innovative

Load Profile Training Manual

Revision Date: 2016-02-04

INTRODUCTION

GOALS OF LOAD PROFILE TRAINING

By the end of Load Profile Training, participants should be able to:

- Understand the structure of the Innovative database, including:
  - Innovative record structure
  - Variable-length and fixed-length fields in each record type
  - Fixed-length field values
- Understand the load profiles already in place on the system
- Update and create load profiles for the following record types: bibliographic, authority, patron, item, order and holdings/checkin. All record types need to be provided in MARC format
- Be familiar with the issues that need to be considered for each new load profile, including:
  - Overlay settings for the following record types: bibliographic, authority and patron. Overlay of linked records is not supported with this workshop
  - Protecting fields in the bibliographic, authority or patron record from overlay
  - Using Record Templates
  - Attaching item, order, and holdings/checkin records
  - Deriving call numbers and location codes
- Update existing load profiles
- Create new load profiles
- Create simple translation tables (for example, to translate an alphabetic value in the incoming data into a numeric value in the Innovative record)
- Add menu options to allow for batch loading files of MARC records using the new load profiles
- Test load tables and translation tables before putting them into production

TRAINING AGENDA

Day One

Introductions

Review of training goals

Review of Innovative record structure

Anatomy of a data load

Load tables (m2btab files)

Review of standard load profiles

Global Variable Functions (@)

Day Two

Questions from day one

Special Processing Functions (%)

Command Functions (#)

Translation tables (m2bmap files)

Data analysis before profiling

Menu options for locally created load profiles

Viewing a file of MARC records

The m.marcload.local file

Test loading records

Review

Support from Innovative

Day Three (Workshop)

Load profile for an authority control project

Load profile for patron records

Load profile for loading bibliographic records with attached order records

Load profile for loading e-book bibliographic records

INNOVATIVE RECORD STRUCTURE

All record types in the Innovative system contain fixed- and variable-length fields.

RECORD TYPES

Authority Records

Authority records do not link to other records.

Authority records are used to redirect a search from a subject, name, or title heading that is not used in a library's catalog (the invalid form) to one that is used (the valid form). Innovative supports the full MARC format for authority records. Each type of authority record must be loaded through its own load table to ensure that the MARC fields are assigned to the correct field group tag which allows them to be indexed appropriately.

Bibliographic Records

A bibliographic record may link to:

- 300 holdings/checkin records
- 5000 item records
- 200 order records
- 5000 volume records

Bibliographic records represent one title in your collection. Bibliographic records may be stored in a brief non-MARC format or in the full MARC format, including MARC tags, indicators, subfield codes, and diacritics. The call number may be in the bibliographic and/or the item record.

Contact Records

A contact record may link to:

- an unlimited number of resource records

Contact records contain information about contacts for electronic resources. The loading of contact records will not be covered in Load Profile Training.

Course Records

A course record may link to:

- an unlimited number of bibliographic records
- 350 item records

Course records contain data about courses that have material on reserve. The loading of course reserve records will not be covered in Load Profile Training.

Holdings/Checkin Records

Holdings/Checkin records cannot exist alone. A single holdings/checkin record may link to:

- one bibliographic record
- 1000 item records for monographic holdings

Holdings/Checkin records contain data about serials issues. Checkin cards exist within holdings/checkin records. Holdings/Checkin records are usually stored in non-MARC format, but certain fields (most notably, call numbers) may be stored with MARC tags. This training will not cover loading holdings/checkin records with checkin cards.

If your library uses monographic holdings, this record is referred to as a holdings record and it can be linked to bibliographic records and to multiple item records.

Invoice Records

Invoice records do not link to other records.

Invoice records contain data from the process of paying for materials in Sierra and Millennium Acquisitions. The loading of invoice records will not be covered in Load Profile Training.

Item Records

Item records cannot exist alone. A single item record may link to:

- one or more bibliographic records. 1000 bibliographic records for "bound with" titles. It is not possible to achieve this via record loading; it must be done manually
- one or more holdings records for monographic holdings

- one patron record (circulation checkouts)

Item records usually represent one copy or volume. The item record contains the barcode and other copy- or volume-specific information, but no bibliographic information. The call number may be in the bibliographic and/or the item record.

Item records are usually stored in a non-MARC format, but certain fields (most notably call numbers) may be stored with MARC tags.

License Records

License records cannot exist alone. A license record may link to:

- one or more resource records

License records contain information about license details for electronic resources. The loading of license records will not be covered in Load Profile Training.

Order Records

Order records cannot exist alone. A single order record may link to:

- 100 funds
- one and only one bibliographic record
- one and only one resource record

Order records contain acquisitions information about the title. Order records are stored in a non-MARC format.

Patron Records

A single patron record may link to:

- 2100 item records (circulation checkouts)
- one or more session records

Patron records contain information about people who use the library. Patron records typically originate in a text format, which is then converted into a pseudo- MARC format for loading. Once they are loaded, patron records are stored in a non-MARC format.

Program Records

Program records contain information about library-sponsored programs including description, fee and location information for patrons and additionally alerts, publishing dates and ticklers for staff. The loading of program records will not be covered in Load Profile Training.

Resource Records

A single resource record may link to:

- 1000 license records
- 1000 contact records
- 1000 order records
- one or more holdings/checkin records

Resource records contain information about electronic resources. The loading of resource records will not be covered in Load Profile Training.

Session Records

Session records contain information about library-sponsored programs including start date and time, duration, status, instructor's contact information, registration list and waitlist. The loading of session records will not be covered in Load Profile Training.

Vendor Records

Vendor records may link to:

- an unlimited number of order records.

Vendor records contain information about book and serial vendors. The loading of vendor records will not be covered in Load Profile Training.

Volume Records

Volume records may link to:

- one bibliographic record
- one or more item records

Volume records contain information about multi-volume titles and the associated bibliographic and item records. The loading of volume records will not be covered in Load Profile Training.

VARIABLE-LENGTH FIELDS

Variable-length fields are always referred to in load profiles by their alphabetic field group tags. You can get a list of the field group tags in your system using the following character-based menu options:

- **M > MANAGEMENT information**
  - **I > INFORMATION about the system**
    - **C > CODES used**
      - **T > TAGS of variable-length fields**

NOTE

The standard system variable-length fields are listed below, for use in training only. Individual libraries may have customized their variable-length field codes and/or labels. Please check your library's list of variable-length fields for exact values.

**** See the following pages of bibliographic, order, holdings/checkin, authority, item, and patron variable-length field lists. ****

Bibliographic Records

TAGS OF BIBLIOGRAPHIC VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	LABEL	INDEX TAG
!	REC INFO	
"	BIB INFO	
_	LEADER	
c	CALL #	c
a	AUTHOR	Wa
t	TITLE	Wkt
e	EDITION	
p	PUB INFO	
r	DESCRIPT	
s	SERIES	Wat
n	NOTE	Wt
d	SUBJECT	Wd
b	ADD AUTHOR	at
u	ADD TITLE	t
x	CONTINUES	
z	CONT'D BY	
w	RELATED TO	
o	BIB UTIL #	o
i	STANDARD #	i
l	LCCN	
g	GOV DOC #	g
h	LIB HAS	
k	TOC DATA	Wat
y	MISC	
1	LOCATIONS	

NOTE

Please check your library's list of variable-length fields for exact values. This is especially important when writing instructions to protect fields in the bibliographic record from overlay. See the @ov_protect section for more information.

Order Records

TAGS OF ORDER VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	LABEL	INDEX TAG
!	REC INFO	
#	ORDER INFO	
i	IDENTITY	
x	FOR CURR	
n	NOTE	
z	INT NOTE	
s	SELECTOR	
r	REQUESTOR	
q	VEN ADDR	
v	VEN NOTE	
f	VEN TITL #	
b	PO INFO	
p	BLANKET PO	
j	TICKLER	
k	TICKLERLOG	
0	PAID	
1	LOCATIONS	
2	FUNDS	
3	REC COPIES	
4	LIST PRICE	
5	REOPEN DAT	
6	STATUS REP	

Holdings/Checkin Records

TAGS OF HOLDINGS/CHECKIN VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	LABEL	INDEX TAG
!	REC INFO	
$	CHECK INFO	
_	LEADER	
c	CALL #	
i	IDENTITY	
y	CAPTIONS	
h	LIB HAS	
n	NOTE	
z	INT NOTE	
q	VEN ADDR	
v	VEN NOTE	
f	VEN TITLE #	
m	MESSAGE	
w	BIND INFO	
b	BIND TITLE	
j	TICKLER	
k	TICKLERLOG	
1	LOCATIONS	
2	CKIN INFO	
4	ROUTING	

Authority Records

TAGS OF AUTHORITY VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	LABEL	INDEX TAG
!	REC INFO	
%	AUTH INFO	
_	LEADER	
o	LC ARN	z
a	NAME AUTHR	at
b	NAME S FRM	at
c	NAME SA	at
d	SUBJ AUTH	d
e	SUBJ S FRM	d
f	SUBJ SA	d
t	UT AUTH	t
u	UT SEE FRM	t
v	UT SA	t
n	NOTE	
y	MISC	

NOTE

Please check your library's list of variable-length fields for exact values. This is especially important when writing instructions to protect fields in the authority record from overlay. See the @ov_protect section for more information.

Item Records

TAGS OF ITEM VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	LABEL	INDEX TAG
!	REC INFO	
&	ITEM INFO	
b	BARCODE	b
c	CALL #	c
v	VOLUME	
m	MESSAGE	
x	INT NOTE	
y	URL	
r	RESER NOTE	
a	ITEM FIELD	
6	COURSE ID	
7	SAVE ITEM	

Patron Records

TAGS OF PATRON VARIABLE-LENGTH FIELDS

FIELD GROUP TAG	**LABEL**	**INDEX TAG**
!	REC INFO	
'	PATRON INF	
n	PATRN NAME	n
a	ADDRESS	
h	ADDRESS2	
t	TELEPHONE	
p	TELEPHONE2	
u	UNIQUE ID	u
m	MESSAGE	
b	P BARCODE	b
z	EMAIL ADDR	
=	PIN	
1	FAMILY ID	

NOTE

Please check your library's list of variable-length fields for exact values. This is especially important when writing instructions to protect fields in the patron record from overlay. See the @ov_protect section for more information.

FIXED-LENGTH FIELDS

To view the code values and labels for fixed-length fields in Sierra/Millennium select**:** **Admin | Parameters | General**. The most common ones that are translated/mapped are:

Branches (Location Codes)
Fixed-Length Codes (Material Type, Item Status and others)
Item Types

See **also Admin | Parameters | Circulation**

Patron Type
Pcode3

In the character-based system, you can get a list of the fixed-length fields and the codes and labels in your system using the following menu path:

```
M > MANAGEMENT information
    I > INFORMATION about the system
      C > CODES used
          X > FIXED-length codes
```

As fixed-length field values vary widely from system to system, there are no standard code values to include in this manual.

NOTE

The fixed-length field values in the incoming data must be defined in the validation tables in **Admin | Parameters | General** or **Circulation**. Code values that are undefined will not be loaded; fixed-length field values in Record Templates will be loaded instead (see the discussion of the **@dflt** Global Variable Function).

Fixed-Length Fields in Load Tables

In load tables fixed-length fields are always identified by their line number in an internal file that defines each field. This file is not accessible to system users.

Standard labels for fixed-length fields are listed below, along with their corresponding line number which you add to the load table.

The *Innovative Guide and Reference* also gives line numbers in the column labeled Field Number when you follow the links for each record type on page no. 105775 (**Sierra/Millennium Reference | How Innovative Systems Store Information | Fixed-length Fields**). Your system may use different labels, but the function of the fixed-length field and its line number are the same on every Innovative system.

If you use Create Lists you see the line numbers to the left of the field name when constructing the Boolean Search, for example **61 Item Type**.

NOTE

Some fixed-length fields are reserved for system use only, and it is not possible to load data into these fields. Only the fields marked below with an asterisk (*) may be loaded via a load profile.

If a fixed-length field is not loaded, it should be protected from overlay. See the discussion of the **@ov_protect** Global Variable Function.

See the following pages of bibliographic, order, holdings/checkin, authority, item and patron fixed-length field lists include the field name and corresponding line number.

Both the short and long labels (if one exists) are shown. A subset of fixed-length fields are reserved for system use only and it is not possible to load data into these fields. Fields marked with an asterisk (*) may be loaded via a load table (AKA m2btab file).

Bibliographic Records

LINE #	**FIXED FIELD LABEL**
024	LANG or Language*
025	SKIP or Skip*
026	LOCATION or Location*
027	COPIES or Copies* (Few libraries use this field)
028	CAT DATE or Cat. Date* (set via @cdate trigger)
029	BIB LVL or Bib Level (BCODE1)*
030	MAT TYPE or Material Type (BCODE2)*
031	BCODE3 or Bib Code 3*
089	COUNTRY or Country*
107	MARCTYPE or MARC Type* (Rarely used. Set via defaults for new records.)

Order Records

LINE #	FIXED FIELD LABEL
001	ACQ TYPE or Acq Type*
002	LOCATION or Location*
003	CDATE or Cat Date*
004	CLAIM or Claim*
005	COPIES or Copies*
006	CODE1 or Order Code 1*
007	CODE2 or Order Code 2*
008	CODE3 or Order Code 3*
009	CODE4 or Order Code 4*
010	E PRICE or Est. Price*
011	FORM or Form*
012	FUND or Fund*
013	ODATE or Order Date*
014	ORD NOTE or Order Note*
015	ORD TYPE or Order Type*
016	RACTION or Recv Action*
017	RDATE or Recv Date*
018	RLOC or Recv Location*
019	BLOC or Billing Location*
020	STATUS(O) or Status* (See **NOTE** below)
021	TLOC or Transit Location*
022	VENDOR or Vendor*
023	LANG or Language*
032	PAID DATE or Paid Date
033	INV DATE or Invoice Date
034	PAID AMT or Paid Amount
100	COUNTRY or Country*
106	VOLUMES or Volumes*

NOTE

Many order record status codes are system-generated and cannot be loaded. ONLY the following order status codes can be safely loaded via a load table:

a	Fully Paid
c	Serial On Order
d	Serial Paid
o	On Order
z	Cancelled
1	Pending / On Hold
2	Approval Rejection

Holdings/Checkin Records

LINE #	FIXED FIELD LABEL
035	LABEL TYPE or Label Type*
036	SCODE1 or Serial Code 1*
037	SCODE2 or Serial Code 2*
038	COPIES or Copies*
039	CLAIMON or Claim On
040	LOCATION or Location*
041	RLOC or Recv Location*
042	VENDOR or Vendor*
118	SCODE3 or Serial Code 3*
119	SCODE4 or Serial Code 4*
120	UPDCNT or Update Count*
121	PCOUNT or Piece Count*
137	ECHECKIN or E-Checkin* (Serials E-Checkin servers only)
159	MEDIA TYPE or Media Type* (Serials E-Checkin servers only)
266	INHERIT LOC or Inherit Location* (Monographic Holdings records)

Authority Records

LINE #	**FIXED FIELD LABEL**
111	AMARCTYPE or MARC Type* (Rarely used. Set via defaults for new records.)
112	ACODE1 or Auth. Code 1*
113	ACODE2 or Auth. Code 2*
114	ASUPPRESS or Auth. Suppress*

Item Records

LINE #	**FIXED FIELD LABEL**
057	BIB HOLD
058	COPY # or Copy No.*
059	ICODE1 or Item Code 1*
060	ICODE2 or Item Code 2*
061	ITYPE or Item Type*
062	PRICE or Price*
063	OUT DATE or Checkout Date
064	OUT LOC or Checkout Location
065	DUE DATE or Due Date
066	PATRON# or Patron No.
067	LPATRON or Last Patron
068	LCHKIN or Last Checkin
069	INV DATE or Inventory Date
070	IN LOC or Checkin Location
071	# RENEWALS or No. of Renewals
072	# OVERDUE or No. of Overdues
073	ODUE DATE or Overdue Date
074	IUSE3 or Item Use 3*
075	RECAL DATE or Recall Date
076	TOT CHKOUT or Total Checkouts*
077	TOT RENEW or Total Renewals*
078	LOUTDATE or Last Checkout Date
079	LOCATION or Location*
087	LOANRULE or Loanrule
088	STATUS or Status* (see **NOTE** below)
093	INTL USE or Internal Use*
094	COPY USE or Copy Use*
097	IMESSAGE or Item Message*
108	OPACMSG or OPAC Message*
109	YTDCIRC or Year-to-Date Circ*
110	LYCIRC or Last Year Circ*
127	AGENCY or Item Agency* (Consortium Management Extensions Only)
161	VI CENTRAL or VI Central (INN-Reach only)
162	IR DLSSITE or IR Dist Learn Same Site
264	HLDG TAG or Holdings Item Tag*
265	INHERIT LOC or Inherit Location* (Monographic Holding records)

**** See next page for a continuation of the item fixe-length fields. ****

NOTE

Some item status values are system-generated and cannot be loaded via a load table. The following item status codes CANNOT be loaded:

n	Billed
t	In Transit
z	Claims Returned
$	Lost and Paid
!	On Holdshelf

At INN-Reach libraries, the following item status codes CANNOT be loaded:

@	Off Campus
&	Requested
#	<System> Requested
%	<System> Returned
(	<System> Paged
)	<System> Cancelled
-	<System> Re-Requested

Patron Records

LINE #	FIXED FIELD LABEL
043	EXP DATE or Expiration Date*
044	PCODE1 or Patron Code 1*
045	PCODE2 or Patron Code 2*
046	PCODE3 or Patron Code 3*
047	P TYPE or Patron Type*
048	TOT CHKOUT or Total Checkouts*
049	TOT RENWAL or Total Renewals*
050	CUR CHKOUT or Current Checkouts
051	BIRTH DATE or Birth Date*
053	HOME LIBR or Home Library*
054	PMESSAGE or Patron Message*
055	HLODUES or Highest Overdues
056	MBLOCK or Manual Block*
095	CL RTRND or Claims Returned
096	MONEY OWED or Money Owed
099	FIRM*
101	BLK UNTIL or Block Until
102	CUR ITEMA or Current Item A
103	CUR ITEMB or Current Item B
104	PIUSE*
105	OD PENALTY or Overdue Penalty
122	ILL REQUES or ILL Request
123	DEBIT BAL or Debit Balance
124	CUR ITEMC or Current Item C
125	CUR ITEMD or Current Item D
126	PCODE4 or Patron Code 4* (Consortium Management Extensions only)
158	PAT AGENCY or Patron Agency* (Consortium Management Extensions only)
160	VP CENTRAL or VP Central (INN-Reach only)
163	CIRCACTIVE or Last Circ Activity*
263	LANG PREF or Preferred Language* (Staff Interface or WebPAC in multiple languages)
268	NOTICE PREF or Notice Preference*
269	REG ON REC or Registrations on Record (Program Registration Only)
270	TOTAL REG or Total Registrations (Program Registration Only)
271	TOTAL ATTEND or Total Programs Attended (Program Registration Only)
297	WAIT ON REC or Waitlists on Record (Program Registration Only)

REGULAR EXPRESSIONS

Regular expressions are used throughout this manual. Lists of expressions you are likely to encounter are below. Please refer to a UNIX guide or manual for more information about meta-characters and regular expressions (Innovative uses *UNIX in a Nutshell* and John Muster's *UNIX Made Easy*).

^	beginning of a line or string
$	end of a line or string
^xx$	anchored expression – the string or line contains only xx
.	any character
..	two of any character
*	the single character that immediately precedes the * occurs any number of times (including none)
.*	any character occurring any number of times
[ ]	character class defining a single character
[0-9]	- within [] means a range of consecutive ASCII characters, here a single digit which could be any of 0 through 9
[^x]	^ within [] means not, here any character that is not x
+	preceding character occurs one or more times
?	preceding character occurs zero or one times (rarely used)
{}	specify the number of times the preceding character occurs
[a-z]{2}	two of any characters from a to z
( )	define strings
()$0	used in m2bmap files; the string defined by the () is given a numeric label (counting starts at 0) for data manipulation, e.g., ([0-9]{2})$0 uses a label of 0 for any two digit number
\0	used in m2bmap files, the string labeled 0 is placed in the context of a larger string that will be loaded into a field
|	default divider between elements in maps
\	escape or turn off the meaning of the following meta-character

STANDARD M2BTAB FILE EXTENSIONS

The extension of the m2btab file identifies the type of record being converted, for example, m2btab.asub loads subject authority records. Standard m2btab file extensions that you may encounter on your system are given below.

m2btab extension	Record type(s) to load
.a	Authority records from a bibliographic utility (interactive downloading only)
.amesh	MeSH authority records
.anam	Name and title authority records
.a sub	Subject authority records
.b	Bibliographic and item records from a bibliographic utility (usually OCLC) for interactive downloading and Z39.50 transfer
.batch	Bibliographic and item records via batch loading
.bcard	Bibliographic and checkin records with checkin cards
.bip	Books in Print
.bo	Bibliographic and order records from a bibliographic utility (interactive downloading)
.boven	Bibliographic and order records from a book vendor
.bta	Bibliographic and item records from initial data load
.catex	Bibliographic and item records from OCLC's CatExpress product
.click	Bibliographic and order records loaded via Quick Click
.conser	Bibliographic CONSER serials records from OCLC
.course	Course records
.fse	Full screen editor (bibliographic records)
.fse.i	Full screen editor (item records)
.fse.c	Full screen editor (holdings/checkin records)
.holdings	Bibliographic and holdings records in MARC 21
.net	Bibliographic and order records from netLibrary
.order	Bibliographic and order records via batch loading
.p	Patron records

m2btab extension	**Record type(s) to load**
.pcat	Bibliographic, order, and item records with invoice data from a book vendor (Extended Approval Plan Interface)
.rb	Bibliographic and item records from RLIN
.r	bo Bibliographic and order records from RLIN
.s	df B&T Link
.t	oc Table of Contents (TOC) data delivery from INN-View
.ts	Bibliographic and order records from third party vendor, e.g., Title Source records
.vendor	Vendor records

LOAD TABLES: THE M2BTAB FILES

INTRODUCTION

The load tables AKA m2btab files convert MARC records to the internal storage format of the Innovative system.

