Sigma Systems, Inc.

Overview Data Model Kuali Student Accounts System (KSA)

Sigma Systems
March, 2012

Change Log

Author	Date	Changes
Paul	3/2/2012	Added a changelog.
Paul	3/2/2012	Per discussion with Michael, cleaned up the KSSA_CREDIT_PERMISSION table. ALLOWABLE_DEBIT_TYPE_ID. Cleaned up KSSA_ALLOCATION to create an autonumber primary key, and made ACCOUNT_ID -> ACCOUNT FLAG ->FLAG_TYPE, KSSA_INFORMATION.FLAG_ID became FLAG_TYPE_ID_FK
Paul	3/7/2012	Cleared formatting for data elements. Simplified layout.
Paul	3/7/2012	Did the major part of the account table.
Paul	3/12/2012	Cleaned up account tables.
Paul	3/12/2012	Standardized FRIENDLY_NAME->NAME. All types now have DESCRIPTION instead of INFORMATION. Added DESCRIPTION where missing to standardize _TYPE tables. For easier subclassing. Removed SIMPLE_TYPE (unused). CURRENCY.ID changed to an INT.
Paul	3/14/2012	Removed problematic _FK issue in ACNT table.
Paul	3/16/2012	Added missing IS_DEFAULT to ELECTRONIC_CONTACT. ACNT.KIM_ENTITY -> ACNT.ENTITY_ID.
Paul	3/28/2012	To improve OR/M: Cleaned up CREDIT||DEBIT TYPE tables into a single TRANSACTION_TYPE table. Dropping CREDIT_TYPE, DEBIT_TYPE, CREDIT_TYPE_TAG, DEBIT_TYPE_TAG. Created new TRANSACTION_TYPE, TRANSACTION_TYPE_TAG. TRANSACTION_TYPE gained TYPE element. Elements from DEBIT||CREDIT_TYPE were merged into TRANSACTION_TYPE. AUTH_TEXT became AUTH_TXT to follow naming convention. All tables that referenced CREDIT||DEBIT type names (TRANSACTION_TYPE, GL_BREAKDOWN, CREDIT_PERMISSION and TRANSACTION) changed their foreign key names to reference the standard TRANSACTION_TYPE key names. (TRANSACTION_TYPE_ID_FK, TRANSACTION_TYPE_SUBCODE_FK)
Paul	3/29/2012	LEVEL became ACCESS_LEVEL due to Oracle problems. Affected INFORMATION, FLAG_TYPE and TAG. CREDIT_PERMISSION. ALLOWABLE_DEBIT_TYPE_ID became ALLOWABLE_DEBIT_TYPE_MASK.
Paul	3/30/2012	CURRENCY identifier is now a standard number, and the ISO value is used for lookup if required.
Paul	5/29/2012	Added ENTITY_TYPE to ACTIVITY to permit direct interrogation of AuditibleEntity items.
Paul	5/30/2012	Cleaned up the ACTIVITY table.
Paul	5/30/2012	Added UI Pack tables.
Paul	5/31/2012	Added UIPack Version and overrides.
Paul	6/1/2012	Added dob to ChargeableAccount for bio lookup. ACTIVITY changed to add ALTERED_ENTITY_PROPERTY. Added general ledger classes.
Paul	6/2/2012	Added identity types to the account data model.
Paul	6/7/2012	Added ACNT_PREF
Paul	6/8/2012	Added BATCH_RECEIPT
Paul	6/12/2012	Altered UI pack tables to permit auditable languages.

Paul	6/15/2012	Added REFUND_TYPE, REFUND. Made alterations to all AuditableEntries to reflect CREATION_DATE
Paul	6/26/2012	Added TRANSACTION_TYPE_PERMISSION
Paul	7/3/2012	TRANSACTION_TYPE_PERMISSION -> TRANSACTION_TYPE_ROLE. UI_PACK removed version.
Paul	7/16/2012	Working on cash limit tracking model.
Paul	7/18/2012	Added the CODE reference for all auditable entities.
Paul	7/20/2012	Added external statement tables.
Paul	7/24/2012	Changes to underlying account tables to permit reuse of PERSON_NAME, POSTAL_ADDRESS and ELECTRONIC_CONTACT tables. Addition of BILL_AUTHORITY to support KSA-RM-BP.
Paul	8/1/2012	Added GL_TYPE_ID_FK to the TRANSACTION table, to account for some differences with payment billing.
Paul	8/2/2012	Changed to dual-keys for account/name, etc. mapping.
Paul	8/2/2012	Minor changes to permit more flexible use of the general ledger type columns. (See TRANSACTION_TYPE)
Paul	8/3/2012	Added session to GL tables.
Paul	8/7/2012	Added system preference table.
Paul	8/8/2012	Added IRS1098T table.
Paul	8/13/2012	Removed session from GL transactions.
Paul	8/14/2012	Added TRANSACTION.RECOGNITION_DATE
Paul	8/15/2012	Added ACNT_BLOCK_TYPE table.

Contents

Data Models for the Transaction and Associated Classes

General Notes

- All tables are preceded with KSSA (Kuali Student – Student Accounts) to prevent table name clashes. This follows the rules established by Rice.
- Appropriate contractions are used as suggested on the Rice and KS wiki, including but not limited to TRN-Transaction and ID- Identifier, AMNT- amount. Foreign keys, where they apply to our tables, are tagged with _FK.
- These data models have been designed to support the permanence layer of the Transaction class, its children and its associated classes. Only the permanence layer will access this data structure directly.

There are a number of reused data structures that are not repeated in the document. It is assumed that they are understood.

CREATOR_ID and EDITOR_ID are the identifiers for the entity who creates, and if appropriate, subsequently edits a record. CREATION_DATE is the date when an entry first comes in to being. LAST_UPDATE is the date stamp for the last alteration to the structure. ACCESS_LEVEL refers to a general access level that is defined by the institution. It is stored as a plain INT, and a user must have a LEVEL equal to or greater than the LEVEL of the referenced information to be able to view it. Few levels are expected in reality (as roles exist to give much more granular controls) but an example might be 0 for Student (this is expected) 1 for external staff (departments, etc.) 2 for internal bursar staff (and default for memos, etc.) and 3 for high-level employees in the bursar's office.

As all tables begin KSSA_, the prefix is not reused in the descriptions.

Transaction Type

Generic

This model underlies the TransactionType object, which is an associated class with the Transaction class. Every Transaction HAS_A TransactionType.

Most often, transactions are categorized using a preset numbering system, (for example, TUT*** are tuition codes, etc.) However, there are times when the transaction codes do not permit flexible categorization for certain reporting purposes. Tags are an optional way to allow control over categories of transactions.

These tables underpin subclasses of CreditType and DebitType. For improved OR/M, there is a type disambiguation stored in the TRANSACTION_TYPE table.

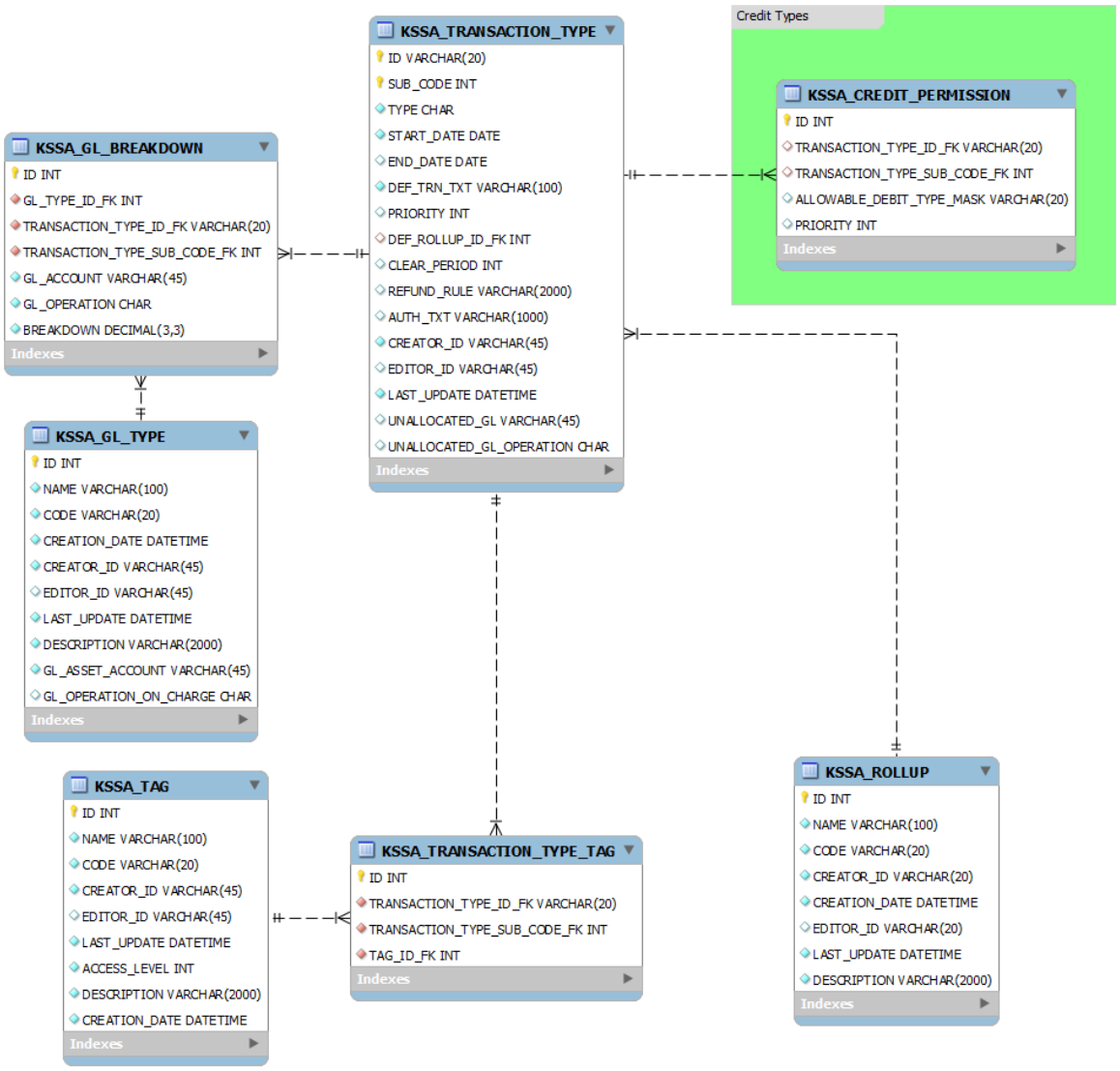

Table: TRANSACTION_TYPE

(Central definition of a transaction type)

ID	Transaction type identifier.
SUB_CODE	As transaction types can change over time, the SUB_CODE field can be used to look up the specific details of a transaction type at a certain period in time.
TYPE	C indicates that this a Credit Type, D that this is a Debit Type.
START_DATE	Mandatory field, defining when the transaction type came into existence. By default, it is the date that the transaction type was created.
END_DATE	Optional field. If null, it is the current transaction type.
DEFAULT_TRN_TXT	Default text for this transaction as shown on the statement. When a transaction is created, the default text is taken from this field, unless otherwise specified in the transaction.
PRIORITY	This field is used by payment application. As a general rule,

	higher priority debits will be paid before lower priority debits. Equal priority debits are paid off FIFO.
DEF_ROLLUP_ID_FK	Links the transaction to a default rollup. This can be overridden in the transaction. This foreign key is the primary key in the ROLLUP table.
CLEAR_PERIOD	**(Credit Types Only)** A time period, specified in days, after which the payment is considered to be 'cleared'. For example, an institution may implement a 10-day hold on check payments.
REFUND_RULE	**(Credit Types Only)** Currently just a text placeholder for a way of encoding this rule. It is only applicable if the isRefundable flag is true in the credit object. Some use cases are listed below. Note that the general REFUND_RULE for a TRANSACTION_TYPE can be overridden in the TRANSACTION table. If TRANSACTION.REFUND_RULE = null, then TRANSACTION_TYPE.REFUND_RULE is used. Otherwise TRANSACTION.REFUND_RULE Is followed. TRANSACTION.REFUND_RULE permits the same rules as TRANSACTION_TYPE.REFUND_RULE, as well as also permitting a refund to another KSA account. See the next page for the refund rule encoding, under "Refund Rule Use Cases"
AUTH_TXT	**(Credit Types Only)** This is a friendly text field to assist the user when processing payments. It stipulates the expected reference that the payment will be. In the case of a credit card, for example, the authorization code from the credit card company might be the reference stored for the transaction. In the case of a check, the bank information and the check number might be stored. This is institution and payment specific.
UNALLOCATED_GL	**(Credit Types Only)** If a credit cannot be allocated, this is the GL account that is affected. This may be a suspense account, an unearned income account, etc.
UNALLOCATED_GL_OPERATION	**(Credit Types Only)** Defines whether the GL account in the previous line will Credit or Debit the GL account.

Refund Rule Use Cases

Many types of refunds are referred to as "cash". This term is used loosely to mean the money is returned to the student with no restrictions. In reality, most "cash" refunds would take place via banking transfer or paper check. In this context, a "cash" refund means that the money is unrestricted in its repayment method, and will be refunded using the student's default refund method.

The refund rule allows for the following scenarios:

- A refund in cash or equivalent may be issued on this amount after the clearing period (in the case of cash, the clearing period would be 0) (example, cash, checks)
- A refund to the original source may be made after the clearing period. (example, credit cards)
- A refund to the original source may be made for a specified period of time, after which a refund in cash or equivalent may be made. (for example, credit cards.)
- A refund may be made to another KSA account, for example, if a student with sponsorship might not be eligible for a refund of sponsorship money if they do not use all the money on their account.

