Database Schema

Training Guide

Version 4.0

Under the Hood: MRI Database Schema

Agenda

Objectives

MRI System Tables

- **MRITABLE**
- **MRIFIELD**
- **MRIINDEX**
- **MRIRELN**

General Ledger

- **GLCD**
- **GACC**
- **JOURNAL**
- **GHIS**
- **GLSUM**
- **ENTITY**
- **PERIOD**
- **BMAP**

Accounts Payable

- **BANK**
- **VEND**
- **SESS**
- **INVC**
- **HIST**
- **SCHK**

Commercial Management

- **BLDG**
- **CMPD**
- **SUIT**
- **MOCCP**
- **LEAS**
- **CMRECC**
- **INCH/SECINCH**
- **GLMT/SECGLMT**

Residential Management

- **RMPROP**
- **RMPD**
- **RMBLDG**
- **UNIT**
- **RMLEASE**
- **RMRECC**
- **NAME**
- **PROSPECT**
- **CHGCODE**
- **CLSS**
- **SCHD**

CM/RM Transactions

- **CMBTCH**
- **CMRCPT/CMMISC/CMBNONT**
- **CMLEDG**
- **CMLEDGAPPLY**
- **RMBTCH**
- **RMRCPT/RESCHGCRD/RMBMISC/RMSDADJ**
- **RMLEDG**
- **RMLEDGAPPLY**

Questions and Answers

Under the Hood: MRI Database Schema

Objectives

This class was designed to give readers a basic understanding of the main tables in the MRI modules GL, AP, CM, and RM. The goal of the class is to provide an overview of the key tables and fields in each module, the relationships between these tables, and some information about how and when the tables are populated.

The class assumes a basic knowledge of SQL queries, but the queries needed for this class are generally of the format:

SELECT * FROM table WHERE field='value'

For example, to find all records in the BLDG table in the state of Ohio, the query is:

SELECT * FROM BLDG WHERE STATE='MD'

Armed with this query and a basic understanding of the MRI database schema, you should be able to put most of the information in this class to good use.

Table conventions:

For all tables in the database, the primary key is indicated by a * in front of the appropriate field or fields.

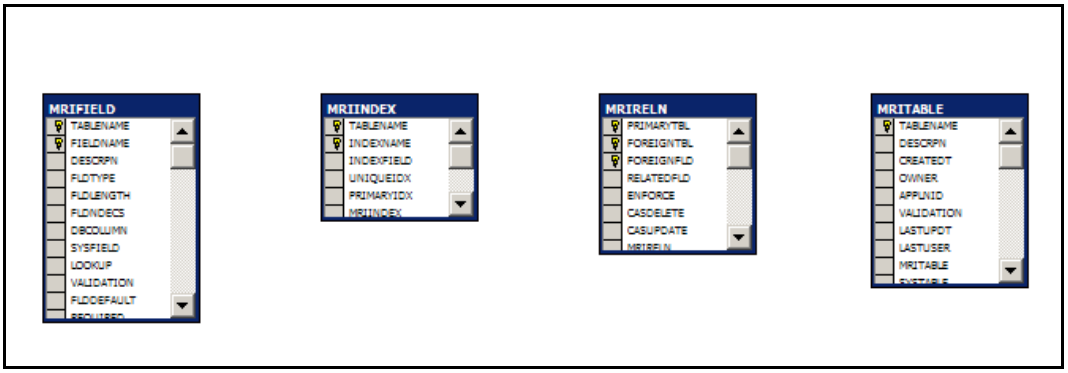
Field	Description
*TABLENAME	Table name
DESCRPN	Table description
APPLNID	Application/module: CM,RM,AP,etc.

MRI System Tables

The following tables are part of every MRI database and describe the database structure itself. These can be useful for finding a particular field or table based on their descriptions.

MRI uses these tables to keep track of tables, fields, and relationships within the database. During table modifications in Table Design or during an upgrade, these tables are maintained as modifications are made to the table. Conversely, any changes made to MRI tables outside of MRI may result in limited functionality and/or loss of the fields during an upgrade.

Table	Description
MRITABLE	All tables in the database
MRIFIELD	All fields in all tables
MRIINDEX	Indexes, primary keys for each table
MRIRELN	Foreign key relationships between tables



MRITABLE

MRITABLE lists every table in the database, along with descriptions, which module each table belongs to, the date and user who created it, etc.

Field	Description
*TABLENAME	Table name
DESCRPN	Table description
APPLNID	Application/module: CM,RM,AP,etc.

Some useful queries:

List all non-MRI (custom) tables:

*SELECT * FROM MRITABLE WHERE MRITABLE='N'*

Find tables containing "amenities" in the description:

*SELECT * FROM MRITABLE WHERE DESCRPN LIKE '%AMENTITIES%'*

MRIFIELD

MRIFIELD lists every field in every table in the database, along with information about the field type, length, etc.

Field	Description
*TABLENAME	Table name
*FIELDNAME	Table description
DESCRPN	Application/module: CM,RM,AP,etc.
FLDTYPE	A=Alphanumeric (character) N=Numeric D=Date C=Counter (autonumber) M=Memo (text) I=Identity (auto timestamp) B=Binary
FLDLENGTH	For A-type fields, indicates the character length. For N-type, determines the SQL data type if FLDNDECS is 0: 1-4=smallint, 5-9=int, 10+=float. If FLDNDECS is 2, the datatype is money. For all other non-zero values, the datatype is float.
SYSFIELD	System catalog reference
LOOKUP	Indicates a foreign key relationship/lookup to another table. If the LOOKUP is "CODELIST", look to the VALIDATION field for CT= for the codetype.
VALIDATION	Contains any validation rules for the field. Also contains CT= for CODELIST lookups. For versions prior to 4.0, field that have been customized will contain ~USERCHANGES= x ~ in this field.
REQUIRED	Indicates a mandatory field.

Some useful queries:

Find all the required fields in the BLDG table:

*SELECT * FROM MRIFIELD WHERE TABLENAME='BLDG' AND REQUIRED='Y'*

Show all fields in the 'JOURNAL' table with field lookups:

*SELECT * FROM MRIFIELD WHERE TABLENAME='JOURNAL' AND LOKUP IS NOT NULL*

Show all fields in the database that store account numbers:

*SELECT * FROM MRIFIELD WHERE SYSFIELD='ACCTNUM'*

MRIINDEX

MRIINDEX lists every index, including primary keys, on every table in the database.

Field	Description
*TABLENAME	Table name
*INDEXNAME	Index name (primary key indexes are named UPKCL_tablename)
INDEXFIELD	The field or fields that make up the index.
PRIMARYIDX	Indicates whether the index is the primary key.

Some useful queries:

Find the primary key fields for table RMLEASE:

*SELECT * FROM MRIINDEX WHERE TABLENAME='RMLEASE'*

MRIRELN

The MRIRELN table lists every relationship (foreign key) between any two given tables in the database. All relationships in MRI are "many-to-one", meaning that for one record in the "primary" table, many records can exist in the "foreign" table. Consider the relationship between MRITABLE and MRIFIELD. In the MRIFIELD table, the TABLENAME field must reference a valid table in MRITABLE—but for any given record in MRITABLE, there can be (and almost always are) many records in MRIFIELD. In this relationship, MRITABLE is considered to be the "primary" table (the "one" side of the relationship), and MRIFIELD is considered to be the "foreign" table (the "many" side of the relationship).

MRIRELN is used by MRI to determine join relationships, lookup lists, and a variety of other functionality. It's also very useful for determining entity relationships when you're writing queries or trying to find out about the database schema.

Field	Description
*PRIMARYTBL	The primary or “one” table in the relationship.
*FOREIGNTBL	The foreign or “many” table in the relationship.
RELATEDFLD	The field or fields that are shared between the two tables.

Some useful queries:

What fields in the BLDG table reference other tables in the database?

