

For Internal Use Only



How to Batch Load BIB Records

14.2 and higher

Last update: 13 March 2003

Table of Contents

1	OVERVIEW	3
2	CHECK INPUT FILE AGAINST DATABASE (P_MANAGE_36).....	4
	2.1 Setting up tab_match	4
	2.2 Load Catalog Records (p_manage_18).....	6
3	MODIFY MARC RECORD FILE (P_FILE_08)	7
	3.1 Using fixes to limit which records get loaded	8
4	CREATE HOLDINGS AND ITEM RECORDS USING BIB DATA (P_MANAGE_50).....	9

1 Overview

ALEPH contains four batch services that can be used to load bibliographic (BIB) records with options:

- ❑ **Check Input File Against Database (p_manage_36).** Checks an input file against the database and separates the file into three files – exact matches, non-matches, and multiple matches.
- ❑ **Load Catalog Records (p_manage_18).** Loads BIB records into the database, with the option of either updating existing records or of adding new records
- ❑ **Modify MARC Record File (p_file_08).** Performs a set of fixes on a file of bibliographic records. Fixes can either be performed before the records are loaded into the database (with p_file_08), or while the records are being loaded in, by specifying a fix section in p_manage_18.
- ❑ **Create Holdings and Item Records Using BIB Data (p_manage_50).** Creates Items and Holdings records based on embedded information in the BIB record in the ALEPH database.

2 Check Input File Against Database (p_manage_36)

The p_manage_36 batch service compares records in a file outside the ALEPH database to records in the ALEPH database, based on an index specified in the tab_match table in the library's tab directory. Four files are created:

1. Records that do not match anything in the database.
2. Records that find an exact match in the database.
3. Records that find multiple matches in the database.
4. A list of record numbers that corresponds to the file from 2 (records that find an exact match in the database).

The external file should be in ALEPH sequential format.

To convert a file from MARC format to ALEPH sequential format:

1. Run the p_file_01 and p_file_02 batch services.
2. Set up the match section in the match table XXX01/tab/tab_match.

2.1 Setting up tab_match

The following is an example of a match section from tab_match:

```
! 1                2                3
!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
YBP  match_doc_uid          I-035
OCLC match_doc_gen          TYPE=IND,TAG=035##,CODE=035,SUBFIELD=a
OCLC match_doc_gen          TYPE=IND,TAG=599##,CODE=SYS,SUBFIELD=a
TEST match_doc_acc          match_doc_acc
```

Column 1 is the match section – all lines of a particular section will be used. Specify this match section in p_manage_36. Where there are two lines in a section, the program tries to match on the first line, and if that fails, tries to match on the second line.

Column 2 is the match type. Most users only need the match_doc_gen type.

Column 3 contains the parameters for the match type. In the above example, the OCLC section checks the contents of tag 035 in the input file (TAG=035##) against the direct index “035”, as specified in tab00.lng. The second OCLC line indicates that the 599 tag is checked in the incoming file against the direct index “SYS”. In both cases, the data in subfield \$a of the incoming file is being examined.

Notes:

When using match_doc_uid, the third column is either the tag from the incoming file (T-035) or the direct index in ALEPH (I-035). When using this, the tag in the incoming file must be the tag that is indexed by the direct index in ALEPH. The program checks the tab11 value for the tag and matches against that direct index (and vice versa).

When using `match_doc_acc`, you must specify another table that lists the tags to match against.

Output Files from `p_manage_36`

There are three output files from `p_manage_36`:

Output file 1. This includes, in sequential format, the records that did not match any documents in the database. This file must be loaded into the database with the `p_manage_18` batch service, using the “Add new records to the database” option.

Output file 2. This includes, in sequential format, the records that find an exact match in the document. The system numbers on the left-hand side of the file are the ALEPH system numbers that match. These records should be loaded into the database with the `p_manage_18` batch service, using one of the replace options.