For step-by-step instructions on how to create new and edit existing m2btab files, see the HOW TO CREATE A NEW LOAD TABLE (M2BTAB) and HOW TO EDIT AN EXISTING LOAD TABLE sections in this manual.

DATA ELEMENTS

An m2btab entry consists of twelve data elements, separated by the vertical bar character. The twelve data elements are described below.

```
1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12
```

NOTE

If you encounter functions in the m2btab files on your system that are not described in this document, do not change them. Some functions are too complex to be changed without purchasing profiling services from Innovative.

Element 1: MARC Tag

The MARC tag(s) and indicators to be converted or an 'L' for Leader. The MARC tag can be represented by a number, a range of numbers, a comma-separated list of numbers or ranges of numbers, or a regular expression. This field should be preceded by the forward slash and caret ('/^') to be interpreted as a regular expression. If the indicators are not specified, all indicators will be loaded. When a range of MARC tags is used (such as 500-511 below), it’s not possible to specify the indicator values; regular expressions should be used instead of a range if indicator values are specified. Some examples are:

500-511	Take data from fields 500 through 511
L	Take data from the Leader
/^6...0	LC subject headings only (2nd indicator = 0)

/^6[0-5]..[01]	LC and LC Children's subject headings only (2nd indicator=0 or 1) from MARC tags 600-659
501,503,506-545	Take data from fields 501, 503, and 506 through 545

Element 2: New MARC Tag

Re-map data from the incoming MARC Tag (Element 1) into a specified New MARC Tag during the load.

NOTE

Only a specific MARC tag can be entered here (not a regular expression). Also, to specify one indicator, BOTH indicators must be entered, e.g. to map a 092 MARC field to a 099 MARC field with a 1st indicator of 9, you must enter a space for the second indicator:

```
092|0999 |+|0|0|b|c|0|y|N|0|
```

Element 3: Subfield

The MARC subfield(s) from which the data is to be extracted. Subfield tags and the order in which they will appear in the Innovative variable- length Field Group Tag (Element 7) are determined by the following conventions:

NOTE

Unless otherwise specified (see below), subfields will be loaded in the order in which they occur in the incoming record.

%	no subfields (MARC tags 001-009)
+	all subfields in the order found in the incoming record
-xyz	all subfields except |x, |y, and |z
x>y	subfield |x in MARC field becomes subfield |y in III field
x:yz	extract data from subfields |x, |y, and |z. The ':' means extract the subfield |x data first. Subfields |y and |z are then extracted in the order in which they occur in the incoming record.

*** See next page for a continuation of Subfield. ***

M<subfield>

Creates a new variable-length field (as specified in Element 7, Variable-length Field Group Tag) for each occurrence of the specified subfield.

S<integer and subfield>

Extract only the specified occurrence of a subfield and store it in a new field. This is used when an incoming field has multiple occurrences of a subfield and you want to extract only one of the occurrences. Counting starts at 0, so for example:

S1a

would extract data from the second occurrence of subfield |a.

T Place the literal text string contained in Element, Special, into the record. The literal string starts with the two indicators for the field, followed by a '|' subfield delimiter, a subfield code, and finally the data. Multiple subfield codes may be entered. The MARC tag that is created is defined in Element 2, New MARC Tag. If a MARC Tag is specified in Element 1, the field will be created ONLY if the MARC tag exists in the incoming bib record. If Element 1 is not specified, the field will always be created when a new record is created.

*** See next page for a continuation of Subfield. ***

Examples:

`(c>a)de`	Get |c, |d, and |e in any order. If there is a |c, convert it to subfield |a.
`c>a:de`	Get |c and convert to |a; then get |d and |e in any order.
`a:2`	Get |a; then get |2.
`f:ac:e`	Get |f; then get |a and |c in any order; finally, get |e.

```
/^949 1||Mn|0|0|i|n|0|n|N|1|
```

Each subfield |n in a 949 MARC tag will create a separate n-tagged field in the output item record.

```
/^949 1||S1a|0|0|i|x|0|n|N|1|
```

Load the second occurrence of 949 subfield |a in the x- tagged field of the item record.

```
|655|T|0|0|b|d|0|y|N|0| 7|aElectronic books.
```

Insert this field in each bibliographic record

```
655 7|aElectronic books.
```

Element 4: Offset

Offset position from which to get the data in the field specified (i.e., 008 field or Leader). If the Subfield (Element 3) is a single subfield, then the offset refers to the beginning of the subfield; otherwise offset is ignored. The offset must be set to 0 when loading the entire field. Counting begins at 0.

Element 5: Number of Bytes

Number of bytes to take from the beginning of the Offset (Element 4) or Subfield (Element 3). This element may be used to limit sizes of fields. The number of bytes must be set to 0 when loading the entire field for variable-length fields, and when using m2bmap files to load fixed-length fields. Counting begins at 1.

Element 6: Record Type

The target Innovative record type in which to load the data. The choices are **b** for bibliographic, **o** for order, **c** for holdings/checkin, **a** for authority, **i** for item, and **p** for patron. The Record Type element needs to be consistent with Element 11, Pass Number.

Element 7: Variable-Length Field Group Tag

Field group tag in which to load the data. Each line in the m2btab can map data to a variable-length field or to a fixed-length field, but not to both.

This element should be a blank space when loading into a fixed-length field and the Fixed Field Number (Element 8) is non-zero.

NOTE

The Variable-Length Field Group Tag that you use must be valid in your system's TAGS OF VARIABLE-LENGTH FIELDS table. If it is not, please contact the Help Desk to have Innovative define the field for you. To see the valid field group tags on your system, follow this path:

M > MANAGEMENT information

I > INFORMATION about the system

C > CODES used

T > TAGS of variable-length fields

Each record type may have up to 26 unique variable-length field group tags. Variable-length field group tags are limited to lower-case alphabetic letters only, with a few exceptions – "1" for the bibliographic Locations field and "=" for the patron PIN field.

Element 8: Fixed Field Line Number

The fixed-length field in which to load the data. Each line in an m2btab can map data to a variable-length field or to a fixed-length field, but not to both. The Fixed Field Number should be 0 when loading into a variable- length field and the Variable-length Field Group Tag (Element 7) is non- blank.

*** See next page for a continuation of Fixed Field Number. ***

NOTE

Fixed-length fields are defined in an internal system table and are identified by their line number in this file. See the “FIXED FIELDS IN LOAD PROFILES” section above for the appropriate number for each field.

The *Innovative Guide and Reference* also gives fixed field line numbers (in the column labeled Field Number) when you follow the links for each record type on page no. 105775 (**Sierra/Millennium Reference | How Innovative Systems Store Information | Fixed-length Fields**).

Element 9: MARC Format

This data element determines whether the incoming field will be stored with MARC tagging and subfields. For fixed-length fields, this element should be set to 'n'. See also the **@marc** Global Variable Function below.

y Store data as MARC. Usually used for variable-length fields.

n Store data as non-MARC. Always used for fixed-length fields.

Element 10: Permanence

The Permanence flag determines how long data in this field is retained during interactive hardwired serial port interface downloading. Values are:

N Field is not retained, i.e., the field needs to be stored in every incoming record in order to be loaded.

G Field is retained for all subsequent records, until the next record containing the field (or containing a command line command) resets it. Mnemonic: **G**lobal. **Rarely used.**

NOTE

Any “G” values are ignored when records are downloaded using Innovative’s OCLC Networked Interactive Interface.

Element 11: Pass Number

The Pass Number determines when the line is read during the conversion. The base record type (pass 0) is read and created first, and subsequent pass numbers are read later. This element needs to be consistent with Element 6, Record Type, and with the Global Variable Function @link. Values are:

0	first pass for base record type
1	second pass for linked records generated by the @link function in Element 12, Special.
2	third pass for linked records generated by the second @link function in Element 12, Special (if more than one type of linked record will be created)

In the following example, item records will be created in the second pass:

```
@link="i:1:945"
          ^ pass
```

Element 12: Special

If the first character of this field is one of the three characters listed below, the corresponding predefined special routine is applied:

@	Global Variable Function
%	Special Processing Function
#	Command Function

An entry in the Special element that begins with any other character is a comment, unless T is used in Element 3, Subfield (see above). When Element 3 contains T, the Special element contains the data to be inserted. The literal string starts with the two indicators for the field, followed by a '|' subfield delimiter, a subfield code, and finally the text to be inserted. Multiple subfield codes may be entered. For example, Element 12 might contain the following when Element 3 contains a T:

```
00|aItems without bib records
```

GLOBAL VARIABLE FUNCTIONS (AKA TRIGGERS)

When m2btab is read, it is scanned for Global Variable Functions (GVFs), which assign values to specified global variables. These in turn control various aspects of how the marc2inn program processes the source record.

Global Variable Functions have the format @name("string") or @name="string", where name is the name of the function and string is its value. For many Global Variable Functions, setting the value to the NULL string ("") cancels its effect, although commenting out the line is preferred (see HOW TO TURN OFF AN INSTRUCTIONS IN A LOAD TABLE section in this manual).

Certain Global Variable Functions can be set with interactive commands. To allow a command line command to "trigger" a Global Variable Function, precede the Global Variable Function with a command function (e.g., #com="recs" @recs="b").

NOTE

If you encounter functions in the m2btab files on your system that are not described in this document, do not change them. Some functions are too complex to be changed without purchasing profiling services from Innovative.

@atab
Standard value: "a"

Specifies the suffix of the table to be used for converting authority records (e.g., @atab="a" uses m2btab.a; "a" is the default). This means that when an authority record is encountered when downloading bibliographic records, the loading program automatically switches to the table specified by @atab. When a non-authority record is subsequently encountered, the program switches back to the table specified by the -f flag in the marc2inn line in the m.marcload.local file. This command only applies to interactive downloads. It is not recognized in batch record loads.

If the table loaded by the @atab command does not contain the function "@main=a", the loading program will generate an error and will not switch to the alternate table.

@bldmarc
Standard value: "" (null)

Controls the construction of the call number field in RLIN transfers (not applicable for OCLC). The value specifies the MARC tag of the field into which the call number is to be constructed, followed by a list of the MARC tags and subfields in the incoming record from which to build the field. The call number tag must be separated from the field/subfield list by a colon and a space. Each incoming MARC field/subfield group in the list must be enclosed in square brackets (e.g., [090a] or [950ab]). For example:

```
@bldmarc="090: [950ab]"
```

This means that the call number is placed into MARC field 090 and is built from the |a and |b subfields of field 950 in the incoming record. Subfield |a of the 950 is copied to subfield |a of the 090 and subfield |b of the 950 is copied to subfield |b of the 090.

If the call number field is built from multiple MARC fields and/or subfields in the incoming record, they appear as a list, with an implicit AND applied to the elements. No spaces or punctuation can appear between the elements. For example:

```
@bldmarc="090: [090a][950bc][955d]"
```

This means that the call number field is built from the following components of the incoming record:

subfield |a of MARC field 090
AND subfields |b and |c of MARC field 950
AND subfield |d of MARC field 955

A conditional OR may be applied to two or more fields from the incoming record by separating the incoming MARC field/subfield groups with the vertical bar character ('|'). The first of these groups that is found in the incoming record is used for the call number field. For example:

```
@bldmarc="090: [090ab]|[950a]|[050c]|[955ab]"
```

This means that the call number field is built from the following components of the incoming record:

subfields |a and |b of MARC field 090
OR subfield |a of MARC field 950
OR subfield |c of MARC field 050
OR subfields |a and |b of MARC field 955

The order of these MARC field/subfield groups determines the preference order. In the above example, if an 090 is found in the incoming record, it will be used for the call number and nothing will be taken from the 950, 050, or 955 fields. Similarly, if no 090 is found, but a 950 field is found, the 950 is used for the call number and nothing is taken from the 050 or 955.

AND and OR logic may be mixed. In such a case, the OR portion(s) of the value must be enclosed in parentheses, as in the following example:

```
@bldmarc="090: [950d]([950ab]|[090ab])[950e]"
```

This means that the call number field is built from the following components of the incoming record:

subfield |d of MARC field 950
AND (|a and |b of MARC field 950
OR |a and |b of MARC field 090)
AND subfield |e of MARC field 950

@busy
Standard value: "y"

Specifies what action to take if a record is busy during an attempt to overlay. Possible values are:

y = accept, load a new record
n = reject, do not load a new record

@busy_file
Standard value: Not used

Specifies the name of a file to which new MARC records are written when the target record of an overlay is busy. This file may later be used as input to the loading program to do the overlay after the record(s) have been unbusied. For example:

```
@busy_file="filename.lmarc"
```

NOTE

It is important that your busy file name has the same extension (*.lmarc) as the file you originally loaded so that it will appear in the list of files to load.

If the specified file already exists, new MARC records are appended to it. The **@busy** trigger should be set to "n" to create this file.

@call_sspace
Default value: "n"

Use call_sspace="y" to strip spaces before a period in a call number.

@callnum
Standard value: "nnnny"

This function consists of five "on/off" flags which control various aspects of call number construction. Its primary use is to determine when and how to add pre-stamps or post-stamps to call numbers. Change the standard value only if pre-stamps and post-stamps are used. This function is frequently found but rarely used in load profiles.

"Input stamps" are strings in front of or behind the four-character holding symbol, which is listed in the OCLC 049 field in the MARC record. "Auto stamps" are specified in the holding symbol table entry for the given holding symbol. A flag is switched "on" by a 'y' or a 't' in the appropriate position in the string; an 'n' or other character switches it "off". An example is:

```
          123456
@callnum="yyyyy090"
```

First Flag. If there is no call number in the MARC record, assemble a call number from auto stamps (derived from the holding symbol table).

Second Flag. Place input stamps in the call number field. The order of placement is determined by the next two flags.

Third Flag. Put auto pre-stamp first when prepending an auto pre- stamp and an input pre-stamp to the beginning of the call number. Otherwise, the input pre-stamp goes first.

Fourth Flag. Put auto post-stamp first when appending an auto post- stamp and an input post-stamp to the end of the call number. Otherwise, the input post-stamp goes first.

Fifth Flag. Call number is stored in MARC format. If a number follows, use that as the MARC tag; otherwise, use the tag specified in the holding symbol table for the given holding symbol.

Optional Element. MARC tag for call number when storing in MARC format.

@cdate
Standard values: Batch load: "n"
Interactive download: "y"

Determines which date is entered into the CAT DATE field in the bibliographic record. When @cdate="y", the system date is used. When @cdate="n", the date in the new record default template specified by @dflt is used. A catalog date keyed on the command line (#com="ct") will override this setting.

The CAT DATE prints on the New Headings Report and can assist in authority control when the system is set to display only bibliographic records with completed CAT DATE fields in the New Headings Report.

@clsi
Standard value: "n"

If @clsi="y" up to 6000 bibliographic record numbers are written to a circular file for transfer to a foreign circulation system.

@comline

```
/^949  ||a|0|400| | |0|n|G|0|@comline
```

In this example, the 949 field (with blank indicators) is defined as the command line. All of the commands must go into subfield a, following the formatting that is discussed in the *Innovative Guide and Reference* starting on page no. 101512. The command line may be up to 400 characters long. This line should be commented with a # symbol if it should not be used during the data load (see HOW TO TURN OFF AN INSTRUCTIONS IN A LOAD TABLE section in this manual).

@dflt
Standard value: "" (null)

Specifies the default template for each record type that should be used to insert fixed-length field data if the incoming record has no value for a fixed-length field. If a default template is not specified in the load table, the first default template listed in the system is used.

NOTE

In Millennium, you can view, edit and create default templates via the **Admin | Settings | Record Templates** tab.

In the character-based system, follow the path:

A > ADDITIONAL system functions

A > ALTER system parameters

D > DEFAULTS for new records

When creating default templates, it's a good idea to specify values in all the fixed-length fields, even if it's a "-" for a null value. If the incoming record has no value for this fixed-length field and the fixed-length field is left blank in the default template, it may not be possible to edit that fixed-length field after the record has been loaded.

Example:

```
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt="bib,order"
```

In this example, bibliographic records will be loaded using the default template named "bib" and order records will be loaded using the default template called "order."

NOTE

The @dflt trigger can be altered for any load table in the character-based system via this menu path:

A > ADDITIONAL system functions

A > ALTER system parameters

S > SYSTEM codes

O > Set system OPTIONS

D > DATABASE maintenance

23 > Change defaults used in record loading*

*The line number for changing defaults may vary at your library.

An "X" identifies the default template currently in use for the selected load table.

@diac
Standard value: "" (null)

Determines whether to use an external table to replace diacritics with Roman letters (their nearest ASCII equivalent) in the specified record types.

A "y" (@diac="y") means use the system table that translates diacritics to their nearest ASCII equivalent. This setting is strongly discouraged, as it irretrievably removes the diacritics from the data.

A NULL string (@diac="") means do not use the mapping translation table and retain the diacritics in the form {nnn}.

@diac_sub_table
Standard value: Not used

Specifies the suffix of a diac.* file to use for translating diacritics which have the high bit set into a curly braced form which can be interpreted by the Innovative system.

To load UTF8 encoded data:
```
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@diac_sub_table="utf8"
```

To load Windows-1252 encoded data:
```
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@diac_sub_table="win1252"
```

@disp
Standard value: "n"

When true, prints default fields and settings on the logging printer. Should always be false (@disp="n") and only triggerable by the command "disp" (#com="disp").

NOTE

The permanence flag (element 10) should always be set to “N” for the @disp line.

@holdsymb
Standard value: "049a"

The MARC tag and subfield that contain the holding symbols referred to in the m2bholdsymb table. This field is most often used when downloading records from OCLC. Common values are:

OCLC	`@holdsymb="049a"`

The holding symbol is used to determine the bibliographic record's call number and location code using the m2bholdsymb table. If your library is not using m2bholdsymb to derive locations, call numbers and/or subject headings, and if there is no holding symbol field in the incoming record, the @holdsymb trigger MUST be set to null (@holdsymb="").

NOTE

In the character-based system, the holding symbol table can be viewed via this path:

M > MANAGEMENT information

I > INFORMATION about the system

C > CODES used

H > HOLDING symbols

1 > OCLC Holding Symbols (m2bholdsymb)

Contact the Customer Services Help Desk to make changes to this table.

Administrators who have had Advanced System Access and Administration (ASAA) training can edit their holding symbol table in Millennium by changing the current mode to **ASAA** and choosing the drop-down menu **Select file | Holding Symbol | Select**.

For more information about the Holding Symbol table, see page no. 105807 of the *Innovative Guide and Reference*.

@holdsymb_bn_command
Standard value: Not used

If this trigger is set to "y" and the @holdsymb trigger is used, the content of the field specified with the **@holdsymb** trigger is ignored and the location code specified with the '*bn' command will be used for all subsequent records. By default (i.e., without this trigger being set to "y"), the location code specified with the '*bn' command is used only for the record in which it appears and the field specified with the **@holdsymb** trigger is used.

@holdsymb_first
Standard value: Not used

If set to "y" and the call number field selected from m2bholdsymb occurs multiple times, this trigger causes selection of the first call number. By default, the last call number is selected.

@holdsymbtab
Standard value: Not used

Specifies an alternate m2bholdsymb file (see NOTE below), which must be named m2bholdsymb.ext, where ext is a unique file extension. The syntax is:

```
@holdsymbtab="ext"
```

NOTE

If you need to use an alternate mr2bholdsymb file, you will need to ask the Customer Services Help Desk to create it for you.

@init
Standard value: "n"

When set to "y", re-reads the m2btab table and all templates. Should always be false (@init="n") and only triggerable by the command "init" (#com="init"). This function has the effect of resetting all global variables to their default values (i.e., clearing remembered command line values).

NOTE

The Permanence flag (element 10) should always be 'N' for the @init function.

@item
Standard value: `#com="i/a"@item`

Causes the creation of an item record from data contained in a special MARC field. The tag of the MARC field to use is given by the @link Global Variable Function. There are no "values" for @item; rather, this function is triggered exclusively by the command line command "i" (#com="i/a") which is used to specify the barcode of the item record to be created (e.g., i=+709384).

See also **@itemprefix, #com="i/a"**, and **@link**.

@itemprefix
Standard value: "" (null)

Automatically supplies the beginning digits of barcodes keyed at the command line with the "i" command (#com="i/a"). It allows the user to avoid keying all 14 digits of a barcode. For example, if the string is set to "31306," only the nine remaining numbers of the barcode, preceded by a '+', need to be keyed on the command line. The item prefix value can be changed by the command line command "ip" (#com="ip").

Spaces are allowed in the pattern string (e.g., "3 1306").

@ldx
Standard value: "" (null)

Specifies a MARC field for storing the Innovative record number for specific record overlays. For example, @ldx="907" will instruct the software to look in field 907 for the Innovative record number. If found, the incoming record will replace the existing Innovative record.

Overlay of the bibliographic record via @ldx can be accomplished by specifying the MARC tag that contains an Innovative bibliographic or order record number. Any MARC field from 010-999 may be used since the record number must be in subfield |a, although 907 is the most common for bibliographic record numbers, and 935 is most often used for order record numbers. Make certain that the MARC field you choose does not have any other uses. MARC fields 001-009 cannot be used since these fields do not have subfields.

The record number must be stored in subfield |a of the source tag and must include the "." prefix and Innovative check digit (e.g., 907 |a.b1000532). If field 907 is not present or does not include an Innovative record number, the system uses the value defined with **@ov_rec_number**.

When not using the function, be sure to use the standard value of
`@ldx=""`.

NOTE

In PromptCat load tables, m2btab.pcat, this Global Variable Function often appears as @ldx=”935” because the 935 tag in the incoming record contains the Innovative order record number. This load table is used within the Acquisitions module. When a match is found on the order record number, the @ov_action trigger determines whether the bibliographic record is overlaid.

@leader_utf8

If set to "y", the loading program checks byte 9 of the MARC leader. According to the MARC 21 format, leader byte 9 is set to "a" in UTF-8 encoded records and to " " (blank) in MARC-8 encoded records. If the loading program finds "a" in leader byte 9, the software translates the incoming UTF-8 to its Unicode value.