The refund rule works with the clearing period to allow these use cases. For cash, there is no refund rule (null) and the clearing period would be 0. For checks, a clearing period appropriate with the banking relationship in force would be set for checks, with a null refund rule.

A refund rule starting with S is a "source" refund, meaning that it should be refunded in the same way as it was paid. Performing this actual refund (for example, to a credit card, to PayPal, etc.) would be handled elsewhere. S takes a numerical parameter in parentheses, which defines the number of days for which this "source" refund is required. A zero means "forever". For example, a PayPal transaction may have a refund rule of S(0) meaning any refund against this amount should always be returned to PayPal. An S(45) refund rule may be applied to a credit card transaction, meaning for the first 45 days, it must be returned to the credit card, however, after that time period, it may be refunded as "cash". This might also be coupled with a clearing period, allowing, for example, a four-day waiting period before a credit card transaction can be refunded.

A refund rule with A is an "account" refund, meaning that the refund should be applied to another KSA account. For example, a sponsor account. Account refunds take two parameters in parenthesis. As with an source refund, the first is the "timeout" period of the source refund. This would be an uncommon situation, but is included to keep the syntax of the rules standard. In most cases, this parameter will be 0, meaning that it does not time out. The second parameter is the account to which the refund will be applied. A clearing period may also be applied, so for example, a sponsor payment might have a clearing period of 180 days, meaning that the money may remain on the account for that period of time, but after that time, if any money remains, it would be refunded to the sponsor account.

Table: ROLLUP

Defines rollups within the system.

ID	Primary autonumber key, autonumber.
NAME	Name of the rollup as displayed on the UI

CODE	The short code for the rollup. This is used as a matching column when a rollup is transmitted to the system. It is always unique. Example "FINAID"
DESCRIPTION	Longer description available upon inspection.

Table: TRANSACTION_TYPE_TAG

(Simple association table of TRANSACTION_TYPES to TAG.)

ID	Primary key, autonumber field for the association.
TRANASCTION_TYPE_ID_FK, TRANSACTION_TYPE_SUB_CODE_FK	Foreign keys of the TRANSACTION_TYPE table that associate the TRANSACTION_TYPE with the TAG
TAG_ID_FK	Foreign key for the TAG to associate the TRANSACTION_TYPE with the TAG.

Table: TAG

(General storage of tags within the system.)

ID	Autonumber primary key for the tag table.
NAME	Short, plain text name of the tag. For example "Tuition", "Books and Supplies".
CODE	The short code for the tag. This is used as a matching column when a tag is transmitted to the system. It is always unique. Example "TUIT"
ACCESS_LEVEL	As defined in the introduction, levels define who can or cannot see a certain item. The higher a user's level, the more the can see. The higher an item's level, the fewer users can see it. For example, tags such as "Tuition" may be visible to all users (level = 0) whereas certain tags might have a higher level to make them less visible to students and other staff.
DESCRIPTION	This is a general field that is used in UX presentment to give further information about the item. In this case, for example, if there were a tag of 1098T (a US tax form that reports tuition) and it were visible to the student, the student would be able to click on the tag and find out more about what it means that certain transactions were tagged as 1098T.

Table: GL_BREAKDOWN

(Breakdown table to the general ledger.)

ID	Primary key autonumber identifier for the GL_BREAKDOWN table.
GL_TYPE_ID_FK	By default this is 0. This field allows for other 'types' of breakdowns to be defined, and can be referenced through the rules engine. For example, if a school writes off its transactions to a different set of GL accounts, then type 1

	could be defined as the write-off type, and could be referenced here. Type 0 is reserved, and is always used for the standard transaction breakdown.
TRANSACTION_TYPE_ID_FK/ TRANSACTION_TYPE_SUB_CODE_FK	These foreign keys reference the TRANSACTION_TYPE table and identify a specific debit type at a specific period in time.
GL_ACCOUNT	A single general ledger account.
GL_OPERATION	Can be (C)redit or (D)ebit. When a transaction is created with this GL type, this is the operation that will be used against this general ledger account.
GL_BREAKDOWN	If the debit type maps to a single general ledger account, then this field is 0. Where a debit type maps to a number of general ledger accounts, then the percentage breakdown for each is store in this field. Then final account in the list is given a breakdown 0, which means "allocate the remainder to this account". An example is given below.

General Ledger Breakdown

Where schools divide single transactions over multiple general ledger accounts, these will be listed in this table, with BREAKDOWN amounts to spread the payment. In this case, percentages can be allocated, and there will be one "bucket" account, which receives the remainder of the funds. This prevents problems with fractional currency being unallocated or over allocated or under allocated. For example, a transaction that divides into two general ledger accounts as a 50/50 split would be defined as:
ACCOUNT 1 – 50%
ACCOUNT 2 – 0 (Bucket account)

Therefore a $100 transaction would divide as:
50% of $100 = $50. ACCOUNT 1 received $50
ACCOUNT 2 gets the remainder, therefore $100-$50 = $50 to account 2.

For a $99.99 transaction
50% of $99.99 = $49.995. With a rounding up, ACCOUNT 1 would be credited $50.00
ACCOUNT 2 would be credited with $99.99-$50 = $49.99

Table: GL _TYPE

Defines GL Types within the system.

ID	Primary autonumber key, autonumber. By default, 0 is the standard type.
NAME	Name of the GL breakdown type.
DESCRIPTION	Longer description available upon inspection.
CODE	Unique code used to match the GL_BREAKDOWN type.
GL_ASSET_ACCOUNT	The general ledger asset account for this breakdown. For example, the default asset account for accounts receivable transactions will be the accounts receivable general ledger. Other types may include third party, payment billing (notes

	receivable) etc.
GL_OPERATION_ON_CHARGE	Can be (C)redit or (D)ebit. When a charge is triggered with this GL type, what operation will be triggered against that account (credit or debit). When a payment is applied to this charge, the opposite operation will be applied.

Table Relating only to Credit Types

Table: CREDIT_PERMISSION

(Stores the relationship between credit types (generally payments) to debit types (generally charges). Used by payment application to decide what charges can be paid by what payments. A credit type may have many credit permissions. This table is primarily used by the payment application algorithm.)

ID	Autonumbered primary key.
TRANSACTION_TYPE_ID_FK, TRANSACTION_TYPE_SUB_CODE_FK	Identifies the exact credit type that is being referenced.
ALLOWABLE_DEBIT_TYPE_MASK	This can either be a foreign key for a debit type, or a masked debit type. If a payment can be applied to any charge, then a wildcard would be in this field. Masking follows the basic SQL wildcard options.
PRIORITY	Priority states the priority of a credit to pay off a certain group of transactions. This is only used if the system needs to break a tie between Debits of the same priority. If two debits have the same priority and the credit is allowed to pay them both, it will pay off the higher priority codes first, before applying the remainder to the remaining codes.

Transaction

Generic

The transaction data model is used to store the discrete transactions within the KSA system. Every charge and payment (and deferment) is stored as a transaction.

TRANSACTION is identified by ID and lists all the headline information about any transaction. Every Transaction can be either a Credit or a Debit. A Credit can be a Deferment or a Payment, a Debit is a Charge.

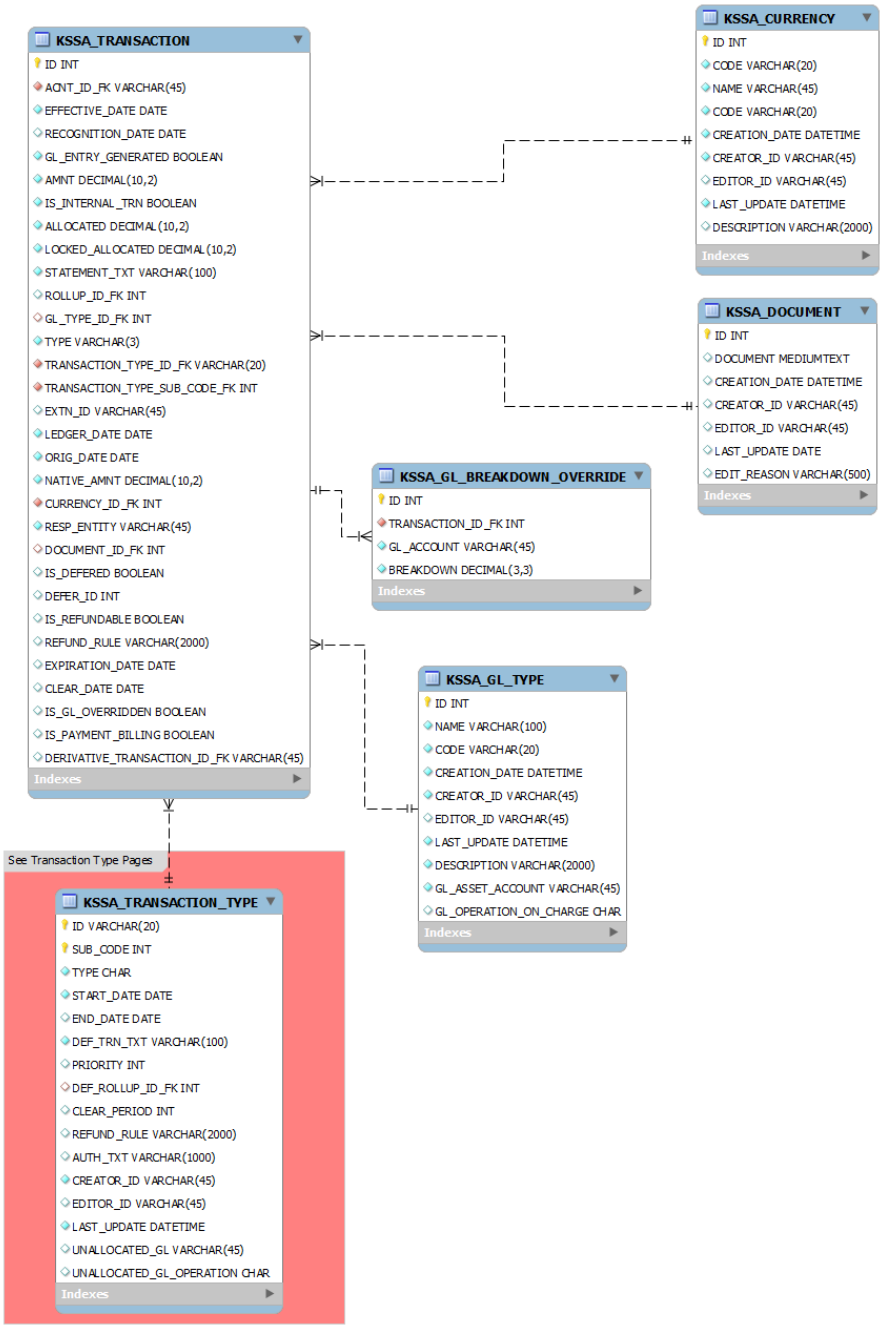

Table: TRANSACTION

(Stores transactional financial data.)

ID	Autonumbered primary key to make each transaction unique.
ACNT_ID_FK	KSA account number with which this transaction is associated.
EFFECTIVE_DATE	This is the date that the transaction is considered "current" on the account. This is the date around which all processing is based. A transaction does not begin to 'age' until the this

	date is current., nor is it registered in the general ledger. This date is also used in relation to the credit and debit types to define how the transaction behaves.
RECOGNITION_DATE	This date drives the recognition period as used in the general ledger. By default, it is set to the EFFECTIVE_DATE, but this can be changed. This date only effects the recognition code on the general ledger, and not "when" the actual general ledger transactions are generated.
GL_ENTRY_GENERATED	Indicates whether or not the transaction has become effective, and been entered into the general ledger?
AMNT	This is the value of the transaction in the system specified currency. Note that although the decimalization is overkill for US dollars, the wider field is used to accommodate other currencies.
IS_INTERNAL_TRN	Boolean that dictates whether a transaction is considered "internal" or not. Internal transactions are generally not presented to customers.
ALLOCATED	The amount of currency) that is allocated. For a payment, it is the amount of the payment that is allocated to a charge. For a charge, it is the amount of the charge that has been paid.
LOCKED_ALLOCATED	This works the same as ALLOCATED, except this payment allocation cannot be de-allocated by the automatic payment allocation module. A CSR might choose to lock certain payments and charges together, or a school might choose to freeze all allocations from a previous period to stop the system from de-allocating earlier allocations.
STATEMENT_TXT	'Friendly text' for a transaction that is displayed to explain the purpose of the transaction. This is derived from the credit or debit type in the first instance, but might be overridden in an individual transaction.
ROLLUP_ID_FK	A field used to group similar transactions in to a rollup. For example, if a number of add/drops occur within a period, all of the charges and refunds might be given the ROLLUP_ID that refers to "Tuition". Then, on the initial view of the statement, the word "Tuition" would appear with a net of the values. Further inspection would allow the full list of transactions that make up the rollup appear. The ROLLUP table is defined above.
GL_TYPE_ID_FK	Reference to the GL Type table. By default, the general ledger type is 0, which is used for all transactions that do not specify a general ledger type. For special transactions (for example, third party charges, etc.) a different general ledger type may be used.
TYPE	References the type of transaction; acceptable values are TCP for TRANSACTION->CREDIT->PAYMENT, and TCD for TRANSACTION->CREDIT->DEFERMENT and TDC for TRANSACTION->DEBIT->CHARGE.
TRANSACTION_TYPE_ID_FK	Links to the TRANSACTION_TYPE.
TRANSACTION_TYPE_SUB_CODE_FK	This links to the sub code of the TRANSACTION_TYPE

