USER DOCUMENTATION

How to Load Records Into a Union Catalog

Release 15.2 and later
Last Update: February 20, 2002

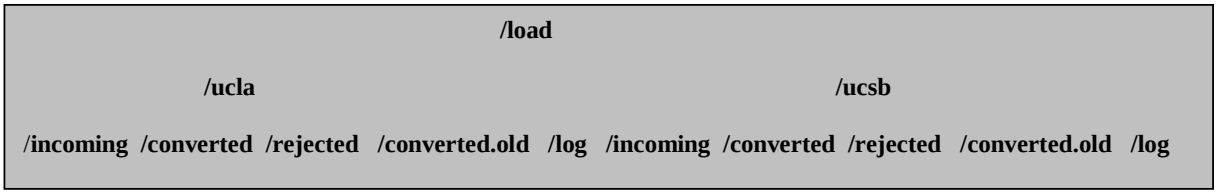
Table of Contents

1 Introduction

The process of loading records into the ALEPH Union Catalog has two stages, conversion and load. The conversion stage includes all data conversion, enrichment and validation; the load stage includes all updates to the database.

2 Setup and Workflow

ALEPH Union Catalogs [XXX90] contain a directory called **load** and this directory contains a subdirectory for each input stream. An input stream is any contributing library or organization that had distinct conversion needs. Each input stream's directory has the following five sub-directories; incoming, converted, rejected, converted.old, and log. The ALEPH configuration tables that control record conversion and validation are all located in the data_tab directory of the Union Catalog [xxx90/tab]



Files of incoming records are placed in the incoming directory of the appropriate input stream in MARC format. The p_union_03 process picks the records up from there, converts them, validates them, and moves them to one of two directories; *converted* or *rejected*. The logfile of the p_union_03 process is written to the log directory. A second process, p_union_13, loads records into the database. p_union_13 takes files from the converted directory and moves them to the *converted.old* directory. Records cannot be rejected at the load stage. The logfile of the p_union_13 process is written to the log directory.

3 Record Conversion

There are three stages to record conversion: conversion from the incoming format into ALEPH sequential; field-by-field data conversion or enrichment; and record validation. During the record conversion stage, a file of MARC records is converted into ALEPH sequential format (p_file_01 and p_file_02) and the incoming character set is converted from ANSEL to Unicode. There are no configuration tables consulted at this stage. Once the records are in ALEPH sequential format, the records are "fixed" – this process is regulated by tab_fix, an ALEPH configuration table located in the data_tab directory of the Union Catalog [xxx90/tab]. Tab_fix should include a section for each input stream and can be used to call "fix" programs that are specific to the input stream (and were written by Ex Libris according the specifications of CDL) as well as generic ALEPH "fix" programs. A list of the specific programs and what they do is contained in Appendix A, a list of generic fix_doc programs can be found in the Database Management Guide. Note that fix_doc programs should be

entered in the table in the order in which they should be performed. In the example below, fix_doc_ucla takes an incoming 901 field and manipulates it in order to create the 901 fields that will be present in the database. The second fix_doc routine, fix_doc_sid_cdl, takes the 901 field, itself a product of fix_doc_ucla, and uses it to generate an SID field.

3.1 tab_fix

```
UCLA    fix_doc_ucla
UCLA    fix_doc_sid_cdl
```

Key to the table:

Column 1 – Routine name/input stream
Column 2 – Program name
Column 3 – Program arguments

A second ALEPH configuration table, check_doc, calls record validation routines for each input stream. The check_doc table works in conjunction with check_doc.eng, check_doc_doc and check_doc_mandatory. Each input stream uses a location specific validation program as well as generic routines. Check_doc configures the combination of specific and generic routines that will be performed on records within each input stream while check_doc_doc contains a list of fields that should be present in all incoming records. Note that each validation routine has an error code associated with it; the severity of the error and the text of the error are set in check_doc_mandatory and check_doc.eng respectively.

