Aleph 24
Configuration Guide

Version 24

CONFIDENTIAL INFORMATION

The information herein is the property of Ex Libris Ltd. or its affiliates and any misuse or abuse will result in economic loss. DO NOT COPY UNLESS YOU HAVE BEEN GIVEN SPECIFIC WRITTEN AUTHORIZATION FROM EX LIBRIS LTD.

This document is provided for limited and restricted purposes in accordance with a binding contract with Ex Libris Ltd. or an affiliate. The information herein includes trade secrets and is confidential.

DISCLAIMER

The information in this document will be subject to periodic change and updating. Please confirm that you have the most current documentation. There are no warranties of any kind, express or implied, provided in this documentation, other than those expressly agreed upon in the applicable Ex Libris contract. This information is provided AS IS. Unless otherwise agreed, Ex Libris shall not be liable for any damages for use of this document, including, without limitation, consequential, punitive, indirect or direct damages.

Any references in this document to third-party material (including third-party Web sites) are provided for convenience only and do not in any manner serve as an endorsement of that third-party material or those Web sites. The third-party materials are not part of the materials for this Ex Libris product and Ex Libris has no liability for such materials.

TRADEMARKS

"Ex Libris," the Ex Libris bridge , Primo, Aleph, Alephino, Voyager, SFX, MetaLib, Verde, DigiTool, Preservation, URM, Voyager, ENCompass, Endeavor eZConnect, WebVoyage, Citation Server, LinkFinder and LinkFinder Plus, and other marks are trademarks or registered trademarks of Ex Libris Ltd. or its affiliates.

The absence of a name or logo in this list does not constitute a waiver of any and all intellectual property rights that Ex Libris Ltd. or its affiliates have established in any of its products, features, or service names or logos.

Trademarks of various third-party products, which may include the following, are referenced in this documentation. Ex Libris does not claim any rights in these trademarks. Use of these marks does not imply endorsement by Ex Libris of these third-party products, or endorsement by these third parties of Ex Libris products.

Oracle is a registered trademark of Oracle Corporation.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Ltd.

Microsoft, the Microsoft logo, MS, MS-DOS, Microsoft PowerPoint, Visual Basic, Visual C++, Win32,

Microsoft Windows, the Windows logo, Microsoft Notepad, Microsoft Windows Explorer, Microsoft Internet Explorer, and Windows NT are registered trademarks and ActiveX is a trademark of the Microsoft Corporation in the United States and/or other countries.

Unicode and the Unicode logo are registered trademarks of Unicode, Inc.

Google is a registered trademark of Google, Inc.

Copyright Ex Libris Limited, 2022. All rights reserved.

Document released July 3, 2022

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

Table of Contents

General **18**

5426_to_unicode2 ..... **18**

853_chrono **19**

853_numbering **19**

aleph_start **20**

aleph_startup **26**

allowed_languages **26**

bib_format **27**

cash_09_translate **28**

check_circ **28**

check_circ_override **29**

check_doc **30**

check_doc (error messages) **31**

check_doc.<lng> **32**

check_doc_doc **33**

check_doc_field_006 ..... **34**

check_doc_field_007 ..... **36**

check_doc_field_008 ..... **37**

check_doc_field_ldr **38**

check_doc_line **39**

check_doc_line_contents 41

check_doc_mandatory 42

check_doc_new_acc 44

check_doc_new_acc_aut 44

check_doc_tag_text 45

circ_events_users 46

circ_logger 46

check_doc_unique_index 47

codes.<lng> 47

config.xml 49

data_supplier_priorities 52

drm_routines 52

edi_in_attr 53

edi_out_attr 54

edit_doc.<lng> 56

edit_doc_777 57

edit_doc_999.<lng> 58

edit_doc_999_aut_<aut_lib>.<lng> 63

edit_field.<lng> 65

edit_paragraph.<lng> 67

expand_doc_bib_z30 68

expand_doc_bib_z403 70

fix_doc.<lng> **71**

form_description.<lng> **73**

form_print_method **73**

form_sms **74**

form_sub_library_address **75**

formats.<lng> **76**

generic_fix **77**

ill_bor_charge **78**

ill_format_type **79**

ill_formats_map.<lng> **81**

ill_lend_charge **82**

ill_unit_group.<lng> **83**

job_list **83**

job_list.conf **84**

ldap.conf **85**

library_relation **86**

license **89**

marc_country_codes **89**

marc_exp.dat **90**

marc_language_codes **91**

marc8_xxx_to_unicode **91**

no_utf_mac **92**

no_utf_pc 92

oaiconf.dtd 93

oaiconf.xml 93

oaipubconf.xml 94

oclc_server_defaults 96

order_sta_update 97

path_convert 98

pc_server_defaults 98

pc_tab_acq_fast_cat.<lng> 107

pc_tab_cat_conv 108

pc_tab_circ_fast_cat.<lng> 109

pc_tab_col.<lng> 110

pc_tab_crs_fast_cat.<lng> 113

pc_tab_exp_field.<lng> 114

pc_tab_exp_field_extended.<lng> 116

pc_tab_fast_circ 117

pc_tab_find.<lng> 118

pc_tab_find_course.<lng> 118

pc_tab_scan.<lng> 119

pc_tab_sear.<lng> 120

pc_tab_short.<lng> 121

pc_tree_c1501 123

permission.dat **124**

pqf.properties **125**

prof_alephe **130**

remote_catalog.dat **130**

rep_tab_short.<lng> **130**

scancode.dat **131**

server_ip_allowed **134**

tab_85x_import **135**

tab_96_def **135**

tab_98_def **136**

tab_99_def **137**

tab_abbrev **138**

tab_acq_index **139**

tab_alephadm.<lng> **140**

tab_alert **141**

tab_attr_sub_library **142**

tab_aut **145**

tab_author_num.conf **146**

tab_base.<lng> **149**

tab_base_count **151**

tab_binding **153**

tab_bib_aut_match **155**

tab_block_circ **155**

tab_booking **156**

tab_bor_address **157**

tab_bor_id.<lng> **159**

tab_buf_z403 ..... **160**

tab_cat_hidden_fields **162**

tab_ccl_boolean_operators **162**

tab_character_conversion_line **163**

tab_check_circ **166**

tab_checksum **170**

tab_circ_log.<lng> **172**

tab_code_prefix **173**

tab_days **174**

tab_dbs.<lng> **174**

tab_delinq.<lng> **175**

tab_delivery_locations.<lng> **176**

tab_doc_publish **177**

tab_events.<lng> **178**

tab_exp_own.<lng> **180**

tab_expand **181**

tab_expand_duplicate_field **184**

tab_expand_external **185**

tab_expand_extract **186**

tab_expand_join **187**

tab_expand_join_simple **189**

tab_expand_sort **190**

tab_expand_split **190**

tab_export **191**

tab_export_03 ..... **192**

tab_fast_patron_register **193**

tab_filing **194**

tab_filing_call_no **202**

tab_fix **203**

tab_fix_local_notes.conf **206**

tab_fix_new **206**

tab_fix_notes **207**

tab_fix_z103 ..... **208**

tab_doc_trans **209**

tab_font_publish **210**

tab_hld_stmt **211**

tab_hol_item_create **212**

tab_hol_item_map **215**

tab_hol_mapping **216**

tab_hold_request **217**

tab_hold_request_form 227

tab_photo_request_form 229

tab_ill_bib_info 230

tab_ill_bib_key 231

tab_ill_change_status_out 231

tab_ill_change_status_in 232

tab_ill_held_media_map.<lng> 233

tab_ill_partner_conf 233

tab_ill_preferred_media 233

tab_ill_status_borrow 234

tab_ill_status_lend 235

tab_ill_tree_bor_msg 237

tab_ill_tree_lend_msg 237

tab_io_remote 238

tab_item_history.<lng> 238

tab_item_list_order 240

tab_itm_mapping 240

tab_label.<lng> 241

tab_label_parse 242

tab_late_return 245

tab_library_group 246

tab_loader 246

tab_loader_def **249**

tab_locate **252**

tab_location_name.<lng> **254**

tab_map_privileges **254**

tab_mapping **255**

tab_mapping_mng500 ..... **256**

tab_match **257**

tab_match_acc **258**

tab_match_script **259**

tab_md_ns_info **261**

tab_merge **261**

tab_merge_adv_overlay **263**

tab_merge_overlay **265**

tab_mime_type **267**

tab_month **267**

tab_move_record **268**

tab_ncip_record_id **271**

tab_oclc **272**

tab_own **273**

tab_fix_notes **273**

fix_z103_sort_by_my_own in the table tab_fix_notes **276**

tab_patron_log.<lng > **277**

tab_photo_request **278**

tab_pinyin **279**

tab_preferred **280**

tab_publish **281**

tab_rfid **282**

tab_rfid_translate **283**

tab_roster **284**

tab_service **284**

tab_sfx **288**

tab_sfx_denied **290**

tab_sip2_alert **290**

tab_sip2_conf **291**

tab_sip2_sort_bin **293**

tab_sip2_translate **293**

tab_sort **294**

tab_sub_lib_group.<lng> **295**

tab_sub_lib_sort **296**

tab_sub_library.<lng> **297**

tab_sub_library_address.<lng> **300**

tab_sublibrary_grouping **301**

tab_subfield_punctuation **302**

tab_suf_bor_stat **303**

tab_suf_default **303**

tab_suf_lib **304**

tab_suf_loc_borid **305**

tab_tag_text **305**

tab_type_config.<lng> **306**

tab_vat_percent.<lng> **309**

tab_vendor_sub_lib **309**

tab_version **311**

tab_weights **311**

tab_word_breaking **314**

tab_www_item_desc.<lng> **318**

tab_xyz **319**

tab_yankee_def **320**

tab_z30_sort **320**

tab_z31_sort **328**

tab_z36_sort **330**

tab_z38_sort **331**

tab_z0101 ..... **333**

tab_z0101_text **334**

tab_z103 ..... **336**

tab_z105 ..... **340**

tab_z105_filter **342**

tab_z105_z30 342

tab_z121 342

tab_z311 343

tab_z403_directory.<lng> 344

tab_z403_extension.<lng> 344

tab_z403_info.<lng> 345

tab_z701.<lng> 346

tab00.<lng> 347

tab01.<lng> 349

tab02 351

tab04 352

tab05.<lng> 353

tab06 354

tab07 355

tab11_acc 356

tab11_aut 357

tab11_ind 359

tab11_word 360

tab12 363

tab14 364

tab15.<lng> 365

tab_patron_log.<lng > 365

tab16 ..... **369**

tab17 ..... **375**

tab18.<lng> **376**

tab20 ..... **381**

tab22 ..... **384**

tab22_arc **387**

tab24 ..... **390**

tab25.<lng> **390**

tab27 ..... **391**

tab30 ..... **393**

tab31 ..... **394**

tab32 ..... **396**

tab33.<lng> **397**

tab34 ..... **398**

tab35 ..... **400**

tab36 ..... **401**

tab37 ..... **402**

tab37_booking_delivery **405**

tab37_booking_pickup **406**

tab37_campus_filter **407**

tab37_campus_preferred **408**

tab38 ..... **409**

tab39 ..... **410**

tab40.<lng> **411**

tab41 ..... **412**

Tab_req_preferred **413**

Tab_req_preferred_loc **415**

tab42 ..... **415**

tab43 ..... **417**

tab44 ..... **419**

tab45.<lng> **420**

tab46.<lng> **422**

tab47.<lng> **423**

tab48.<lng> **424**

tab50 ..... **426**

tab100 ..... **427**

tag_text.dat **465**

tagonnew.dat **466**

unicode_to_filing_01 ..... **467**

unicode_to_word_gen **469**

union_global_param **470**

union_preferred **471**

user_function.<lng> **472**

www_f_func **473**

www_f_heading **474**

www_f_sort_heading.<lng> **475**

www_f_tab_short_999 ..... **476**

www_server.conf **477**

www_tab_myshelf_short.<lng> **484**

www_tab_self_check **485**

www_tab_short.<lng> **486**

www_x_func **489**

xsl_creator.<lng> **490**

Z111_index table **490**

Index **493**

General

This documentation describes the tables for the management and configuration of the **ALEPH** libraries. The tables in this guide are listed alphabetically. The index at the end of the guide is an aid for accessing a table by its subject or topic.

ALEPH Library Structure
An **ALEPH** installation consists of several libraries. The central library is the ADMinistrative library, which manages all administrative aspects of the library including items control, acquisitions, serials, ILL, and circulation. The ADM library is linked to several types of records, each of which is held in a separate library or databases:

Type of Record	**Format of Record**
BIBliographic records	Any MARC format
HOLdings records	MARC 21 format
AUThority records	MARC 21/UNIMARC formats
ILL records for the bibliographic details of the ILL request	

Note: One installation can have more than one ADM library and/or BIB/HOL/AUT/ILL databases.

ALEPH has the following convention for naming libraries of one installation. A numeric suffix is added to the name of the library. For example, for the MARC 21 demonstration library:

Name	Explanation
USM01	BIBliographic records library
USM10	AUThority records library
USM40	ILL records library – BIB and ADM records
USM30	Course Reading library
USM50	ADMinistrative library
USM60	HOLdings records library

5426_to_unicode2

Location of the table: alephe/unicode directory

Purpose of the table: Translates ISO 5426 to Unicode

This table translates ISO 5426 to Unicode. ISO5426 character set is the European standard for bibliographic data, and although it is different from ANSEL (NISO Z39.47) it serves the same function. It is used by the French equivalent of "Books in Print".

853_chrono

Location of the table: alephe/tab directory

Purpose of the table: Table for names of months and seasons

This table is used for supplying chronology text for months and seasons for the description field of Serials items records, when 85X/85XX fields are used for building the description.

The months should be listed from January through December, and the seasons should be listed from Spring through Winter.

The language in the ADM 008 field is matched with the language code in this table.

Structure of table:

Col. 1 language code – 3 characters, lowercase.

Col. 2 M – month
S – season

Col. 3 Text

Example of the table:

```
!1  2 3
!!!-!-!!!!!!!!!!!!!!!!!!!
eng M Jan.
eng M Feb.
eng M Mar.
…
eng S Spring
eng S Summer
eng S Autumn
eng S Winter

fre M janv.
fre M févr.
fre M mars
fre M avril
```

853_numbering

Location of the table: alephe/tab directory

Purpose of the table: Abbreviations to be used with numbering

This table defines the ordinal suffix to add to enumeration levels a-h. The ordinal suffix will be used when the 853 field has a + sign before the enumeration caption. In this case, the enumeration and its ordinal suffix are placed before the caption.

For example:

$$b+^quarter will create

1st quarter
2nd quarter, and so on.

Use a ? sign in the table to denote a wildcard (for example, ?1 for 21st, 31st, 41st, and so on. Note that the program uses the first row whose mask matches the number -- for the above example "11" (that is, 11th) should be listed before ?1.

Language is taken from pos. 35-37 in the 008 of the ADM record.

Structure of table:

Col. 1 language code – 3 characters, lowercase.
Col. 2 number
Col. 3 ordinal suffix

Example of the table:

```
!1      2       3
!!!-!!!!!!!!!-!!!!!
eng 11          th
eng 12          th
eng 13          th
eng ?1          st
eng ?2          nd
eng ?3          rd
eng ?           th

fre 1           er
fre ?           e
…
```

aleph_start

Location of the table: alephe directory

Purpose of the table: Node environment definitions

Related table(s):

1) If setenv ADJACENCY is set to 0, the variable setenv set_prox_limit should be defined in www_server.conf.
2) setenv due_date_format works in combination with switch RECALL-METHOD in tab100.
3) setenv fine_factor_fix is set in tab16, col. 11.
4) setenv REQUEST_STARTUP_LIBS links up to tab37 or tab38.

General note:
Note that there are some definitions that offer more than one “setenv” option. The non-active option does not have to be commented out – the system will always use the **last** line defined for that option.

In order for changes to aleph_start to take effect, you must exit aleph, re-log-in, and then restart the daemons and servers. (For batch jobs submitted through Services this means that the batch_que must be restarted.) Running aleph_shutdown and aleph_startup restarts all daemons and servers.

Note that the number of aleph libraries defined under the ALEPH_LIBS environment variable is limited to 500.

Following is an explanation of some of the parameters in aleph_start. The parameters are listed alphabetically ignoring 'setenv':

setenv ACC_AUT_STARTUP_LIBS "usm01 usm50"
For use with AUThority library/ies. It links the bibliographic database headings to authority records. This line parallels activation of UTIL E/8 "Start update BIB ACC from AUT".

setenv ADJACENCY
ADJACENCY_TYPE **0 -** does not build adjacent words
ADJACENCY_TYPE **2 -** builds adjacent words
ADJACENCY_TYPE **3 -** builds adjacent Latin words only

If set to 0, the variable setenv set_prox_limit should be defined in www_server.conf.

Note that setting "2" improves performance for adjacency searching, but approximately 4x storage space per document is required.

Use Type 3 when using CJK indexing (2gram or Morpheme word breaking routines).

setenv alephe_synch_root $alephe_synch_dev/alephe
Used for update synchronization of alephe tables and error files

setenv aleph_utf /tmp
Defines the path to the utf files.

setenv bor_id_concat_char
This parameter is linked to the Web OPAC and patron retrieval by concatenation of ID and verification.

It is possible to retrieve a patron by using the ID (as Z308-KEY-DATA) and the password/verification (Z308-VERIFICATION).

The data is loaded to the system such that Z308-KEY-DATA comprises of
a patron name (for example, "DANNY"),
a concatenation character (for example, a hyphen "-")
and the verification (for example, "X"):
Z308-KEY-DATA: DANNY-X
Z308-VERIFICATION : X

The concatenation character is defined in this variable.

The purpose of this development is to make it possible to use a single ID (but with different verifications) for different tasks in the Web OPAC.

Column 8 in tab_bor_id.<lng> should be set to "N" for codes (Z308-KEY-TYPE) used for this purpose, to prevent patrons from changing their verification.

Note that if the variable is not set, or it is set to spaces, then no concatenation of ID and verification will be made in the retrieval process.

setenv correct_852_subfields "2hijklm"
The link between the item and the holdings record is created by entering the holdings record system number in the item record. When the link is created the system will override location (sublibrary, collection and call number and call number type) information in the item using the location information from the holdings record.

The following information will be overridden:
Item sublibrary by $$b of 852
Item collection by $$c of 852
Item call number type by 1st indicator of 852
Item call number by subfields from 852 as defined in the above variable.
Note too, that the location field in expand_doc_bib_psts and expand_doc_bib_locXX are created in accordance with the definitions in correct_852_subfield .

setenv date_style_s 24
setenv date_style_f 16
setenv time_style_s 04
setenv time_style_f 04
For a full explanation of the date and time styles refer to pc_server_defaults.

setenv due_date_format 1
This setenv controls what displays when there is just a single "Due date" column. It works in combination with switch RECALL-METHOD in tab100.

- **1 -** Display a single "effective due date", based on RECALL-METHOD in tab100.
- **2 -** Display the recall-due-date, if there is one (even if it is later); otherwise, display the regular due date.
- **3 -** Always display only the z36_due_date; (for recall methods 2 and 3 this would mean that the recall-due-date will never display).*
- **4 -** Display both: the regular due date and (if there is one) the recall-due-date.

setenv fine_factor_fix 000
It is possible to define a multiplication factor for the fine rate set in tab16, column 11. This is useful for countries whose currencies are written in the thousands (for example,, 10,000 lira).

setenv fine_rounding Y:

Round fine up or down to get rid of the decimal points. Rounds up when the sum after the decimal point is more than 0.50 and rounds down when the sum is lower than 0.50.

setenv ill_total_limit_period 0814
Defines the option to calculate the total ILL limit of a patron according to a period of time. The variable has format MMDD. The calculation is done from the current date to this variable date (the assumption is that the variable date is earlier than the current date).
All ILL requests that are created between these dates are counted and compared to the total patron's limit (Z303-ILL-TOTAL-LIMIT).
In order to ignore this calculation, either delete the variable or enter 0000.

setenv include_all_aut
Y= all accesses which connected to authority record will be displayed in the browse list of Web OPAC. Even if there are no records connected to this access.
This applies only to codes using Z01 and not to those which are using Z0102. Z0102 continues to use the XYZ mechanism.

setenv local_currency USD
Definitions with regard to the library's local currency. If defined here:
The currency table is never consulted. The ratio for the defined currency is always 1.00
This currency will be the default currency for allocation (if not currency filled in the order form.)
Will always be the currency for transfer transactions.
There is an error message when trying to add/replace/delete a ratio for the currency.

In order to change the local currency as defined in this parameter:
In the GUI-ADMIN module, define currency XXX, and set its ratio to 1.000
Define XXX as the local currency in ./alephe/aleph_start:
setenv local_currency USD
Run the command: source alephe_root/aleph_start
Restart PC server (UTIL W/3/3)

setenv MESSAGE_STARTUP_LIBS " $z105_library"
For messaging between libraries for mutual update (for example, update of Authority record triggers re-linking BIB heading). This line parallels activation of UTIL E/11 "Start messaging update".

setenv new_marcive_loader Y

setenv NCIP_SERVER_STARTUP Y
This controls the startup of the NCIP server

setenv new_oclc_server Y
A new generic loader has been developed. In order to use it the above two parameters must be defined and set to Y.

The loader can work in a multi-ADM environment. The ADM library will be determined according to the Holding Code (in the 049 field).

setenv OCLC_SERVER_STARTUP Y
This controls the startup of the OCLC server

setenv pw_library USR00
setenv usr_library USR00
setenv z105_library USR00

Note that the library name in those definitions must be in uppercase.

Every installation has a default "library" USR00. The purpose of this library is as follows:
for non-ILS applications (such as Union Catalog, DigiTool) that nevertheless require USERS (patrons and librarians), there is no need for a complete ADM library; there is only need for records related to patrons (ID, address, profile, SDI) and passwords.
Z105 messages on the same server are written in and handled by the USR00 library. ue_11 is activated only in this library, which sends all update triggers to the correct destination.
for applications that have multiple ADM libraries -- passwords are always shared across libraries, and USR00 can be used for storing them (rather than choosing one of the ADM libraries).

setenv REQUEST_STARTUP_LIBS "usm50"
Links up to tab37 or tab38 for managing retrieval and printing of hold or photocopy requests. This line parallels activation of UTIL E/6 “Start request handling” for the library.

setenv server_sms "10.1.235.14:1894"
Defines the host IP and port where the SMS Proxy is located.

setenv sfx_admin_base_url http://demo.exlibrisgroup.com:9003/sfxadmin
Relates to a service "service-sfx-admin". This service acts very much like the regular "service-sfx" only will take the user to the admin module, instead of the sfx menu.

The service accepts the following parameters (which can be used in the PC GUI file "ext_srv.ini" under the ACQ or CAT modules):

DOC_LIBRARY
DOC_NUMBER
SFX_CATALOG
USER_LIBRARY
FILE_NAME

setenv sfx_base_url htt://demo.exlibrisgroup.com:9003/demo
setenv sfx_base_url_usm50 http://demo.exlibrisgroup.com:9003/demo
The SFX base URL can be ADM-library sensitive. When constructing the openURL link to SFX, the hierarchy where to find the sfx-base-url is as follows:

Use a web cookie under the name "BASE-URL". The value of the cookie should contain the host-name and the sfx instance to use in the format of http://host-name:port/instance
Use an environment variable "sfx_base_url_<ADM_LIBRARY>" (as in the above example)
Use a global variable in www_server.conf named "sfx_base_url" (as in the above example)

For example,
if no cookie is present and the definitions in www_server.conf contain:
setenv sfx_base_url http://demo.exlibrisgroup.com:9003/default
setenv sfx_base_url_usm50 http://demo.exlibrisgroup.com:9003/usm50
setenv sfx_base_url_usm51 http://demo.exlibrisgroup.com:9003/usm51
And if the profile of the patron (or the IP profile or the ALEPH profile) has USM50 as the bor-library, the base-url will always be
http://demo.exlibrisgroup.com:9003/usm50.
A patron from library USM52 will have the default base of http://demo.exlibrisgroup.com:9003/default.

setenv sfx_openurl_version ver-0.1
Support for dual openurl syntax. In aleph_start you can define the openurl version you have on your site. The version parameter default is "ver-0.1". It is used as an extension to the sfx-record Web page. In this manner, all SFX links will use the "sfx-record-ver-0.1" web page. If the file does not exist, the regular "sfx-record" is used.

setenv SIP2_STARTUP_LIBS "usm50"
Defines the library that will start the Self Service Server.

setenv SRU_SERVER_STARTUP Y
This controls the startup of the SRU server.

setenv unique_order_number_2 Y
setenv sw_z103_filter_suppressed Y
Variables for enabling check on index of z68_order_number_2

Setenv_user_list_aleph_local
This variable defines whether "ALEPH" in Z305 will be taken into account when showing the Local user list.

setenv vendor_library USM50
Applicable only for multi-ADMs in a shared vendor model.

setenv WWW_HOST ram19

This definition is needed by the www_server in cases where the hostname is not the external name. If this variable is not defined then the hostname is used as the back link in all Web pages.

setenv Z39_SERVER_STARTUP Y
setenv Z39_GATE_STARTUP Y
This controls the startup of the Z39 gate and server.

setenv JAPANESE_ENABLED FALSE
The variable can be set with the values FALSE (default) or TRUE.
Setting the variable with TRUE enables special Japanese features such as defining the Patron's Salutation field (Patron's Katakana name) and Vendor's Contact 5 (vendor's Katakana name) as mandatory fields.

setenv order_unit 2ISSN
setenv letter_type 12

The above order_unit and letter_type variables are used to enable the updating of the 906$$bfield of the BIB record with the date a monograph order is sent to the vendor (this is applied in the ISSN formal notice workflow). They work in conjunction with z105 message: SEND-ORDER – type "j".

For more information, refer to the **Updating the BIB Record with the Date of the ISSN Formal Notice** section of the *Aleph 21 System Librarian's Guide-Acquisitions*.

For more details of other elements of the table, refer to the *Aleph 20.1 System Librarian's Guide*.

aleph_startup

Location of the table: alephe/root directory

Purpose of the table: Startup definitions

For more details, please refer to the *System Librarian's Guide*.

allowed_languages

Location of the table: alephe/tab directory

Purpose of the table: Language code definitions

Related table(s):

1) pc_tab_crs_fast_cat.<lng>
2) pc_tab_exp_field.<lng> (section LANG).

This table lists the language codes recognized by the system. If not listed here, the system will not recognize the language code and will return an error message (for example, when entering data in the vendor language field in the online GUI).

Note that in addition to the definitions listed in the table, the code and text MUST be defined in pc_tab_expand as well (section LANG).

Structure of the table:

Cols. 1-20

Language codes

Example of the table:

```
!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-
ACE ACH ADA AFA AFH AFR AJM AKA AKK ALB ALE ALG AMH ANG APA ARA ARC ARM
ART ARW ASM ATH AVA AVE AWA AYM AZE
BAD BAI BAK BAL BAM BAN BAQ BAS BAT BEJ BEL BEM BEN BER BHO BIK BIN BLA
BUG BUL BUR
```

bib_format

Location of the table: tab directory of the **ADM** library

Purpose of the table: Formats for bibliographic display in templates

Related table(s):

1) edit_paragraph.<lng>

This table defines the creation of the bibliographic format for administrative printouts. The system turns to the edit_paragraph.<lng> table for the display of the bibliographic information in the administrative printouts.

Note that a default line should be defined:

```
default                          300
```

Structure of the table:

col. 1	template name
col. 2	document format, as defined in the BIB library's edit_paragraph.<lng>.

Example of the table:

```
!      1                          2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!
default                          300
acq-additional-claim             301
acq-arrival-slip                 301
acq-claim-m-report               300
acq-claim-o-report               300
…
bor-list                         210
bor-list-renew-all               210
copy-info                        012
hold-delete-letter               014
hold-delete-letter-s             014
hold-request-letter-01           014
```

cash_09_translate

Location of the table: tab directory of the **ADM** library

Purpose of the table: Sublibrary translation to Campuses

Related table(s):

1) tab_sub_library.<lng>

This is a table that translates sublibraries to a department code, for report purposes. The report used is cash-09 report.

Structure of the table:

Col. 1 Sublibrary;
Col. 2 Campus Code

Example of the table :

```
! 1    2
!!!!!-!!!!
UELEC 3020
```

check_circ

Location of the table: aleph/error_<lng> directory

Purpose of the table: Error messages linked to circ tables
Related Tables: Error messages linked to:

2) tab_check_circ
3) check_circ_override
4) tab_block_circ

Note: In version 17.0 the directory error_<lng> was moved from alephe/error_<lng> to aleph/error_<lng>.

Error messages linked to tab_check_circ, check_circ_override and tab_block_circ

Structure of the table:

Col. 1 Error code
Col. 2 ALPHA
Col. 3 Error message

Example:

```
!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

0098 L Successful renew (override)
0099 L Successful renew

0050 L Unable to extend loan period.

!*check_circ_1_a

0101 L Delinquency - $1 $2
0102 L Delinquency - $1 $2
0103 L Delinquency - $1 $2
```

check_circ_override

Location of the table: tab directory of the **ADM** library

Purpose of the table: Minimum user level for override

Related table(s):

1) …/aleph/error_<lng>/check_circ >

This table defines the minimum password level that is required in order to be able to override a circulation transaction that has been trapped by a circulation check procedure.
A minimum password level is defined for each check procedure. The check procedure is identified by using the number of the error message that displays when the check procedure traps the transaction.

Error codes are system-defined, and can be found in aleph/error_<lng>/check_circ.

Structure of the table:

col. 1 Error code from aleph/error_<lng> check_circ.
col. 2 Minimum patron level for override;
col. 3 Description of the error (non-functional)

Example of the table:

```
!!!!-!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
!*check_circ_1_a
0101 00 check_circ_1_a: Delinquency - $1 $2
0102 00 check_circ_1_a: Delinquency - $1 $2
!*check_circ_1_b
0111 00 check_circ_1_b: Local Delinquency - $1 $2
0112 00 check_circ_1_b: Local Delinquency - $1 $2
0113 00 check_circ_1_b: Local Delinquency - $1 $2
0115 00 check_circ_1_b: Status not valid for this sublibrary
```

check_doc

Location of the table: tab directory of the library

Purpose of the table: Check routines

The check_doc table is used to define the check programs that are used in the system.

Up to 100 program names may be assigned to library's check_doc function.

Column 1 contains the check type that defines when the check program is performed.

Following are the reserved check types:
CATALOG-INSERT: performed when the cataloging record is saved, updated or when the Check Record option is selected from the Cataloging module.
CATALOG-DELETE: performed when the Delete Record from Server option is selected from the Cataloging module.
BATCH-DELETE: performed when the Delete Bibliographic Records (p-manage-33) batch process is run.
NAV-MAP-DELETE: performed when the Total Delete option is selected from the Record Manager of the Cataloging module to delete a bibliographic record together with all associated records.
P-MANAGE-11: performed when the Check Database Tables (p-manage-11) batch process is run
Z39-INSERT: performed when a record is inserted via Z39.50 ES Update.
Z39-REPLACE: performed when a record is replaced via Z39.50 ES Update
Z39-DELETE: performed when a record is deleted via Z39.50 ES Update

Column 2 contains the check program(s) that should be performed for the specific check type defined in column 1.

Column 3 - Program arguments. Certain check_doc programs require additional information such as table names. These additional parameters are defined in column 3. The documentation for each check_doc program indicates whether it takes parameters, and if so, how they should be formatted. Note that if a check_doc program does not use parameters, it will ignore the contents of column 3.

Structure of the table:

col. 1 Check type
col. 2 Check program
col. 3 Program arguments

Example of the table:

```
!        1                          2                                3
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!…!!!!!>
CATALOG-INSERT       check_doc_new_acc
CATALOG-INSERT       check_doc_new_acc_aut
CATALOG-INSERT       check_doc_unique_index
CATALOG-INSERT       check_doc_tag_text
CATALOG-INSERT       check_doc_line
CATALOG-INSERT       check_doc_line_contents
CATALOG-INSERT       check_doc_lkr
CATALOG-INSERT       check_doc_doc
```

Note:
Some checks are not related to the document itself but to another document.
For example check_doc_unique_index. The ***message*** window has a button to be able to retrieve the related record.

This feature is implemented on the following routines:

check_doc_unique_index
check_doc_delete_lkr
check_doc_locate
check_doc_aut_duplicate (AUT library)

Note that as a result the logic of check_doc_locate has changed slightly, and will now show up to 5 related records.

check_doc (error messages)

Location of the table: aleph/error_<lng>

Purpose of the table: Validation messages (system-driven) for check_doc programs

Note: In version 17.01, the directory error_<lng> was moved from alephe/error_<lng> to aleph/error_<lng>.

The error messages defined in this table are system-driven and are between the range of 0001-4999.

Structure of the table:

Col. 1 Error message
Col. 2 ALPHA
Col. 3 text

Example of the table:

```
0002 L Document has $3 item(s) attached to ADM record $1 in library $2
0003 L ADM record has $1 item(s) attached.
0004 L Document has $3 order(s) attached to ADM record $1 in library $2
0005 L ADM record has $1 order(s) attached.
0006 L Document has $1 subscription(s) attached to ADM record $3 in library $2
0007 L ADM record has $1 subscription(s) attached.
```

check_doc.<lng>

Location of the table: tab directory of the library

Purpose of the table: Validation messages (table driven) for check_doc programs

Related table(s):

2) Programs are defined in check_doc
3) Error messages relating to:
    a. check_doc_doc
    b. check_doc_mandatory

This table provides for validation messages for the check doc programs. The error messages defined in this table are table dependent and should be between the range of 5000-9999 only – for example, checks in program check_doc_doc and check_doc_mandatory.

In the following example (in check_doc_doc) the message number defined for a particular check is **6003.** This matches up with the message line **6003** in check_doc.<lng:

check_doc_doc

```
D  BK 6003 2450#   Y 1####   N
```

check_doc.<lng>

```
6003 L Use 245 1# when there is a main entry (1XX)
```

Structure of the table:

Col. 1 Error message
Col. 2 ALPHA
Col. 3 text

Example of the table:

```
! 1  2              3
!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
5001 L A record cannot have more than 1 main entry (1XX).
5002 L Required 245 field is either missing or duplicated.
5003 L Required 260 field is either missing or duplicated.
5007 L Required 310 field is either missing or duplicated.
5008 L Required 008 field is either missing or duplicated.
5120 L Required 256 field is either missing or duplicated.
5121 L Tag 538 is required for Computer Files.
```

check_doc_doc

Location of the table: tab directory of the library

Purpose of the table: Checks between tags

Related table(s):

1) Col. 3 – error number as listed in check_doc.<lng>
2) formats.<lng>
3) tab01.<lng>

Subtable OC
Occurrences - giving up to 5 field codes, with implied *OR*.
Structure of the table:

Col.1 OC

Col. 2 Record format as registered in the FMT field in the document record. XX should be entered if check valid for all formats.

Col. 3 error number – between 5000 and 7000 - as listed in the check_doc.<lng> table (located in the tab directory of the library)

Col. 4 Minimum number of occurrences.

00 indicates that the field is not mandatory.

Any number from 01 indicates that the field must be present.

Col. 5 Maximum number of occurrences. The upper limit is "99". It is not possible to define an unlimited number of occurrences.

Cols. 6 -10

Fields for which number of occurrences is being defined. Up to 5 field codes with OR implied.

Examples:
For BK format, a record cannot have more than one main entry:

OC BK 1001 00 01 100## 110## 130##

For BK format, a record must have a title statement and the field cannot be repeated:

OC BK 1002 01 01 245##

Subtable D

Structure of the table:

Col.1 D

Col. 2 Record format as registered in the FMT field in the document record. XX should be entered if check valid for all formats.

Col. 3 error number - 5000 – 8999 - as listed in the check_doc.lng table (located in the tab directory of the library)

Col. 4 field code for first part of condition (for example, 100##). Note that wildcard may be used.

Col. 5 defines whether the check relates to the field being present or not

- Y=present,
- N=not present

Col. 6 field code for second condition of the check. Use # for marking wildcard (for example, 245##). Leave blank of there is no second condition.

Col. 7 defines whether the check relates to the field being present or not

- Y=present,
- N=not present

Example:
If 2450# is present then no main entry (1XX) should be present:

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

Example of the table :

```
!  2   3     4     5   6     7
!--!!-!!!!-!!!!!---!-!!!!!---!
!
OC XX 5001 00 01 100## 110## 111## 130##
OC XX 5002 01 01 245##
OC BK 5003 01 01 260##
OC SE 5007 01 01 310##
OC XX 5008 01 01 008##
OC CF 5120 01 01 256##
OC CF 5121 01 99 538##

D  BK 6003 2450#   Y 1####   N
D  BK 6004 2451#   Y 1####   Y
D  BK 6006 4901#   Y 8#0##   Y
D  XX 6100 240##   Y 130##   N
```

check_doc_field_006

Location of the table: tab directory of the **BIB** library

Purpose of the table: Checks on MARC 21 tag 006

Related table(s):

1) formats.<lng>

Note: If you specify check_doc_line in the check_doc table, these check_doc_field_006 values will be checked. You do **not** need to and **should not** include a separate entry for check_doc_field_006 in the check_doc table.

Structure of the table:

Col.1	Format of document to check:. Use XX for any format
Col. 2	The field position used as a matching point for the character specified in column 3
Col. 3	Match character. The field character at the offset given in column 2 must match this value for the check to apply. A space here means that the check is always done.
Col. 4	Start of position range to check
Col. 5	End of position range to check
Col. 6	Type of check to apply: • 1 = check for valid values • 2 = check for obsolete values • 3 = run another check program
Col. 7	Check values. Value depends on the check type: • 1 = list of values, separated by commas, that are valid for this position range. If the position range is only one character, the commas may be omitted. • 2 = list of values, separated by commas, that may be present but are obsolete. If the position range is only one character, the commas may be omitted. • 3 = name of check program to run

Example of the table:

```
!   2  3  4   5  6  7
!!-!!!-!-!!!-!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
XX        000-017 3 check_fixed_field_length
XX        000     1 acdefgijkmoprst

XX 000 a 001      1 ^abcdefghijklmop|
XX 000 a 002      1 ^abcdefghijklmop|
XX 000 a 003      1 ^abcdefghijklmop|
XX 000 a 004      1 ^abcdefghijklmop|
XX 000 a 001-004 3 check_val_left_just
                  3 check_val_alpha_order
```

check_doc_field_007

Location of the table: tab directory of the **BIB** library

Purpose of the table: Checks on MARC 21 tag 007

Related table(s):

1) formats.<lng>

Note: If you specify check_doc_line in the check_doc table, these check_doc_field_007 values will be checked. You do **not** need to and **should not** include a separate entry for check_doc_field_00 in the check_doc table.

Structure of the table:

Col.1 Format of document to check:. Use XX for any format

Col. 2 The field position used as a matching point for the character specified in column 3

Col. 3 Match character. The field character at the offset given in column 2 must match this value for the check to apply. A space here means that the check is always done.