See also **@diac_sub_table**, which can be used in systems without Unicode storage.

@link
Standard values:
Item records: "i:1:945"
Order records: "o:1:#1"

Governs the creation of one or more linked records. Format is:

```
rectype:pass_no:[MARC tag][#link_ct][:tag[itag]]
```

pass_no is the number (1, 2 or 3) corresponding to the Pass Number (Element 11 above). If the optional MARC tag is not specified in @link,

#link_ct is the total number of linked records to create; data within these linked records are loaded using load table entries that match the Pass Number. For example:

```
@link=”o:1:#1”
```
creates one linked order record from Pass 1 entries

If the MARC tag is specified, one linked record is created for each occurrence of that MARC tag in the record (link_ct, if present, is ignored). The MARC tag can be a regular expression. For example:

```
@link=”o:1:960”
```
creates one linked order record for each 960 field

```
@link=”i:1:945”
```
creates one linked item record for each 945 field

```
@link=”c:1:852”
```
creates one linked holdings/checkin record for each 852 field

To create a generic item record for each bibliographic record, item record creation can be triggered by the presence of a 245 tag in the bibliographic record. Item record fixed-length field data can be specified in the item record default template defined in the **@dflt** trigger.

```
@link="i:1:245"
```

creates one linked item record for each 245 field.

The optional element :tag[itag] specifies that the new linked record should be discarded if the specified tag or tag/itag already has an index entry for another record in the database. For example:

```
@link="i:1:945:b"
```

This will check the barcode index and prevent a new item record from being created if an item record with the same barcode already exists in the database. This prevents duplicate items from being created. Bibliographic records and other item records will still be loaded; the discarded item record will appear in the Error report for in the Record Loading Statistics.

To temporarily cancel this trigger, specify its value as the NULL string (i.e., @link="") or comment out the line (see HOW TO TURN OFF AN INSTRUCTIONS IN A LOAD TABLE section in this manual).

@locmerge
Standard value: "y"

When set to "y", merges bibliographic locations when overlaying a bibliographic record. If **@ov_action="a"** (see below), the locations from the incoming record are merged with those in the existing record: the incoming bibliographic location codes are added to the end of the variable-length LOCATIONS field (field group 1), and the fixed-length LOCATION field is set to “multi.”

If @locmerge is set to "n" or is absent, the bibliographic location codes in the existing record will disappear when the record is overlaid, replaced by the bibliographic location code(s) in the incoming record.

If variable-length field '1' is used as one of the values of the **@ov_attach_insert** trigger (see below), the behavior of that trigger with respect to bibliographic locations will override the @locmerge trigger.

@m2b_multifund

Standard value: Not used

Specifies the MARC tag and, optionally, subfields in the incoming record that contain data used by the record load process to create multiple copy, location, and fund fields in order records. The value of this trigger is in one or four parts, as shown below:

```
@m2b_multifund="MARCtag[:Copy,Location,Fund]"
```

For example:

```
@m2b_multifund="998:o,t,u"
```

At a minimum, the MARC field must be specified. If the subfields are not specified, the following subfields are used as defaults:

```
@m2b_multifund="988"
```

Copy Subfield	o
Location Subfield	t
Fund Subfield	u

The copy subfield contains a count of the total number of copies for the order. The data in the location subfield and fund subfield specify how this total number of copies is partitioned among the locations and funds in a comma-separated list of location or fund codes along with the number of copies in each.

Each location or fund code is separated from the number of copies for that code by the forward slash character. For example, if 8 copies were being distributed to four locations, the contents of the location subfield might look like the following:

```
north/2,south/1,west/3,east/2
```

Similarly, if these 8 copies were charged to two funds, with 3 copies charged to fund code "aaa" and the remaining 5 to fund code "bbb", the contents of the fund subfield would look like the following:

```
aaa/3,bbb/5
```

To summarize, if the @m2b_multifund trigger line were:

```
|||0|0| | |0|n|G|0|@m2b_multifund="998:o,t,u"
```

the MARC field in the incoming record could be:

```
998 |o8|tnorth/2,south/1,west/3,east/2|uaaa/3,bbb/5
```

@m2b_normalize_020
Standard value: Not used

When this function is set to "y" and an overlay is pending based on the ISBN number, the ISBN number in the 020 field is normalized in the incoming and target record before comparison.

The fields are normalized by stripping out all dashes, spaces, and any characters following the number itself. If the existing record is overlaid, the complete content of the incoming 020 field, including all dashes, spaces, and extra characters, is retained.

@main
Standard values:
Authority: "a"
Bibliographic: "b"
Patron: "p"

Identifies the base record type (the main record that is being converted). Possible values are "a", "b", or "p". If you change this, be sure to check the values of **@ov_protect** and **@title**.

@marc
Standard value: (see below)

Controls which record types may contain MARC fields. The record type must be set in this function if Element 9, MARC Format, will be set to "y" for any lines of that record type in the table.

Possible values are:

@marc="bic" record types b, i, and c can have MARC fields

@marc="bo" record types b and o can have MARC fields

@msg
Standard value: varies

Text of the message that prints on the logging printer (or logging window) in interactive interfaces or at the top of the screen in batch loads when a new m2btab table is read (e.g., "BIBLIOGRAPHIC records will be created").

@odate
Standard value: "y"

Determines whether the system date is put in the Order Date (AKA ODATE) fixed-length field when downloading an order record. Possible values: "y", "n".

@ov_action
Standard value: "o"

Determines how to process the incoming record depending on how many matches were found on the **@ov_tag** value. Possible actions are to INSERT the new record, OVERLAY on an existing record (observing protected fields), ATTACH any new linked records to the existing record (but do not overlay), or REJECT the new record altogether.

NOTE

See also the **@ov_attach_delete** and **@ov_attach_insert** functions, which require an ATTACH setting in @ov_action.
When using any of the **@ov_priority** functions (e.g., **@ov_priority_tag**, **@ov_priority_date**), the value of @ov_action must indicate an OVERLAY or the **@ov_priority** functions will not be processed

Possible values are:

Value	# Matches	Action
“a”	0	INSERT
	1	ATTACH (“match and attach”)
	2 or more	INSERT
“b”	0	REJECT
	1	ATTACH (“match and attach”)
	2 or more	REJECT
“c”	0	INSERT
	1	ATTACH (“match and attach”)
	2 or more	REJECT
“d”	0	INSERT
	1	OVERLAY – NO LINKED RECORDS CREATED
	2 or more	INSERT
	(RLS 2009B_1.4 & Later)	

“e”	0	INSERT
	1	ATTACH (“match and attach”)
	2 or more	ATTACH to first match found
“o”	0	INSERT
	1	OVERLAY
	2 or more	INSERT
“p”	0	INSERT
	1	OVERLAY
	2 or more	REJECT
“r”	0	INSERT
	1 or more	REJECT
“u”	0	REJECT
	1	OVERLAY (see NOTE below)
	2 or more	REJECT

NOTE

If attempting to overlay on a specific record number via the **@ldx** trigger, the **@ov_action**="u" setting will REJECT the new record if the record number is not found, while **@ov_action**="o" will INSERT a new record under these circumstances.

@ov_attach_delete
Standard value: Not used

Value is a list of variable-length field group tags (and optionally MARC tags) to be deleted from the database bibliographic record for an overlay initiated by **@ov_action**="a", "b", or "c". For example:

@ov_attach_delete="y(856)"

@ov_attach_insert
Standard value: Not used

Value is a list of variable-length field group tags (and in Release 2009B optionally MARC tags) to be inserted into the existing bibliographic record from the incoming record when an overlay is initiated by **@ov_action**="a", "b", or "c". The field group tags must be listed in alphabetical order. For example:

@ov_attach_insert="n(590)n(591)v(995)y(035)"

If the variable-length field group tag '1' (LOCATIONS) is used as one of the values for this trigger, fixed-length field 26 (LOCATION) in the existing record is overlaid with the locations from the incoming record, completely replacing any existing locations in the field. If variable-length field '1' is NOT used with this trigger, the value of fixed-length field 26 is determined by the **@locmerge** trigger. If **@locmerge**="n" or is absent, the locations in the incoming record are overlaid with the locations from the existing record. If **@locmerge**="y", the locations in the existing record and the incoming record are merged.

See also the description of the **@locmerge** trigger.

@ov_priority
Standard value: Not used

This routine sets the priority of encoding level values contained in byte 17 of the MARC Leader of incoming bibliographic records. This is consulted when a matched record is to be overlaid based on **@ov_action**. The format is a list of colon-separated values in descending order of priority, reading from left to right.

NOTE

Use of **@ov_priority** varies widely depending upon the nature and source of the incoming data.

The syntax is:

@ov_priority = " :1:5:8:u:z"

Here, a blank is the highest priority, followed in order by 1, 5, 8, u, and z. If either the existing record or the incoming record has codes that are not in **@ov_priority**, or if the existing record has no encoding level (i.e., no 008 MARC field), then the existing record is overlaid. If both records have encoding level values that are found in the

@ov_priority list, then the existing record is overlaid only if the incoming record's priority is equal to or greater than the existing record's. The value of **@ov_priority_action** determines the action to take if the incoming record's priority is less than the existing record's.

@ov_priority_action
Standard value: Not used (defaults to "r")

Specifies the action to be performed if the incoming record's encoding level (as determined by **@ov_priority**) is less than the record to be overlaid.

Possible values are:

a	Attach the record
i	Insert the record
r	Reject the record

If this function is not in the m2btab file, then the value is set to "r". The **@ov_action** value must indicate an overlay, for instance **@ov_action**="o"

@ov_priority_type
Standard value: Not used (defaults to "d")

Specifies the action to be performed based on the results of **@ov_priority_tag**'s comparison of priorities of the incoming record versus the record to be overlaid.

Possible values are:

a	Overlay if incoming priority is less than existing priority
b	Overlay if incoming priority is less than or equal to existing priority
c	Overlay is incoming priority is equal to existing priority
d	Overlay if incoming priority is greater than or equal to existing priority
e	Overlay if incoming priority is greater than existing priority

@ov_protect
Standard values: see below

Authority records:

```
@ov_protect="a=F112-114V0123456789"
```

Bibliographic records:

```
@ov_protect="b=V023456789hy(962)k(970,971)n(972)"
```

Patron records:

```
@ov_protect="p=F48-50,54-56,95,96,99,101-105,122-125,
158,163,263,268-271,297Vbmxy0123456789="
```

List of fixed-length (F) and/or variable-length (V) fields to be protected when overlaying records. Normally, an overlay replaces fields in the database record with those from the incoming record.

Protecting a variable-length field causes new fields with the same tag to be added rather than overlaid on the old field.

Protecting fixed-length fields simply ignores the corresponding fields in the incoming record.

NOTE

If you are copying @ov_protect instructions from this manual, the Load Profile wiki, or from a load table another library sent you, it is especially important to check the field group tags in @ov_protect and in Element 7 of the load table to confirm they match your library's profile. *They may be different on your system!* If you protect the wrong field group tag in @ov_protect, you are at risk for losing data.

Syntax:

```
"<rectype>=V<variable-length fields>F<fixed-length fields>"
```

The 'V' and 'F' sections may be in either order, although only variable- length fields may appear after the 'V' and only fixed-length fields may appear after the 'F'.

NOTE

Fixed-length fields must be referred to by their line numbers in the internal file where they are defined. See FIXED FIELDS IN LOAD TABLES section in this manual.
While range notation, such as "48-50", is allowed for fixed-length fields, it is NOT allowed for numeric variable-length fields. To protect a range of variable-length fields, they must each be explicitly listed (e.g., "V2345678" for all numeric variable-length fields from '2' to '8').
If a fixed-length field is not loaded, it should be protected from overlay.

The following example protects variable-length fields 8 (HOLD) and h (LIB HAS) and fixed-length field 28 (CAT DATE):

```
@ov_protect="b=V8hF28"
```

NOTE

If the syntax of an **@ov_protect** argument is incorrect, then the record loading program will abort the load.

The usual value for bibliographic records is

`@ov_protect="b=V023456789hy(962)k(970,971)n(972)"` (i.e., all numeric variable-length fields except '1', plus the 'h' field, ‘y’ 962, ‘k’ 970 and 971 and ‘n’ 972).

NOTE

It is possible to protect the '=' variable-length field (PIN field) in patron records. For example:
p=F48-50,54-56,95,96,102-103V=mx89
See also the **%encryptpin** Special Routine.
The @ov_protect trigger can be set via the Database Menu option in the character-based system, but some values such as the PIN will not show up in this display.

A > ADDITIONAL system functions
A > ALTER system parameters
S > SYSTEM codes
O > Set system OPTIONS
D > Database maintenance
22 > Edit overlay protection list

The fields to protect may be specified by MARC tag in addition to the ability to specify by field group tag. The protected field(s) may be specified as a comma-delimited list enclosed in parentheses in the following form:

<rec type>=V<field group tags>(MARCTAG[,MARCTAG,...][:code])

MARCTAG may include indicators and wildcards (specified with periods). For example:

```
b=V78c(050,090,092)hd(6...7)
```

This example will protect field groups 7, 8, and h; MARC tags 050, 090, and 092 if they are in the c field group; and all 6xx fields in the d field group whose second indicator is '7' in both the incoming and the database record (i.e., the record that is to be overlaid).

The optional ':code' parameter specifies the action to perform when the protected tag is found in the record that is about to be overlaid. The absence of either of these codes in the @ov_protect option will cause both the incoming fields to be added and the existing fields to be retained in the record.

Valid codes are:

'd'	Does not load fields from the incoming record if there is any field in the existing record in the same field group tag. If there are no existing fields, then loads the incoming fields. If there are no incoming fields, then protect the existing fields.
'k'	Delete fields in existing record if there is any field in the incoming record in the same field group tag. If there are no existing fields, then load the incoming fields. If there are no incoming fields, then protect the existing fields.

Some examples of the use of ':code' are:

```
b=Vc(050:d)
```

This will protect any 'c'-tagged MARC 050 fields in the existing bibliographic record (the one that is to be overlaid). In addition, it will not load any 'c'-tagged MARC 050 fields from the incoming record. Note that the specified MARC tag applies to BOTH the incoming AND the existing record.

```
p=Vb(:k)
```

This will only protect the 'b'-tagged fields if the incoming record does not have a 'b'-tagged field for patron records. Otherwise, the incoming 'b'- tagged field overlays the existing 'b'-tagged field in the record.

The following table illustrates what would happen on the system if a NOTE field (field group tag is “n”) is not protected, is protected with one of the special codes, or is protected with no special code.

PROTECTION	INCOMING DATA	TARGET DATA	RESULT
No protection Field “n” not in @ov_protect	No data for NOTE	NOTE = Target	No NOTE field
	NOTE = Incoming	NOTE = Target	NOTE = Incoming
	NOTE = Incoming	No data	NOTE = Incoming
Field “n” in @ov_protect with no "colon code"	No data for NOTE	NOTE = Target	NOTE = Target

	NOTE = Incoming	NOTE = Target	NOTE = Incoming
			AND
			NOTE = Target

	NOTE = Incoming	No data	NOTE = Incoming
Protection "D" Field “n” in @ov_protect with "colon d" (:d)	No data for NOTE	NOTE = Target	NOTE = Target
	NOTE = Incoming	NOTE = Target	NOTE = Target
	NOTE = Incoming	No data	NOTE = Incoming
Protection "K" Field in @ov_protect with "colon k" (:k)	No data for NOTE	NOTE = Target	NOTE = Target
	NOTE = Incoming	NOTE = Target	NOTE = Incoming
	NOTE = Incoming	No data	NOTE = Incoming

@ov_rec_number
Standard value: "r"

Specifies the action to be performed if an overlay is done on an Innovative record number via the **@ldx** trigger and the overlay fails. Also specifies the action to take if an overlay fails due to a bad check digit in the record to be overlaid. Possible values are:

"**i**" Insert the record

"**r**" Reject the record

"**t**" Use the values of **@ov_tag** and **@ov_action** to attempt to overlay the record.

@ov_tag
Standard values: see below

Specifies how to overlay existing records with the incoming record. The command line command "ov" (#com="ov") is often used to trigger @ov_tag. The overlay is always rejected if a record to overlay is not found.

Possible Values:

field group tag[index tag]

Overlay records with matching data in the specified variable-length field group tag (e.g., @ov_tag="o" means overlay bibliographic records on a bibliographic utility number). The field group tag must be indexed. Include the index tag only if it is a different letter from the field group tag. For example,

@ov_tag="fi"

means overlay on those fields tagged as 'f' and indexed in the 'i' index.

field group tag[index itag]:field group tag[index tag]:...

Hierarchy of matching tags separated by colons. Same as above except that incoming record tags are compared against each member of the list until a match is found or until the end of the list is reached. If a match is found, @ov_action determines the result as before. For example:

@ov_tag="o:i:l"

@ov_tag="o:gi:l"

@ov_tag="!"

Do not overlay, look for duplicates (based on the indexed form of the full title in the first 245 tag). Adds the bibliographic record to the database and sets the STATUS of the order record to "1." Reports any duplicates found.

record number

Overlay on specific Innovative record number. For example,

```
@ov_tag=".b1003113"
```

NOTE

See also the **@ldx** trigger, which overlays on the Innovative record number when the record number is stored in a MARC tag (usually 907) in the incoming record.

@ov_tag=" " (space)

Turns off overlay when **@ldx**="". Records will be loaded as new. See HOW TO LOAD RECORDS AS NEW section in this manual.

field group tag(MARC tag)index itag

Overlays are allowed on the combination of Innovative variable-length field group tags and MARC tags. The MARC tag(s) (mtaglist) are enclosed in parentheses following the one-character field group tag and may contain regular expressions. An index tag may also be specified immediately following the list of MARC tags, if the index tag letter is different from the field group tag. For example:

```
@ov_tag="o(019)l"

@ov_tag="i(020)"
```

Multiple field group tag/MARC tag combinations are separated by colons. For example:

```
@ov_tag="o(001):f(035)o:i(020)"
```

This will attempt to overlay on field group tag 'o' (MARC tag 001) in index 'o', and if that fails, it will look for a match on field group tag 'f' (MARC tag 035) in index 'o', and if that fails, the system look for a match on field group tag 'i' (MARC tag 020) in index i.

This function directs the software to look at the index tag and MARC tag of the incoming record and only at the index tag of the target record. If there are duplicate index entries, the program may find more than one existing record that matches the incoming record. In that case, the program will not overlay the matching records.

NOTE

If the field group tag(MARC tag) is l(010), (i.e., the LCCN field), and there are at least eight numbers in the normalized form of the field, then only the first eleven characters of the field are used in the comparison. If there are fewer than eight numbers in the normalized form of the field, then the entire normalized 010 field is used in the comparison.

@ov_tag CONFIRMATION TESTS

Confirmation tests may be applied to specific field group tags in existing records to allow or prevent overlays based on data in the existing record. These tests are run before any of the **@ov_priority** tests are run.

Confirmation tests must be enclosed in square brackets. The syntax for specifying a test on a single field group tag is:

```
@ov_tag="tag[ov_test]"
```

```
@ov_tag="tag(mtaglist)itag[ov_test]"
```

The bracketed confirmation test follows the field group tag, the MARC tag list, and the index tag, if the latter two are present. For example:

```
@ov_tag="l(010)o[ov_245]"
```

In the case of multiple matches, the loading program checks each record until either a match is found or the last matching record is checked, whichever comes first. Therefore, only ONE record will be overlaid, which is the FIRST one that meets all criteria. If confirmation fails, the bibliographic record number of the LAST matching record is reported. In all cases, only one error is reported, with this format:

<n> Match(es) on tag <tag> <failed data> (with <bibrec#>) did not confirm on <name>, record <bibrec#> inserted / <245 field>

The <failed data> is the value of the match point, such as the OCLC number in the incoming record. For example:

1 Match(es) on tag o 29463723 (with b2221123) did not confirm on 245, record b221366 inserted / Poems

NOTE

If the loading program finds two or more matches, it will overlay on the first match that passes the confirmation test, and it will NOT give any indication that more than one match was found.

Confirmation test values:

ov_245

Compares normalized forms of the entire contents of subfields a, b, and p of the 245 title MARC tag in the existing and incoming records. When normalizing, the system does the following:

- strips all apostrophes
- translates each & to the string "and"
- replaces all other punctuation with a single space
- collapses multiple spaces to a single space
- replaces subfield delimiters with a space
- converts all characters to lower case

Example:

```
@ov_tag="o[ov_245]:i[ov_245]"
```

In this example, for each existing record whose field group tag 'o' matches that of the incoming record, subfields a, b, and p of the 245 title field are first normalized as indicated above. These normalized fields are then compared to the same normalized field in the incoming record and, if a match is found, the record is overlaid.

If the confirmation test fails, then, for each existing record whose field group tag 'i' matches that of the incoming record, the same sort of normalization and comparison is done for the existing record's 245 title field against that in the incoming record. If the fields match, then the existing record is overlaid. If this second confirmation test fails, the record is inserted and an error message is given.

ov_1xx

This confirmation test is used for authority record overlays only. If a match is found on the **@ov_tag**, the incoming authority record will overlay the existing authority record only if the field group tag of the 1xx fields in both records are the same. This prevents a name authority record from overlaying a subject authority record and vice versa. For example,

```
@ov_tag="oz[ov_1xx]"
```

ov_edition
ov_place
ov_year

Confirmation tests on any of the following pre-defined MARC tag/subfields can be entered in brackets following each field group tag.

The MARC tag/subfields, names, and normalization rule used are:

MARCTAG	SUBFIELD(S)	NAME
245	abp	ov_245
250	a	ov_edition
260	a	ov_place
260	c	ov_year

ov_verify_marc_tag

This confirmation test is TRUE if the incoming field's MARC tag is the same as the matched tag in the existing record. For example, if the MARC tag in the incoming record is "019" and the existing record has both a "001" and a "019" field with matching data, only the "019" field in the existing record is considered a match when this rule is used. In all cases, the action taken on the record is based on the setting in ov_action.