Output file 3. This includes, in sequential format, the records that find multiple matches. These should probably be looked over by a staff member.

A fourth output file is created, corresponding to Output file 2. This fourth file is a list of system numbers of the records in output file 2. Libraries can choose to use this file as input for the `p_print_03` batch service, which prints out a copy of the existing BIB records before they are overlaid by running `p_manage_18`.

3 Load Catalog Records (p_manage_18)

There are two options to choose from when running p_manage_18:

- Add new records to the database
- Update current records in the database

If you choose the option “Add new records to the database”, bear in mind that there is no check on any indexes or on anything else – the records just load starting with the next DOC number. Libraries can choose to “fix” the record when it is loaded – for example, by moving the vendor’s unique number from the 001 field to the 035 field.

Choosing the “Update current records in the database” option allows you to select one of four sub-options:

- Append fields to a record
- Replace fields within a record
- Delete fields in a record
- Replace entire record

The “Replace entire record” sub-option can be used in conjunction with the “fix_doc_overlay” fix routine to perform a merge on the existing and incoming documents. When using the fix_doc_overlay fix routine, you must specify, in the tab_fix table, the merge section from tab_doc_merge to use:

```
MRG    fix_doc_overlay          03
```

For example, the following fix section uses section 03 from tab_doc_merge:

```
03 2 Y #####  
03 1 Y 783##
```

For more information on setting up merge sections, see the header of tab_doc_merge.

The other available options when updating records – Append fields to a record, Replace fields within a record, and Delete fields in a record - are self-explanatory.

Finally, p_manage_18 creates a secondary output file that contains the document numbers of the records that were overlaid. This is similar to the secondary output file created with p_manage_36. The file of document numbers can easily be used as an input file for p_manage_50.

4 Modify MARC Record File (p_file_08)

Libraries can choose to modify a MARC record file before loading or during loading into the ALEPH database. Common examples of fixes are

1. Moving the vendor's number from 001 to 035.
2. Adding the library's OWN field to the record.
3. Updating the date in the 008 with the open date of the record.

Many standard fix routines are provided by ALEPH, but there are times when a library would like to perform a customized fix on a document. This can be done with a fix script. Fix scripts are stored in the \$data_tab/import directory of the BIB library (xxx01/tab/import). An example of a fix script is

```
!-!!!!-!!-!-!!!-!!!-!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!
!!!!>
1 001                COPY-SYSTEM-NUMBER                035  ,L,a
1 035##              REPLACE-STRING                    ocm,(OCoLC)
1 035##              REPLACE-STRING                    ^,
1 LDR                ADD-FIELD                          OWN  ,L,$$aLIN
1 001                DELETE-FIELD
1 003                DELETE-FIELD
1                    SORT-FIELDS
```

The above script moves the contents of the 001 field to the 035 field, first indicator 9, and give it a subfield a. (L is for the script of the field – Latin). Then, it will replace the string “ocm” with “(OCoLC)”, remove spaces from the 035 field, and add an OWN field with the contents “LIN”.

For more information on commands available in the fix scripts and their parameters, see the document, *How to Set Up a Script for the Correction of Records in ALEPH Sequential Format*.

To apply the above script, specify the fix “fix_doc_do_file_08” and include the file name in column 3 of tab_fix. For example, if the above file were called oclctest.fix, the line in tab_fix would be:

```
! 1                2                3
!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
LOAD  fix_doc_do_file_08                oclctest.fix
```

Then, you can specify the section “LOAD” in a number of places. (Note: the text “LOAD” is merely an example – libraries can choose any name for a fix section, and then refer to that section later.)

If you would like to modify the records before they are loaded in the database, run the p_file_08 batch service and specify “oclctest.fix” in the processing script section.

If you would like to modify the records as they are loaded in the database, load the records using p_manage_18 and specify LOAD as the fix section.

