For Internal Use Only

How to Load YBP files

Release 14.2

Last update: 7 October 2001

Ex Libris

Table of Contents

1 Overview

The Yankee Book Peddler (YBP) loader (p_file_96) batch service loads MARC records, creates associated Holdings, Item, and Order records, and encumbers the appropriate fund.

The input file from YBP is a standard MARC record with extra fields:

`049 $$a -`	contains the OCLC codes for local library's locations.
`049 $$b -`	contains the item barcode (not required)
`980 $$a -`	invoice date (YYYYMMDD)
`980 $$d -`	vendor note
`980 $$f -`	invoice #
`980 $$g -`	number of copies
`980 $$b -`	list price
`980 $$e -`	net price
`981 $$b -`	fund code

If there is only a 980 $b (user-defined) field, then an encumbrance is created with the 980 $b field. Otherwise, the encumbrance is created with the 980 $e (user-defined) field, and the 980 $b field is not used. Additionally, the loader program accepts prices in two forms: with or without two decimal places (for example 5995 or 59.95). ALEPH checks for the existence of a decimal point.

A 981 $d field maps to Z68-LIBRARY-NOTE, and a 980 $f field maps to the Z68-VENDOR-REFERENCE-NO in the Z68 (acquisition order records) table.

To load YBP files, carry out the following procedures:
1. Specify options in tab_yankee_def.
2. Specifying the Order Sequence.
3. Create the fix routine, and specify the fix routine in tab_fix.
4. Run the YBP loader from the Web Services menu/ Unix prompt.
5. Carry out multiple test runs.

These procedures are explained in the rest of this document.

2 Specifying options in tab_yankee_def

You must specify the following in your BIB library’s tab_yankee_def (XXX01/data_tab/tab_yankee_def) table. For SUB-LIBRARY and COLLECTION, the options are the values from 049 $$a -- in this example, XZLA and XZLL. Thus records with a holdings code of XZLA are assigned the XLINC sublibrary, and records with a holdings code of XZLL are assigned the XLAW sublibrary.

```
CATALOGER                      ########## YBPSR
ORDER-SEQ                      ########## mono
Z30-SUB-LIBRARY                XZLA       XLINC
Z30-SUB-LIBRARY                XZLL       XLAW
Z30-COLLECTION                 XZLA       GEN
Z30-COLLECTION                 XZLL       GEN
Z30-MATERIAL                   ########## BOOK
Z30-ITEM-STATUS                ########## 01
Z30-CATALOGER                  ########## YBPSR
Z30-CALL-NO-TYPE               ########## 1
Z30-ITEM-PROCESS-STATUS        ########## OR
Z68-SUB-LIBRARY               XZLA       XLINC
Z68-SUB-LIBRARY               XZLL       XLAW
Z68-ORDER-TYPE                ########## M
Z68-ORDER-STATUS              ########## SV
Z68-ARRIVAL-STATUS            ########## No
Z68-VENDOR-CODE               ########## YBP
Z68-RUSH                      ########## N
Z68-DELIVERY-TYPE             ########## S
Z68-TARGET-DIRECT             ########## N
Z68-E-CURRENCY                ########## USD
Z68-METHOD-OF-ACQUISITION     ########## P
Z68-ORDER-DELIVERY-TYPE       ########## LI
Z68-MATERIAL-TYPE             ########## M
Z601-VENDOR-CODE              ########## YBP
Z601-USER-NAME                ########## LOAD
z601-CURRENCY                 ########## USD
z601-TYPE                     ########## ENC
```

3 Specifying the Order Sequence

From the ADM library (XXX50), go into UTIL/G/2 (Display/Update library parameters), and create or identify an existing order number sequence to use for this group of orders. If no sequence is specified, the next order number is the next in line for the global order number. To have all of the YBP order follow the same sequence, specify the sequence with the line:

```
ORDER-SEQ                     ########## mono
```

The above order sequence corresponds to the following line in UTIL/G/2:

```
22. last-order-no-mono          12     y      U  M
```

When the above sequence is in use, all YBP Orders start with the prefix "M" – for example, M345.

4 Creating the fix routine, and specifying the fix routine in tab_fix

The yankee_add_own.fix table (in the XXX01/data_tab/import directory) specifies the OWN field for records. All incoming records should have an OWN field. This table also moves the 001 value to the 035 value. This fix can have any name, but it must be included in the tab_fix section specified in the command line (see #9 below).