@ov_title
Standard value: Not used; defaults to "r"

When set to either "i" or "r", the existing record will be overlaid only if the first character of the normalized form of the first 't' field in the existing record is the same as that of the incoming record. If the record fails this test, then the new record is inserted (when **@ov_title**="i") or rejected (when **@ov_title**="r").

An optional parameter allows specification of the number of characters to compare. For example, **@ov_title**="r:4" compares the first four characters of the titles and performs the overlay only if the first four characters of both titles are equal.

If this function is not in the m2btab file, then the value is set to "r".

@password
Standard value: Not used

WARNING

If the specified password is invalid, the incoming record will be rejected.

Sets the user ID for entries in the Heading Reports and determines into which account or serial unit Order or Holdings/Checkin records will be loaded (for libraries that have multiple accounting and/or serial units). User IDs are derived from the "Passwords and authorizations" file. The value of this Global Variable Function should be the user's initials, not the password associated with the initials or login.

@poprint
Standard value: "y"

Indicates whether purchase orders are to be printed for downloaded order records. The "po" command (#com="po") is often used to trigger the **@poprint** function.

Possible values:

"n"	do not print PO
"p"	print PO, put 'p' to TLOC
"y"	print PO, do not put 'p' to TLOC

@pre_map
Standard value: Not used

Specifies a mapping file that is used to change fields in the input MARC records prior to record conversion. This function will not be covered in Load Profile Training. If your library needs this type of data manipulation, please contact the Customer Sales department for a price quote for profiling services.

@rdate
Standard value: "n"

Determines whether the system date is put in RDATE when downloading an order record. Possible values are "y" or "n".

@recs
Standard value: Must equal m2btab filename suffix

Specifies which m2btab table to use. Value equals filename suffix, for example:

`@recs="b"`	m2btab.b
`@recs="xyz"`	m2btab.xyz

If the table loaded by the **@recs** command contains the **@main** function and the value of the **@main** function is not "b", "p", or "r", the software will generate an error and will not switch to the alternate table.

@rep_call (or @repcall)
Standard value: "\\"

Specifies that if the given character is found in call numbers encountered in the field identified by the **@holdsymb** Global Variable Function, then it is to be replaced with a blank (i.e., " "). If the library uses **@holdsymb**, the **@rep_call** trigger should be used (if not, see the "**%replace**" Special Processing Function description below).

Note that the names **@rep_call** and **@repcall** are synonyms. They both do the same thing.

In most cases, the backslash character is specified, preceded by the escape character (also a backslash) as in the following example:

```
@rep_call="\\"
```

@speriod
Standard value: "n"

Set to "y" to remove periods at the end of fields, or to "n" to leave periods in place.

@test
Standard value: “n”

If true, updates are not made permanent. Possible values: "y", "n". The command line command "test" or "zz-0" (#com="test") often triggers @test.

@title
Standard value: "n"

Indicates whether records will require a title (MARC tag 245) in order to be valid. Applies only to "base" record types (i.e., bibliographic, authority, and patron records).

Possible values are "y" or "n". Should be false ("n") for authority and patron records. May be changed to “y” for bibliographic records.

@year_2000
Standard value: Not used

Controls conversion of 21st century dates for date fixed-length fields. Value is a two-digit number enclosed in double quotes (e.g., @year_2000="YY"). Dates in incoming records whose two-digit Year field is less than or equal to the specified value will be entered into the Innovative record as 21st century dates.

For example, if @year_2000="10", then dates in the range 01/01/00 through 12/31/10 would be entered into the Innovative records as 21st century dates (i.e., 01/01/2000 through 12/31/2010).

If the load table is for patron records (i.e., the @main trigger is set to "p") and @year_2000 is not given a value in the m2btab table, a default value of "90" will automatically be used.

SPECIAL PROCESSING FUNCTIONS

Special processing functions cause data to be filtered in a specified way when it is transferred from the MARC field to the corresponding Innovative field group tag. Special Processing Functions have the format %name="string" or %name("string"), where name is the function name and string is its value. These routines only apply to the line in the m2btab where they occur.

NOTE

If you encounter functions in the m2btab files on your system that are not described in this document, do not change them. Some functions are too complex to be changed without purchasing profiling services from Innovative.

%001

Standard values:

Authority:	`%001(start="!-~",char=" -~")`
Bibliographic:	`%001(start="1-9",char="!-~",valid="y")`
Patron:	not used

Special processing for 001 fields. Consists of five variables that control how the 001 field is processed. Each variable is assigned a value in double quotes. If a variable is not listed, its default value is used.

Example:

```
%001(bcode="",char="",skip="",start="",valid="")
```

bcode This setting is no longer valid. If it appears in a load table, remove it or set it to "bcode=n".

char Specifies the range of characters considered valid in the 001 field (i.e., "!-~" means all characters between ASCII 33 and ASCII 126 inclusive). Default is NULL.

skip The number of characters to skip over in an RLIN record number transfer (tag 001). A value of "4" will strip the "RLIN" text and "8" will strip "RLINXXXX" where "XXXX" is the library identifier. If this option's value is negative (i.e., for non-RLIN records), the system skips all characters until it reaches one in the range specified by start. Do not confuse with **valid**, which applies only to OCLC data. Default is "-1".

start The range of valid characters that can begin a record number from the 001 field (for example, "1-9"). This is ignored if a value is specified in **skip**. Also ignored if the %001 **valid** variable is set to "y". Default is NULL.

valid If true ("y"), strips "ocl7", "ocm", or "ocn" and any leading zeros from OCLC record numbers. Do not confuse with **skip**, which applies only to RLIN. Default is false ("n").

Note that when this is set to "y", any incoming 001 field that does not begin with "ocl7", "ocm", or "ocn" will be rejected. The loading program will generate an "invalid 001" error.

This same functionality applies to records in OCLC's Harvard (prefix "har") and PromptCat (prefix "pct") databases. Any values for the %001 **start** variable are ignored in this case.

When outputting records from the Innovative system, it is possible to reinsert the "ocm" prefixes that were removed during %001 processing. Contact the Customer Services Help Desk to enable this feature.

%008
Standard value: "y"

Indicates whether elements of the Leader in a MARC record are appended to the 008 field. If so, (%008="y"), Leader offset bytes 5-7 and 17-18 are appended to the 008 field as bytes 40-44, and byte 8 of the MARC Leader is appended to byte 45 of the 008 field in the Innovative record. Byte 8 of the Leader contains the archival control byte. (Byte 8 is appended to the end of the 008 because this feature was added in a later release of the Innovative software.) The length of the resulting 008 field in the Innovative record is 46 bytes. These values are moved back into the Leader when the record is output from the Innovative system.

%bracket
Standard value: "h" (for MARC field 245)

Value is a list of subfields that will be bracketed when placed into the bibliographic record. For example, |hgmd becomes [gmd].

%dedup_item_call
Standard value: Not used

If the **@holdsymb** Global Variable Function is used (i.e., not "NULL"), then if %dedup_item_call is set to "y" in an item record call number field entry, the item's call number (space, delimiters, and other codes are first removed) is compared to the bibliographic level call number in field group tag 'c'. If they are equal, the item call number will not be output.

%dedup_item_call_norm
Standard value: Not used

Same as **%dedup_item_call**, but the call numbers from the item and bibliographic records are first normalized before the comparison.

%encryptpin
Standard value: "y"
(PIN field in patron records only)

For the specified MARC field and single subfield, the software will encrypt data in that field and insert it into the PIN variable-length field group tag '='. This is used to load a PIN field into patron records. Note that an entry with this function will not load data into any other variable- length field. For example:

```
600||a|0|0|p|=|0|n|N|0|%encryptpin="y"
```

If the incoming data is from another Innovative system and the PIN field is already encrypted, use the **%noencryptpin** function to prevent double- encryption and to load the data as-is. Note that the system will always interpret data in the PIN field as encrypted once the record has been loaded. It is never possible to see the true value of the PIN in an Innovative record; you are always seeing the encrypted value.

%first
Standard value: Not used

If set to "y", causes only the first occurrence of a given MARC field to be loaded.

%foreign

Standard value: Not used

Value is foreign currency code and conversion rate, used to convert EPRICE fixed-length field in order records. Example:

```
%foreign="bpd0.440000"
```

If used, the following two lines must appear in the m2btab table in this order:

```
998||z|0|20|o|  |0|n|N|1|#com="ff"%foreign
998||s|0|20|o| |10|n|N|1|#com="ep"
```

At the start of conversion, you must then add commands for currency conversion to the command line. For example, to convert Spanish pesetas:

```
ep=10483.00;ff=spa.010;
```

The decimal point and trailing zeros in "ep=" are essential. Omitting them will cause the system to incorrectly read the value as 104.83. The order of these two commands is not important.

If the value is a 3-character currency code (e.g., %foreign="spa"), then the rate will be looked up in the foreign currency file. If the code is not in the system's foreign currency file, a message is logged to the error file.

%last

Standard value: Not used

If set to "y", causes only the last occurrence of a given MARC field to be loaded.

%map

Standard value: Not used

The name of a translation table (usual value "m2bmap.*"). Incoming MARC data is compared to a pattern in the translation table. If it matches, then the data is changed according to a substitution string, which can be either a literal string or an expression. Example:

```
%map=("m2bmap.loc")
```

See the TRANSLATION TABLES: THE M2BMAP FILES and HOW TO CREATE A NEW TRANSLATION TABLE (M2BMAP) sections in this manual.

%noencryptpin
Standard value: Not used
(PIN field in Patron records only)

For the specified MARC field and single subfield, the software will insert the PIN data "as is" into the PIN variable-length field tag '='. This is used to load a PIN field into patron records. Note that an entry with this function will not load data into any other field group tag.

The %noencryptpin function should only be used to prevent double encryption when the incoming data is from another Innovative system and the PIN field is already encrypted in the incoming record. Note that the system will always interpret data in the PIN field as encrypted once the record has been loaded. It is never possible to see the true value of the PIN in an Innovative record; you are always seeing the encrypted value.

If the incoming data for the PIN field is not already encrypted, use the **%encryptpin** function to encrypt the data while loading it

%replace
Standard value: Not used

Value is a character substitution list for the field. Format is ("char1","char2") where char1 is mapped into char2. char2 may be omitted to map to null string. Example:

```
%replace("/"," ")
```

Can be used to replace a string of characters. Example:

```
%replace("Revision","rev.")
```

%strip_blanks
Standard value: "y"

When set to "n", leading blanks will not be stripped from the incoming field. By default, leading blanks are always stripped from all fields.

%vendaddr
Standard value: Not used

Value is either a vendor code to be placed in the VENDOR fixed-length field of an order record or, if enclosed in square brackets, a vendor address to be placed in the VEN. ADDR. variable-length field. Examples:

```
%venaddr="ebsco"
%venaddr="[8000 Forbes Pl$Springfield, VA]"
```

COMMAND FUNCTIONS

These functions control what action is taken in response to command line or “xpo” commands when interactively downloading records from a bibliographic utility interface. The statement #com="cmd" (where cmd is any command given below) adds the specified command to an internal action table. When a command is entered on the command line, the corresponding data is inserted into the record in the MARC tag and subfield indicated by MARC tag and subfield (elements 1 and 3 in an m2btab table line). Typically, the MARC tag used is 999.

Fixed-Length Field Commands:

Those commands that set fixed-length field values (e.g., #com="b1") simply insert the values keyed at the command line into the corresponding MARC tag and subfield of the MARC record when it is first read. This value is transferred to the appropriate fixed-length field during record conversion.

Triggers:

Command line commands that are followed in the table by a Global Variable Function or a Special Processing Function name are known as "triggers" because they enable the user to activate a feature from the command line. For example:

```
#com="ov"@ov_tag=" "
```

The Command Functions in current use are listed below.

NOTE

For more information about the use of command line commands, please see the information beginning at page no. 101512 of the *Innovative Guide and Reference*.

COMMAND	RECORD	FIXED-LENGTH FIELD	TRIGGER	VALUES
ag	Item	AGENCY	-	-
at	Order	ACQ TYPE	-	-
b1	Bib	BCODE1	-	-
b2	Bib	BCODE2	-	-
b3	Bib	BCODE3	-	-
bl	Order	BLOC	-	-
bn	Bib	LOCATION	-	-
br	Order	LOCATION	-	-
c1	Order	CODE1	-	-
c2	Order	CODE2	-	-
c3	Order	CODE3	-	-
c4	Order	CODE4	-	-
cd	Order	CDATE	-	-
cl	Order	CLAIM	-	-
clsi	-	-	@clsi	y or n
co	Bib	COPIES	-	-
cop	Item	COPY #	-	-
cp	Order	COPIES	-	-
ct	Bib	CAT DATE	-	-
cy	Bib	COUNTRY	-	-
cz	Order	COUNTRY	-	-
dflt	Bib, Order, Item	-	@dflt	List of templates separated by commas
disp	-	-	@disp	y or n
ep	Order	E PRICE	-	-
fd	Order	FUND	-	-
fm	Order	FORMAT	-	-
i/a	Item	BARCODE	@item	Item barcode preceded by +. Any number of fixed-length field commands can follow, separated by slashes. This usage allows creation of multiple item records for a single bib. NOTE: i/a is used in the m2btab; the command line command is "i="
i1	Item	ICODE1	-	-
i2	Item	ICODE2	-	-
im	Item	IMESSAGE	-	-
init	-	-	@init	y or n
ins	Bib	-	@password	user initials
ip	-	-	@itemprefix	item barcode prefix
la	Bib	LANG	-	-

COMMAND	RECORD	FIXED-LENGTH FIELD	TRIGGER	VALUES
ln	Order	LANG	-	-
loc	Item	LOCATION	-	-
marc	-	-	@marc	b, o, a, I, c, r (y = all)
od	Order	ODATE	-	-
om	Item	OPACMSG	-	-
on	Order	ORD NOTE	-	-
ot	Order	ORD TYPE	-	-
ov	-	-	@ov_tag	overlay match point: field group/index tag, ! for duplicate checking, or record number
po	-	-	@poprint	y or n
pr	Item	PRICE	-	-
rc	Order	RACTION	-	-
rd	Order	RDATE	-	-
recs	-	-	@recs	extension of m2btab file name
rl	Order	RLOC	-	-
st	Order	STATUS	-	-
sta	Item	STATUS	-	-
test	-	-	@test	y or n
tl	Order	TLOC	-	-
ty	Item	ITYPE	-	-
v	Item	VOLUME	-	-
vd	Order	VENDOR	-	-
vl	Order	VOLUMES	-	-

TRANSLATION TABLES: THE M2BMAP FILES

The m2bmap file is a family of translation tables used during record loading to change the data contained in MARC fields in the incoming record to new data in the Innovative record. The m2bmap file is specified in the m2btab table using the Special Processing Function **%map**. Many m2bmap files can be added to a single m2btab table.

For step-by-step instructions on how to edit m2bmap files, see the HOW TO CREATE A NEW TRANSLATION TABLE (M2BMAP) section in this manual.

IMPORTANT NOTE

Do NOT begin lines in m2bmap with a # to de-activate or comment the line. Lines leading with a number sign (#) are interpreted as leading with a literal.

GLOBAL VARIABLE FUNCTIONS (AKA TRIGGERS)

Global variable functions are placed at the top of the translation table.

@delimiter=<char>

Change the default field delimiter to <char> from '|'. You must change the delimiter if the vertical bar is used in either the comparison or replacement expressions. For example, if you want to change the delimiter to a colon, include the line **@delimiter=:**

@case=true

When mixed-case data is processed through a translation table, the expressions are "normalized" to lower case for ease of comparison. Use this Special Instruction to cause comparisons and replacements to be case- sensitive and avoid unwanted case changes to the data. See "NOTES ON USING M2BMAP FILES: Avoiding Unwanted Case Changes," below.

@bar_subfield=true

Cause the Innovative system to replace the character '|' with the ASCII character 1F (hex) so the system will recognize the vertical bar and the following character as a subfield delimiter (i.e., |b would be interpreted as subfield b). This feature is rarely used.

@stop_on_map=true

Cause the Innovative system to stop checking the rest of the lines in the translation table once a match is found. By default each line of the incoming file will be checked against all lines of the m2bmap file. Using this feature ensures that unwanted data changes do not occur. For instance:

```
@stop_on_map=true
abc|123
123|xyz
```

If the @stop_on_map trigger is not used, the software could change the data from "abc" to "xyz" instead of from "abc" to "123".

DATA ELEMENTS

Data elements in m2bmap are normally delimited by the vertical bar character ('|'). To use a different delimiter, include the special instruction **@delimiter**=<char> at the top of the m2bmap file.

```
              1                      2

<comparison expression>|<replacement expression>
```

1. Comparison expression
A literal string and/or a regular expression to search for in the incoming data file. If a matching value is found, the data will be replaced according to the value in element 2, the replacement expression. See Syntax for Comparison Expressions below.

2. Replacement expression
A literal string and/or an expression specifying the value which will replace the expression found by the comparison expression search. See Syntax for Replacement Expressions below.

For instance:

```
STACKS|mnstk
```

The value STACKS will be replaced with mnstk, as in a LOCATION code.

```
CDROM|003
```

The value CDROM will be replaced with 003, as in an ITYPE code.

Syntax for Comparison Expressions

The expression to search for can be a literal string or it can have the syntax of a UNIX regular expression. To retain a portion of the incoming string in the replacement string, use a regular expression "wildcard" to match that portion. The wildcard expression must be enclosed in parentheses and must be followed by $n (where n is a number to identify the portion of the incoming string that is being retained). The $n comes immediately after the wildcard expression.

The syntax to use for a Comparison Expression differs, depending on whether the m2btab line that calls the m2bmap file has an entry in its variable-length field group tag (element 7). For m2btab lines with a field group tag entry (not loading to a fixed-length field), the Comparison Expression will be matched against the entire MARC field, including all subfields. When an m2btab line lacks a field group tag entry (i.e., when loading into a fixed-length field, as indicated by the number 40 in element 8 in the example below), the Comparison Expression will be matched against only the data in the specified subfield. For example, the following two lines will be treated differently:

```
852||b|0|0|c|i|0|n|N|1|%map=("m2bmap.1")
852||b|0|5|c|  |40|n|N|1|%map=("m2bmap.2")
```

In the first example, since the field group tag is present (element 7 contains the entry "i"), the specified m2bmap file will be passed the entire 852 MARC field with all its subfields, to which the comparison is applied. On the other hand, in the second example, no field group tag is present (element 7 is blank), so the specified m2bmap file will be passed only the data in subfield |b of the 852 field, to which the comparison is then applied.

As an example, assume you want to change subfield 'd' of a particular MARC field of the incoming records (note that, in this case, the delimiter for m2bmap must be changed using @delimiter=<char> to a character other than '|'). Further, assume that the m2btab line that calls this m2bmap file contains a field group tag entry in element 7. The entire MARC field will be passed to m2bmap, so that in the following comparison expression:

```
(.*)$0|d(.*)$1
```

the number 0 identifies the string containing all the characters that precede the subfield 'd' delimiter, and the number 1 identifies the string containing all the characters that come after the subfield 'd' delimiter.

Comparison expressions match on the longest possible string in the given line. For example, if the comparison expression

```
(.*)$0,(.*)$1
```

were applied to the following incoming string

Clause1, Clause2, Clause3

then string 0 would be “Clause1, Clause2” and string 1 would be “Clause3”. Note that the match was on the second comma, not the first. To make a string of everything up to the first comma, use the comparison expression

```
([^,]*)$0,(.*)$1
```

which stores everything-not-a-comma up to the first comma as string 0 and everything after the first comma as string 1. If this comparison expression were applied to "Clause1, Clause2, Clause3", then string 0 would be "Clause1" and string 1 would be "Clause2, Clause3".

Comparison expressions that make use of meta-characters in UNIX regular expressions will need to be “escaped” with the "\" character. See the section above on REGULAR EXPRESSIONS, and please refer to a UNIX guide or manual for more information about meta-characters (Innovative uses *UNIX in a Nutshell* and John Muster’s *UNIX Made Easy*).

The comparison expression may also contain non-alphabetic characters, which are entered in the C language "escaped HEX" format as shown below:

```
^\0xCA\0xC1$|<replacement expression>
```

This will match on any data whose first two characters have the hex value "CA" and "C1". The leading caret and trailing dollar sign is regular expression syntax specifying that the data must begin the line and that the line must match in its entirety.

Syntax for Replacement Expressions

Enter an expression to specify how to replace the line that was found by the comparison expression search. This expression can consist of literal strings and/or escaped numbers that identify which portions of the incoming line to retain (see comparison expressions above). The numbers that were preceded by a $ to represent each string in the comparison expression are preceded by a backslash (\) in the replacement expression, just as in regular expression substitution. Note that regular expression ampersand substitution (&) is not supported.

Continuing the subfield 'd' delimiter example from above, to insert the digits "19" after the subfield 'd' delimiter and before the data in subfield 'd', and to retain all other data intact, the replacement expression would be:

```
\0|d19\1
```

For example, the colon-delimited m2bmap containing the following:

```
@delimiter=:
@case=true (.*)$0|d(.*)$1:\0|d19\1
```

would translate the incoming string "10010|aRice, Anne,|d41 -" to "10010|aRice, Anne,|d1941-". A colon is used for the data element delimiter because '|' has been used in the comparison expression (the special instruction @delimiter=: needs to appear at the top of the file). Note that '|d' must be included as a literal in the replacement expression because it is not included in either of the wildcard expressions. You would also want to be sure that the special instruction @case=true was included to prevent "Rice, Anne," from being converted to "rice, anne,".