4.1 Using fixes to limit which records get loaded

Some libraries would like to limit which records are loaded, based on information in the fixed fields. The following two fix scripts can be used to place limitations:

- OCLC cancelled records – if position 022 of the leader equals the binary number 003, then do not load the record.

```
!1 2      3 4 5 6 7      8                                9
!-!!!!!!-!!-!!-!!!-!!!-!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!
!!!!!!>
1 LDR          022 022          COND-LOAD-VAL-POS          N,\003
```

- OCLC replaced records – if position 022 of the leader equals the binary number 017, then do not load the record.

```
!1 2      3 4 5 6 7      8                                9
!-!!!!!!-!!-!!-!!!-!!!-!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!
!!!!!!>
1 LDR          022 022          COND-LOAD-VAL-POS          N,\017
```

If the two scripts above are run using p_file_08, that is, before loading into the database, p_file_08 separates the file into two files – one file containing records where the leader position matches, and the other file where the leader position does not match. These two files can then be loaded, not loaded or modified as the library sees fit.

5 Create Holdings and Item Records Using BIB Data (p_manage_50)

If new BIB records that are loaded contain embedded item and holding information, libraries can use the p_manage_50 batch service to create holding and items records.

p_manage_50 takes the following parameters:

1. An input file containing a list of document numbers for which holdings and/or items will be created. The file should be located in \$alephe_scratch, and can either be created with p_ret_01 (Retrieve Catalog Records) or p_manage_18. If this field is left blank, the program checks for a range of document numbers.
2. Range of document numbers. These two parameters are used only if the input file is not specified.
3. ADM Library – specify the administrative library name (for example, EXU50) if you want to create items. If this field is left blank, items cannot be created.
4. HOL library – specify the Holdings library name (for example, EXU60) if you want to create holdings records. If this field is left blank, holdings records are not created.
5. Document expand code -- this must match one of the expand routines defined in column 1 of the tab_expand table. The purpose of this field is to allow the 'main field' to be manipulated before item and holdings records are created based on the information contained in the 'main field'. For example, if the sublibrary of the resulting item records depends on several fields scattered through the bibliographic record, the library can use an appropriate expand program to add the appropriate sublibrary value in the main field. If this field is used, then the value of the 'Delete Item Field' is set to 'N'. When using expand programs to manipulate the bibliographic record before creating the item/holdings records, it is preferable not delete the item field.
6. Main field – enter the field where the HOL/item information is stored – for example, 949##.
7. Delete main field – if you choose Yes, the field specified above is deleted after HOL and ITM records are created.
8. Item creation table – the name of the table in the \$data_tab directory that has the parameters for creating items. This is normally called tab_hol_item_create.

Examples of entries in this table are listed below.

```

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!
CREATE-ITEM-2ND          0
Z30-BARCODE              p          N
Z30-MATERIAL             k          N
Z30-ITEM-STATUS         x          N
SUB-LIBRARY              b          N b

```


N - (new) create holdings record only if the bibliographic record does not already have any holdings records.

R - (replace) delete any existing holdings records and replace with new ones.

A - (append) add new holdings records whether or not holdings records already exist.

11. Item Creation Mode - Modes are N, R, A and M. This field is used only if an administrative library has been specified, and defaults to N if left blank.

N - (new) create items only if the bibliographic record does not have already any items.

R - (replace) delete any existing item records and replace with new ones.

A - (append) add new items whether or not items already exist.

M - (match) Create item records in addition to existing items, but replace items that match the barcode specified.

12. Holdings / Item Creation Interaction Flag - values are A and N. This field is used only if both the administrative and the holdings libraries have been specified, and defaults to N if left blank.

A - (all items) derive new holdings records from all items that end up attached to the bibliographic record; i.e., both previously existing ones and new ones.

N - (only new items) derive new holdings records only from the items created from the information embedded in the bibliographic record.

13. BIB, ADM and HOL catalogers and cataloger level – specify the catalogers and levels to be used when creating new records or updating existing records.