — Deposit Type Code Validation Form

FEE	Fee Deposit
HOU	Housing Deposit
MEA	Meal Deposit
PHO	Phone Deposit
TUI	Tuition Deposit

TTVPAYT — Payment Type Code Validation Form

N	Miscellaneous Credits

TTVSRCE — Charge/Payment Source Code Validation Form

Note

Except where noted, **System Required** is *Y* (selected) and **Allowed on Transaction Entry** is *N* (not selected).

A	Admissions Module
B	Location Management Room Chg
C	Contract
D	Deposit
E	Exemptions
F	Financial Aid
G	Recurring Receivables Assessments
H	Academic History Module

TTVSRCE — Charge/Payment Source Code Validation Form

I	Installment Process
L	Billing Late Charge
M	Memo
N	Non-Sufficient Fund Charge
P	Application of Payment Process
R	Registration Module
S	Miscellaneous Transaction
T	Charge/Payment Transaction (**Allowed on Transaction Entry** = *Y*)
U	Location Management Phone Charge
V	Location Management Meal Charge

TVVAUTH — TItle IV Authorization Validation Table

HLD	Hold excess TIV funds for future
PY	Pay Prior Year minor inst charges
TIV	Pay non-inst charges with Title IV
PTV	ParentPlus pay noninst charges
PPY	ParentPlus pay prior year balance
PHD	ParentPlus hold excess TIV funds

A Frequently Asked Questions

This section includes guidance on questions that may be encountered as you implement e-Bill functionality to store and view statement files.

Do I have to create Adobe Portable Document Format (PDF) files to use the e-Bill enhancement?

No, it is not a requirement that the PDF format be used, only that the output from TSRCBIL be divided into multiple files: one per statement. The plain text output from the billing process may be parsed, stored, and viewed.

However, the conversion to PDF allows for a much better presentation of the statement through the addition of graphical elements including column separators and the branding of this important communication with your logo. Since third-party tools exist to perform this conversion, SunGard Higher Education has not attempted to recreate this functionality.

How can I parse the TSRCBIL output into multiple files?

Each statement includes a unique number (although there may be more than one page for a statement), and it is recommended that this number be used to name the individual files as they are parsed.

Here is an example of a Perl script that can be used to perform the parsing of the TSRCBIL output. Note that the correct value must be set for the `$stmt_line variable` depending on whether you are parsing STATEMENT or SCHEDSTMT output. If the page break appears as something other than CTRL-L in your output, the escape sequence needs to be changed in both places it occurs.

Sample Perl Script

```
#!/usr/bin/perl
#
# parse_cbil.pl - splits tsrcbil output into separate files,
#   basing the file names on the stmt_number in the output.
# For STATEMENT mode, initialize $stmt_line = 3; to find the stmt_number
#   on the 3rd line of each page.
# For SCHEDSTMT mode, initialize $stmt_line = 58; to find the stmt_number
#   on line 58 of each page. If you have changed the BLANK_PAPER_ line
#   definitions in the TSRCBIL source code, adjust this value as needed.
# Note that the script is searching for CTRL-L as a page break using \cL
#   in two places below.  Change as necessary for the environment.
#
# Usage is: perl parse_cbil.pl <tsrcbil_nnnnnnn.lis
#

open (SLOG, ">>tempfile.lis")||die "Cannot open temp file out\n";
$stmt_line = 3;
$i = 0 ;
$k = 0 ;
while (<STDIN>) {
  $i++;
  if ($i==$stmt_line)
  {
    @fields = split;
    $fname=$fields[0];
  }
  if (/\cL/)                  # Start of a page
  {
    if ($k==1)                # Continue writing multi-page statement
    {
      print SLOG $_;
      $i = 1 ;
      $k=0;
    }
    else                      # Close and rename file, start next file
    {
        close (SLOG);
        # rename it to the correct name
        rename ("tempfile.lis","${fname}.lis");
        # open next
        open (SLOG, ">>tempfile.lis")||die "Cannot open temp file out\n";
        $i = 1 ;
        $_ =~ s/\cL//;          # Omit the page break
        print SLOG $_;          # Print the line out
    }
  }
  elsif ((/ * CONTINUED /) && ($i > 8))
  {
    $k=1;                     # Mark as multi-page statement
    print SLOG $_;
  }
  elsif (/REPORT CONTROL INFORMATION /)
  {
    $fname="control";         # This is the Control Report
    print SLOG $_;
  }
  else
  {
    print SLOG $_;            # Print the line out
  }
}
close (SLOG);
rename ("tempfile.lis","${fname}.lis");
```

How can I Store Statements from the File System?

Once individual files for each statement have been created, use BANINST1 to create an Oracle directory that points to the folder where the files are located. Grant read access as necessary if a user other than BANINST1 will run the following script:

```
create or replace directory AR_LOAD_DIR
as '/server/banner/dataload/ar';
create public synonym AR_LOAD_DIR for AR_LOAD_DIR;
grant read on directory AR_LOAD_DIR to <my-user>;
```

PL/SQL Script

The `tb_statement` API is used to store each file in the database. This example is for Plain Text files, assuming that the `tsrcbil.lis` output was parsed into files named `<stmt_number>.lis`. It uses the Job Submission number from the TSRCBIL run to identify all of the statement numbers produced, and updates the record with the file with that number.