SAMPLE TRANSLATION TABLES (M2BMAP FILES)

Simple Location Code Translation Table, m2bmap.loc1

```
@stop_on_map=true
Juv Ref|mjref
South|s
SJUV|sjuv
ant|manth
JUV|mjuv
REF|mref
```

Location Code Translation Table with Regular Expressions, m2bmap.loc2

```
@stop_on_map=true
r[0-9][0-9]|ref01
```

(Replace location codes starting with "r" and followed by any two digits with "ref01")

Item Type Translation Table, m2bmap.itype

```
@stop_on_map=true
a|001
b|000
c|002
d|003
e|010
f|015
g|006
k|007
m|009
p|011
r|012
s|013
t|014
```

(Replaces alphabetic characters with numeric values)

Date Translation Table, m2bmap.date

```
@stop_on_map=true
([0-9]{4})$0([0-9]{2})$1([0-9]{2})$2|\1-\2-\0
```

(Rearrange the order of elements in a date field from yyyymmdd to mm- dd-yyyy)

NOTES ON USING M2BMAP FILES

Avoiding Unwanted Case Changes

By default, comparisons are normalized to lower case (thus making them case-insensitive). If the replacement expression is an expression rather than a literal string, the replacement expression will also be normalized to lower case, which can result in unwanted "case" changes being made to the database. Use the **@case=true** Special Instruction to avoid this problem. In the above example, if the Special Instruction **@case=true** were included, the m2bmap line,

```
(.*)$0|d(.*)$1:\0|d19\1
```

would map "Rice, Anne" in the incoming string "10010|aRice, Anne,|d41 -" to "Rice, Anne" (no change). If the Special Instruction **@case=true** were NOT included, "Rice, Anne" would be converted to "rice, anne" (all lower case).

Order of Lines in a m2bmap File

Each line of m2bmap is applied successively to each line of the incoming file, and if a line of incoming data is changed, succeeding lines of m2bmap will be applied to the changed data, which may result in unwanted changes (unless the special instruction **@stop_on_map** is set to "true", in which case, the process stops after the first match). Whether or not **@stop_on_map** is set, the order of the lines in the file is important.

The lines in the map should be ordered from the longest and most specific comparison expression to the shortest and most general comparison expression. For example, the following map file will NOT convert data in the desired manner:

BAD map file:

```
@stop_on_map=true
@case=true
main|stack
lmain|law
cmain|curr
mref|ref
```

If an incoming record contained the location "lmain", the above m2bmap file would match on "main" (which is contained in "lmain") and convert the location to "stack". To work properly, the entry for "main" in the m2bmap file must come after the other comparison strings that contain “main”. For example:

GOOD m2bmap file:

```
@stop_on_map=true
@case=true
cmain|curr
lmain|law
main|stack
mref|ref
```

Alternatively, you can indicate that the entry occupies the entire incoming line, by using ^ and $ to define the beginning and end of each line:

```
^lmain$|law
```

If ^ and $ are used for every possible incoming value (comparison string), then the lines can occur in any order.

Mapping Fixed-Length Fields

When mapping fixed-length fields, make sure that the byte size specified in m2btab is large enough to handle the possible values in the comparison expression. If not, you should increase this element in m2btab beyond the largest possible value, or set it to 0 to allow the entire string to be read.

For instance, if you want to translate item level status codes (single digit replacement expression) based on five letter codes in a MARC tag (five digits in the comparison expression), you will need to change m2btab from:

```
949||s|0|1|i|  |88|n|G|1|%map=(”m2bmap.status”)
         ^
```

to

```
949||s|0|5|i|  |88|n|G|1|%map=(”m2bmap.status”)
         ^
```

or

```
949||s|0|0|i|  |88|n|G|1|%map=(”m2bmap.status”)
         ^
```

This must be done, even if the replacement value will only be a single character. If not, the system will read a truncated version of the incoming data and use that as the comparison expression, which can result in unwanted data changes, or no data changes when they are desired.

NOTE

When translating to a fixed-length field, the replacement expression must be the fixed-length code, not its label.

Command Functions and Special Processing Function %map

If the m2btab table driving the conversion contains a #com Command Function in the form:

```
#com="<com>"%map("m2bmap.xxx")
```

then the software will look up the m2bmap.xxx table whenever the command line command <com> is given (e.g., **loc** for location or **ty** for type).

If the m2btab table contains a just the Special Processing Function %map:

```
/^940||l|0|5|b|  |26|n|N|0|%map=("m2bmap.loc")
/^945||t|0|0|i|  |61|n|N|1|%map=("m2bmap.itype")
```

then if the specified MARC field/subfield exists in the incoming record, the software will convert the data in that field as specified in the m2bmap.xxx table.

LOAD BUTTONS: THE M.MARCLOAD.LOCAL FILE

The m.marcload file controls the load options that appear when you launch Sierra or Millennium and choose **Data Exchange | Select Process: Load records via locally-created load profiles (local)**. All locally created load tables will be accessed via a single m.marcload file called m.marcload.local.

For step-by-step instructions on how to modify m.marcload.local, see the HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL) section in this manual.

GLOBAL VARIABLE FUNCTIONS (AKA TRIGGERS)

The first several lines of the file may contain the following triggers (these can appear in any order):

@HEADER=<header>

The text to present at the top of the screen when the **U > Load MARC records via a locally-created load profile** menu option is invoked. If not specified, there will be no screen title.

@MAXFILES=<n>

The maximum number of data files which match the pattern(s) specified in the menu lines. If this number is exceeded, the program will allow no other operation than REMOVE files. Therefore, you must be very careful that you do not exceed the number of data files specified. If you come within two files of MAXFILES (e.g., 8 files when MAXFILES=10), the system will warn you that you are approaching the limit.

@MAXBYTES=<n>

The maximum number of bytes in all data files combined. Standard value for <n> is 50000000 (50 million bytes or 50MB). As with MAXFILES, exceeding this limit causes the program to disallow any operation other than REMOVE files. Care must be taken not to exceed this limit. If you come within 2MB of this value, the system will warn you that you are approaching the limit.

@DIRNAME=<dir>

The directory that contains the data files. This should not be changed.

@MENULINES=<n>

Number of menu lines. The default is 5.

DATA ELEMENTS

The remainder of the file consists of literal menu lines and the corresponding programs and settings. Like the m2btab file, the data elements in the m.marcload file are separated by the vertical bar character, and lines can be turned off by commenting them out with an initial # character.

Each line has the following structure:

```
C > TEXT||program -flags & filenames
1   2   3 4       5        6
```

1. C >
   Unique character that the user keys in order to select this menu option.

2. TEXT
   Text describing the menu option. For example:

```
J  > Load CJK records
K  > LOAD a bib/order MARC file (.titlesource3)
M  > LOAD a bib/item MARC file (.coutts)
```

NOTE

It is important that you make the text for the menu options as descriptive as possible, so that the appropriate load profile can be chosen when records are loaded.

3. | |
   File name prompt; rarely used. The default prompt (if this field is blank) is "Enter file name:" To change this message, enter it in this element:

```
 U > UPLOAD records from PC|Assign what name to the
file?|
```

4. program
   The program that is executed when this option is selected. The Innovative programs that you will need to know about are:

marc2inn

The MARC record conversion program which loads preprocessed, blocked MARC records into the Innovative system.

> **NOTE**
>
> The only flag to the marc2inn program you should ever change is the –f flag, which specifies the suffix of the m2btab that is to be used.

marccopy

The program that converts/preprocesses unblocked MARC files into the blocked MARC format that can be loaded into the Innovative system. It will not be necessary to change the marccopy lines in the m.marcload.local file.

marcview

The program that allows you to view a MARC file. It will not be necessary to change the marcview lines in the m.marcload.local file.

xftp

The program that allows host-initiated FTP into and out of the Innovative system. It will not be necessary to change the xftp lines in the m.marcload.local file.

5. -flags

   Flags are settings that are passed to the program when it is executed. For example, in the following line, -i identifies the input file (the raw MARC file), and –o identifies the output file (the blocked MARC file):

   ```
   marccopy -u -i%#.lfts -o%%.lmarc
   ```

   The only flag that you will need to change in the m.marcload.local file is the –f flag in the marc2inn line, which identifies the m2btab table to use by its extension/suffix. For example, to load records through m2btab.local, the marc2inn line should look like the following:

   ```
   marc2inn -H"48,24,2" -x -forder -i -so %#.lmarc
   ```

6. filenames

   Filenames are the data files, and they are specified as follows:

   %S.ext

   The system will prompt for the file name, append the given extension and then check to see if it already exists. If it does, the program will get verification from the user before passing control over to the specified program and potentially overwriting the existing file. For example:

```
xftp get %S.lfts
```

%#.ext
Program will prompt for file by number from the presented list. It will then pass control over to the specified program with the selected file name as an argument. The following line will prompt the user to choose a data file that has an extension of .lfts:

```
marccopy -u -i%#.lfts -o%%.lmarc
```

%%.ext
Used to specify additional occurrences of the file name as determined by the %S or %# prompt flags. For example, the following line

```
marccopy -u -i%#.lfts -o%%.lmarc
```

tells the marccopy program to create a blocked file whose name is the same as the unblocked %# file; the only difference being the extension. If the unblocked file is datafile.lfts, the new blocked file is named datafile.lmarc.

THE LOAD PROFILE MAINTENANCE MODULE

This is the module you use to create new and update load tables (m2btabs) and translation/mapping tables (m2bmaps), and activate LOAD buttons in Data Exchange.

HOW TO EDIT AN EXISTING LOAD TABLE (M2BTAB)

1. All manipulation of load tables and translation tables will be done via the "Additional system functions" menu:

   A > ADDITIONAL system functions
   M > Read/write MARC records
   X > Load Profile maintenance

   You will see the following screen:

```
                    Load Profile Maintenance

E > Edit m2btab and m2bmaps
L > Edit m.marcload.local
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (E,L,Y,Q)
```

2. Select option **E > Edit m2btab and m2bmaps.**

3. Select option **P > Load PROFILES (m2btab)** to view a list of existing load profiles.

NOTE

The suffix of each m2btab table reflects the type of record being converted, e.g., m2btab.asub for subject authority records. See the section above on STANDARD M2BTAB FILE EXTENSIONS for a list of file extensions you may encounter.

4. Review the list of m2btab files and decide which m2btab should be updated. The list will look something like this:

```
                           M2BTAB Suffixes

01 > a
02 > anam
03 > asub
04 > b
05 > bo
06 > bta
07 > fse
08 > p

Key a number or
F > FORWARD               C > COPY an existing m2btab
J > JUMP                  G > GET a file using
P > PRINT                 S > SEND a files out of INNOPAC using FTS
Y > DISPLAY file SIZE & DATE
Q > QUIT

Choose one (1-98,F,J,P,Q,C,G,S)
```

5. When you have decided which table you want to update, enter its line number. This will present you with the following menu:

```
              M2BTAB File Validation and Maintenance

E > EDIT m2btab.b
V > VERIFY m2btab.b
U > UPDATE m2btab with changes
R > RESTORE original m2btab
M > marc2inn error MESSAGES
Q > QUIT
Choose one (E,V,U,R,M,Q)
```

6. Select **E > EDIT m2btab.<ext>**. This will put you into Innovative's full screen editor.

NOTES

You may find it helpful to press ^F to display the full menu of full screen editor commands.

Innovative recommends that you copy an existing line, rather than create a new line. By copying an existing line, it is less likely that syntax errors will occur due to missing or extra vertical bar characters. The easiest way to copy a line is to arrow down to the line you wish to copy, key D > DELETE line, then key U > UNDELETE line twice.

For more information on the full screen editor, please see 101370 of the *Innovative Guide and Reference*.

Alternatively, you may use FTS to download the table to an FTP server where you can use a different editing program. Then upload the file back into the Innovative system:

A > ADDITIONAL system functions M > Read/write MARC records
X > Load Profile maintenance
E > Edit m2btab and m2bmaps
P > Load PROFILES (m2btab)
S > SEND a files out of INNOPAC using FTS
G > GET a file using FTS

NOTE

If you plan to update the file using a different editor, use an editor that will not introduce any hidden or non-printing characters to the file. Please see record 101691 of the *Innovative Guide and Reference* for more information about FTS.

7. Update the load table according to the loading requirements you and/or technical services staff have defined.

NOTE

See reference section LOAD TABLES: THE M2BTAB FILE for a list of loading instructions including Global Variable Functions (@), Special Processing Functions (%), and other values that can be included in an m2btab file.

8. Once you have completed your changes to the load profile, key **^E > END** to exit editing mode.

Next verify that the load table is syntactically correct, key **V > VERIFY m2btab.<ext>**.

If no errors are found, you will see the message:

```
      Congratulations!! You have a perfect M2BTAB table!
```

If any errors are found, you will be presented with a screen that looks like the following, where **E** signifies that there is an error, and **W** signifies a warning:

```
m2btab.b  verify : ERRORS = 4 : WARNINGS = 3
   LINE          ERROR MESSAGE
   1 > E 88 Invalid RECTYPE data element 6 : y
   2 > W 88 Entry creates MARC field but rectype 'y' not in @marc
   3 > E 89 Invalid RECTYPE data element 6 : y
   4 > W 89 Entry creates MARC field but rectype 'y' not in @marc
   5 > W 94 Entry creates MARC field but rectype 'i' not in @marc
   6 > E 107 Invalid MTAG data element 1 :ABC
Key a number or
P > PRINT
Q > QUIT
Choose one (1-6,P,Q)
```

You will not be able to exit from this function and save the m2btab until you have resolved all of the errors. To resume editing of the m2btab, chose one of the line numbers from the error. This will put you back into the full screen editor. As of this writing, it is not possible for the full screen editor to take you to the line where the error occurred, so it will be necessary for you to move down through the file using the arrow keys.

NOTE

Verifying the syntax of your m2btab is very important. However, this will not protect your database against actions that are syntactically correct but nevertheless damaging to your database, such as forgetting to protect certain fields from overlay. This is why testing (see step 11) is crucial!

9. Once you have resolved all error messages, select the **U > UPDATE m2btab with changes** menu option to save your changes permanently.

10. If for some reason you want to revert to a previous version of your load profile, select the **R > RESTORE original m2btab** menu option.

11. Test the new load profile before putting it into production. See the HOW TO TEST A NEW M2BTAB AND/OR M2BMAP FILE section of this manual.

HOW TO CREATE A NEW LOAD TABLE (M2BTAB)

1. All manipulation of load tables and translation tables will be done via the "Additional system functions" menu:

A > ADDITIONAL system functions
M > Read/write MARC records
X > Load Profile Maintenance

You will see the following screen:

```
                     Load Profile Maintenance

E > Edit m2btab and m2bmaps
L > Edit m.marcload.local
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (E,L,Y,Q)
```

2. Select option **E > Edit m2btab and m2bmaps**.

3. Select option **P > Load PROFILES (m2btab)** to view a list of the existing load profiles.

NOTE

The suffix of each m2btab table reflects the type of record being converted, e.g., m2btab.asub for subject authority records. When creating a new m2btab file, it's important to give it a suffix (also called file extension) that is mnemonic. See the section above on STANDARD M2BTAB FILE EXTENSIONS for a list of file extensions you may encounter.

4. Review the list of m2btab files and decide which m2btab should be used as a pattern for the one you are about to create. (It is too difficult to begin a new m2btab from scratch, as syntax errors due to missing vertical bar delimiters or other data elements are inevitable.)

The list of m2btabs will look something like this:

```
                         M2BTAB Suffixes

01 > a
02 > anam
03 > asub
04 > b
05 > bo
06 > bta
07 > fse
08 > fse.i
09 > p

Key a number or
F > FORWARD                C > COPY an existing m2btab
J > JUMP                   G > GET a file using
P > PRINT                  S > SEND a files out of INNOPAC using FTS
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (1-98,F,J,P,Q,C,G,S)
```

5. When you have decided which table you want to update, select the **C > COPY an existing m2btab** menu option. The program will prompt you for the line number of the m2btab that you wish to copy. Then, it will prompt you for the new suffix:

Copy to what new suffix?:

After entering the new suffix for your copied m2btab file, you will see the following menu:

```
              M2BTAB File Validation and Maintenance

E > EDIT m2btab.bicoutt
V > VERIFY m2btab.bicoutt
U > UPDATE m2btab with changes
R > RESTORE original m2btab
M > marc2inn error MESSAGES
Q > QUIT
Choose one (E,V,U,R,M,Q)
```

6. Select **E > EDIT m2btab.<ext>**. This will put you into Innovative's full screen editor.

NOTES

You may find it helpful to press ^F to display the full menu of full screen editor commands.

Innovative recommends that you copy an existing line, rather than create a new line. By copying an existing line, it is less likely that syntax errors will occur due to missing or extra vertical bar characters. The easiest way to copy a line is to arrow down to the line you wish to copy, key D > DELETE line, then key U > UNDELETE line twice.

For more information on the full screen editor, please see 101370 of the *Innovative Guide and Reference*.

Alternatively, you may use FTS to download the table to an FTP server where you can use a different editing program. Then upload the file back into the Innovative system:

A > ADDITIONAL system functions M > Read/write MARC records
X > Load Profile maintenance
E > Edit m2btab and m2bmaps
P > Load PROFILES (m2btab)
S > SEND a files out of INNOPAC using FTS G > GET a file using FTS

NOTE

If you plan to update the file using a different editor, use an editor that will not introduce any hidden or non-printing characters to the file. Please see record 101691 of the *Innovative Guide and Reference* for more information about FTS.

7. Update the load table according to the loading requirements you and/or technical services staff have defined.

NOTE

See reference section LOAD TABLES: THE M2BTAB FILE for a list of loading instructions including Global Variable Functions (@), Special Processing Functions (%), and other values that can be included in an m2btab file.

8. Once you have completed your changes to the load profile, key **^E > END** to exit editing mode.

Next verify that the load table is syntactically correct, key **V > VERIFY m2btab.<ext>**.

If no errors are found, you will see the message:

```
Congratulations!! You have a perfect M2BTAB table!
```

If any errors are found, you will be presented with a screen that looks like the following, where **E** signifies that there is an error, and **W** signifies a warning:

```
m2btab.b  verify : ERRORS = 4 : WARNINGS = 3
   LINE          ERROR MESSAGE
   1 > E 88 Invalid RECTYPE data element 6 : y
   2 > W 88 Entry creates MARC field but rectype 'y' not in @marc
   3 > E 89 Invalid RECTYPE data element 6 : y
   4 > W 89 Entry creates MARC field but rectype 'y' not in @marc
   5 > W 94 Entry creates MARC field but rectype 'i' not in @marc
   6 > E 107 Invalid MTAG data element 1 :ABC
Key a number or
P > PRINT
Q > QUIT
Choose one (1-6,P,Q)
```

You will not be able to exit from this function and save the m2btab until you have resolved all of the errors. To resume editing the load profile, chose one of the line numbers from the error/warning screen. This will put you back into the full screen editor. As of this writing, it is not possible for the full screen editor to take you to the line where the error occurred, so it will be necessary for you to move down through the file using the arrow keys.

NOTE

Verifying the syntax of your m2btab is very important. However, this will not protect your database against actions that are syntactically correct but nevertheless damaging to your database, such as forgetting to protect certain fields from overlay. This is why testing (see step 12) is crucial!

9. Once you have resolved all error messages, select the **U > UPDATE m2btab with changes** menu option to save your changes permanently.

10. If for some reason you want to revert to a previous version of your load profile, select the **R > RESTORE original m2btab** menu option.

11. Add the new load button to Data Exchange. See the HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL) section of this manual.

12. Test the new load profile. See the HOW TO TEST A NEW M2BTAB AND/OR M2BMAP section of this manual.

HOW TO CREATE A NEW TRANSLATION TABLE (M2BMAP)

Since the structure of the translation table AKA m2bmap is simple, it is usually easier to create a new one from scratch than it is to copy an existing one.

1. All manipulation of load tables and translation tables will be done via the "Additional system functions" menu:

A > ADDITIONAL system functions
M > Read/write MARC records
X > Load Profile Maintenance

You will see the following screen:

```
                    Load Profile Maintenance

E > Edit m2btab and m2bmaps
L > Edit m.marcload.local
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (E,L,Y,Q)
```

2. Select option **E > Edit m2btab and m2bmaps**.

3. Select option **T > TRANSLATION Tables (m2bmap)** to view the existing m2bmap files.

You will see a screen that looks something like this:

```
                       M2BMAP Suffixes

   01 > bloc
   02 > oloc
   03 > itype
_____________________________________________________
Key a number or
P > PRINT               C > COPY an existing m2bmap
Q > QUIT                + > ADDITIONAL options
N > Create a NEW m2bmap file
Choose one (1-18,P,Q,C,N,G,S,Y,+)
```

4. To create a new m2bmap, select the **N > Create a NEW m2bmap file** menu option.

At the **Enter new m2bmap suffix:** prompt, enter the suffix for the new m2bmap that you are creating. Each m2bmap must have a unique file name. The suffix should be as mnemonic as possible.

5. Once you have entered the new suffix, you will be placed in the full screen editor. Key **^N > NEW line** to get into editing mode.

Enter Global Variable Functions (@) at the top of the file. Then enter a comparison expression followed by the separator (usually a vertical bar AKA pipe, e.g. "|") followed by the replacement expression, for example:

```
@stop_on_map=true
REF|ebref
GEN|eban
FIC|ebaf
CAR|ebj
```

NOTE

See reference section TRANSLATION TABLES: THE M2BMAP FILES for a list of instructions including Global Variable Functions (@) and other values and expressions that can be included in an m2bmap file.

6. When you are finished editing, key **^E > END** to save the m2bmap and end the full screen editor session.

7. Edit the load table (m2btab) to add a pointer to the m2bmap using the Special Processing Function %map. For example:

```
/^960||t|0|0|o|   |02|n|N|1|%map=("m2bmap.oloc")
                                ^^^^^^^^^^^^^^^^^^^^
```

In this example, the data in 960$t will be translated/mapped per the instructions in m2bmap.oloc and the new value will load in the Location field in the order record.

In your m2btab file, check and adjust element 5 (Number of Bytes) to ensure that it is greater than or equal to the largest number of characters in the comparison expression. Or set it to 0 so the entire incoming string is read (the most common choice).

In your m2btab file, check elements 7 (Variable-length field group tag) and 8 (Fixed-length field number):

- If element 7 is contains a space and element 8 has a number, you are loading into a fixed-length field, and the comparison expression should encompass the data within the subfield only. In the example above, the comparison expression is the data in the 960 subfield t.
- If element 7 has a single character and element 8 has a 0 (zero), you are loading into a variable-length field, and the comparison expression must be the entire MARC tag.

8. It is important when using m2bmap files that you test the load table and translation table prior to putting it into production. See the HOW TO TEST A NEW M2BTAB AND/OR M2BMAP section of this manual.

HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL)

1. Add a new LOAD button to **Data Exchange process: Load records via locally-created load profiles** by editing the m.marcload.local file via the "Additional system functions" menu:

   A > ADDITIONAL system functions
   M > Read/write MARC records
   X > Load Profile Maintenance

   You will see a screen that looks like this:

```
                     Load Profile Maintenance

E > Edit m2btab and m2bmaps
L > Edit m.marcload.local
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (E,L,Y,Q)
```

2. Select option **L > Edit m.marcload.local**.

   This will put you into Innovative's full screen editor, and you will see a file that looks similar to this:

```
@HEADER=Load records via locally-created load profiles
@MAXFILES=50
@MAXBYTES=1000000000
@DIRNAME=marc
F > Get MARC records using FTS||xftp get %S.lfts
P > PREPROCESS records loaded via FTS||marccopy -u -i%#.lfts -o%%.lmarc
L > LOAD a MARC file||marc2inn -HL -x -fzzz -I -i -so %#.lmarc
C > LOAD a MARC file (slow rate)||marc2inn -H"1,0,0" -I -x -fzzz -i -so %#.lmarc
M > View MESSAGE log||review -c marc2inn.log
V > VIEW a MARC file||marcview %#.lmarc
U > View an UNPROCESSED MARC file||marcview -u %#.lfts
Q > QUIT||
```

NOTE

There are other m.marcload files on the system that have been set up by Innovative and should not need to be adjusted. If you need assistance with one of the other m.marcload files, please contact the Customer Services Help Desk.

3. Add a new marc2inn line for the new load table (m2btab file) you created.

> **NOTE**
> Innovative recommends that you copy an existing line, rather than create a new line. By copying an existing line, it is less likely that syntax errors will occur due to missing or extra vertical bar characters.

a) Arrow down to the marc2inn line you want to copy, for example:

```
L > LOAD a MARC file||marc2inn -HL -x -fzzz -I -i -so %#.lmarc
```

b) Copy the line by pressing **^D > DELETE** line then press **^U > UNDELETE** line twice.

c) In the new marc2inn line you have created:

1) Change the letter of the menu option that appears at the beginning of the line to an unused capital letter. In the example above letters F, P, L, C, M, V, and U are used.

2) Update the menu text to describe what type of records will be loaded. For example, **N > LOAD a Marcive bib/item file**. The menu text displays when the mouse is hovered over the LOAD button in Data Exchange so make it descriptive. Alternatively, include the m2btab suffix in the menu text. For example, **N > LOAD a bib/item file (.marcive)**

3) Update the –f parameter in element 3 to the suffix of your new m2btab, for example: **-fmarcive**

For example, the m.marcload.local file now looks like this:

```
@HEADER=Load records via locally-created load profiles
@MAXFILES=50
@MAXBYTES=1000000000
@DIRNAME=marc
F > Get MARC records using FTS||xftp get %S.lfts
P > PREPROCESS records loaded via FTS||marccopy -u -i%#.lfts -o%%.lmarc
L > LOAD a MARC file||marc2inn -HL -x -fzzz -I -i -so %#.lmarc
C > LOAD a MARC file (slow rate)||marc2inn -H"1,0,0" -I -x -fzzz -i -so %#.lmarc
N > LOAD a Marcive bib/item file||marc2inn -HL -x -fmarcive -I -i -so %#.lmarc
M > View MESSAGE log||review -c marc2inn.log
V > VIEW a MARC file||marcview %#.lmarc
U > View an UNPROCESSED MARC file||marcview -u %#.lfts
Q > QUIT||
```

4. When you are finished editing m.marcload.local, key **^E > END** to end your full screen editor session. You will be asked: **Save changes to m.marcload.local? (y/n)**. Key **y** to save the file. Press the space bar to get back to the Load Profile Maintenance menu.

5. Launch Sierra/Millennium. **Data Exchange**. Select **Data Exchange process: Load records via locally-created load profiles**. You should see your new LOAD button. If there are a lot of LOAD buttons, a scroll bar will appear.

Or alternatively, if you are already logged into Sierra/Millennium, follow these steps:

a) Go to Data Exchange mode.

b) You may need to switch to another Data Exchange process and/or a different mode, e.g. Catalog, then go back to Data Exchange process: Load records via locally-created load profiles to see the new LOAD button.

HOW TO ADD A NOTE TO A LOAD TABLE (M2BTAB)

If you wish to add a note to a load table, begin a line with the '#' character and start keying your note. This is called “commenting out” a line in a load table.

Innovative highly recommends that you add a note to the top of every load table (m2btab) that you create or modify to indicate who did the work, when the work was done, and what this particular m2btab is designed to do. For example:

```
#Load profile for Baker & Taylor bib and order records.
#Created by Suzi Smith, June 2010
#Edited by Joe Jones, Dec 2010 to allow for overlay
#on the ISBN
```

HOW TO TURN OFF AN INSTRUCTION IN A LOAD TABLE (M2BTAB)

If you wish to turn off or disable an instruction in a load table, begin the line with the '#' character. This is called “commenting out” a line in a load table.

In the following example, the 270 MARC tag will not load because the 270 loading instruction has been turned off. The loading program will ignore the instruction:

```
#270||+|0|0|b|p|0|y|N|0|
```

Similarly, Global Variable Functions (for example, @comline) can be turned off by commenting out the line:

```
#/^949 ||a|0|400| | |0|n|G|0|@comline
```

HOW TO LOAD RECORDS AS NEW (M2BTAB)

Make sure the following two instructions in the m2btab file are set as follows:
Approximately the 12th line of a load table:

```
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag=" "
```

Approximately the 25th line of a load table:

```
|||0|0| | |0|n|G|0|@ldx=""
```

HOW TO LOAD ALTERNATE ALPHABETS AND DIACRITICS (M2BTAB)

Because most alternate alphabet records from MARC 21 sources contain the alternate alphabet fields in a MARC 880 field, alternate alphabets are loaded via a single line in the load profile:

```
880||+|0|0|b|y|0|y|N|0|
```

In MARC 21 alternate character representations are stored in the 880 field, and the subfield 6 contains the link to the MARC tag of the corresponding English language field. When these 880 fields are loaded, the hexadecimal representations of the alternate characters will be automatically translated to the storage format that is used in the Innovative system (ASCII characters enclosed in curly braces).

When loading records with diacritics, it is important to know the character encoding of the incoming file. The load table needs an instruction to translate incoming characters in alternate alphabets to the storage format that is used in the Innovative system. See the **@diac** and **@diac_sub_table** instructions in this manual.

NOTE

For more information on alternate characters, please see record 101350 (Chinese/ Japanese/Korean Characters) in the *Innovative Guide and Reference*.

Please contact the Customer Services Help Desk if diacritics are not loading as they should.

PROFILING FROM START TO FINISH

Setting up a new load profile includes editing an m2btab, m.marcload.local, and possibly an m2bmap file. However, before you start editing these files you need a list of loading goals or requirements. For example, you might come up with a list such as:

- Overlay bibs (replace bibs in the database) using the ISBN as a match point
- Protect the bib call number from overlay
- Create item records
- Load item location, item type, and item barcode

Also, you need a file to work with. Once you have both, you can begin your work in Sierra/Millennium's Data Exchange mode.

1) Prepare the data:
   a) Select the process **Load records via locally-created load profiles**
   b) FTP the data file to Data Exchange
   c) **Prep** the file.

2) **View** and analyze the data file in Data Exchange. See the HOW TO ANALYZE DATA section in this manual. Does the data support the loading requirements? Using the example above, you would:
   a) Check the records for ISBNs
   b) Find the field that contains item information, specifically shelving location, item type, and item barcodes
   c) Determine whether the incoming location and/or item type values need to be translated/mapped. If yes, what are the unique values for each? You may need to ask your vendor or supplier for this information.

3) As you are viewing records, write down the block numbers of interesting-looking records that would be good candidates for testing. The block numbers uniquely identify the location of a record in a file and appear above the record data.

4) Create the new load table (m2btab file) and customize it to meet the loading requirements. See the HOW TO CREATE A NEW LOAD TABLE (M2BTAB) and ISSUES TO BE CONSIDERED WHEN CREATING A NEW LOAD TABLE sections in this manual.

5) Optionally, create any translation/mapping tables (m2bmap files) that are needed. See the HOW TO CREATE A NEW TRANSLATION TABLE (M2BMAP) section in this manual.

6) Your m2btab file specifies which record templates to use in the @dflt trigger. In Sierra/Millennium, go to **Admin | Settings | Record Templates**. Do the required templates exist? Are they up-to-date?

7) Create a LOAD button by modifying the m.marcload.local file. See HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL) section in this manual.

8) If you are satisfied your load profile is in good shape, test load the entire file. Are there any errors to resolve?

9) Next load for real the record(s) you identified in Step 3. Is there any data loss? Data duplication? Are values being translated/mapped correctly? Resolve any problems in the load profile and repeat testing until the data is loading as required. See the HOW TO TEST A NEW M2BTAB AND/OR M2BMAP FILE section in this manual.

10) After steps one through nine are successfully completed, you are ready to load records with your new load profile!

HOW TO ANALYZE DATA

When you obtain data from a new source, it is important to analyze the data to gain an understanding of the loading possibilities. For example, if you wish to assign different Item Types when the records load, but there is no field in the records that has item type information, then it is not going to be possible to assign different Item Type values during the load.

Also during data analysis you refine your list of loading requirements. For example, if you see some interesting message type data in the item creation field that you weren't expecting, but you wish to load, then you want to take note and check that the load table (m2btab file) has an instruction to load the message data.

This section assumes you have already transferred and prepped your file in **Data Exchange process: Load records via locally-created load profiles**.

1) In Data Exchange, click once on the "lmarc" file. Click button **View**.

    a. Do the records have the data you expect it to contain?

    b. Do the records contain diacritical marks and/or are in a non-Roman alphabet? You may see the diacritics enclosed in curly brackets, for example: {u0304}. If yes, then you'll want to be sure to include these records in your test load to confirm that the character encoding instruction is correctly set up in the load table.

    c. Do the records contain control numbers? This is important if you wish to overlay (replace) records in the database. If you are using the Innovative bibliographic record number as a match point for overlay, do you see it in the records? What MARC tag is it in?

        i. Does the bibliographic 001 field need any special processing? For example, do you want to strip OCLC prefixes and leading zeros?

    d. If you plan to load item, order, and/or holdings/checkin records, do you see linked record creation fields in the records? The item creation field is typically 945 or 949, but can be stored in any non-standard MARC tag. The order creation fields are typically 960 and 961. The MARC 21 holdings/checkin creation fields are 85x and 86x.

    e. Do the records contain location information? What MARC tag and subfield is it stored in?

> **NOTE**
> For more information about Viewing MARC Records in Data Exchange, see page no. 106006 in the *Innovative Guide and Reference*.

2) Click button **Count** and a new display will appear labeled **Counting Tags**. You will see the file name and a count of records and a count of each MARC tag in the file.

   a. Do the counts make sense? For example, in a MARC 21 bibliographic file, do you see the same number of 245 fields as number of records in the file? If you are planning to load linked records (item, order, and/or holdings/checkin), are the counts for the MARC tags that have the linked record data plausible?

   b. Are there any 999 fields in the file? If yes, you'll need to change the lines that begin with "999" or "/^999" in the load table (m2btab file) to another MARC tag that isn't in the data, perhaps 998, to prevent data in the 999 from being interpreted as commands.

> **NOTE**
> For more information about Counting MARC Tags in Data Exchange, see page no. 107506 in the *Innovative Guide and Reference*.

3) **Print** the **Counting Tags** display. You will use this later when you are editing the load table (m2btab file). Specifically, are there any MARC tags that need to be added to the load table? Remember, if a MARC tag is excluded from the load table, it will not load.

4) Use the **Search** button to look for values in specific MARC tags and subfields that are of interest to you. For example, if you are unsure what types of control numbers are in 001, search for the tag to see a display of 001 data in list form. Click button **Search**. In the **MARC Tags and Indicators** box key 001. Click button **Search**. In the Field Data column you will see a list of 001 data preceded by the MARC tag 001.

> **NOTE**
> For more information about Searching Files in Data Exchange, see page no. 107507 in the *Innovative Guide and Reference*.

ISSUES TO BE CONSIDERED WHEN CREATING A NEW LOAD TABLE

Before creating a new load profile, here is a list of questions that may be helpful in deciding which instructions need to be updated in the load table.

1. What type of records will you be loading? Bibliographic only? Bibliographic and order? (**@main, @link, Element 6, Element 11**)
2. Should the fields load as MARC and/or non-MARC? (**@marc, Element 9**)
3. What mnemonic file extension will be assigned to the new load table? (**@recs**)
4. What Record Templates should be used? (**@dflt**) Have the Record Templates been defined in Sierra/Millennium?
5. Should incoming records overlay existing records or load as new? (**@ov_tag, @ldx**)
6. What is the match point for overlay? (**@ov_tag, @ldx**)
7. When a match is found, what action should be taken? (**@ov_action**)
8. Should fields be protected from overlay? (**@ov_protect**) Should fields be protected from overlay conditionally? (**@ov_protect and :d** and **:k** qualifiers)
9. How are locations and call numbers derived? Should the Holding Symbol table be used? (**@holdsymb**)
10. Is the character encoding of the records something other than MARC-8? (**@diac_sub_table**)
11. Do you wish to set up a busy file in case a record cannot be overlaid by the loading program because it is locked for editing? (**@busy, @busy_file**)
12. Do you want to assign the system data as the Cat. Date in the bibliographic record during the data load? (**@cdate**)
13. Are there any non-standard MARC tags that you wish to load? If yes, make sure they are included in the load table. (**Element 1**)
14. Are there are MARC tags that you do not want to load? If yes, make sure they are excluded from the load table. (**Element 1**)
15. Are there are any subfields specific to a MARC tag that you do or do not want to load? (**Element 1, Element 3**)

16. Does the bibliographic 001 field require special processing? (**%001**)
17. What field will be used to create linked records specifically item, order, and/or holdings/checkin? (**@link**)
18. Will any incoming values need to be translated/mapped during the load? (**%map**)

HOW TO TEST A NEW M2BTAB AND/OR M2BMAP FILE

Before putting a new load profile into production, it is important for you to test it thoroughly. The goal of testing is to confirm that the load table is loading the data as required with no data loss or data duplication.

Prior to testing, you will have developed a list of loading requirements, updated your load table, and created a LOAD button in Data Exchange process: Load records via locally-created load profiles.

1) Have the list of loading requirements handy.

2) Identify test records that will demonstrate that each loading requirement is being met. Typically the first couple of the records is sufficient for testing, but sometimes you have to go further into the file to find good test records.

3) Load the file in *test* mode. In Sierra/Millennium and **Data Exchange process: Load records via locally-created load profiles**, click once on the 'lmarc' file you wish to test load. Click the appropriate **LOAD** button. At the bottom of the screen click button **Test** to begin test loading records.
    a) Warning messages will display in Output Messages when a fixed-length field code is invalid. The value from the Record Template is inserted instead. You can scroll back through the Output Messages at the conclusion of the load to review warnings.

4) After the test load has completed, you will see a RECORD LOADING STATISTICS display. Confirm that the number of records loaded as NEW, OVERLAYED, and REJECTED makes sense. For example, you will see a display similar to this one:

```
                                 RECORD LOADING STATISTICS
Input file     - overlay.lmarc           Start date - April 05 2:23PM
Error file     - overlay.errlog          End date   - April 05 2:23PM
m2btab file    - m2btab.b
Number of input records    - 5
Number of errors           - 0
                   NEW                                  EXISTING    INPUT       TOTAL
                   RECORDS     REC #S ASSIGNED          RECORDS     RECORDS     RECORDS
                   CREATED     START       STOP         OVERLAYED   REJECTED    READ
BIBLIOGRAPHIC      1           b1001231    b1001231     4           0           5
```

5) If there were errors reported by the loading program during the test load they will be saved to an errlog file. Click the **Close** button to exit the Output Message display. Look for the errlog file for the file you just test loaded in the list of files under **Data Exchange process: Load records via locally-created load profiles**. Click once on the errlog file and click button **View**. There is information about the load in two tabs. The **Errors** tab lists the loading errors. The **Statistics** tab shows the Record Loading Statistics.

6) Adjust the m2btab and/or m2bmap files as necessary to correct any errors reported by the loading program.

7) Repeat the test load to confirm that the errors reported by the loading program are resolved.

8) If records should load as NEW (no OVERLAY), continue testing by loading a small number of records into the database. If records should be OVERLAID, go to step 9.

   a) If all records are loading as NEW (no OVERLAY), proceed with loading the first 10 records in the file. In Siera/Millennium and **Data Exchange process: Load records via locally-created load profiles**, click once on the 'lmarc' file you wish to load. Click the appropriate **LOAD** button. Check that the **Start Block** is set to 1. **Set the Maximum Records to Load** to 10. Click button **Use Review Files**. This will allow you to copy the records into a review file in Create Lists mode at the conclusion of the load. At the bottom of the screen click button **Load** to begin loading records.
   b) Review the loaded records. Methodically check the loaded records against the list of loading requirements.
      a. Is the data loading as expected?
      b Is there any data loss?
      c. Are all of the fields present that are supposed to be?
      d. Is there any data duplication?
      e. Were linked records correctly created?
      f. Are the MARC fields in the correct variable-length field group tag?
      g. Did the m2bmaps work?
   c) If the data did not load as expected, adjust the m2btab and/or m2bmap files accordingly.
   d) Delete the 10 records that were previously loaded.
   e) Repeat the test load until the data is loading as expected.

9) If records should OVERLAY, continue testing by loading a small number of records into the database.
   a) Print the first 10 records from the MARC file you will be loading, and retrieve and print the same records in Sierra/Millennium (if they are already in the database and you plan to overlay them). This will give you a backup copy of the database records in case the record is corrupted by the load you are about to do.
   b) In Sierra/Millennium and Data **Exchange process: Load records via locally-created load profiles**, click once on the 'lmarc' file you wish to load. Click the appropriate **LOAD** button. Check that the **Start Block** is set to 1. Set the **Maximum Records to Load** to 10. Click button **Use Review Files**. This will allow you to copy the records into a review file in Create Lists mode at the conclusion of the load. At the bottom of the screen click button **Load** to begin loading records.
   c) Review the records that were loaded. Methodically check the loaded records against the list of loading requirements.
      a. Did the incoming record overlay the correct database record?
      b. Were there any fields that should have been protected from overlay that weren't?
      c. Is the data loading as expected?
      d. Is there any data loss?
      e. Are all of the fields present that are supposed to be?
      f. Is there any data duplication?
      g. Were linked records correctly created?
      h. Are the MARC fields in the correct variable-length field group tag?
      i. Did the m2bmaps work?
   d) If the data did not load as expected, adjust the m2btab and/or m2bmap files accordingly.
   e) Edit the database records to restore them to their pre-overlaid state.
   f) Repeat the test load until the data is loading as expected.

10) As an alternative to loading the first 10 records in a MARC file, you can load selected records from your MARC file on a record-by-record basis, so that you can test all possible conditions your m2btab and/or m2bmap(s) are designed to handle.

    You can find these records by searching the MARC file for particular values in a tag or subfield. Note down the starting block number that appears in the upper right hand corner when viewing the file in Data Exchange.

After clicking the appropriate LOAD button, set the **Start Block** to the record that is targeted for your test. For example, the good test record may occupy Blocks 110-114 in the file so set the **Start Block** to 110. Do not change the **Stop Block**. Set the **Maximum Records to Load** to 1. Click button **Load**. This will load the one record that occupies Blocks 110-114 in the data file.

NOTE

For testing order records, Innovative recommends not loading the status code, but using the defaults for new records to supply the default value of 1. This way, you can verify that the fields (except status) loaded correctly without encumbering funds, so that you don't have to post, cancel the order, post again, and delete the test order records.
Once you have completed your testing, remember to change the default status field to a normal value before loading.

SUPPORT FROM INNOVATIVE

CONSULTATION

If you have tested your new load profile and have not been able to make it work as you intend, please contact the Help Desk. The Help Desk will open a call to track the query and you may be referred to Customer Sales if special support needs to be purchased from Innovative.

Consultation on locally-created load profiles falls outside of Innovative's standard maintenance service agreement and will be available at Innovative's current rates.

TECHNICAL ASSISTANCE

If data corruption problems have occurred in your database due to erroneous locally-created load profiles, then please contact the Help Desk. The Help Desk will open a call to track the query and will refer you to Customer Sales if technical assistance will need to be purchased from Innovative.

Consultation on locally-created load profiles falls outside of Innovative's standard maintenance service agreement and will be available at Innovative's current rates.

PROFILING SERVICES

The library may still purchase load profiling services from Innovative in cases when it does not wish to do its own profiling or when the desired load behavior cannot be accomplished with the functions described in this document.

PRACTICE AND TRAINING EXERCISES

PRACTICE EXERCISE 1:

Copying and Editing an Existing Load Table

Objectives:

Note: Student should have been instructed in how to log on to the training system

- Explain the proper way to access load table functionality
- Utilize the basic functions of copying and editing load files
- Familiarize the student with the Full Screen Editor
- Reinforce data profiling concepts

Additional material: Editing Functions of the Full Screen Editor: *Innovative Guide and Reference* page no. 101371.

1. All manipulation of load tables and translation tables will be done via the "Additional system functions" menu:

   A > ADDITIONAL system functions
   M > Read/write MARC records
   X > Load Profile Maintenance

   At this point, you will see a screen that looks like this:

```
                    Load Profile Maintenance

E > Edit m2btab and m2bmaps
L > Edit m.marcload.local
Y > DISPLAY file SIZE & DATE
Q > QUIT
Choose one (E,L,Y,Q)
```

2. Select option **E > Edit m2btab and m2bmaps**.

3. Select option **P > Load PROFILES (m2btab)** to view the existing load profiles.

NOTE

The file extension of each m2btab table reflects the type of record being converted, e.g., m2btab.asub for subject authority records. When creating a new m2btab file, it's important to give it a file extension (also called suffix) that is mnemonic. See the section above on "STANDARD M2BTAB FILE EXTENSIONS" for a list of suffixes you may encounter.