	definitions. The sub code is derived from the type id and the effective date at the point of instantiation.
EXTN_ID	External transaction identifier. For external systems that generate a transaction identifier will populate this with the foreign identifier. If the transaction comes via batch, then the batch id will be here. For payments, the expected authorization code will be inserted here. For example, a credit card payment might use the authorization code as the EXTN_ID.
LEDGER_DATE	The date that the transaction was entered in to the ledger within the KSA-RM system. The "ledger" referred to is the list of transactions relating to an account that is held within the KSA system. It does not refer to the general ledger.
ORIG_DATE	Records when the transaction was generated. This is most useful for transactions occurring in legacy systems that upload their transactions in batches. In many cases, this will be the same as the LEDGER_DATE as transactions will be generated and posted on the same day.
NATIVE_AMNT	Amount of the transaction in the native currency of the transaction. Where the native currency and the system currency are the same, NATIVE_AMNT and AMNT will be the same.
CURRENCY_ID_FK	Currency identifier for the transaction. In most cases, this will equal the system currency. Currencies are stored in the CURRENCY table.
RESP_ENTITY	This is the identifier for the entity who created the transaction. As this can be a non-KSA entity, the KIM entity identifier is stored.
DOCUMENT_ID_FK	Identifier for an XML document that holds information regarding the transaction. This could be a number of different elements. For example, a bookstore transaction could send the names of the books and their prices in the document. This document is for information presentment and is not intended to be used as part of the accounting process.
IS_DEFERED	Boolean that is only applicable to debit transactions. It will be false for credits. If True, then this debit has been deferred. The identifier for the deferment will be stored in DEFER_ID.
DEFER_ID	Reciprocating transaction reference if a transaction is deferred. In the case of a deferment transaction, this will point to the transaction which it defers. In the case of a deferred transaction, this will point at the deferment transaction.
IS_REFUNDABLE	A Boolean that is only applicable to credits. It answers the question "is this credit refundable?" If false, the transaction cannot be refunded if it causes a credit balance. This would be the case for types of credits.
REFUND_RULE	Defines the refund processing rules. If this is blank, then it defaults to the refund rule defined in CREDIT_TYPE. However, if this value is set, then it overrides the rules in

	CREDIT_TYPE and takes priority. This rule follows the same format as CREDIT_TYPE refund rule, except it also permits the option of refunding to another KSA account. This override refund rule is required to permit overpayment refunds to other sources. Use cases would be ParentPLUS loans, as well as sponsorship overpayments. As with REFUND_RULE in the TRANSACTION_TYPE data model, the exact meaning of this field is not yet fully defined, as research is ongoing into use cases.
EXPIRATION_DATE	Only applicable to DEFERMENT transactions, otherwise it is set to null. On this date, the deferment will be expired.
CLEAR_DATE	Only applicable to PAYMENT transactions, otherwise it is set to null. The clear date is used in refund processing as the date as which a transaction is considered to be 'like cash'. For example, when processing a check payment, an institution may choose not to refund against a check until 10 days have passed. By default, this is derived from the system date, as well as the TRANSACTION_TYPE.CLEAR_PERIOD.
IS_GL_OVERRIDDEN	Answers the question "is the default GL breakdown for this transaction overridden?" If this is TRUE, then the default breakdown to the GL as referenced through the debit type on the transaction is ignored, and the values derived from KSSA_GL_BREAKDOWN_OVERRIDE are used instead.
IS_PAYMENT_BILLING	Set to true if the transactions form part of a payment plan.
DERIVATIVE_TRANSACTION_ID_FK	If the charge is derived from other charges (for example, payment billing, third-party billing) then this field will point to the original transactions from which this transaction is derived.

Table: GL_BREAKDOWN_OVERRIDE

(If the general ledger breakdown for a transaction type is not the appropriate breakdown for the transaction, then the IS_GL_OVERRIDEN is set to true, and this table is referenced to give the desired GL breakdown for the transaction.)

ID	Autonumbered primary key.
TRANSACTION_ID_FK	Foreign key to link the transaction to row(s) in this breakdown table.
GL_ACCOUNT	The GL account number to be credited.
BREAKDOWN	The percentage breakdown of the amount. See KSSA_GL_BREAKDOWN table for more information relating to these fields.

Table: CURRENCY

(Stores information about currencies used within the system. A transaction may not be denominated in a currency that does not exist within this table. The system currency will also be present in this table.)

ID	System generated primary key.
CODE	Identifier for the currency in ISO 4217 format. (Examples, EUR, GBP, AUD)
NAME	Name of the currency in the appropriate language for the system. For example Euro, British Pound, Australian Dollar.
DESCRIPTION	If an extended description is warranted in the UI, it would be placed here.

Table: DOCUMENT

(Stores documents as they relate to transactions. It contains details about the transaction that are useful to the user. An example might be if a student purchases books in the bookstore, the document could contain which books were purchased. This information is not intended to be 'interpreted' by the system, rather provide customer service information to the student. The exact format of the document and permissible entries are still under review.

ID	Autonumbered PK for the document.
DOCUMENT	The actual XML formatted document.
CREATION_DATE	Date on which the document was created.
EDIT_REASON	Documents are not generally editable. When a user has the appropriate permissions to edit a document, they should indicate a reason for the document being edited. This will be stored in this field.

Allocation

Generic

The allocation table links together transactions so that the system can track which payments were allocated to which charges. This also means that payments can be de-allocated from charges, and reallocated dynamically, should situations change on the account.

The KSSA_ALLOCATION table is the table of record for allocation information. The TRANSACTION data structure contains grouped information on allocations, but only the ALLOCATION table contains the actual reference of payments and charges. If for whatever reason, the two were to become unsynchronized, the ALLOCATION table would be the table to verify.

Examples of re-allocation would include the posting of charges to the account that are of higher priority than earlier transactions, or the refund of a transaction that had previously been paid.

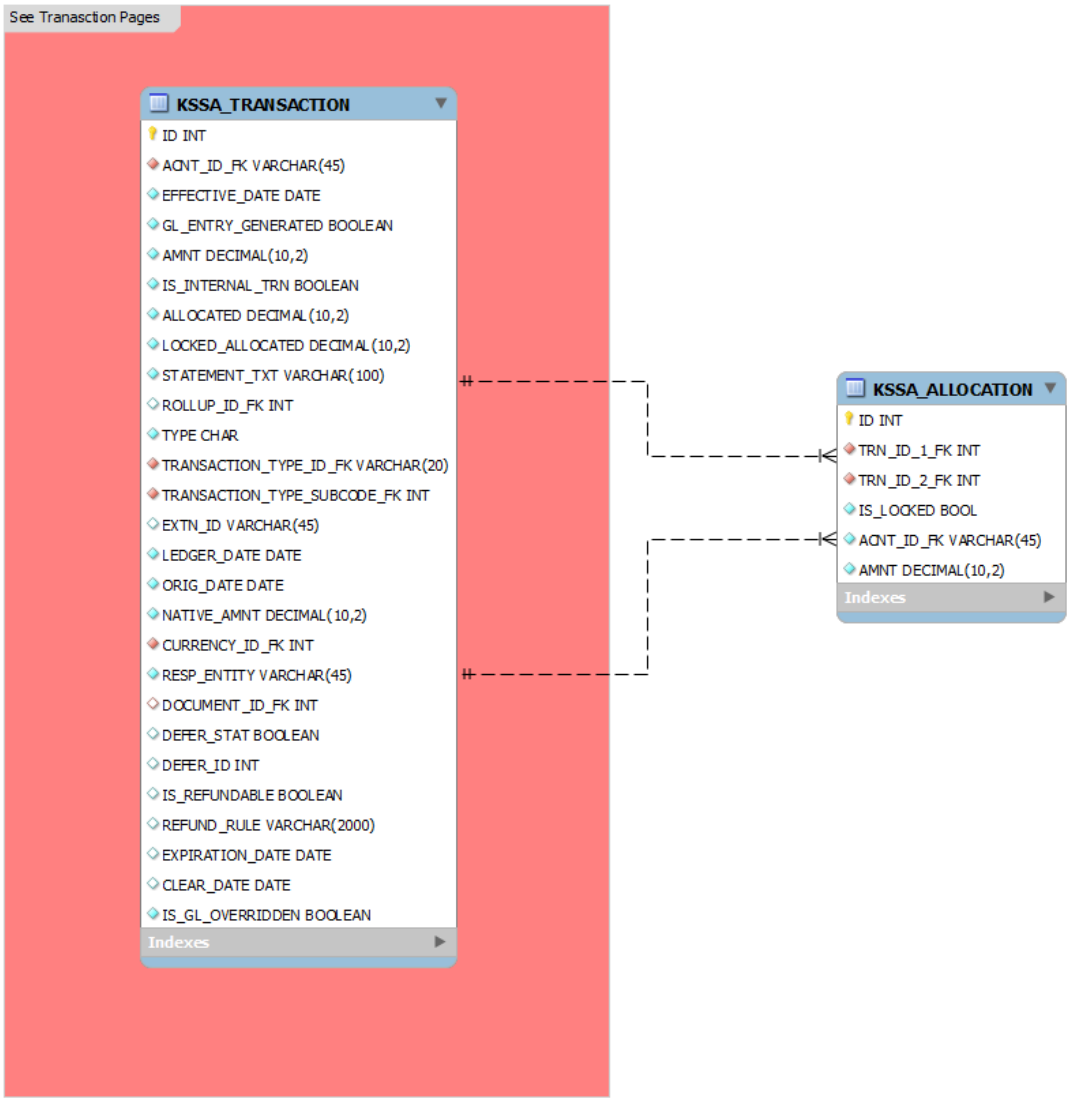

Table: ALLOCATION

(Tracks allocations between different transactions.)

ID	Autonumbered PK for the allocation.
TRN_ID_1_FK	Transaction identifier for the first transaction. This transaction is the "payer" – that is to say that the transaction identified in this key is the credit balance that will pay off a debit balance.
TRN_ID_2_FK	Transaction identifier for the second transaction. This transaction is the "paid" transaction. That is to say that the transaction identified by this key will have its debit balance a paid with funds from transaction 1.
IS_LOCKED	If this is a locked allocation (that is to say that the payment

	allocation routine is not allowed to deallocate the payment) then this value will be true.
ACNT_ID_FK	This value can be derived from either of the transactions, as allocations can only occur on a single account. However it is included to permit easier tracking of allocations to accounts. It points to the account identifier as referenced in the ACNT table.
AMNT	The amount of the allocation. This is recorded in the default system currency. There may be many allocations from a payment that are used to pay off a single charge, and many charges may be paid off by a single payment.

Information

Generic

Account information is the general data format for types of information that can be attached to an account. It supports the Information Class and its children, Memo, Flag and Alert. All information types are associated with an account, and may be optionally associated with a single transaction.

In a practical sense, a Memo is a small piece of text that can be viewed by a CSR to give more information as to why something is happening on an account. It may be generated by the system or by a user, and exists to give a human readable log of actions on the account.

A flag is a predefined, computer readable piece of information on an account. Flags can be read by humans, but can also be used as part of an automated decision making process. For example, there might be an insufficient funds flag. Based on configuration, the system may issue certain types of holds or bars, depending on such a flag.

An alert is a message placed on an account that is displayed when an account is accessed by a CSR. A practical example might be that the address on the account is found to be incorrect. In addition to putting a flag on the account to show an incorrect address, the system might also have an alert that simply informs the CSR that there is a problem with the address on the account. The CSR would then be able to make a decision as to how to proceed with a student, if presented with that information.

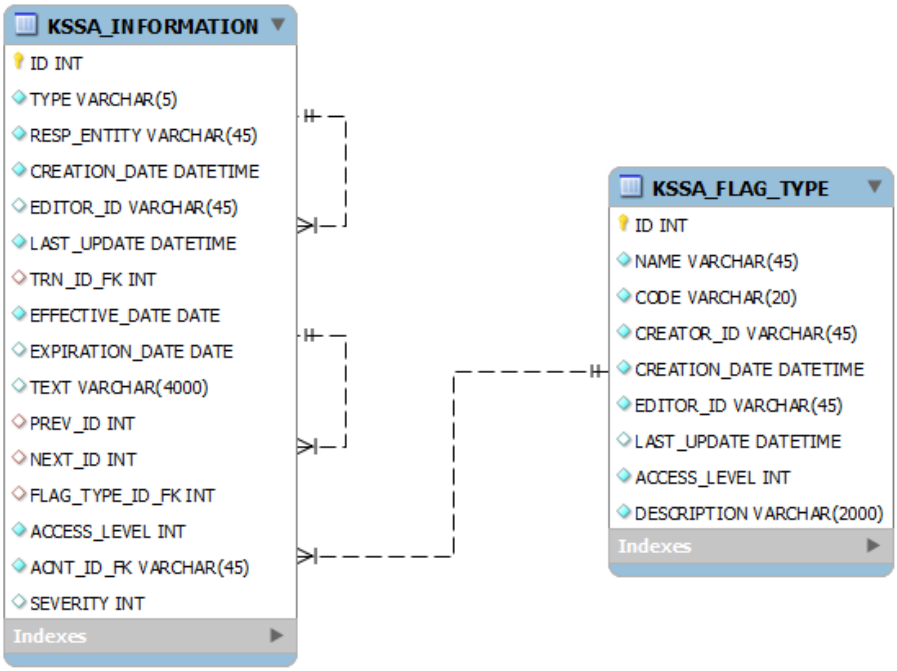

Table: INFORMATION

(Stores multiple types of information about an account or a transaction.)