*SELECT * FROM MRIRELN WHERE FOREIGNTBL=’BLDG’*

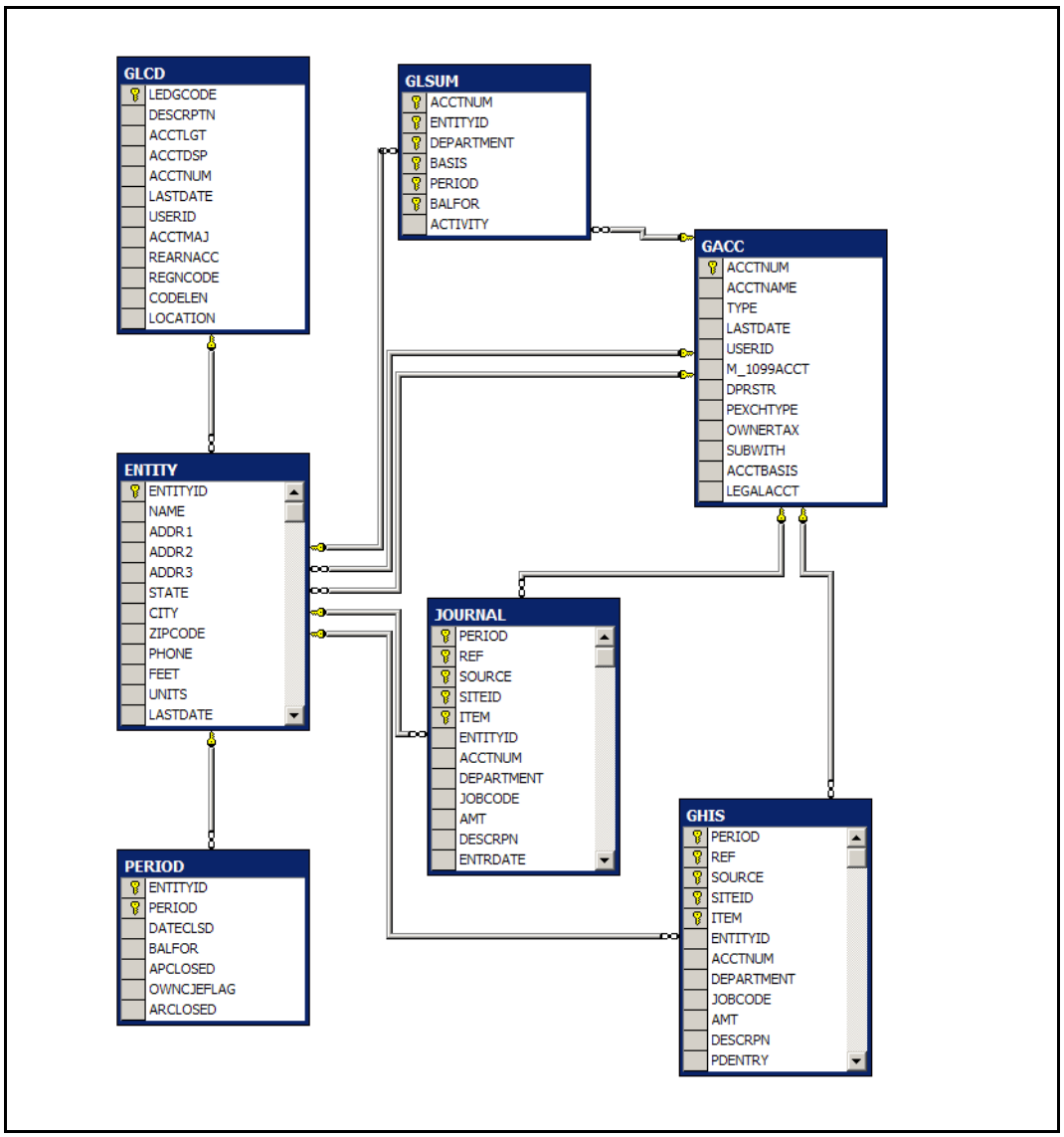
What fields elsewhere in the database reference the BLDG table?

*SELECT * FROM MRIRELN WHERE PRIMARYTBL=’BLDG’*

General Ledger

The tables listed below are the most important in the General Ledger module.

Table	Description
GLCD	Chart of accounts header table
GACC	Chart of accounts
JOURNAL	Current and future period detail transactions
GHIS	Historical detail transactions
GLSUM	Summary account balances
ENTITY	Financial property table
PERIOD	GL calendar
BMAP	Defines the relationship between entities, banks, and cash accounts in your chart.



GLCD/GACC

GLCD contains one record for each chart of accounts in MRI. The default chart is "MR". To enable multiple charts, the management option "Multiple Ledger Codes" (MGNT.MULTLED) must be set to "Y", which is the default value.

The GLCD values define the format of your account numbers, such as length, major/minor segments, and display format.

Field	Description
*LEDGCODE	The two-character ledger code
DESCRPTN	Chart description
ACCTLGT	The total length of the account, not including dashes or the ledger code.
ACCTDSP	Display format. Use 0 or 9 for digits, @ for characters.

ACCTNUM	Determines if the account numbers are numeric only.
ACCTMAJ	The length of the major account segment. Must be less than or equal to the ACCTLGT.
REARNACC	The retained earnings account for the chart. All P&L accounts will be closed to this account at year end, and all balance sheets will sum P&L to this account at runtime.

GACC contains one record for each account in the chart. If the major account length (GLCD.ACCTMAJ) is different from the account length (GLCD.ACCTLGT), any account with all 0's in the minor position will be considered a major account. For example, if the account length is 7 and the major account length is 4, major accounts will be those ending in "000". The major account must be entered before any minor account can be added.

Field	**Description**
*ACCTNUM	The account number, including the ledger code. (i.e. MR1000111)
ACCTNAME	The name of the account
TYPE	This can be I (income/expense), B (balance sheet), or C (cash). C accounts are essentially balance sheet accounts, but are treated differently by the Cash Balance report. I accounts are closed to Retained Earnings at year end and are summed to RE at runtime for any balance sheet report.

Some useful queries:

Display all cash accounts:

*SELECT * FROM GACC WHERE TYPE='C'*

Return all accounts in the MR chart, without the ledger code:

SELECT SUBSTRING(ACCTNUM,3,7) FROM GACC WHERE ACCTNUM LIKE 'MR%'

(this assumes an account length of 7)

JOURNAL/GHIS

All detail transactions in GL are stored in JOURNAL or GHIS according to their period. Current and future transactions are stored in JOURNAL; historical transactions are stored in GHIS.

Field	Description
*PERIOD	Period in YYYYMM format
*REF	6-character journal reference #
*SOURCE	2-character journal source. AP/CM/RM are reserved for the "create journal entries" processes.
*SITEID	2-character site id or @
*ITEM	Item number of the detail line within a set of journal entries.
ENTITYID	A valid entity from ENTITY
ACCTNUM	A valid account from GACC
DEPARTMENT	A valid department from GDEP
JOBCODE	A valid jobcode from GJOB
AMT	The item amount. Debits are stored as positive, credits as negative.
DESCRPN	Journal description
ENTRDATE	Entry date. Typically should be within the month specified in PERIOD.
BASIS	A valid basis from BTYP
REVERSAL	(JOURNAL only) Indicates that the journal entry should be reversed next month.
STATUS	(JOURNAL only) P for "posted", U for "unposted", or S for "system posted" (used by create journal entries programs)
BALFOR	(GHIS only) B for entries in the balance forward period; N for all other activity.

At month-end close, all JOURNAL entries are moved to the GHIS table. (Additionally, any standard entries in GSTD are created in the new period, and any entries marked with a REVERSAL flag are copied to the new period with their amounts reversed).

Each "set" of journal entries is identified by a combination of PERIOD/ REF/ SOURCE/ SITEID. Within each set of entries, the ITEM field counts from 1 to x. There cannot be more than 32767 lines in a journal entry.

Within any given journal entry, the AMT column should always add up to 0, because debits (positive) should equal credits (negative).

GHIS has a special flag, BALFOR, which indicates if the activity is to be recorded in the "balance forward period", sometimes called the 13th period. Balance forward entries are recorded after the year-end period, and before the year-open period. The actual value of PERIOD is the same as the year-open period (i.e. January). During the GL year-end close, the year-end balance is computed for each account and entered in the balance forward period in GHIS. Any year-end closing activity, such as closing P&L accounts to retained earnings, or other closing accounts as indicated in GCLS, are also made in the balance forward period.

Some useful queries:

Show all journal entries for a given account:

*SELECT * FROM JOURNAL WHERE ACCTNUM='MR1000111' AND ENTITYID='100' AND BASIS='A'*

Show the balance forward activity for a given account:

*SELECT * FROM GHIS WHERE ACCTNUM='MR1000111' AND ENTITYID='100' AND BASIS='A' AND PERIOD='200601' AND BALFOR='B'*

Find any journal entry that is out of balance:

SELECT PERIOD, REF, SOURCE, SITEID, SUM(AMT) FROM JOURNAL GROUP BY PERIOD, REF, SOURCE, SITEID HAVING SUM(AMT)<>0

GLSUM

All activity entered into JOURNAL or GHIS is accumulated into balances in GLSUM by ACCTNUM, ENTITYID, PERIOD, BASIS, and DEPARTMENT.

Field	Description
*ACCTNUM	A valid account from GACC
*ENTITYID	A valid entity from ENTITY
*DEPARTMENT	A valid department from GDEP
*BASIS	A valid basis from BTYP
*PERIOD	Period in YYYYMM format
*BALFOR	B for balfor balances, N for

	activity
ACTIVITY	The total activity for the period or balance forward period. Debit balances are positive, credit balances negative.

Balances in GLSUM are kept up-to-date by SQL triggers. As an account is debited, the appropriate balance(s) in GLSUM is added to. As an account is credited, the appropriate balance(s) in GLSUM is subtracted from.

The BALFOR=B record in GLSUM indicates the year-opening balance for a given account. All other records (BALFOR=N) indicate the monthly activity for a given account. To compute the year-end balance for an account, find the most recent B record, and add all subsequent records:

SELECT SUM(ACTIVITY) FROM GLSUM WHERE ACCTNUM='MR1000111' AND ENTITYID='100' AND DEPARTMENT='@' AND BASIS='A' AND PERIOD>= '200601'

All financial reports in MRI, aside from the general journal report, compute account balances and activity from GLSUM, not from the detail tables JOURNAL/GHIS. Consequently, it's important that the GLSUM table contain an accurate reflection of GL balances. Normally the SQL triggers will keep GLSUM up-to-date no matter how the data comes into JOURNAL/GHIS. If GLSUM is incorrect, it can be rebuilt using the Rebuild Summary Table option in MRI GL for Windows.