4. Review the list of m2btab files. The list will look something like this:

```
                          M2BTAB Suffixes

01 > a
02 > anam
03 > asub
04 > b
05 > bo
06 > bta
07 > fse
08 > p

Key a number or
F > FORWARD              C > COPY an existing m2btab
J > JUMP                 G > GET a file using
P > PRINT                S > SEND a files out of INNOPAC using FTS
Y > DISPLAY file SIZE & DATE
Q > QUIT

Choose one (1-98,F,J,P,Q,C,G,S)
```

5. Selecting **C > COPY m2btab.<ext>** will bring up the following prompt:
   Enter file number to COPY: (1-8) ____
   Choose the "**b**" file suffix to copy. In this example that is selection 04, then press <enter>.

   This will bring up the following prompt:
   Copy to what new suffix?: ____
   Enter a distinctive suffix for your new file, using all lower case. This file will be used in a later exercise.

6. A screen like the following will be displayed:

```
               M2BTAB File Validation and Maintenance

E > EDIT m2btab.mke
V > VERIFY m2btab.mke
U > UPDATE m2btab with changes
R > RESTORE original m2btab
M > marc2inn error MESSAGES
Q > QUIT
Choose one (E,V,U,R,M,Q)
```

**** Exercise continues on the next page ******

7. Selecting **E > EDIT m2btab.<ext>** puts you into Innovative full screen editor (FSE). At the top of the screen is the brief display of the FSE controls. It looks like this:

```
^E > END          ^F > Show FULL Menu          OVERWRITE
```

This brief menu indicates two choices (**^E** and **^F**) at the top of the screen, along with a message indicating whether you are in "**OVERWRITE**" (chosen here) or "**INSERT**" mode (not shown) The caret (^) denotes that the control key must be held down while the menu choice letter is being pressed (e.g., ^F means **<CTRL>F**):

Do not confuse this shorthand notation with the "^" symbol that is located "above" the numeral 6 on the keyboard. *You must press the "control" key (i.e.,<CTRL>), not the "^", to access the editor functions!*

8. Pressing **<CTRL>F** toggles between the brief and full menu display. **Please try that now.**

Below is an example of the full menu display. Notice the controls for editing your file. It may be helpful to toggle/display the FSE full menu as you are editing files until the commands become more familiar.

```
^N > NEW line          ^O > Toggle OVERWRITE/insert          ^E > END
^U > UNDELETE line     ^W > Diacritics coded/DISPLAYED
^D > DELETE line       ^B > Begin BLOCK
^X > DELETE char
```

9. Practice using the editor commands on your test file to make the following three changes:
   - Adding a note at the top of the file (page no. 99)
   - Modify the @recs Global Variable Function (page no. 62)
   - Change the @msg trigger to reflect the types of records that will now be loaded. (page no. 47)

****** Exercise continues on the next page ******

10. Verify the file by selecting “**V > VERIFY m2btab.<ext>**”.

```
                M2BTAB File Validation and Maintenance

E > EDIT m2btab.mke
V > VERIFY m2btab.mke
U > UPDATE m2btab with changes
R > RESTORE original m2btab
M > marc2inn error MESSAGES
Q > QUIT
Choose one (E,V,U,R,M,Q)
```

The following congratulatory message will appear:

```
   Congratulations!! You have a perfect M@BTAB table!
   Press <SPACE> to continue
```

Press <Space> to continue as instructed. If you did not see this feedback, contact the instructor for assistance.

11. Update the file by selecting “**U > UPDATE m2btab with changes**”.

```
                M2BTAB File Validation and Maintenance

E > EDIT m2btab.mke
V > VERIFY m2btab.mke
U > UPDATE m2btab with changes
R > RESTORE original m2btab
M > marc2inn error MESSAGES
Q > QUIT
Choose one (E,V,U,R,M,Q)
```

PRACTICE EXERCISE 2:

Specifying Record Templates, the Cat. Date field and Creating Linked Records

Objectives: To practice with and gain understanding of record templates, automatic date selection, and creation of linked records.

Instructions: Using the load table copied in Practice Exercise 1, apply the following changes.

1. Assume that you wish to use the bib and item record templates “tapeb” and “tapei”, respectively. Modify the appropriate load table setting accordingly.
   (Page no. 37)

2. Assume that you want to load the Cat. Date that is in your chosen bibliographic record template. Modify the appropriate load table setting accordingly.
   (Page no. 37)

3. Assume that you are loading a set of ONLY bibliographic records. Modify your load table such that one linked item record is created for every bibliographic record that is loaded.
   (Page no. 44)

PRACTICE EXERCISE 3:

Troubleshooting Load Tables 101

Objectives: Practice troubleshooting structural and non-structural errors introduced into a load table.

1. Copy m2btab.practice to another extension (as shown in Practice exercise 1).
2. Attempt to verify the table.
3. Correct the structural errors until the load table can be verified.
4. Once the table has been verified, inform the instructor.
5. Review the non-structural errors as a group exercise with the instructor.

PRACTICE EXERCISE 4:

Applying Required Changes to an Existing Load Table

Objectives: Modify an existing load table to perform the functions determined in data analysis.

Scenario:
Create a new load table for bib and item records based on data in file export.lmarc. Copy m2btab.b to a file with a new extension.

Bibs Requirements:

- Load bib records as new records to database. Do not overlay existing records.
- Load OCLC control number in 001 field in bib record with 'ocm' prefix. Do not remove the ocm prefix from the 001 field.
- Insert 590 subfield “a” into all bib records with data ‘Special data load.’ The 590 field is assigned to field group tag n (NOTE).

Items Requirements:

- Create item records from 949 field, any indicators
- Create a map to load a correct numeric code for the item type field from 949 subfield “t”

  Valid Item Type codes are:
  000 = Books
  003 = Books, Reference
- Load item barcode from 949 subfield “b”
- Load call number from 949 subfield “d” to a 099 subfield “a” into the bib record. There are multiple 949 fields in each bib record, load call number from the first 949 field in record only.

PRACTICE EXERCISE 5:

Overlaying of Bibliographic Records

Objective: To practice overlay of bib records.

Scenario:

Copy m2btab.practice5 to a new file. Using your copied load table, make the following modifications to load the file overlay.lmarc.

Make the following changes:

- Set overlay trigger to match on bib record number stored in the 907 MARC tag
- Set overlay trigger to reject records that don't match the bib record number.
- Change @ov_protect trigger to do the following:
- Protect cataloging date
- Protect bibliographic record location code in the fixed-length field and the bibliographic location variable-length field
- Protect internal note 599 MARC field in the existing record
- Protect existing 699 MARC fields. Incoming 699 MARC fields will be discarded.

TRAINING EXERCISE 1:

Creating and Evaluating Load Table Entries

Objectives:

To practice and review the material on the twelve data elements that comprise an m2btab entry in a load table. Material may be reviewed on page no. 26-32 of this manual

1. Evaluate the following two lines. How do they differ from one another?

```
100|  |-w|0|0|b|a|0|y|N|0|
100|  |+|0|0|b|a|0|n|N|0|
```

2. Explain what this line will do:

```
600,610,611,650-651|   |-w|0|0|b|d|0|y|N|0|
```

3. To load all subject headings, what MARC tags and subfields should be used for the following line? (Hint: in 6xx range)

```
_______|  |__|0|0|b|d|0|y|N|0|
```

4. Create a line to map the system control number from 001 to 935. (Hint: Pay attention to MARC tags and subfields.)

5. Complete the following load table line:
   - Fill in the offset and byte columns so the language code will load from MARC field 008.
   - (Hint: it starts with offset 35 of tag 008 and has 3 bytes -
   - Sample MARC tag line: **008** 980817s1999   nyu 000 1 **eng@**)

```
008||  |___|___|b|  |24|n|N|0|language
```

6. There is one error in the following lines. Please identify the error.
   (Hint: pass indicator):

```
|||0|0|  |  |0|n|G|0|@link="i:1:949"
/^949|  |d|0|0|b|  |26|n|N|1|bib location
/^949|  |d|0|0|i|  |79|n|N|1|item location
```

7. From the following sample 949 marc tag line, create a new load table entry with these specifications:

   - Move the call number from 949|d to 092|a
   - Both indicators in MARC field 092 will be blank.
   - Place pre-stamp from 949|f in front of call number:

94910|b32141021998426|tJ|**fJUVENILE|d917.291 MORRISON**|p2501|nCC

In order to create this field in the bibliographic record during the load:

```
092   |fJUVENILE|a917.291 MORRISON
```

TRAINING EXERCISE 2:

Practice Applying Overlay Triggers

Objectives:
To review frequently used overlay triggers by completing the provided load table lines to fulfill the requirements of the given scenarios.

Scenario I:
Fill in the blanks so that the sample load table lines will meet the following requirements:

- match point is the OCLC record number
  (field = 001, field group tag =o, and index tag =o)
- 0 match found, create a new record;
  1 match found, overlay
  2 or more matches found, create a new record

```
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag="_____"
|||0|0| | |0|n|G|0|@ov_action="_____"
```

Scenario II:
Fill in the blanks so that the sample load table lines will meet the following requirements:

- match point is the ISBN number in the MARC field 020 (field group tag = i and index tag = i)
  Add a confirmation test on the 245 title field (page no. 58)

- 0 match found, reject
  1 match found, overlay
  2 or more matches found, reject

- Protect fields:
  Location
  Cat. Date
  MARC tag 856 data (field group tag = y)

Note: The trigger "@m2b_normalize_020" is typically used when the ISBN is used as the match point. See page no. 47.

```
/^998||t|0|10|b| |0|n|G|0|#com="ov"@ov_tag="___"
|||0|0|b| |0|n|G|0|@ov_action="__"
|||0|0|b| |0|n|G|0|@ov_protect="___________________"
```

TRAINING EXERCISE 3:

Using Special Functions

Objectives:
To gain a better understanding in employing special functions such as %001, %replace, %first, and %last.

1. You want to load the OCLC record number into MARC field 001, so it strips the prefix "ocl7", "ocm", or "ocn" and the leading zeros from OCLC record numbers.

   Example:

Before load:	001 ocm06051528
After load:	001 6051528

Select the correct answer:

A. `001||%|0|0|b|o|0|y|N|0|%001(start="1-9",char="!-~",valid="n")`
B. `001||%|0|0|b|o|0|y|N|0|%001(start="1-9",char="!-~",valid="y")`
C. `001||%|0|0|b|o|0|y|N|0|`

2. You want to change the phone number format from 917 558 6209 to 917-558-6209 during a patron record load (replace the space “ “ with the dash “-“).
   Fill in the blanks:
   `225||+|0|0|p|t|0|n|N|0|`______________________________

3. A vendor is providing multiple call numbers in MARC field 099. You want to load only the first occurrence of MARC field 099.
   Fill in the blanks:
   `099||a|0|0|b|c|0|y|N|0|`______________________________

TRAINING EXERCISE 4:

Translation Tables (m2bmap)

Objectives:
To practice writing a translation table and referencing to it in a load table.

Scenario:
The incoming MARC file has location codes in 949 subfield “l” (lowercase letter L). Construct the line in the m2btab file that will load the item location codes from the 949 subfield “l”. Create a new translation table called ‘m2bmap.itemloc’ for the following location codes:

Incoming data	**Innovative code**
STACKS	main
REF	mnref
REF2L	mnref
STACK	main
STAFF	staff

APPENDIX 1 – BIBLIOGRAPHIC AND ITEM LOAD TABLE

STANDARD BIBLIOGRAPHIC AND ITEM RECORD LOAD PROFILE FOR BATCH LOADS (m2btab.batch)

```
#standard bib/item load table
|||0|0| | |0|n|G|0|@main="b"
|||0|0| | |0|n|G|0|@marc="bic"
|||0|0| | |0|n|G|0|@atab="a"
|||0|0| | |0|n|G|0|@msg="Bib and item records will be created"
/^999||m|0|12| | |0|n|G|0|#com="ins"@password=""
/^999||z|0|10| | |0|n|G|0|#com="recs"@recs="batch"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||s|0|30| | |0|n|G|0|#com="ip"@itemprefix=""
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt="biblio,item"
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag="o"
|||0|0| | |0|n|G|0|@ov_action="o"
|||0|0| | |0|n|G|0|@ov_protect="b=V023456789hy(962)k(970,971)n(972)"
|||0|0| | |0|n|G|0|@holdsymb="049a"
#|||0|0| | |0|n|G|0|@pre_map="m2bpre_map."
|||0|0| | |0|n|G|0|@locmerge="y"
/^949 ||a|0|400| | |0|n|G|0|@comline
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@leader_utf8="y"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="y"
|||0|0| | |0|n|G|0|@speriod="n"
008|| |35|3|b| |24|n|N|0|lang(b)
008|| |15|3|b| |89|n|N|0|country
L|| |7|1|b| |29|n|N|0|bib lvl
L|| |6|1|b| |30|n|N|0|mat type
/^999||a|0|30|b| |26|n|G|0|#com="bn"
/^999||b|0|5|b| |27|n|G|0|#com="co"
/^999||c|0|8|b| |28|n|G|0|#com="ct"
/^999||d|0|1|b| |29|n|G|0|#com="b1"
```

```
/^999||e|0|1|b| |30|n|G|0|#com="b2"
/^999||f|0|1|b| |31|n|G|0|#com="b3"
/^999||g|0|3|b| |24|n|G|0|#com="la"
/^999||k|0|3|b| |89|n|G|0|#com="cy"
001||%|0|0|b|o|0|y|N|0|%001(start="1-9",char="!-~",valid="y")
002-009||%|0|0|b|y|0|y|N|0|%008="y"
010||+|0|0|b|l|0|y|N|0|%strip_blanks="n"
013-019||+|0|0|b|y|0|y|N|0|
020-024||+|0|0|b|i|0|y|N|0|
025-026||+|0|0|b|y|0|y|N|0|
027-028||+|0|0|b|i|0|y|N|0|
030-049||+|0|0|b|y|0|y|N|0|
066-072||+|0|0|b|y|0|y|N|0|
074||+|0|0|b|g|0|y|N|0|
086||+|0|0|b|g|0|y|N|0|
088||+|0|0|b|y|0|y|N|0|
100-111||-w|0|0|b|a|0|y|N|0|
130||-w|0|0|b|t|0|y|N|0|
210-222||+|0|0|b|u|0|y|N|0|
240||-w|0|0|b|t|0|y|N|0|
241-243||+|0|0|b|u|0|y|N|0|
245||+|0|0|b|t|0|y|N|0|%bracket="h"
246-247||+|0|0|b|u|0|y|N|0|
250||+|0|0|b|e|0|y|N|0|
254-258||+|0|0|b|y|0|y|N|0|
260-262||+|0|0|b|p|0|y|N|0|
263-265||+|0|0|b|y|0|y|N|0|
270||+|0|0|b|p|0|y|N|0|
300-399||+|0|0|b|r|0|y|N|0|
400-490||-w|0|0|b|s|0|y|N|0|
500-599||+|0|0|b|n|0|y|N|0|
600-699||-w|0|0|b|d|0|y|N|0|
700-720||-w|0|0|b|b|0|y|N|0|
730-740||-w|0|0|b|u|0|y|N|0|%bracket="h"
751-755||+|0|0|b|y|0|y|N|0|
760-777||+|0|0|b|w|0|y|N|0|
780||+|0|0|b|x|0|y|N|0|
785||+|0|0|b|z|0|y|N|0|
786-787||+|0|0|b|w|0|y|N|0|
800-811||-w|0|0|b|s|0|y|N|0|
830||-w|0|0|b|s|0|y|N|0|%bracket="h"
840||-w|0|0|b|s|0|y|N|0|
841-845||-w|0|0|b|y|0|y|N|0|
856||+|0|0|b|y|0|y|N|0|
866-868||+|0|0|b|h|0|y|N|0|
880||+|0|0|b|y|0|y|N|0|
882||+|0|0|b|y|0|y|N|0|
936||+|0|0|b|y|0|y|N|0|
987||+|0|0|b|y|0|y|N|0|
994||+|0|0|b|y|0|y|N|0|
|||0|0| | |0|n|G|0|@link="i:1:949 1"
/^949 1|z|ab|0|0|i|c|0|y|N|1|
/^949 1||c|0|0|i|v|0|n|N|1|#com="v"
/^949 1||g|0|3|i| |58|n|N|1|#com="cop"
#/^949 1||h|0|3|i| |127|n|N|1|#com="ag"
/^949  1||i|0|0|i|b|0|n|N|1|#com="i/a"@item
/^949 1||l|0|5|i| |79|n|N|1|#com="loc"
/^949 1||m|0|0|i|m|0|n|N|1|
/^949 1||n|0|0|i|x|0|n|N|1|
/^949 1||o|0|1|i| |108|n|N|1|#com="om"
/^949 1||p|0|8|i| |62|n|N|1|#com="pr"
/^949 1||q|0|5|i| |59|n|N|1|#com="i1"
/^949 1||r|0|1|i| |60|n|N|1|#com="i2"
/^949 1||s|0|1|i| |88|n|N|1|#com="sta"
/^949 1||t|0|3|i| |61|n|N|1|#com="ty"
/^949 1||u|0|1|i| |97|n|N|1|#com="im"
```

APPENDIX 2 – BIBLIOGRAPHIC AND ORDER LOAD TABLE

STANDARD BIBLIOGRAPHIC AND ORDER RECORD LOAD PROFILE FOR BATCH LOADS (m2btab.order)

```
#standard bib/order load profile
|||0|0| | |0|n|G|0|@main="b"
|||0|0| | |0|n|G|0|@marc="bo"
|||0|0| | |0|n|G|0|@atab="a"
|||0|0| | |0|n|G|0|@msg="Bib and order records will be created"
/^999||m|0|12| | |0|n|G|0|#com="ins"@password=""
/^999||z|0|10| | |0|n|G|0|@recs="order"
/^999||y|0|1| | |0|n|G|0|#com="po"@poprint="n"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||s|0|30| | |0|n|G|0|#com="ip"@itemprefix=""
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag="i:o"
|||0|0| | |0|n|G|0|@m2b_normalize_020="y"
|||0|0| | |0|n|G|0|@ov_action="a"
|||0|0| | |0|n|G|0|@ov_protect="b=V023456789hy(962)k(970,971)n(972)"
|||0|0| | |0|n|G|0|@holdsymb=""
#|||0|0| | |0|n|G|0|@pre_map="m2bpre_map."
|||0|0| | |0|n|G|0|@locmerge="y"
#/^949  ||a|0|400| | |0|n|G|0|@comline
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@leader_utf8="y"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="n"
|||0|0| | |0|n|G|0|@odate="y"
|||0|0| | |0|n|G|0|@rdate="n"
|||0|0| | |0|n|G|0|@speriod="n"
|||0|0| | |0|n|G|0|@m2b_multifund="960:o,t,u"
008|| |35|3|b| |24|n|N|0|lang(b)
008|| |35|3|o| |23|n|N|1|lang(o)
008|| |15|3|b| |89|n|N|0|country(b)
008|| |15|3|o| |100|n|N|1|country(o)
```

```
L|| |7|1|b| |29|n|N|0|bib lvl
L|| |6|1|b| |30|n|N|0|mat type
/^999||a|0|30|b| |26|n|G|0|#com="bn"
/^999||b|0|5|b| |27|n|G|0|#com="co"
/^999||c|0|8|b| |28|n|G|0|#com="ct"
/^999||d|0|1|b| |29|n|G|0|#com="b1"
/^999||e|0|1|b| |30|n|G|0|#com="b2"
/^999||f|0|1|b| |31|n|G|0|#com="b3"
/^999||g|0|3|b| |24|n|G|0|#com="la"
/^999||k|0|3|b| |89|n|G|0|#com="cy"
001||%|0|0|b|o|0|y|N|0|%001(start="1-9",char="!-~",valid="y")
002-009||%|0|0|b|y|0|y|N|0|%008="y"
010||+|0|0|b|l|0|y|N|0|%strip_blanks="n"
013-019||+|0|0|b|y|0|y|N|0|
020-024||+|0|0|b|i|0|y|N|0|
025-026||+|0|0|b|y|0|y|N|0|
027-028||+|0|0|b|i|0|y|N|0|
030-049||+|0|0|b|y|0|y|N|0|
066-072||+|0|0|b|y|0|y|N|0|
074||+|0|0|b|g|0|y|N|0|
086||+|0|0|b|g|0|y|N|0|
088||+|0|0|b|y|0|y|N|0|
100-111||-w|0|0|b|a|0|y|N|0|
130||-w|0|0|b|t|0|y|N|0|
210-222||+|0|0|b|u|0|y|N|0|
240||-w|0|0|b|t|0|y|N|0|
241-243||+|0|0|b|u|0|y|N|0|
245||+|0|0|b|t|0|y|N|0|%bracket="h"
246-247||+|0|0|b|u|0|y|N|0|
250||+|0|0|b|e|0|y|N|0|
254-258||+|0|0|b|y|0|y|N|0|
260-262||+|0|0|b|p|0|y|N|0|
263-265||+|0|0|b|y|0|y|N|0|
270||+|0|0|b|p|0|y|N|0|
300-399||+|0|0|b|r|0|y|N|0|
400-490||-w|0|0|b|s|0|y|N|0|
500-599||+|0|0|b|n|0|y|N|0|
600-699||-w|0|0|b|d|0|y|N|0|
700-720||-w|0|0|b|b|0|y|N|0|
730-740||-w|0|0|b|u|0|y|N|0|%bracket="h"
751-755||+|0|0|b|y|0|y|N|0|
760-777||+|0|0|b|w|0|y|N|0|
780||+|0|0|b|x|0|y|N|0|
785||+|0|0|b|z|0|y|N|0|
786-787||+|0|0|b|w|0|y|N|0|
800-811||-w|0|0|b|s|0|y|N|0|
830||-w|0|0|b|s|0|y|N|0|%bracket="h"
840||-w|0|0|b|s|0|y|N|0|
841-845||-w|0|0|b|y|0|y|N|0|
856||+|0|0|b|y|0|y|N|0|
866-868||+|0|0|b|h|0|y|N|0|
880||+|0|0|b|y|0|y|N|0|
882||+|0|0|b|y|0|y|N|0|
936||+|0|0|b|y|0|y|N|0|
987||+|0|0|b|y|0|y|N|0|
994||+|0|0|b|y|0|y|N|0|
|||0|0| |  |0|n|G|0|@link="o:1:#1"
#order record fixed fields
/^960||a|0|0|o|  |01|n|N|1|acq type
/^960||b|0|0|o|  |04|n|N|1|claim
/^960||c|0|0|o|  |06|n|N|1|code1
/^960||d|0|0|o|  |07|n|N|1|code2
/^960||e|0|0|o|  |08|n|N|1|code3
/^960||f|0|0|o|  |09|n|N|1|code4
/^960||g|0|0|o|  |11|n|N|1|format
/^960||h|0|0|o|  |14|n|N|1|ord note
/^960||i|0|0|o|  |15|n|N|1|ord type
/^960||j|0|0|o|  |16|n|N|1|raction
/^960||k|0|0|o|  |18|n|N|1|rloc
/^960||l|0|0|o|  |19|n|N|1|bloc
/^960||m|0|0|o|  |20|n|N|1|status
/^960||n|0|0|o|  |21|n|N|1|tloc
/^960||p|0|0|o|  |03|n|N|1|cdate
/^960||q|0|0|o|  |13|n|N|1|odate
/^960||r|0|0|o|  |17|n|N|1|rdate
#/^960||z|0|20|o|  |0|n|N|1|%foreign
/^960||s|0|0|o|  |10|n|N|1|e price
/^960||t|0|0|o|  |02|n|N|1|location
/^960||o|0|0|o|  |05|n|N|1|copies
/^960||u|0|0|o|  |12|n|N|1|fund
/^960||v|0|0|o|  |22|n|N|1|vendor
/^960||w|0|0|o|  |23|n|N|1|lang
/^960||x|0|0|o|  |100|n|N|1|country
/^960||y|0|0|o|  |106|n|N|1|volume
#order record variable fields
/^020||a|0|0|o|b|0|n|N|1|PO info
```

```
/^961||a|0|0|o|i|0|n|N|1|identity
/^961||b|0|0|o|x|0|n|N|1|for curr
/^961||c|0|0|o|n|0|n|N|1|note
/^961||d|0|0|o|z|0|n|N|1|int note
/^961||f|0|0|o|s|0|n|N|1|selector
/^961||g|0|0|o|q|0|n|N|1|ven addr
/^961||h|0|0|o|v|0|n|N|1|ven note
/^961||i|0|0|o|f|0|n|N|1|ven title#
/^961||l|0|0|o|r|0|n|N|1|requestor
/^961||m|0|0|o|p|0|n|N|1|blanket po
```

