Generic Fix Operations

Version 18+

ExLibris The bridge to knowledge

CONFIDENTIAL INFORMATION

DISCLAIMER

TRADEMARKS

Web address: http://www.exlibrisgroup.com

What is a Generic Fix?

Generic Fix is a home made fix procedures which are often created by libraries when a standard “ready made” fix procedure does not fit the specific needs of the library

When can we use a Generic Fix?

The fix procedure may “occur” at a variety of times, depending on where and how it is added to the tab_fix table
The fix may occur:

1. while in ALEPH sequential file on server
2. during upload
3. While in editing mode of cataloging module
4. When saved to server in cataloging module
5. When viewed in web
6. When copied via Z39.50 in cataloging module

Location of home made fixes

1. The home made fix procedures are created by the library in the $data_tab/import directory of the library.
2. They may be created in any library to work on records of that library (BIB, AUT, etc)
3. After the fix procedure is created it is added as the “program arguments” (col. 3) to $data_tab/tab_fix with the “fix_doc_do_file_08” program name (col. 2), with the desired routine name (col. 1).

Structure of the fix procedure file

The file which defines the home made fix procedure is made of nine columns:
COL 1. Iteration. The operations are performed in order of Iteration.
COL 2. Field tag
COL 3. Format
COL 4. First position filter
COL 5. Position range start
COL 6. Position range end
COL 7. Occurrence filter
COL 8. Operation code. A list of valid Operation codes is in the header to the table and explained later on this document
COL 9. Operation parameters

Iteration

The operations are done in order of Iteration, then in the order within every iteration. In other words, operations in Iteration 1 are processed before those in Iteration 2. Within Iteration 1, the operations are processed in the order listed in the script.

Example of iteration

```
!   2    3   4   5    6     7                  8                           9
!-!!!!!-!!-!-!!!-!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!
1 001                          COPY-SYSTEM-NUMBER             035   ,L,a
1 035##   1st change           REPLACE-STRING                 ocm,(OCoLC)
1 LDR                          ADD-FIELD                      OWN   ,L,$$aGLOBAL
2 852##   2nd change           CHANGE-FIELD                   949
3 949##                        DELETE-SUBFIELD                c
3 949##   3rd change           REPLACE-STRING                 $$i,^
3 949##                        CHANGE-SUBFIELD                h c
3 949##                        CHANGE-SUBFIELD                b h
4 949##   4th change           REPLACE-STRING                 \101,\102
4 998##                        DELETE-FIELD
```

Generic Fix Operations

ADD-FIELD
Adds a field to the record with the coding and contents specified. An occurrence of the new field is added to the record for each existing occurrence of the tag specified in col.2. To add a field once to all records in a file, use a mandatory non-repeatable tag (for example, LDR) in col.2.

Parameters: (comma-separated)
1. field code (5 characters: tag plus indicators)
2. field alpha (1 character)
3. field contents (include subfield codes, for example, $$a)

ADD-SUBFIELD
Adds a subfield to the specified field. An occurrence of the subfield is added to all ocurrances of the tag specified in column 2.

Parameters: (comma-separated)

1. subfield code
2. contents

CHANGE-FIELD
Changes the tag of a field.

Parameters: 1. field tag (3 characters)

CHANGE-FIRST-IND
Changes the value of the first indicator of a variable field.

Parameters: (comma- or space-separated)
1. value to match; '#' acts as a wildcard (1 character)
2. value to set indicator to upon a match (1 character)

CHANGE-FIRST-IND-MATCH
Changes the value of the first indicator of a variable field. It is similar to CHANGE-FIRST-IND except that it has a third parameter, which is a string to match upon. If there is such a string in the tag's content, then the indicator is changed as specified.

Parameters: (comma- or space-separated)
1. value to match; '#' acts as a wildcard (1 character)
2. value to set indicator to upon a match (1 character)
3. string to match upon. It can include subfield delimiters ($$) and wildcards (#).

CHANGE-SECOND-IND
Changes the value of the second indicator of a variable field.

Parameters: (comma- or space-separated)
1. value to match; '#' acts as a wildcard (1 character)
2. value to set indicator to upon a match (1 character)

CHANGE-SECOND-IND-MATCH
Changes the value of the second indicator of a variable field. It is similar to CHANGE-FIRST-IND except that it has a third parameter, which is a string to match upon. If there is such a string in the tag's content, then the indicator is changed as specified.

Parameters: (comma- or space-separated)
1. value to match; '#' acts as a wildcard (1 character)
2. value to set indicator to upon a match (1 character)
3. string to match upon. It can include subfield delimiters ($$) and wildcards (#).

CHANGE-SUBFIELD
Changes every occurrence of a subfield code in a variable field to another value.

Parameters: (comma- or space-separated)
1. subfield code to match; there is no wildcard (1 character)
2. value to change subfield code to upon a match (1 character)

CONCATENATE-FIELDS
Concatenates the first occurrence of a tag all occurrences of another given tag.

Parameters (comma-separated):

1. Field code (5 character: tag plus indicators)

3. Subfields to concatenate (list of subfields is a single string without comma delimiters)

COND-LOAD-VAL-POS
Determines whether to continue processing the record or reject it, based on the value of the field position (col.5) in a fixed field.