ENTITY

An entity in MRI represents the financial aspect of a property. Typically an entity represents a single building or property, but you can set up an entity that has no properties associated with it (in the case of a cost center) or multiple properties (in the case of an office park or multi-use building).

Field	Description
*ENTITYID	Entity id
NAME	Entity name
PROJID	A valid project from PROJ
CURPED	The current GL period in YYYYMM format
YEAREND	The next year end period in YYYYMM format.

PERIOD

The PERIOD table keeps track of the current GL and AP period for each entity.

Field	Description
*ENTITYID	A valid entity id from ENTITY
*PERIOD	Period in YYYYMM format.
DATECLSD	Date of GL close. Should be blank only for current and future GL periods.
BALFOR	Indicates that the period is a balance-forward period. Should be one per year, and the calendar should start with a B record.
APCLOSED	Indicates whether AP has been closed for the period. Should only be "Y" for the last (most recent) record.

The financial calendar for a given entity is a combination of the PERIOD records, and the fields CURPED and YEAREND in ENTITY.

The calendar should start with a B (balance forward) period, and every 12th record after should also be a B. The DATECLSD field should be populated for every period except the current GL period, and any future GL periods if allowed by ENTITY.MAXOPEN.

Only the last record should have APCLOSED='N'. Closing AP is what creates the next PERIOD record, always. For a year-end close, the AP close will create the new (January) record with a BALFOR value of N. This will be changed to a B by the year-end close.

Balances for all GL reports are computed by going back to the last B period and adding activity since. So, having an incorrect or missing B record can cause your GL balances to be computed incorrectly on reports. Also, some parts of the program compute the current GL period by looking at the oldest PERIOD record with a blank DATECLSD. Consequently, having an old record in the calendar with a blank DATECLSD can also cause subtle GL problems in some reports.

The current GL period should be first (oldest) record with a blank DATECLSD, and should match what's in ENTITY.CURPED. The value of ENTITY.YEAREND should be 12 months later than the last (latest) B record in PERIOD.

Some useful queries:

To find the current GL period for an entity:

SELECT CURPED FROM ENTITY WHERE ENTITYID='100'

SELECT MIN(PERIOD) FROM PERIOD WHERE ENTITYID='100' AND DATECLSD IS NULL

To find the current AP period for an entity:

SELECT MAX(PERIOD) FROM PERIOD WHERE ENTITYID='100' AND APCLOSED='N'

To find the most recent and next year end period:

SELECT MAX(PERIOD) FROM PERIOD WHERE ENTITYID='100' AND BALFOR='B'

SELECT YEAREND FROM ENTITY WHERE ENTITYID='100'

BMAP

The BMAP table defines the relationship between entities, banks, and cash accounts in your chart.

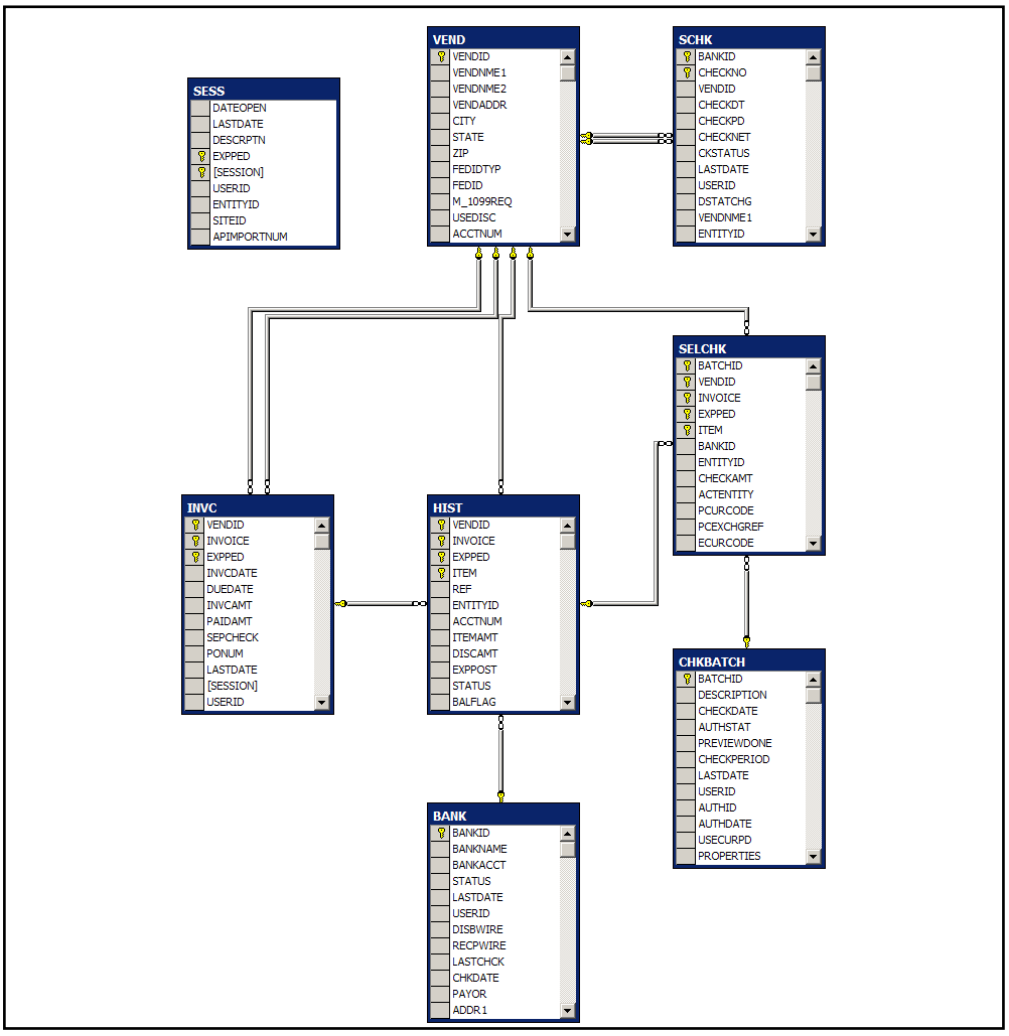
Field	Description
*ENTITYID	A valid entityid from ENTITY
*CASHTYPE	A valid cash type from CTYP
BANKID	A valid bankid from BANK.
ACCTNUM	A valid cash account from GACC.

The BMAP table is involved in any cash transaction involving the entity, including checks cut from Accounts Payable and cash received in Commercial Management or Residential Management. The combination of ENTITYID and CASHTYPE is used in the subledgers to identify which bank to pay checks to (in the case of AP), and which GL accounts to debit/credit when journal entries are created (in the case of AP, CM, and GL.

Accounts Payable

The tables listed below are the most important in the Accounts Payable module.

Table	Description
VEND	Vendor master table
BANK	Bank master table
SESS	Invoice sessions
INVC	Invoice header table
HIST	Invoice detail table
SCHK	Check detail table
CHKBATCH	Check selection batch header
SELCHK	Check selection detail



VEND

The VEND table contains a record for each vendor.

Field	Description
*VENDID	Vendor ID
VENDNME1	Vendor Name

BANK

The BANK table contains a record for each bank.

Field	Description
*BANKID	Bank ID
BANKNAME	Bank Name
STATUS	C = Closed I = Inactive P = Pending

SESS

The SESS table contains a record for each invoice session. If "Session Reporting" is enabled (APOPTION.SESSION), sessions are used to group invoices together during invoice entry. Aside from reporting, AP sessions have no impact on the data.

Field	Description
*SESSION	Session ID
DESCRPTN	Session description
EXPPED	The expense period for the session in YYYYMM format.
ENTITYID	A valid entity from ENTITY. If APOPTION.SESSENTITY ("Single Entity Per Session") is enabled, this field is enabled in invoice entry.

Some useful queries:

List all invoices for a given session:

*SELECT * FROM INVC WHERE SESSION= 'xxxx'*

List all sessions for a given period:

*SELECT * FROM SESSION WHERE EXPPED='200605'*

INVC/HIST

INVC and HIST contain the transaction detail for accounts payable. For any vendor invoice, there will be one record in INVC (the invoice header), and at least one record in HIST (the invoice detail).

INVC Fields	Description
*VENDID	A valid vendor ID from VEND.
*INVOICE	Invoice number
*EXPPED	The expense period for the invoice in YYYYMM format.
INVCDATE	The entry date of the invoice.
DUEDATE	The due date for the invoice
INVCAMT	The original invoice amount. Should be the sum of HIST.ITEMAMT for all HIST records for this invoice.
PAIDAMT	The actual invoice amount paid.
SESSION	If Session Reporting is enabled, this will contain a valid session id from SESSION.