APPENDIX 3 – BIBLIOGRAPHIC LOAD TABLE FOR AUTHORITY CONTROL PROJECT

BIBLIOGRAPHIC LOAD PROFILE FOR AUTHORIZED HEADINGS (m2btab.lti)

```
|||0|0| | |0|n|G|0|@main="b"
|||0|0| | |0|n|G|0|@marc="b"
|||0|0| | |0|n|G|0|@atab="a"
|||0|0| | |0|n|G|0|@msg="Bib records will be overlaid"
/^999||m|0|12| | |0|n|G|0|#com="ins"@password=""
/^999||z|0|10| | |0|n|G|0|#com="recs"@recs="lti"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||s|0|30| | |0|n|G|0|#com="ip"@itemprefix=""
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag=" "
|||0|0| | |0|n|G|0|@ov_action="u"
|||0|0| | |0|n|G|0|@ov_protect=”b=F24,26,28-31V0123456789hk(970,971)
l(010:d)g(086:d)y(0..:d)i(0..:d)u(21.:d)u(24[67]:d)t(245:d)e(25.:d)p(2
..:d)r(3..:d)n(5..:d)u(740:d)q(7..:d)x(780:d)z(785:d)w(787:d)y(2..:d)y
(8..:d) c(:d)o(0..:d)”
|||0|0| | |0|n|G|0|@ov_rec_number="r"
|||0|0| | |0|n|G|0|@holdsymb=""
#|||0|0| | |0|n|G|0|@pre_map="m2bpre_map."
|||0|0| | |0|n|G|0|@locmerge="n"
#/^949||a|0|400| | |0|n|G|0|@comline
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@ldx="907"
|||0|0| | |0|n|G|0|@busy="n"
|||0|0| | |0|n|G|0|@busy_file="busy.lmarc"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="n"
|||0|0| | |0|n|G|0|@speriod="n"
008|| |35|3|b| |24|n|N|0|lang(b)
008|| |15|3|b| |89|n|N|0|country
L|| |7|1|b| |29|n|N|0|bib lvl
L|| |6|1|b| |30|n|N|0|mat type
/^999||a|0|30|b| |26|n|G|0|#com="bn"
/^999||b|0|5|b| |27|n|G|0|#com="co"
```

```
/^999||c|0|8|b| |28|n|G|0|#com="ct"
/^999||d|0|1|b| |29|n|G|0|#com="b1"
/^999||e|0|1|b| |30|n|G|0|#com="b2"
/^999||f|0|1|b| |31|n|G|0|#com="b3"
/^999||g|0|3|b| |24|n|G|0|#com="la"
/^999||k|0|3|b| |89|n|G|0|#com="cy"
001||%|0|0|b|o|0|y|N|0|
002-006||%|0|0|b|y|0|y|N|0|
007||%|0|0|b|y|0|y|N|0|
008-009||%|0|0|b|y|0|y|N|0|%008="y"
010||+|0|0|b|l|0|y|N|0|
020-024||+|0|0|b|i|0|y|N|0|
025-027||+|0|0|b|y|0|y|N|0|
028||ab|0|0|b|i|0|y|N|0|
030-049||+|0|0|b|y|0|y|N|0|
066||+|0|0|b|y|0|y|N|0|
069-074||+|0|0|b|y|0|y|N|0|
086||+|0|0|b|g|0|y|N|0|
088||+|0|0|b|y|0|y|N|0|
100-111||-w|0|0|b|a|0|y|N|0|
130||-w|0|0|b|u|0|y|N|0|
210-214||+|0|0|b|u|0|y|N|0|
222||+|0|0|b|y|0|y|N|0|
240||-w|0|0|b|t|0|y|N|0|
241-243||+|0|0|b|u|0|y|N|0|
245||+|0|0|b|t|0|y|N|0|%bracket="h"
246-247||+|0|0|b|u|0|y|N|0|
250||+|0|0|b|e|0|y|N|0|
254-257||+|0|0|b|y|0|y|N|0|
260-262||+|0|0|b|p|0|y|N|0|
263-265||+|0|0|b|y|0|y|N|0|
270||+|0|0|b|p|0|y|N|0|
300-399||+|0|0|b|r|0|y|N|0|
400-490||-w|0|0|b|s|0|y|N|0|
500-599||+|0|0|b|n|0|y|N|0|
600-699||-w|0|0|b|d|0|y|N|0|
700-711||-w|0|0|b|b|0|y|N|0|
730-740||-w|0|0|b|u|0|y|N|0|%bracket="h"
752-755||+|0|0|b|y|0|y|N|0|
760-777||+|0|0|b|q|0|y|N|0|
780||+|0|0|b|x|0|y|N|0|
785||+|0|0|b|z|0|y|N|0|
787||+|0|0|b|w|0|y|N|0|
800-811||-w|0|0|b|s|0|y|N|0|
830||-w|0|0|b|s|0|y|N|0|%bracket="h"
840||-w|0|0|b|s|0|y|N|0|
856||+|0|0|b|y|0|y|N|0|
880||+|0|0|b|y|0|y|N|0|
```

APPENDIX 4 – AUTHORITY LOAD TABLES

APPENDIX 4A – STANDARD AUTHORITY RECORD PROFILE (m2btab.a)

```
#for loading authority records via OCLC interactive interface
|||0|0| | |0|n|G|0|@main="a"
|||0|0| | |0|n|G|0|@marc="a"
|||0|0| | |0|n|G|0|@msg="Authority records will be created"
/^999||z|0|10| | |0|n|G|0|#com="atab"@atab="a"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||s|0|30| | |0|n|G|0|#com="ip"@itemprefix=""
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag=" "
|||0|0| | |0|n|G|0|@ov_action="o"
|||0|0| | |0|n|G|0|@ov_protect="a=F112-114V0123456789"
/^949 ||a|0|400| | |0|n|G|0|@comline
|||0|0| | |0|n|G|0|@holdsymb=""
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@leader_utf8="y"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="y"
|||0|0| | |0|n|G|0|@speriod="n"
001||%|0|0|a|y|0|y|N|0|%001(start="!-~",char="  -~")
003-005||%|0|0|a|y|0|y|N|0|
008||%|0|0|a|y|0|y|N|0|%008="y"
010||+|0|0|a|o|0|y|N|0|%strip_blanks="n"
014-099||+|0|0|a|y|0|y|N|0|
100-111||+|0|0|a|a|0|y|N|0|
130||+|0|0|a|t|0|y|N|0|
148||+|0|0|a|d|0|y|N|0|
150-151||+|0|0|a|d|0|y|N|0|
155||+|0|0|a|d|0|y|N|0|
180-185||+|0|0|a|y|0|y|N|0|
260||+|0|0|a|n|0|y|N|0|
360||+|0|0|a|n|0|y|N|0|
400-411||+|0|0|a|b|0|y|N|0|
430||+|0|0|a|u|0|y|N|0|
448||+|0|0|a|e|0|y|N|0|
450-451||+|0|0|a|e|0|y|N|0|
455||+|0|0|a|e|0|y|N|0|
480-485||+|0|0|a|y|0|y|N|0|
500-511||+|0|0|a|c|0|y|N|0|
530||+|0|0|a|v|0|y|N|0|
548||+|0|0|a|f|0|y|N|0|
550-551||+|0|0|a|f|0|y|N|0|
555||+|0|0|a|f|0|y|N|0|
580-585||+|0|0|a|y|0|y|N|0|
640-646||+|0|0|a|y|0|y|N|0|
663-688||+|0|0|a|n|0|y|N|0|
700-711||+|0|0|a|c|0|y|N|0|
730||+|0|0|a|v|0|y|N|0|
748||+|0|0|a|f|0|y|N|0|
750-751||+|0|0|a|f|0|y|N|0|
755||+|0|0|a|f|0|y|N|0|
780-785||+|0|0|a|y|0|y|N|0|
788||+|0|0|a|n|0|y|N|0|
856||+|0|0|a|y|0|y|N|0|
880||+|0|0|a|y|0|y|N|0|
```

APPENDIX 4B – STANDARD NAME AUTHORITY PROFILE (m2btab.anam)

```
#standard name/title authority record load table
|||0|0| | |0|n|G|0|@main="a"
|||0|0| | |0|n|G|0|@marc="a"
|||0|0| | |0|n|G|0|@msg="Name authority records will be created"
|||0|0| | |0|n|G|0|#com="atab"@atab="anam"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag="oz[ov_1xx]"
|||0|0| | |0|n|G|0|@ov_action="o"
|||0|0| | |0|n|G|0|@ov_protect="a=F112-114V0123456789"
|||0|0| | |0|n|G|0|@holdsymb=""
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@leader_utf8="y"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="y"
|||0|0| | |0|n|G|0|@speriod="n"
001||%|0|0|a|y|0|y|N|0|%001(start="!-~",char="  -~")
003-005||%|0|0|a|y|0|y|N|0|
008||%|0|0|a|y|0|y|N|0|%008="y"
010||+|0|0|a|o|0|y|N|0|%strip_blanks="n"
014-099||+|0|0|a|y|0|y|N|0|
100-111||+|0|0|a|a|0|y|N|0|
130||+|0|0|a|t|0|y|N|0|
151||+|0|0|a|a|0|y|N|0|
260||+|0|0|a|n|0|y|N|0|
360||+|0|0|a|n|0|y|N|0|
400-411||+|0|0|a|b|0|y|N|0|
430||+|0|0|a|u|0|y|N|0|
451||+|0|0|a|b|0|y|N|0|
500-511||+|0|0|a|c|0|y|N|0|
530||+|0|0|a|v|0|y|N|0|
551||+|0|0|a|c|0|y|N|0|
640-646||+|0|0|a|y|0|y|N|0|
663-688||+|0|0|a|n|0|y|N|0|
700-711||+|0|0|a|c|0|y|N|0|
730||+|0|0|a|v|0|y|N|0|
748||+|0|0|a|c|0|y|N|0|
751||+|0|0|a|c|0|y|N|0|
788||+|0|0|a|n|0|y|N|0|
856||+|0|0|a|y|0|y|N|0|
880||+|0|0|a|y|0|y|N|0|
```

APPENDIX 4C – STANDARD SUBJECT AUTHORITY LOAD PROFILE (m2btab.asub)

```
#standard subject authority record load table
|||0|0| | |0|n|G|0|@main="a"
|||0|0| | |0|n|G|0|@marc="a"
|||0|0| | |0|n|G|0|@msg="Subject authority records will be created"
|||0|0| | |0|n|G|0|#com="atab"@atab="asub"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag="oz[ov_1xx]"
|||0|0| | |0|n|G|0|@ov_action="o"
|||0|0| | |0|n|G|0|@ov_protect="a=F112-114V0123456789"
|||0|0| | |0|n|G|0|@holdsymb=""
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@callnum="nnnny"
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@leader_utf8="y"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="y"
|||0|0| | |0|n|G|0|@speriod="n"
001||%|0|0|a|y|0|y|N|0|%001(start="!-~",char="  -~")
003-005||%|0|0|a|y|0|y|N|0|
008||%|0|0|a|y|0|y|N|0|%008="y"
010||+|0|0|a|o|0|y|N|0|%strip_blanks="n"
014-099||+|0|0|a|y|0|y|N|0|
100-111||+|0|0|a|d|0|y|N|0|
130||+|0|0|a|d|0|y|N|0|
148||+|0|0|a|d|0|y|N|0|
150-151||+|0|0|a|d|0|y|N|0|
155||+|0|0|a|d|0|y|N|0|
180-185||+|0|0|a|y|0|y|N|0|
260||+|0|0|a|n|0|y|N|0|
360||+|0|0|a|n|0|y|N|0|
400-411||+|0|0|a|e|0|y|N|0|
430||+|0|0|a|e|0|y|N|0|
448||+|0|0|a|e|0|y|N|0|
450-451||+|0|0|a|e|0|y|N|0|
455||+|0|0|a|e|0|y|N|0|
480-485||+|0|0|a|y|0|y|N|0|
500-511||+|0|0|a|f|0|y|N|0|
530||+|0|0|a|f|0|y|N|0|
548||+|0|0|a|f|0|y|N|0|
550-551||+|0|0|a|f|0|y|N|0|
555||+|0|0|a|f|0|y|N|0|
580-585||+|0|0|a|y|0|y|N|0|
640-646||+|0|0|a|y|0|y|N|0|
663-688||+|0|0|a|n|0|y|N|0|
700-711||+|0|0|a|f|0|y|N|0|
730||+|0|0|a|f|0|y|N|0|
748||+|0|0|a|f|0|y|N|0|
750-751||+|0|0|a|f|0|y|N|0|
755||+|0|0|a|f|0|y|N|0|
780-785||+|0|0|a|y|0|y|N|0|
788||+|0|0|a|n|0|y|N|0|
856||+|0|0|a|y|0|y|N|0|
880||+|0|0|a|y|0|y|N|0|
```

APPENDIX 5 – PATRON LOAD TABLE

STANDARD PATRON LOAD TABLE (m2btab.P)

```
#standard patron record load table
|||0|0| | |0|n|G|0|@main="p"
|||0|0| | |0|n|G|0|@marc=""
|||0|0| | |0|n|G|0|@msg="Patron records will be created"
/^999||z|0|10| | |0|n|G|0|@recs="p"
/^999||x|0|1| | |0|n|G|0|#com="clsi"@clsi="n"
/^999||w|0|1| | |0|n|N|0|#com="test"@test="n"
/^999||v|0|1| | |0|n|N|0|#com="init"@init="n"
/^999||u|0|1| | |0|n|N|0|#com="disp"@disp="n"
/^999||o|0|20| | |0|n|G|0|#com="dflt"@dflt=""
#if tag "b" is the ov_tag, remove it from the @ov_protect string
/^999||t|0|10| | |0|n|G|0|#com="ov"@ov_tag=" "
|||0|0| | |0|n|G|0|@ov_action="o"
|||0|0| | |0|n|G|0|@ov_protect="p=F48-50,54-56,95,96,99,101-105,122-
125,158,163,263,268-271Vbmxy0123456789="
|||0|0| | |0|n|G|0|@holdsymb=""
|||0|0| | |0|n|G|0|@bldmarc=""
|||0|0| | |0|n|G|0|@callnum="nynnn"
|||0|0| | |0|n|G|0|@diac=""
|||0|0| | |0|n|G|0|@diac_sub_table="usmarc"
|||0|0| | |0|n|G|0|@ldx=""
|||0|0| | |0|n|G|0|@busy="y"
|||0|0| | |0|n|G|0|@title="n"
|||0|0| | |0|n|G|0|@cdate="n"
|||0|0| | |0|n|G|0|@year_2000="20"
|||0|0| | |0|n|G|0|@speriod="n"
#fixed-length fields
079||a|0|10|p| |163|n|N|0|last circ date
080||a|0|10|p| |43|n|N|0|exp date
081||a|0|1|p| |44|n|N|0|pcode1
082||a|0|1|p| |45|n|N|0|pcode2
083||a|0|3|p| |46|n|N|0|pcode3
084||a|0|3|p| |47|n|N|0|ptype
085||a|0|5|p| |53|n|N|0|home libr
086||a|0|1|p| |56|n|N|0|mblock
087||a|0|1|p| |54|n|N|0|pmessage
088||a|0|4|p| |126|n|N|0|pcode4
089||a|0|10|p| |51|n|N|0|birth date
```

```
#variable-length fields 020||+|0|0|p|u|0|n|N|0|id
030||+|0|0|p|b|0|n|N|0|barcode
100||+|0|0|p|n|0|n|N|0|name
220||+|0|0|p|a|0|n|N|0|address1
225||+|0|0|p|t|0|n|N|0|tel1
230||+|0|0|p|h|0|n|N|0|address2
235||+|0|0|p|p|0|n|N|0|tel2
400||+|0|0|p|m|0|n|N|0|message
500||+|0|0|p|x|0|n|N|0|note
550||+|0|0|p|z|0|n|N|0|email
600||a|0|0|p|=|0|n|N|0|%encryptpin="y"
856||+|0|0|p|y|0|n|N|0|image
```

WHAT'S NEW (2016-02-04)

1. Replaced references to Millennium with Sierra/Millennium
2. Removed references to *Release Silver (2001) and later*
3. Added description of Day Three workshop (p. 2)
4. Removed statement that the @leader_utf8 trigger should only be used with systems with Unicode storage (p. 44)
5. Added descriptions of ov_tag confirmation tests ov_edition, ov_place, ov_verify_marc_tag and ov_year (p. 60)
6. Removed use of regular expressions with the %replace special processing function (p. 68)

WHAT'S NEW (2011-07-07)

1. Revised instructions in section HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL) (p. 101-103).

WHAT'S NEW (2011-04-06)

1. Added WHAT'S NEW page (IUG request)
2. Added revision date (IUG request)
3. Removed references to *Release 2005 and later*
4. Added number of volume records a single bibliographic record can link to (p. 3)
5. Added reference to the volume record type (p. 6)
6. In NOTE - Clarified that default values will be derived from Record Templates (p. 14)
7. Updated m2btab Element 3, Subfield, documentation. Removed references to infrequently called options F and K and added option S including an example (p. 28-30).
8. Removed reference to RLIN function in @ldx trigger (p. 44)
9. Updated @ov_attach_delete and @ov_attach_insert to include new functionality to add an optional value of a MARC tag (p. 51)
10. Corrected the documentation of @ov_priority_action="a" from overlay to attach (p. 52)
11. Added new NOTE about copying @ov_protect from example load tables (p. 53)
12. Removed reference to RLIN bcode function in %001 trigger (p. 65)
13. Removed reference to RLIN in %replace trigger (p. 69)
14. Removed reference to RLIN command function "ri" (p. 71-73).
15. Renamed "Load Profile Training Module" to THE LOAD PROFILE MAINTENANCE MODULE (p. 88)

16. Renamed and rewrote section “Editing an Existing Load Table (m2btab).” The new section is HOW TO EDIT AN EXISTING LOAD TABLE (M2BTAB) (p. 88-92)
17. Clarified that m2btabs can be downloaded to an FTP server not any computer (p. 90)
18. Renamed and rewrote section “Creating a New Load Table (m2btab). The new section is HOW TO CREATE A NEW LOAD TABLE (M2BTAB) (p. 93-97)
19. Renamed and rewrote section “Creating a New Translation Table.” The new section is HOW TO CREATE A NEW TRANSLATION TABLE (M2BMAP) (p. 98-100)
20. Renamed and rewrote section “Creating New Loading Options.” The new section is HOW TO CREATE A NEW LOAD BUTTON (M.MARCLOAD.LOCAL) (p. 101-103)
21. Renamed the “Commenting Lines in m2btab with #” section into a new section HOW TO ADD A NOTE TO A LOAD TABLE (p. 104)
22. Added two new sections HOW TO TURN OFF AN INSTRUCTION IN A LOAD TABLE (M2BTAB) and HOW TO LOAD RECORDS AS NEW (p. 104)
23. Renamed the “Loading Alternate Alphabets” section into a new section HOW TO LOAD ALTERNATE ALPHABETS AND DIACRITICS (p. 105)
24. Added a new section PROFILING FROM START TO FINISH which includes re-organized and updated documentation on HOW TO ANALYZE DATA and an update to ISSUES TO BE CONSIDERED WHEN CREATING A NEW LOAD TABLE (p. 106- 111)
25. Renamed and rewrote section “Testing New m2btab and m2bmap Files.” The new section is HOW TO TEST A NEW M2BTAB AND/OR M2BMAP FILE (p. 112-115)