ID	Autonumbered primary key for the piece of information.
TYPE	Set to IM (Information->Memo) IMF (Information->Memo->FollowUpMemo), IF (Information->Flag) or IA (Information->Alert) to define the type of class that is derived from this entry.
RESP_ENTITY	The identifier for the entity that created the account information.
CREATION_DATE	This is the date that is set when the information is saved to the system.
TRN_ID_FK	This may be populated if the memo relates specifically to a transaction. As an example, if a deferment were issued on an account, the system may prompt the user to enter a reason for the deferment, which would be then linked as a memo to the specific transaction.
EFFECTIVE_DATE	A date for the information is the date on which the memo is considered "in force", allowing information to be placed on the account that does not show until the future.
EXPIRATION_DATE	The date after which the information is considered not effective. None of these dates affect the actual existence of the account information (that is to say that an expired piece of information is not deleted) but they allow a counselor a faster view of currently applicable information, rather than having to trawl through comments that no longer make sense. A comment can be set to expire on its own (by setting

	the EXPIRATION_DATE when the memo is created) or by calling the expire() method, which will populate the EXPIRATION_DATE with the previous day's date.
TEXT	TEXT is the actual text of the memo or the alert. This is a VARCHAR2 field, limiting the size of the memo to 4000 characters. Should more space be needed, a follow-on memo could be created.
PREV_ID	A non-null value here indicates that the memo is part of a chain, and links to the memo that precedes it. The existence of this field indicates that the memo is a follow-on memo.
NEXT_ID	Indicates that a memo has had information added to it. This identifier points the system to the next memo in the chain.
FLAG_TYPE_ID_FK	This is a foreign key, linking to the identifier for a flag that is stored in the table KSSA_FLAG. Flags are entirely user defined. As well as their flag code, flags also have a human-readable friendly name.
ACCESS_LEVEL	Indicates the level of user who can view this information. All users of the KSA system have a LEVEL indicator, and anything that is their level or below is visible (for memos, flags, tags, etc.) It is envisaged that students would have a level of 0, so they could view certain flags, tags, and alerts. It is not expected that any memos would be visible to them. In the case of a flag, the flag has a level identified with it, which is loaded as the default level of the class.
ACNT_ID_FK	Identifies the account identifier against which the information is lodged.
SEVERITY	Identifies the severity of the flag. The higher this number, the more severe. The actual meaning of the severity is decided by any rule that acts upon this value.

Table: FLAG_TYPE

(Stores the attributes for the types of flags that exist in the system.)

ID	Autonumbered primary key.
NAME	A friendly name for the flag. This would be displayed to the user.
CODE	Short code to identify the flag type.
ACCESS_LEVEL	Minimum level of the user who could see this type of flag.
DESCRIPTION	Readable description of the flag with more detail than the high-level NAME field. For example "This user has bounced a check on their account. The bursar's office will not accept checks from those users who have in the past bounced a check. They should use another payment method."

Activity Data Model

Generic

KSSA_ACTIVITY persists the objects that make up the activity log, supporting the activity service. This provides a security log of activity that happens within the system.

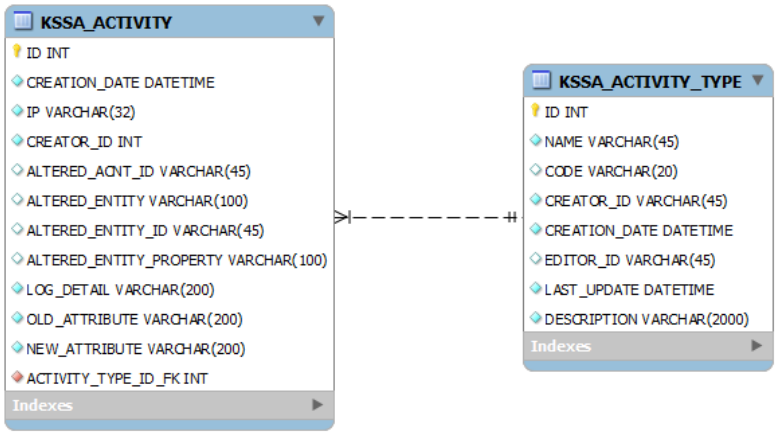

Table: ACTIVITY

(Tracks activity within the KSA system.)

ID	Autonumbered primary key for the ACTIVITY table.
CREATION_DATE	Date and time of the logged activity.
IP	IP address of the originating system that caused the activity.
MAC	Optional MAC address of the system that caused the activity.
CREATOR_ID	The account identifier for the person or system that triggered the activity.
ALTERED_ACNT_ID	The account against which the activity was logged, if applicable.
ALTERED_ENTITY	The type of entity that was altered, as derived from the code. For example “Currency” etc.
ALTERED_ENTITY_ID	In conjunction with ALTERED_ENTITY, the identifier for the specific entity that was altered/ created.
ALTERED_ENTITY_PROPERTY	The specific property that was altered by the change.
LOG_DETAIL	Readable explanation of the activity that has occurred. For example “A new credit type was created within the system.”
OLD_ATTRIBUTE	If the value of an attribute is changed during this auditable activity, then the old value is stored here. For example, if a last name is changed from “Blogs” to “Bloggs” then OLD_ATTRIBUTE would be equal to ‘Blogs’ (and NEW_ATTRIBUTE would be equal to ‘Bloggs’
NEW_ATTRIBUTE	An optional attribute that describes the activity in more detail. For example, if a new debit type, called “Bookstore Charge” was created, then the attribute would be set as “Bookstore Charge”.
ACTIVITY_TYPE_ID_FK	A classification of the problem, for example EXCEPTION, SECURITY, etc. These types are defined in the

KSSA_ACTIVITY_TYPE table. It is envisaged that the basic types of error will be predefined in the software, but their names can be configured, and other types can be added in future revisions.

Table: ACTIVITY_TYPE

(Stores the types of activities that exist in the system. While these types can be configured, it is envisioned that the KSA system will come with predefined types that can be customized (by name) but it is not, at this time, envisioned that the administrator would add new types of activities.) LEVEL The level is the level of the activity, which is a numerical value, The lower the value, the more transactional the detail, the higher, the more serious the detail. We will adopt the Apache LogLevel names for this purpose. These are emerg, alert, crit, error, warn, notice, info, debug. More information can be found at http://httpd.apache.org/docs/2.0/mod/core.html#loglevel

ID	Autonumbered primary key.
NAME	A friendly name for the activity. This would be displayed to the user.
CODE	Short code for the activity type.
DESCRIPTION	UI Description of the activity type.

Account

Generic

The KSSA_ACNT table is a central reference point that ties together a great number of tables.

KSSA_ACNT is the central account table. ID is the account identifier as is referenced throughout the rest of the system as ACNT_ID_FK. This is the identifier of a KSA account.

KSSA_ACNT and many of its related tables are used to populate the Account objects. Two key tables store the information that relates to an account. In addition to the core KSSA_ACNT table, there is also KSSA_ACNT_PROTECTED_INFO; a table of details that are considered more sensitive than some other types of data. For the sake of security, a regular user that has reporting access to the data in KSA would NOT have access to this table. This populates the AccountProtectedInformation class, which, in addition to not being loaded as default, triggers memo entries when data within that class are accessed. This table has a one-to-one relationship with ACNT, and therefore KSSA_ACNT_PROTECTED_INFO.ID = KSSA_ACNT.ID

KSSA_TRANSACTION is defined earlier in this document.

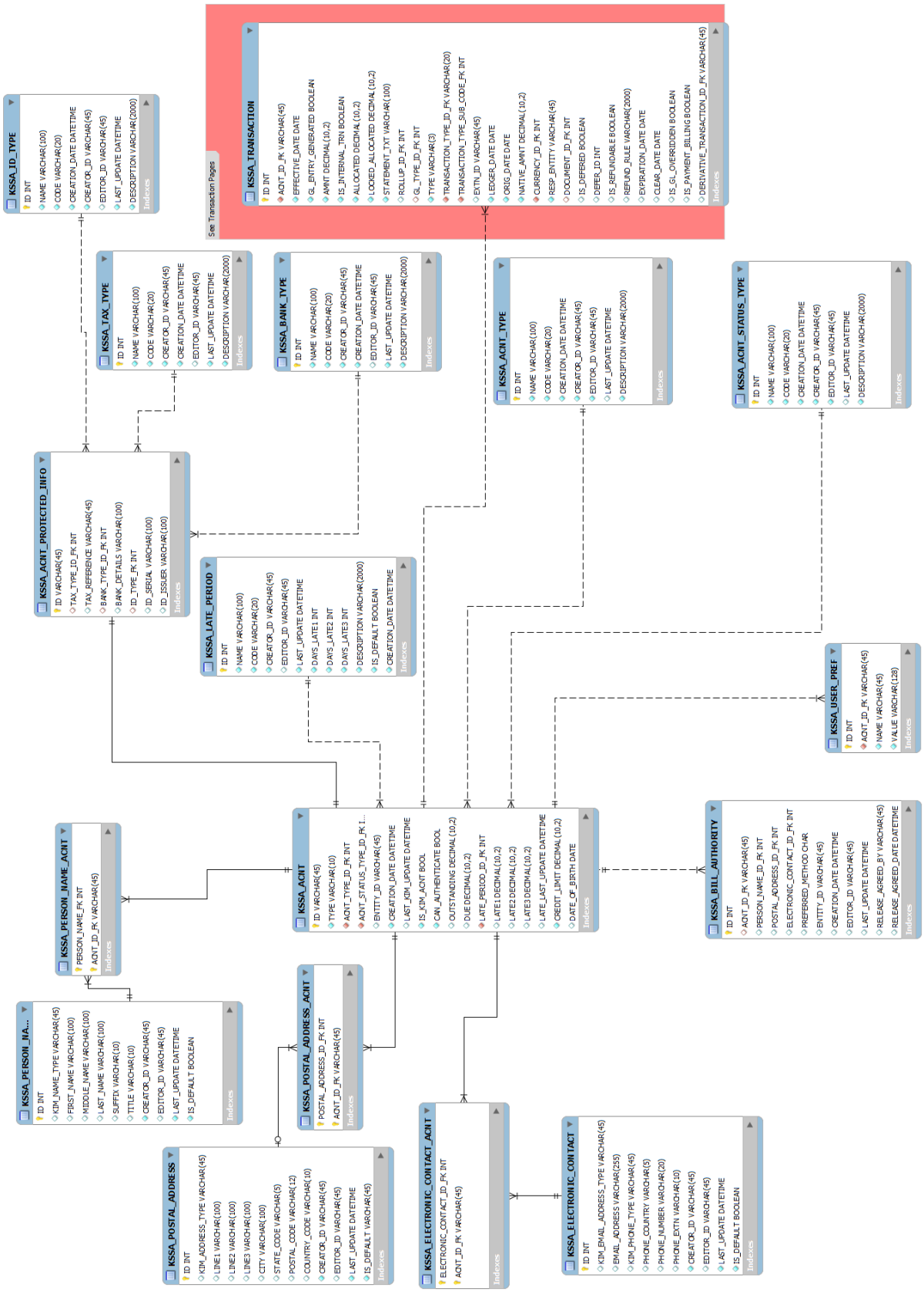
See Transaction Pages
KSSA_TRANSACTION
ID INT
ACNT_ID_FK VARCHAR(45)
EFFECTIVE_DATE DATE
GL_ENTRY_GENERATED BOOLEAN
AMNT DECIMAL(10,2)
IS_INTERNAL_TRN BOOLEAN
ALLOCATED DECIMAL(10,2)
LOCKED_ALLOCATED DECIMAL(10,2)
STATEMENT_TXT VARCHAR(100)
ROLLUP_ID_FK INT
GL_TYPE_ID_FK INT
TYPE VARCHAR(3)
TRANSACTION_TYPE_ID_FK VARCHAR(20)
TRANSACTION_TYPE_SUB_CODE_FK INT
EXTN_ID VARCHAR(45)
LEDGER_DATE DATE
ORIG_DATE DATE
NATIVE_AMNT DECIMAL(10,2)
CURRENCY_ID_FK INT
RESP_ENTITY VARCHAR(45)
DOCUMENT_ID_FK INT
IS_DEFERED BOOLEAN
DEFER_ID INT
IS_REFUNDABLE BOOLEAN
REFUND_RULE VARCHAR(2000)
EXPIRATION_DATE DATE
CLEAR_DATE DATE
IS_GL_OVERRIDDEN BOOLEAN
IS_PAYMENT_BILLING BOOLEAN
DERIVATIVE_TRANSACTION_ID_FK VARCHAR(45)
Indexes
KSSA_ID_TYPE
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATION_DATE DATETIME
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DESCRIPTION VARCHAR(2000)
Indexes
KSSA_TAX_TYPE
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATOR_ID VARCHAR(45)
CREATION_DATE DATETIME
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DESCRIPTION VARCHAR(2000)
Indexes
KSSA_BANK_TYPE
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATOR_ID VARCHAR(45)
CREATION_DATE DATETIME
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DESCRIPTION VARCHAR(2000)
Indexes
KSSA_ACNT_TYPE
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATION_DATE DATETIME
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DESCRIPTION VARCHAR(2000)
Indexes
KSSA_ACNT_STATUS_TYPE
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATION_DATE DATETIME
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DESCRIPTION VARCHAR(2000)
Indexes
KSSA_ACNT_PROTECTED_INFO
ID VARCHAR(45)
TAX_TYPE_ID_FK INT
TAX_REFERENCE VARCHAR(45)
BANK_TYPE_ID_FK INT
BANK_DETAILS VARCHAR(100)
ID_TYPE_FK INT
ID_SERIAL VARCHAR(100)
ID_ISSUER VARCHAR(100)
Indexes
KSSA_LATE_PERIOD
ID INT
NAME VARCHAR(100)
CODE VARCHAR(20)
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
DAYS_LATE1 INT
DAYS_LATE2 INT
DAYS_LATE3 INT
DESCRIPTION VARCHAR(2000)
IS_DEFAULT BOOLEAN
CREATION_DATE DATETIME
Indexes
KSSA_USER_PREF
ID INT
ACNT_ID_FK VARCHAR(45)
NAME VARCHAR(45)
VALUE VARCHAR(128)
Indexes
KSSA_PERSON_NAME_ACNT
ID INT
PERSON_NAME_FK INT
ACNT_ID_FK VARCHAR(45)
Indexes
KSSA_ACNT
ID VARCHAR(45)
TYPE VARCHAR(10)
ACNT_TYPE_ID_FK INT
ACNT_STATUS_TYPE_ID_FK I...
ENTITY_ID VARCHAR(45)
CREATION_DATE DATETIME
LAST_KIM_UPDATE DATETIME
IS_KIM_ACNT BOOL
CAN_AUTHENTICATE BOOL
OUTSTANDING DECIMAL(10,2)
DUE DECIMAL(10,2)
LATE_PERIOD_ID_FK INT
LATE1 DECIMAL(10,2)
LATE2 DECIMAL(10,2)
LATE3 DECIMAL(10,2)
LATE_LAST_UPDATE DATETIME
CREDIT_LIMIT DECIMAL(10,2)
DATE_OF_BIRTH DATE
Indexes
KSSA_BILL_AUTHORITY
ID INT
ACNT_ID_FK VARCHAR(45)
PERSON_NAME_ID_FK INT
POSTAL_ADDRESS_ID_FK INT
ELECTRONIC_CONTACT_ID_FK INT
PREFERRED_METHOD CHAR
ENTITY_ID VARCHAR(45)
CREATION_DATE DATETIME
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
RELEASE_AGREED_BY VARCHAR(45)
RELEASE_AGREED_DATE DATETIME
Indexes
KSSA_PERSON_NA...
ID INT
KIM_NAME_TYPE VARCHAR(45)
FIRST_NAME VARCHAR(100)
MIDDLE_NAME VARCHAR(100)
LAST_NAME VARCHAR(100)
SUFFIX VARCHAR(10)
TITLE VARCHAR(10)
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
IS_DEFAULT BOOLEAN
Indexes
KSSA_POSTAL_ADDRESS_ACNT
POSTAL_ADDRESS_ID_FK INT
ACNT_ID_FK VARCHAR(45)
Indexes
KSSA_POSTAL_ADDRESS
ID INT
KIM_ADDRESS_TYPE VARCHAR(45)
LINE1 VARCHAR(100)
LINE2 VARCHAR(100)
LINE3 VARCHAR(100)
CITY VARCHAR(100)
STATE_CODE VARCHAR(5)
POSTAL_CODE VARCHAR(12)
COUNTRY_CODE VARCHAR(10)
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
IS_DEFAULT VARCHAR(45)
Indexes
KSSA_ELECTRONIC_CONTACT_ACNT
ELECTRONIC_CONTACT_ID_FK INT
ACNT_ID_FK VARCHAR(45)
Indexes
KSSA_ELECTRONIC_CONTACT
ID INT
KIM_EMAIL_ADDRESS_TYPE VARCHAR(45)
EMAIL_ADDRESS VARCHAR(255)
KIM_PHONE_TYPE VARCHAR(45)
PHONE_COUNTRY VARCHAR(5)
PHONE_NUMBER VARCHAR(20)
PHONE_EXTN VARCHAR(10)
CREATOR_ID VARCHAR(45)
EDITOR_ID VARCHAR(45)
LAST_UPDATE DATETIME
IS_DEFAULT BOOLEAN
Indexes