HIST Fields	Description
*VENDID	A valid vendor id from VEND.
*INVOICE	Invoice number
*EXPPED	The expense period for the invoice in YYYYMM format.
*ITEM	The detail item number within the invoice.
REF	Line item description.
ENTITYID	A valid entity from ENTITY.
ACCTNUM	The expense account for the line item. Must be a valid account in GACC.
ITEMAMT	The line item amount.
STATUS	R = Ready to pay H = Hold

	P = Paid V = Voided W = Withdrawn C = Carried Forward U = Unused check D = Deleted M = Manual check I = Info Only
CHECKNO	Check number if paid
CHECKDT	Check date if paid
CASHTYPE	Cash type, used to determine appropriate bank and cash account from the BMAP table.
EXPPOST CKPOSTED	The period in which the expense and check sides of the detail line were posted to GL.
EXPGLREF CKACCRLREF CKCASHGLREF	The GL reference number (JOURNAL.REF) of the expense, accrual check, and cash check entries in GL.

For a given invoice, the HIST record will always contain at least one detail record. Other line items will be created by MRI for voided checks, taxes, and other information related to the payment of the invoice.

The combination of EXPPED, INVOICE, and VENDID uniquely identifies each invoice, with HIST.ITEM numbers counting from 1 to x for all detail. Consequently, the same invoice number can be used repeatedly (in different periods) for the same vendor.

When checks are paid for the invoice, the check number and date is recorded in the appropriate HIST record. At the same time, a record is created in SCHK (see below) with the details of the check. Since a single check may pay multiple line items, the SCHK record will show the total amount paid, which will be the sum of all HIST.ITEMAMTs for items paid.

When journal entries are created from AP, the HIST.ACCTNUM is used for the expense account for each line item, and the ENTITY.APACCTNUM is used for the accounts payable (debit) side of the journal entry, in the case of accrual accounting. For paid items, the cash account specified in BMAP is used, based on the ENTITYID/CASHTYPE specified in HIST. Periods and GL reference numbers are recorded in the fields as listed above. The JOURNAL.SOURCE field in GL will be populated with “AP”.

Some useful queries:

Show all AP details related to a give GL reference. :

*SELECT * FROM HIST WHERE EXPGLREF='006149' AND EXPPED='200303*

Find all HIST records that have no corresponding INVC record ("orphan records"):

*SELECT * FROM HIST WHERE NOT EXISTS (SELECT * FROM INVC WHERE INVC.EXPPED=HIST.EXPPED AND INVC.INVOICE=HIST.INVOICE AND INVC.VENDID=HIST.VENDID)*

Find all instances where the same invoice has been entered more than once for the same vendor:

SELECT VENDID,INVOICE,COUNT(*) FROM INVC GROUP BY VENDID,INVOICE HAVING COUNT(*) > 1

SCHK

The SHK table contains a record for each check paid out of MRI.

Field	Description
*BANKID	The bank out of which the check was paid. Must be a valid bank in BANK.
*CHECKNO	The check number.
VENDID	The vendor the check was paid to. Must be a valid vendor in VEND.
CHECKDT	The date of the check.
CHECKPD	The period in which the check was paid.
CHECKNET	The check amount
CKSTATUS	U = Unused C = Cleared X = In process (bankrec) V = Void O = Outstanding (bankrec)

Some useful queries:

List all outstanding checks for bank 100:

*SELECT * FROM SCHK WHERE CKSTATUS='O' AND BANKID='100'*

Find a given check for a given bank:

*SELECT * FROM SCHK WHERE BANKID='100' AND LTRIM(CHECKNO)='1593'*

CHKBATCH/SELCHK

During check printing, all HIST items selected for printing are listed in a check batch. CHKBATCH contains a header record for the batch, and SELCHK lists the individual line items to be paid.

CHKBATCH Fields	Description
*BATCHID	Batch ID number
DESCRIPTION	Batch description
CHECKDATE	Check date for all checks in this batch.

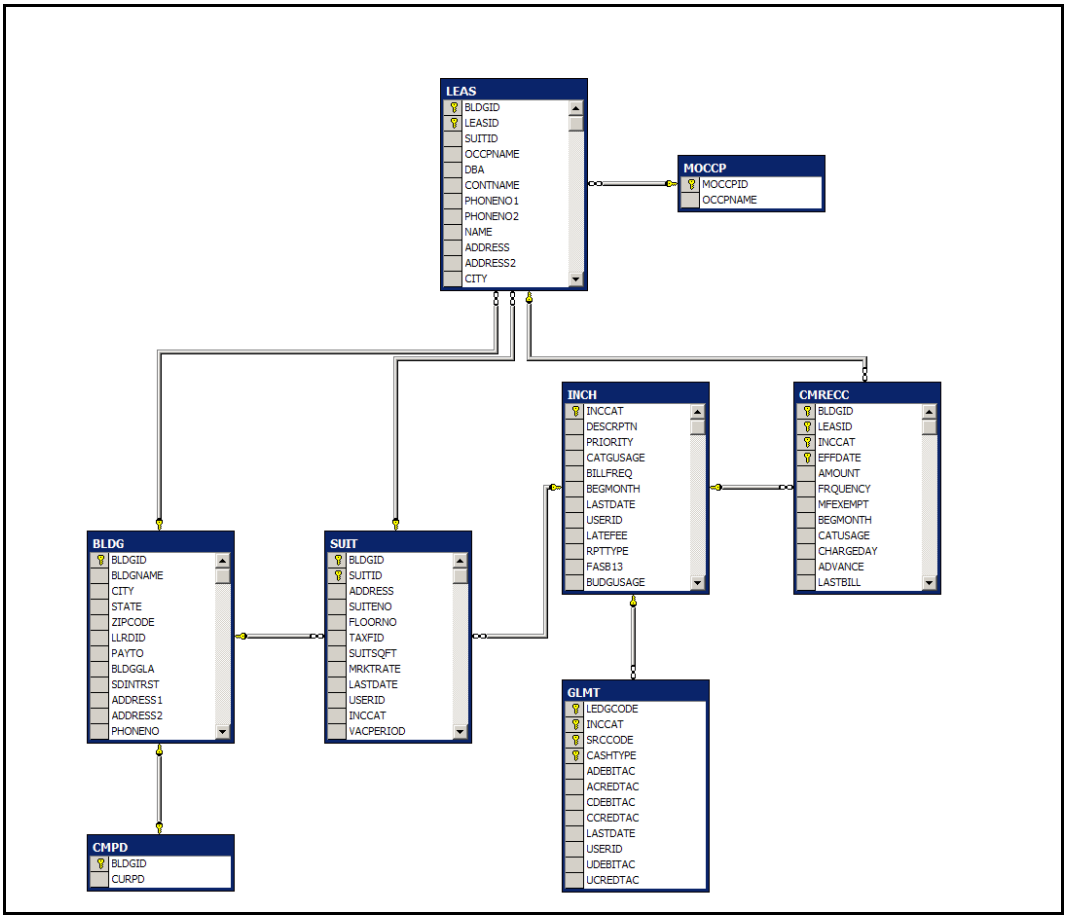
SELCHK Fields	Description
*BATCHID	The selection batch id from CHKBATCH
*VENDID *INVOICE *EXPPED *ITEM	These fields identify the HIST record being paid.
BANKID	The bank to be paid from. Must be a valid bank from BANK.
ENTITYID	The entityid from which the check is paid. Should match HIST.ENTITYID.
CHECKAMT	The item amount to be paid.

CHKBATCH and SELCHECK only exist while the checks are being selected for payment. After the checks are paid, the check selection batch is deleted.

Commercial Management

The tables listed below are the most important in the Commercial Management module.

Table	Description
BLDG	Building table
CMPD	Current CM period for each building
SUIT	Suite table
MOCCP	Master Occupants
LEAS	Lease table
CMRECC	Recurring charges
INCH/SECINCH	Income categories
GLMT/SECGLMT	CM/GL master interface chart



INCH/SECINCH

Income categories such as “rent” or “common area maintenance” are setup as codes in the INCH and SECINCH tables. INCH contains rental codes, SECINCH contains security deposit codes.

Field	Description
*INCCAT/SECINCCAT	Income category code
DESCRPTN	Description
PRIORITY	When cash receipts are automatically allocated to open charges, indicates the priority of each category. Lower numbers indicate a higher priority.
RPTTYPE	Indicates where charges of this category will be listed on the rent roll: B = Base rent R = Recoveries O = Other

GLMT/SECGLMT

“Master Interface Chart”: The GLMT and SECGLMT tables determine how journal entries in GL will be created from transactions in CM.

Field	Description
*LEDGCODE	A valid GL ledger code from GLCD.
*INCCAT	A valid income category from INCH.
*SRCCODE	Source code or “type” of transaction. CH = CH/NC transactions CR = CR/PR transactions CN = CN transactions
*CASHTYPE	The cash type for cash transactions. Must be a valid cash type from CTYP.
ADEBITAC ACREDTAC	The debit and credit GL accounts for each transaction, respectively. Each account must be a valid

CDEBITAC CCREDTAC	account in GACC.

When GL journal entries are created from CM, each transaction results in a debit and credit in JOURNAL. The GLMT/SECGLMT tables map each type of transaction to appropriate GL accounts. For cash transactions, the cash account is determined by the BMAP table.

For CH (charge) transactions, the CH code in GLMT is used to obtain the debit and credit accounts needed. For NC (non-cash adjustment) transactions, the CH code in GLMT is also used, but the debit and credit accounts are reversed. The same rules are used for CR (cash receipt) and PR (payment reversal) transactions.