Col. 4 Start of position range to check

Col. 5 End of position range to check

Col. 6 Type of check to apply:

- 1 = check for valid values
- 2 = check for obsolete values
- 3 = run another check program

Col. 7 Check values. Value depends on the check type:

- 1 = list of values, separated by commas, that are valid for this position range. If the position range is only one character, the commas may be omitted.
- 2 = list of values, separated by commas, that may be present but are obsolete. If the position range is only one character, the commas may be omitted.
- 3 = name of check program to run

Example of the table:

```
!   2  3  4   5  6  7
!!-!!!-!-!!!-!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
XX       000     1 acdfghkmoqrstvz

XX 000 a 000-007 3 check_fixed_field_length
XX 000 a 001     1 dgjkqrsuyz|
                 2 abcefhijmnoptvwx
```

check_doc_field_008

Location of the table: tab directory of the library

Purpose of the table: Checks on MARC 21 tag 008

Related table(s):

1) formats.<lng>

Note: If you specify check_doc_line in the check_doc table, these check_doc_field_008 values will be checked. You do **not** need to and **should not** include a separate entry for check_doc_field_008 in the check_doc table.

Structure of the table:

Col.1 Format of document to check:. Use XX for any format

Col. 2 The field position used as a matching point for the character specified in column 3

Col. 3 Match character. The field character at the offset given in column 2 must match this value for the check to apply. A space here means that the check is always done.

Col. 4 Start of position range to check

Col. 5 End of position range to check

Col. 6 Type of check to apply:

- 1 = check for valid values
- 2 = check for obsolete values
- 3 = run another check program

Col. 7 Check values. Value depends on the check type:

- 1 = list of values, separated by commas, that are valid for this position range. If the position range is only one character, the commas may be omitted.
- 2 = list of values, separated by commas, that may be present but are obsolete. If the position range is only one character, the commas may be omitted.
- 3 = name of check program to run

Example of the table:

```
!   2  3  4   5  6  7
!!-!!!-!-!!!-!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
XX       000-039 3 check_fixed_field_length
XX       006     1 bcdeikmnpqrstu|
XX 006 b 007-014 3 check_val_blank
XX 006 c 007-010 3 check_val_date_4_or_u
XX 006 c 011-014 3 check_val_all_9
XX 006 d 007-010 3 check_val_date_4
XX 006 d 011-014 3 check_val_date_4
XX 006 e 007-014 3 check_val_date_8
XX 006 i 007-010 3 check_val_date_4
XX 006 i 011-014 3 check_val_date_4
XX 006 k 007-010 3 check_val_date_4
```

check_doc_field_ldr

Location of the table: tab directory of the library

Purpose of the table: Checks on the MARC ldr

Related table(s):

1) formats.<lng>

Note: If you specify check_doc_line in the check_doc table, these check_doc_field_ldr values will be checked. You do **not** need to and **should not** include a separate entry for check_doc_field_ldr in the check_doc table.

Structure of the table:

Col.1 Format of document to check:. Use XX for any format

Col. 2 The field position used as a matching point for the character specified in column 3

Col. 3 Match character. The field character at the offset given in column 2 must match this value for the check to apply. A space here means that the check is always done.

Col. 4 Start of position range to check

Col. 5 End of position range to check

Col. 6 Type of check to apply:

- 1 = check for valid values
- 2 = check for obsolete values
- 3 = run another check program

Col. 7 Check values. Value depends on the check type:

- 1 = list of values, separated by commas, that are valid for this position range. If the position

range is only one character, the commas may be omitted.

- 2 = list of values, separated by commas, that may be present but are obsolete. If the position range is only one character, the commas may be omitted.
- 3 = name of check program to run

Example of the table:

```
!   2  3  4   5  6  7
!!-!!!-!-!!!-!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
XX       000-023 3 check_fixed_field_length
XX       005     1 acdnp
BK       006     1 at
                 2 bn
CF       006     1 m
MP       006     1 ef
                 2 b
MU       006     1 cdij
                 2 b
```

check_doc_line

Location of the table: tab directory of the library

Purpose of the table: Checks on tags

Related table(s):

Format of D (dependency between subfields) table:

1) aleph/error_<lng/check_doc
2) formats.<lng>
3) tab01.<lng>

The **AL** section of this table enables you to define the following checks:
Valid indicators and/or subfield codes for the tag.
Presence of mandatory subfields.
Non-repeatability of non-repeatable subfields.

The **D** section of this table enables you to determine the rules for checking dependencies among subfields of a single field.

Structure of the table:

Col. 1 Type of check - AL or D

col. 2 Record format as in the FMT field. Use XX for all formats.

Format of AL (allowed) table:

col. 4 Tag

col. 5 1) - (hyphen) for indicator
2) Subfield code

col. 6 1) Indicator - possible values for 1st indicator
2) Subfield - 0=non-mandatory; 1=mandatory

col. 7 1) Indicator - possible values for 2nd indicator
2) Subfield - 1=not repeatable;2-9=subfield can be repeated up to the number of times entered here; "-"=unlimited occurrences.

Note: The indicator portion (for all formats) must be listed before the subfield portion, for each field.

Example of the **AL** table:

```
!  2   -    4    5 6 7
!!-!!-!!!!-!!!---!-!-!
AL XX      010   -
AL XX      010   a 0 1
AL XX      010   b 0 -
AL XX      010   z 0 -
AL XX      010   8 0 -
…
AL XX      050   -   0
AL XX      050   - 0 0
AL XX      050   - 1 4
AL XX      050   a 1 -
AL XX      050   b 0 1
AL XX      050   3 0 1
AL XX      050   8 0 -
…
```

Format of D (dependency between subfields) table

col. 3 Error number of message to be displayed as defined in aleph/error_<lng>/check_doc

col. 4 Tag

col. 5 First subfield code

col. 6 Specific content of subfield

col. 7 Type of dependency
Y subfield present
N subfield cannot be present

col. 8 Second subfield code

col. 9 Specific content of subfield

col. 10 Type of dependency
Y subfield present
N subfield cannot be present

col. 11 Third subfield code .. and so on. (up to 3 subfields can be defined)

Example of the **D** table:

```
!*D:                                                  1 1                1
!*  2   3    4    5      6         7 8      9         0 1      12        3
!!-!!-!!!!-!!!---[-!!!!!!!!!!!!!!-!-[-!!!!!!!!!!!!!!-!-[-!!!!!!!!!!!!!!-!-


!D  XX 9036 780   a                Y t                Y
!D  XX 9036 300   a                Y c                Y
```

check_doc_line_contents

Location of the table: tab directory of the library

Purpose of the table: Checks on contents of a tag

Related table(s):

1) formats.<lng>
2) tab01.<lng>

This table checks the *contents* of defined fields.

Structure of the table:

col. 1 format (Use ## for all).

col. 2 tag and indicators

col. 3 subfield code (If blank field is taken as is).

col. 4 name of check program, values are:

isbn - verifies that the ISBN entered in the field is a valid ISBN.

issn - verifies that the ISSN entered in the field is a valid ISSN.

length - verifies that the length of a numeric string matches the values defined in column 5.

number_length - verifies that the number_length of a numeric string matches the values defined in column 5.

range- verifies that the numeric string entered in the field matches the range defined in column 5.

col. 5 value(s) to check. Valid options are:

range <from> <to>

For example, for subfield $c of MARC 21 field 260, enter reasonable values for the range of the year (for example, 1750-2003):

```
## 260   c range          1750 2003
```

length <length>

For example, for subfield $c of MARC 21 field 260, the length is "4" for the year:

```
## 260   c length         4
```

Example of the table:

```
!!-!!!!!-!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!
## 020## a isbn
## 022## a issn
## 7#### x issn
## 260## c range                1850 2002
## 022## a length              9
```

Note: *number-length*. This is the same as *length,* but takes into account only the first numerical string. Example:

c.1996;xx is translated to 1996 and then counted.

check_doc_mandatory

Location of the table: tab directory of the library

Purpose of the table: Checks on forbidden errors in a document record

Related table(s):

1) Check_doc_doc
2) check_doc.<lng>.
3) aleph/error_<lng>/check_doc (error messages)

This table can be used to define whether cataloging check routines should activate a trigger or be defined as forbidden.

For example, if the check_doc_doc table is used to define that a record must have a 245 field (for example, OC XX 5002 01 01 245##) then you can set error message 5002 to activate a trigger or be defined as forbidden.

In addition, it is possible to define (through the check type column - col.1) that the error message activates a trigger or is forbidden only in particular instances for example, when records are saved – check type CATALOG-INSERT).

Note that column 3 defines whether the error activates a trigger or is forbidden:
T - Error writes a cataloging trigger record, but allows update of the database. Triggers can be later retrieved through the Cataloging module or in batch mode.
M - Forbidden error, does not allow database update. For example:
if the record is missing the 245 field,
if the relevant error is set to M
and the check type is CATALOG-INSERT

Error codes from 5001 to 7000 are defined by the user in check_doc.<lng>.
Error codes from 0001 to 5000 and from 7001 to 9999 are system-defined, and can be found in aleph/error_<lng>/check_doc (error messages)

Structure of the table:

Col. 1 Check type

The check type defines when the check program is performed.

Valid values are:

CATALOG-INSERT: performed when the cataloging record is saved, updated or when the Check Record option is selected from the Cataloging module.

CATALOG-DELETE: performed when the Delete Record from Server option is selected from the Cataloging module.

BATCH-DELETE: performed when the Delete Bibliographic Records (p-manage-33) batch process is run.

NAV-MAP-DELETE: performed when the Total Delete option is selected from the Record Manager of the Cataloging module to delete a bibliographic record together with all associated records.

P-MANAGE-25: performed when the P-MANAGE-25 check option is selected when running the FIX and Check Catalog Records (p-manage-25) batch process.

Note that if this column is left blank, then the error code defined in column 3, applies for all check types.

P-MANAGE-11: performed when the Check Database Tables (p-manage-11) batch process is run

Z39-INSERT: performed when a record is inserted via Z39.50 ES Update.

Z39-REPLACE: performed when a record is replaced via Z39.50 ES Update

Z39-DELETE: performed when a record is deleted via Z39.50 ES Update

Col. 2 Identifying number of the check program

Col. 3 **T** - Errors that writes a cataloging **trigger** record, but allows database update. Triggers can be retrieved in the Cataloging client or in batch mode.

M -.Error which does not allow database update.

Col. 4 Description (non-functional)

Note: Defining error 9999 as **T** allows an application to override the system limit of number of errors preventing a record update (set at 40). If 9999 is not defined in check_doc_manatory it defaults to **M**(andatory).

Examples of system-defined codes:

- 0101 – non unique index entry
- 0110 - new access entry

Example of the table:

```
!         1           2   3                    4
!!!!!!!!!!!!!!!!!!!-!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
CATALOG-DELETE       0011 M ADM record $1 in library $2 points to current d
with link type $3
CATALOG-DELETE       0012 M HOL record $1 in library $2 points to current d
with link type $3
CATALOG-DELETE       0013 T BIB record $1 in library $2 points to current d
with link type $3
CATALOG-DELETE       0002 M Document has item(s) attached to ADM record
…
NAV-MAP-DELETE       0002 T Document has item(s) attached to ADM record
NAV-MAP-DELETE       0004 T Document has order(s) attached to ADM record
NAV-MAP-DELETE       0006 T Document has subscription(s) attached to ADM re
…
```

check_doc_new_acc

Location of the table: tab directory of the library

Purpose of the table: Fields that should be ignored when checking new headings in the headings list

Related table(s):

1) tab01.<lng>

This table defines the fields that are indexed but should be **IGNORED** for purposes of the check messages regarding new acc headings.

Structure of the table:

Col. 1 Field Code of the fields that should be ignored while checking for unique headings in the Heading List (ACC index). Use the hash (#) as a placeholder for undefined tags and/or indicators.

Example of the table:

```
! 1
!!!!!
050##
082##
245##
260##
```

check_doc_new_acc_aut

Location of the table: tab directory of the **BIB** library

Purpose of the table: Fields that should be ignored when checking new headings in the headings list together with check in the AUT library

Related table(s):

1) tab_aut

2) tab01.<lng>

This table defines the fields that should be **IGNORED** when checking for unique Headings index (ACC) entries, combined with a check in the relevant AUT library as defined in tab_aut.

Structure of the table:

Col. 1 Field Code

Example of the table:

```
! 1
!!!!!
050##
082##
245##
260##
```

check_doc_tag_text

Location of the table: tab directory of the library

Purpose of the table: Check of pre-defined texts for subfields

Related table(s):

1) tag_text.dat
2) tab01.<lng>

The check routine allows for validation of pre-defined texts for fields (as defined in tag_text.dat)

Structure of the table:

Col. 1 Tag + indicators
Col. 2 ALPHA
Col. 3 Subfield
Col. 4 Value or description

Note: Use # in cols. 1-3 in order to indicate any character

Example of the table:

```
! 1   2 3                        4
!!!!!-!-!-!!!!!!!!!!!!!!!!!!!!!…
LKR   L a UP
LKR   L a ITM
LKR   L a ANA
LKR   L a PAR

655## L a Biographies
655## L a Catechisms
655## L a Essays
…
STA## L a CIRC-CREATED
STA## L a PROVISIONAL
STA## L a SUPPRESSED
STA## L a DELETED
```

circ_events_users

Location of the table: tab directory of the **USR** library

Purpose of the table: Set patrons who are relevant for circulation gateway events (Z502 and aleph_q_gate)

circ_events_users defines the local user status and the sublibraries (Z305) for which circulation events are created.

Circulation events are recorded only for users that have a local user record (Z305) in the relevant sublibrary (Z305-SUB-LIBRARY) with one of the defined patron statuses (Z305-BOR-STATUS).

Structure of the table:
Col. 1 - Patron local sub library (Z305-SUB-LIBRARY).
Set ##### for all sublibraries.
Col. 2 thru 7 - Patron status (Z305-BOR-STATUS).
Set ## for all statuses
For example ./usr00/tab/ circ_events_users

The following is an example of the table:

```
! 1   2  3  4  5  6  7
!!!!!-!!-!!-!!-!!-!!-!!>
MED   01 02 03 04 11 12
TML   01 02 03 04 05 06
WID   01 02 03 04 05 10
```

circ_logger

Location of the table: aleph/error_<lng> directory

Purpose of the table: Error messages linked to the circulation log
Related Tables:

1) tab_circ_log.<lng>

Error messages linked to tab_circ_log.<lng>

Structure of the table:

Col. 1 Error code
Col. 2 ALPHA
Col. 3 Error message

Example:

```
0005 0000 L Loan deleted
0006 0000 L Late return, Original due date:$1. Actually returned
0007 0000 L Late recall return, Recall due date:$1. Actually retu
$2
0008 0000 L Return date:$1.
0009 0000 L Self check loan due date:$1.
0010 0000 L Salf check return.
0011 0000 L Staff renewal. New due date $1.
0012 0000 L Web renewal. New due date $1.
0013 0000 L Bach renewal. New due date $1.
0014 0000 L Salf Check renewal, new due date $1.
```

check_doc_unique_index

Location of the table: tab directory of the library

Purpose of the table: Fields that should be ignored when checking for duplication

Related table(s):

1) tab01.<lng>

Structure of the table:

Col. 1 Field Code

Example of the table:

```
!!!!!
010##
024##
028##
037##
086##
```

codes.<lng>

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Valid tags and aliases for the Cataloging Client

Related table(s):

1) tab01.<lng>

This table defines the valid tags and aliases for the Cataloging Client. In the online cataloging module the list can be activated using the F5 function key.
When cataloging, the cataloging template can be set to display only the MARC tag, or the MARC tag and a library defined alias (for example, 245 - Main title).
The option for work with aliases is set in:

DisplayTagInfo (=Y, or =N) in the [Editor] section of the PC's catalog.ini file.
If option not set, defaults to *N*.

Structure of the table:

Col. 1 tag

Col. 2 Y or N

- Y - tag can be accessed from the 'new field (F5)'option in the edit menu on the client
- N - will not be accessible from the 'new field (F5)'option. However, if the tag is displayed in the template, or record, it will display with its alias, and with its description on the status bar.

Col. 3 Y or N

- Y - tag can be edited only by form
- N - tag can be edited using the editor (via the template) or a form

Col. 4 Y or N

- Y - tag cannot have subfields
- N - tag can have subfields

Col. 5 ALPHA of name (alias)

Col. 6 Name (alias) of tag. This information displays on the template (if DisplayTagInfo in the catalog.ini file is set to Y)

Col. 7 ALPHA of tag description

Col. 8 Tag description. This information displays on the status bar and in the list of fields (F5).

Example of the table:

```
!1    2 3 4 5 6               7 8
!!!!!-!-!-!-!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
LDR   N Y Y L Leader          L Leader (LDR)
001   Y Y Y L Control No.     L Control Number
003   Y N Y L Control No. ID  L Control Number Identifier
005   Y N Y L Date and Time   L Date and Time of Latest Transaction
006   Y Y Y L Add. Fixed Data L Fixed Length Data/Additional Characterist
007   Y Y Y L Phys.Descrip.   L Physical Description Fixed Field
008   Y Y Y L Fixed Data      L Fixed Length Data Elements
…
110   Y N N L Corporate Name  L Main Entry - Corporate Name
111   Y N N L Meeting Name    L Main Entry - Meeting Name
130   Y N N L Main Uni Title  L Main Entry - Uniform Title
210   Y N N L Abbrev. Title   L Abbreviated Title
211   N N N L Acron. Title    L Acronym or Shortened Title [OBSOLETE]
212   N N N L Var. Acc. Title L Variant Access Title [OBSOLETE]
214   N N N L Augmented Title L Augmented Title [OBSOLETE]
222   Y N N L Key Title       L Key Title
```

config.xml

Location of the table: alephe/sru_server directory

Purpose of the table: The `config.xml` file is the main configuration file of the `sru_server`. It defines search bases, the location of the backend `z39_server`, the format of responses, and so on.

Related table(s):

1. pqf.properties

Structure of the table
The table is in XML structure. The root element of `config.xml` is <`proxy`> element. It contains the following elements:

- <`target`> - Mandatory. This element must be defined for every ALEPH base which is exposed through the `sru_server`. See *url Element* for an explanation of this element.
- <log>. Optional. This element defines the content of the log file. See part 0 *syntax Element* for an explanation of this element.

<target> Element Structure

This element has 2 attributes:

- **name** – Unique id of the target
- **database** – Must be equal to the ALEPH base code and to base of the `z39_server`

The element includes the following sub-elements:

- **<url>** – Mandatory, single occurrence
- **<syntax>** – Optional, repeatable
- **<explain>** – Mandatory, single occurrence
- **<cql2rpn>** – Mandatory, single occurrence

Each element is detailed below.

For example:

```
<target name="server1" database="usm01">
    <url>il-aleph02:9991</url>
    <syntax type="xml" marcxml="1" identifier = "info:srw/schema/1/marcxml-
    1.1">
          <name>marcxml</name>
    </syntax>
    <explain>
          <serverInfo>
                <host>il-aleph02</host>
                <port>9000</port>
                <database>usm01</database>
          </serverInfo>
    </explain>
    <cql2rpn>pqf.properties</cql2rpn>
</target>
```

url Element

The element contains the host and port of the `z39_server`.

syntax Element

Each base exposed by the `sru_server` can be configured to use the DCXML structured responses or the MARCXML structured responses.

- **DC XML** – If the DCXML structured responses are required, then no `<syntax>` element should be defined. In this case the `sru_server` accepts `searchRetrieve` requests without the `recordSchema` parameter and returns the DC XML structured responses. An error is returned if the `recordSchema` is specified in the request.

- **MARCXML –** If MARCXML structured responses are required then the `<syntax>` element should be defined as follows:

```
      <syntax type="xml" marcxml="1"
            identifier="info:srw/schema/1/marcxml-v1.1">
            <name>marcxml</name>
      </syntax>
```

The `sru_server` accepts `searchRetrieve` requests without the `recordSchema` parameter or with `recordSchema` equal to `marcxml`. It returns MARCXML structured responses.

explain Element

This element contains information that is returned in the explain response. The element must contain a <serverInfo> element with the basic information about the server, in the following format:

```
<serverInfo>
     <host>host</host>
     <port>port</port>
    <database>database</database>
</serverInfo>
```

- **Host** – sru_server host
- **Port** – sru_server port
- **Database** – The exposed base code. The database must be equal to the database code in the database attribute of the <target> element.

Additional information may be added to the <explain> element according to the http://explain.z3950.org/ specification.

cql2rpn Element

This element contains the name of the file that contains translation of the CQL language used by the SRU\SRW protocol to the RPN language used by the Z39.50 protocol. The file is used when CQL search query received by the sru_server is translated to a RPN query and passed to the z39_server.

The ALEPH file for CQL to RPN translation is named pqf.properties and is stored in the alephe_tab/sru_server directory. That is why the content of this element **must** be equal to pqf.properties. For a description of the pqf.properties file see section *pqf.properties*.

log Element

This element defines the contents of the log file created by the user. It contains one or more log formats separated by spaces. For example:

```
<log>clients-requests client-apdu server-apdu</log>
```

Possible values are:

- **client-apdu** – Messages received by the sru_server are logged.
- **server-apdu** – Messages sent by the sru_server to the z39_server are logged.
- **clients-requests** – Information about the requests received by the sru_server is logged. This is the default value.
- **servers-requests** – Information about the requests sent to the z39_server is logged.
- **client-ip** – The IP addresses of clients connecting to the sru_server are logged.

data_supplier_priorities

Location of the table: tab directory of the BIB library

Purpose of the table: Define supplier priority when merging document records

Related table(s):

1) tab_preferred

This table sets a priority for each supplier. When there are two matching records, the number assigned in this table determines which record is preferred.
The preferred_doc_suppl_prio routine defined in tab_preferred consults the data_supplier_priorities table for the supplier code located in field 970$v. The supplier with the lower number has the higher priority, so that 001 is highest possible priority and 999 is the lowest.

Note
The preferred_doc_suppl_prio routine in tab_preferred also consults a weighing table (status table) when determining the supplier priority.

The structure of table is as follows:

```
Col. 1  Supplier code
Col. 2  Priority
```

The following is an example of the table:

```
! 1                                                    2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!
SUPP1                                                031
SUPP2                                                002
SUPP3                                                020
SUPP4                                                002
SUPP5                                                001
```

drm_routines

Location of the table: tab directory of the **BIB** library

Purpose of the table: Check routines for user authorization and payment related to digital rights

This table check routines for user authorization and payment related to digital rights.

It defines the routine to use to determine whether the patron has the right to view an electronic object. The view routines are set in the "AUT" lines.
The table also defines the routine to use to set the cost of the view. The payment routines are set in the "PAY" lines, and are not yet active.

The routines available for viewing permission are
bor_permission: currently available routine is "1", which checks the "Link Permission" in the patron's profile (Z61-856-PERMISSION).

course_enrollment: currently available routine is "3", which checks whether the patron is listed under the course in the Z107 table.
Structure of the table:

Col. 1 Code type

- AUT - check authorization
- PAY - commit payment (patron and vendor)

Col. 2 Procedure name

Col. 3 Check Procedure

Col. 4 Procedure parameters

Example of the table:

```
! 1           2          3                    4
!!!-!!!!!!!!!!!!!!!!!!!-!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
AUT bor_permission       1
AUT bor_status           2
AUT course_enrollment    3
!
PAY bor_permission       1
PAY bor_status           2
PAY course_enrollment    3
```

edi_in_attr

Location of the table: tab directory of the **ADM** library

Purpose of the table: EDI incoming messages attributes

Related table(s):

1) tab_character_conversion_line

This table is for defining special attributes of incoming EDI messages as needed for specific vendors.
The first field defines the Vendor code
The second field contain up to 100 switches each 1 character in length
The third field identifies the label from tab_character_conversion_line that determines the program and the tables needed for the character conversion.

Description of the different switches:

1 EDI-IN-FTX-ONLY
N (default) = In serial INVOIC messages FTX, IMD and DTM tags will be loaded into Z75-I-NOTE depending on the available room.
Y = Only FTX will be loaded into Z75-I-NOTE.

2 EDI-IN-ADDED-AMOUNT-SEP

N (default) = In INVOIC messages (serial as well as monograph) the added amount will be devided between the lines.
Y = The added amount will be stored separately in an extra line.

Structure of the table:

Col. 1 Vendor code
Col. 2 Switches
Col. 3 Character conversion identifier

Example of the table:

```
!          1                      2
!                                                    3
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!…!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!…
SWETS
                                                UTF_TO_8859_8
```

edi_out_attr

Location of the table: tab directory of the **ADM** library

Purpose of the table: EDI outgoing messages attributes

Related table(s):

1) tab_character_conversion_line
2) tab35

This table is for defining special attributes of outgoing EDI messages as needed for specific vendors.
The first field defines the Vendor code
The second field contain up to 100 switches each 1 character in length
The third field identifies the label from tab_character_conversion_line that determines the program and the tables needed for the character conversion.

Description of the different switches

1 EDI-OUT-REMOTE-FILE-NAME
0 (default) = **ALEPH**'s format. edi.YYYYMMDD.HHMMSS.n.xxxxx. Where xxxxx is the sub-library code or the active-library code. Sub-library code for orders from a single sub-library, claims or cancellations. Active-library code for orders from multiple sub-libraries.
1 = Blackwell format of remote file name to be sent through FTP. The format is: EPInn..DDDMMYY.R. DDMMYY is the date and nn is the sequence number of the file sent during the same day, and all the rest of the characters are constant.
2 = EBSCO format. "CLAIMS.FIL"
3 = Ingram format. This format is suitable for VMS computers. 8 numeric characters (taken from the clock) and 3 constant characters- epo for ORDERS.

4 = Harrassowitz format. YYYYMMDDHHMMSS.XXX.edi.xxxxxx.
XXX is the EDI library code defined in tab35 and xxxxxx is the message type (orders, ostenq or ordchg)
5 = Brodart format. Aleph's format succeeded by ".ord" for orders.

Additional values to this switch (6-9) may be added easily. Each value corresponds to a different program.

2 EDI-OUT-FTP-MODE
A (default) = ASCII mode will be used for sending the outgoing messages. EDI file will include many lines, one line for each segment.
B = Binary mode will be used for sending the outgoing messages. The EDI file will include one long line.

3 EDI-OUT-ORDER-NUMBER-1
0 (default) = Z68-ORDER-NUMBER-1 is not added to the EDI ORDERS message as a second order number.
1 = Z68-ORDER-NUMBER-1 is added to the EDI ORDERS message as a second order number.

4 EDI-OUT-ORDER-NUMBER-2
0 (default) = Z68-ORDER-NUMBER-2 is not added to the EDI ORDERS message as a third order number.
1 = Z68-ORDER-NUMBER-2 is added to the EDI ORDERS message as a third order number.

5 EDI-OUT-FTP-PASSIVE
N (default) = "passive" command is not inserted in the batch file for ftp delivery.
Y = "passive" command is inserted in the batch file.

6 EDI-OUT-FTP-SEND-COMMAND
A (default) = APPEND command does not overwrite a file on the remote computer if the remote file has the same name as the new file. If the new file differs in its name of all the files on the remote computer no file will be overwritten.
P = PUT command overwrite a file on the remote computer if the remote file has the same name as the new one.

7 EDI-OUT-BUDGET
Y (default) = Budget code will be sent in relevant EDI outgoing messages.
N = Budget code will not be sent in EDI messages.

Structure of the table:

- Col. 1 Vendor code
- Col. 2 Switches
- Col. 3 Character conversion identifier

Example of the table:

```
BLACKWELL                1
SWETS                0A
UTF_TO_8859_8
EBSCO                2B
INGRAM               3   Y
MIZRAHI              0B   A
```

edit_doc.<lng>

Location of the table: tab directory of the library

Purpose of the table: Document format definition for GUI/WWW and printouts

Special note: This table can be duplicated with any base extension as defined in cols. 4, 5 and 6 of tab_base.<lng>. For example edit_doc.serials for the logical base "serials".

Related table(s):

edit_doc.<lng>

1) edit_paragraph.<lng>

The edit_doc table defines a concatenation of a number of paragraphs It is used in conjunction with edit_doc.<lng> and edit_paragraph.<lng>.

Structure of the table:

Col.1 document format number. Note that format **001** cannot be configured.
formats 777 and 999 have separate configuration in: edit_doc_777 and in edit_doc_999.<lng>.
format 012 **can be used** by the system for display of bibliographic information in the OPAC list of items display.

Col. 2 Prefix

Col. 3 Suffix

Col. 4 Paragraph identifier as defined in *edit_paragraph*

Use two number signs - ## - to indicate that data following should start on new line

Example of the table:

```
!1        2                 3          4
!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!
037 System number^  ##  ##           001
037                                  110
037                                  120
037                                  130
037                                  140
037                                  004
038 System number^  ##               001
038                                  110
038                                  120
038                                  130
038                                  140
038                                  004
```

edit_doc_777

Location of the table: tab directory of the library

Purpose of the table: Record conversion format for export to reference manager such as Endnote

Related table(s):

1) tab01.<lng>

This table defines the format 777 for the Save/Mail option in OPAC. The format converts the bibliographic records into "ISI ResearchSoft Tagged Output Format" – the format used to import records into reference managers, for example, Endnote. The conversion is driven by the table edit_doc_777. This function uses the form www-set-mail-777-00 (in the directory xxx01/root/ form_<lng>) for output.

Structure of the table:

Col. 1 Tags from ISI ResearchSoft Tagged Output Format. The required tags TY (beginning of record) and ER (end of record) are filled in automatically by the system

Col. 2 Field tag + indicators

Col. 3 1) Subfield(s) of the field (blank indicates entire field).

2) Minus (-) sign followed by subfield(s) to be stripped

Col. 4 Starting position: Defines the position from which to take data from a fixed field. For example, if column 1 is YR, the year might be taken from a fixed field. In this case, you will define the position in the fixed field from which to commence taking 4 positions, counting from base 01. If the fixed field has a subfield code, add 3 to the starting position in order to take it into account.

for example, 0008 to define the 8th position of the 008 field in MARC 21, 0013 to define the 9th position of the 100 field in UNIMARC.

Example of the table:

```
!1   2        3      4
!!-!!!!!-!!!!!!!!!!-!!!!
ID SID   bc
AU 100## a
AU 110## a
AU 700## a
AU 710## a
AU 711## a
TI 245##
T2 246##
T2 240##
T3 440##
T3 490##
PY 008##            0008
```

edit_doc_999.<lng>

Location of the table: tab directory of the library

Purpose of the table: Full document format

Related table(s):

1) This table can be duplicated with any base extension as defined in cols. 4, 5 and 6 of tab_base.<lng>.

2) edit_field.<lng> indicator in
    a. col. 6 and
    b. col. 10 - **E** - External Link.
3) The BROWSE and FIND links defined in col. 10 link to Authority and Word lists in
    a. tab11_aut and
    b. tab11_word
4) Display of some fields may relate to expand programs in tab_expand.
5) The order of the list of sublibraries is defined in tab_sub_lib_sort.
6) Each line in tab_buf_z403 must have a corresponding tag in this table.
7) Col. 3 in tab_buf_z403 can be used in place of Cols. 3, 4 and 6 in this table, to define parameters for the FULL display in the Web OPAC.
8) tab_service related to col. 13
9) tab_z103 for **ANU** and **AND** type of links
10) formats.<lng>
11) tab01.<lng>

This table defines the ***linked*** display format for documents for viewing in the GUI Search and in the Web OPAC. Display can include "expanded" fields that might be added from HOLdings, items or ACC (Z01) fields.

In addition, there are definitions that determine how links to items display (ITM...), and whether links to other records will display.

NOTE: the links are BROWSE and FIND links to Authority and Word lists as defined in tab11_aut and tab11_word in of the tab directory of the library.

Note the feature for external links: the link (with or without the copyright flag) passes through a program that checks the patron permission to view the link. Dependent on *profile* definitions a patron will/will not be able to see these links (as defined in Z61-856-PERMISSION).

Note the following definitions – in addition to field tag - for col. 1

UP - display of the Up link in linked records
PAR - display of Parallel records
DN - display of Down link in linked records
ANU - (analytical up) for MAB format
AND -(analytical down) for MAB format.
ANU and **AND** are used together with the update_z103_mab_ana program that can be added to tab_z103.
ICON - For interaction with Syndetics.
ITM - hypertext link to holdings information of display of Sublibrary (GUI Search)
ITMG - hypertext link to Global-Holdings (all items) (GUI Search)
ITM1 - global; that is, displays one ITM line in doc; leads to display of:
bib record display
information from HOL records (if set to display)
all items in one list (all years/vols/libraries)
ITM2 -by year; that is, intended for serials, displays ITM for each year (based on Z30-YEAR); leads to display of:
bib record display
item records for the particular year.
ITM3 - by sublibrary; that is, displays ITM line for each sublibrary (taking items and HOL records together); leads to display of:
BIB record display
information from HOL records (if set to display) of the sublibrary
item records of the particular sublibrary
ITM4 - link to remote library items, with no circulation information (ACC central database)
ITM5 - by sublibrary (taken from default sublibrary in patron profile). If patron is signed-in, uses default sublibrary from personal patron profile. If signed-in patron does not have profile, or patron is not signed-in, takes IP address of the station (if there are patron records with "IP" as user-id, and patron personal profiles have been defined). If no personal patron profile can be matched, the personal patron profile of user-id **ALEPH** is used. Leads to display of:
BIB record display
information from HOL records (if set to display) of the sublibrary
item records of the particular sublibrary
item records of the last year of issue item records

Note -- this is a way to control access to databases by IP address - by creating a PROFILE for denied bases for 'ALEPH' patrons, for IP users, and for signed-in users.
ITM6 - link for formatting remote items data. Used by the Ex Libris Gate (Z39.50 and **ALEPH**), where link to remote holdings could be through "jump to" (**ALEPH** link) or OPAC record syntax (Z39.50). Therefore, relevant only within EXTnn libraries, and for installations using the Ex Libris Gate.
ITM9 -by year; that is, intended for serials, displays ITM for each year (based on Z30-YEAR) that has visible items; leads to display of: bib record display item records for the particular year.
This is similar to ITM2 only without displaying links that leads to empty list.
ITMH - link to holdings and items, each sublibrary + collection listed on a separate line. Note that the holdings and items display is NOT sensitive to collection, and displays the data at the sublibrary level. Up to 200 sublibraries can be displayed. **Note:** Will only display if there is a holding (HOL) record
ITML - link to remote item information for non-ALEPH local systems. The link is based on the LOW tag which contains the sublibrary code of the local library.
LOC - display of item information built using **expand_doc_bib_loc_usm** and **expand_doc_bib_loc_disp.** Links to items in the same way as **ITM3** - that is, tag text links to library information, and location links to list of items display. **NOTE** that in order for this to work correctly, col. 10 must have an ***L*** definition.
PST - item information, built using expand_doc_bib_loc_[n]... and expand_doc_sort_loc... Links the same as ITM3; that is, tag text links to library information, and location links to list of items display.
DIS - display line (could be to define a blank line)
ERR - if there is a pointer to a non-existing BIB record, line 1000 from www_f_heading is displayed in this line. Therefore, this code **must** be present in the table.

Note that it is possible to define more than one **ITM**x line; for example, a library might want to define **ITM2** and **ITM3**.

Order of Sublibraries:
The first sublibrary to display will be the default sublibrary, set through the profile record, (Z61-BOR-SEARCH-SUBLIBRARY).
The order of the rest sublibraries will be alphabetical. This option concern the display codes **ITM3** and **ITM4** as set in the table tab_sub_lib_sort.

Structure of the table:

Col. 1 Format – display can be sensitive to bibliographic format. Use **#** for all formats:

```
SE SYS                  D LSys. no. Serial     Y
BK SYS                  D LSys. no. Book       Y
AU SYS                  D LSys. no. Aut        Y
## SYS                  D LSys. no.            Y       E
```

Col. 2 Field tag - use wildcard # as necessary for indicators

Col. 3 Subfield to filter on

Col. 4 Contents to filter for (use 3+4 for filter on language)

(for example, if there is "2 usm50" in these columns, only records with usm50 in subfield 2 will have this field indexed). If the contents is prefixed by a hyphen, the content is considered negative (for example, if there is "2 -usm50" in these columns, only records that do not have usm50 in subfield 2 will have this field indexed).

NOTE!! Filter text must be entered in lowercase (irrespective of case in the record itself). Use * to indicate truncated text, use # to indicate single wildcard for text match.

Col. 5 Display subfield

Col. 6 Edit_field indicator

Col. 7 ALPHA

Col. 8 Heading

Col. 9 Display new line. (Y or blank)

In the Web OPAC multiple entries of the same field can be displayed in a run-on mode, in effect forming a single block of data. However, the size of this block of data is limited to 2000 characters. Long, repeated fields might be therefore be cut off. In order to avoid this, repeated fields can be divided into separate entries - each on a separate line.

Col. 10 Display Link Type:

For Web OPAC:

S - Simple-link : Wraps URL in free text with <a href> tags

Z - Link to Services (as defined in tab_service)

A - External Link

L - Links from LOC/PST line to items display

M - Electronic resource link

- in effect only if the field contains subfield "u"
- uses col.3 of tab_buf_z403 instead of cols.3, 4 and 6 of edit_doc_999 for filter and link definitions

For GUI Search:

Z - Find + Browse Link

S - Browse Link

F - Find link

Q - Links (BIB to BIB and AUT to AUT links) **E** - External Link, uses the ***856*** line in edit_field.<lng> for field formatting.

Note the difference between **S** and other link-types like **M** or **L**. **M** / **L** create a link BACK TO THE SERVER with the appropriate information, while "S" type merely wraps the URL with html tags so that no copyrights warning is shown or patron authorization is checked.

Col. 11 Display language. No longer used.

Col. 12 E for display of END-GROUP. This defines the end of a group of alternative field codes. The first occurrence of the first field of the group will display, other fields will be ignored.

For fields with only a single line, an 'E'. should be registered at the end of the line. If there is more than one line per field, the 'E' should be registered ONLY on the last line (end group). In this case, DO NOT put an 'E' for any of the other lines.

Col. 13 The ACC (Z01) file that should be accessed when "headings" are requested in those fields where defined subfields are indexed on different ACC lists (for example, 260 $$a is indexed in the PLAce list, 260 $$b is indexed in the PUBlisher list.). In this case, subfield elements of the tag that are not indexed should be marked with XXX. Used only in GUI Search functions. For Web OPAC, this functionality is defined in col. 3 of tab_service.

Example:

```
## 260##              a D LPlace              Y Z     E PLA
## 260##              b D LPublisher          Y Z     E PUB
## 260##              c D LYear               Y       E XXX
```

Col. 14 W, G or blank:

W - display in the Web OPAC only

G - display in GUI Search only

Blank - display in both Web OPAC and GUI Search

Col. 15 Brief format. "00" is used except in special cases

Col. 16 Limit of linked records that will display. If there are more than the given number, system displays a single line that creates a set and displays a Brief List instead. Default is 99

Example of the table:

```
!                                                 1     1       1
!1   2   3      4     5 6 7         8          9 0 11  2  13   4 15 16
!!-!!!!!-!-!!!!!!!!!!-!-!-!!!!!!!!!!!!!!!!!!!!-!-!-!!!-!-!!!!!-!-!!-!!
MU SYS                  D LSys. No. Music      Y
MP SYS                  D LSys. No. Map        Y
BK SYS                  D LBook Number         Y
SE SYS                  D LRecord No.- Serial  Y
## SYS                  D LRecord Number       Y       E
## 050##                D LLC No.              Y       E
## TYP##                D LRecord Format       Y       E
## 020##                D LISBN                Y Z     E
## 022##                D LISSN                Y Z     E
## 043##                D LGeographic Area     Y       E       W
## 1####                D LMain Entry          Y Z     E AUT
## 880## 6 100*         D LVernacular Author   Y Z     E
## 245##                D LTitle               Y Z     E
CF SYS                  D LSysno Computer File Y
## 880## 6 245*         D LVernacular Title    Y Z     E
## 240##                D LTitle               Y Z     E
```

edit_doc_999_aut_<aut_lib>.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: display of AUT information on cross references

Related table(s):

1) tab01.<lng>
formats.<lng>
2) edit_field.<lng>

In the Web OPAC, clicking on a reference displays the linked Authority Record. The display of the record is defined in the edit_doc_999_aut_<aut_lib>.lng table.