Be sure to use the correct large object type `(p_blot_code)` for the file being stored, as this determines how the browser will display the file when it is viewed.

```
set echo off;
set serveroutput on;
ACCEPT bill_run_in PROMPT 'Enter Job Submission Number '

DECLARE
   lv_file_name          VARCHAR2(30);
   stmt_refcur           tb_statement.statement_ref;
   stmt_rec              tb_statement.statement_rec;
   bill_run              tbbstmt.tbbstmt_bill_run%TYPE;
   lv_count              NUMBER := 0;
   COMMIT_POINT CONSTANT PLS_INTEGER := 10;

BEGIN
   bill_run := &bill_run_in;

   stmt_refcur := tb_statement.f_query_all(
                  p_bill_run => bill_run);
      LOOP
         FETCH stmt_refcur INTO stmt_rec;
         IF stmt_refcur%FOUND
         THEN
           IF stmt_rec.r_media_id IS NULL THEN
             lv_file_name := TO_CHAR(stmt_rec.r_stmt_number)||'.lis';
             tb_statement.p_update(
                p_stmt_number      => stmt_rec.r_stmt_number,
                p_user_id          => 'APITEST',
                p_data_origin      => 'STATEMENT_LOAD',
                p_blot_code        => 'TXT',
                p_directory        => 'AR_LOAD_DIR',
                p_filename         => lv_file_name,
                p_loadfromfile     => 'Y');
           END IF;
           lv_count := lv_count + 1;
           IF lv_count > COMMIT_POINT THEN
              gb_common.p_commit;
              lv_count := 0;
           END IF;
         END IF;
         EXIT WHEN stmt_refcur%NOTFOUND;
      END LOOP;
      CLOSE stmt_refcur;
      gb_common.p_commit();
EXCEPTION
  WHEN OTHERS THEN
    gb_common.p_rollback();
    dbms_output.put_line('Error processing statement:'||
        stmt_rec.r_stmt_number);
    DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;
```

Note

If you prefer to keep the files in the physical file system you could do so by changing the value of `p_loadfromfile` above to *V* (validate) or *N* (No validation) in order to store only a file location pointer in the database. ■

Please refer to the *Banner General Release Guide* and the technical documentation for additional information about the file storage parameters and options available for handling the statement files.

How might local modifications or Third Party tools be used in processing statements?

Handling statements for the e-Bill enhancement has three basic steps, two required and one optional. Local modifications or tools from other vendors might help with any or all of these steps.

- Step1 - Create individual statement files (required)
- Step 2 - Enhance and brand the output, convert to binary file type (optional)
- Step 3 - Store the file (required)

For Step 1, if a local process has been written to do the billing, it might be modified to use the `tb_statement` API to generate a statement number. The output could be a single file including the statement number, or the process might be altered to create individual output files named with the statement number. For either baseline TSRCBIL or a local process creating one file, a third-party tool that can parse a plain text file could be used in place of the Perl script above to create the individual files.

For Step 2, any tool which can read a plain text file, enhance it with user-defined changes, and create a binary document file with which the resulting output can be used. A tool that can do steps 1) and 2) together simplifies the processing.

For Step 3, any tool that can communicate with the Banner database through a PL/Sql block can use the `tb_statement` API to store the file. The sample script above shows the basic form such a call would take as an update to an existing record in tbbstmt.

For additional information, please refer to the following FAQs available in the Solutions section of the Customer Support Center:

- FAQ #1-IHS99 (Using AR e-Bill)
- FAQ #1-IPPO9 (Using Evisions with AR e-Bill)

Where does the statement number appear in the billing output?

- When you run TSRCBIL in STATEMENT mode, the statement number appears on the 3rd line, beginning in the first position.

- When you run TSRCBIL in SCHEDSTMT mode, the statement number link appears on line 58 (unless the value of the BLANK_PAPER_BOTTOM_TOTALS_START_LINE has been modified in the TSRCBIL source code). The column in which it begins is dependant on the width of the schedule bill; the width varies with the choices made for columns to display on TSASBRL, but will be aligned to the beginning of the *Student Id* literal in the next line of the payment stub.
- When you run TSRCBIL in INVOICING or SCHEDINV mode, values from the statement number sequence are used during processing, but these values will be deleted from the TBBSTMT table. As a result, there will be gaps in the values that are in the tbbstmt_stmt_number column.

Why do I see the same statement when I try to look at a different bill date?

Your Internet Options can affect your ability to view statements. If you are having difficulty viewing the statement you selected, access the Internet Options for your browser and check the following.

1. Ensure that your preferences for temporary file handling are set up to look for newer versions of stored pages.
2. Clean out the cache.

Index

A

C

D

E

F

I

L

O

P

R

S

T

U

V

W

4375 Fair Lakes Court
Fairfax, Virginia 22033
United States of America
www.ellucian.com