The GLBLDG and SECGLBLDG tables can be populated if a given building uses different accounts from the master interface (GLMT) mapping.

BLDG

The BLDG table contains one record for each commercial building.

Field	Description
*BLDGID	Building ID
BLDGNAME	Building name
LLRDID	A valid landlord id from the LLRD table.
MNGRID	A valid manager id from the MNGR table.
ENTITYID	Associates the building with an entity from the ENTITY table.
BILLDATE	Indicates the most recent rentup date.

CMPD

Stores the current CM period for each building.

Field	Description
*BLDGID	Building ID
CURPD	Current CM period in YYYYMM format.

SUIT

The SUIT table contains one record for each suite in every building.

Field	Description
*BLDGID	Building ID from BLDG table
*SUITID	Suite ID
SUITENO	The physical suite number for address purposes
FLOORNO	The floor number, used for stacking plans.
SUITSQFT	The current square footage

Useful queries:

Show all suites for a given building:

*SELECT * FROM SUIT WHERE BLDGID='100'*

Find the total square footage for all suites in a building:

SELECT SUM(SUITSQFT) FROM SUIT WHERE BLDGID='100'

MOCCP/LEAS

The LEAS table holds one record for each lease. A given LEAS record can be thought to represent the physical lease document for a particular suite. The MOCCP table represents the "master occupant," which represents a tenant. A single MOCCP record might be associated with one LEAS record, or if the tenant is occupying multiple spaces, there can be many LEAS records associated with a single MOCCP.

When a new lease is created in MRI, both an MOCCP and LEAS record are created.

MOCCP Fields	Description
*MOCCPID	Master occupant ID
OCCPNAME	Occupant name.

LEAS Fields	Description
*BLDGID	Building ID from BLDG table
*LEASID	Lease ID
SUITID	Suite ID from the SUIT table

MOCCPID	Master occupant ID from the MOCCP table.
GENERATION	The generation or revision number of the lease. The initial LEAS is always generation 1. A renewal or change in terms results in a new LEAS record with one higher generation number.
ADDLSPACE	Will be "N" for the primary suite. If multiple suites are associated with the MOCCPID, all other LEAS records after the primary space will be marked with "Y"
OCCPNAME	Occupant name
OCCPSTAT	Occupancy status: C = Current I = Inactive N = New P = Proposed
EXECDATE RENTSTRT OCCUPNCY BEGINDATE EXPIR VACATE STOPBILLDATE	Dates indicating execution, rent start, occupancy, lease begin, expiration, vacate, and stop bill dates.

For any given tenant, there will always be at least one LEAS and one MOCCPID. The LEAS record will have a GENERATION of 1, and ADDLSPACE='N'. If additional spaces are added to the main lease, additional LEAS records will be added with the same GENERATION and ADDLSPACE='Y'. Lease renewal or revision of terms will result in a new set of LEAS records with GENERATION=2 and so forth.

Because each LEAS has its own terms (start and vacate date, etc) and its own set of recurring charges (see CMRECC below), a single occupant can be billed different rates for the different spaces that they occupy. Alternately, all billing can be assigned to the main LEAS, with the additional LEAS records used merely to show occupancy. The field LEAS.PRIMARYCHGS determines if charges are allowed only on the primary lease, or on all leases associated with the MOCCPID.

Useful queries:

Show all leases for a given MOCCPID:

*SELECT * FROM LEAS WHERE MOCCPID='STEVESM'*

Show only leases with the most recent GENERATION number:

*SELECT * FROM LEAS WHERE MOCCPID='STEVESM' AND GENERATION = (SELECT MAX(GENERATION) FROM LEAS WHERE MOCCPID='STEVESM')*

CMRECC

All past, current, and future recurring charges for each lease are stored in the CMRECC table.

Field	Description
*BLDGID	Building ID from BLDG table
*LEASID	Lease ID from LEAS table
*INCCAT	Income category form the INCH table
*EFFDATE	Effective (starting) date for the charge
AMOUNT	Charge amount
FRQUENCY	Charge frequency. For a list of frequencies, run: SELECT * FROM CODELIST WHERE CODETYPE='BILLFREQ'
LASTBILL	The date through which the charge has been billed. If rentup was run on 6/1/06, a monthly charge would have a LASTBILL date of 6/30/06.
ENDDATE	The date on which the charge ends. Should typically be blank, unless the charge ends mid-lease and is not replaced by a subsequent charge for the same INCCAT.
INEFFECT	Indicates if the charge is currently in effect. For any given lease (BLDGID/LEASID), there should only be one CMRECC record for each INCCAT with INEFFECT='Y'.

Each recurring charge for a given lease is stored as a record in the CMRECC table. If a charge changes mid-lease, multiple CMRECC records are created with each charge amount, with the appropriate EFFDATE for each amount.

CM Rentup uses the CMRECC table to generate CMLEDG records. Rentup also maintains the INEFFECT and ENDDATE columns as needed.

Useful queries:

Show all charges in effect for a given lease:

*SELECT * FROM CMRECC WHERE BLDGID='100' AND LEASID='000005' AND INEFFECT='Y'*

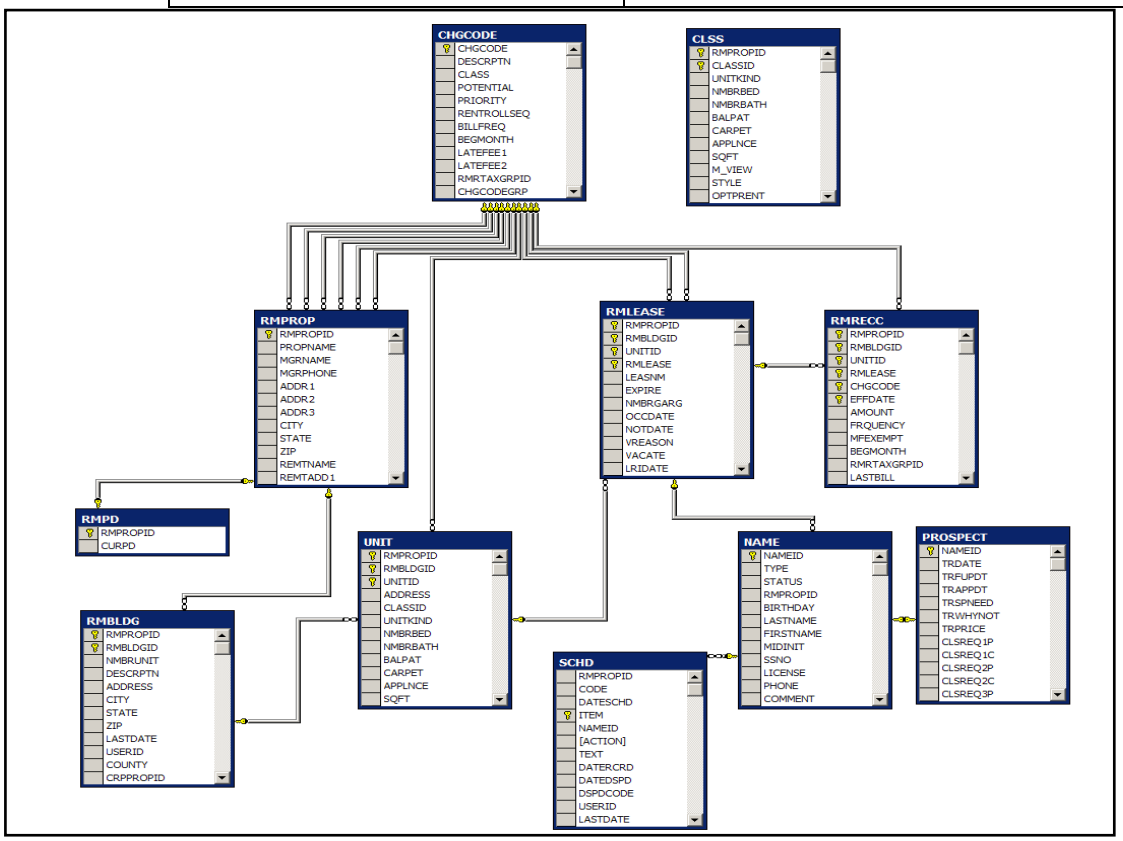
Show all charges in effect on a given date for a given lease:

*SELECT * FROM CMRECC WHERE BLDGID='100' AND LEASID='000005' AND EFFDATE = (SELECT MAX(EFFDATE) FROM CMRECC B WHERE B.BLDGID=CMRECC.BLDGID AND B.LEASID=CMRECC.LEASID AND B.INCCAT=CMRECC.INCCAT AND EFFDATE <= '3/1/06')*

Residential Management

The tables listed below are the most important in the Residential Management module.

Table	Description
RMPROP	RM Property table
RMPD	RM Current period table
RMBLDG	RM Building table
CLSS	Unit class table
UNIT	Unit table
RMLEASE	RM lease table
RMRECC	Recurring charges
NAME	Residents, applicants, and prospects
PROSPECT	Prospect table
CHGCODE/SECCODE	Charge code table
RMGLMT/RMSECGLMT	RM Master interface chart
SCHD	Scheduler table



CHGCODE/SECCODE

Charge code categories such as "rent" or "utilities" are setup as codes in the CHGCODE and SECCODE tables. CHGCODE contains rental codes, SECCODE contains security deposit codes.