In MARC 21, subfield $w, 4th position (that is, $wxxx4) - of both the 4XX and 5XX - contains a code that enables the generation or suppression of a cross-reference from 4XX or 5XX fields. If the fourth position of subfield $w contains a blank or contains an 'n' (or the subfield does not exist), then the cross-reference is generated and the fields are shown in all relevant places: browse list, expanded authority record, under the heading in the brief record, and so on. If the fourth position of subfield $w contains something different, then the cross-reference is suppressed.

To support this, it is necessary to use the filter columns in this table for the display and in the tab20 table for the enrichment definitions. The following is an example of the lines for this table:

```
## 4#### w !!!n          D LSeen from           Y       E
## 4#### w !!! *        D LSeen from           Y       E
## 4#### w -            D LSeen from           Y       E
## 5#### w g!!n         D LBroader term        Y X     E
## 5#### w g!! *        D LBroader term        Y X     E
## 5#### w h!!n         D LNarrower term       Y X     E
## 5#### w h!! *        D LNarrower term       Y X     E
## 5#### w h            D LNarrower term       Y X     E
## 5#### w g            D LBroader term        Y X     E
## 5#### w -            D LSee also            Y       E
```

Structure of the table: (For full details of the structure of the table see edit_doc_999.<lng>)

Col. 1 Format

Col. 2 Display tag - use wildcard # as necessary for indicators

Col. 3 Subfield to filter on

Col. 4 Contents to filter for

Col. 5 Display subfield

Col. 6 Edit_field indicator as defined in the **authority library**

Col. 7 ALPHA

Col. 8 Heading

Col. 9 Display new line (not active – define as Y)

Col. 10 Display Link Type:

This column is relevant only for record fields that constitute a heading field.

Blank - The field always displays. If the field does NOT exist as a heading in the database, the field simply displays. If the field DOES exist as a heading in the database, the field displays with FIND and BROWSE link.

X - The field displays only if the field exists as a heading in the database.

Col. 11 Display language - no longer used.

Col. 12 E for display of END-GROUP. If there is more than one line per field, put **E** ONLY for the last line (end group). In this case, DO NOT put an **E** for any of the other lines.

Col. 13 Not used

Col. 14 Not used

Example of the table:

```
## SYS                  D LSys. no.            Y       E
## 1####                D LHeading             Y       E
## 260##                D LSub. CSR            Y       E
## 360##                D LSub.  CSAR          Y       E
## 4#### w !!!n         D LSeen from           Y       E
## 4#### w !!! *        D LSeen from           Y       E
## 4#### w -            D LSeen from           Y       E
## 5#### w g            D LBroader term        Y X     E
## 5#### w g!!n         D LBroader term        Y X     E
…
```

edit_field.<lng>

Location of the table: tab directory of the library Purpose of the table: Editing of fields and headings for OPAC display and print

Related table(s):

1. edit_paragraph.<lng> - cols. 3
2. edit_doc_999.<lng> (col.6) www_tab_short.<lng> - col. 8

This table is used to define the display and print options of a tag and of entries in the Browse Headings lists. It is used in conjunction with *edit_paragraph* and *edit_doc.* If a tag is not defined in this table, it will be displayed as is. Therefore, tags entered in the database *without* punctuation, should be defined here in order to add punctuation.

Line type Identifier:

1 - defines field code (col.3), line id (col.4), editing (col.5), subfields to strip (col.6), field prefix (col.9), and field suffix (col.10)

2 - defines subfield codes, prefixes and suffixes.

The ID of the line (col.4) in this edit_field table serves as a link to the edit_field ID defined in the

- edit_paragraph.<lng> (col.3), -
- edit_doc_999.<lng> (col.6), and
- www_tab_short.<lng> (col.8) tables

in the library's tab directory.

In addition, there are set codes, as follows:

- **H** is used for ACC (headings) display in the Web OPAC
- **C** is used for ACC (headings) display in GUI clients
- **S** is used for pc_tab_short display in GUI
- **L** is used for location (using expand_doc_bib_loc_usm and expand_doc_bib_loc_disp) with link to display of library info and display of items list
- **1** in HOL is used for 852 display in the holding window
- **2** is used for 852 in items (copies) display

- **B** is used for brief records (Z0101) display

The following are conventions in the USM01 tables:

- **D** is used for "name tags" format in GUI and WWW (edit_doc_999) –
- **3** is used for the paragraphs that make up the catalog card format (edit_doc.eng - format 037)
- **P** is used for the paragraphs that make up the citation format (edit_doc.eng - format 040)

is a wildcard throughout the table for alpha, tags, indicators, subfields and edit_field ID.

In order to cover all instances not specifically listed in the table, the last 2 lines of this table should be:

```
1 # ##### #
2                          # A ^
```

This table can be up to 1000 lines long.

Structure of the table:

Col. 1 line type identifier – line 1 or line 2:

Col. 2 Not used. Should have #

Col. 3 tag+indicator. Note that Z0101 is used for brief records display

Col. 4 ID of edit_field line.

Col. 5 Filter:

- **A or blank** filters for subfields, but does not change their order
- **B** filters for subfields and sets their order

Col. 6 - (minus) followed by subfield codes to strip (for example, -w)

Col. 7 subfield code, or # to indicate "all subfields". Up to 20 subfields can be defined

Col. 8 ***A*** to indicate all occurrences of subfield(s) within the tag

Col. 9 subfield or field prefix

Col. 10 subfield or field suffix.

May be entered in order to start each repeated field on a new line.

Example of the table:

```
! 2   3   4 5     6      7 8     9         10
!-!-!!!!!-!-!-!!!!!!!!!!-!-!-!!!!!!!!!!-!!!!!!!!!!
1 # SUB## H
2                         a A ^
2                         b A ^
2                         c A ^
2                         d A ^
2                         e A ^
2                         f A ^
2                         g A ^
2                         k A ^
…
1 # AUT## #   -69
2                         # A ^
1 # TIT## #   -69
2                         v   ^v.^
2                         # A ^
1 # 020## D
2                         # A ^
1 # 020## 3
2                         a   (          )
```

Note : If the **856** entry appears without subfields in edit_field.<lng> (that is, only the following line appears:

1 # 856## D

the display will be according to the following algorithm:
Use subfield z, or if missing
use subfield u, or if missing
compose by using subfields a, p, d and f as done for URL creation

edit_paragraph.<lng>

Location of the table: tab directory of the library

Purpose of the table: Paragraphs for displaying bibliographic information

Related table(s):

1) edit_field.<lng> – col. 4
2) edit_doc.<lng>
3) bib_format in the ADM library
4) tab01.<lng>

The edit_paragraph table is used for formatting bibliographic record information for display and printing. It defines a concatenation of a number of fields, including "fields" that are included through expand_doc procedures.. It is used in conjunction with
edit_field.<lng> and edit_doc.<lng> in the BIB library, and the bib_format table in the ADM library.

Text can be added to the displayed record fields using col.4 (prefix) and col.5 (suffix). In addition, if you enter DISP in place of a field tag in col.2, you then use cols. 4 and 5 to enter a textual phrase. The DISP "tag" can be repeated as many times as required.

Within a paragraph, a "new line" can be forced between fields by using ## in cols. 4 or 5 (prefix or suffix). When edit_doc is used for formatting bibliographic record information, one format group can include any number of paragraphs. For edit_doc, it is preferable to define a separate paragraph for each new line.

Structure of the table:

Col.1	paragraph identifier
Col. 2	tag + indicators; wildcard may be used **DISP** may be used in order to *display* information (as defined in the prefix), without linkage to a tag
Col. 3	edit_field.<lng> identifier (as defined in Col. 4 of edit_field table.)
Col. 4	prefix
Col. 5	suffix
Col. 6	**A** - all occurrences of the tag **blank** - first occurrence only

Example of the table:

```
!1    2   3     4          5      6
!!!-!!!!!-!-!!!!!!!!!!-!!!!!!!!!!-!
000 DISP    Missing
000 DISP    Paragraph
!*   system number
001 SYS## D [          ]
!*   Author+Title
002 1#### D            .^         A
002 245## D
!*   Title
003 245## D
!*   Note 5##
004 DISP                ##
004 5#### D ^^^         ##         A
```

expand_doc_bib_z30

Location of the table: tab directory of the **BIB** library

Purpose of the table: Item fields expanded into the bibliographic record.

Related table(s):

1) tab_expand - expand_doc_bib_z30 program.

This table defines which fields from the item record (Z30) are expanded into the bibliographic record. The table is used together with the expand_doc_bib_z30 program.
The program creates a new virtual field - Z30-1 (for copy items) or Z30-2 (for issue items) - that contains the item's information.

Structure of the table:

Col. 1 Type of Item:

- Y - Issue items
- N - Copy items
- # - All types of items

Col. 2 Item Field Name. Note that active-library may also be added.

Col. 3 Subfield into which the item field is expanded

Col. 4 Space, zero or blank

- Space - If field is empty the subfield is not added to the expanded field.
- Zero - If the field has zeroes the subfield is not added to the expanded field (used for date fields).
- Blank - Subfield is always taken

Col. 5 Translation Flag

Defines if the code of the item field is to be translated to a name (for example, sublibrary code to sublibrary name). Note that this is available for:

- active-library
- z30-sublibrary
- z30-collection
- z30-item-status
- z30-item-process-status.

Values are:

- Y - Translate
- N - Do not translate
- X - Not applicable

Example of the table:

```
!                  2                     3     4      5
!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!-!
# active-library                         l space      N
# active-library                         L space      Y
# z30-material                           m space      X
# z30-sublibrary                        1 space      N
# z30-sublibrary                        A space      Y
# z30-collection                         2 space      N
# z30-collection                         B space      Y
# z30-call-no                            3 space      X
# z30-call-no-2                          9 space      X
# z30-inventory-number                   6 space      X
```

expand_doc_bib_z403

Location of the table: tab directory of the **BIB** library

Purpose of the table: Expand object data into the bibliographic record.

Related table(s):

1) tab_expand - expand_doc_bib_z403 program.

This table defines which fields from the object data record (Z403) are expanded into the bibliographic record.. The table is used together with the expand_doc_bib_z403 program.
The expand program reates a new virtual field, Z403, that contains the object's information. The table also determines the subfield structure of the new expanded field.

The following parameters can be specified in the tab_expand table in order to modify the program's defaults:

- TAG - This parameter can be used in order to specify that a different new expanded field should be created instead of the default Z403.

- USAGE-TYPE - This parameter can be used in order to specify the type of object's that are included in the expand. By default the program only expands information from objects where the Z403-USAGE-TYPE is set to VIEW. This parameter can be used to specify that other object types should be included.

- CONF - This option can be used in order to determine that a different table should be used instead of the default expand_doc_bib_z403 table.

- DISPLAY-LINK - If this switch is set to "N" Object's data will not be expanded if Z403_DISPLAY_LINK is set to "N" or if the Z403_EXPIRY_DATE is already reached. By default (= "Y") all data will be expanded.

Structure of the table:

Col. 1 # - All types of objects

Col. 2 Object Field Name

Col. 3 Subfield into which the object field is expanded

Col. 4 Space - If field is empty the subfield is not added to the expanded field.

Zero - If the field has zeroes the subfield is not added to the expanded field (used for date fields).

Blank - Subfield is always taken

Col. 5 Translation Flag. Defines if the code of the object field is to be translated to name (for example, sublibrary code to sublibrary name).

Note this is available for:

- Z403-sublibrary, Z403-usage-type, and
- Z403-restrict-sublibrary.

Values are:

- Y – Translate
- N - Do not translate
- X - Not applicable

Example of the table:

```
!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!-!
# z403-title                          y space      X
# z403-sublibrary                   l space      Y
# Z403-usage-type                     t space      N
```

fix_doc.<lng>

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Fix_doc definitions

Related table(s):

1) tab_fix - Col. 1 of this table should match the definition in Col. 1 of tab_fix

Defines the fix_doc routines that are included in the Cataloging module under the Edit menu/Fix record and the Edit menu/Derive record options.

Structure of the table:

Col. 1 Procedure identifier (as defined in Col. 1 in tab_fix)

col. 2 Derive or Fix:

Defines if a new record is going to be created when performing a fix routine, or if the current record is going to be fixed.

- Y=Open as new document
- N=fix current document.
- C = Conditional Open as a new record. Only active when the record has a system number (that is, is not NEW).

col. 3 ALPHA – should always be L

col. 4 Text that displays in the window when user invokes 'fix record' or the Derive record option from the edit menu in the cataloging client.

Note that in the online Cataloging module:

- lines with "N" will appear in the Edit Menu under ***Fix record***
- lines with "Y" will appear in the Edit Menu under ***Derive new record.***

Derive new record options:

Create an analytic entry from an existing document.
The fix creates a template with an empty 245 field; copies the 260, 300, 050 008 and LDR fields from the parent record; creates an LKR record with ANA link and $$n=245 of parent record.

The record is created for MARC 21 records as follows:

- LDR: with position 8 as b and not the default which is a.
- 008: default as a book. Note: it would be correct to update the year in this field from the year in the LKR. The language should be the same as the 008 of the parent record.
- 050: If the fields exist in the parent record they are copied into the derived record.
- 080: If the fields exist in the parent record they are copied into the derived record.
- 245: will be opened but will contain no text.
- 260: will be taken from the parent record
- 300: $$c will be taken from the parent record
- LKR: points to parent record with subfields as follows:
  - $$a ANA
  - $$b sysno. of parent record
  - $$n title of parent record

Create an authority record in the AUT library using the same field 100, and creating a 670 built out of 245 $$a and 260 $$c.

Example:

```
! 1   2 3           4
!!!!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
INS   N L Perform fixes as executed when document is updated (INS)
PUNC  N L Fix document's punctuation.
04-01 N L Convert UNIMARC Records to USMARC Records
008   N L Update 008 field from 260 field
ANA   C L Create a new Analytic record
AUT   Y L Create a new record in the Authority DB, based on 1XX, 7XX, 6X
INS2  N L INS2 procedures
TEST  N L Testing various fixes
OCLC  N L OCLC fix routines
```

form_description.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: Defining a subject for a form sent by e-mail

Structure of the table:

Col. 1 Form name

Col. 2 Form format. If left blank the entry will apply for all formats.

Col. 3 Description

Example of the table:

```
!      1                          2  3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!-!!!!!!!!!!!!!!!!!!!!!!!>
opac-print                          An Opac printout sent to you
```

form_print_method

Location of the table: tab directory of the library

Purpose of the table: Print method for different printouts

This table defines the print methods for XML based printouts.
The default method is XML_XSL. Therefore, there is no need to define printouts that use XML_XSL in this table.

If you want to use an external tool (not XML_XSL), it should be defined in Col. 3 in the following way:

EXECUTE <COMMAND>

This assumes that there is an executable that accepts XML as input and produces the requested output.

If you want to use a server-side external tool (not XML_XSL), it should be defined in **Col. 3** in the following way:

SERVER-EXECUTE <DIRECTORY>

This assumes that you will have the XML file under /print in <DIRECTORY> directory. The execution is not under **ALEPH**'s responsibility.

Col. 4 determines whether to translate this XML.
Col 5. determines whether to notify the user with a message about the XML file created. The translate option can also be used for the XML_XSL method.

For example:
To use **Iexplore.exe**, assign it a symbolic name, for example **IE**.
Insert in Col.3: EXECUTE IE

In addition, the following line should be added to the al500/alephcom/tab/alephcom.ini file on the client:

```
[PrintExecute]
IE=C:\Program Files\Plus!\Microsoft Internet\Iexplore.exe
```

The XML will then be handled by Iexplorer.

Structure of the table:

- col. 1 Form name
- col. 2 Form format
  If left blank the entry will apply for all formats
- col. 3 Print method
  EXECUTE <COMMAND> - Use external tool based on alephcom.ini definition for <COMMAND>
  SERVER-EXECUTE <DIRECTORY> - Create the XML file on the server under /print/<DIRECTORY>
- col. 4 translate:
  Y=XML file will be translated
  N=XML file will not be translated
- col. 5 user notification
  Y=User will be notified about the new file created
  N=User will not be notified about the new file created

Example of the table:

```
!      1                          2  3
4 5
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!-!!!!!!!!!!!!…!!!!!!!!-!-
item-copy-label                 20 EXECUTE LABEL_PRINT
item-issue-label                00 EXECUTE LABEL_PRINT
```

form_sms

Location of the table: tab directory of the **ADM** library
Purpose of the table: Set SMS sendng definitions
Related table(s):

1) $aleph_root/error_lng/sms_messages

2) tab18.<lng>

This table defines the printing of which Circulation form will trigger an additional SMS message sending to the patron.
hold-request-letter-*xx*. - The *xx* is the hold request send action.
loan-recall-*x* - The *x* is the recall letter type (1 or 2).
overdue-letter-*x* - The *x* is the optional letter number.
overdue-summary-*x* - The *x* is the optional letter number.
overdue-sum-single-*x* - The *x* is the optional letter number.

The text of the SMS messages is defined in the $aleph_root/error_lng/sms_messages table.

Structure of the table:

Col. 1 Form name

Col. 2 Form format

Col. 3 Message Number:

The SMS text number. This number must be aligned with the codes used in the $aleph_root/error_lng/sms_messages table;

Col. 4 Activate SMS

Y = Send SMS message for this printout

N = do not send SMS message for this printout

Example of the table:

```
! 1                              2  3    4
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!-!!!!-!
hold-request-letter-01            1201 Y
overdue-letter-0               00 1204 Y
overdue-summary-0              00 1205 Y
overdue-sum-single-0           00 1206 Y
loan-recall-2                     1207 Y
loan-recall-1                     1207 Y
```

form_sub_library_address

Location of the table: tab directory of the **ADM** library

Purpose of the table: Address types for forms

Related table(s):

1) Adress type in col. 2 of this table should match address type in col. 2 of tab_sub_library_address.<lng>

This table defines address types for Acquisitions, ILL and Circulation forms.

The table is used to link between the print form and the particular address for those forms that include sub_library address (taken from the alephe/tab_sub_library_address.<lng> table).

The ***address type*** in column 2 is matched against the ***address type*** in column 2 of the alephe/tab_sub_library_address table. If no match is found, address type ***1*** acts as default.

Structure of the table:

Col. 1 Form template name

Col. 2 Sublibrary address type

Example of the table:

```
!      1                          2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!
transfer-slip                     1
serial-rout-list                  1
serial-item-label                 1
serial-claim-letter-01            1
return-receipt                    2
photo-request-wait                1
photo-request-slip                1
```

formats.<lng>

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: List of formats for Cataloging

This table defines the codes (2 characters) of the valid formats which can be entered in the FMT field of the document record.

Structure of the table:

Col. 1 Code

Col. 2 Character set

Col. 3 Format description text

Example of the table:

```
BK L Books
CF L Computer file
MP L Maps
MU L Music
SE L Serials
VM L Visual materials
MX L Mixed materials
```

generic_fix

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Processing script for fix_doc_do_file_08 and p-file-08

Related table(s):

1) tab01.<lng>
2) formats.<lng>

This is a ***sample*** table of a processing script for fix_doc_do_file_08 and p-file-08.

The fix_doc_do_file_08 program is a generic fix program that modifies cataloging records based on a supplied processing script. Many standard fix programs are provided by **ALEPH**, but there are times when a library would like to perform a customized fix on a record. This can be done by the fix_doc_do_file_08 program.

The processing script ***must*** be located in the tab/import directory of the library. The table name is user-defined. You can create multiple tables to define different fix procedures. The script is in the format of a normal **ALEPH** table with 9 columns.

The generic_fix table in the USM01 $data_tab/import directory is an example of a processing script. Specifications for the conversion script can be found in the "How to Set Up a Script for the Correction of Records How-to document."

Structure of the table:

col. 1 Iteration. The operations are performed in order of iteration. 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.

col. 2 Document record field code. # can be used to indicate truncation (for example,, 245## for 2451, 2452, 24501).

col. 3 Record's format code (FMT field). # can be used as a wildcard. If left blank, no filtering for record format is performed.

col. 4 First position filter. If the position filter is not blank, then the operation is performed only if the first position in the field matches.

col. 5 Position range start. Used for operations on fixed fields to specify the position range. Note that field positions are counted from zero

col. 6 Position range end. Used for operations on fixed fields to specify the position range. Note that field positions are counted started from zero.

col. 7 Occurrence filter. Contains a five-digit number, or

- FIRST,
- LAST,
- NOT-F (not first), or
- NOT-L (not last).

col. 8 Operation code. Following are the valid operation codes:

- ADD-FIELD

- CHANGE-FIELD
- CHANGE-FIRST-IND
- CHANGE-FIRST-IND-MATCH
- CHANGE-SECOND-IND
- CHANGE-SECOND-IND-MATCH
- CHANGE-SUBFIELD
- COND-LOAD-VAL-POS
- COPY-FIELD
- COPY-SYSTEM-NUMBER
- DELETE-FIELD
- DELETE-FIELD-COND
- DELETE-FIXED-COND
- DELETE-SUBFIELD
- DELETE-SUBFIELD-DELIMITER
- EDIT-SUBFIELD-HYPHEN
- FIXED-CHANGE-VAL
- FIXED-CHANGE-VAL-RANGE
- FIXED-FIELD-EXTEND
- FIXED-RANGE-OP
- REPLACE-STRING
- SORT-FIELDS

col. 9 Operation parameters

Example of the table:

```
!   2   3  4  5   6    7                  8                            9
!-!!!!!-!!-!-!!!-!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!…
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!…!!!
1 001                     COPY-SYSTEM-NUMBER             035  ,L,a
1 035##                   REPLACE-STRING                 ocm, (OCoLC)
1 LDR                     ADD-FIELD                      OWN  ,L,$$LIN
2 852##                   CHANGE-FIELD                   949
3 949##                   DELETE-SUBFIELD                c
3 949##                   REPLACE-STRING                 $$i,^
3 949##                   CHANGE-SUBFIELD                h c
3 949##                   CHANGE-SUBFIELD                b h
```

ill_bor_charge

Location of the table: tab directory of the ILL library

Purpose of the table: Parameters for charging patrons for ILL borrowing (outgoing) requests

Related table(s):

1) tab18.<lng>

This table defines the parameters for charging patrons for ILL borrowing (outgoing) requests. This table works only if tab18.<lng> of the ADM library is set as follows:

transaction line 0015 - ILL material arrival set to Y in col. 4, and with a zero amount (0) in col. 5.

Structure of the table:

Col. 1 Borrowing ILL Unit
Col. 2 Patron status
Col. 3 Supplier status
Col. 4 Page size
Col. 5 Request Media type
Col. 6 Fixed charge1. Sum with option to decimal point e.g.: 2.5
Col. 7 Fixed charge2. Sum with option to decimal point e.g.: 2.5
Col. 8 Fixed charge3. Sum with option to decimal point e.g.: 2.5
Col. 9 Start counting after page X
Col. 10 Fee per page. Sum with option to decimal point e.g.: 2.5

Example of the table:

```
!1                   2  3  4  5       6             7        8       9      10
!!!!!!!!!!!!!!!!!!!!-!!-!!-!!-!-!!!!!!!!!!!!-!!!!!!!!!!!!-!!!!!!!!!!!!-!!!-
!!!!!!!!!!
ILL_LAW              ## ## ## L 5.5         4.5
ILL_LAW              ## ## ## C 0.5         1                       010 1
ILL_LAW              ## ## ## E 12                                   000
LAW_LT               ## ## ## L 10          2
LAW_LT               ## ## ## C 2                                    000
LAW_LT               ## ## ## E 1                                    000
!LAW_LT               ## ## ## C 2           1                       010 1
!LAW_LT               ## ## ## E 10                                   000
HYL_LT               ## ## ## L 5           0           0
```

ill_format_type

Location of the table: tab directory of the ILL library

Purpose of the table: Specifications for creation of ILL Record's format field

This table is used to define the specification for the creation of the ILL record's 592 (format) field, created by the expand_doc_type program.

The table and the expand program together create a 592 field that contains the format of the record based on the contents of various fields present in the record. These fields

can be a single field or a combination of field values, such as LDR and 008. A code for MN (monograph) and SE (serials) must be defined.

Multiple 592 fields might be created in case of multiple matching of record content and table definitions. The system will create them in the order that they are matched, and this order will influence the system behavior. Therefore, the order of the lines in the table is significant.

Structure of the table:

Col. 1 Target tag

Col. 2 Format code. The value entered in this column is expanded into sub-field $a of the new field created by expand_doc_type. If the column is left blank, then the format name (value of column 3) is added to sub-field $a of the new field. For example, if the table contains the following line:

592 MN Monograph LDR F06-01 EQUAL a

then a new 592 field with the following structure will be added when position 06 of the LDR field contains an 'a':

592 L $$aMN$$bMonograph

If this column is left blank, the new field will be created/ expanded as follows:

592 L $$aMonograph

Col. 3 Format name. If a format code is present (column 2), then the format name is added/expanded into sub-field $b of the new field. If no format code is defined, then the format name is added/expanded into sub-field $a of the new field. For example, if the table contains the following line:

592 MN Monograph LDR F06-01 EQUAL a

then a new 592 field with the following structure will be added when position 06 of the LDR field contains an 'a':

592 L $$aMN$$bMonograph

If this column is left blank, the new field will be created/ expanded as follows:

592 L $$aMonograph

Col. 4 Field tag. The field from the record used for determing the material type that is expanded into the new field. In the following line, the LDR (position 06 with 'a') is used to define that the record is for a monograph:

592 MN Monogaph LDR F06-01 EQUAL a

Col. 5 Sub-field(s)/position. This column contains the sub-field codes or the fixed field positions (of the field defined in column 4) to be checked. In the following line, the program checks position 06 of the LDR:

592 MN Monogaph LDR F06-01 EQUAL a

In the following line, the program checks sub-field $a of the 490 field:

592 MN Monogaph 490## a EQUAL a

Col. 6 Match criteria. This column is used to define the match criteria in relation to the contents of the sub-field or the fixed field positions defined in columns 4 and 5. Following are the available options:

EQUAL
N-EQUAL
EXIST
N-EXIST
MATCH
N-MATCH

Col. 7 Contents. This column contains the contents of the field or of the fixed field position that are used to match on (according to the match criteria defined in column 6). Use [] to enclose multiple values to match on. The relationship between the values is of type OR. In the following line, the match is based on values 'e' or 'f' of position 06 of the LDR field:

592 Map LDR F06-01 EQUAL [e,f]

Col. 8 Case-sensitive matching flag – Y or N

Example of the table:

```
! 1   2            3                          4       5         6              7
!!!!!-!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!-
!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!-!
592   CF Conference                     LDR   F06-01     EQUAL      a
                                        008   F29-01     EQUAL      1
592   CF Conference                     111##            EXIST
592   CF Conference                     711##            EXIST
592   CF Conference                     811##            EXIST
592   CF Conference                     006   F00-01     EQUAL      a
                                        006   F12-01     EQUAL      1
592   SE Serial                         LDR   F07-01     EQUAL      s
592   SE Serial                         006   F00-01     EQUAL      s
```

ill_formats_map.<lng>

Location of the table: tab directory of the ILL library

Purpose of the table: Translation of ILL item type code

This table translates the ISO IllRequest.ItemId.ItemType code for an incoming ILL request to a textual description.

Structure of the table:

Col. 1 IllRequest.ItemId.ItemType code

Col. 2 Textual Description

Example of the table:

```
!            2
!-!!!!!!!!!!!!!!!!!!!!!!!!!
1 Monograph
2 Serial
3 Other
```

ill_lend_charge

Location of the table: tab directory of the ILL library

Purpose of the table: Parameters for charging patrons for ILL lending (incoming) requests

Related table(s):

2) tab18.<lng>

This table defines the parameters for charging lending request when the 'ship'(supply) action is submitted.
This table works only if tab18.<lng> of the ADM library is set as follows:

transaction line 0016 (Library charge for lending ILL request) is set to Y in col. 4, and with a zero amount (0) in col. 5.

Note that in col. 3 of this table it is possible to set the amount according to the request's Level of Service.

Structure of the table:

Col. 1 Lending ILL Unit code
Col 2. Requester status
Col. 3 Requester (borrower) code
Col. 4 Request level of service tag
Col. 5 Supplied media type
Col. 6 Fixed charge1. Sum with option to decimal point e.g.: 2.5
Col. 7 Fixed charge2. Sum with option to decimal point e.g.: 2.5
Col. 8 Start counting after page X
Col. 8 Fee per page. Sum with option to decimal point e.g.: 2.5

Example of the table:

```
        1           2          3            4 5    6         7          8     9
!!!!!!!!!!!!!!!!!!!-!!-!!!!!!!!!!!!!!!!!!!!-!-!-!!!!!!!!!!-!!!!!!!!!!-!!!-!!!!!!!!!!
ILL_HYL              FO ILL_LAW              < L 7          2.5
ILL_HYL              ## ILL_LAW              # L 10         2.5
ILL_HYL              FO ILL_LAW              # C 2.5        1           10  1
ILL_HYL              ## ILL_LAW              # E 12
```

ill_unit_group.<lng>

Location of the table: tab directory of the ILL library

Purpose of the table: Defining ILL Unit groups for shared copyright policy

Structure of the table:

Col. 1 ILL Unit code

Col. 2 Group code

Example of the table:

```
!1                      2
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!
U15AR                GRP01
U15ED                GRP01
```

job_list

Location of the table: alephe/tab directory and tab directory of the library

Purpose of the table: Job daemon scheduling of programs and procedures

Related table(s):

1) job_list.conf

Note that this list can define either programs or procedures.
If programs, col. 5 lists the program commands.
If procedures, col. 5 lists the library code, and cols. 6 and 7 the name parameters of the procedure.

Column definition for **programs**

Col. 1 day

Col. 2 hour

Col. 3 queue (y/n)

Col. 4 log name OR target name (see following)

Col. 5 program to run OR library code (see following)

Column definitions for **procedures**

Cols. 1, 2, 3

as above

Col. 4 target

Col. 5 library

Col. 6 procedure name

Col. 7 parameters

Formatting of parameters/log file name

%D - day (00-06) [Sunday=00 Monday=01 ... Saturday=06]

%H - hour (00-23)
%M - minutes (00-59)

xxxx,print_%D_%H_%M -> if run at 09:00 on Sunday xxxx,print_00_09_00

%DATE[+/-][D/W/M]nnn - create date relative to the current date
'+' - after current date '-' - before current date
D - days W - weeks M - month

xxxx_%DATE+D007 -> set parameter to 7 days from now
yyyy_%DATE+M001 -> set parameter to 1 month from now

Note that macros, or templates can be used based on definitions in job_list.conf:

```
W1 22:00:00 Y                USM50 p_cir_01                USM50
02 D1       Y                USM50 p_cir_01                USM50
```

After editing this file restart jobd (UTIL E/16/9).

Example of the table:

```
!     2     3        4                    5
!!-!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!>
!     2     3        4                5                6
!!-!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!…!!!!!!-!!!!
W2 23:00:00 N                       /aleph/daily_backup/cron_backup_to_ta
/aleph/daily_backup/ backup_log.dat
!*
00 19:30:00 N disk_size_0           df -k
!*
00 23:00:00 Y                       USM01 util_a_10_b
00 23:00:00 Y                       USM01 util_a_11_b
```

job_list.conf

Location of the table: alephe/tab directory

Purpose of the table: Job_list definitions

Related table(s):

1) job_list

This table for the definition of templates or macros for daily or weekly slots that can then be defined in the structure of the table:
Weekly procedure slots:

Col. 1 Code
Col. 2 W(eekly)
Col. 3 Flags (Sun - Sat) Y/N

Daily procedure slots:

Col. 1 Code

Col. 2 D(aily)
Col. 3 Start time
Col. 4 End time
Col. 5 Interval

Example of the table:

```
!1    2    3
!!----!-!!!!!!!
W1    W NYYYYYN
!     2   3     4     5
!!----!-!!!!!-!!!!!-!!!!!
D1    D 09:00 17:00 01:00
```

ldap.conf

Location of the table: tab directory of the **USR** library

Purpose of the table: LDAP configuration file

Related table(s):

1) aleph_start

The LDAP authentication script is activated using check_bor_id_02. In other words, only if the verification type is "02" (z308-verification-type) then the LDAP script becomes relevant.

Required setup:

1) defining the script location in aleph_start: setenv aleph_authen_dir $alephm_source/check
2) defining LDAP server name/port/search-parameters - construct a configuration file under the name $usr00_dev/usr00/tab/ldap.conf.

The general workflow is as follows:–
the patron's Z308 must exist in the database with verification type 02. The verification field itself is irrelevant as we compare the password with LDAP, not with the local verification so this field can be empty or equal the z308-key-data.
Once a patron attempts to log in, the system identifies the verification is an LDAP one (seeing "02" in z308-verification-type) and activates the LDAP script (from the location defined in aleph_start as "aleph_authen_dir").
The script returns "Y" or "N" and from there on, the patron is authenticated as in regular authentication.

Example of the table:

```
[general]
        host_name = exchange01
        port      = 389

        search_base = o=ExLibris
        search_filter = (uid=USERNAME)

[xml setting]
        xml_root_node = bor_authentication
```

Note that the reserved word "USERNAME" will be replaced with the patron login-name before the search takes place.

library_relation

Location of the table: alephe/tab directory

Purpose of the table: Relation between libraries by library type

Related table(s):

1) tab_fix can be used instead of library-relation to define an **AUT** type of relationship.
2) tab_aut – related to **BLK.** This setting is relevant only if the BIB library sets alternate AUT libraries for authority control of a single browse list.

This table defines links between libraries. The system requires that the relationship between the BIB/ADM/HOL and ILL libraries be defined. Relation to AUT library is not required except for the following cases:
ERR, which is relevant only for applications that use p-manage-31
AUT, which is relevant only for fix_doc_new_aut_n programs.

Relation types:
ADM
BIB library /ADM library (libraries)
Defines which ADM libraries are related to the BIB library in the first column. Only one line can be used to define this relationship. It is possible to list up to 60 ADM libraries. However, Ex Libris does not recommend linking more than 20 ADM libraries to one BIB library, if the ADM libraries are using shared bibliographic records.

HOL
BIB/ADM libraries / HOL library (libraries)
defines the HOL library that is related to the BIB library or the ADM library listed in the first column.
All HOL records are actually linked only to BIB records. An item record can be linked to a HOL record through Z30-HOL-DOC-NUMBER-X.

For sites that have multiple ADM libraries, the setup can be a single HOL library, irrespective of the number of ADM libraries related to the BIB library, or multiple HOL libraries, with one HOL library for each ADM library.

An advantage of multiple HOL libraries is simpler authorization control for the HOL records. Both setups function equally well in **ALEPH**. Note that the setup is different for single-HOL-multi-ADM, and multi-HOL-multi-ADM. An explanation follows:

If the setup is a single HOL library, there is no need to define ADM-HOL relation. The relation occurs from ADM to BIB to HOL, and from HOL to BIB to ADM. A sample setup is:

ADM USM01 USM50 USM51
HOL USM01 USM60
BIB USM50 USM01
BIB USM51 USM01
BIB USM60 USM01

If the setup is a HOL library for each ADM (in which case each ADM MUST have its own HOL), all relations must be defined, for ADM-HOL, ADM-BIB, HOL-BIB, BIB-HOL, HOL-ADM. A sample setup is:

ADM USM60 USM50
ADM USM61 USM51
HOL USM01 USM60 USM61
HOL USM50 USM60
HOL USM51 USM61
BIB USM50 USM01
BIB USM51 USM01
BIB USM60 USM01
BIB USM61 USM01

NOTE: if there are multiple HOL libraries that are linked to items in a single ADM library, the setup is the same as single HOL, and there is no need to define a HOL-ADM and ADM-HOL relations.

BIB
HOL/ADM to BIB
defines which BIB libraries are related to the HOL library or the ADM library in the first column.

ILL
ILL to ADM
The first column defines the ILL library for the administrative library which is defined in the second column.

CRS
ADM library/Course Reserve library
Defines which Course library is defined for the ADM library listed in the first column.

ERR
AUT library / AUT library
For p-manage-31 (load authority records) defines the library on which the "unacceptable" records are written.

AUT
BIB library/AUT library
For fix_doc_new_aut_n (for example, fix_doc_new_aut_2) defines the default authority library for which the new derived authority record is created. Note that this entry is not mandatory. If this relation type is not defined, the authority library can be defined through the tab_fix table. If no authority library is defined in the library_relations table and in the tab_fix table, the system uses the default of XXX10 as the authority library.

PID
BIB library/BIB library
The PID library relation links an indexing library to an actual library for the purpose of parallel indexing. The link enables running index jobs in the indexing library in parallel to normal workflow of the actual library. The actual library is not locked during this period.

BLK
AUT library / AUT library
This setting is relevant only if the BIB library sets alternate AUT libraries for authority control of a single browse list, for example, tab_aut setup is:

```
SUB   2 USM10 USM12
```

The above setting infers that the USM10 AUThority record is preferred over the USM12 record.

The BLK line in this table is used by the system to define the actual preference. The outcome is that when the link in a BIB heading is changed from the unpreferred AUT library to the preferred AUT library, the unpreferred AUT record's 008/14-16 is changed to "bbb", effectively de-activating the record for future use.

PAS
This relation sets the list of libraries that appears in the Access Rights window when creating privileges for an operator. Each ADM library defines the libraries its users can modify. Only the libraries in the relation will be shown in the permissions list (GUI and WEB).

For example:

```
PAS USM50 USM01 USM10 USM60
```

If you have connected to the GUI with an operator whose Z66-USER-LIBRARY is USM50, this is the list of libraries that will be displayed in the Access Rights window.

Note: A single ADM environment does not have to determine the PAS relation. In this case all users will have in the user_library field (Z66) the value:
ADMIN, and every user will be able to see the privileges of all other users in the environment.

Note that up to 200 libraries can be defined per relation type. In addition, it is not permitted to break a relation type for a particular library into separate lines.

The total number of libraries that can be defined is 500.

Structure of the table:

Col. 1 Relation type

Col. 2 First library

Col. 3-6 Second to fifth libraries

Example of the table:

```
!1    2     3     4     5     6
!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!>
ADM USM01 USM50 USM51 USM52 USM53 USM54 USM55 UXP50
…
HOL USM01 USM60
HOL USM30 USM60
…
BIB USM50 USM01 USM30
BIB USM51 USM01
```

Note that if changes are made to the table, the GUI server must be killed and reactivated in order to see the changes reflected in the online modules.

license

Location of the table: alephe/tab directory

Purpose of the table: License parameters

Parameters pertaining to the institution's **ALEPH 500** license are defined here, as per the parameters issued by ***Ex Libris.***

marc_country_codes

Location of the table: alephe/tab directory

Purpose of the table: Country codes that may be used in MARC records

This table contains the list of country codes that may be used in MARC records in various places. The table may be used to validate input entered into fixed fields, for example,, the 008 field.