3.2 check_doc

```
UCLA            check_doc_doc
UCLA            check_doc_ucla
```

Key to the table:

Column 1 – Routine name/input stream
Column 2 – table name/program name

3.3 check_doc_doc:

```
OC XX 5002 01 01 245##
OC XX 5008 01 99 852##
```

Key to part one of the table:

Column 1 – Section id [OC = Occurrence]
Column 2 – Record format [BK,SE,etc]
Column 3 – Error message code
Column 4 – Minimum number of occurrences
Column 5 – Maximum number of occurrences
Column 6-10 – Fields that must occur

```
D  BK 7003 2450#   Y 1####   N
D  BK 7004 2451#   Y 1####   Y
```

Key to part two of the table:

Column 1 – Section id [D = Dependency]
Column 2 – Record format [BK,SE,etc]
Column 3 – Error message number
Column 4 – Field code
Column 5 – Type of dependency [Y- present, N- not present]
Column 6 – Second field code
Column 7 – Type of dependency [Y- present, N- not present]

3.4 check_doc.lng

```
5008 L Required 852 field is missing.
5009 L 901 field subfield "c" should be "LAD".
```

Key to the table:

Column 1 – Error message number
Column 2 – Alpha (should always be L)
Column 3 – Error message text

3.5 check_doc_mandatory

```
UCLA        5008 M Required 852 field
UCLA        5009 M 901 $$c must be "LAD"
```

Key to the table:

Column 1 – Routine name
Column 2 – Error message number
Column 3 – Error type [M= record cannot be loaded, T=record will be loaded and will have ERR field]

3.6 p-union-03

The batch process that converts records is called p_union_03. It can be run with the following parameters; Active library, input stream. If an input stream is specified then the process will only check the incoming directory for that stream, if no input stream is specified, the process checks all incoming directories. Note that the list of allowed input streams will vary from installation to installation.

4 Record Loading

The load process, p_union_13, takes records that have already been converted, fixed and validated, and loads them into the ALEPH Union Catalog database. The records for load are taken from the converted directory of each input stream. The load process itself does the following; loads new records, overlays or deletes existing records, indexes new and updated records, and builds empty record equivalency tables (Z120) or flags existing equivalency tables for update.

In order to determine if a record should be loaded as new or update, the load process searches the SID field of the incoming records in the database. If an identical SID field is found, then the system compares the last transaction date field (005) of the incoming record with the same field in the database record. Incoming records that have a transaction date that is smaller than or equal to the transaction date of the database record will be rejected. If the incoming record has a transaction date that is later than the database record, it will overlay the database record. Note that if the incoming record is deleted, it will overlay and delete the existing record. The status deleted is assigned through record status - that is, position 5 of the leader is *d*. The log file of the load process is written to the log directory in each input stream.

4.1 p_union_13

The load process, p_union_13, can be run with the following parameters; Active library, input stream, flag - overlay records with same transaction date (Y/N). If an input stream is specified then the process will only check the converted directory for that stream, if no input stream is specified, the process checks all incoming directories. Note that the list of allowed input streams will vary from installation to installation.

Appendix A: Fix Doc programs

Program Name	What it does
Fix_doc_sid_cdl	Creates SID fields from 901 fields.
Fix_doc_ucla	1. Create 901 from existing 901 Incoming: 901 $$a V $$b AAA0000 $$c LAD Outgoing: 901 $$a LAD $$b AAA0000 drop the original $$a and move the $$c to $$a. 2. Change 958 to 007 input: 958 1 $$a mr baaadm artnnac199712 $$T007 output 007 mr baaadm artnnac199712 So - the contents of subfield T determines the field that $$a will be stored in. The 007 doesn't have a subfield at all 3. If 901 $$b first two digits=04 then change 008 pos. 33 to blank 4. Construct 852 field - Incoming records will have 920-963

Appendix B: Check doc programs

Program Name	What it does	Who should use it
Check_doc_ucla	Checks that the 901 field has a $$c LAD – error message 5009	UCLA