Field	Description
*CHGCODE/SECCODE	Charge code ID
DESCRPTN	Description
CLASS	A = Allowance (concessions) R = Rentable items M = Memo O = Other
POTENTIAL	Indicates whether the charge code is included in vacancy potential calculations.
PRIORITY	When cash receipts are automatically allocated to open charges, indicates the priority of each category. Lower numbers indicate a higher priority.

RMGLMT/RMSECGLMT

"Master Interface Chart": The RMGLMT and RMSECGLMT tables determine how journal entries in GL will be created from transactions in RM.

Field	Description
*LEDGCODE	A valid GL ledger code from GLCD.
*CHGCODE	A valid charge code from CHGCODE
*SRCCODE	Source code or "type" of transaction. CH = CH/NC transactions CR = CR/PR transactions CN = CN transactions
*CASHTYPE	The cash type for cash transactions. Must be a valid cash type from CTYP.
ADEBITAC	The debit and credit GL accounts

ACREDTAC CDEBITAC CCREDTAC	for each transaction, respectively. Each account must be a valid account in GACC.

When GL journal entries are created from RM, each transaction results in a debit and credit in JOURNAL. The RMGLMT/RMSECGLMT tables map each type of transaction to appropriate GL accounts. For cash transactions, the cash account is determined by the BMAP table.

For CH (charge) transactions, the CH code in RMGLMT is used to obtain the debit and credit accounts needed. For NC (non-cash adjustment) transactions, the CH code in RMGLMT is also used, but the debit and credit accounts are reversed. The same rules are used for CR (cash receipt) and PR (payment reversal) transactions.

The RMGLBLDG and RMSECGLBLDG tables can be populated if a given building uses different accounts from the master interface (RMGLMT) mapping.

RMPROP

This table lists all properties in the database. A property typically represents an entire residential apartment complex.

Field	Description
*RMPROPID	Property ID
PROPNAME	Property name
ENTITYID	Ties the property to a valid entity from the ENTITY table.
BILLDATE	The most recent RENTUP date.
PROCDATE	Actual day on which RENTUP was run.
WEB	Indicates if the property is available in MRI Web.

RMBLDG

Each physical building in a residential property is recorded as a different record in the RMBLDG table. This table is used mostly for grouping and unit identification.

Field	Description
*RMPROPID	Property ID
*RMBLDG	Building ID
DESCRPTN	Building description

CLSS

Each type or class of unit within a property is listed as a record in the CLSS table. Default information for each unit in a class includes pricing, amenities, number of beds and baths, and square footage.

Field	Description
*RMPROPID	Property ID
*CLASSID	Unit class ID
UNITKIND	Unit "kind" (highrise, garden, etc). To see all unit kinds, run SELECT * FROM CODELIST WHERE CODETYPE='UNITKIND'
NMBRBED	Descriptive number of beds
NMBRBATH	Descriptive number of baths

UNIT

The UNIT table contains a record for each unit in a property.

Field	Description
*RMPROPID	Property ID
*RBLDGID	RM Building ID from RMBLDG
*UNITID	Unit ID
CLASSID	Unit class id from CLSS
USTATUS	Unit status. Typical values are: A = Vacant available B = Notice available O = Occupied For a complete list of statuses, run SELECT * FROM CODELIST WHERE CODETYPE='UNITUSTATUS'
MAXCURLEA	Maximum number of residents allowed
NOCURLEA	Number of current leases.

Useful queries:

Show all units for a given property:

*SELECT * FROM UNIT WHERE RMPROPID='800'*

Show only vacant, available units:

*SELECT * FROM UNIT WHERE RMPROPID='800' AND USTATUS='A'*

UNIT

The UNIT table contains a record for each unit in a property.

Field	Description
*RMPROPID	Property ID
*RBLDGID	RM Building ID from RMBLDG
*UNITID	Unit ID
CLASSID	Unit class id from CLSS
USTATUS	Unit status. Typical values are: A = Vacant available B = Notice available O = Occupied For a complete list of statuses, run SELECT * FROM CODELIST WHERE CODETYPE='UNITUSTATUS'
MAXCURLEA	Maximum number of residents allowed
NOCURLEA	Number of current leases.

Useful queries:

Show all units for a given property:

*SELECT * FROM UNIT WHERE RMPROPID='800'*

Show only vacant, available units:

*SELECT * FROM UNIT WHERE RMPROPID='800' AND USTATUS='A'*

RMLEASE

Each residential lease is stored as a separate record in RMLEASE. Each new resident results in a new RMLEASE record.

Field	Description
*RMPROPID	Property ID
*RBLDGID	RM Building ID from RMBLDG
*UNITID	Unit ID
*RMLEASE	A numeric value representing the lease number. The first lease in a given unit is number 1. Each subsequent lease or renewal increments this number.
LEASNM	Length of lease in months. MTM is used for month-to-month residents.
EXPIRE OCCDATE NOTDATE VACATE	Dates representing the expiration, occupy date, notice date, and vacate date.
CURTERMSTART	The start date of the current lease. For first-year leases this is the same as the OCCDATE

Useful queries:

Show the current lease for a given unit:

*SELECT * FROM RMLEASE WHERE RMPROPID='800' AND RMBLDGID='001' AND UNITID='104' AND VACATE IS NOT NULL*

RMRECC

All past, current, and future recurring charges for each lease are stored in the RMRECC table.

Field	Description
*RMPROPID	Property ID from RMPROP table
*RMBLDGID	Building ID from RMBLDG table
*UNITID	Unit ID from UNIT table
*RMLEASE	Lease ID from RMLEASE table
*CHGCODE	Charge code from CHGCODE table
*EFFDATE	Effective (starting) date for the charge
AMOUNT	Charge amount

FRQUENCY	Charge frequency. For a list of frequencies, run: SELECT * FROM CODELIST WHERE CODETYPE='CHGCODEFREQ'
LASTBILL	The date through which the charge has been billed. If rentup was run on 6/1/06, a monthly charge would have a LASTBILL date of 6/30/06.
ENDDATE	The date on which the charge ends. Should typically be blank, unless the charge ends mid-lease and is not replaced by a subsequent charge for the same CHGCODE.
INEFFECT	Indicates if the charge is currently in effect. For any given lease (RMPROPID/ RMBLDGID/ UNITID/ RMLEASE), there should only be one RMRECC record for each CHGCODE with INEFFECT='Y'.

Each recurring charge for a given lease is stored as a record in the RMRECC table. If a charge changes mid-lease, multiple RMRECC records are created with each charge amount, with the appropriate EFFDATE for each amount.

RM Rentup uses the RMRECC table to generate RMLEDG records. Rentup also maintains the INEFFECT and ENDDATE columns as needed.

Useful queries:

Show all charges in effect for a given lease:

*SELECT * FROM RMRECC WHERE RMPROPID='800' AND RMBLDGID='A' AND UNITID='A27' AND RMLEASE=3 AND INEFFECT='Y'*

Show all charges in effect on a given date for a given lease:

*SELECT * FROM RMRECC WHERE RMPROPID='800' AND RMBLDGID='A' AND UNITID='A27' AND RMLEASE=3 AND EFFDATE = (SELECT MAX(EFFDATE) FROM RMRECC B WHERE B.RMPROPID=RMRECC.RMPROPID AND B.RMBLDGID=RMRECC.RMBLDGID AND B.UNITID=RMRECC.UNITID AND B.RMLEASE=RMRECC.RMLEASE AND B.CHGCODE=CMRECC.CHGCODE AND EFFDATE <= '3/1/06')*

NAME

All residents, coresidents, roommates, prospects, and applicants are stored in the NAME table.

Field	Description
*NAMEID	Name ID
TYPE	A = Applicant P = Prospect R = Resident T = Transfer
STATUS	A = Active (prospects, applicants) C = Current * = Co-resident R = Other resident N = New O = Old I = Inactive (prospects, applicants)
RMPROPID	A valid property from RMPROP
RESPROPID RMBLDGID UNITID RMLEASE	For all residents, these four fields identify the RMLEASE record associated with the resident.
NAMEGROUP	For NAMEID of the main resident (status=C) for the unit. For C-status names, this is the same as the NAMEID.
PREVNAMEID	For transfers, the NAMEID of the previous leas.
TABID	For prospects, indicates the screen in the application web pages (web only)

Useful queries:

Show all current residents in a given property:

*SELECT * FROM NAME WHERE RMPROPID='800' AND TYPE='R' AND STATUS='C'*

For a given co-resident, find the main resident:

*SELECT * FROM NAME WHERE NAMEID = (SELECT NAMEGROUP FROM NAME WHERE NAMEID='HO0000000012')*

PROSPECT

All prospective residents are stored in PROSPECT as well as in NAME. When a guest card or phone card is entered, a record is created in both NAME and PROSPECT.