Structure of the table:

Col. 1 - 15

Country code – each column consists of up to three lowercase

characters

Example of the table:

```
!1   2   3   4   5   6   7   8   9   10  11  12  13  14  15
!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!
aa  abc ae  af  ag  ai  aj  aku alu am  an  ao  aq  aru as
at  au  aw  ay  azu ba  bb  bcc bd  be  bf  bg  bh  bi  bl
bm  bn  bo  bp  br  bs  bt  bu  bv  bw  bx  cau cb  cc  cd
ce  cf  cg  ch  ci  cj  ck  cl  cm  cou cq  cr  ctu cu  cv
cw  cx  cy  dcu deu dk  dm  dq  dr  ea  ec  eg  enk er  es
et  fa  fg  fi  fj  fk  flu fm  fp  fr  fs  ft  gau gb  gd
```

marc_exp.dat

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Subfields for Open Form and List of fields (F5)

Related table(s):

1) tab01.<lng>
2) formats.<lng>

This table sets the subfields that will display when *open form* (ctrl-F) and when selecting a field from the List of fields (F5) is invoked in the Cataloging client. The table should include only the fields for which a form screen has NOT been created. (Form screens are defined and located in the directory pc_tab/catalog of the library). Note that not ALL the subfields need be defined - only those most in use.

Structure of the table:

Col. 1 Tag

Col. 2 Indicators. Use wildcards (#) to indicate any indicator

Col. 3 Record format as registered in the FMT field in the document record. Use XX for all formats.

Col. 4 subfields

Example of the table:

```
!1  2  3            4
!!!-!!-!!-!!!!!!!!!!!!!!!!!!!!!!
013    XX abcdef68
015    XX a68
016 #  XX az28
017    XX ab68
018    XX a68
020    XX acz68
022 #  XX ayz68
024 ## MU acdz268
```

marc_language_codes

Location of the table: alephe/tab directory

Purpose of the table: Language codes that may be used in MARC records

This table contains the list of language codes that may be used in MARC records in various places. The table may be used to validate input entered into fixed fields, for example,, the 008 field.

The table is limited to 200 codes.

Structure of the table:

Col. 1 - 15

language code – each column consists of up to three lowercase characters

Example of the table:

```
!1   2   3   4   5   6   7   8   9   10  11  12  13  14  15
!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!
ace ach ada ady afa afh afr aka akk alb ale alg amc amh
ang apa ara arc arg arm arn arp art arw asm ast ath ava ave
awa aym aze bad bai bak bal bam ban baq bas bat bej
bel bem ben ber bho bik bin bis bla bnt bra bre btk
```

marc8_xxx_to_unicode

Location of the table: alephe/unicode directory

Purpose of the table: Character conversion tables

This is a generic description related to a number of tables that convert characters from the MARC-8 character set to Unicode. The following tables have been defined:

marc8_ara_to_unicode
marc8_eacc_to_unicode
marc8_ext_ara_to_unicode
marc8_ext_rus_to_unicode
marc8_greek_to_unicode
marc8_heb_to_unicode
marc8_lat_to_unicode
marc8_rus_to_unicode

When converting from UTF to MARC-8 the programs will take the first occurrence in the second column that matches the Unicode input character. It is possible to rearrange the tables so that the line with the value desired for export precedes other lines for the same Unicode character. MARC-8 values do not have to be sorted.

Structure of the table:

Col. 1 Hexadecimal value of the Unicode character

Col. 2 Hexadecimal value of the MARC-8 character

Col. 3 Comment from the source table

Example of the marc8_greek_to_unicode table:

```
037E 3F  GREEK QUESTION MARK
0391 41  GREEK CAPITAL LETTER ALPHA
0392 42  GREEK CAPITAL LETTER BETA
0393 44  GREEK CAPITAL LETTER GAMMA
0394 45  GREEK CAPITAL LETTER DELTA
0395 46  GREEK CAPITAL LETTER EPSILON
03DA 47  GREEK LETTER STIGMA
03DC 48  GREEK LETTER DIGAMMA
0396 49  GREEK CAPITAL LETTER ZETA
0397 4A  GREEK CAPITAL LETTER ETA
0398 4B  GREEK CAPITAL LETTER THETA
0399 4C  GREEK CAPITAL LETTER IOTA
039A 4D  GREEK CAPITAL LETTER KAPPA
039B 4E  GREEK CAPITAL LETTER LAMDA
```

no_utf_mac

Location of the table: alephe/tab directory

Purpose of the table: defines a list of browsers/versions (user agents) that do not fully support UTF-8.

The check of browser's UTF support is now table based. This table is defined for MAC browsers. The table contains a list of browsers/versions (user agents) that do not fully support UTF-8.

The main server checks the user agent of each request against the values in this table. If the current user agent is found in the table, then the browser is considered as not supporting UTF-8.

It is possible, therefore to configure the list of browsers that will be considered as "not supporting UTF-8" by adding the user agent to the table. It is assumed that a user agent that does not appear in the table, supports UTF-8.

no_utf_pc

Location of the table: alephe/tab directory

Purpose of the table: defines a list of browsers/versions (user agents) that do not fully support UTF-8.

The check of browser's UTF support is now table based. This table is defined for PC browsers. The table contains a list of browsers/versions (user agents) that do not fully support UTF-8.

The main server checks the user agent of each request against the values in this table. If the current user agent is found in the table, then the browser is considered as not supporting UTF-8.

It is possible, therefore to configure the list of browsers that will be considered as "not supporting UTF-8" by adding the user agent to the table. It is assumed that a user agent that does not appear in the table, supports UTF-8.

oaiconf.dtd

Location of the table: alephe/oai

Purpose of the table: DTD of the OAI Data Provider

Related table(s):

1. oaiconf.xml

oaiconf.xml

Location of the table: alephe/oai

Purpose of the table: Used for defining OAI Data Provider settings

Related table(s):

1. oaiconf.dtd

Structure of the table:
The table is in XML format. It defines the following:

1. Available sets (shown in ListSets response, can be used in ListRecords and ListIdentifiers requests)
2. timeZone (for example, –00.00, used in protocol response to specify responseDate)
3. repositoryName (used in Identify response)
4. baseURL (used in Identify response)
5. adminEmail (used in Identify response)

Exampel of table

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE oaiconf SYSTEM "oaiconf.dtd">
<oairoot>
<set>
  <setSpec>USM01</setSpec>
  <setName>USM01 Test library</setName>
</set>
<set>
  <setSpec>UNI01</setSpec>
```

```
  <setName>UNI01 Test library</setName>
</set>
<timeZone>-00:00</timeZone>
<repositoryName>ExLibris Israel</repositoryName>
<baseURL>http://ram43:8991/OAI-script</baseURL>
<adminEmail>Irina.Dijour@exlibris.co.il</adminEmail>
</oairoot>
```

oaipubconf.xml

Location of the table: alephe/oai

Purpose of the table: Used for defining Puiblishing based OAI Data Provider settings

Related table(s):

1. oaiconf.dtd
2. oaiconf.xml

Structure of table:

The `oaipubconf.xml` file contains the following elements:

- **<oairoot>** – The root element

- **<set>** – A repetitive element which contains the following sub-elements:
    - **<setSpec>** – Contains set specs which occurs in OAI requests/responses. This element is mandatory.

    - **<setName>** – Contains the set name. This element is mandatory

    - **<internalSet>** – Contains the name of the Publishing set from which the records are taken, followed by a colon, followed by format oai_dc or marc21.

      For each set at most 2 <internalSet> elements can be defined – one for oai_dc records and one for marc21 records. As OAI Data Provider must support oai_dc format, for each set a <internalSet> element for oai_dc records must be defined.

  For example, if the OAI Data Provider set EXAM is exposed in oai_dc format and in marc21 format then:

  - There are two published sets – EXAM_DC and EXAM_21. Both have the same records, with the only difference being that EXAM_DC is in OAI_DC_XML format and EXAM_21 in OAI_MARC21_XML format.
  - The <set> tag is configured as follows:

    <set>
    <setSpec>EXAM</setSpec>
    <setName>Example Set</setName>
    <internalSet>EXAM_DC:oai_dc</internalSet>
    <internalSet>EXAM_21:marc21</internalSet>

</set>

Note: The first <set> element listed in the file must contain the default set (a set containing all the records exposed by OAI Data Provider, in oai_dc format).

- **<repositoryName>** – Name of the repository. This element is mandatory.

- **<baseURL>** – URL used in OAI requests. It must have the following form:
  - *http://<ALEPH web server host>:<ALEPH apache port>/OAI.*

  This element is mandatory.

- **<adminEmail>** – Email of the repository administrator. This element is mandatory.

- **<description>** – Information about the repository. This element is mandatory. It must contain an <oai-identifier> element with a description of the item identifiers naming policy. It may also contain sub-elements with additional information about the repository.

The <oai-identifier> element must have the following format:

```
          <oai-identifier
          xmlns="http://www.openarchives.org/OAI/2.0/oai-identifier"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai-
identifier
             http://www.openarchives.org/OAI/2.0/oai-identifier.xsd">
        <scheme>oai</scheme>
        <repositoryIdentifier>il-aleph05</repositoryIdentifier>
        <delimiter>:</delimiter>
        <sampleIdentifier>oai:il-aleph05:USM01-
000000001</sampleIdentifier>
          </oai-identifier>
     </description>
```

The <repositoryIdentifier> element must contain the repository ID used as part of OAI identifiers. It must start with a Latin letter.

The <sampleIdenfitier> element must contain a sample identifier. The syntax must be:

oai:<repository id>:<physical library>-<doc number>

Example of the table:

```
<?xml version="1.0" encoding="UTF-8"?>
<oairoot>
<set>
  <setSpec>OAI-SET</setSpec>
  <setName>OAI SET</setName>
  <internalSet>OAISETD:oai_dc</internalSet>
  <internalSet>OAISETM:marc21</internalSet>
```

```
</set>
<set>
  <setSpec>HISTORY</setSpec>
  <setName>History</setName>
  <internalSet>HISTORYD:oai_dc</internalSet>
  <internalSet>HISTORYM:marc21</internalSet>
</set>
<set>
  <setSpec>SCIENCE</setSpec>
  <setName>Science</setName>
  <internalSet>SCIENCED:oai_dc</internalSet>
  <internalSet>SCIENCEM:marc21</internalSet>
</set>
<repositoryName>ExLibris Repository</repositoryName>
<baseURL>http://il-aleph05:8995/OAI</baseURL>
<adminEmail>Irina.Dijour@exlibris.co.il</adminEmail>
<description>
     <oai-identifier
         xmlns="http://www.openarchives.org/OAI/2.0/oai-identifier"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai-
identifier
            http://www.openarchives.org/OAI/2.0/oai-identifier.xsd">
        <scheme>oai</scheme>
        <repositoryIdentifier>il-aleph05</repositoryIdentifier>
        <delimiter>:</delimiter>
        <sampleIdentifier>
          oai:il-aleph05:USM01-000000001</sampleIdentifier>
      </oai-identifier>
</description>
</oairoot>
```

This example defines three sets: OAI-SET, SCIENCE, and HISTORY. All the sets expose data in oai_dc and marc21 format. The set OAI-SET is the default set. It contains all the records contained in sets SCIENCE and HISTORY. It can also contain additional records.

- When the user asks for the OAI-SET set (or does not specify a set), and the request is for the oai_dc format - records from the Publishing set OAISETD are exposed.
- When the user asks for the OAI-SET set (or does not specify set), and the request is for the marc21 format - records from the Publishing set OAISETM are exposed.
- When the user asks for the HISTORY set, and the request is for the oai_dc format - records from Publishing set HISTORYD are exposed.
- When the user asks for HISTORY set, and the request is for the marc21 format - records from Publishing set HISTORYM are exposed.

oclc_server_defaults

Location of the table: alephe/rootdirectory

Purpose of the table: Used for defining environment variables for the OCLC server

order_sta_update

Location of the table: tab directory of the ADM library

Purpose of the table: Defines values to be set in BIB STA$$a as a result of a first order creation and first order arrival. It works in conjunction with z105 message: UPDATE-ORDER - type 'k'.

Related table(s):

1) tab_z105 – UPDATE-ORDER action-message k

The table order_sta_update defines the values populated in STA$$a as a result of a first order creation and first order arrival. It works in conjunction with z105 message: UPDATE-ORDER - type 'k'.

Structure of the table:

```
COL  1.  The action that trigger the STA update:
          ORDER-CREATE or ORDER-ARRIVAL.
          ORDER-CREATE action can be set for all order types (column
3:
          M/S/O).
          ORDER-ARRIVAL action can be set for Monograph
           and Standing order (column 3: M/O).
 COL  2.  Order's  sublibrary/Order unit filter.
             Library that use 'ordering unit' (TAB100-USE-ORDER-
UNIT=Y)
             should set the Order Unit code. Otherwise, set Order's
             Sublibrary code. Set ##### for all units/sublibraries.
 COL  3.   Order Type filter: M/S/O/#
               M- Monograph, O-Standing order, S-Serils
               # - match all;
 COL  4.  The value to be set in BIB STA$$a field.
```

The following is an example of the table:

```
!     1              2     3    4
!!!!!!!!!!!!!!!!!!!-!!!!!-!-!!!!!!!!!!
ORDER-CREATE         ORDU1 M A2
ORDER-CREATE         ORDU1 O A3
ORDER-ARRIVAL        ORDU1 # C1
```

path_convert

Location of the table: tab directory of the library

Purpose of the table: Re-direction of directories for sharing between libraries

The path_convert table serves to re-direct directories and/or files and configuration tables from one path to another. This allows sharing setup tables between libraries.

The left column lists the path where the system expects to find the directory or file, and the right column lists the path that should be used instead.

Structure of the table:

Col. 1 Actual path

Col. 2 Redirect path

Example of the table:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!------!!!!!!!!!!!!!!!!…!!!>
$usm01_dev/usm01/www_s_eng            $alephe_root/www_s_eng
$usm01_dev/usm01/www_r_eng            $alephe_root/www_r_eng
$usm01_dev/usm01/www_u_eng            $alephe_root/www_u_eng
$usm01_dev/usm01/www_f_eng            $alephe_root/www_f_eng
$usm01_dev/usm01/www_c_eng            $alephe_root/www_c_eng
$usm01_dev/usm01/www_y_eng            $alephe_root/www_y_eng
$usm01_dev/usm01/www_q_eng            $alephe_root/www_q_eng
```

Note that path_convert under UTIL Y/6 has an important sub-menu option:

4. Check path convert of library file name

This enables the system to check path convert for files in a specific library or under $alephe_root.

pc_server_defaults

Location of the table: alephe_root directory

Purpose of the table: Default values for GUI applications

Related table(s):

1) setenv due_date_format works in combination with RECALL-METHOD in tab100. If this line is defined in aleph_start, there is no need to set it in www_server.conf or pc_server_defaults.

This file defines various default values for working with the GUI environment:
Note: If a change is made in this file, the server needs to be killed (UTIL W/2/4) and reactivated (UTIL W/3/3) in order to be able to see the changes online.

General note:
Note that there are some definitions that offer more than one "setenv" option. The non-active option does not have to be commented out – the system will always use the **last** line defined for that option.

Explanation of some of the elements in the file:

setenv PC_NUM_SERVERS 5
This determines how many backend servers are running

setenv acq_user_z71_sort_routine 00
setenv acq_user_z71_sort_order A
These parameters define the sort routine and the sort order for the Acquisitions Logger in the online client

Sort routines:

- 00-by the open date and hour of the transaction (the most common used sort)
- 01-by sequence no. of the log (mostly for conversions)
- 02-by sequence no. of the log, the logs with the Z71-ACTION-DATE will always be sorted on to by Z71-ACTION-DATE.

Sorting order:

- A – ascending
- D – descending

setenv borname_style
Setenv borname_style_u 02
setenv borname_style_s 01
setenv borname_style_f 03
Defines the username style for:

- U – user input
- S – Screen display
- F – Form

The numbers denote format

- 01 - name, title
- 02 - name
- 03 - title name
- default if not defined – name (02)

setenv bor_list_address 02,03,
Display of patrons in CIRC – defines which address lines will display.

- 02,03 will display patron name together with the second and third lines of the address (up to five lines can be defined – 02,03,04,05.) The first line is the patron name, copied automatically from the name field.
- 00 - name only with no address elements.

setenv call_no_support N

This line defines whether the system is using call numbers instead of barcodes for item identification for Circulation. Libraries that use this setup must have a unique Z30_CALL_NO value for each item, and must have an index on this field.

setenv call_number_type 7
This variable defines the call number type that is assigned to the temporary items that are created when an ILL borrowing request arrival is registered. If not defined, the default valus is set to “7”.

setenv check_url_timeout 30
check_doc_url in the table check_dochad no definitive timeout. This was necessary if a URL could not have been reached within the pc_server_timeout. In this case the record could not have been saved on the server. The timeout is now defined. The default is 10 seconds.

setenv circ_cataloger_request_date 001
Defines the number of days after the creation of a quick catalog record in CIRC that a hold request for patron CATALOGER will be created.

setenv circ_user_z36_sort_routine 01
setenv circ_user_z36_sort_order A
Defines the sorting of the patron’s loan list. Two routines are available:

- 00 - "standard" sort using the due-date
- 01 - sort using sublibrary, item-status, collection, due-date

Sort order can be

- (A)scending or
- (D)escending

If the standard sort is sufficient then these lines can be commented out.

setenv circ_user_z37_sort_routine 01
setenv circ_user_z37_sort_order A
Defines the sort of the patron’s hold list. Three routines are available:

- 00 - "standard" sort using priority, request date, open date and hour
- 01 - sort using sublib, item-status, collection, open date and hour
- 02 - sort using sublib, item-status, collection, status, open date and hour

Sort order can be

- (A)scending or
- (D)escending

If the standard sort is sufficient then these lines can be commented out.

setenv circ_user_z38_sort_routine 01
setenv circ_user_z38_sort_order A
As above, for photocopy requests

setenv course_request_days 20

Defines how long before a course starts should a hold request be effective.

The "request date" of a hold request created from Course Reading is set to default to *nn* days before the starting date of the course. This *nn* number of days is set in the course_request_days line. If there is no value in this variable, the number 14 is used as a default (the 'hold request date' is 14 days before the Course's open date).

setenv create_statistics Y
The Z34 records can be used for statistical analysis of server use. If set to ***Y*** the counter “last-z34-sequence” should be defined in the Z52 table (UTIL G/2).

setenv date_style_s 12
setenv date_style_f 12

Defines the date display for:

- S – Screen
- F – Form

The first digit of the number denotes the separator style. The second digit, the *date* style.
default: dd/mm/yy

First digit - separator
0 separator '-'
1 separator '/'
2 separator ' '

Second digit – date style
1 dmy
2 dmy w/century
3 dmy w/century, long month
4 dmy w/century, short month
5 mdy
6 mdy w/century
7 mdy w/century long month
8 mdy w/century short month

For example

date style 14 will display
25/Dec/2003

date style 25 will display
12 25 2003

First digit – no separator
no separator (not 0,1,2)

Second digit – date style
1 ddmmyy
2 mmddyy
3 yymmdd
4 ddmmyyyy

5 mmddyyyy
6 yyyymmdd

For example
date style 34 will display
25122000

setenv default_lock_period 300
Locked acquisition, item, ILL and circulation records are automatically unlocked after the period defined in this section .
The period is defined in seconds. By default (if not otherwise defined), the variable has been set to lock records for 300 seconds

setenv default_recall_type "01"
Defines the default recall type for the Create Hold Request in the Circulation interface. If a value is not found, the program defaults to "03".

setenv doc_lock_period 3600
Defines the period after which the documents which are locked by the cataloguing lock function will be automatically unlocked. The period is defined in seconds.

setenv due_date_format 1
This setenv controls what displays when there is just a single "Due date" column. It works in combination with switch RECALL-METHOD in tab100.

- **1 -** Display a single "effective due date", based on RECALL-METHOD in tab100.
- **2 -** Display the recall-due-date, if there is one (even if it is later); otherwise, display the regular due date.
- **3 -** Always display only the z36_due_date; (for recall methods 2 and 3 this would mean that the recall-due-date will never display).*
- **4 -** Display both: the regular due date and (if there is one) the recall-due-date.

* Though this is an option, it is not recommended.
NOTE: **value "1" is recommended**. If this line is set in aleph_start (and it must be set there for batch jobs), there is no need to set this line in www_server.conf or pc_server_defaults

setenv expand_tree_style 2
How to expand catalog navigation tree and overview trees :

- **1 -** Do not Expand, only the specific record node will be visible
- **2 -** Expand only record nodes (ADM,BIB and HOL)
- **3 -** Expand all tree nodes (ADM,BIB,HOL,Z30,Z68,Z16 ...)

Note that it is possible to expand ALL nodes of a record tree in the Overview Tree, by right-clicking the mouse and selecting "Expand All Nodes" from a pop-up menu.

In other words, no matter how the above variable is defined, all tree nodes can be expanded in the Overview Tree at any point, by right-clicking the mouse and selecting "Expand All Nodes".

external binding settings

setenv ext_bind_process_status SB
setenv ext_perform_loan Y
setenv ext_default_return_hour 2359

setenv ext_bind_ref_no_1 LCNum,010,a
setenv ext_bind_ref_no_2 ISBN,020,a
setenv ext_bind_ref_no_3 ISSN,022,a

Up to 3 additional fields are supported for including data from BIB for external binding. Each has 3 sections:

- Text description of the tag,
- the BIB tag,
- subfield

setenv file_lock_period 1800
This variable defines the lock period of files (tables) edited through ALEPHADM. It must be at least two minutes longer than the warning time out period defined in alephadm.ini (variable 'WarningTimeOut').The period is defined in seconds.
The lock period for a record is defined either in aleph_start or in pc_server_defaults. By default, the system will use the values listed above.

setenv ill_expected_return_date 028
This line is used for defaulting the due date when an item is loaned from the ILL service to an ILL partner.

setenv ill_return_for_user 003
The due date on an ILL item will be the "expected date of return" ***minus*** the number defined in this field.
Note in relation to this: The vendor form has two delay periods, one for arrival, and one for return. If the delay for return is 0, the delay will be taken from this variable.

setenv item_hol_tree_style 3
Item-HOL connection using the field Z30-HOL-DOC-NUMBER - How to show it in catalog navigation tree and overview trees:

- 1 - Do not show the connection between item and HOL record
- 2 - Show item under its ADM record and under the linked HOL
- 3 - Show item only under its linked HOL record and show HOL libraries before ADM libraries

setenv item_label_restore Y
When an item label is printed using the print label facility in the items module, item status, sublibrary, collection and location are automatically restored to original values, as kept in the Z30 field of the ADM record.

setenv item_object_tree_style
Defines how to expand object navigation and object overview (ADAM) :

1 - Show all objects under each ADM and do not show objects under the BIB
2 - Show all objects under each ADM but only if the Z403-SUBLIBRARY matches the ADM, or Z403-SUBLIBRARY is spaces. Do not show objects under the BIB.
3 - Show objects under each ADM but only if the Z403-SUBLIBRARY matches the ADM. "Open to all" objects should be under the BIB.
4 - Show all objects under the BIB.

setenv item_schedule_preview_period 00:30
Sets a preview period in order to see requests in the following time slot for short loan/advanced booking items. If an item is not currently requested and is available, the item will be loaned for the remainder of the current slot and for the next slot as well.

Note: that this environment is defined directly in $aleph_proc/sc_server for short term loans using Self-Check machines.

setenv item_schedule_release_period_60 00:30
setenv item_schedule_release_period_61 00:45
setenv item_schedule_release_period_all 01:00
No longer applicable. See tab_booking

setenv max_unit_price_diff_percent 1
In the Acquisitions module, when a new invoice is created, or an existing one is modified, and the actual unit price exceeds the estimated unit price by more than a defined percent, an alert message displays.
The maximum percent difference is determined by setting the value of the max_unit_price_diff_percent variable.
Note that the alert message is displayed only ONCE, and it does NOT prevent the invoice data from being saved in the database. That is, even if the alert message is discarded, and the "Line Item" window is closed via the "Cancel" button, the invoice record (Z75) is updated.

setenv modify_item_arrival Y
Defines whether the item-specific check-in screen displays for serial issue check-in. If set to N, only the general check-in screen displays.

setenv overdue_delinq_period 00
The number of overdue days after which the patron has global delinquency code **70** registered in the delinquency field of the global patron record. This happens automatically, therefore set to 00, comment out, or remove line if not required.

setenv pc_cat_hol_item_support
N - It is not possible to highlight the holdings library node in the record manager section of the GUI and choose "Record manager > Load Create record". When the parameter is "N" it will always give an "ADM" link even when the record is created from HOL library.

Y - When the cataloger highlights the holdings library node in the record manager section of the GUI and chooses "Record manager > Load Create record", a HOL type record is created.

setenv pc_sort_field "01 D02 A"
setenv pc_sort_field_aut "02 A03 A"
setenv pc_sort_field_sub "03 D02 A"
setenv pc_sort_field_shl "DISPLAY"

Sort options – see www_server.conf defaults for more detail
Note, however, that it is possible to write special value "DISPLAY" for a specific access code. This value means that the sort will be as it was displayed in the browse list, and not based on parameters defined in tab_sort.

setenv PC_SERVER_TIMEOUT
This is used to determine the max time in seconds that the server attempts to handle a single transaction. The value may be set up to 600 seconds. The default setting is 60.

setenv pc_transactions_log

- "Y" - a file pc_ser_<port> is created in $LOGDIR and transactions are written to this file.
- "N" - the file pc_ser_<port> is not created in $LOGDIR.

It is recommended in production versions to set this parameter to "N", to improve performance

setenv pc_tree_view_max_branch 10
setenv pc_filter_tree_view_max_branch 99
Defines the limit after which the display of the navigation tree in the online clients is truncated with an appropriate message. The range defined must be between 1 and 750. This means that the maximum number of leaves that can be displayed can be up to 750.
Trees are limited to a maximum of 800 lines.

setenv pc_z37_date_to M006
This line defines the default value for "to date" (date of last interest) for the Create Hold Request function in the Circulation interface. The format is <x><nnn>, where x can be D (days), M (months) or Y (years).

setenv remote_z30_lib_01 XXX50
setenv remote_z30_lib_02 YYY50
Defines the environment variables "remote_z30_lib_nn (01,02...)" for every ADM library that needs to be checked when the local ADM library fails to find an item (by barcode).

setenv security_path_p "$TMPDIR;$FILE_TMPDIR"
setenv security_path_g $TMPDIR;$FILE_TMPDIR…
Security path definitions for upload/download of files:
security_path_p – upload - put (used especially for alephadm)
security_path_g – download - get (used especially for printouts/temporary files).

setenv serial_bind_period 50
serial_bind_period - min no. days to bind non-checked-in issues. This relates to the 'Print Completed Volumes' option to be found the under Utilities menu in the online Items module.
This parameter defines the maximum number of days to wait for an item after its issue date (Z30-issue-date), before considering it 'unarrived' for the purpose of this report.

setenv set_default_z110_create_order D
This defines ascending or descending sort of system numbers for sets which are more than the defined sort limit

setenv set_prox_limit 100
Defines the limit of number of records for proximity searching

setenv set_result_set_limit
Even though the number of hits for a find command was unlimited, the actual records in a set was limited to 1000. It is now possible to change this limit in the range of 1000-20000

Please note that this will affect performance for large sets.

Also the Web screen short-head-2 should be changed.

From:
Records $0100 -$0200 of $0300 (maximum display and sort is 1000 records)

To:
Records $0100 -$0200 of $0300 (maximum display and sort is $4500 records)

setenv time_style
setenv time_style_s 04
setenv time_style_f 04
Time display for:

- S – Screen
- F – Form

Numbers denotes format
default: hhmm
01 hh:mm
02 hh-mm
03 hh mm
04 hh:mm AM/PM
05 hh-mm AM/PM
06 hh mm AM/PM
07 hhmm AM/PM
Date and Time formatting can be set in **ALEPH** for screen display and for forms. The "setting" for the user input for the GUI clients can be done on the client itself by activating the F9 key.

Note:
setenv pc_sort_limit

setenv set_result_set_limit

are no longer in use. The set is limited to 1000 results and all the results are sorted. If the query found more than 1000 records, only the first 1000 will be shown.

pc_tab_acq_fast_cat.<lng>

Location of the table: tab directory of the **BIB** and **ADM** libraries

Purpose of the table: Fields for cataloging in Acquisitions

Related table(s):

1) tab01.<lng>

This table defines fields for the quick cataloging option in the Acquisitions GUI module.
Quick cataloguing is functional in the library defined in the acq.ini file on the client:

[FastCatDlg]
Library=<library code – for example, USM01>

In the BIB library, the table should contain the bibliographic fields.

In the ADM library, the table should contain the administrative fields that will be included in the ADM record.

Note: The total number of lines for the BIB and ADM tables must not exceed 10
Note: In the ADM library, if no fields are necessary, the table should be left empty but it MUST be present.

Structure of the table:

Col. 1 indicates whether the field is

Y = the field is optional.

N = the field is **mandatory**

H – the field is hidden, but its data is written in the record. The field and the text in col. 4 are always written on the record;

N = the field is **mandatory**

G = At least one of the fields, that are marked with G, is mandatory when fast cataloging is performed.

H – the field is hidden, but its data is written in the record. The field and the text in col. 4 are always written on the record;

Col. 2 field code.

Col. 3 subfield.

Col. 4 Field name or text of hidden field

Example of the table:

```
Y 100   a Author
N 245   a Title (mandatory)
Y 260   a Place
Y 260   c Date
Y 022   c ISSN
G 035   a Sys Control No (Group)
G 020   a ISBN (Group)
G 022   a ISSN (Group)
```

Col. 2 field code.

Col. 3 subfield.

Col. 4 Field name or text of hidden field

Example of the table:

```
!   2   3           4
!-!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Y 100   a Author
N 245   a Title (mandatory)
Y 260   a Place
Y 260   b Publisher
Y 260   c Date
Y 020   a ISBN
Y 245   b Additional name
Y 245   d Dates
Y 245   c Optional name
```

pc_tab_cat_conv

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: Remote conversion programs

The standard conversion programs, performed when the Import Records function of the Cataloging module is used, are usually based on a program that is executed from the client. In addition it is also possible to convert a file of selected records by sending it to the server - where the conversion is performed - instead of converting the records at the level of the PC.
This table is used to define the remote conversion programs

For remote conversions, note that the conversion line in under the [ConvertFile] section of the convert.ini file must be defined as explained below.

ConvertN=TextALPHA,Text,ProgramPath,[Parameter1,Parameter2,ParameterN]

For server conversions:
ProgramPath: Must be REMOTE
Parameter1 : Must be a routine that matches column 1 of this table.
Note that following routines can be used for the conversion of CDMARC and MAB2-Diskettenformat records:

MAB pc_cat_conv_mab_d 850_TO_UTF,MAB
CDMARC pc_cat_conv_cdmarc 8859_1_TO_UTF

Structure of the table:

Col. 1 Routine name. This is the identifier of the conversion that is being performed (free-text). It matches Parameter1 in the conversion lines for remote conversions in the convert.ini file

Col. 2 Program name - Enter the conversion program that should be performed for the specific conversion routine defined in column 1

Col. 3 Program arguments - Certain conversion routines require additional information, such as character conversion routines. This column is used to define additional parameters for conversion programs.

Example of the table:

```
! 1                2                                3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
MAB         pc_cat_conv_mab_d              850_TO_UTF
!CDMARC      pc_cat_conv_cdmarc             8859_1_TO_UTF
CDMARC      pc_cat_conv_cdmarc
MARC        pc_cat_conv_marc
MARC_FRNCH pc_cat_conv_marc                DIA_TO_UNICODE2
SEQ         pc_cat_conv_aleph_seq
SEQ_8859_1 pc_cat_conv_aleph_seq           8859_1_TO_UTF
SEQ300      pc_cat_conv_aleph300_seq       ALEPH300_TO_UTF
```

pc_tab_circ_fast_cat.<lng>

Location of the table: tab directory of the **BIB** and **ADM** libraries

Purpose of the table: Fields for cataloging in Circulation

Related table(s):

1) tab01.<lng>

This table defines fields for the quick cataloging option in the Circulation GUI module.
Quick cataloguing is functional in the library defined in the circ.ini file on the client:

[FastCatalog]
Library=<library code – for example, USM01>

The tables is should appear on both the ADM and the BIB libraries.

In the BIB library, the table should contain the bibliographic fields.

In the ADM library, the table should contain the administrative fields that will be included in the ADM record.

Note: The total number of lines for the BIB and ADM tables must not exceed 10

Note: In the ADM library, if no fields are necessary, the table should be left empty but it MUST be present.

Structure of the table:

Col. 1 indicates whether the field is

Y = the field is optional.

N = the field is **mandatory**

G = At least one of the fields, that are marked with G, is mandatory when fast cataloging is performed.

H – the field is hidden, but its data is written in the record. The field and the text in col. 4 are always written on the record;

Col. 2 field code.

Col. 3 subfield.

Col. 4 Field name or text of hidden field

Example of the table:

```
Y 100   a Author
N 245   a Title (mandatory)
Y 260   a Place
Y 260   c Date
Y 022   c ISSN
G 035   a Sys Control No (Group)
G 020   a ISBN (Group)
G 022   a ISSN (Group)
```

Note: If an application does not want to "dirty" its database, non-standard fields can be used (for example, AU, TL, and so on.), which can then be indexed separately. When the item is created (after registering the BIB information) a hold is automatically placed for a patron CATALOGER (which therefore must be defined in the library's patron file).
In pc_server_defaults the parameter:

setenv circ_cataloger_request_date

defines the number of days after the creation of a quick catalog record in CIRC that a hold request for user CATALOGER will be created.

pc_tab_col.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definition of columns in GUI screens

This table defines the columns of information that will display in List windows in the GUI clients. A unique identifier defines every display list (for example, PC_ITEM_ITEM) and each column is assigned a column number. These two elements must **NOT** be changed..

The display order of columns is set by the order of the lines within the display list.

The columns of display are defined in percentages. The percentages of the columns to be displayed should not exceed 100.
Set the percentage column to 000 for non-display of a column.

Note that SOME list window columns are NOT controlled by this table. They are controlled by the /alephcom/tab/lng /tab_col table on the GUI client.

Example of the table in /alephcom/tab/<lng>

```
CONNECT             #L Library              01 100 01 C04     Library text
LOCATE_LIST         #L Database             01 070 01 C04     Database text
LOCATE_LIST         #L No Records           02 030 01 C04     No. records
ACQ_BUDGET_YEAR     #L Year to Invoice      01 100 01 C04     Year to invoice
CAT_EMPTY_MESSAGE   #L No Information       01 100 01 C01     Empty message cap
CAT_CATALOGER_LIST  #L Cataloger            01 030 01 C04     Cataloger
CAT_CATALOGER_LIST  #L Level                02 020 01 C04     Level
CAT_CATALOGER_LIST  #L Date                 03 025 01 C04     Date
```

Structure of the table:

Col. 1 ID of list. The first part of name usually points to the application - for example, PC_ITEM, PC_CASH, and so on.

Col. 2 ALPHA

Col. 3 column header

Col. 4 column number. This number is linked to a defined element in the list, and CANNOT be changed. For for example, in PC_ITEM_ITEM, Collection will ALWAYS be 06, Barcode will ALWAYS be 04.

In order to change the display sequence of the columns change the order of the lines.

Col. 5 Percentage of screen for column display. Defined columns and their percentages should total 100%.

Enter 000 if you do not want a column to display.

Col. 6 Font code. The value of the font code is set in the client's Alephcom\tab\font.ini file, in the last two characters of the "ListBox" definition. (for example, ListBox01.) The client can be forced to ignore the pc_tab_col setup, by using ListBox##. **Note:** When a bitmap is used (for example, checkmark) the characters ***it*** are entered here.

Col. 7 Color code. The color value is defined in the client's Alephcom\tab\alephcom.ini file, in the [TextListBox] section. **Note:** When a bitmap is used (for example, checkmark) the characters ***it*** are entered here

Col. 8 Font2 – this is for font differentiation in a list. Relevant only in instances where the line has a system differentiation, for example, arrived/non-arrived issue.

Col. 9 Color2 - this is for color differentiation in a list. Relevant only in instances where the line has a system differentiation, for example, arrived/non-arrived issue.

Col. 10 Explanatory note, comment

NOTE In the online GUI applications, right-clicking on any headings column lists the columns that can be displayed for that window. The user can then choose to select, or deselect column options, or to take the system default. The user setup is saved on the client in the new alephcom/tab/UserTabCol.ini file.

Note too that you can display the "Link" field (title hard-coded) at the bottom of the callout bubble invoked by right-clicking on a line in a GUI list.

The "Link" field contains the name of the section in pc_tab_col which represents the list (for example, PC_CASH_SUMMARY), together with various record keys and other pieces of information which are used by programs to identify a line uniquely and to process its data.

A flag in .\Alephcom\Tab\Alephcom.ini (under the [TextListBox] section) determines whether or not the "Link" field will be displayed at the bottom of the callout bubble or not:

ShowLineLinkInBubble=Y

The flag's value can be easily changed online via a check box entitled "Display link field at bottom of callout bubble"; it is located in the "General Configuration" tab (**ALEPH** menu/Options/Customize...).

The following is an example of a "Link" field:

Link [PC_ACQ_LIST] "000001650000152003049 010 NNNM00005"

In the example above, the section in pc_tab_col is enclosed in square brackets, and the rest of the data is enclosed in double quotes. In this case, the data is composed of the following parts:

Z68-REC-KEY: 00000165000015
Z68-ORDER-NUMBER: 2003049
Z68-VENDOR-CODE: 010
Z68-INVOICE-STATUS = "D" ? (Y/N): N
Z68-ARRIVAL-STATUS = "C" ? (Y/N): N
Z68-ORDER-STATUS = "CLS" ? (Y/N): N
Z68-ORDER-TYPE: M
Z68-NO-UNITS: 00005

Example of the table:

```
!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!-!!!-!!-!!!-!!-!!!-!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!
TREE_ERROR_LIST      L                       01 020 it map        error message l
ist
TREE_ERROR_LIST      L Message               02 080 01 C01        error message l
ist
!*
!*   Common - Item circulation History - Hold Request List
!*
PC_COM_HOLD_H_LIST   L Patron ID             01 015 04 C04        Patron ID
PC_COM_HOLD_H_LIST   L Patron Name           02 025 03 C03        Patron Name
PC_COM_HOLD_H_LIST   L Pickup                03 000 04 C04        Pickup location
PC_COM_HOLD_H_LIST   L Req Date              04 020 05 C01        Request Date
PC_COM_HOLD_H_LIST   L End Req               05 020 06 C06        End Request Dat
E
```

Note:

```
PC_SERIAL_ITEM_L     L Loan  01 003 01 C01 06 C02    Out on loan?
```

If this line is defined as a display line (item information in the Acquisitions/Serials client), issues out on loan will display with an asterisk (*).

Note:

```
PC_CIR_RR_LOAN_LIST  L Due Hour              10 000 03 C03   Due Hour
PC_CIR_RR_LOAN_LIST  L RR                    06 005 it map
```

In some parts of the system, items might be marked with ✓. This is defined with the terms ***it map*** in the font and color columns as can be seen in the above example.