Table: ACNT
(Tracks the fundamentals of an account within the KSA system.)

ID	The account identifier. It is referenced throughout the rest of the system as ACNT_ID_FK. Under normal circumstances, this will be the user id/ net id, as derived from KIM, for adhoc accounts, it will be either generated by the system, or by the creator.
TYPE	Used to disambiguate the classes that can be produced from these tables. Possible values are AND for Account->NonChargeable->Delegate, ACD for Account->Chargeable->DirectChargeAccount and ACS for Account->Chargeable->SponsorAccount. Friendly versions of these names for UI purposes can be queried in the ACNT_TYPE table, searching on the CODE field.
ACNT_STATUS_TYPE_ID_FK	References the ACNT_STATUS_TYPE table, referencing the status of the account. The account status is a single value that, can be used to impose certain limits on the account. This is a high-level status of the account, and is used in combination with other account indicators, such as flags.
ENTITY_ID	If the account is derived from a KIM account, then the KIM identifier is stored here. This is often referred to as the NetID of the user, and will often follow the format of first initial/last name. If the entity is not a KIM identity, then the identifier from the other system will be used here, and the account will be set as IS_KIM_ACNT=false.
CREATION_DATE	The date that the account was established within the KSA system.
LAST_KIM_UPDATE	If the account is a KIM account, LAST_KIM_UPDATE is the date and time that the details were validated with the KIM datastore.
IS_KIM_ACNT	Boolean that answers the question “did this account originate from the KIM system?” If true, then the KIM fields are implied. If false, then the KIM fields will be null.
CAN_AUTHENTICATE	Boolean that answers the question “can this user authenticate into the KSA system?”
OUTSTANDING	The balance of the account, in the system currency that is outstanding on the account. This includes any amounts that are not yet due.
DUE	The balance of the account that is due at this moment in time. That is, all the transactions which have an effectiveDate of today or before.
LATE_PERIOD_ID_FK	LATE_PERIOD_ID_FK references the late payment table. This permits the system to have configurable groups of late payment “buckets”. For example, a normal student might be considered “late” once their balance has been due for 30

	days, and then they are progressively later at 60 and 90 days. Whereas for some customers (maybe a sponsoring employer) there may be an agreement that the account is not past due until 60/90/120 days.
LATE1..3	The aged balance in "buckets" according to the LATE_PERIOD table.
LAST_LATE_UPDATE	Date and time when the account last went through the ageing process.
CREDIT_LIMIT	Credit limit for the account. Enforcement of this limit is during the transaction creation process.
DATE_OF_BIRTH	DirectCharge accounts only, the date of birth of the "student" or other account holder. Stored as a disambiguator.

Table: ACNT_PROTECTED_INFO

(Tracks types of data that are required for system functioning that are considered more sensitive than other types of data.)

ID	The account identifier. It is referenced throughout the rest of the system as ACNT_ID_FK. This is the identifier of a KSA (not a KIM) account.
TAX_TYPE_ID_FK	A foreign key to the TAX_TYPE table. This is a configurable list of types of tax identifiers that the system might accept. If the system is deployed in the US, a TAX_TYPE might be "Social Security Number".
TAX_REFERENCE	The actual tax identifier as referenced in TAX_TYPE.
BANK_TYPE_ID_FK	A foreign key to the bank type table, permitting the storage of different types of bank information. For example, an ACH type used in the US would require a routing number, and account number, and an account type (checking, savings, etc.) An IBAN type would store the information differently.
BANK_DETAILS	The actual detail as references in BANK_TYPE, for example, the actual IBAN of the account holder.
ID_TYPE_FK	Foreign key to the ID_TYPE table that defines the different types of identity documentation that are accepted.
ID_SERIAL	The actual serial number of the document defined by ID_TYPE. For example, if the document is a passport, this would likely be the passport's number.
ID_ISSUER	The issuing authority of a document. For a passport, this would often be a country. For a US Driver's license, it would most often be the state that issued the license.

Table: BANK_TYPE

(A simple type table that defines the different types of bank information that might be stored in the KSA system.)

ID	Autonumbered primary key.
NAME	Friendly name for the type of bank information. For example "ACH" or "IBAN".
CODE	Short code to reference the bank type.
DESCRIPTION	A longer description of the expected value in the field.

Table: TAX_TYPE

(A simple type table that defines the different types of tax information that might be stored in the KSA system.)

ID	Autonumbered primary key.
NAME	Friendly name for the tax type identifier. For example "U.S. Social Security Number" or "British National Insurance Number".
CODE	Short code to reference the tax type.
DESCRIPTION	A longer description of the tax number type to assist operators in understanding where this information may come from, expected format, etc.

Table: ID_TYPE

(A simple type table that defines the different types of identity information that might be stored in the KSA system.)

ID	Autonumbered primary key.
NAME	Friendly name for the tax type identifier. For example "US Driver's license", "French Passport"
CODE	Short code to reference the ID Type.
DESCRIPTION	A longer description of the identification number type to assist operators in understanding where this information may come from, expected format, etc.

Table: PERSON_NAME_ACNT

(Simple reference table to link names to accounts)

PERSON_NAME_ID_FK	Foreign key identifying a single name in the PERSON_NAME table.
ACNT_ID_FK	Foreign key identifying a single account in the ACNT table.

Table: PERSON_NAME

(Tracks a person's name. This structure closely mimics the KIM name standard, and is stored in its own table to permit changes that may occur in the future. This format might be problematic for internationalization.)

ID	Name identifier.
KIM_NAME_TYPE	If the name is derived from KIM, the name type is stored here. This permits us to update the name from KIM by retrieving the correct name record if a student has more than one registered name. If the field is null, then the name is only stored within KSA.
FIRST_NAME	First name of the person.
MIDDLE_NAME	Middle name of the person.
LAST_NAME	Last name of the person.
SUFFIX	Freeform suffix, allowing for appended titles or generational information.
TITLE	Freeform prefix, allowing for titles that come before the name.
IS_DEFAULT	Answers the question "is this the default name used on the account"?

Table: POSTAL_ADDRESS_ACNT

(Simple reference table to link addresses to accounts)

POSTAL_ADDRESS_ID_FK	Foreign key identifying a single address in the POSTAL_ADDRESS table.
ACNT_ID_FK	Foreign key identifying a single account in the ACNT table.

Table: POSTAL_ADDRESS

(Tracks a postal address. This structure closely mimics the KIM name standard, and is stored in its own table to permit changes that may occur in the future. In particular, this format may be a problem for internationalization.)

ID	Autonumbered primary key
KIM_ADDRESS_TYPE	If the address is derived from KIM, then this is set to the address type. If it is null, then the address is stored locally only, in KSA.
LINE1..3	Lines 1 through 3 of the address.
CITY	City part of the address. This should be interpreted openly as the locality name.
STATE_CODE	If the country has states or other major localities as part of its addresses, then they are stored here.
POSTAL_CODE	If the address has a coded address, it is stored here.

COUNTRY_CODE	Code for the country of the address.
IS_DEFAULT	Answers the question "is this the default address on the account?"

Table: ELECTRONIC_CONTACT_ACNT
(Simple reference table to link names to accounts)

ELECTRONIC_CONTACT_ID_FK	Foreign key identifying a single set of electronic contact information in the ELECTRONIC_CONTACT table.
ACNT_ID_FK	Foreign key identifying a single account in the ACNT table.

Table: ELECTRONIC_CONTACT
(Tracks the electronic contact information for an account. Closely mimics KIM information.)

ID	Autonumbered primary key
KIM_EMAIL_ADDRESS_TYPE	If the email address is derived from KIM, then this is set to the address type. If it is null, then the email address is stored locally only, in KSA.
EMAIL_ADDRESS	The actual email address.
KIM_PHONE_TYPE	If the phone number is derived from KIM, then this is set to the phone type. If it is null, then the number only exists within KSA.
PHONE_COUNTRY	The country code for the number. In the absence of a precedent on the use of this field, and the constraints of the field, we will store an ISO3166 country code.
PHONE_NUMBER	The major numerical part of the number.
PHONE_EXTN	The extension part of the number, if applicable.
IS_DEFAULT	Answers the question "is this the default contact information used on the account"?

Table: ACNT_STATUS_TYPE
(Storage for the different account statuses that can be applied to an account.)

ID	Autonumbered primary key.
NAME	Friendly name for the account status. For example "In good standing".
CODE	Short code describing the status type.
DESCRIPTION	A longer description of what it means to be in that status.

Table: ACNT _TYPE
(Storage for the different account types that can be applied to an account.)

ID	Autonumbered primary key.
NAME	Friendly name for the account status. For example "Freshman Account".
CODE	Short code describing the account type.
DESCRIPTION	A longer description of what it means be of that type.

Table: LATE_PERIOD

(Stores the different period definitions against which an account might be aged.)

ID	Autonumbered primary key.
NAME	The name of the late period definition as displayed to the user.
DAYS_LATE1..3	The number of days after the effective date of a transaction that the account is considered to be in the appropriate late bucket. The standard reference model would be LATE1=30, LATE2=60, LATE3=90.
DESCRIPTION	A more verbose description of the late model, giving a representative better information as to what types of accounts this might be the appropriate late period definition for.
IS_DEFAULT	Boolean that answers the question "is this the default late period definition?"

Table: USER_PREF

(Stores simple key/value pairs for an account permitting the storage of simple preferences.)

ID	Autonumbered primary key.
ACNT_ID_FK	Reference to the account to which this key/value pair belongs.
NAME	Name of the key.
VALUE	Value of the key.

Table: BILL_AUTHORITY

(Storage for persons who are to receive a copy of the billing statement, in addition to the student themselves (as defined by the default name and address on the account).

ID	Autonumbered primary key.
ACNT_ID_FK	Account to which this record belongs.
PERSON_NAME_ID_FK	Identifier of the name record to be used on the bill.
POSTAL_ADDRESS_ID_FK	Identifier of the address record to be used on the bill.