Field	Description
*NAMEID	The NAMEID as recorded in the NAME table.
TRDATE	Traffic date
URPROP	Property ID for this prospect, from RMPROP.
URBUILD	Building ID for reserved unit, if any. From RMBLDG.
URUNIT	Unit ID for reserved unit, if any. From UNIT

Useful queries:

Show any prospect who has a given unit reserved:

*SELECT * FROM PROSPECT WHERE URPROP='800' AND URBUILD='A' AND URUNIT='A29'*

Show all prospect records for currently active prospects at a property:

*SELECT * FROM PROSPECT WHERE NAMEID IN (SELECT NAMEID FROM NAME WHERE RMPROPID='800' AND TYPE='P' AND STATUS='A')*

SCHD

The SCHD table contains all activities from the RM scheduler, such as moveins, moveouts, and transfers.

Field	Description
RMPROPID	Property ID from RMPROP

CODE	Status code: O(pen) or C(losed)
DATESCHD	Schedule date
*ITEM	Unique identifier
NAMEID	The name id associated with this item, from the NAME table.
ACTION	“MOVE IN” or “MOVE OUT”
TEXT	Item description
DATERCRD	Date item was recorded
DATEDSPD	Date item was processed

Useful queries:

Show all items scheduled on a given day:

*SELECT * FROM SCHD WHERE DATESCHD=’3/15/06’*

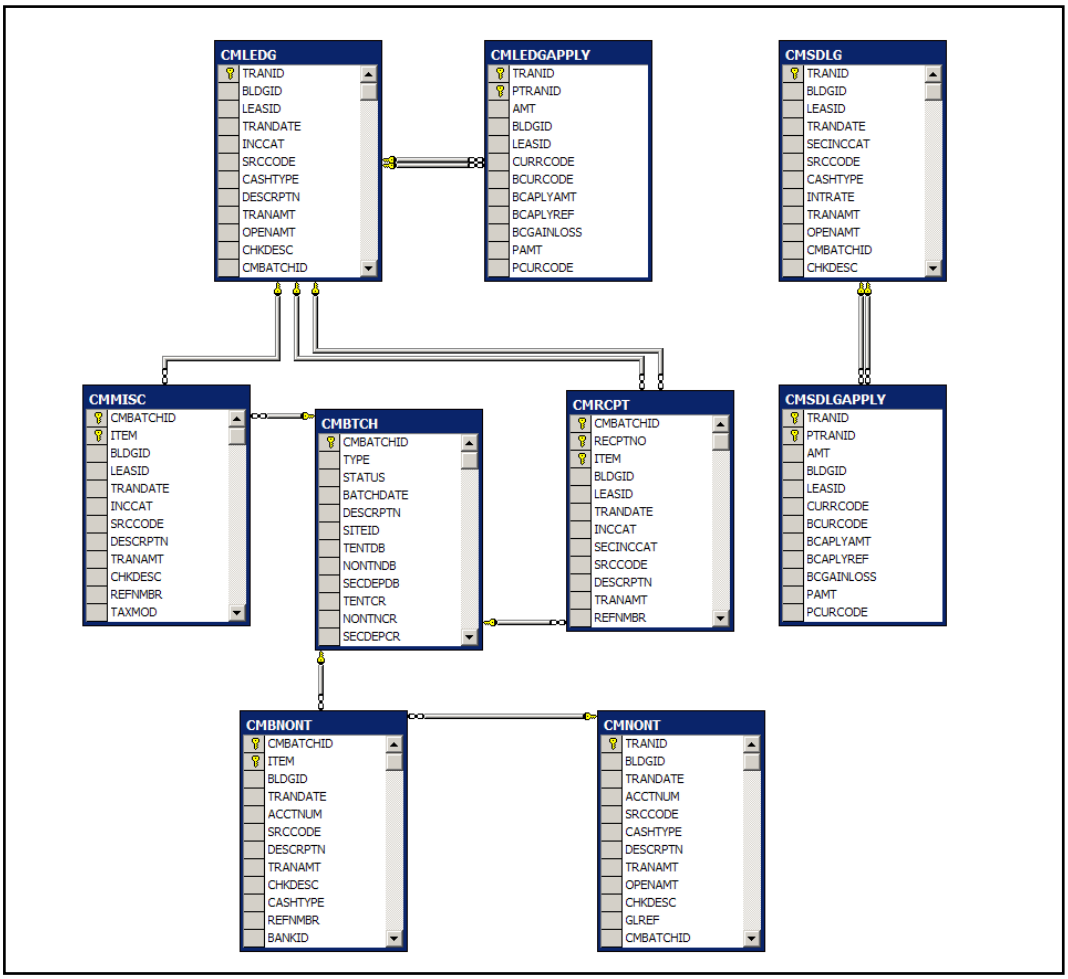
Show all items associated with a given NAMEID:

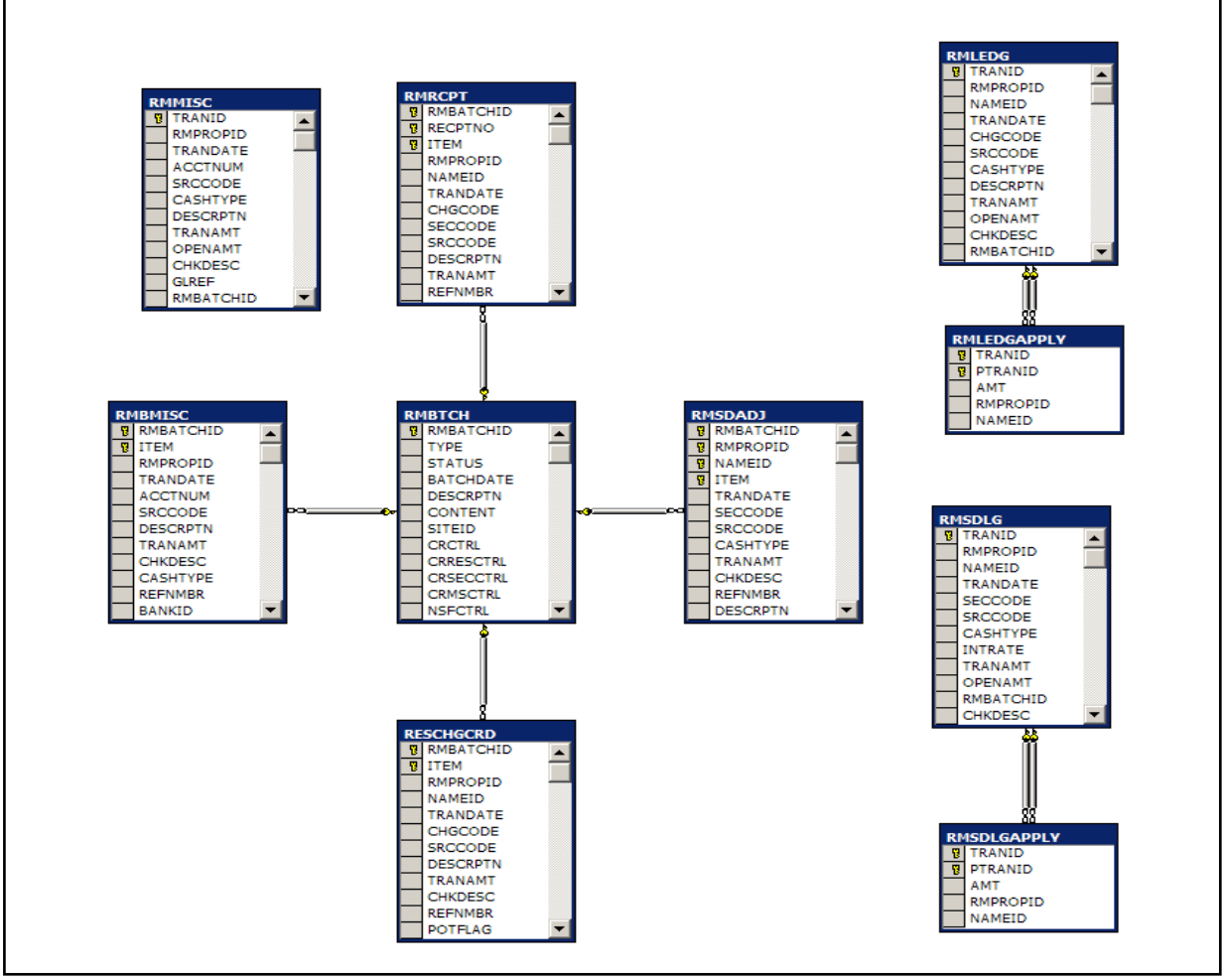
*SELECT * FROM SCHD WHERE NAMEID=’MR10000015’*

RM/CM Transaction Tables

Although the table names are slightly different, both RM and CM use nearly identical table structures to store transactions:

CM Table	RM Table	Description
CMBTCH	RMBTCH	Batch header table
CMRCPT CMMISC CMBNONT CMSDADJ	RMRCPT RESCHGCRD RMBMISC RMSDADJ	Temporary tables for unposted batches (receipts, charges, non-tenant, security)
CMLEDG CMSDLG	RMLEDG RMSDLG	Tenant and security deposit ledger tables
CMLEDGAPPLY CMSDLGAPPLY	RMLEDGAPPLY RMSDLGAPPLY	Ledger apply tables





CMBTCH/RMBTCH

For each transaction batch, there is one header record which contains the batch id, control totals, and posted status.