Note:

```
PC_CIR_LOAN_LINE     L Status                04 015 01 C03        Item Status
```

Allows for the display of item status ***text*** in the lower part of the loan window.

pc_tab_crs_fast_cat.<lng>

Location of the table: tab directory of the ADM libraries

Purpose of the table: Fields for fast cataloging in Course Reading in Circulation GUI

Related table(s):

1) tab01.<lng>

This table sets the fields for the BIB and ADM records created when using "Catalog New" action in Course Reading in the Circulation module.
The bibliographic record is added to the Course Reading library of the ADM library, defined in library_relation.

The total number of lines defined in this table must not exceed 10.

Structure of the table:

Col. 1 indicates the field status

Y = the field is optional
N = the field is mandatory
G = At least one of the fields, that are marked with G, is mandatory when fast cataloging is performed.
H = the field is hidden. The field content is taken from this table, and not input in the form.The field code, field name and subfield code display on the input form.

Col. 2 field code.

Col. 3 subfield.

Col. 4 Field name or text of hidden field

Col 5. indicates whether the line is a BIB tag.
B = The tag is a BIB tag of the Course Reading XXX30 record.
If nothing is defined in col.5; the tag is an ADM tag.

Example of the table:

```
!   2   3           4                       5
!-!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!
Y 090   a Call number                       B
Y 1001  a Personal Author                   B
```

pc_tab_exp_field.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Field values for the GUI pull-down menus

Related table(s):

1) pc_tab_exp_field_extended.<lng> for sublibrary-sensitive field values
2) Values in pc_tab_exp_field.<lng> must match values defined in the following tables:.
    a. allowed_languages,
    b. tab_acq_index,
    c. tab_binding
    d. tab_z30_sort,
    e. tab15.<lng>,
    f. tab31.

The table defines the field values for the pull-down menu options in the different GUI applications.

Structure of the table:

Col. 1 Internal command

Col. 2 ALPHA

Col. 3 Description (text as will display in the pull-down menu) - (up to 50 characters)

Col. 4 Link to values as defined in the different tables (up to 100 characters)

Example of the table:

```
!        1               2      3                                                 4
!!!!!!!!!!!!!!!!!!!!-----!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>

SERIAL-SORT-TYPE         L Sublibrary/Item seq.                                SE
RIAL-0
SERIAL-SORT-TYPE         L Year/Vol./Issue/Part Ascending                       SE
RIAL-1
SERIAL-SORT-TYPE         L Year/Vol./Part/Issue Ascending                       SE
RIAL-2
SERIAL-SORT-TYPE         L HOL no./Year/Vol./Issue/Part Ascending               SE
RIAL-3
…
ITEM-PROCESS-STATUS      L Not in process
ITEM-PROCESS-STATUS      L Order initiation                                     OI
ITEM-PROCESS-STATUS      L On order                                             OR
ITEM-PROCESS-STATUS      L In process                                           IP
ITEM-PROCESS-STATUS      L Cataloging                                           CT
,,,
CIRC-LEVEL               L Beginner                                             01
CIRC-LEVEL               L Intermediate                                         02
CIRC-LEVEL               L Supervisor                                           20
```

Note:
In the online Acquisitions General invoice form, the default invoice type and status were hard coded to "REG" in previous versions. This has been changed and will now default to the first defined entry in the relevant section of pc_tab_exp_field.<lng>

```
ACQ_INVOICE_TYPE         L Regular                  REG
ACQ_INVOICE_TYPE         L Proforma                 PR0
ACQ_INVOICE_TYPE         L Deposit                  DEP
ACQ_INVOICE_TYPE         L Prepaid                  PRE
```

Note:
When printing from catalog it is possible to print server defined printouts using the expand list CATALOG-PRINT in pc_tab_exp_field.LNG.

The configurable printouts type code is: FORM
It is defined in the following manner:

FORM,
format no,

Example:

```
CATALOG-PRINT            L MARC tags                                  FORM,001
CATALOG-PRINT            L Name tags                                  FORM,002
CATALOG-PRINT            L MARC Public View                           FORM,099
CATALOG-PRINT            L Catalog card                               FORM,037
CATALOG-PRINT            L Citation                                   FORM,038
```

The expand list also contains a line with the printout code INTERNAL.
This means the printout of the catalog record as was existing in previous versions.

pc_tab_exp_field_extended.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Sublibrary-sensitive field values for the GUI pull-down menus

Related table(s):

1) pc_tab_exp_field.<lng> for general, rather than sublibrary-specific definitions

This table contains entries that were moved from pc_tab_exp_field.<lng> in order to allow for sublibrary sensitivity.

Only the fields with the following internal strings can be used with pc_tab_exp_field_extended.<lng>:
BOR-STATUS
BOR-TYPE
INVENTORY-NUMBER
ITEM-LOCATION
ITEM-LOCATION-2
LOCATION-GR
LOCATION-USIG
LOCATION-PREFIX
LOCATION-SOURCE
LOCATION-SUFFIX
Structure of the table:

Col. 1 Internal command
Col. 2 Sublibrary
Col. 3 Text
Col. 4 Description
Col. 5 Code - Up to 100 characters

Example of the table:

```
BOR-STATUS          ##### L Undergrad
01
BOR-STATUS          ##### L Graduate
02
BOR-STATUS          ##### L Academic Staff
03
BOR-STATUS          ##### L Visitor
04
BOR-STATUS          ##### L Other institutions
05
BOR-STATUS          ##### L Administrative Staff
```

pc_tab_fast_circ

Location of the table: tab directory of the **ADM** library

Purpose of the table: Parameters for fast circulation

This table defines the parameters for fast circulation. It defines which system checks should be activated, and whether the trap can be overridden.
Fast circulation will be used most often by distant branches with poor communication lines, or by book buses connecting to the main library via a modem.

Loan options
ON_HOLD checks if item is held for another patron
BOR_ITEM checks patron/item limits

Return options
OVERDUE checks if item is late
RECALL checks if item has been recalled

Structure of the table:

Col. 1 Name of the program [up to 10 characters]
Col. 2 Type
L – Loan
R-Return
Col. 3 Check (Y/N)
Col. 4 Overridable (Y/N)

Example of the table:

```
!   1       2 3 4
!!!!!!!!!!-!-!-!
ON_HOLD     L Y N
BOR_ITEM    L Y N

OVERDUE     R Y N
RECALL      R Y N
```

pc_tab_find.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Fields for searching for a record in the ILL module

Related table(s):

1) – tab00.<lng> Word index definitions

This table allows the library to define by which fields the database can be keyword searched from within the ILL module.

Structure of the table:

Col. 1 Find code

Col. 2 Find Code Text

A maximum of 8 codes can be defined.

Example of the table:

```
! 1                          2
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

ADM   Administrative Record
BIB   Bibliographic Record
ISBN  ISBN
ISSN  ISSN
WTI   Title
WAU   Author
WYR   Year of publication
SICI  SICI
```

Note the option of searching for an item by SICI number.
When searching by the SICI number, the system looks for all BIB records that match the ISSN part of the SICI number.
For every matched BIB record, it will then look for items in the ADM library that match the filter fields (Chronology and enumeration).
Finally there will be a display of all the BIB records that have matched items. If there is only one BIB record, it will highlight the matched item, from the list.

pc_tab_find_course.<lng>

Location of the table: tab directory of the ADM libraries

Purpose of the table: Define the configuration for the Search form of GUI-Circulation-Course Reserve (searching for a bibliographic record in order to add it to a Course Reading List)

This table defines the find parameters for the SEARCH function in the Course Reading module. One set of parameters is used in common for both the main library catalog and the Course Reading database.

The total number of lines defined in this table must not exceed 8.

Structure of the table:

Col 1 - Find code

Col 2 - Find Code Text

Example of the table:

```
! 1   2
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
BIB   BIB System Number
BAR   Barcode
WTI   Title Keywords
WAU   Author Keywords
```

pc_tab_scan.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Fields for browsing for a record in the ILL module

Related table(s):

1) – tab00.<lng>Access definitions

Structure of the table:

Col. 1 Scan code

Col. 2 Scan Code Text

A maximum of 8 codes can be defined.

Example of the table:

```
! 1                        2
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!

AUT   Author
TIT   Title
WAU   Word from Author
WTI   Word from Title
SYS   System number
ISBN  ISBN
ISSN  ISSN
SRS   Series
```

pc_tab_sear.<lng>

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: Definition of GUI search options

Related table(s):

1) Col. 2 - Base code as defined in col. 1 of tab_base.<lng>
2) Col. 5 - according to code defined in col. 1:
    a. SC and FI in tab00.<lng>
    b. FO and PF in edit_doc.<lng>,
    c. SO in tab_sort.

The table is structured as follows:

Col.1 **SC** – SCAN

FI – FIND

FO - DISPLAY FORMAT

SO – SORT

PF - PRINT/SAVE formats

FF - FIXED FIELD - For Fixed Field Search..

If no such lines are defined for a base, the "Anywhere in the record" field will appear.

LN Record language

Col. 2 Base code as defined in col. 1 of tab_base.<lng>

Col. 3 ALPHA code

Col. 4 Heading displayed

Col. 5 According to code defined in col. 1:

SC - link to the ACCess code as defined in tab00.<lng> in the tab directory of the library

FI - link to the WORD files in as defined in tab00.<lng> in the tab directory of the library

FO - link to edit_doc.<lng> format (must be expressed in 3 digits).

SO - link to field tag codes as defined in tab_sort table in the tab directory of the library

PF - link to edit_doc format for print/save formats

FF – For the ‘fixed field’ search options in GUI Search

LN – Record language

```
!           2            3              4                      5
!!--!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!
SC  USM01                L Titles                            TIT
SC  USM01                L Authors                           AUT
SC  USM01                L Authors + Titles                  AWT
SC  USM01                L Series                            SRS
SC  USM01                L Subjects (LC)                     SUL
SC  USM01                L Subjects (MeSH)                   SUM
SC  USM01                L Subjects (all)                    SUB
…
SC  MUSIC                L Titles                            TIT
SC  MUSIC                L Main Entry                        AUT
SC  MUSIC                L Performers                        PRF
SC  MUSIC                L Librettists                       LBT
…
FI  MUSIC                L Anywhere in the record            WRD
FI  MUSIC                L Titles                            WTI
FI  MUSIC                L Main entry                        WAU
```

pc_tab_short.<lng>

Location of the table: tab directory of the library

Purpose of the table: Display of the 'Brief format' of a record in the Search function

Related table(s):

1) tab_expand – creating a virtual field using various expand routines

2) edit_field.<lng> - line S
3) tab_base_count, col. 4 and 5 for codes BASE1 and BASE2 (defined in col. 4)
4) tab01.<lng>

This table defines the columns displayed in the brief list format in the Search function available in the GUI modules.

Each column can contain either bibliographic or system information (defined in col.4).
Columns that contain bibliographic information are defined by field tag. The tag can also be a virtual field created by an expand routine. The content is formatted using edit_field.<lng> (line ID in edit_field is 'S').

Columns can be defined for system information for the following:
LOC+, **PST** and **PSTS** can be used in order to display up to three locations in the brief display. The locations displayed are separated from each other using site-defined line 0002 in the aleph/error_<lng>/sear_brief_edit table.

- **LOC+** - displays the LOC field (created by

expand_doc_bib_loc_disp and using edit_field), and a link to relevant holdings. This is limited to three locations --- otherwise displays $aleph_root/error_lng/sear_brief_edit: line 0001

- **PSTS** - displays the PSTS field (created by expand_doc_bib_psts_disp and using edit_field), and a link to relevant holdings. This is limited to three locations --- otherwise displays alephe/error_lng/sear_brief_edit: line 0001
- **PST** - like PSTS, for the PST field created by the expand_doc_bib_loc_[n] and expand_doc_sort_x group of programs. Refer to **ALEPH** System Librarian's Guide/ Indexing section / Expand routines, tables and Expanded fields.

BASE1/BASE2 - can be used to display whether the record is present in a particular logical base. The logical bases for each column are configured in $alephe_tab/tab_base_count, col. 4 and 5.
LOW - is a site-specific option, and is used to indicate that the bibliographic record includes an "own" field for the library of the connected user.
Example of the sear_brief_edit table:

```
!!!!-!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
0001 0000 L More locations...
0002 0000 L --
```

Structure of the table:

Col.1 Format:

"00" is used except in special cases

Col.2 Column heading

Col. 3 Column number

Col. 4 Field code and subfield code.

Code **SUM** will number the documents in the list and their relative positioning in the list.

Codes **BASE1/BASE2** will display whether the record is present in a particular logical base.

Code **LOW** indicates that the bibliographic record includes an

"own" field for the library of the connected user.

Col. 5 1st alternative field and subfield

Col. 6-8

alternative fields and subfields.

Col. 9 Percentage of screen display. The total length of the line is 100%, which is divided among the different fields displayed on the line.

Col. 10 Font (font number as defined on the PC in the alephcom/tab/font.ini file.) The relevant lines in font.ini start with "ListBox…"

Col. 11 Color (color number as defined on the PC in the alephcom/tab/alephcom.ini file.)

Col. 12 For fixed fields, starting position for text extraction.

Note: The count is from base 001. When determining character position, if the fixed field begins with a subfield code (as in UNIMARC field 100), the calculation does NOT have to take the three positions of the subfield into account.

For example, for:

MARC 21 field 008, date-1 is 008 (listed as 07-10 in the MARC 21 manual).

UNIMARC field 100, date-1 is 010 (listed in the UNIMARC manual as 09-12).

Col. 13 For fixed field, number of characters for display

Col. 14 Language code can be entered here. It will act as a filter on the field, if the field has $$9 with a language code.

Example of the table:

```
!1     2                3   4       5       6       7       8    9   10  11 12  13
14
!!-!!!!!!!!!!!!!!!!!!!-!-!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!-!!-!!!-!!!-!!!-
!!!
00 Doc no.              1 SUM                                    010 01 C08
00 Call No              2 050##  LOC##                           015 01 C01
00 Author               3 100##a 700##a                          025 02 C02
00 Title                4 245##  240##                           030 03 C03
00 Year                 5 008    260##c                          010 03 C03 008 004
00 $1                   6 BASE1                                  005 01 C01
00 $2                   7 BASE2                                  005
```

pc_tree_c1501

Location of the table: aleph/error_<lng>

Purpose of the table: Messages related to the Navigation Windows in the online **ALEPH** modules

Note: In version 17.0 the directory error_<lng> was moved from alephe/error_<lng> to aleph/error_<lng>.

Structure of the table:

Col. 1 Error code
Col. 2 Error position
Col. 3 ALPHA
Col. 4 Error message

Example of the table:

```
0010 0000 L Cannot find ADM record
0011 0000 L Cannot find BIB record
1001 0000 L $1 - $2
1002 0000 L ITEMS
1003 0000 L ORDERS
1004 0000 L Barcode - $1
1005 0000 L Order number - $1
1006 0000 L HOL - $1
1007 0000 L LOANS
…
```

permission.dat

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Allowed and denied tags for users in cataloging

Related table(s):

1) tab01.<lng>

This table defines which tags can/cannot be edited and by whom.
The # sign can be used for any portion of the tag-indicator combination.

NOTE that each user that has cataloging authorization **must** also be listed here in order to be able to catalog.

A library that does NOT want this control should delete this table, in which case all users that have cataloging authorization can use all cataloging fields.

Denied tags will display in gray (or as defined in the catalog.ini variables):
- DeniedFieldColor,
- DeniedFieldBackGroundColor.
The Cataloging GUI's status bar will note that the tag is denied as well as the usual tag definition.
Actions such as editing, delete, open subfield, open new tag and so on. will not be available for denied tags.

The entries are automatically added when the cataloging tables are repacked (UTIL M/7).

Structure of the table:

Col. 1 User name

Col. 2 Tag code (use # for wildcards)

Col. 3 Permission

Col. 3 Permission

- Y=allowed
- N=denied

Col. 4 Internal. This column is used only by the system in intermediate files.

Example of the table

```
!1          2     3 4
!!!!!!!!!!-!!!!!-!-!!!!!!!!!!
URI         ##### Y
URI         245## N
JUDY50      ##### Y
SHIRLY50    ##### Y
```

pqf.properties

Location of the table: alephe/sru_server directory

Purpose of the table: defines the translation of CQL queries received by the `sru_server` to Z39.50 RPN queries sent to the z39_server

Related table(s):

1. Config.xml

Structure of the table:

The table consists of the following parts:

- Definition of context sets supported by the `sru_server`
- Mapping Z39.50 Attributes to CQL Patterns

Context Sets Definitions
Context sets define indexes, relations, and other CQL expression components. The components of a CQL query may take special meanings, depending on the context set they are related to, as explained below

For each context set, a line should be added with the following syntax:

set.<prefix> = <set specification>
For example:
set.cql = info:srw/cql-context-set/1/cql-v1.1
set.dc = info:srw/cql-context-set/1/dc-v1.1

set.bath = http://zing.z3950.org/cql/bath/2.0/

Note:
set.cql must be defined; each set prefix used in the translations below must belong to one of the defined context sets.

Mapping Z39.50 Attributes To CQL Patterns
A Z39.50 query consists of a term and optional attributes. There are 6 attributes types:

Attribute Name	Attribute Code
Use	1
Relation	2
Position	3
Structure	4
Truncation	5
Completeness	6

Each component of the CQL query (such as index and relation) is translated to one or more Z39.50 attribute.

The translation line has the following syntax:

CQL pattern = Z39.50 Attribute Pair

The following CQL patterns can be translated:

- Index
- Relation
- Position of anchor sign (^) in the term
- Position of wildcard character(*) in the term

Each attribute pair has the form `type = value`

- Type is one of 6 Z39.50 attribute types: 1, 2, 3, 4, 5 or 6
- Value is attribute value

If the Z39.50 attribute pair is not defined the CQL pattern (for example, CQL index) is recognized, but no Z39.50 attributes are added to the Z39.50 query.
For example:

index.rec.id = 1=12
index.dc.title = 1=4
index.dc.subject = 1=21

relation.eq = 2=3
relation.scr = 2=3

truncation.right = 5=1
truncation.left = 5=2

Translation of CQL Index to Z39.50 Attributes

A line must be defined for each recognized CQL index. The syntax is:

```
index.<set>.<name> = <z39.50 attributes list>
```

- **Set** – The context set defined in the context sets definition part described above.
- **Name** – The CQL index name.

For example:

```
index.dc.title =  1=4
```

In this example, if the `sru_server` receives the query `dc.title=history` it is translated to the Z39.50 term `history` with the Z39.50 `Use` attribute set to 4 (title).

Note:

- The `index.cql.serverChoice` denotes the default attribute pair that is used when no index has been defined in the CQL query. Therefore, it must be defined. For example, consider a setup such as:

  ```
  index.cql.serverChoice = 1=1016.
  ```

  In this setup, if the `sru_server` receives a query such as:

  ```
  query=history
  ```

  the query is translated to the Z39.50 term `history`, with the Z39.50 `Use` attribute set to 1016 (any).

- If the CQL index should not be translated to a `Use` attribute then the Z39.50 attribute pair part should be left empty.

  For example, consider a setup such as
  `index.dc.title = .`

  In this setup, if the `sru_server` receives a query such as:
  `dc.title=history`

  it is translated to the Z39.50 term `history`. Z39.50 `Use` attribute is not set.

Translation of CQL Relation to Z39.50 Attributes

CQL relations supported by the `sru_server` are translated to Z39.50 attributes by using the relation keyword. The syntax is as follows:

```
relation.<relation name> = <z39.50 attributes list>
```

For example:

- `relation.<  =  2=1`
  Relation < (less than) is translated to relation attribute 1.
- `relation.le  =  2=2`

Relation <= (less than or equal) is translated to relation attribute 2.

- `relation.eq = 2=3`
  Relation = (equal) is translated to relation attribute 3.

- `relation.ge = 2=4`
  Relation >= (greater than or equal) is translated to the relation attribute 4
- `relation.> = 2=5`
  Relation > (greater than) is translated to the relation attribute 5

Note:

- A special pattern `relation.*` can be used to translate unmatched relations. For example, if there is a definition *`relation.* = 2=3`*, then any unmatched relation will be translated to relation attribute 3.

- A special pattern `relation.scr` must be defined. This relation sets the Z39.50 relation attributes that are used for queries which do not contain a relation. It is recommended to define it as `relation.scr = 2=3`.
- A structure attribute may be defined using *`relation`* pattern or using *`structure`* pattern. A line such as the following must be defined:

  ```
  structure.* = [<Z39.50 attribute list>]
  ```

  For example:

  ```
  structure.* = 4=1
  ```

- Relation modifiers can be translated using the `relationModifier.<mod>` keyword. The syntax is `relationModifier. <mod> = <z359.50 attributes list>`.

Translation of CQL Anchor Sign to Z39.50 Attributes

In CQL the anchor sign (^) defines to which side (left, right, or both) the search term should be anchored. If ^ does not occur in the expression, then it is not anchored to any side.

The ^ sign is translated to Z39.50 attributes using the *`position`* keyword with the following syntax:

```
position.<anchor position> = <Z39.50 attributes list>
```

The <`anchor position`> can have the following values**:**

- first
- any

The anchor position may be translated with the Z39.50 Position (3) attribute, for example:

- `position.first = 3=1`
- `position.any = 3=3`

The `position.*` line may be used to define the default anchor for all unmatched anchors.

Translation of CQL Wildcard Sign to Z39.50 Attributes

The CQL wildcard sign (*) is translated to Z39.50 attributes using the truncation keyword. The syntax is:

```
truncation.<wildcard position> = <Z39.50 attributes list>
```

The <wildcard position> can have one of the following values:

- left
- right
- none

The wildcard (*) may be translated with the Z39.50 Truncation (5) attribute, for example:

- `truncaion.right = 5=1`
- `truncaion.left = 5=2`
- Possible translation for `none` is `truncaion.none = 5=100`

Adding Z39.50 Attributes to All Queries

There is an option to add Z39.50 attributes to all queries sent by the `sru_server` to the `z39_server`. The syntax is:

```
always = <Z39.50 attributes list>.
```

For example, if `always = 6 = 1` is defined, the completeness attribute (6) with the value incomplete field (1) is added to all z39.50 queries.

Example of the table

```
#context sets definition

set.cql         = info:srw/cql-context-set/1/cql-v1.1
set.dc          = info:srw/cql-context-set/1/dc-indexes/v1.0/
set.rec         = info:srw/cql-context-set/2/rec-1.0

# index translation
index.cql.serverChoice = 1=1016
index.rec.id           = 1=12
index.dc.title         = 1=4
index.dc.subject       = 1=21
index.dc.creator       = 1=1003

#relation translation
relation.eq            = 2=3
relation.scr            = 2=3

#truncation translation
truncation.right       = 5=1
truncation.left        = 5=2
```

```
#default values
position.*     =
structure.*    =
```

prof_alephe

Location of the table: alephe/root directory

Purpose of the table: Used to set the ALEPH environment needed for running scripts

remote_catalog.dat

Location of the table: pc_tab/catalog directory of the **BIB** library

Purpose of the table: List of remote catalogs for Z39.50 Update Service

Structure of the table:

Col. 1 Base code

Col. 2 Base name

Example of the table:

```
!          1                      2
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!>
PICA_GGC              Pica Catalog
USM01_Z39             USM01 (Z39)
```

rep_tab_short.<lng>

Location of the table: tab directory of the library

Purpose of the table: Fields for printing BIB information in columnar form for a report (p-print-08)

Related table(s):

1) edit_field.<lng> (line identifier is '***S***')
2) tab01.<lng>

This table defines the structure of bibliographic print-outs in tabular form for the p-print-08 "line-list" report. The bibliographic fields are used to define the content of a column. The tags can also be virtual fields created by expand routines. The content can be formatted using edit_field (line identifier is '***S***').

Tags can be listed on the same line in order of preference, so that the first tag found is taken to print. If ***Y*** is added at the end of the line, all occurrences of all tags listed will print.

Structure of the table:

Col. 1 format number

Col. 2 ALPHA

Col. 3 heading

Col. 4-8

1^{st} – 5^{th} alternative field + subfield. # can be used as a wildcard.

Note: Tags can be listed on the same line in order of preference, so that first tag found is taken to print

Col. 9 Width of the column, expressed as a percentage of the line. Therefore, all columns together should total 100

Col. 10 font

Col. 11 color

Col. 12 ---

Col. 13 Starting position for fixed fields. The count is from base 001.

NOTE: when determining character position, if the fixed field begins with a subfield code (as in UNIMARC field 100), the calculation does NOT have to take the three positions of the subfield into account.

For example:

for MARC 21 field 008, date-1 is 008 (listed as 07-10 in the MARC 21 manual).

For UNIMARC field 100, date-1 is 010 (listed in the UNIMARC manual as 09-12).

Col. 14 Number of characters for fixed field

Col. 15 **Y** for all occurrences of the field

Example:

```
1  2          3             4      5      6      7      8     9  10 11  12  13
 14  15
!!!-!-!!!!!!!!!!!!!!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!-!!-!!!-!!!-!!!
-!!!-!
001 L Doc. No.             SYS                                  010 02 C01
001 L Author               1####                                020 03 C02 ---
     Y
001 L Title                245##                                030 01 C03 ---
001 L Publ.                260##b                               010 04 C04 ---
001 L Year                 260##c                               010 04 C04 ---
001 L Items                LOC##                                020 05 C05 ---
     Y
```

scancode.dat

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Table for defining which authority lists are accessed on defined cataloging tags

Related table(s):

1) tab01.<lng>
2) tab00.<lng>
3) tab_base.<lng>

This tables defines the selection of bases and the headings file of each one that is accessed when the "Search Headings" function is invoked in the Cataloging module.

Note: The first line of this table must have the word ***LOCAL*** in column one, and the code of each of the bases that can be consulted, in columns 3-10.

The second and subsequent lines define which browse index is consulted for a particular field tag+indicators+subfield. The subfield is required only for cases where the particular subfield is being searched for. A hash sign (**#**) can be used as wildcard for indicators.

Up to 10 bases can be defined (cols. 3-11). Col. 2 is used to define subfields that are **NOT** overwritten when a heading is chosen from a browse list.

In column 1, if you are using specific indicators (that is, not #), make sure that specific lines are listed before general lines, since the first match found is always taken.

Example:

24510
245##

For the Search Field Headings of Other Base and Search Subfield Headings of Other Base options, it is possible to specify a base more than once to enable the user to define more than one scan code for the field or subfield option. For example, if the table is defined as follows:

```
!!!!!!-!!!----!!!----!!!----!!!----!!!----!!!----!!!----!!!----!!!-- …
LOCAL  USM01  USM10  USM12  USM11  USM01  USM01
100##  AUT    PER                   AUT    SUB
```

When selecting the Search Field Headings of Other Base option for the 100 field, the cataloger is prompted to select one of the following options:

1. Scan USM10 with PER
2. Scan USM01 with AUT
3. Scan USM01 with SUB

Note that the first base specified in the table is the local library and it is not displayed when selecting to scan the headings from other libraries. For this reason, to enable the user to select this option when using this function, the column must be repeated.

Structure of the table:

Col. 1 Tag+indicators+subfield code. Subfield is defined in the 6th position. # can be used for indicators. The 6th position may be used for defining a specific subfield that may be searched. In column 12, you can set the subfields that should be ignored when the search heading string is built (such as control subfields: 6805w). The definition set in column 12 works for the Search Field Headings tool (F3 / Ctrl + F3), but not for Search subfield headings.

Col. 2 Up to 20 subfield codes that should not be overwritten. When the heading is copied to the cataloging draft, these subfields in the cataloging draft will then be retained.

Col. 3 Access Code. At the top of the column is the code of the searchable base. Below that, for each field tag defined in column 1, is the code of the headings list of the selected base.

Col. 4-11 Heading (ACC) File code of additional bases defined. (Code of the base defined as part of the header).

Col.12 Sub-field code. Up to 20 sub-field codes that should be ignored when building the string for the Search Field Headings tool (F3 / Ctrl + F3). Not relevant for search Subfield Headings (F4 / Ctrl + F4).

Note that when an heading which is a see reference is selected, the preferred heading from the authority record will be taken. For example:

Scan in USM01, select "AT & T" (which is a 4xx heading), "American Telephone and Telegraph Company" (which is the 1xx heading of the record) is taken instead.

If the heading is not in an AUT library and there is no reference to an AUT document, the heading is taken as is via correction using tab_subfield_punctuation.

Example of the table

```
!  1   2                    3                    4                    5
           6                    7                    8                    9
               10                   11
!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!
!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!
LOCAL                        USM01                USM10                USM12
           USM01                USM11

050##                        LCC
086##                        SUD
100##  d                     AUT                  PER
           AUT
110##                        AUT                  COR
111##                        AUT                  MET
130##                        TIT                  TIT
```

server_ip_allowed

Location of the table: alephe/tab directory

Purpose of the table: Access definitions to Web and GUI

This table lists the IP addresses that are allowed to access the **ALEPH** Web interface, and **ALEPH** GUI interface.
Structure of the table:

Col. 1 Server type:
N = NCIP server
P - PC server
W - Web server
X = X-Server (part of WWW server services)

Col. 2 Allowed server IP address
A - allowed
D -denied

Col 3 IP address. * signifies wildcard

Example of the table:

```
! 2      3
!-!-!!!!!!!!!!!!!!>
W A 10.1.235.*
W A *.*.*.*

X A *.*.*.*
P A *.*.*.*
N A *.*.*.*
```

tab_85x_import

Location of the table: tab directory of the **ADM** library

Purpose of the table: Matching fields for p-serial-52 (import 85XX records)

Related table(s):

1) tab01.<lng>
2) tab00.<lng>

The p-serial-52 batch procedure is used to add 853-855 and 853X-855X fields from other libraries to the ADM records of the identical titles in your library.

Using the table it is possible to define:

- which index or word file should be used for the match,
- whether the entire field from the input file or a specific subfield should be used for the match,
- which subfield in the field in the database should be used.

It is not possible to match against a headings index.

A field can be repeated in the table so that the system checks subfields in the input file against various subfields in the database. (See, for example, 022 in the example below). In such a case, the field is entered only once in the service - the program will check all the occurrences defined in the table.

Structure of the table:

Col. 1 Field tag
Col. 2 Subfields in the imported data
Col. 3 Subfields in the database data
Col. 4 Index type (Word or Index)
Col. 5 Database index for search

Example of the table:

```
! 1             2                     3          4   5
!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!-!!!!!
001                                              I CNO
010## a                    a                     I 010
!020##                                            I 020
022## a                    a                     I ISSN
022## a                    z                     I ISSN
022## a                    y                     I ISSN
035##                                            I 035
```

tab_96_def

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Default values for fields in the Item, Order and Budget records that are created using the Generic Vendor Records Loader (p_file_96)

This table defines default values for fields in the Item, Order and Budget records that are created using the Generic Vendor Records Loader (p_file_96).

Structure of the table:

Col. 1 Z30/Z68/Z601 Field name

Col. 2 Match value. Value to match in the incoming bibliographic record or hashes to indicate that the value in column 3 will be used regardless of the bibliographic record.

Col. 3 Value to place in the field defined in column 1

Example of the table:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!>
ORDER-SEQ                        ########## mono

Z30-SUBLIBRARY                  BXMM       WID
Z30-SUBLIBRARY                  BXMA       MED
Z30-SUBLIBRARY                  YBPDD      WID
Z30-SUBLIBRARY                  ########## WID
Z30-COLLECTION                   BXMM       GEN
Z30-COLLECTION                   BXMA       GEN
Z30-COLLECTION                   ########## GEN
Z30-MATERIAL                     ########## BOOK
Z30-ITEM-STATUS                  ########## 01
Z30-CATALOGER                    ########## MASTER
```

tab_98_def

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Default values for p-file-98 (BNA loader)

This table provides default values for various record fields when records are added to the database using p-file-98 (the BNA – Blackwell North America - loader).

Default values can be set for:
Z30-CATALOGER, Z30-SUBLIBRARY, Z30-COLLECTION, Z30-MATERIAL, Z30-ITEM-STATUS, Z30-ITEM-PROCESS-STATUS and Z30-CALL-NO-TYPE

Structure of the table:

Col. 1 Name of the field for which value is being defined

Col. 2 Input value - text that is supplied in the input record from BNA

Col. 3 Record value - text that is written in the ALEPH record field

Example of the table:

```
!             1                     2          3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!…

Z30-CATALOGER                    ########## BNA
Z30-SUBLIBRARY                  ########## UEDUC
Z30-ITEM-STATUS                  ########## 11
Z30-MATERIAL                     ########## BOOK
Z30-COLLECTION                   ########## GEN
```

tab_99_def

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Default values for p-file-99 (Marcive loader)

This table is an example of an input file for p_file_99 (Marcive loader) which can receive values from an input file. The input file will be the last parameter of the p_file_99 script. The tab_99_def example defines default values for BIB, Z30 and HOL record fields.

Default values can be set for:
CATALOGER-NAME - subfield $$a of the CAT field in the BIB, ADM and HOL records
Z30-CATALOGER for item records
SUBLIBRARY, COLLECTION, CALL-NO-TYPE, MATERIAL-TYPE, ITEM-STATUS and ITEM-PROCESS-STATUS for item record fields
HOL-OWN-PREFIX and **HOL-OWN-SUFFIX** together create OWN field in HOL record. OWN field is not created if both are blank.
HOL-007 sets the first two characters of the 007 field in the HOL record.
HOL-008 sets the value of the 008 field in the HOL record (first six bytes should be "xxxxxx" --system puts the create date in this position)
HOL-LIB-SYMBOL sets the content of subfield $$a of the HOL record 852 field.

The MARCIVE loader procedure examines the location stamp in the first subfield of the 049 field (up to 10 characters), and uses this data to set the sublibrary and collection.

Structure of the table:

- Col. 1 Name of the field for which value is being defined
- Col. 2 Input value - text that is supplied in the input record from Marcive
- Col. 3 Record value - text that is written in the ALEPH record field

Example of the table:

```
CATALOGER-NAME                  ########## MARCIVE
Z30-CATALOGER                   ########## MARCIVE
Z30-ALPHA                       ########## L
Z30-SUBLIBRARY                 ########## ULINC
Z30-ITEM-STATUS                 ########## 42
Z30-MATERIAL                    ########## BOOK
```

tab_abbrev

Location of the table: tab directory of **BIB** the library

Purpose of the table: Expanding abbreviations to full text

Related table(s):

1) tab_expand - expand_doc_fix_abbreviation
2) tab01.<lng>

This table is used in conjunction with the expand_doc_fix_abbreviation routine. The fix_abbreviation routine is used in order to change abbreviations into full text. In actual fact, the routine can be used to replace any text string in a bibliographic record with a different text string.

The text exchange is defined per field tag + indicators. Field tags can be grouped by using # for wildcard.

For each line, you can define X, Y or N.
X defines that the tag is not to be considered at all, for any abbreviation fixing. Therefore, an X line should not include text in cols. 3 and 4.
N defines that the text exchange should be ignored for the particular tag, or tag grouping. An N line is necessary only if it is followed by a Y line which uses #.
Y defines that the text exchange should occur.

For example:

```
2#### Y Ft.                     FORT
```

changes Ft. to FORT in all fields that begin with 2.

But

```
245## N Ft.                     FORT
2#### Y Ft.                     FORT
```

changes Ft. to FORT in all fields that begin with 2, except for field tag 245.

Note:
If there is more than one line for an abbreviation, with different codes, the lines with "N" should appear first.

All lines with the same abbreviation should appear one after the other.
Characters outside the ASCII range can be expressed in <backslash><hexadecimal notation>, for example, \266D for MUSICAL FLAT SIGN.

Structure of the table:

col. 1 Field tag + indicators;
col. 2. Usage: X, N or Y as explained above.
col. 3 Text to be changed
col. 4 Expanded form

Example of the table:

```
!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
255## Y 1st                   FIRST
043## Y u-at---               Australia
043## Y u-at-ne               Australia New South Wales
043## Y u-at-no               Australia Northern Territory
043## Y u-at-qn               Australia Queensland
043## Y u-at-sa               Australia South Australia
043## Y u-at-tm               Australia Tasmania
043## Y u-at-vi               Australia Victoria
```

Note the following for defining this in tab_expand:
It is possible to define whether the tab_abbrev definition **adds** or **replaces** a line to the document. If you want to replace the line in the document instead of adding one, you may use col. 3 in tab_expand with "REPLACE". This is useful for display purposes (where the abbreviated term need not display). Note that ‘add’ is the default if not defined.

Example (tab_expand):

```
!   1                    2                                3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
U39-DOC    expand_doc_fix_abbreviation
U39-DOC    expand_doc_fix_abbreviation    ADD
U39-DOC    expand_doc_fix_abbreviation    REPLACE
```

All lines with the same abbreviation should appear one after the other.

tab_acq_index

Location of the table: tab directory of the **ADM** library

Purpose of the table: Display of bibliographic fields in the Acquisition's Order index

Related table(s):

1) - pc_tab_exp_field.<lng> ACQ_INDEX_TYPE
2) tab00.<lng>

3) tab01.<lng>

4) edit_field.<lng>

This table defines the bibliographic fields for the ACQ Order Index. The name of the index must also be added to the list of field values in pc_tab_exp_field.<lng>-ACQ_INDEX_TYPE.
In order to create the index retrospectively run 'Rebuild Acquisitions Order Index' (p-acq-04) from the Acquisitions batch service.

Make sure that a field defined in column 2 is properly defined in the "S" section of edit_field.<lng> table in the BIB library.

Note that the index entry is updated only if the order record is updated - that is, a change in the bibliographic record will not automatically update the index in the Acquisitions Client.

Structure of the table:

col. 1 Index name (up to 5 characters)
col. 2 bibliographic field code+subfield code
col. 3 1st alternative
col. 4 2nd alternative
col. 5 3rd alternative
col. 6 4th alternative

Example:

```
! 1     2      3      4      5      6
!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!!!!-!!!!!!