Parameters: (comma-separated)
Condition type; can be "Y" or "N":

If "Y" and the value of the field position (col.5) is in the list supplied in the second parameter, the record is rejected.
If "N" and the value of the field position is not in the list, the record is also rejected.
In all other cases, the record is accepted. (1 character)

List of values of the field position that determine whether or not to accept the record

COND-LOAD-VAL-FIELD
Determines whether to continue processing the record or reject it, based on the presence or absence of a particular field tag

Parameters:
Condition type; can be "Y" or "N":

COND-LOAD-VAL-MATCH
Determines whether to continue processing the record or reject it, based on the presence or absence of a particular field tag + subfield + content string

Parameters: (comma-separated)
1. Condition type; can be "Y" or "N":
2. Text string in the field that determines whether or not to accept the record

COPY-FIELD
Copies the entire contents of each matching field to another field. Any attempt to copy to the same tag, or to a tag that matches the pattern in col.2 is ignored in order to prevent an infinite loop. (If it is necessary to duplicate a field, tag and all, first COPY-FIELD to a temporary tag, then CHANGE-FIELD to the desired tag.)

Parameters: (comma-separated)
field code of new field (5 characters: tag plus indicators)
field alpha (1 character) (defaults to L)

COPY-SYSTEM-NUMBER
Copies the entire contents of a fixed length control field to new variable field and subfield, optionally adding a prefix.

Parameters: (comma-separated)
field code of new field (5 characters: tag plus indicators)
field alpha (1 character)
new subfield (1 character)

optional prefix to add to the contents of the fixed length control field after they are copied to the new variable field

DELETE-FIELD
Unconditionally deletes a fixed or variable field.

Parameters: *none*

DELETE-FIELD-COND
Deletes a variable field if it contains the specified string. Matching is exact and case-sensitive.

Parameters: (comma-separated)
condition type; can be "Y" or "N":
If "Y" and the match string is present in the field, the field is deleted.
If "N" and the match string is not present in the field, the field is also deleted. In all other cases, the field is retained.
match string

DELETE-FIXED-COND
Deletes a fixed field if the specified position (col.5) or range (col.5-6) matches the pattern given.

Parameters: (comma-separated)
condition type; can be "Y" or "N":
If "Y" and the match string pattern matches the values present in the position range, the field is deleted.
If "N" and the match string pattern does not match the values present in the position range, the field is also deleted. In all other cases, the field is retained.
match string; '#' in any position is interpreted as a wildcard.

DELETE-SUBFIELD
Removes all occurrences of the specified subfield and contents from the variable field. If the last subfield is removed, the entire field is deleted.

Parameters: 1. subfield to remove (1 character)

DELETE-SUBFIELD-DELIMITER
Removes all occurrences of the specified subfield delimiter (for example, , $$a) only. The delimiter is replaced with a single space. The delimiter of the first subfield in the field will not be removed.

Parameters: 1. subfield delimiter to remove (1 character)

EDIT-SUBFIELD-HYPHEN
Inserts a hyphen if it is not already present at the specified position within each occurrence of the specified subfield. An insertion does not take place if the existing contents are not long enough.

Parameters: (comma-separated)

subfield in which to insert the hyphen (1 character)
position within subfield at which to insert the hyphen (3 digits, leading zeroes required)

FIXED-CHANGE-VAL
Changes the value of the specified range of positions (cols.5-6) in a fixed field if the current value matches a pattern.

Parameters: (comma-separated)
pattern; '#' is a wildcard. Note that the pattern must be exactly as long as the specified position range.
replacement values. Must be exactly as long as the specified position range.

FIXED-CHANGE-VAL-RANGE
Replaces every occurrence of a character found anywhere in the specified range of positions (cols.5-6) in a fixed field with another character.

Parameters: (comma-separated)
character to match; '#' is a wildcard
replacement character (use ^ for blank, | for fill character)

FIXED-FIELD-EXTEND
Extends a fixed field if it is at least the specified minimum length but less than the maximum, by appending the specified character to bring it up to the maximum length.

Parameters: (comma-separated)
minimum length of field required for it to be extended (3 digits, leading zeroes required)
length to extend field to (3 digits, leading zeroes required)
character to pad field with (1 character, use ^ for blank, | for fill character)

FIXED-RANGE-OP
Performs the specified operation on a position range (cols.5-6) of a fixed field.

Parameters: 1. field operation. Must be either LOWER or LJ.
LOWER: changes all characters in the range to lowercase
LJ: left-justifies the non-blank values in the range

REPLACE-STRING
Replaces all occurrences of the specified string with another within a variable field. The strings can include subfield codes.

Parameters: (comma-separated)
pattern to match; '#' acts as a positional wildcard; otherwise, matching is exact and case-sensitive.
replacement string; may be the empty string

SORT-FIELDS
Sorts the fields in the record by tag number, based on the usual sort order for the ALEPH library in which this script is being run. Note that this operation is the only

one that does not use tag parameters in col.2. It is recommended to run this as the 9th and last iteration in each script.

Parameters: *none*

STOP-SCRIPT
Stops the script from running. All other operations scheduled after this command are not performed.

Parameters: 1. subfield code and subfield contents for the condition (for example, $$a = *PUB*); can be left blank.