Field	Description
RMBATCHID/CMBATCHID	Table name
TYPE	Table description
STATUS	Batch status: O(pen), C(losed), or L(ocked)
BATCHDATE	Batch date
DESCRPTN	Batch description

CMRCPT,CMMISC,CMBNONT,CMSDADJ

RMRCPT,RESCHGCRD,RMBMISC,RMSDADJ

As a batch is being entered, but before it is posted, transactions are recorded to these temporary tables. Resident receipts, miscellaneous billing adjustments, non-tenant transactions, and security deposit transactions are each stored in a separate table until the batch is posted. Although an unposted transaction might apply to a posted transaction (such

as a cash receipt applying to a posted charge), none of the transactions in this table have any effect on the balances of posted transactions until they are posted.

CMLEDG/RMLEDG

CMSDLG/RMSDLG

Once a batch is posted, the transactions are created in either xxLEDG (for tenant/resident transactions) or xxSDLG (for security deposit transactions).

Field	Description
*TRANID	A unique transaction ID
BLDGID/LEASID (CM)	The building and lease id associated with the transaction, from LEAS.
RMPROPID/NAMEID (RM)	The nameid of the resident associated with the transaction, from NAME.
NAMEGROUP (RM)	The namegroup of the resident associated with the transaction, from NAME. For primary residents, this is the same as NAMEID. For co-residents, it is the NAMEID of the primary resident.
TRANDATE	Transaction date
CHGCODE (RM) / INCCAT (CM)	The chargecode or income category for the transaction, from CHGCODE or INCH.
SRCCODE	The source code, or type of transaction. CH = Charge NC = Non-cash adjustment CR = Cash receipt PR = Payment reversal CN = Concession NS = NSF check
CASHTYPE	For cash transactions, the cash type from CTYP
DESCRPTN	Description
TRANAMT	Transaction amount

OPENAMT	The unapplied amount of the transaction.
CHKDESC	For cash transactions, the check number of the receipt
RM/CMBATCHID	Batch ID for this transaction, from RMBTCH or CMBTCH
REFNMBR	The TRANID of the RMLEDG/CMLEDG that this transaction was applied to.
PERIOD	Period in YYYYMM format.
GLREF	The GL reference number (JOURNAL.REF) of the journal entry created from this transaction.
POSTED	Posted, Y or N.

Each transaction (cash receipt, charge, adjustment) is listed as a separate record in the appropriate ledger table. A transaction is initially entered with the TRANAMT and OPENAMT columns the same. Once a transaction is applied against another transaction (a receipt pays off a charge, for example, or a payment reversal reverses a receipt), the OPENAMT is adjusted to show the unapplied amount. A charge which is fully applied has an OPENAMT of 0.

When a transaction is applied against another transaction, the REFNMBR field of the first transaction is set to the TRANID of the transaction it applied to. For example, if 000012, a cash receipt, is applied to 000008, a charge, then the REFNMBR field in 000012 is set to "000008":

TRANID	INCCAT	SRCCODE	DESCRPTN	TRANAMT	OPENAMT	REFNMBR
000008	RNT	CH	Autochg	100	0	
000012	RNT	CR	Rent pymt.	-100	0	000008

A transaction can only apply to a transaction with the same INCCAT or CHGCODE. A cash receipt for RNT, for example, can't be applied to an outstanding charge for LAT. To apply a RNT receipt to a LAT charge, the system creates two new transactions called a CreditApply pair, sometimes called a PR/CR pair:

TRANID	INCCAT	SRCCODE	DESCRPTN	TRANAMT	OPENAMT	REFNMBR

000015	LAT	CH	Late fee	100	0	
000016	RNT	CR	Rent pymt.	-100	0	
000017	RNT	PR	CreditApply	100	0	000016
000018	LAT	CR	CreditApply	-100	0	000015

This shows that transaction 17, a payment reversal, was applied to transaction 16, the cash receipt, to close the receipt. Then a new cash receipt, 000018, was created for INCCAT "LAT", and applied to the original charge, 000015, to close it out.

Because a single transaction can apply to many transactions, as is the case when a single RNT payment closes out multiple RNT charges, it is sometimes inadequate to have a single REFNMBR to represent the TRANID of the applied transaction. For this reason, a special "APPLY" table exists to track how each transaction applies to each other transaction. This is explained in detail below.

When CM or RM Create Journal Entries is run, each transaction creates a debit/credit pair in the JOURNAL table. This is done by consulting the appropriate GLMT table to find the debit and credit accounts, and in the case of cash transactions, by looking at BMAP to find the correct bank and cash account. When a transaction is journalized, the JOURNAL.REF number is recorded in CMLEDG.GLREF or RMLEDG.GLREF.

Some useful queries:

Find all open transactions for a given resident:

*SELECT * FROM RMLEDG WHERE NAMEID='0000000088' AND OPENAMT<>0*

Find the current outstanding balance for a given lease:

SELECT SUM(TRANAMT) FROM CMLEDG WHERE BLDGID='100' AND LEASID='000005'

CMLEDGAPPLY/RMLEDGAPPLY

CMSDLGAPPLY/RMSDLGAPPLY

Any time a transaction applies to another transaction in CM or RM, the application is recorded in the appropriate APPLY table.

Field	Description
*TRANID	The transaction being applied
*PTRANID	The transaction being applied *to*.
AMT	The amount of TRANID applied to PTRANID.

BLDGID/LEASID (CM) RMPROPID/NAMEID (RM)	The building and lease id associated with the transaction, from LEAS. The nameid of the resident associated with the transaction, from NAME.

To understand the function of CMLEDGAPPLY (all the …APPLY tables work the same), let's look at a simple transaction from above:

TRANID	INCCAT	SRCCODE	DESCRPTN	TRANAMT	OPENAMT	REFNMBR
000008	RNT	CH	Autochg	100	0	
000012	RNT	CR	Rent pymt.	-100	0	000008

The cash receipt 000012 was applied to the charge 000008, closing both transactions. Here's how this transaction is recorded in CMLEDGAPLY:

TRANID	PTRANID	AMT
000012	000008	-100

For any transaction in CMLEDG, we can compute the OPENAMT by taking the TRANAMT and doing two things:

1. *Adding* any CMLEDGAPPLY.AMT where the transaction appears as CMLEDGAPPLY.PTRANID

2. *Subtracting* any CMLEDGAPPLY.AMT where the transaction appears as CMLEDG.TRANID

In this case, the original TRANAMT for 000008 is 100. To this we *add* (-100), because 000008 shows up as the PTRANID in CMLEDGAPPLY. 000008 doesn't show up as TRANID in CMLEDGAPPLY, so we *subtract* nothing. The new open balance is 100 + (-100) = 0.

The original TRANAMT for 000012 is -100. To this we *add* nothing, because 000012 does not show up as the PTRANID in CMLEDGAPPLY. But we do *subtract* (-100) from it, because 000012 shows up as the TRANID in CMLEDGAPPLY. So the new balance is (-100) – (-100) = 0.

The open amount for any transaction can always be recomputed as follows:

SELECT TRANAMT + (SELECT SUM(AMT) FROM CMLEDGAPPLY WHERE PTRANID='000012') – (SELECT SUM(AMT) FROM CMLEDGAPLY WHERE TRANID='000012') FROM CMLEDG WHERE TRANID='000012'

This query simply takes the original TRANAMT, adds any amounts where the transaction is found as a CMLEDGAPPLY.PTRANID, and subtracts any amounts where the transaction is found as a CMLEDGAPPLY.TRANID.

In some cases, the TRANID and PTRANID are reversed in CMLEDGAPPLY. For example, the transaction above might be recorded like this:

TRANID	PTRANID	AMT
000008	000012	100

Notice that the AMT is now recorded as a positive 100. Although the transaction is recorded "backwards" in CMLEDGAPPLY, the calculation still works:

The open amount for transaction 000008 is 100 (the original TRANAMT), *plus* nothing (since 000008 does not appear in CMLEDGAPPLY.PTRANID), *minus* 100, since 000008 does show up as CMLEDGAPPLY.TRANID. The new open amount is 100 + 0 – 100 = 0.

The sign of CMLEDGAPPLY insures that the appropriate balances work out, regardless of which transaction is chosen as TRANID and which is chosen as PTRANID.

Let's take a look at the creditapply transaction we looked at above:

TRANID	INCCAT	SRCCODE	DESCRPTN	TRANAMT	OPENAMT	REFNMBR
000015	LAT	CH	Late fee	100	0	
000016	RNT	CR	Rent pymt.	-100	0	
000017	RNT	PR	CreditApply	100	0	000016
000018	LAT	CR	CreditApply	-100	0	000015

In CMLEDGAPPLY, this is recorded as such:

TRANID	PTRANID	AMT
000017	000016	100
000018	000015	-100

Or, possibly, it might look like this:

TRANID	PTRANID	AMT
000016	000017	-100
000015	000018	100

In either case, the records in CMLEDGAPPLY record the application of transaction 000017 to transaction 000016, and that of 000018 to 000015.

Some useful queries:

To find the outstanding balance of any given transaction:

SELECT TRANAMT + (SELECT SUM(AMT) FROM RMLEDGAPPLY WHERE PTRANID=’000016’) – (SELECT SUM(AMT) FROM RMLEDGAPPLY WHERE TRANID=’000016’) FROM RMLEDG WHERE TRANID=’000017’