ELECTRONIC_CONTACT_ID_FK	Identifier of the electronic contact record to be used on the bill.
PREFERRED_METHOD	Currently can be M(ail) or E(mail).
ENTITY_ID	Entity identifier of the entity that created this entry. This ought to be the entity identifier for the account holder references in ACNT_ID_FK
CREATION_DATE	Date when this entry was created.
EDITOR_ID	Identifier of the user who altered this record.
LAST_UPDATE	Last recorded date that the record was changed.
RELEASE_AGREED_BY	Account identifier of the user who agreed to release this record to another person. Ought to be the same as ACNT_ID_FK
RELEASE_AGREED_DATE	Date and time stamp of the time when the user agreed to allow these records to be released to another user.

Account Block Type

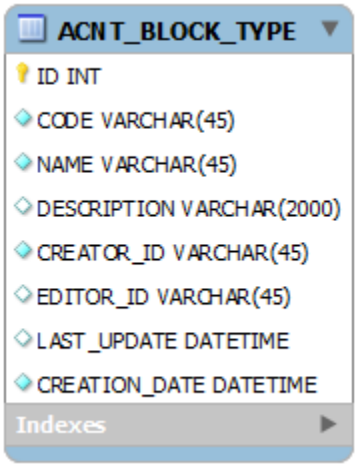

Table: ACNT_BLOCK_TYPE

(Auditable entity table to store account blocks available in KSA.

ID	Autonumbered primary key.
CODE	Short code used to identify the block.
NAME	Name of the block.
DESCRIPTION	Description of the block to give the user more information as to what effect the block has.
CREATOR_ID	Entity identifier of the entity that created this entry. This ought to be the entity identifier for the account holder references in ACNT_ID_FK
CREATION_DATE	Date when this entry was created.
EDITOR_ID	Identifier of the user who altered this record.
LAST_UPDATE	Last recorded date that the record was changed.

CREATION_DATE	Timestamp when the entry was created.

UI Pack (Localization of UI – Language)

Generic

The UI Pack tables store the UI strings for different languages to be used in KSA. By default, KSA supports the use of XLIFF localization files for UI text.

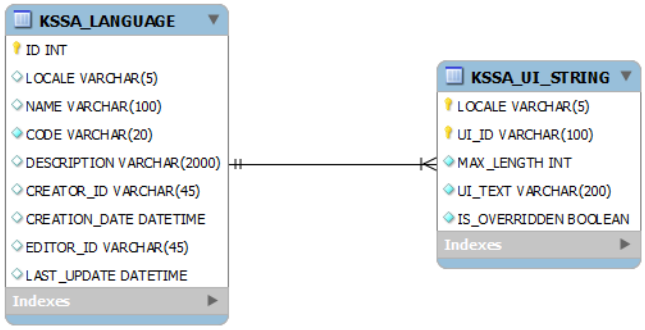

Table: UI_STRING

A simple lookup table that allows the UI to lookup the appropriate text for identifiers on the screen.

LOCALE	Locale, stored in standard format as defined in ksa.properties (ex. fr_FR, en_US).
UI_ID	Identifier as used on the UI.
MAX_LENGTH	Maximum length of the string to be displayed on the UI. This is a requirement of the XLIFF standard.
UI_TEXT	The actual text to be displayed on the UI in place of the UI_ID.
IS_OVERRIDEN	By default, this is false, however, if the string is edited outside of a common language pack (i.e. the institution changes the text for their own implementation) then this is set to true. This is used when deciding whether or not to overwrite a string when loading a new language pack.

Table: LANGUAGE

A simple dictionary service that stores the languages on the system.

ID	System generated primary key.
LOCALE	Locale, stored in standard format as defined in ksa.properties (ex. fr_FR, en_US).
CODE	Short code for the locale.

NAME	Name of the language in readable form. (For Example: French (France).)
DESCRIPTION	If an extended description is warranted in the UI, it would be placed here. This is likely to be blank for most languages.

General Ledger Processing (In Progress)

Generic

These tables record the general ledger transactions that are stored and then transmitted to the general ledger. When a transaction becomes effective, any general ledger transactions relating to that transaction are recorded in the GL_TRANSACTION table. Depending on configuration, these may then be moved directly to the transmission table, or this process might be done in batch mode.

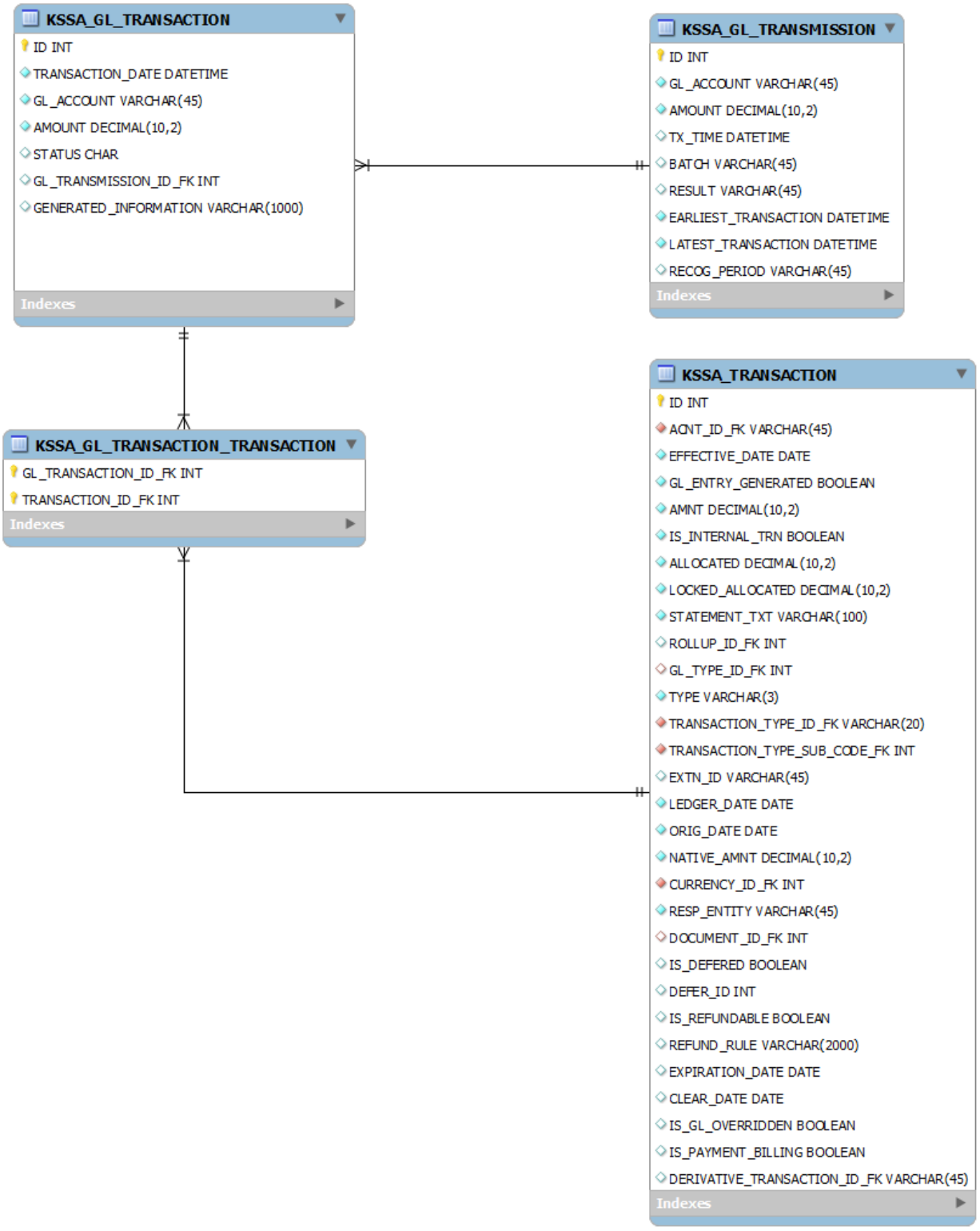

Table: GL_TRANSACTION

This table stores individual transactions to the general

ledger. Any transaction that creates a general ledger transaction may have one or more entries in this table.

ID	Autonumbered primary key for the transaction.
TRANSACTION_DATE	Date the GL transaction was generated.
GL_ACCOUNT	The general ledger account number that this transaction will affect.
AMOUNT	The value of the transaction in the system currency.
STATUS	W- Waiting transaction. This will not be sent to the general ledger until the status is set to Queued. Q – queued, transaction is ready to be transmitted. P – In process, this transaction is currently being processed and made into a transmission. C – Complete, this transaction has been entered into the GL_TRANSMISSION queue. F- Failed.
GL_TRANSMISSION_ID_FK	Identifier for the transmission that this transaction is part of. This is only set once the transaction has a status of C(omplete)
GENERATED_INFORMATION	String of text that explains where the general ledger transmissions were generated.

Table: GL_TRANSMISSION

This table stores general ledger transactions that are to be/have been transmitted to the general ledger. Depending on configuration, these entries may be individual transactions (real time entries), batch transmissions, or rolled-up batch transmissions.

ID	Autonumbered primary key for the transmission.
GL_ACCOUNT	The general ledger account that is affected.
AMOUNT	Amount of the transmission in the system currency.
TX_TIME	Time and date when the transmission was made to the general ledger.
BATCH	Batch identifier. In real time, each transmission will be viewed as a "batch of one".
RESULT	If the general ledger returns a result, it will be stored here.
EARLIEST_TRANSACTION	In a transaction rollup, this is the earliest transaction date in that rollup. For non-rolled up transactions, this will be the effective date of the original transaction.
LATEST_DATE	In a transaction rollup, this is the latest transaction date in that rollup. For non-rolled up transactions, this will be the effective date of the original transaction.
RECOG_PERIOD	This field can be filled using the period recognition service of the general ledger services. It is a user-identified period of time during which the revenue was recognized.

Table: GL_TRANSACTION_TRANSACTION

Simple association table to allow multiple transactions to be able to reference a single general ledger transaction.

GL_TRANSACTION_ID_FK	Reference identifier for the general ledger transaction.
TRANSACTION_ID_FK	Reference identifier for the KSA transaction.

Batch Transaction Processing

Generic

KSA can accept transaction uploads of batches, following the XML schema provided. The system will keep track of the batches and the results of the batches so that the system can be reconciled with external systems.

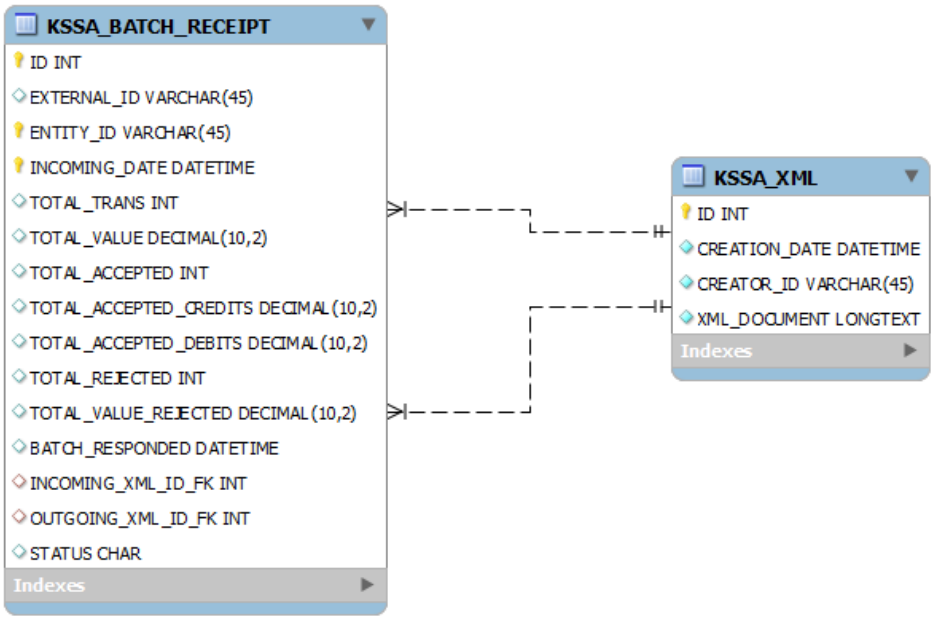

Table: BATCH_RECEIPT

Stores the "receipt" for each batch.

ID	Autonumbered primary key for the transmission.
EXTERNAL_ID	External identifier from the batch. This is in the XML upload, and will be generated by the external system.
ENTITY_ID	User or system that uploaded the transactions.
INCOMING_DATE	When the batch goes through initial verification, the datetime stamp of this action will be stored here.

TOTAL_TRANS	The number of total discrete transactions in the batch.
TOTAL_VALUE	The total value of all transactions within the batch.
TOTAL_ACCEPTED	The total number of accepted transactions in the batch.
TOTAL_ACCEPTED_CREDITS	The total value of all accepted credits.
TOTAL_ACCEPTED_DEBITS	The total value of all accepted debits.
TOTAL_REJECTED	The number of transactions that were rejected.
TOTAL_VALUE_REJECTED	The total value of all rejected transactions.
BATCH_RESPONDED	When the system sends its reply to the external system, the datestamp is recorded here.
INCOMING_XML_ID_FK	Pointer to XML table for the incoming XML message.
OUTGOING_XML_ID_FK	Pointer to XML table for the outgoing XML message.
STATUS	Status of the batch. Q – In Process ("queued") F – Failed – The entire batch was rejected. This means that the batch can be resent (after corrections) with the same batch number. P – Partial, at least one of the transactions was added to the system. The batch number associated with this batch cannot be reused. E – Entire, the whole batch was accepted. The batch number cannot be reused.