The yankee_add_own.fix table looks like this:

```
1 LDR           ADD-FIELD              OWN  ,L,$$aOL
1 001##         CHANGE-FIELD           035
1 035##         CHANGE-SUBFIELD        a
```

The first line adds the OWN field with “OL” in subfield a. The last two lines move the YBP system number from 001 to 035 $$a.

Add the following line in tab_fix in the XXX01/data_tab directory:

```
YANK   fix_doc_do_file_08            yankee_add_own.fix
```

5 Instructions for running the YBP loader from the Web Services menu

The YBP loader is run from the ALEPH Staff Services menu and is listed as Load YBP Records or (p_file_96) under the rubric Load Catalog Records.

Note the following:

- ❑ If the Input file is located in the $data_scratch directory, simply enter the name of the file.
- ❑ If the input file is located in another directory, enter the path relative to the Scratch directory, or enter the full path name.

  - ♦ Example of a relative path: `../source/filename` .
  - ♦ Example of a full path: `/aleph/a50_5/usm01/source/filename`

Enter the day and hour you want the procedure to run. The system uses the 24 hour clock, so if you want to run the procedure at 11pm, enter “23”.

6 Instructions for running the YBP loader from the Unix prompt

Enter the following command line from XXX01 (BIB library):

```
csh -f $aleph_proc/p_file_96  USM01,yank509.mrc,tab_yankee_def,
USM60,USM50,Y,N,Y,yank,Y
```

Note that line contains the following parameters:
[1]: BIB library (`USM01` in example)
[2]: MARC input file. The script converts it into ALEPH sequential (`yank509.mrc`)
[3]: defaults table - should be located in the bib data_tab directory. (`tab_yankee_def`)
[4]: HOL library - if blank then no HOL records are created. (`USM60`)
[5]: ADM library - if blank then no z30 (item/adm) records are created (`USM50`)
[6]: create Z30 Y/N (Item) (`Y`)
[7]: create Z68 Y/N (Order) (`N`)
[8]: create Z601 Y/N (Encumbrance) (`Y`)
[9]: Name of tab_fix section to run before loading bib records (`yank`)
[10]: Update database Y/N (enter N to do a test run) (`Y`)

To export the log to the data_scratch directory (as opposed to the log displaying on-screen), add a ">" specification. For example:

```
csh –f $aleph_proc/p_file_96 BCT01,11100109.MRC,
tab_yankee_def,,BCT50,Y,Y,Y,YANK,Y & p_file_96.log.0124 &
```

7 Multiple Test Runs with the Same Input File

ALEPH accepts duplicate BIB records but does not accept items with duplicate barcodes. How can you do multiple test runs which create item records, holdings records, and so on?

In a relatively small (for example under 50,000 record) database the following scenario should be practical:

From the Web Services menu, run the Export Tables (p_file_03) batch service listed under the Manage Database Tables rubric.

To export all library tables and save them in a file, choose the All Tables option. This exports all the tables. Run this batch service for the XXX01 (BIB), XXX50 (ADM), and XXX60 (HOL) libraries.

If you want to do a re-run with the same data you can run the Import Tables (p_file_04) batch service to restore the files to how they were.

Note that when you run p_file_04, any updates made to the records online between the export and the re-import are lost. Care must be taken when running p_file_04 to ensure that the production files are not specified as the target instead of the test.

8 The p_file_96 error log:

You may see messages like this in the error log:

```
Error: problems in Z68 record:  z68-rec-key - 00004333600001
No errors: 02
000000001- 0309
000000002- 0312
```

The “No. errors” is the **number** of errors and is followed by the specific errors. The number on the left (“000000001”) is just a sequence number; the number on the right (“0309”) is the actual error. These errors are produced by the butil/b_file_96_a.cbl program. This program calls check_record/check_z30.cbl and _z68.cbl (which, in turn, calls check_z68 a.c). The last two programs pass error codes corresponding to particular fields back to the p_file_96. Generally, errors are due to an incorrect parameter being entered into the xxx01/tab/tab_yankee_def table – or a parameter being omitted. (The butil/p_file_96_fill_z30, *z68,* and z601 programs read the tab_yankee_def and load values into the records.)

To view more information in the log when running the job, type the following

```
$ setenv DEBUG Y
```

Then restart the batch queue (UTIL/C/3 then UTIL/C/2). The log shows the Z30, Z68 and Z601 Oracle tables that are being created.