Table: XML

Simple storage of XML documents.

ID	Autonumbered primary key for the XML document.
CREATION_DATE	Date that the document was stored by KSA.
CREATOR_ID	Identification of the user of system that sent the message.
XML_DOCUMENT	CLOB storing the actual XML document.

Refund Processing (In Progress)

Generic

Refund processing is the job of KSA-RM-RM (Receivable Management – Refund Management). As with general ledger processing, processing of refunds can be a batch or an individual job. This allows schools to do large payment runs, as well as "on demand" payments when needed.

As a general principle, refunds are first entered into the REFUND table, where they are checked, before interaction takes place to actually create the refund.

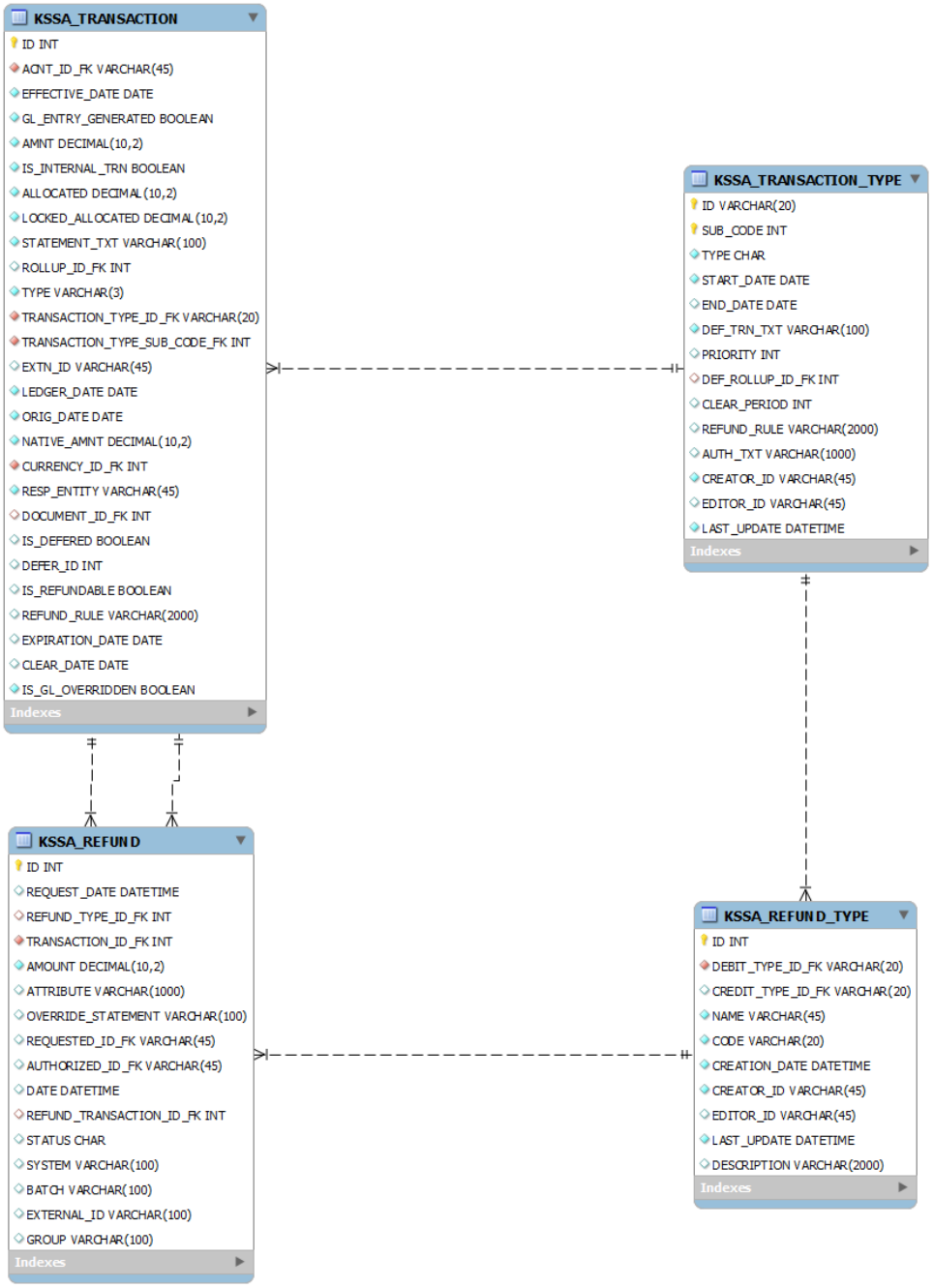

Table: REFUND

ID	Autonumbered primary key.
REQUEST_DATE	Date and time when the refund request was made.
TRANSACTION_ID_FK	The transaction identifier for the transaction to be refunded.
AMOUNT	The amount of the refund in the system currency.
REFUND_TYPE_FK	Foreign key reference to REFUND_TYPE, indicating “how” a

	refund might be executed. For example, there might be a refund type for “paper check”, “ACH”, “Credit Card”, etc.
ATTRIBUTE	If a refund has a special attribute, it will be stored here. As an example, for an ACH refund, the account to which a transaction is refunded might be stored here.
OVERRIDE_STATEMENT	When an account refund is made (where the actual refund credit is sent to another KSA account) the statement text will by default, use the statement text as defined in the transaction type stored in the next two attributes. The system can override this text by changing it in this field. An example use case would be where a refund is made to a sponsor account, and the school wants the statement to clarify which student account the refund came from.
REQUESTED_ID_FK	The user id of the person or system that requested the refund.
AUTHORIZED_ID_FK	The user if of the person or system that authorized the refund. This will be updated when the refund becomes verified in the next column.
DATE	Date of the actual refund. This will match the date of the refund transaction.
REFUND_TRANSACTION_ID_FK	The identifier of the generated refund transaction.
STATUS	The status of the refund. It can be (U)nverified, (V)erified or (R)efunded, (C)anceled (reversed) or (F)ailed.
SYSTEM	Name of the system that actually created the refund. This is used to assist a CSR in tracing a refund.
BATCH	If the refunds are run under a batch identifier, it will be stored here.
EXTERNAL_ID	If the external system creates an ID for the refund (authorization for a credit card refund, for example) it can be stored here.
GROUP	UUID used to batch together groups of refund into a single transaction.

Table: REFUND_TYPE

ID	Autonumbered primary key.
CODE	Short code for the refund type.
DEBIT_TYPE_ID_FK	When a refund is created, it is issued as a debit (charge) on the account. This is the transaction type for the debit that will be made.
CREDIT_TYPE_ID_FK	When an account refund is made, this is the credit type that is applied to the receiving account.
NAME	Friendly name of the refund type. For example “Paper Check”, “Check to be collected at the office”, “ACH Transfer” etc.
DESCRIPTION	Longer description for the type to be displayed in the UI

upon inspection.

Cash Limit Tracking (In Progress)

Generic

Cash-limit tracking is designed to address the requirements of U.S. schools to file the IRS form 8300, but it is designed to be reasonably expandable to allow for use in other areas with similar regulations.

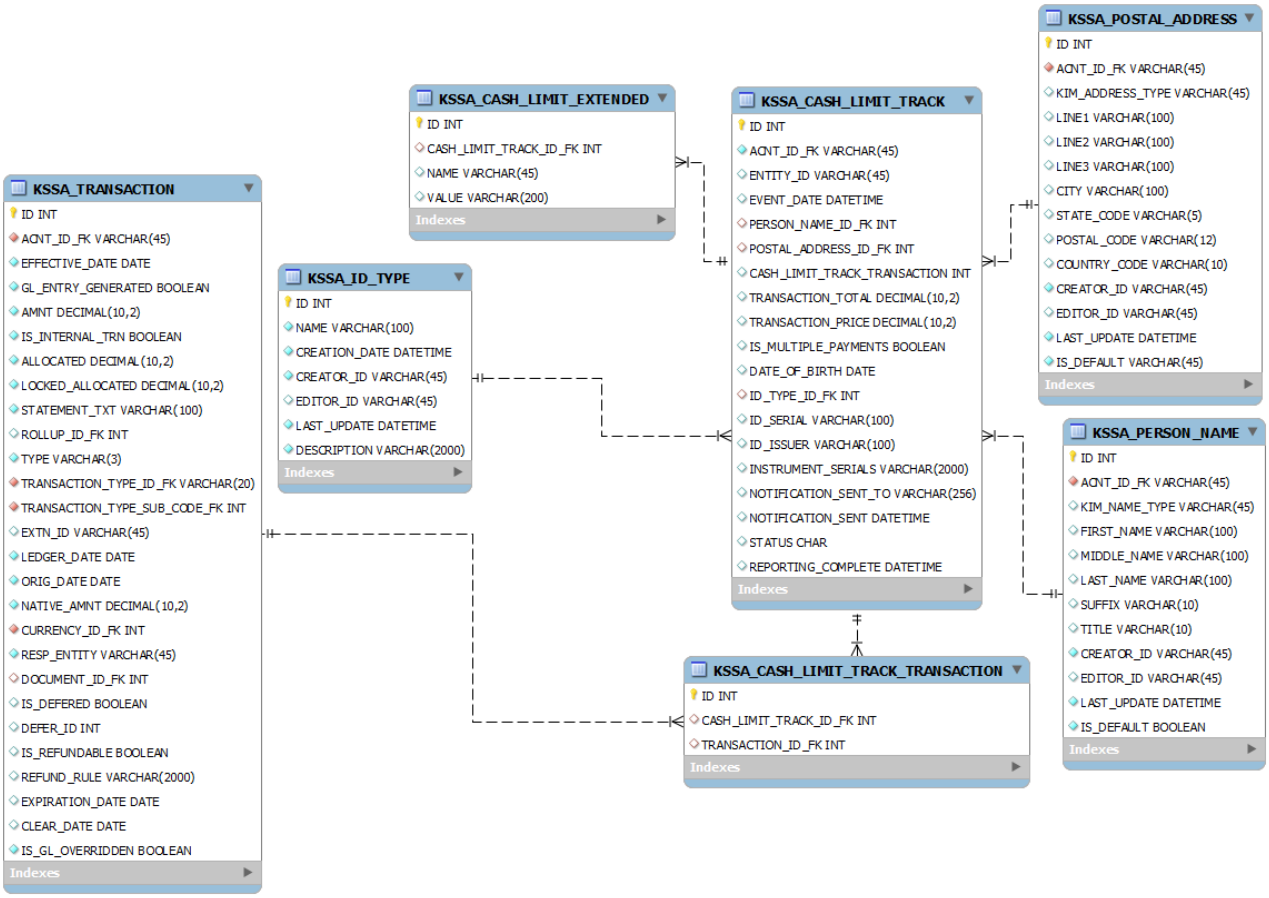

Table: CASH_LIMIT_TRACK

Core table of the cash-limit tracking system, recording a cash-limit event.

ID	Autonumbered primary key.
ACNT_ID_FK	Account to which this event refers.
ENTITY_ID	The user id of the person who triggered this event.
EVENT_DATE	Date and time of the event (i.e. the transaction that triggered the event).
PERSON_NAME_ID_FK	Pointer to the name on the account. Note that on creation of the event, a new name and address field is created, therefore

	this reference will not be equal to the NAME reference in the ACNT table.
POSTAL_ADDRESS_FK_ID	Pointer to the address on the account. As with the name, a new entry is created when an 8300 is generated, so this will not be the same pointer that can be addressed through the ACNT table.
CASH_LIMIT_TRACK_TRANSACTION	Pointer to the many-to-many relationship that exists between this table and the transaction table.
TRANSACTION_TOTAL	The total of all the transactions reference in this event. Note that the rules engine may have to lower this amount, due to IRS reporting rules.
TRANSACTION_PRICE	Total price of the transaction being paid. This is set via the rules engine.
IS_MULTIPLE_PAYMENTS	If there is more than one transaction referenced in this event, this will be set to TRUE. Otherwise, it will be FALSE.
DATE_OF_BIRTH	Birthdate of the person referenced in ACNT_ID_FK. This is duplicated data to ensure the report cannot change.
ID_TYPE_ID_FK	Type of identification provided on the account.
ID_SERIAL	Serial number of the identification. For example, passport number.
ID_ISSUER	Name of the issuing authority of the identification, for example “Canada”.
INSTRUMENT_SERIALS	Serial numbers of known monetary instruments.
NOTIFICATION_SENT_TO	Email address of the person to whom details of this event were sent.
NOTIFICATION_SENT	Date and time when the notification was sent.
STATUS	Status of the transaction. (Q)ueued, (C)omplete, (I)gnored. See the process flows for when an event might be ignored.
REPORTING_COMPLETE	Date and time when the reporting (the 8300) was completed.

Table: CASH_LIMIT_EXTENDED

Simple key/value pair table for extended attributes of the cash/limit tracking system. Most of these values will be set through the rules system.

ID	Autonumbered primary key.
CASH_LIMIT_TRACK_ID_FK	Link to the cash-limit event.
NAME	Name of the value.
VALUE	Associated value.

Table: CASH_LIMIT_TRACK_TRANSACTION

Simple association table.

ID	Autonumbered primary key.

CASH_LIMIT_TRACK_ID_FK	Link to the cash-limit event.
TRANSACTION_ID_FK	Link to the transaction.

Access Control (in process)

Generic

Access control is derived from the KIM databases through the services supplied in Rice. KSA extends this functionality a little to allow roles to also translate to access to certain transaction types. In this way, transaction types, or more generally, groups of transaction types can be limited to certain roles on the system.

The only persisted entity of Access Control is this roles/transaction type permission table. All other Access Control classes within KSA are non-persisted, as they rely on KIM databases.

This is a standard KSA auditable entity.

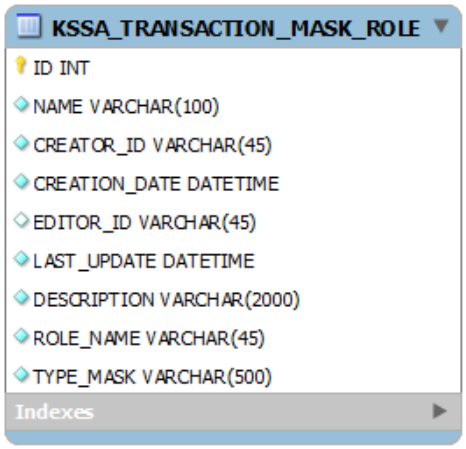

Table: TRANSACTION_MASK_ROLE

This table relates KIM roles to KSA transaction types, or transaction type masks.

ID	Autonumbered primary key.
NAME	Name of the role/relationship.
DESCRIPTIONS	UI-based description of the relationship.
ROLE_NAME	The identifier of the KIM role that is implicated in this relationship.
TYPE_MASK	Mask or actual transaction code. This is a Java regular expression that will be matched to the transaction codes (TRANSACTION_TYPE) within the KSA system. Any user with a role of (KIM_ROLE_ID) can use all transaction codes that match with the pattern in this field.

External Statements

Generic

The external statement table keeps track of the location of statements generated for the student by an external system. This permits “point-in-time” snapshot statements to be referenced by the system.

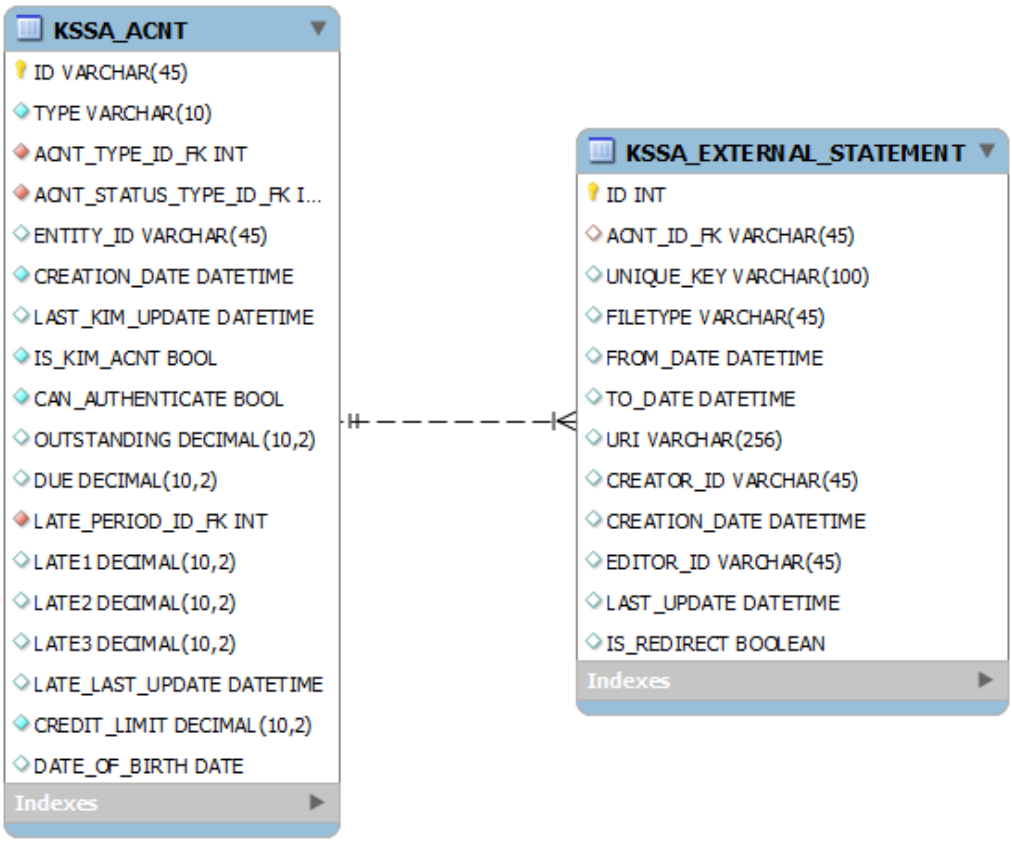

Table: EXTERNAL_STATEMENT

Maps locations where externally generated statements are stored, to allow user access to the statements.

ID	Autonumbered primary key.
ACNT_ID_FK	Link to the account against which the statement was generated.
UNIQUE_KEY	A unique identifier for the statement generated in such a way that it would not be possible to guess other numbers in the system. This can be used to permit access to the statements without authentication. Access to such statements could be removed through configuration.
FILETYPE	Mime type for the file (for example application/pdf)
FROM_DATE	Date of the start of the statement period.
TO_DATE	Date of the end of the statement period.
URI	Resource locator for the statement. It is expected that KSA

	will have the privilege to be able to access this file.
IS_REDIRECT	Boolean that tells the system whether the URI should act as a redirect, or if the system should fetch the file at the location and send it to the user.

System Preferences

Generic

System preferences, like user preferences, are a simple key/pair value system that allows the system to store system-wide values that affect system operation. The names and values of standard keypairs can be found in the *System-wide Configuration Options* document.

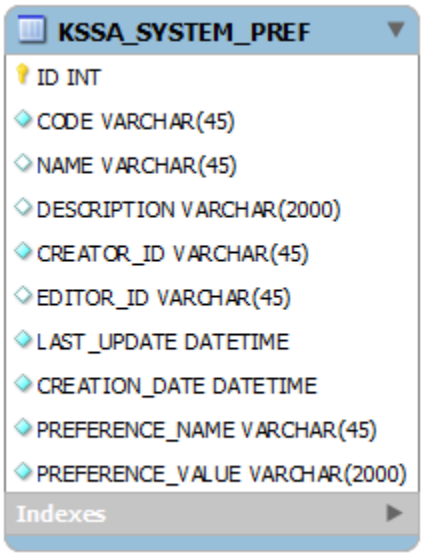

Table: SYSTEM_PREFERENCE
Simple key-pair value table for storing system-wide preferences.

ID	Autonumbered primary key.
CODE	Lookup code for the preference. Not used.
NAME	Name for the preference.
DESCRIPTION	Longer description for the preference.
PREFERENCE_NAME	Actual lookup name for the preference.
PREFERENCE_VALUE	Value of the preference.

IRS Form 1098T

Generic

The 1098T is a federal tax form required in the United States. KSA, when correctly configured, can produce an XML representation of the 1098T which can support the actual production of the paperwork that has to be filed with the IRS and passed on to the student. The system also keeps a partial record of the 1098T to permit the re-evaluation of records in the future. It is not designed to be able to produce a new copy of the 1098T, as the student's social security number is generally suppressed in this table. This is controlled in the method call, via ssnMask.

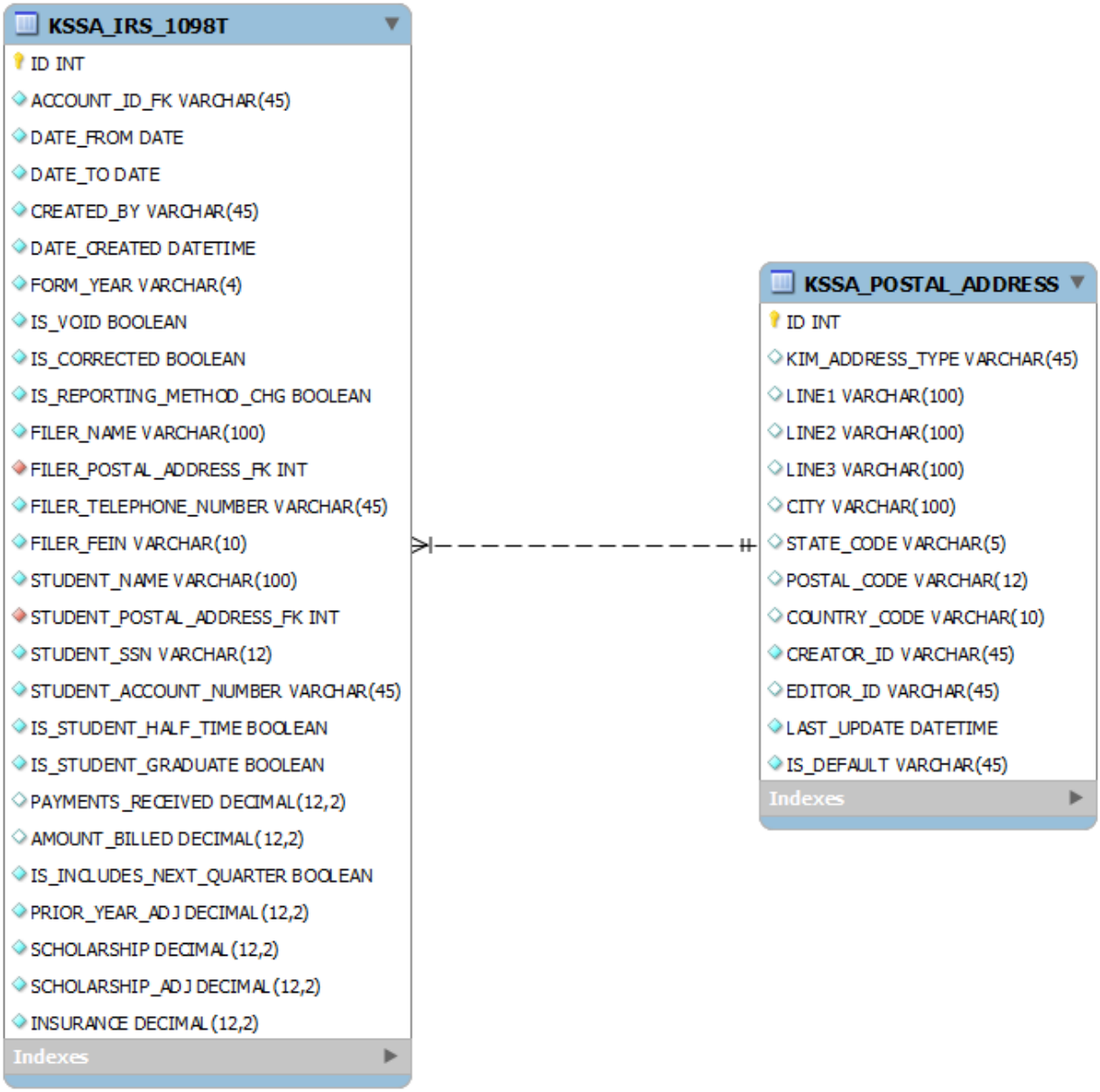

Table: IRS_1098T

Simple key-pair value table for storing system-wide preferences.

ID	Autonumbered primary key.
ACCOUNT_ID_FK	KSA account identifier for the form.
DATE_FROM	Starting date of the report. In most cases, this will be 1/1/xxxx
DATE_TO	Ending date of the report. In most cases this will be 12/31/xxxx
CREATED_BY	Entity that produced the 1098T. 1098T production is usually limited to a subgroup of power users.
DATE_CREATED	Timestamp when the actual entry for the 1098 was created.
FORM_YEAR	The IRS requires the form to be produced for a specific year. This is the four digit year for the production of the form. (example, 2012)
IS_VOID	If the form is voided, this will be true.
IS_CORRECTED	If the form is a correction, this will be true.
IS_REPORTING_METHOD_CHG	If the school has changed its reporting method from the previous year, this field will be true.
FILER_NAME	Name of the filing institution. This is set in the preferences and will be the same for each student in a run of statements.
FILER_POSTAL_ADDRESS_FK	Pointer to the address record for the institution. Note that many records may point to a single address record for the institution.
FILER_TELEPHONE_NUMBER	As with FILER_NAME, but reflecting the telephone number.
FILER_FEIN	As with FILER_NAME, but reflecting the federal employer identification number.
STUDENT_NAME	Concatenated (non-atomic) name for the student.
STUDENT_POSTAL_ADDRESS_FK	Pointer to the address record for the student. Note that a second address record will be created for the student so that if the student's address changes, this record will still reflect the record at the time of 1098T production.
STUDENT_SSN	A masked copy of the student's social security number (see ssnMask on the method.)
STUDENT_ACCOUNT_NUMBER	This may reflect the accountId string, but is stored here in case the school chooses to override this as the account number.
IS_STUDENT_HALF_TIME	If the student is at least half time, this Boolean will be true.
IS_STUDENT_GRADUATE	If the student is a graduate student then this Boolean will be true.
PAYMENTS_RECEIVED	Value of all payments received to be reported. KSA does not support this type of reporting in the current version, so this v value should be blank.
AMOUNT_BILLED	Value of appropriate charges billed to the student. This is the reporting method that KSA supports in the initial version, so this value can be calculated.
IS_INCLUDES_NEXT_QUARTER	If the amount reported includes charges for the next quarter, this field will be true.

PRIOR_YEAR_ADJ	If the AMOUNT_BILLED from the previous year needed to be adjusted, the adjustment amount will be stored here.
SCHOLARSHIP	Total amount of scholarships applied to the student's account.
SCHOLARSHIP_ADJ	If the scholarship amount from the previous year had to be adjusted, the amount will be reported here.
INSURANCE	If there is an insurance reimbursement or refund amount, it will be stored here.