TIT   245##a
AUT   100##  110##  111##  130##
ISBN  020##
ISSN  022##
```

tab_alephadm.<lng>

Location of the table: tab directory of the library **AND** the alephe/tab directory

Purpose of the table: List of files for activation from the online ALEPHADM

This table lists that files and their location for activation from the ALEPHADM module. It is *library* sensitive, and there should be a table for each library type (BIB, ADM, and so on.). In addition the table, in the alephe/tab directory defines the **common node** tables accessible for update via the online ALEPHADM module.

The length of the table is limited to 5000 lines.

Structure of the table:

Col. 1 Directory
Col. 2 Filename
Col. 3 Language extension (Y/N)
Col. 4 ALPHA
Col. 5 Description (50 characters)

Example of the **BIB** table:

```
!    1                              2                 3 4          5
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!-!!!!!!!!!!…>
pc_tab/catalog       scancode.dat                    N L BROWSE list inde
CAT field tag
pc_tab/catalog       permission.dat                  N L Cataloger field
permissions
pc_tab/catalog       tag_text.dat                    N L Pre-defined text
for subfields
pc_tab/catalog       tagonnew.dat                    N L Default Fields f
records
pc_tab/catalog       codes                           Y L Field codes and
pc_tab/catalog       fix_doc                         Y L Fix routines dis
in the Cataloging module
pc_tab/catalog       pc_tab_find                     Y L Search screen in
Cataloging module
…
```

Example of the **ADM** table:

```
tab                  tab14                           N L Item Reshelving
tab                  tab15                           Y L Item Statuses
tab                  tab16                           N L Due dates, fines
limits
tab                  tab17                           N L Library Opening
```

Example of the **ALEPHE** table:

```
tab                   tab_base                        Y L Definition o
databases
tab                   allowed_languages               N L Allowed langu
tab                   tab_days                        N L tab_days
tab                   tab_month                       N L Months codes
abbrevi
ations
```

tab_alert

Location of the table: tab directory of all libraries
Purpose of the table: Enable sending email alerts when a batch jobs finishes execution

Structure of the table:

Col. 1 The method in which the alert is sent. Currently only emails (E) are supported

Col. 2 The information of the recipient of the alert. Currently only email addresses may be specified

Col 3 The type of task for which the alert should be sent to the specified recipient.

It currently may be used to specify the job type, as is stored in the Z100-JOB-TYPE field. Possible values are CIRC,ACQ,CAT,ILL,INDEX,UNION,ITEMS,ADAM,STAFF,SYSTEM.

Enter **ALL** to specify all batch jobs.

Col 4 The encoding which is used when sending out the email.

Example of the table:

```
!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!
E example1@test.com                                  CIRC
E example2@test.com                                  ALL
```

tab_attr_sub_library

Location of the table: tab directory of the **ADM** library

Purpose of the table: Sublibrary relationships based on station ID / IP address relating to loan/return, cash and reading room management

Related table(s):

1) tab_sub_library.<lng> - codes in col. 3 (except for type 8) must be defined in this table. Note that type 5 can either be the ADM code or a sublibrary code.

Note that from version 18.0 this has been transferred from $alephe_tab to the ADM/tab directory.

The table is limited to 6000 lines.

This table defines, for a particular workstation, or group of workstations, the sublibraries that come under its jurisdiction. Different type definitions (col. 2) define different options:

NOTE:
For Attribute types 1, 2, 3 and 6 in column 2, you can set up to 500 sublibraries.
For Attribute types 4, 5, 7 and 8 in column 2, you can set up only one sublibrary/code.

Sublibraries of the same type must be grouped together. The sublibrary code must be filled to 5 characters, and listed with one space between each sublibrary. In other words, if the sublibrary code is only 3 characters, it will then have 2 blanks for fill, and an additional blank for separator before the next sublibrary code.

Structure of the table:

col. 1 attribute identification for ip address identification

col. 2 attribute type:

1 - item can be loaned and returned
2 - item can be returned only
3 - home sublibraries (for transfer)
4 - reading room loan handling
5 - cash generation sublibrary
6 - sorting of cash transactions
7 - preferred sublibrary for sorting routine 06
8 - cash receipt counter suffix
9 - item can be returned only, transfer depends on type 3 setup

cols. 3 sublibrary code/counter suffix
cols. 4-7 sublibrary codes

Attribute types (col. 2)
Type 1 - item can be loaned and returned
Use type 1 if you want to be able to loan and return items of the sublibrary at this station. If the sublibrary of the item is not listed as type 1 for this workstation, the item cannot be loaned or returned, and a message is displayed, when a loan or return is attempted.

Type 2 - item can be returned only
Use type 2 if you want to be able to return (but not loan) items of this sublibrary at this workstation. If the sublibrary of the item is not listed as type 2 for this workstation, the item cannot be returned, and a message is displayed, when a return is attempted. Note that type 2 does not consider the setup of type 3 for transfers. To use the setup of type 3 for transfers, use type 9.
Type 3 - sublibraries of items (for transfer)
Use type 3 to define that the items returned belong to this sublibrary. If the item's sublibrary is not listed in the type 3 group, the return is considered a "transfer" return, that is, the item belongs to a different library. A message is displayed, and a transfer slip is printed.
The item will appear as "In Transit" until it is returned to its own sublibrary. If there is a patron record that has the item's sublibrary code as its ID, the item is also loaned to the sublibrary for the transfer. Such a patron has to have a local (Z305) record. This Z305 record must have at least
z305_loan_permission set to "Y",
z305_loan_check set to "N",
z305_ignore_late_return set to "Y",
z305_rr_permission set to "Y".
It is also advisable to set z305_expiry_date to the latest possible date and the z305_cash_limit to the highest possible sum.

A type 3 sublibrary must also be registered as type 1 or 2 or 4 for the same Station ID/IP address for the purpose of circulation activities.

Type 4 - reading room loan handling

Use type 4 to define a Reading Room workstation. It is important to note that when there is a Reading Room sublibrary defined for an Station ID/IP address it means that all the loan transactions performed at this Station ID/IP address will be treated as Reading Room loans.

This means that the item's sublibrary will change to the Reading Room and will remain as such as long as the item is on loan. When the item is returned, the item's sublibrary will change back according to the item's original sublibrary (Z30_SUB_LIBRARY). When a Station ID/IP station serves a regular sublibrary (that is, not a Reading Room sublibrary), type 4 should be left empty.

Type 5 - cash generation sublibrary
Use this type for cash transactions that are not item-specific. When a transaction of this type (for example self-registration of patrons in the Web OPAC, photocopy request, ILL processing) occurs at the Station ID/IP address, the sublibrary registered here will be written in the cash record.

You can define only one sublibrary of type 5. If no sublibrary is entered as type 5, the active ADM library is automatically taken.

Type 6 - sorting of cash transactions
This type filters the cash transaction records listed in the Patron Activity/Cash pane. Only the transactions of the libraries that are registered here will display. The amount in the "sum to pay" is adjusted accordingly.

Type 7 - preferred sublibrary for items sorting routine 06
Defines which sublibrary will be the first when sorting items list by sorting routine 06. Only one sublibrary code should be defined for this type in column 3.

This type can also be used in order to associate a workstation with a specific sublibrary. This setup is used to determine the sender sublibrary for a Transfer Slip between sublibraries.

Type 8 - cash receipt counter suffix
If defined, the cash receipt number will use the counter last-cash-rec-<suffix> instead of last-cash-receipt-no. (UTIL G/2). You can enter any code up to 5 characters in column 3. Only one code should be set up.

For example, if you enter xxxxx as type 8, the counter should be last-cash-rec-xxxxx.

If one of the parameters is missing, the system will use the last-cash-receipt-no counter.

Type 9 - item can be returned only and transfered according to type 3.
Use type 9 if you want to be able to return (but not loan) items of this sublibrary at this workstation. This type is the same as type 2, but the transfer depends on the type 3 setup.

Example of the table:

```
!       1            2   3     4     5     6     7
!!!!!!!!!!!!!!!!!!!-!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!>
10.1.235.85          1 UARCV UELEC URLEC UGDOC URDOC
                     1 UEDUC UHLTH ULAW  UMUSI ULINC
                     1 USCI  URPTN USM50
                     1 ELEC5 GDOC5 EDUC5 LINC5 LINC6
                     3 UARCV UELEC URLEC UGDOC URDOC
                     3 UEDUC UHLTH ULAW  UELEC ULINC
                     3 USCI  UUPTN URPTN UREXU
                     3 LINC5 EDUC5 GDOC5 ELEC5
                     4
                     5 ULINC
                     6
                     7 UEDUC
                     8
                     9
```

tab_aut

Location of the table: tab directory of the **BIB** library

Purpose of the table: Matching of BIB library's Headings (ACC) index to AUThority records

Related table(s):

1) tab00.<lng>- (ACC) code in BIB library

There are several aspects to the link between BIB and AUT type libraries. One of these is the enrichment of the BIB headings table (Z01) from the AUT including cross-references from non-preferred terms and language equivalents of headings.

For the update of Z01 from AUT :

- The UTIL E/8 daemon must be running.
- If a new heading is inserted in BIB the relation to AUT is set to –NEW-.
- Using tab_aut, the E-8 daemon checks all records marked as –NEW- against headings in the AUT library.
- If a parallel record is found in the linked AUT, a link to the AUT record is added to BIB heading record.
- **tab_aut** defines, for each BIB headings list, which AUT library to search for a match.
- For more on AUT-BIB links refer to UTIL E-8.

NOTE: In order for tab_aut to work correctly there must be a GENERAL (with code GEN) headings table in the **AUT** library to which all authority terms and their cross references are indexed.
Note: Maximum number of lines – 100.

Structure of the table:

Col. 1 headings list (ACC) code in BIB library

Col. 2 Usage code from tag 008 of the AUT library which must be a or c:

- 1 - pos. 14 (Main heading)
- 2 - pos. 15 (Subject)
- 3 - pos. 16 (Series)

If left blank there is no check on the usage codes in the authority record.

Col. 3 code of 1st AUT library

Col. 4 code of 2nd AUT library

Col. 5 code of 3rd AUT library

Col. 6 code of 4th AUT library

The **usage code** filters the match according to the values for heading use defined in the 008/14-16 field of the authority record. The AUT record is considered a match only if both the text and the usage codes match.

Example of the table:

```
! 1   2   3     4     5     6
!!!!!-!-!!!!!-!!!!!-!!!!!-!!!!!
AUT   1 USM10
TIT   1 USM10
SRS   3 USM10
SUL   2 USM10
SUB   2 USM10 USM12
```

tab_author_num.conf

Location of the table: tab directory of the library

Purpose of the table: Defines the library's policy for assigning the following Item Call Number fields of the special "852 Form": Item Cal Number, Author Number and Chronology Number.

Related table(s):

1) tab_cautter
2) tab_cutter_three
3) tab_lee_jai_chul
4) tab_jang_il_sei

Different policies can be set per sublibrary. This can be achieved by setting the sublibrary code as the extension of the table file name. For example, to define the policy for sublibrary code: "law", the following file name should be set: tab_author_num.conf.law.
The relevant table is called by the system according to the item's sublibrary code. If there is no table with the relevant sublibrary code or the item is not assigned yet

with a sublibrary, the default table, tab_author_num.conf, is used. For more information, refer to the Item Call Number- Author Number and Chronology Number section of the *Aleph 20.01 System Librarian's Guide – Items*.

Sample of tab_author_num.conf

```
[CALL-NO]
LAW = 090##,a,1,056##,a,1,082##,a,2
WID = 050##,a,2,052#1,a,1,090##,a,3
##### = 050##,a,2,052#1,a,1,090##,a,9

[AUTHOR-NAME]
fields = 100,110,111,245

[LATIN-TYPE]
type = 1

[HANGUL-TYPE]
type = 8

[COMMON-KOREAN]
common_words = ae40,c774,bc15,cd5c,c815,c870,ac15,c7a5,d55c,c724

[SPECIAL-LETTER-VALUE]
special_letter = 3131


[CHRON-TYPE]
type = 1
sequence = korean
```

[CALL-NO] section

Defines the BIB fields that are applied in order to build the Call Number ($$i of Z30-CALL-NO / Z30-CALL-NO-2) and Call Number type (Z30-CALL-NO-TYPE/ Z30-CALL-NO-TYPE-2). For example:

WID = 050##,a,2,052#1,a,1,090##,a,9

“WID” is the item sublibrary code, BIB tag 050##$$a is the first candidate tag to populate $$i of the item call number. If 050##$$a is used for populating the call number, then the value "2" is set in the Call Number type. If the BIB record has no 050##$$a, the system keeps checking the next option: 052#1,a,1 and so on.

Note that in case you define of tab_author_num per sublibrary (for example, tab_author_num.wid) or if all the sublibraries use the same algorithm; you may set the "match all" line. For example:

= 050##,a,2,052#1,a,1,090##,a,9

[AUTHOR-NAME] section
Define the BIB fields that are applied in order to determine whether the BIB record is Korean (Hangul or Hanja) or Else. For example:

AUTHOR-NAME = 100,110,111,700,710,711

The system first looks for 100 and then for 110 and then 111, and so on, until it finds a matching field.

Note that user may override the BIB value defined in AUTHOR-NAME of tab_author_num.conf by clicking the button which located to the right of the **Author Number** field (at the "852 Form"). Clicking this button displays a list of alternative BIB tags for calculating the Author Number.

[LATIN-TYPE] section
Used for generating the **Author Number** field for non-Korean material (e.g. Latin). For example:
type = 1

Values can be:
0 = Author Number of Item Part fields is displayed null (it is not automatically generated by the system). Users may select the relevant Author Number method from the Author Number drop-down list or manually type in a value.
1= Upon opening the "852 Form" the **Author Number** field is populated with Cutter Sanborn method. Users may override the automatically filled value.
2= Upon opening the "852 Form" the **Author Number** field is populated with Cutter three-figure method. Users may override the automatically filled value.
3= Upon opening the "852 Form" the **Author Number** field is populated with the Cutter three-figure without caret mark method. You can override the automatically filled value.

[HANGUL-TYPE] section
Used for generating the **Author Number** field for Korean (Hangul and Hanja) material. For example:
type = 4

Values can be:
0 = Author Number of Item Part fields is displayed null (it is not automatically generated by the system). Users may select the relevant Author Number method from the Author Number drop-down list or manually type in a value.
1 thru 8 = Upon opening the "852 Form" the **Author Number** field is populated with Lee Jai-Chul method according to the routine type as defined in ./xxx01/tab/tab_lee_jai_chul. Users may override the automatically filled value by selecting another method from the combo-box list or by manually typing in a value.
9 = Upon opening the "852 Form" the **Author Number** field is populated with the Elrod method. Users may override the automatically filled value.
a = Upon opening the "852 Form", the **Author Number** field is populated with the Jang Il Sei method. Users may override the automatically filled value.

[COMMON-KOREAN] section

Used by the system if the Author Number is assigned using one of the Lee Jai Chul types.

It is used to define the commonly used Korean names. For example:

common_words = ae40,c774,bc15,cd5c,c815,c870,ac15,c7a5,d55c,c724

[SPECIAL-LETTER-VALUE] section

Used by the system if the Author Number is assigned using Lee Jai Chul types 5, 6, or 8. It is used to define the exception letter that should be handled differently. For example:

special_letter = 3131

[CHRON-TYPE] section

Used by the system to to define policy for assigning the Chronology Number. This section has two parameters: "sequence" and "type". For example:

type = 1

sequence = korean

The "type" parameter defines whether the library wants the Chronology field to be automatically populated when the "852 Form" is opened.

Types values can be "0" or "1".

0 – Chronology Number is displayed null (it is not automatically generated by the system). Users may manually select the "Chronology Number" option from the drop-down list or type it in.

1 – Upon opening the "852 Form" the **Chronology Number** field is automatically populated based on the Chronology Number algorithm. Users may override the automatically filled value.

The "sequence" parameter defines according to which sequence list the chronology number sequence part is set: Korean sequence list or Latin sequence list.

Values can be "korean" or "latin".

tab_base.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Database definitions

Related table(s):

1) tab_base_count – base sensitive browse list
2) tab_item_list_order – base sensitive links to library holdings
3) edit_doc_999.<lng>

The tab_base table defines the logical and physical databases that can be accessed by the user through Web OPAC and GUI Search functions.

Logical bases:

Logical bases are defined by setting a FIND command that serves as a pre-filter or scope. The find command can be up to 500 characters long.

In order to set a logical base to include everything EXCEPT a group of records, use ***alldocuments*** to define everything, together with ***not***.
for example, alldocuments not wsp=suppressed.
Note that "alldocuments" is actually all records that are indexed in the WRD word group.

Limited access
As part of the tab_base definitions, it is possible to define the libraries that can be entered without signing in. To **prevent** a non signed-in patron from entering a base, ***** (asterisk) should be entered in col. 2.

Example:

```
ISRAEL               * Israel              USM01             USM01 wti=history
```

Extension code - Base-sensitive Web screens:
When the patron connects to a base (logical or physical) in the Web OPAC, the system uses HTML pages that have an extension that is the same as the base code. In addition to the base code itself, up to 3 extensions can be defined (in col. 4-6). In this way, it is possible to set a common extension for a group of base codes. The system chooses the first match found from the 3 columns. If no matching HTML page is found, the system uses the page that does not have an extension.

Extension code - Base-sensitive full record display
The extension code mechanism described for Web HTML pages is also used for tables that govern full record display:
edit_doc.<lng>
edit_doc_999.<lng>
edit_doc_999_aut_<aut_lib>.<lng>
This mechanism serves both Web OPAC and GUI Search functions.

Base-sensitive browse list:
In the browse list, the display of the counter of records related to a heading can be set to base-sensitivity for more than one logical base, using /alephe/tab/tab_base_count
See tab_base_count for a more detailed explanation.

Base-sensitive links to library holdings (items):
The list of sublibraries (using the ITM3 or ITM5 lines in edit_doc_999, and in the drop-down list of libraries on the list of items display), can be filtered according to logical bases, using /lib/tab/tab_item_list_order.

Logical base efficiency - Z0102:
In order to make the browse functionality more efficient for small logical bases, you should utilize the Z0102 Oracle table. Usage is defined in this tab_base table in column 8. Note that up to 200 small bases can be defined using this mechanism.

Use UTIL H/1/10 to determine whether Z0102 should be generated for a particular base. It is recommended Z0102 be used only for bases that constitute less than 10% of the total database. It cannot be used for bases defined by "alldocuments not ...".

Structure of the table:

Col. 1 base code

Col. 2 * - Limit access to signed-in patrons;

Col. 3 base name (as will display in the Web OPAC)

Col. 4-6 WWW base extension code x 3 (that is, up to 3 possible fallbacks).

Col. 7 Code of the physical library

Col. 8 Use Z0102 (limited to 200)

Y = Use Z0102

N = Do not use

Col. 9 Find command (maximum 500 characters including spaces.)

Example of table:

```
!         1           2            3             4     5     6     7   8     9
!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!-!!!!!!!!!>
USM01                   USMARC BIB (USM01)   USM01 USM01       USM01
USM10                   USMARC AUT (USM10)   USM10 USM01       USM10
USM40                   USMARC ILL (USM40)   USM40 USM01       USM40
…
SERIALS                 Serial (USM01)       USM01             USM01 Y wfm=SE
RUSSIAN                 Russian (USM01)      USM01             USM01 N wln=RUS
FRENCH                  French Lang.         USM01             USM01 Y wln=fre
USA                   * United States        USM01             USM01 Y wrd = unit
ed states and wsl = uarcv or ueduc
!*=========================
LINC                    Lincoln Library      USM01             USM01 N wsl=ulinc
…
HEAL                    Health Library       USM01             USM01 N wsl=uhlth
LIEDUC                  Linc/Educ            USM01             USM01 N wsl=(ulinc
 or ueduc)
```

tab_base_count

Location of the table: alephe/tab directory

Purpose of the table: Logical bases (record counters) in Web and GUI browse list

Related table(s):

1) tab_base.<lng>
2) pc_tab_col.<lng> - PC_SEAR_SCAN

The tab_base_count is used for OPAC display of browse lists, that takes logical bases into account. Using this table, up to three columns can be set, for three logical bases. Each column displays a counter of the number of relevant records.

Standard browse list display:
Browse lists of Web OPAC and GUI Search functions display the headings relevant to the logical base being searched, and the counter of records is calculated

accordingly. For libraries that do not use this tab_base_count table, the browse list will show the counter and the heading.

Multiple base browse list display:
Browse lists of Web OPAC and GUI Search functions can be set up to display separate counters for the number of records related to a heading for a "primary" logical base, and up to two "secondary" bases. The primary base must be set up in tab_base to include all the records that will be included in each of the secondary bases.

The primary base is set in column 1, and the count of the number of records displays in the "Total" column in Web OPAC, and in "Docs" column in GUI Search functions. The secondary bases are set in cols. 2 and 3 for Web OPAC, and in columns 4 and 5 for GUI Search functions. The base code displays at the head of each of these columns.

Display in Web OPAC:
In order for the base counter columns to display in the Web OPAC, scan-acc-head and scan-acc-body HTML pages must be set up properly. Proper setup includes:
add the base code or an alternate extension (defined in tab_base.<lng>) to the HTML file name (for example, scan-acc-head-educ)
include column definitions in scan-acc-head, for example,

```
<th class="text3" nowrap>Total</th>
<th class="text3" nowrap>$0500</th>
<th class="text3" nowrap>$0600</th>
<th class="text3">Entry</th>
```

include column definitions in scan-acc-body, for example,

```
<td class=td1 id=centered width=8%>$0200</td>
<td class=td1 id=centered width=8%>$0700</td>
<td class=td1 id=centered width=8%>$0800</td>
<td class=td1 >$0400$0500$0600</td>
```

Display in GUI Search:
In order for the columns to display in GUI Search functions, pc_tab_col.<lng> must include the following lines:

PC_SEAR_SCAN	L $1	05 010 01 C01	Base-1 count
PC_SEAR_SCAN	L $2	06 010 01 C01	Base-2 count

The browse list of headings is filtered to display the headings that are relevant to the base defined in col.1. Therefore, the OPAC headings display will make sense only if the second and third columns are subsets of the first column.

For example:

Col. 1 - entire database

Col. 2 – Ex Libris University

Col. 3 - Math/Physics library in Wazawoo University

OR

Col. 1 - Ex Libris University

Col. 2 - Math/Physics library in Ex Libris University

Col. 3 - Humanities library in Ex Libris University

Structure of the table:

Col. 1 Base code

Col. 2-3 Additional Base in WWW browse list

Col. 4-5 Additional Base in GUI browse list

Example of the table:

```
!         1                   2                   3                   4
            5
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!
!!!-!!!!!!!!!!!!!!!!!!
LIEDUC               LINC                EDUC                LINC
    EDUC
USC                  LIEDUC              LAW                 LIEDUC
    LAW
HESCI                HEAL                SCI                 HEAL
    SCI
!USM01                USC                 EDUC                USC
     EDUC
```

tab_binding

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines binding readiness parameters

Related table(s):

1) pc_tab_exp_field.<lng>-
    a. BINDING-PRIORITY
    b. BIND-DISCARD-TYPE
    c. BINDING-PRIORITY

The table specifies sets of parameters for binding or discard rules for journal subscriptions.
Each set is identified by a unique two-digit number. The identifier is entered in the Discard/Binding field of the subscription form.

The drop-down list of values for the subscription form is set in the BIND-DISCARD-TYPE menu of the library's pc_tab_exp_field.<lng> configuration table.

The Bind Alert Report (p-bind-01)and Serial Binding Slip (p-bind-02) services are submitted with parameters for Binding Priority, Binding Type and Binder Code.

When the service is run, tab_binding is used to determine whether the Discard/Binding rule in the subscription record matches the service's parameters.

Issues linked to the subscriptions are retrieved and analyzed for readiness for binding.

Issues that have material type ISSBD are ignored, unless the value in column 3 is 999 (ready for binding when last issue of the volume has been received).
However, if the LAST issue is ISSBD, the volume is not included in the report.

Structure of the table:

Col. 1 Binding rules identifier.

Col. 2 Grouping argument. Defines the method used for determining "number issues" considered to make up a volume. Presently this column can contain only: 01 - represent enum_a (that is, uses Z30-ENUMERATION-A)

Col. 3 No. of issues. Number of issues that constitute a volume for binding. Example: if a volume is complete in 12 issues, but is bound in 2 volumes, this can be set to "6"; if a volume is complete in 4 issues, which are bound in a single volume, this can be set to 999.

Col. 5 EDA Delay. The number of days to elapse after Expected Arrival Date of last issue of a volume. For example, if this is set to "60", the group is considered complete, even though the last issue has not arrived when 60 days have elapsed since its expected arrival date.

Col. 6 Priority. Used to describe binding priority. For example, high, secondary, low. Note: the text placed in this column must be exactly the same as the text set in the menu BINDING-PRIORITY, column 4 of the table pc_tab_exp_field.<lng>

Col. 7 Type. Used to describe the standard of binding for example,: high, average, basic. The text placed in this column must be exactly the same as the text set in the menu BINDING-TYPE, column 4 of the table pc_tab_exp_field.<lng>

Col. 8 Binder. The library might choose to open Z70 Vendor records and/or Z303 Patron records for Binders. In this case this parameter will be the relevant code/ID.;

Example of the table:

```
!1  2  3    4    5       6          7              8
!!-!!-!!!-!!!!-!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!
01 01 999 0003 0001 high       standard   BINDER01
02 01 999 0003 0001 high       standard   BINDER02
03 01 003 0003 0001 low        high       BINDER02
04 01 999 0015 0001 high       basic      BINDER01
05 01 002 0003 0001 secondary  standard   BINDER01
06 01 002 0003 0001 secondary  basic      BINDER02
07 01 006 0003 0001 high       standard   BINDER01
08 01 004 0003 0001 high       basic      BINDER02
90 01 999 0365 0365 discard    discard    DISCARD
91 01 999 0730 0730 discard    discard    DISCARD
92 01 999 1825 1825 discard    discard    DISCARD
```

tab_bib_aut_match

Location of the table: tab directory of the BIB library

Purpose of the table: Defines 6XX tags and AUT Index codes for the batch service: Create Additional Subject Heading(s) from Authority (manage-46)

Structure of the table:

Col 1. BIB Tag and indicator

Col. 2 For second position 7 only: Subfield 2 content

Col. 3 AUT library code

Col. 4 AUT Index Heading code. Irrelevant for 6XX 2nd indicator 6

Col. 4 Filing procedure for the “text comparison”

Example of tab_bib_aut_match

```
! 1       2        3     4    5
!!!!!-!!!!!!!!!!-!!!!!-!!!!!-!!
650#0            USM10 LCS
650#2            USM10 MLC
650#7 aat        USM10 AAT
6###6            USM10
```

tab_block_circ

Location of the table: tab directory of the **USR** library

Purpose of the table: Parameters for blocking loans

Related table(s):

1) tab_check_circ – checks 9a and 9b
2) aleph/error_<lng>check_circ

3) tab31
4) tab_sub_library.<lng>
5) tab15.<lng>

This table is used to determine the values for checks 9a and 9b in tab_check_circ.
9a is for checks at the ADM level and 9b is for checks at the sublibrary level.
Values can be set for the patron at Global (ADM) level (9a) for
maximum number of overdue loans or
maximum number of recalled overdue loans or
maximum fines owing.
Values can be set for the patron at sublibrary level (9b) for
maximum number of overdue loans or
maximum number of recalled overdue loans.
The cash limit does not work at the sublibrary level.
If the patron passes the threshold, the circulation transaction for which the check was defined in tab_check_circ will be blocked.
If the transaction is blocked, lines 0241, 0242, 0243, 0251 or 0252 from the file aleph/error_<lng> /check_circ display the reason for the block.

Structure of the table:

col. 1 Patron sublibrary
col. 2 Patron status
col. 3 Item sublibrary
col. 4 Maximum number of overdue loans
col. 5 Maximum number of recalled overdue loans
col. 6 Maximum amount of fines owing. Amount can be entered as full number (for example, 500), or with decimal point (for example, 500.00). Leading zeroes are not required (for example, 500, not 000000000500).

Example of the table:

```
!  1   2   3   4  5       6
!
!!!!!-!!-!!!!!-!!-!!-!!!!!!!!!!!!!

USM50 ## ##### 05 04          10000
UARCV ## ##### 02 02
UELEC 03 ##### 04 02
UELEC 06 ##### 04 02
UELEC ## ##### 02 01
UEDUC ## ##### 03 02
```

tab_booking

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defining head/tail times to the starting/ending of booking request

Related table(s):

1) tab_hold_request - check_booking_request_l
2) tab_sub_library.<lng>
3) tab15.<lng>

This table contains the head/tail times (in minutes) to be added to the starting/ending of booking requests. The table is sensitive to sublibrary / material-type.

Structure of the table:

Col. 1 Sublibrary

Col. 2 Item Status

Col. 3 Material Code

Col. 4 Head Time. The time that will be added at the beginning of a request. The column format is [M/H/D]nnn, defining minutes, hours or days

Col. 5 Tail Time. The time that will be added to the end of a request. The column format is [M/H/D]nnn, defining minutes, hours or days

Col. 6 Release Time. The time that has to pass from a request start time until the item becomes available to other patrons.

Col. 7 Delete Interval. The amount of time before a request start time that the patron can still delete the request.

Col. 8 Max advanced starting-date. Limits how far into the future booking can be made. check_booking_request_l (of tab_hold_request) refers to this variable.

Col. 9 Preview Period. Determines how much time in advance a booking can be activated. For for example, if the booking period is between 12:00 and 18:00, and this column is set to 5 minutes, the patron can activate the booking from 11:55.

Example of the table:

```
! 1    2   3     4    5   6    7    8    9
!!!!!-!!-!!!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!
WID   01 BOOK  M077 M025 M040 M020 M010 M060
##### 44 ##### H000 M000 M000 H000 M000 M060
##### ## ##### H000 M030 M015 H001 Y001 M060
!##### ## ISSUE M030 M030 M030 H001 Y001 M060
!##### ## ##### M000 M000 M000 H000 Y001 M060
```

tab_bor_address

Location of the table: tab directory of the **ADM** library

Purpose of the table: Patron address periods

Related table(s):

1) tab100 – DEFAULT-ADDRESS

Note that it is possible to set default information in the tab100 switch - DEFAULT-ADDRESS instead of using this table.

This table is used by libraries which prefer to define the "current patron address" by time periods in the year (such as semester 1, semester 2, summer holiday, spring break), rather than by the convention of address type
01 for permanent address
02 for mailing address, and so on.
together with valid from-to dates in the address record itself.

Define time periods in this table if you want your address records to be defined accordingly. Otherwise, leave the table empty and the system will automatically work with the "01" and "02" method.

Note that the table MUST be present (even if it is empty).

The table defines the periods from-date - to-date, and then the patron address type that is used for the time period.
The dates in the table must be continuous. They must NOT overlap, and there must be NO gaps.

The patron address type is entered in the relevant patron address record, when the record is created or updated. Note that the dates from-to in the patron record are ignored, and the first address record with the matching address type is used by the system.

Each line defines which address-type identification is in effect for a time period.

If the address period is defined in this table:
When the patron updates his address record in the Web OPAC, the current record will be updated. If there is no current record, the record that suits the immediately preceding line in this table will be used (only one line up) to create a new (current) record.

In the same manner, when the system searches for the appropriate patron record for printed products, if there is no record with the address type expected according to the time period in the table, the system takes the record that has the same type as is registered in the previous line in the table, and duplicates it, assigning the expected address type.

If the address period is NOT defined in this table:
Updating the address in the Web OPAC (using the Update Patron Address form, filename bor-update) will NOT change the last record, it will create a new record with an increased sequence number.

When the system determines which address record to use, if there is more than one record with relevant valid from-to dates, the system chooses the record type 02 (mailing address) with the highest sequence number. If the system does not find a suitable 02 address, the system then searches for a suitable 01 (permanent) address. If no address is found, no address is printed.

Structure of the table:

col. 1 From date

col. 2 To date

col. 3 Address type

Example of the table:

```
19990101 19990631 21
19990701 19991231 22
```

tab_bor_id.<lng>

Location of the table: tab directory of the **USR** library

Purpose of the table: management of patron ID's

Related table(s):

1) aleph_start - variable usr_library

This table defines all system-wide behavior relating to the Z308 (global patron's IDs) records. This table exists only in the library defined by the environment variable usr_library (for example, USR00) in aleph_start.

Note that it is possible to use more than one additional ID type as a key to retrieve Z303 (Web OPAC and GUI).

IDs that are NOT used as a key to retrieve Z303 in the GUI (N in column 5), can be duplicated for different patrons. Duplication for the same patron is always allowed.

The ID date can be the same but the verification must be different so there won't be two identical sets of ID and verification.

Duplication is not possible between IDs that are not used as a key to retrieve Z303 in the GUI and those that are used as key in the GUI (Y in column 5).

IDs of type 00 or 01 are always unique through out the system..

Structure of the table:

Col. 1 ID code

00 - system ID. Mandatory and cannot be updated.

01 - barcode. Mandatory, can be updated.

02 - additional ID. Optional, can be updated.

nn - additional IDs, Optional, can be added or deleted.

Col. 2 ALPHA

Col. 3 Description of type of ID

Col. 4 Not used. Enter 00

Col. 5 GUI: can be used as a key to retrieve a Z303 record in the GUI. In this column, for ID code 00 - System ID, the value may be set to "N" if the value for ID code 01 - Barcode is set to "Y". In this

case the system will only look for a patron's Barcode and not for his System ID.

Col. 6 WWW: can be used as a key to retrieve a Z303 record in the Web OPAC. A verification is always required.

Col. 7 Verification field is optional (N) or mandatory (Y):

Y=when updating a patron record in the GUI, this ID must have a verification number.

For example, system ID must have pin code, barcode must have verification-1, and so on.

ID's which are Y in column 6 must also be Y in this column (col. 7)

Col. 8 Update in WWW. this column is used in order to identify which verification (PIN) code is updated from the Web interface. Any verification which is set to "Y" will be updated according to the code entered. Col.8 **must** be set to "Y" in order for the patron be able to recall a forgotten PIN code.

Col. 9 View

Y - ID is viewable.

N - ID is NOT viewable. The ID will still be useable and updateable in the Patron Loader (PLIF) batch, and will be exported in the Export Cash Transactions (cash-05) export file. If no value is defined in this column, the ID will be viewable.

.

Example of the table:

```
!1 2                3                 4  5 6 7 8 9
!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!-!-!-!-!-!
00 L System number                    00 Y Y Y N
01 L Barcode                          00 Y Y Y Y
02 L Additional ID 02                 00 Y Y Y N
03 L Additional ID 03                 00 Y Y Y N
04 L Additional ID 04                 00 N Y Y Y
05 L Additional ID 05                 00 N Y Y Y
06 L Campus Access                    00 Y N N N
```

tab_buf_z403

Location of the table: tab directory of the **BIB** library

Purpose of the table: List of programs for building buf_z403.

Related table(s):

1)
2) edit_field.<lng> - 856 tag definitions
3) edit_doc_999.<lng> - ‘M’ type in col. 10

tab_buf_z403 is used for the creation of BUF_Z403, which is used to govern access to electronic resources.

BUF_Z403 is used for fields which are defined as "M" (link to Electronic resource) in column 10 of edit_doc_999.

The link to the object can be displayed in full "edit_doc_999" display, and in brief display in table format.

Each line in this table must have a corresponding tag in edit_doc_999.<lng>, with link type value (Col. 10) ***M***.

Column 3 in tab_buf_z403 can be used for defining display parameters for the FULL display in Web OPAC. In this case, this column is used instead of columns 3, 4 and 6 in edit_doc_999.

The use of Column 3 is as follows:

[display_tag, display_filters, subfield to filter on, contents to filter for, edit-field indicator,]

The parameters are separated by commas.

Examples for Column 3:

```
856##, ,y,*,Y,
856##, ,y,-,U,
```

In this example, the 856 field is the basis for display.

The
edit_field.<lng> formatting that will be used depends on the presence or absence of $$y in the 856 field.

The program will display the 856 field if the 856 field has subfield y with any data (*).
If it does, ***Y*** is used for matching Col. 4 (ID of edit_field line) in the library's edit_field table. If the 856 field does not contain subfield y (-), the ***U*** is used for matching Col. 4 in edit_field.

Structure of the table:

col. 1 Function name – set to hashes

col. 2 Program name

col. 3 Program arguments

Example of the table:

```
!   1                   2                              3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
########## get_buf_z403_bib               856##,,y,*,Y,
########## get_buf_z403_bib               856##,,y,-,U,
########## get_buf_z403_bib               505##,,u,*,U,
########## get_buf_z403_bib               506##,,u,*,D,
########## get_buf_z403_bib               514##,,u,*,D,
########## get_buf_z403_bib               520##,,u,*,D,
########## get_buf_z403_bib               530##,,u,*,D,
########## get_buf_z403_bib               540##,,u,*,D,
########## get_buf_z403_bib               545##,,u,*,D,
########## get_buf_z403_bib               552##,,u,*,D,
########## get_buf_z403_bib               555##,,u,*,D,
```

tab_cat_hidden_fields

Location of the table: tab directory of the **BIB** library

Purpose of the table: Defines which fields are not displayed in the Catalog Editor

This table defines which fields are not displayed in the Catalog Editor. Since the fields included in this table are not displayed, they cannot be updated through the Cataloging module.

Note that if fields such as 001 and 005 (which have corresponding fix routines in tab_fix) are present in this table, they will be duplicated in the updated record.

Related table(s):

1) tab01.<lng>

Structure of the table:

Col. 1 Field code

Example of the table:

```
! 1
!!!!!
555
```

tab_ccl_boolean_operators

Location of the table: alephe/tab directory

Purpose of the table: Language definitions of CCL Boolean operators

This table defines which words are treated as Boolean operators in given language. For each operator a translation to the **ALEPH 500** standard Boolean operator (AND, OR , NOT) is defined.

Structure of the table:

Col. 1 Language code
Col. 2 CCL operator in given language
Col. 3 ALEPH CCL operator

Example of the table:

```
!1          2              3
!!!-!!!!!!!!!!!!!!!!!!!-!!!>
### |                     OR
### ~                     NOT
### +                     AND
### &                     AND
DAN IKKE                  NOT
DAN OG                    AND
DAN ELLER                 OR
```

tab_character_conversion_line

Location of the table: alephe/unicode directory

Purpose of the table: Unicode conversion tables

Related table(s):

1) tab_filing - FILING-KEY-nn
2) edi_in_attr – col. 3
3) edi_out_attr – col. 3
4) tab_word_breaking – WORD-FIX

This table defines the character conversion procedure and character conversion table to be used in various instances when character conversion is needed.

Some of the instances are:

Name of program	Explanation
UTF_TO_URL	translation of the url link in field 856 from UTF-8 to standard required for URL
UTF_TO_WEB_MAIL	translation of UTF-8 bibliographic data for MAIL and PRINT options in Web OPAC
LOCATE	translation of data for the locate query.
FILING-KEY-nn	translation for filing purposes. This is not system-set, and there can be multiple tables. The "nn" must be coordinated with the char_conv line in the library's /tab/tab_filing table
MARCIVE_TO_UTF	translation of data imported from MARCIVE to UTF-8
RLIN_TO_UTF	translation of data imported from RLIN to UTF-8
YBP_TO_UTF	translation of data imported from to UTF-8
OCLC_TO_UTF	Translation from OCLC imports to UTF-8
UTF_TO_LOW	translation of the file which will be automatically created

Name of program	Explanation
	on the local PC if a LOW tag is added to a document
UTF_TO_VST	translation of data sending by the VST server
UTF_TO_LSSU	translation of data sending to a Sensomatic Self-Check machine
UTF_TO_3M	translation of data sending to a 3M Self-Check machine
ACQ_INDEX	translation UTF to UTF for Order and ILL Indexes
VENDOR_NAME_KEY	translation UTF to UTF for Vendor index
COURSE_NAME_KEY	translation UTF to UTF for Course Reading index
ADM_KEYWORD_KEY	translation UTF to UTF for keyword searching in ADM clients (budget, vendor)
BORROWER_NAME_KEY	translation UTF to UTF for index of patrons
WEB_BOR_REGISTRATION	translation UTF to UTF of all the fields entered in the New Borrower Registration Form of the Web OPAC. Always use the hash ('#') sign in column 3. This procedure uses the widechar_to_ascii character conversion table (Its name and contents may be configured by the customer).
WORD-FIX	Activated when building and searching words.
HEADING-KEYWORD	Used for searching by keywords in headings (duplicate the WORD-FIX line and change it to HEADING-KEYWORD or define for the letter a table according to normalized text filing).
Z	Used by fix_doc_char_conv_z (user defined)

The character conversion procedures are:

- line_utf2line_sb - translates UTF data to single-byte
- line_utf2line_utf - translates UTF data to other UTF
- line_sb2line_utf - translates single-byte data to UTF
- line_utf_decode_xml_ncr - translate NCR-coding to UTF
- line_marc8_2_line_utf - translates MARC-8 to UTF
- line_mab2line_utf - translates MAB to UTF

For line_utf2line_utf, which allows translating a single UTF character to up to 5 characters, there is a mechanism that will allow exchanging a string of up to 5 characters for another string of up to 5 characters. This is accomplished by creating an additional table that has the same name as the basic table, adding ".extended" to the table name. For example, “unicode_to_locate" can have an additional table named "unicode_to_locate.extended".

Structure of the table:

Col. 1 Identifier

Col. 2 Environment

PC, WWW, or any environment

Col. 3 ALPHA

For further refinement of col. 1 . Identifies the ALPHA code of the field, where relevant

Col. 4 Name of procedure to run

Col. 5 Character conversion table

Col. 6 Backslash notation/NCR. This column is relevant only for procedures
line_utf2line_sb
line_sb2line_utf
line_mab2line_utf
line_utf2line_marc8
It indicates that a utf-8 character that cannot be displayed in the current font and for which no fallback has been defined for it in a character conversion table is displayed in the "backslash-hexa" format (e.g. \05E9) for the 3 first procedures above, and in "hexadecimal Numeric Character Reference (NCR)" (for example, ש) for the fourth procedure (conversion to MARC-8).
Note that translation tables can use "01" in order to compress the character.

The following are examples of some specific code page conversions to UTF and vice_versa, which might be required for cataloging fix procedures, import/export of records, and so on.
8859_1_TO_UTF
UTF_TO_8859_1
8859_8_TO_UTF
UTF_TO_8859_8
8859_7_TO_UTF
UTF_TO_8859_7
8859_5_TO_UTF
UTF_TO_8859_5
UTF_TO_MARC-8
UTF_TO_MAB

Note that marc8_to_utf conversion is different from the above procedures. For this procedure, col.6 should be left blank, since the procedure itself is set to use the following tables:
marc8_ara_to_unicode
marc8_heb_to_Unicode
marc8_eacc_to_unicode
marc8_lat_to_Unicode
marc8_greek_to_unicode
marc8_rus_to_Unicode

In addition, some of the marc8_to_utf conversion values are set in the program itself, and not in the tables.

Example of the table:

```
UTF_TO_URL           ##### # line_utf2line_sb       unicode_to_8859_1
UTF_TO_WEB_MAIL      ##### # line_utf2line_sb       unicode_to_8859_1
UTF_TO_WEB_MAIL_ASCI ##### # line_utf2line_sb       unicode_to_ascii
UTF_TO_TELNET_ASCII  ##### # line_utf2line_sb       unicode_to_ascii
LOCATE               ##### # line_utf2line_utf      unicode_to_locate
```

tab_check_circ

Location of the table: tab directory of the **ADM** library

Purpose of the table: checks for loans and renewals

Related table(s):

1) tab_attr_sub_library– type 4
2) tab16 - cols 12 and 21; 99 line
3) tab15.<lng> - cols. 6, 7 and 15
4) tab100 - OVERDUE-LETTER-NO
5) tab_block_circ
6) tab25.<lng> - col. 4 =Y

This table sets the checks that will be performed when an item is loaned or renewed. Column 1 defines when the check is performed:
LOAN-GBL - checks global patron/local patron prior to loan and when "Check" is performed on the Global Patron Information window
LOAN – checks when item is being loaned to a patron. Do not use with check_circ_11_b.
RR-LOAN - item is being loaned at a reading room station (type 4 in tab_attr_sub_library.) Intended for checking the patron using the original sublibrary of the item to find the z305 patron record. Used with check_circ_2_d.
RENEW – checks when item is being renewed. applies to Renew and Renew All functionality.
RENEW-WWW – checks when patron renews loan in the Web OPAC
RENEW-GBL - checks on patron prior to ‘renew all’
RENEW-PRE - checks renew permission in the "Loans List" DLF service for all the loans of a patron.
For performance considerations, it is recommended to avoid defining the following checks:
check_circ_1_d, check_circ_9_a, check_circ_10_a.
RENEW-W-G - checks on global patron/local patron prior to 'renew selected or 'renew all'. These checks are performed when the 'renew selected' or 'renew all' options are chosen in the Web OPAC.
EXTEND - checks for "Change Date" function during Loan in Patron's Loan List window
B-CIR-10 - item renewed to a patron (p-cir-10). This section is only relevant if the service cir-10 is run with "Automatic Renewal" option set to "Yes".
B-CIR-08 - item renewed to a patron (p-cir-08). Note that running the service cir-08 with a format that has a "Hold Status" column, will give

incorrect results if the routine check_circ_6 in the B-CIR-08 section is active.
B-CIR-77 - Update patron records (p-cir-77), enables the batch cir-77 to search for Z305 records containing delinquencies.
ROUT-LOAN - item is being checked out to a routing member
X-GBL - Checks global patron / local patron prior to renewal in X Server
X-RNW- Item renewed to a patron in X Server
SIP-AUTH - checks during the authentication of a patron through in SIP2 server
SIP-LOAN - item is loaned/renewed through SIP2 server
SIP-RETURN - item is returned through SIP2 server
RMT-REG - checks regarding the duplication of a patron from a remote system.
ILL-L - In Web ILL, borrowing library: checks during 'Self-ownership check' that includes item availability check. Valid for ILL request for Loan.
ILL-C - In Web ILL, borrowing library: checks during 'Self-ownership check' that includes item availability check. Valid for ILL request for Copy.
Column 2 defines what check procedure to perform:
1_a - patron (z303) delinquencies
1_b - patron (z305) delinquencies
1_c_a: Patron expiry date: z305_expiry_date
1_c_b: Patron expiry date renew warning: z305_expiry_date. Renew warning according to parameters set in tab30
1_d - overdue check. Works according to tab100 -OVERDUE-LETTER-NO which determines when an item is overdue" (that is, due date has passed and "n" (0/1/2/3/4) notices have been sent)
1_e - block check - z305_end_block_date
1_f: Proxy Patron expiry date - z305_expiry_date. This check works with LOAN, RENEW and RENEW_WWW
2_a - check patron loan permission - z305_loan_permission
2_b - check patron renew permission - z305_renew_permission
2_c - check patron reading room permission -z305_rr_permission, taking the Z305 that matches the reading room sublibrary at which the loan is being performed sublibrary (that is, when item is loaned at a reading room station - type 4 in tab_attr_sub_library).
2_d - check patron loan and reading room permission using the original sublibrary of the item (that is, the sublibrary of the item before it was assigned the Reading Room sublibrary).
2_e - If the patron tries to loan/renew/return a magnetic media (tab25.<lng>, col. 4 = Y) an error message is called up
3 - check patron renew limit according to parameters set in tab16 (col.21)
4_a - check patron loan limits according to parameters set in tab16 (col.12)
4_b - check the patron's overall loan limit in a sublibrary according to parameters set in tab16 - 99 line
4_c - check the patron's overall total loan limits for ADM library according to parameters set in tab16) 99 line

5 - check cash limit checks whether the patron has exceeded the amount allowed in z305_cash_limit. This check retrieves list of charges to include/exclude as parameter.

For example:
CHARGES=-0001,0006,0008,0009
This means that all charges except of 0001,0006,0008,0009 are counted.

CHARGES=+0001,0006,0008,0009
This means that only 0001,0006,0008,0009 charges are counted.

If no parameter is retrieved, all cash charges are taken into account.
5_b - check patron cash limit checks whether the patron has exceeded the amount allowed in z305_cash_limit - uses only O(pen) transactions and not T(ransferred). This check retrieves a list of charges to include or exclude.

For example:
CHARGES=-0001,0006,0008,0009
This means that all charges except for 0001,0006,0008,0009 are counted.

CHARGES=+0001,0006,0008,0009
This means that only 0001,0006,0008,0009 charges are counted.

If no parameter is retrieved, all cash charges are taken into account.

6 - renew - check item requests before renewing.
6_a - loan - check item for holds before loaning
7_a - check item loan permission - check whether the item can be loaned, based on settings in column 6 in tab15.<lng>
7_b - check item renew permission - check whether a loan on this item can be renewed, column 7 in tab15.<lng>.
7_c - check item re-borrowing limit. Checks a restriction on re-borrowing the exact item within a specific time. The specific time is defined in tab15 col. 15 (where value 99 means no limitation)
8_a - check loan record - if loan lost or claimed returned, check if the item's loan record status is currently lost or claimed returned. This check is relevant only for "renew" transactions.
8_b - check renew - overdue letter check if an overdue letter has been printed for the loaned item
9_a - check overdues and fines - for ADM library, checks maximum overdues, recalled overdues and fines, based on tab_block_circ. This check retrieves 2 parameters:

- block circ table name. If not specified, the default table tab_block_circ is consulted.
- list of charges to include/exclude

For example:
CHARGES=-0001,0006,0008,0009
This means that all charges except for 0001,0006,0008,0009 are counted.

CHARGES=+0001,0006,0008,0009
This means that only 0001,0006,0008,0009 charges are counted.

If the CHARGES parameter is not retrieved, all cash charges are taken into account.

The two parameters should be separated by a semicolon (;).

For example:
tab_block_test;CHARGES=+0001,0006,0008,0009

9_b - check overdues – sublibrary. Checks maximum overdues and recalled overdues in a sublibrary based on tab_block_circ
10_a - same as 9_a except that all loans are taken into account - active, lost and claimed return. This check retrieves two parameters:

- block circ table name. If not specified, the default table tab_block_circ is consulted.
- list of charges to include/exclude
  For example:

CHARGES=-0001,0006,0008,0009
This means that all charges except for 0001,0006,0008,0009 are counted.

CHARGES=+0001,0006,0008,0009
This means that only 0001,0006,0008,0009 charges are counted.

If the CHARGES parameter is not retrieved, all cash charges are taken into account.

The two parameters should be separated by a semicolon (;).
For example:
tab_block_test;CHARGES=+0001,0006,0008,0009

10_b - same as 9_b except that all loans are taken into account - active, lost and claimed return
11_b - check item due date relative to patron expiration date
12_a - check if there is a match between the original ILL patron and the patron who is trying to loan the ILL item.
13_a: check ILL item renew. This check works with RENEW and RENEW-GBL. Check if the the renewed items are ILL items (items that have been created as a result of receving borrowing request from supplier). If so, the renew action is enabled via Circ-Loans List only for authorized ILL staff user (staff user with ILL Unit same as the ILL unit that attached to the ILL item). The reason for this check: ILL items renewal require supplier's confirmation and should not be allowed for patrons or non ILL staff. If this check is not defined than ILL items will be renewed as any other items.
14_a: Checks if the patron making the loan already has a like item on loan.
14_b: Checks if the patron making the loan already has a similar item on loan. Similar items are items that have the same enumeration/chronology and are of the same material type. Items in all sublibraries, collections, and statuses are considered similar. This check is the same as 14_a, only the definitions for Like Copy in tab100

(SUB-LIBRARY-DIVISION, HOLD-REQUEST-ITM-STATUS, HOLDREQ-PROCESS-STATUS, and HOLD-REQUEST-COLLECTION) are not taken into account.

Structure of the table:

Col. 1 defines when check is performed

Col. 2 defines check procedure to perform

Col. 3 parameters

Table consulted by checks **9a, 9b, 10a** and **10b**

Cash charge list consulted by checks **5, 5b, 9a** and **10a**

Example of the table:

```
!    1                  2                              3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!…
LOAN-GBL   check_circ_1_a
LOAN-GBL   check_circ_1_b
LOAN-GBL   check_circ_1_c_a
LOAN-GBL   check_circ_1_c_b
!LOAN-GBL   check_circ_9_a
LOAN-GBL   check_circ_10_a
!*
RENEW-GBL  check_circ_1_a
!RENEW-GBL  check_circ_9_a
RENEW-GBL  check_circ_10_a                tab_block_circ_test
```

tab_checksum

Location of the table: tab directory of the **ADM** library

Purpose of the table: Checksum programs

Related table(s):

1) tab_code_prefix – for program checksum_valid_code_prefix

This table is used for defining checksum programs. The programs can be used for creation, validation and correction of:
item barcode,
patron barcode,
budget code prefix,
vendor/supplier code prefix
vendor/supplier additional-code.
Conventions:
validate programs must be called checksum_valid_<>.cbl
create programs must be called checksum_create_<>.cbl
correct programs must be called checksum_correct_<>.cbl
validate routine names must start with CSV-
create routine names must start with CSCR-
correct routine names must start with CSCO-

Available programs are:

checksum_create_z30_by_seq
: creating item barcode, using UTIL G/2 last-barcode-number

checksum_create_z30_by_doc_no
: creating item barcode, using Z30-REC-KEY (ADM record number + item sequence number)

checksum_create_codabar
: creating codabar compatible item barcode

checksum_create_z30_icl
: site-specific procedure for the Danish libraries, which uses barcode-sequence counter in UTIL G/2)

checksum_valid_item_bar_uea and checksum_valid_bor_id_uea
: site-specific procedures for an application in the UK, for validation of item and patron barcodes

checksum_valid_bar_newcast and checksum_valid_bar_newcast
: site-specific procedures for an application in the UK, for validation of item and patron barcodes

checksum_valid_z30_bar_nlc
: site-specific procedure for an application in China for validation of item barcodes

checksum_valid_vendor_abn
: performs validation of the ABN (Australian Business Number) on Vendor Additional Code (Z70-ADDITIONAL-VENDOR-CODE). If data is entered in the Vendor Additional Code field, it must match the ABN check.

checksum_valid_code_prefix
: performs a check of the vendor code prefix and/or the budget code prefix. It works in conjunction with the prefixes defined in the tab_code_prefix table.

checksum_valid_item_bar_keio
: site-specific procedure for an application in Japan. for validation of item barcodes

checksum_valid_bor_id_keio
: site-specific procedure for an application in Japan. for validation of patron barcodes

The instances for which the programs can be used are:

CSCR-Z30-BARCODE	create item barcode
CSCR-Z30-BARCODE-ISS	create issue-item barcode for items generated in prediction patterns
CSCR-CODA	create item barcode
CSCO-VENDOR-ADD-CODE	correct vendor ABN
CSV-Z30-BARCODE	validate item barcode
CSV-VENDOR-ADD-CODE	validate vendor ABN
CSV-LOAN-ITEM-BARCODE	validate item barcode when loan is performed
CSV-LOAN-BOR-ID	validate patron ID when loan is

	performed
CSV-VENDOR-CODE	validate vendor code prefix (using tab_code_prefix)
CSV-BUDGET-NUMBER	validate budget code prefix (using tab_code_prefix)
CSV-SUBMIT-LOAN-ITEM-BARCODE	validate item barcode when item is loaned in GUI
CSV-SUBMIT-LOAN-BOR-ID	validate patron barcode when submitting loan for the patron in GUI
CSV-Z308-BOR-ID	validate patron barcode when patron is created / updated manually

Structure of the table:

col. 1 Routine name

col. 2 Program name

Example of the table:

```
! 1                                     2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
CSV-VENDOR-ADD-CODE            checksum_valid_vendor_abn
CSCO-VENDOR-ADD-CODE           checksum_correct_vendor_abn
!CSV-LOAN-ITEM-BARCODE          checksum_valid_item_bar_uea
!CSV-LOAN-ITEM-BARCODE          checksum_valid_bar_newcast
!CSV-LOAN-BOR-ID                checksum_valid_bor_id_uea
!CSV-LOAN-BOR-ID                checksum_valid_bar_newcast
!CSV-VENDOR-CODE                checksum_valid_code_prefix
!CSV-BUDGET-NUMBER              checksum_valid_code_prefix

!CSV-Z30-BARCODE                checksum_valid_z30_bar_nlc
CSCR-Z30-BARCODE               checksum_create_z30_by_seq
```

tab_circ_log.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Logging of circulation transactions

Related table(s):

3) $aleph_root/error_lng/circ_logger

This table defines whether
a transaction creates a log record,
the transaction is system generated, and
whether it can be entered manually.

The log is part of the online Circulation module.

Structure of the table:

Col. 1 Transaction number

Col. 2 Active/Non active log
  Y=activate logger
  N=do not activate

Col. 3 Order Log:
  Y = Order Log can be entered manually
  N = Order log can not be entered manually

Col. 4 Alpha and Function Description

Example of the table:

```
!1 2 3           4
!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
00 Y Y LGeneral loan note
01 Y N LRegular loan
02 Y N LOffline circulation loan
03 Y N LRegular return
04 Y N LOffline circulation return
05 Y N LLoan deleted
06 Y N LLate return
07 Y N LLate recall return
08 Y N LNo fine late return (within grace period)
09 Y N LSelf check loan
10 Y N LSelf check return
11 Y N LStaff renewal
12 Y N LWeb renewal
13 Y N LBatch renewal
```

Note: the text of the logged events are defined in alephe_error_<lng>/circ_logger.

tab_code_prefix

Location of the table: tab directory of the **ADM** library

Purpose of the table: Prefixes for budget, vendor and ILL supplier codes

This table builds a list of prefixes for budget code, vendor code and ILL supplier code.
The list is checked when tab_checksum is set to perform a check of vendor/supplier and budget code:
CSV-VENDOR-CODE,
CSV-BUDGET-CODE.

Structure of the table:

col. 1 prefix

Example of the table:

```
! 1
!!!!!

AA
```

tab_days

Location of the table: alephe/tab directory

Purpose of the table: Day codes and abbreviations

This table allows for the definition of the days of the week in different languages. The table is used in the Web OPAC and the GUI Circulation for the short loan reservations.
Structure of the table:

- col. 1 Language
- col. 2 Day numeric code (00=Sun, 01=Mon, 02=Tue, 03=Wed, 04=Thu, 05=Fri, 06=Sat)
- col. 3 Day abbreviation
- col. 4 Day name

Example of the table:

```
!1  2   3      4
!!!-!!-!!!-!!!!!!!!!!
ENG 00 Sun Sunday
ENG 01 Mon Monday
ENG 02 Tue Tuesday
ENG 03 Wed Wednesday
```

tab_dbs.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Configuration for Advanced Library Statistics (DBS)

This table is used to define the parameters that are required for the statistical evaluation according DBS (UTIL S/3).
These statistics are used for sites in Germany.

Structure of the table:

- col. 1 Parameter
- col. 2 Value
- col. 3 Description

Example of the table:

```
!      1                    2                 3
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!
REPORT-BEGIN         20000410
REPORT-END           20011212
CHECK-SL-CODE        UEDUC                Erziehung
CHECK-SL-CODE        UHLTH                Gesundheits
CHECK-SL-CODE        ULAW                 Recht
CHECK-EXT-BOR        01
SUSPEND-PROCESS-STA  LL
SUSPEND-Z30-MATERIAL ISSUE
```

tab_delinq.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Block codes and restrictions

1) tab31
2) tab_sub_library.<lng>

This table is used for two purposes:
to define the effect that a block code on the patron's Global and Local records has on patron circulation privileges. In order to block a specific privilege, enter N in the appropriate column. Otherwise leave it blank.
to set the default text description for a block code. When a block code is set in the GUI interface, if the text description field is left blank, the description in this table is automatically applied. Note that although the description field can hold up to 200 characters, the LOAN window displays only the first 40 characters, and only 100 characters are accepted from this table.

Note: Loan and Renewal in col. 4 appear as “not implemented” as these privileges are always blocked whenever there is a delinquency on the patron record.

Structure of the table:

Col. 1 Delinquency Code

Col. 2 Record type

- 1=Patron (Global) (Z303)
- 2=Patron (Local) (Z305)

Col. 3 Sublibrary. Relevant only if col. 2 is type 2 (record type – patron).

for wildcard can be used.

Col. 4 Patron permission

- "N" to set to "no privilege",
- Blank to set to privilege
- a = Z305-LOAN-PERMISSION
- b = Z305-PHOTO-PERMISSION

- c = Z305-OVER-PERMISSION
- d = Z305-MULTI-HOLD
- e = Z305-LOAN-CHECK
- f = Z305-HOLD-PERMISSION
- g = Z305-RENEW-PERMISSION
- h = Z305-RR-PERMISSION
- i = Z305-BOOKING-PERMISSION

Col. 5 ALPHA

Col. 6 Description (displays on patron card in Web OPAC).

Example of the table:

```
!  2   3        4        5         6
!          abcdefgh
!!-!-!!!!!-!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>

00 1                       L Patron has NO global blocks
01 1                       L Patron consistently returns books late
02 1                       L Disruptive behavior in library
03 1                       L Personal item left in library - contact CIRC desk
53 1                       L Patron to contact academic adviser
!
00 2 #####                 L Patron has NO local blocks
01 2 ##### N    NN         L Patron is blocked from placing a hold request.
02 2 ##### NN   NN         L Patron is not allowed to place hold or photocopy request.
```

tab_delivery_locations.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: List of delivery locations for booking requests

This table defines the delivery locations for booking requests.

Structure of the table:

Col. 1 Delivery Location Code

Col. 2 Delivery location description

Col. 3 Address of the delivery location. New line is marked with #

Example of the table:

```
! 1         2                                3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!>
LOC1  Test delivery location 1       delivery location address 1#second line
LOC2  Test delivery location 2       delivery location address 2#second line
```

tab_doc_publish

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definitions related to document publishing

Related table(s):

1) edit_paragraph.<lng>
2) edit_field.<lng>

The table is made up of two columns:
Column 1 - The type of the field.
Four types are configurable, corresponding to the four parts of the HTML document:

T - Title section. Only a single T line is allowed.
D - Meta description section. Only a single D line is allowed.
K - Meta keywords section. Multiple K lines are allowed.
B - Body section. Multiple B lines are allowed.

Column 2 - Parameters that define what fields will be extracted into the section, and how the information is to be formatted.
The parameters are separated by commas. The following parameters are allowed for each field type:

T and **D** - A paragraph number from edit_paragraph.<lng>; for example:

T 100
D 100

The edit_paragraph.<lng> and edit_field.<lng> tables define exactly which fields and subfields are to be included in the paragraph that will be in the Title section (for T) or in the Meta description section (for D).

K - List of fields and sub-fields from which keywords must be created, and the filing code which will be used when extracting the subfields .
Each line includes a single field that is separated from the list of subfields by a comma. The list of subfields is separated from the filing code by another comma.

For example:

K 7####,axyz,01
K 6####,axyz,01

Note that when keywords are created all built-in punctuation marks must be dropped.

B - List of fields, edit_field identifiers from which the body section will be created, with the weight type that the field will be assigned.

Three weight types may be assigned:
Type 1: Headings, created in the extracted HTML file using the HTML markup type <h1></h1>

Type 2: Bold text created in the extracted HTML file using the HTML markup type

Type 3: default

For example:

B 245##,D,1
B 100##,D,1
B 7####,E,2

In this example field 245## will be extracted into the body section according to the edit_field.<lng> code (column 4) D. It will be assigned the weight value 1, hence exported as a heading.

Structure of table:

col. 1 Document section identifier
col. 2 Definitions

Example of the table:

```
!1          2
!-!!!!!!!!!!!!!!!!!!!!>
T 600
D 601
K 245##,abcf,77
K 100##,abcdfg,77
K 7####,abcdfg,77
K 6####,abcdvxyz,77
B 245##,D,1
B 100##,D,1
B 7####,E,1
B 6####,E,2
```

tab_events.<lng>

Location of the table: tab directory of the library

Purpose of the table: Events definitions

This table defines which transactions will write a record on the Z69/Z35 events Oracle tables.

The transactions in the bibliographic library relate to search requests.

Types 10-12 and 40 are written in Z35 for both Web and GUI searches.
Types 20-32 are written in Z69 for Web searches (including z39.50) only.

The transactions in the Administrative library relate to Circulation (loans, returns, holds, photocopy requests). They are written in Z35.

The event type identifier is written in Z69-EVENT-TYPE or Z35-EVENT-TYPE of the Z69/Z35 Oracle tables.

In order to exclude specific event types, change column 2 to N in the relevant lines.

Structure of table:

col. 1 Event Type

col. 2 Activate - Y/N

- Y = activate event type
- N = do not activate

col. 3 Event description.

Example of the ADM table:

```
!1 2      3
!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
31 Y Creating New ILL Request

50 Y Simple Loan
51 Y Reading Room Loan
52 Y Offline Loan
53 Y Advance Booking Loan
54 Y Routing List Loan
55 Y ILL Loan
56 Y Transfer Loan
57 Y Remote Storage Loan
…
```

Example of the BIB table:

```
!1 2      3
!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

10 Y Search Command
11 Y Search command beyond limit
12 Y Search command with zero hits
20 Y Search Command - Multi field (find-a)
21 Y Search Command - Basic search (find-b)
22 Y Search Command - CCL (find-c)
23 Y Search Command - Advanced (find-d)
24 Y Search Command - Multi base (find-m)
25 Y Refine Search
26 Y Cross sets
27 Y My Library Card
28 Y Help
29 Y Scan
```

tab_exp_own.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: defines the possible values for the OWN field

This table defines the possible values for the OWN field.
It lists the possible values for the OWN field given to a staff user in the fields: Z66-USER-OWN-CREATE and Z66-USER-OWN-CHECK.

In the staff user form in the GUI, these 2 fields can be expanded. The values are taken from the table of the user library (Z66-USER-LIBRARY).
for example, If modifying/adding a user of USM50, the expand of these fields will show the list of library USM50. The values in these fields will be checked against this table. If there is an incorrect value an error message will popup.

If it is a supervisor user (Z66-USER-LIBRARY="ADMIN"), then the expand list will be empty and no checks will be done for the fields. This is because ‘supervisor’ users are working on the management of the staff users and not working with the catalog, so these fields are not relevant for them.

Col. 1 OWN description.

Col. 2 OWN values which the user can have

Example of the table:

```
!    1                                                          2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!
Owner 1                                                        OWN-1
Owner 2                                                        OWN-1
Owner 3                                                        OWN-1
Owner 4                                                        OWN-1
Owner 5                                                        OWN-1
Owner 6                                                        OWN-1
```

tab_expand

Location of the table: tab directory of the library

Purpose of the table: List of expand programs

Related table(s):

1) edit_doc_999.<lng>
2) expand_doc_bib_z30
3) pc_tab_short.<lng>,
4) tab_abbrev,
5) tab_expand_join,
6) tab_expand_join_simple,
7) tab_expand_sort,
8) tab_fix_local_notes.conf
9) tab_type_config.<lng>,
10) tab04,
11) www_tab_short.<lng>

This table allows for the defining of *expand* programs for merging linked records (from the same and different libraries) for display and indexing.

Structure of the table:

Col. 1 Expand Menu

Col. 2 Expand procedure

Col. 3 Program arguments

Note: Column 3 is used for passing arguments to the expand-routines. For example, the **expand_doc_sort_field** gets as an argument from the 3rd column, the field+subfield to sort:

```
U39-DOC    expand_doc_sort_field          CAT##,
```

This expand program will sort the CAT fields according to $$h (the hour).

In addition to listing the expand programs that could be active in a library, the table defines *where* in the library the expand will be active.

Expand Procedures:

For the full list of expand programs refer to ALEPH User Guide/ Indexing section / Expand routines, tables and Expanded fields.

However, note that
expand_doc_adm_bib
expand_doc_adm_hol
expand_doc_bib_hol
expand_doc_hol_bib

support the following options:

1) Parameters in column 3 of tab_expand can be either:
    a. a list of up to 10 comma-separated fields to include or exclude (to exclude, precede the list with a single minus (-) sign). For expand_doc_adm_bib and expand_doc_hol_bib it is a list of BIB fields; for expand_doc_adm_hol and expand_doc_bib_hol it is a list of HOL fields.
    or
    b. a merge-type to be used to merge BIB into ADM or HOL (for expand_doc_adm_bib and expand_doc_hol_bib), or merge HOL into ADM or BIB (for expand_doc_adm_hol and expand_doc_bib_hol).
2) BEFORE record2 is added or merged into record1 (for any expand_doc_lib1_lib2), it is itself expanded by using a special section in the format "LIB1-LIB2" in tab_expand of LIB2. For example, in the case of expand_doc_hol_bib, if the section "HOL-BIB" is defined in tab_expand of the BIB library, the BIB record is first expanded by using this section, and only then its fields are added to the expanded HOL record or merged into it (according to the content of column 3 of tab_expand for expand_doc_hol_bib, defined in the HOL library).

Note that if "MERGE-TYPE=" is used in column 3 of tab_expand for expand_doc_adm_hol or expand_doc_bib_hol, it is possible that more than one HOL record will be merged into the ADM or BIB record, for example, for expand_doc_bib_hol, if more than one HOL record is linked to the BIB record, the first HOL record will be merged into the BIB record; then, the second HOL record will be merged into the already merged BIB record, and so on.

In addition to the expand program, the table also defines the *context* wherein the expand will be operative:

Code	**Context**	**Explanation**
ACC	p_manage_02_b update_acc	create headings (service) internal program
ACC-BRIEF	p_manage_35	create Z0101 brief records
BUF-950	www_f_build_950	Usage – SFX gateway
BUF-Z403	p_adam_04 p_adam_08 get_buf_z403	Upload and export of ADAM records

Code	Context	Explanation
CREATE-Z13	p_manage_07_a update_short_doc	create short BIB record (service) internal program
E-DOC-<format no.>	edit_doc_xxx	Specific format record display
GUI-ACCREF	pc_sear_c1004	display of AUT record from BIB heading
GUI-BRIEF	pc_sear_c1005	Brief display/GUI Search functions
GUI-DOC-D	pc_sear_c1001	full display GUI Search functions
GUI-DOC-P	pc_sear_c1013	full print from GUI Search functions
HOLDING	pc_sear_c1024 pc_sear_c1025 get_bib_line	display of item list
HOL-LOC	www_f_edit_hol	for HOL library only. Use with expand_doc_hol_loc_1_a
INDEX	p_manage_05_a update_ind	create indexes (service) internal program
PRE-MERGE		Adds expanded fields to the merged display of a record in Union catalog/view
PRINT-CAT	p_print_04_a	print catalog (service)
PRINT-COL	p_print_08_a	columnar printout (service)
PRINT-CUST	p_print_01_a	custom format (service)
PRINT-REC	p_print_05_a	Print "Non-preferred" Catalog Records (Service)
PUBLISH	p_publish_01_a	Publising of Records for Google
RET	p_ret_01 p_ret_21	- retrieval (service interface) - sort (service interface)
SORT-DOC	p_manage_27_a update_sort_doc	create sort keys (service) internal program
TAB33-DOC	pc_com_c0123 pc_com_tab33_heading	Navigation Window (GUI) Format a doc according to tab33
U39-DOC	u39-doc	record display (UTIL F/4)
UE-08	ue_08	expands doc in the AUT library for ue-08 procedures
UPDATE-Z103	p_manage_12_a	create links (service)
WEB-ACCREF	www_f_accref	display of AUT record from BIB heading (defined in AUT library)
WEB-BRIEF	www_f_short www_f_booking	brief display/Web OPAC [not yet implemented]
WEB-FULL	www_f_edit_doc	full display/Web OPAC
WEB-FULL-1	www_f_edit_doc	full display/Web OPAC format 01
WEB-MAIL	www_f_basket_mail www_f_full_mail www_f_short_mail_print	full print (mail) from Web/OPAC
WEB-SAVE	www_f_save_set_print	full print (save) from Web/OPAC

Code	Context	Explanation
WEB-SCNIND	www_f_scan_ind	display titles in scan list for a direct index
WORD	p_manage_01_a ue_01_w	create word indexing (Service) internal program
Z00R	update_z00r	create short bib record (Service)
Z39_HOL	z39_server	(present service, holdings schema)
Z39_SERVER	z39_server	(present service)
WWW-X	www_x_find_doc www_x_get_holding www_x_present	Retrieve the OAI XML format of an expanded document. Retrieve holdings information for a Union Catalog document number. Retrieve documents according to locations within a specified set number

Example of the table:

```
U39-DOC    expand_doc_join
!U39-DOC    expand_doc_bib_hol_usm
!U39-DOC    expand_doc_type                tab_type_config
!
WORD       expand_doc_bib_loc_usm
WORD       expand_doc_bib_accref
WORD       expand_doc_type               tab_type_config.enc
WORD       expand_doc_fix_abbreviation
!
!ACC-BRIEF  expand_doc_type                tab_type_config
ACC-BRIEF  expand_doc_split_sub1
```

Note:
The expand procedure expand_doc_join now incorporates the former procedures expand_doc_join and expand_doc_join_all.
If you use the Parameter TYPE=ALL in column 3, the functionality equals the former functionality of expand_doc_join_all. Leaving column 3 blank, will provide the functionality of the former procedure expand_doc_join.

Example of the table with the functionality of expand_doc_join_all:

```
!   1                   2                              3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!…!!!!!!!!!!!!!!!!>
U39-DOC    expand_doc_join                TYPE=ALL
```

tab_expand_duplicate_field

Location of the table: tab directory of the library

Purpose of the table: Field duplication

Related table(s):

1) tab01.<lng>

The tab_expand_duplicate_field table is required for expand_doc_duplicate_field, which duplicates a field assigning a new field tag + indicators.

This expand procedure and table was added in order to overcome the problem created when using expand_doc_split, which does not retain the source field. When _split is based on a field created by _duplicate_field, the source field is retained.

Structure of the table:

Col. 1 Input Field tag + indicators

Col. 2 Output Field tag + indicators

Example of the table:

```
! 1   2
!!!!!-!!!!!
1#### A99
4#### A99
260## IMP
```

tab_expand_external

Location of the table: tab directory of the library

Purpose of the table: Splits external fields

Related table(s):

1) tab01.<lng>

This table defines definitions for expand_doc_split_external, that splits external fields (856,505 etc) when finding multiple occurrences of a designated subfield.

The split is done according to the subfield specified in col.2.

Each new field includes a single occurrence of the subfield specified, and all the other subfields. New lines created by this process have the original tag, and the original field is suppressed.

If the field does not contain multiple occurrences of the subfield, the field remains as is.

For example:
856## U

Original data:
8564 L $$4$$uhttp://authorities.loc.gov$$uhttp://dublincore.org
Output:

8564 L $$$4$uhttp://authorities.loc.gov
8564 L $$$4$uhttp://dublincore.org

505## U

Original data
5054 L $$a505 FORMATTED CONTENTS NOTE
$$uhttp://lcweb.loc.gov/marc/bibliographic/ecbdnot2.html#mrcb555
$$uhttp://www.loc.gov/standards/mets
Output:
5054 L $$a505 FORMATTED CONTENTS NOTE
$$uhttp://lcweb.loc.gov/marc/bibliographic/ecbdnot2.html#mrcb555
5054 L $$a505 FORMATTED CONTENTS NOTE
$$uhttp://www.loc.gov/standards/mets

Structure of the table:

Col.	Input Field tag + indicators
Col. 2	Subfield to identify

Example of the table:

```
! 1   2
!!!!!-!-
856## u
505## u
514## u
```

tab_expand_extract

Location of the table: tab directory of the library

Purpose of the table: Extraction of subfields for indexing

Related table(s):

1) tab01.<lng>
2) tab11_acc

The table defines extraction of subfields (in a virtual manner) for indexing.
This allows for separate indexing of every occurrence of a subfield.

Structure of the table:

Col. 1	tag – wildcard may be used
Col. 2	sub field
Col. 3	new field that will be created and that will be used for indexing (using tab11_acc)
Col. 4	Number of subfield occurrences for which the new virtual field is created. For example, it is possible to define that only the first occurrence of subfield $y in the 650 field should be used for the creation of the new field. Default is 9.

Example of the table:

```
! 1   2   3   4
!!!!!-!-!!!!!-!
010## a a010
010## z z010
020## a a020
020## z z020
022## a z022
022## y y022
022## z z022
035## a a035
035## z z035
```

tab_expand_join

Location of the table: tab directory of the library

Purpose of the table: Creation of a virtual field from two or more fields

Related table(s):

1) tab_expand
2) tab_expand_join_simple
3) tab01.<lng>
4) tab11_acc
5) edit_doc_999.<lng>

The system can add virtual fields to a bibliographic record, through the expand_doc_join procedure (as defined in tab_expand). The procedure uses the tab_expand_join table to create the virtual fields, by joining up to 5 fields together in a single new field.

Note that for non-MAB formats, the expand_doc_join will create a virtual field only if at least two of the fields being joined are present. To always create a new virtual field, use expand_doc_join_simple and its accompanying table, tab_expand_join_simple.

For MAB installations, the new virtual field will **always** be created even if only one of the fields being joined is present.

If there are multiple occurrences of the fields, joining is done in pairs if expand_doc_join is used. For example:

a1 a2 b1 b2 b3 will create a1+b1 and a2+b2.

expand_doc_join (option TYPE=ALL) joins every occurrence of the fields. For example,

a1 a2 b1 b2 b3 will create a1 + b1, a1 + b2, a1 + b3, a2 + b1, a2 + b2 and a2 + b3.

Expand_doc_join is useful for indexing, while expand_join_simple is useful for creating virtual fields for display - for example a field which is title plus edition statement [245 and 250].

Note that expand_doc_join strips non-filing initial text (using the non-filing indicator of the field(s) being joined) when the virtual field is created.

Columns 3/4, 6/7, and so on. are used to define which subfield to take and to which subfield to assign it. If the columns are left blank, the entire field is taken, and all subfields are retained as is.

Differences between tab_expand_join and tab_expand_join_simple:

TAB_EXPAND_JOIN	**TAB_EXPAND_JOIN_SIMPLE**
virtual field will be created only if all defined fields are present	new virtual field is always created
up to five fields can be defined	two fields are defined
useful for indexing	useful for display purposes
strips non-filing initial text (using the non-filing indicator of the field(s) being joined) when the virtual field is created	does not strip initial non-filing text from fields before joining them

Structure of the table:

Col. 1 tag + indicator. **Note:** in order to use the new field it must be defined in the relevant tables - tab01.<lng>, tab11_acc, and edit_doc_999.<lng>

Col. 2 first tag for building

Col. 3 subfields to take for match

If this column is empty, all subfields are taken. If you want a selection of subfields, enter the subfield codes, or enter "-" followed by the codes of the subfields that should not be taken. If the field does not have a subfield code (for example, 001), this column should be left empty.

Col. 4 new subfield code. If this column is left empty, the original subfield codes are retained.

There are 3 ways to use this column:

- Take few (or one) subfields from the text and substitute them with exact number of new subfields. for example, take abc and change to xyz
- Take all text (or few subfields) and change them to one new subfield. for example, take abcd and change to x
- Take all BUT few subfields (- sign) and change them to one subfield. for example, take -abcd and change to x

Cols. 5-16

second-fifth tags and subfields for building (as cols. 3-4)

Example of the table:

```
TST01 100## ab    1     100## cd    2     245## -b    t

ATT01 100## a           240## abcd  t
ATT02 100## a           245## abcd  t
AU100 100## -e46        240## -h6   t
AU110 110## -e46        240## -h6
AU111 111## -46         240## -h6
AT100 100## -e46        245## -ch6
AT110 110## -e46        245## -ch6
AT111 111## -46         245## -ch6
```

tab_expand_join_simple

Location of the table: tab directory of the library

Purpose of the table: Joining fields for indexing

Related table(s):

1) tab_expand
2) tab01.<lng>
3) tab11_acc
4) edit_doc_999.<lng>

The system can add virtual fields to a bibliographic record, through the expand_doc_join_simple procedure. The procedure uses this table to create the virtual fields, by joining multiple occurrences of two fields in a single new field.

Note that unlike expand_doc_join, which creates a virtual field only if all the fields being joined are present, expand_doc_join_simple always creates a new virtual field. For MAB installations, expand_doc_join or expand_doc_join_all can be used instead.

Expand_join_simple is useful for creating virtual fields for display - for example a field which is title plus edition statement [245 and 250], whereas expand_doc_join is useful for indexing.

Note that expand_doc_join_simple does not strip initial non-filing text from fields before joining them, only expand_doc_join does this.

Structure of the table:

Col. 1 tag + indicator. **Note:** in order to use the new field it must be defined in the relevant tables - tab01.<lng>, tab11_acc, and edit_doc_999.<lng>

Col. 2 first tag for building

Col. 3 first tag occurrence (nn for index, AA for All)

Col. 4 subfield/s to take for match

Col. 5 new subfield code

Col. 6 second tag for building
Col. 7 second tag occurrence
Col. 8 subfield/s to take for match
Col. 9 new subfield code

Example of the table:

```
! 1     2   3    4     5     6    7   8     9
!!!!!-!!!!!-!!-!!!!!-!!!!!-!!!!!-!!-!!!!!-!!!!!
ATS01 100## 01 a     a     240## 01 -h6   t
ATS02 100## 01 a     a     245## 01 -h6   t
```

tab_expand_sort

Location of the table: tab directory of the library

Purpose of the table: Sort subfields for expand

Related table(s):

1) tab_expand
2) tab01.<lng>

tab_expand_sort can be used in tab_expand in order to set the sort order of subfields of a field.

Structure of the table:

Col. 1 Tag + indicators
Col. 2 Subfield/s to take for sort

Example of the table:

```
! 1            2
!!!!!-!!!!!!!!!!!!!!!!!!
260## cba
```

tab_expand_split

Location of the table: tab directory of the library

Purpose of the table: Split field definitions

Related table(s):

1) tab01.<lng>

The tab_expand_split table is required for expand_doc_split, which splits a field into separate parts, splitting on the subfield specified in col.2. The split will occur on every occurrence of the subfield, thereby creating multiple occurrences of the field.

The "split" includes all the data up to the subfield, and all the data from the subfield, up to the next occurrence of the subfield, or to the end. For example:

700## t A700 T700
causes
7001 $$aMendelssohn-Bartholdy, Felix,$$d1809-1847.
$$tLider ohne Worte,$$mpiano,$$nop.62.$$nNo. 6.
$$pFruhlingslied.$$f1970
to become
A700 L $$aMendelssohn-Bartholdy, Felix,$$d1809-1847
T700 L $$tLider ohne Worte,$$mpiano,$$nop.62.$$nNo. 6.
$$pFruhlingslied.$$f1970

The above example illustrates how the split facilitates correct indexing of author-title added entries.

If the field does not contain the subfield that is defined in column 2, the field is not treated and remains as is.

Structure of the table:

Col. 1 Input tag + indicators

Col. 2 Subfield to split on

Col. 3 Output tag + indicators for text up to the breaking subfield

Col. 4 Output tag + indicators for text after the breaking subfield

Example of the table:

```
! 1   2      3     4
!!!!!-!-----!!!!!-!!!!!
700## t     A700  T700
036##  a
```

tab_export

Location of the table: alephe/tab directory

Purpose of the table: Retrieval definitions for p-export-01

Structure of the table:

Col.1 The target of the export file. It may be a single ALEPH sublibrary or an artificial target name that represents multiple sublibraries.

col. 2 Format

Specifies the format of the export file.

- If MARC or MARC2 are specified, the resulting export file will be in standard MARC Communications Format (MARC - numeric fields only, MARC2 -

the MARC format will include alphanumeric fields).

- If ALEPH is specified, the file will be in ALEPH Sequential Format.

col. 3 Zip Flag

- Y- the file will be zipped.
- N- the file will not be zipped

col. 4 Retrieval Parameters

This column is composed of 8 comma-delimited parameters. They represent 2 sets of parameters:

tag, subfield, "from" value, "up to" value,

tag, subfield, "from" value, "up to" value.

These two sets are used to determine which holdings data should be included in any given execution of the export process.

Parameters example:

```
XPT,a,Y,Y,852,b,ULINC,ULINC
```

The above line instructs the system to extract all holdings data that has:

an 'XTP' tag with subfield $a that contains the value 'Y' and also and 852 tag with subfield $b that contains a value between ULINC and ULINC (that is, only ULINC satisfies the condition).

col. 5 Indicates the name of the directory into which the export file is to be placed.

Example of the table:

```
! 1      2    3                        4                                    5
!!!!!--!!!!!--!--!!!!!!!!!!!!!!!!!!!!!!!!!!!!!…!!!>
ULINC  MARC   N  XPT,a,Y,Y,852,b,ULINC,ULINC            /aleph/u01/outdir
MANY   MARC   Y  XPT,a,Y,Y,852,b,GROUP,GROUP            /aleph/u01/outdir
UARCV  MARC   Y  852,b,UARCV,UARCV,,,,                  /aleph/u01/outdir
```

tab_export_03

Location of the table: tab directory of the **ADM** library

Purpose of the table: Smart barcode export configuration

Related table(s):

1) tab01.<lng>

Structure of the table:

Col. 1 Size of output-field. If size is zero , it will combine with the next non-zero sized field.

Col. 2 Source of output field - can be a

- tag name,
- ERROR-CODE,
- ERROR-TEXT,
- or a fixed text surrounded be quotes – for example, "fixed text" or ‘copy no." and so on.

Col. 3 Subfields to be taken (if col.2 is tag name)

Col. 4 Alternative tag

Col. 5 Alternative subfields

Example of the table:

```
!1              2                        3              4      5
!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!!!….!!!!!-!!!!
!!!!!!
00 Z30-BARCODE
08 "||"
00 Z30-SUBLIBRARY
00 ","
10 Z30-COLLECTION
02 "||"
00 100##                          ca         245##
00 "/"
00 245##                          a          2####
```

tab_fast_patron_register

Location of the table: tab directory of the **USR** library

Purpose of the table: Defining default values for fast patron registration

Default values to be used when registering a new patron in the Circulation GUI via the "Patron Registration Form".

The lines in this table are in the format "FIELD=VALUE". FIELD is up to 30 characters long; VALUE is up to 20 characters long.

The following fields can be assigned default values from this table:

Z303-ILL-TOTAL-LIMIT - 4 digits
Z303-ILL-ACTIVE-LIMIT - 4 digits
Z303-EXPORT-CONSENT - 1 character - "Y" or "N"
Z303-SEND-ALL-LETTERS - 1 character - "Y" or "N"
Z305-BOR-TYPE - 2 characters
Z304-DATE-TO - 8 digits; optional TEXT (1 character) - {Y (Year) / M (Month) / D (Day)}
Z304-DATE-TO relates to address of type "01" and can be either a specific date (8 digits only, for example, Z304-DATE-TO=20041231), or 8 digits with an uppercase

letter - Y (Year), M (Month) or D (Day), such that Z304-DATE-TO will be calculated by adding the specified number of years/months or days to the current date, for example, Z304-DATE-TO=00000002Y.
This means that 2 years will be added to the current date and stored in the field Z304-DATE-TO.

Z304-ADDRESS-TYPE - 2 characters, for example 02

Example of the table:

```
Z303-ILL-TOTAL-LIMIT=0555
Z303-ILL-ACTIVE-LIMIT=0444
Z303-EXPORT-CONSENT=N
Z303-SEND-ALL-LETTERS=N
Z304-DATE-TO=00000002Y
Z305-BOR-TYPE=DR
Z304-ADDRESS-TYPE=02
```

tab_filing

Location of the table: tab directory of the library

Purpose of the table: Definition of filing and normalization routines

Related table(s):

1) tab00.<lng>- col. 5
2) tab01.<lng> - col. 3

This table is used for defining filing and normalization procedures that are used when building:
Headings (Z01),
Index (Z11) entries, and
Sort keys (Z101).
Brief Records (Z0101)
If a change is made to a filing procedure after an index has built, the index must be re-built.
Filing procedures for Z01 Headings have three subsections, defined in column 2 as "D", "N" or "F".
D lines are used when Display Text is generated
N lines are used to create Normalized text. It is based on the Display text,
and is used by the system when deciding whether a heading (Z01) is new,
or whether it matches an already existing heading.
F lines are used to create Filing text. It is based on the Normalized text

Normalization is based on NACO Normalization Rules, and the procedures that make up this normalization are:
comma
del_sub_field_code
compress (with set of punctuation)

to_blank (with set of punctuation)
char_conv (with NACO change case values)
pack_spaces

The filing procedure identifier for Z01 and Z11 is set in col. 5 of tab00.<lng>.
The filing procedure identifier for Z101 is set in col. 3 of tab01.<lng>. This setting is also used for the Service p-ret-21 (Sort Catalog Records) and p-ret-01 (Retrieve Catalog Records), as well as for FILTER in the OPAC (that is, checking records in a set for presence of particular text).
If no procedure is listed in col. 3 of tab01.<lng>, the system defaults to procedure 99. Therefore, the table should always include procedure 99.
The table is also used for the Services Shelf Reading Reports (p-item-04, -05 and -06), using the location type.

Following are **ALEPH**'s reserved routines:
90 - parsing FIND query in OPAC
97 - Brief Jump to text
98 - Z0101 (in BIB libraries)
98 - Search bibliographic records from authority library (in AUT libraries)
99 - Default for sort keys (Z101)

The table is limited to 5000 lines.

A single filing procedure is made up of a number of individual procedures, which are explained below. Some of the procedures are used only for normalization, but the filing text is based on the normalized text, and therefore includes the procedure's outcome.

Note that the order in which the procedures is listed is important. Text that has been manipulated in "N" lines is the basis for the "F" lines, and the order must take this into account.

For example,
numbers (which compresses comma or dot in numbers to change 100,000 and 100.000 to 100000) must come before listing comma or dot in **change-to-blank**.
When using both **numbers** and **expand_num** in the same section, **numbers** must precede **expand_num**.
suppress (which suppresses text enclosed within <<...>>) which must not be preceded by a procedure (such as **to_blank** or **compress**) if the procedure removes the <> signs.
non-filing deletes characters, counting from the start of the field, as specified in the non-filing indicator. Therefore, characters must not be compressed before non-filing. For example, if the apostrophe is compressed before non-filing, the title “L'amour" will file as "mour", since the non-filing indicator "2" will remove the remaining La.

In addition, note that the routines defined in the tab_filing table require an 'F' section, therefore even if this section is not needed (because the 'N' section suffices), an 'F' line still needs to be present with "no" as the filing procedure.

For some of the procedures, characters to be considered are defined in column 4 (for example, in **to_blank**, compress and **end_punctuation**). The character can be keyboard input, or can be in Unicode notation, by entering U+<hexa value> (for example, U+002E).

The tab_filing routine that is used to build the filing text of an ACC index is also used in the search query when browsing the ACC index or using it in a keyword search. For example, if an ACC index is created using routine 01 of tab_filing, its search query also uses routine 01.

List of filing procedures:

Procedure	Action	Example
abbreviation	compress a dot between single characters. Note that This procedure is limited to characters in the 7-bit ASCII range	I.B.M. changes to IBM
bbk	special procedure for Russian filing standards. Sorting sequence is special characters, then Cyrillic, then Latin, then numbers.	
char_conv	perform character conversion procedure according to the procedure name listed in col. 4. This name must match procedure identification in col. 1 of /alephe/unicode /tab_character_conversion_line	
cjk_pinyin	adds ! before each CJK "character", translates the characters to pinyin, using the Z114 table, and adds the Unicode value in decimal notation. The ! causes the pinyin filing-text to be sequenced separately from regular Latin characters.	
cjk_stroke	same as cjk_pinyin, except that each character is translated to stroke value, using the Z114 table.	
chi_pinyin	translates each character to pinyin, using the Z114 table, and adds the Unicode value for each character. Unicode is added in order to differentiate between different characters that have the same pinyin value. Because the pinyin filing-text will be sequenced together with regular Latin characters, this routine should be used for browse lists that use the language code from 008 to create separate browse lists (for example, AUTC).	
chi_stroke	same as chi_pinyin, except that each	

Procedure	Action	Example
	character is translated to stroke value, using the Z114 table.	
comma	all commas are turned into blank except for the first comma in subfield "a" (used for normalization).	
compress	compress (that is, strip) the characters listed in col. 4. The characters can be keyboard input, or in Unicode notation.	the full stop (period) can be input in column 4 as . or as U+002E.
compress_blank	delete blanks	
compress_perblnk	compresses a blank when it is preceded by a period (".").This is needed for the filing of certain LC Call Numbers.	
cut_prefix	removes a string at the start of a line that is contained in parentheses.	(OcoLC)12345 files as 12345
del_lead_space	this routine deletes leading spaces after a subfield. It should be placed before del_subfield, since it will not be active if there is no subfield. This solves the problem of unwanted spaces left by "to_blank" (for example, "$$b(Abc)" which becomes "$$b Abc ", or spaces created by char_conv).	
del_subfield	delete subfield sign	$$x
del_sub_field_code	"$$" is retained, but the subfield code is replaced by "-". This is used for normalization, so that headings will match when the subfield content is the same, even if the subfield codes are different.	
dewey_call_no:	special procedure for correct sequencing of Dewey call numbers	
end_punctuation	deletes the characters listed in col.4 when it is the last character in the heading. Used to remove / : = and so on. at end of title, and so on. Intended for USAGE type "D" for Z01-DISPLAY-TEXT.	
expand_num	expand number (add leading zeroes to fill numbers to 7 digits, for numeric filing). It is possible to use the parameters column of the table (column 4) to specify the subfields in which the procedure should be applied. Use the "-" to specify that all subfields, except those listed in the parameters column should be expanded. If the column is left blank, then numbers are expanded in all subfields.	100 changes to 0000100
end_punctuation	deletes the characters listed in col.4	the full stop (period) can

Procedure	Action	Example
	when it is the last character in the heading. Used to remove / : = and so on. at end of title, and so on. Intended for USAGE type "D" for Z01-DISPLAY-TEXT. The characters can be keyboard input, or in Unicode notation. Removes end punctuation only from the end of line, (if it exists).	be input in column 4 as . or as U+002E.
end_sub_punctuation	deletes the characters listed in col.4 when it is the last character of a subfield	
get_subfields	use only the subfields, or subtract some using "-" as listed in col. 4	
get_subfields_order	similar to the get_subfields routine except that it retains the order of the subfields specified in column 4 of the table	
icelandic_name	changes the order of the subfields 7 and 1, placing subfield 7 after subfield 1. Intended for sorting OPAC browse lists.	Display text: $$aAlexander $$7Alfred $$1Jonsson$$c1943- Becomes Filing text: Alexander Jonsson Alfred 1943-
icelandic_z303_namefor example,	this procedure is used for filing patron names in the Icelandic manner. If the name has more than 2 words, the last word in the name is positioned after the first word.	Hulda Maria Einarsdottir -> Hulda Einarsdottir Maria
issn/isbn	These routines test the validity of the input according to the same checks done for issn/isbn in cataloging and then return only the valid data. Meaning that if **"0001-9054 (one two test)"** is a valid ISSN (and it is, since we ignore the brackets content), only "0001-9054" will be indexed (note that the hyphen is retained). If the input is invalid, then the filing routines will return exactly what's given.	
jpn	translates each character to the decimal value of the Unicode character. This causes the filing-text to be sequenced together with regular characters. Therefore, this routine should be used for browse lists that use the language code from 008 to separate by language, and are separate for "jpn" (for example, AUTJ).	

Procedure	Action	Example
kor	translates each character to the decimal value of the Unicode character. This causes the filing-text to be sequenced together with regular characters. Therefore, this routine should be used for browse lists that use the language code from 008 to separate by language, and are separate for "kor" (for example, AUTK).	
lc_call_no	special procedure for correct sequencing of LC call numbers Note that this routine adds the following three characters to the index records it creates: ! " # For this reason, you cannot have a to_blank or compress line which includes these characters after the lc_call_no line. In addition, note that this procedure is complete within itself, and does not require additional treatment. However, in order to facilitate searching, it is recommend that del_subfield be added.	
mc_to_mac	change initial mc to mac	McKay files as mackay
no	this routine is used when the procedure defined in the table does not contain an 'F' section. The procedures defined in this table require an 'F' section, therefore even if this section is not needed (because the 'N' section suffices), an 'F' line still needs to be present with "no" as the filing procedure.	
non_filing	drop initial text using non-filing indicator. This procedure is relevant only building headings (Z01).	the hobbit files as hobbit
non_numeric	delete non-numeric characters	c1965 changes to 1965
none	data remains as is	
norm_author	intended for normalizing name entries in a merged index. Retains first word and first character of second word, removing punctuation.	for example, $$aSmith, John, $$d1923 is changed to $$aSmith J
numbers	compress a comma and a dot between numbers	2,153 changes to 2153
pack_spaces	compresses all multiple spaces to a single space	
subfield_mab	intended for filing of headings which are based on MAB-authority information. In order to exclude the	

Procedure	Action	Example
	identification number from sorting, the procedure adds three blanks at the beginning of each subfield, from the second subfield on, and adds four blanks to the beginning of $$9. Subfield codes are removed.	
suppress	suppress using <<...>> (drop all text contained within the signs, and the signs themselves). << and >> are the default delimiters (when there are no parameters). Parameters can be "88-89" and/or "<<>>" (comma-delimited), for example, "88-89", "88-89,<<>>", "<<>>,88-89" or "<<>>". The parameter "88-89" means that the control characters U+0088 and U+0089 will be used instead of << and >>. The parameter "<<>>" is the same as the default. If BOTH parameters are specified, the input text will undergo suppression TWICE: once with << and >> as delimiters, and again - with U+0088 and U+0089 as delimiters.	
to_blank	change characters listed in col. 4 to blank. The characters can be keyboard input, or in Unicode notation. Always define "pack_spaces" after "to_blank", in order to compress resulting multiple spaces into one space	the full stop (period) can be input in column 4 as . or as U+002E.
to_lower	to_lower. Uses the table alephe/unicode/unicode_case.	MARC to marc
to_caret	Change subfield sign to ^^	For hierarchical sorting of headings. WARNING! although this will arrange a heading such as "$$aArt $$zZambia" before "$$aArt, Canadian", when the system performs a browse search, the search query is taken word-by-word, character-by-character, transforming multiple blanks to a single blank. Therefore, it is not possible to zero in on "art ^^zambia" in browse search, and although the list will be hierarchically

Procedure	Action	Example
		arranged, it will be difficult to use.
unicode	translates each character to the decimal value of the unicode character. This causes the filing-text to be sequenced together with regular characters. This routine is the same as kor and jpn.	
widener_call_no	this routine is used to file Widener Call Numbers. It separates the call number into seven fixed-size sections	
year_uu	This routine replaces "u" with zero ("0") in "doubtful" year formats, for example, 19uu, 197u. It should be activated BEFORE the routine "expand_num"	19uu becomes 1900 197u becomes 1970

Structure of table:

col. 1 ID - Two-digit identifier of the filing routine. This identifier is used in column 5 of tab00.<lng> and column 3 of tab01.<lng>

col. 2 Usage

Relevant only for headings (Z01)

D - use when creating Z01-DISPLAY-TEXT

N - use when creating Z01-NORMALIZED-TEXT

F or SPACE - use when creating Z01-FILING-TEXT

col. 3 Name of the filing procedure

col. 4 Parameters for the filing procedure (when relevant)

Special note: non-filing should not be used for display and normalized if used for filing.

Example of the table:

```
!1 2        3                                        4
!!-!-!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
01 D end_punctuation      :,=;/.
01 N compress             '[]|
!*rem use following line if you use <<...>>
!01 N to_blank             !"()-{};:.?/\@*%=^_`~
01 N to_blank             !"()-{}<>;:.?/\@*%=^_`~
01 N comma
01 N icelandic_name
01 N del_subfield_code
01 N char_conv            FILING-KEY-10
01 N del_lead_space
01 N to_lower
01 N pack_spaces
!01 N norm_author
01 F del_subfield
```

tab_filing_call_no

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines filing routines for items

This table defines filing routines for items.
It is used for defining filing procedures that can be used when building the filing key for item records, which is stored in Z30-CALL-NO-KEY and Z30-CALL-NO-2-KEY.
The table is used for the Services Shelf Reading Reports (p-item-04, -05 and -06), using the location type.

A single filing procedure is made up of a number of individual procedures, which are explained following. The procedures are the same as for tab_filing, and some may appear to be irrelevant.

Note that the order in which the procedures is listed is important. For example, ***numbers*** (which compresses comma or dot in numbers to change 100,000 and 100.000 to 100000) must come before listing comma or dot in ***change-to-blank***.

For list of filing procedures refer to tab_filing. Note, however, the following addition for tab_filing_call_no:

Procedure	Action	Example
add_call_no_type	adds the value in the call number type field as the first position of Z30-CALL-NO-KEY plus one blank. Because this procedure can cause multiple contiguous blanks, it should be followed by pack_spaces.	

Structure of table:

col. 1 ID - One-digit identifier of the call no type
col. 2 ALPHA
col. 3 Name of the filing procedure
col. 4 Parameters for the filing procedure (when relevant)

Example of the table:

```
1 2        3                      4
!--!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
0  L add_call_no_type
0  L del_subfield
0  L to_blank             &()-[]<>?,./
0  L lc_call_no
!*
1  L add_call_no_type
1  L to_lower
1  L to_blank             :;'",.
1  L dewey_call_no
…
```

tab_fix

Location of the table: tab directory of the library

Purpose of the table: fix_doc programs

Related table(s):

1) fix_doc.<lng> defines the name and/or description of a fix that displays in the ‘fix record’ menu option in Cataloging
2) tab_character_conversion_line – the relavant section from this table can be defined in col. 3
3) tab04
4) tab_merge_overlay

Fix routines are "logical names" for defining a group of fix programs. They are invoked by the *Fix Document* option under the Edit menu in the cataloging client.

Reserved fix routines also define when the programs are run. For example, it is possible to define a group of fix programs to run when the record is loaded to the server. To de-activate a routine either comment out the line, or delete the line.

The table can be up to 500 lines long.

The following are reserved routines:
INS - all programs linked to the INS group will be automatically activated when a record is sent to the server.
INS2 is run automatically whenever a record is updated in the Catalog GUI. The difference between INS and INS2 is that INS2 is executed just before the update in the database, and therefore can make use of the document's system number even if its a new document. However, note that check_doc programs cannot check the outcome of INS2 programs!
INSFS - is run automatically whenever a fast cataloging is done from the administrative modules (Circulation and Acquisitions). This routine is also performed when bibliographic records are created using the Special Request option in the OPAC and when bibliographic records are created in the Course Reading module.
MERGE - all programs linked to MERGE will be activated when the editing option paste document (alt+D) is invoked in the cataloging client.
LOCAT - is run automatically whenever the Locate record function is used in the Catalog GUI.
HOL - is run automatically whenever a HOL record is created. Note that this instance should be defined in the holdings library (XXX60).
P-31 is run automatically whenever authority records are loaded to the authority database by the *Load Authority Records* batch process (p_manage_31).
MNG50 - is run automatically when the Create Holdings and Item Records Using Bibliographic Data (p-manage-50) services is used. The fixes are performed on the ADM and HOL records created by the service.
ILL-L - is run automatically whenever the Locate function is used in the ILL module (Locate button in the BIB Info tab of the ILL request). Note that this instance should be defined in the ILL library (XXX20).
M-36 is run on the records in the input file for the Check Input File Against Database (manage-36) service.
DEL is run automatically when a record is being deleted (either from the GUI or during a batch process).
BNA is run automatically when the Load BNA Records (file-98) service is used to load BNA records.
AUT is run when the new derived authority record is created using the "derived new record" function in GUI Cataloging module.
LDMRG is run automatically on a bibliographic record from MARCIVE when it is merged with a matching record in the database.

Program arguments:
Certain fix_doc programs require additional information such as table names. These additional parameters are defined in column 3. The documentation for each fix_doc program indicates whether it takes parameters, and if so, how they should be

formatted. Note that if a fix_doc program does not use parameters, it will ignore the contents of column three.

It is possible to specify the section used in $alephe_unicode/tab_character_conversion_line in the 3rd column of tab_fix. Previously the section used was always "Z". This will remain the default if the 3rd column is empty.

Note: 58-IN-RECORD-FIX has been implemented for gate definition files. This entry is valid in the conf files in $alephe_root/gate/*.conf and can direct a fix when a record is retrieved from a z39 target. The value of Z58-IN-RECORD-FIX should be a fix routine defined in tab_fix of the EXTNN library.

FIX-DOC programs are defined in this table. The table fix_doc.<lng> is used for defining the text that displays in the window when user invokes ‘fix record’ from the edit menu in the cataloging client.

Structure of the table:

col. 1 logical name/routine used for a group of fixing programs

col. 2 Up to 20 program names may be assigned to each user-defined routine name.

For tab04 (conversion from one cataloging format to another), use tab04-nn where nn is the entry defined in tab04.

For fix_doc_merge use fix_doc_merge_nn where nn is the merge set in tab_merge_overlay

col. 3 Program arguments: Certain fix_doc programs require additional information such as table names. These additional parameters are defined in column 3. The documentation for each fix_doc should indicate whether or it takes parameters from this column, and if so, how they should be formatted. Note that if a fix_doc does not use parameters, it will ignore the contents of column three.

Example of the table:

```
! 1                2                                3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
INS   fix_doc_sort
INS   fix_doc_sort_lkr
INS   fix_doc_tag_008
INS   fix_doc_tag_008_open_date
…
INS2  fix_doc_lkr_up
INS2  fix_doc_001
!*
INSFS fix_doc_tag_008
INSFS fix_doc_tag_008_open_date
```

tab_fix_local_notes.conf

Location of the table: tab directory of the **BIB** library

Purpose of the table: Order of HOL records to be expanded into the BIB record.

Related table(s):

1) tab_expand

There is an option for expanding local data held in HOL records into the BIB record, using mapping, per logical base. The mapping sets which HOL records are expanded into the BIB record for Web OPAC display, dependent on the logical base being viewed.
The expand procedure , called expand_doc_bib_local_notes.cbl, expands the local notes from the BIB's HOL's into the BIB. Two parameters controls the behavior of the program - the search base and it is corresponding section in tab_expand_local_notes.conf, under the right search base paragraph. Under this paragraph , we can define the order of the HOL to be expanded according to their owners.

Tab_expand must have
WEB-FULL expand_doc_bib_local_notes
defined in order for this to be activated.

Example of the table:

```
local notes = 590##,690##
owners subfield = 9
owners list = AA,BB,LIN
merge section = 98
mapping section = LCN-2-HOL
```

tab_fix_new

Location of the table: tab directory of the library

Purpose of the table: Definitions for deriving new records

Related table(s):

1) tab_fix

tab_fix_new contains the configuration for fix_doc_new. It could be defined which tag contents should be copied from the father record to the derived record and which fixed field should be created.

It is possible to define the rules for different types of derivation. The type - according to tab_fix - has to be entered in column 2. The set ID in column 1 has to start with "01" for every new type. Within a type the set ID has to be numbered consecutively.

Example:

```
01 and 590   a 331## a
02 and 590   a 333## a ^/^$1
```

The lines above create only one occurrence of 590 with the contents of 331 and 333 separated by slash. $1 is used for the contents of 333.
It is not possible to create repeated tags.
The program takes always the first occurrence of a tag in the father record.

tab_fix_notes

Location of the table: tab directory of the BIB library

Purpose of the table: Defines texts used for translation in the fix routine fix_doc_notes

Related table(s):

1) tab_fix (fix_doc_notes routine)

tab_fix_notes defines text to be translated with the fix routine fix_doc_notes. Text may be defined for different tags and subfields. For more information, see the **Automatic Translation of Bibliographic Note Fields** section of the *Aleph 21 System Librarian's Guide-Cataloging*.

Col 1. BIB Tag & indicator

Col. 2 Subfield

Col. 3 BIB Source text

Col. 4 Translation text

The following is a partial example of tab_fix_notes (the actual lengths of columns 3 and 4 length are longer than that what they appear in the following table structure example):

```
! 1   2                    3                          4
!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>-!!!!!!!!!!!!!!!!!!!!!!->
245## h [electronic resource]              [ressource électronique]

310## a Annual                            Annuel
310## a Biannual                          Semestriel
310## a Bi-annual                         Semestriel
310## a Biennial                          1 no par 2 ans
310## a Bimonthly                         Bimestrie

500## a Adaptation of:                    Adaptation de :
500## a Added t.p. title                  Titre de la p. de t.
addit. :
```

```
500## a At head of title                    En tête du titre
500## a Caption title                      Titre de départ
500## a Cataloguer's title                Titre du catalogueur
500## a Colophon title                     Titre de l'achevé
d'imprimer
500## a Cover title                       Titre de la couv.
```

tab_fix_z103

Location of the table: tab directory of the **BIB** library

Purpose of the table: Item and HOL filtering

Related table(s):

1) tab_expand – related to the fix_z103_filter_base routine.

For consortium catalogs or systems with many sublibraries the options for the sorting of item and HOL display has been enhanced by the GUI-TREE routine in this table. It enables a sorted display in the navigation tree.

Routine name:
GUI-FULL
GUI-PRINT
GUI-TREE
HOL-LIST
WEB-FULL
WEB-SHORT

Fix z103 Procedures:
The following is a partial list of available programs.
fix_z103_filter_base
fix_z103_filter_suppress
fix_z103_sort_852_b
fix_z103_sort_852_b_item_attr
fix_z103_sort_base
fix_z103_sort_by_my_own
fix_z103_sort_existing_key
fix_z103_sort_lkr_doc_no
Note: fix_z103_sort_by_my_own. This program sorts records according to the value of the OWN tag. All records with OWN tags assigned to the user according to tab_own are positioned at the beginning of the record list.

Note also that the routine fix_z103_filter_base can use expand-routines.

The section from tab_expand is defined by the parameter EXPAND in col. 3 of tab_expand. You can use existing sections in tab_expand or introduce a new section of your own.

Structure of the table:

Col. 1 Routine name

Col. 2 Program name

Col. 3 Program arguments

tab_doc_trans

This table defines the behavior of the fix_doc_trans_doc fix routine. This fix routine adds translated content of specific fields. It retrieves the tab_doc_trans table name as a parameter and performs the translation accordingly.
The tab_doc_trans configuration table specifies:

- Which fields in the document are translated
- Which target field stores the translated content
- Which translation method is used

The configuration table must exist in the library where the fix is performed (the Bibliographic library or the AUT library) and have the following structure:

Column 1: **source field and indicators**
Size: 5
can be used for the fourth and fifth positions to indicate truncation of numeric additions to the field code (for example, 245## for 2451, 2452, 24501)

Column 2: **source subfield(s)**
Size: 10
Column 3: **target field and indicators**
Size: 5
can be used for the fourth and fifth positions to indicate truncation of numeric additions to the field code (e.g., 245## for 2451, 2452, 24501)

Column 4: **target subfield(s)**
Size: 10

Column 5: **translation method**
Size: 25
Options are: HANJA_TO_HANGUL, HANJA_TO_PINYIN,
FIX_HANJA_TO_PINYIN
PINYIN_TO_HANGUL_MOE, PINYIN_TO_HANGUL_CK
KANA_TO_ROMANIZED_KANA, KANA_TO_HANGUL

Column 6 (optional): **translation method to perform on the translated data**
Size:25
Options are: HANJA_TO_HANGUL, HANJA_TO_PINYIN, FIX_HANJA_TO_PINYIN, PINYIN_TO_HANGUL_MOE, PINYIN_TO_HANGUL_CK, KANA_TO_ROMANIZED_KANA, and KANA_TO_HANGUL

Use column 6 if you want to perform another translation on the translated content. After translating the field using the translation method as specified in column 5, another translation is done.

The added field the data after both translations.
Note that if the translated content after using the first transliteration method (as specified in column 5) is identical to the source field (no actual translation is applied) – the second transliteration (as specified in column 6) is not performed, and no fields are added to the document.

For example:
1. In <Bibliographic library>/data_tab/tab_doc_trans:

```
!
!1    2          3     4          5                          6
!!!!! !!!!!!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!
130## anp        940   anp        HANJA_TO_HANGUL
130## anp        940   anp        HANJA_TO_PINYIN
130## anp        940   anp        HANJA_TO_PINYIN            PINYIN_TO_HANGUL_MOE
130## anp        940   anp        HANJA_TO_PINYIN            PINYIN_TO_HANGUL_CK
!
240## anp        940   anp        HANJA_TO_HANGUL
240## anp        940   anp        HANJA_TO_PINYIN
240## anp        940   anp        HANJA_TO_PINYIN            PINYIN_TO_HANGUL_MOE
240## anp        940   anp        HANJA_TO_PINYIN            PINYIN_TO_HANGUL_CK
```

According to the above setup, four corresponding fields with the transliteration content are created for the 130## and 240## fields, subfields a, n, and p:
- Source data is translated using HANJA_TO_HANGUL
- Source data is translated using HANJA_TO_PINYIN
- Transliteration that is created by HANJA_TO_PINYINis translated using PINYIN_TO_HANGUL_MOE
- Transliteration that is created by HANJA_TO_PINYIN is translated using PINYIN_TO_HANGUL_CK

The translated content will be stored in field 940.

tab_font_publish

Location of the table: tab directory of the **BIB** library

Purpose of the table: Font definitions related to document publishing

When the Publishing Platform creates the HTML records in the Z00P records, the fonts are concatenated after the font-family information of the <style> tag

For example, if the setup is:
Bitstream Cyberbit
Arial

The <style> tag will be created as:

"<style type=""text/css"">body {font-family: 'Bitstream Cyberbit','Arial';} </style> ".

Structure of the table:
Col. 1 Font

Example of the table:

```
!         1
!!!!!!!!!!!!!!!!!!!!!!!!!!!>
zyksun
Bitstream Cyberbit
Arial Unicode MS
Arial
Helvetica
serif
```

tab_hld_stmt

Location of the table: tab directory of the **ADM** library

Purpose of the table: Holdings summary statement building

Related table(s):

1) tab_sub_library.<lng>
2) tab40.<lng>
3) tab15.<lng>

When expand_doc_hld_stmt is invoked, the system builds a holdings summary statement in the HOL record, using the Z30 item records that are linked to the HOL record.
The item records that are taken into consideration are:

1) items with enumeration field equal or greater than the HOL S63/4/5 field in the HOL record (or all items, if there is no S63/4/5)
2) items with processing status NP (not published) or NA (not arrived) are not included in the summary statement, but they are used for generating a break Indication.

Items which do not fall into the above two categories can also be excluded from the summary holdings statement by mapping them to be considered as if they have processing status NP or NA. This mapping can be done for items that have a particular sublibrary and/or collection, and/or item status, and/or item process status, and/or Break indicator.

Structure of the table:

Col. 1	Sublibrary code
Col. 2	Collection code
Col. 3	Item status code
Col. 4	Item process status code
Col. 5	Break indicator value
Col. 6	Map to process status NA or NP

Example of the table:

```
! 1     2   3  4  5 6
!!!!!-!!!!!-!!-!!-!-!!
##### ##### ## NA # NA
##### ##### ## NP # NP
##### ##### ## MI # NA
##### ##### ## ## g NA
##### ##### ## ## n NP
```

tab_hol_item_create

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Defines where in the bibliographic record item holdings data elements are to be found for the creation of items and holdings

Related table(s):

1) tab01.<lng>

This table is an example of a table that can be used by the p-manage-50 batch process for the creation of items and/or holdings records. The table defines where in the bibliographic record item and holdings data elements are to be found.

The user can customize more than one table of this type for the convenience of conversions and customers who need to use the p-manage-50
batch job for more than one source data.

The table defines the fields and subfields of the bibliographic record that are mapped into the given Z30 (item) fields and as fields in the HOL record.
Fields for HOL are identified by HOL- as a prefix to the field in the first column. The name of the table can then be used as a parameter to the p-manage-50 batch job.

Note that when overwriting an existing item with a new one, only the fields defined in this table are overwritten; other fields remain untouched.

The list of valid fields follows. Note that the convention being followed is to use the Z30 field name, except that if a field is valid in both the holdings and item record, the ‘Z30-' prefix is omitted.

Field Name	HOL Equivalent
Z30-BARCODE	none
SUBLIBRARY	852$b
Z30-MATERIAL	none
Z30-ITEM-STATUS	none
Z30-CATALOGER	none
Z30-ALPHA	none
COLLECTION	852$c
CALL-NO-TYPE	852 first indicator
CALL-NO	852$hijklm
Z30-CALL-NO-2-TYPE	none

Field Name	HOL Equivalent
Z30-CALL-NO-2	none
Z30-DESCRIPTION	none
Z30-NOTE-OPAC	none
Z30-NOTE-CIRCULATION	none
Z30-NOTE-INTERNAL	none
Z30-INVENTORY-NUMBER	none
Z30-PAGES	none
Z30-ITEM-STATISTIC	none
Z30-ITEM-PROCESS-STATUS	none
COPY	852$t
Z30-TEMP-LOCATION	none

Note that the Z30-CATALOGER is automatically generated by the batch job.

Following are the fields specific to holdings records:

Field Name
866-IND (866 indicators)
866-ALPHA
866-TEXT
867-IND (867 indicators)
867-ALPHA
867-TEXT
868-IND (868 indicators)
868-ALPHA
868-TEXT

Currently the program does not send textual holdings to different holdings records attached to the same bibliographic record. If the values are present they are inserted into all the holdings records created.

Note that it is possible to define fields to add to the holdings records by adding the "HOL-" prefix to the field codes. For example:

```
HOL-OWN                                 Y a          OWN   a
HOL-852                                 Y a          950   l
HOL-866                                 Y            040
```

In the above example, the service:
adds the OWN field to the holdings record with subfield $a based on the OWN field, subfield a, of the bibliographic record
adds subfield $a to the 852 field of the holdings record based on subfield $l of the 950 field of the bibliographic record
maps the 040 field of the bibliographic record to the 866 field of the holdings record.

If the field specified in column 1 already exists, the data extracted from the bibliographic record is added to the existing field.

In addition, the following options have been added to the table:

CREATE-ITEM-1ST (for first indicator)
CREATE-ITEM-2ND (for second indicator)
CREATE-ITEM-SUB (for subfields)

These options are used to define conditions where only the holdings record is created without an item. For example, if the table contains the following line, then the item is only created if the first indicator of the main field is 0:

```
CREATE-ITEM-1ST         0
```

If the table contains the following line, then the item is only created if the main field contains subfield $p:

```
CREATE-ITEM-SUB         p
```

Note that it is possible to define that, for example, the call number field should be created from different fields from the bibliographic record.

Following are sample lines for the creation of the call number from various fields:

```
CALL-NO         k       Y k
CALL-NO         h       Y h     099  a      090  a
```

In the above example,
subfield $k of the item's Z30-CALL-NO field and the holdings 952 field should be created from subfield $k of the main field.
Subfield $h of the item's Z30-CALL-NO field and the holdings 952 field should be created either from $h of the main field (if present), or from subfield $a of the 099 field (if it does not exists in the main field), or from subfield $a of the 090 field (if it does not exists in the 099 field).

Structure of the table:

col. 1 HOL/Z30 Field name

col. 2 Subfield in main tag

col. 3 Retain subfield labels together with the field content. For example: $$h954.32

col. 4 Destination subfields (for subfield labels retained)

col. 5 First tag

col. 6 First subfields

col. 7-10 Second and third tags and subfields

col. 11 Normalization code.

- UPPER - changes all data to uppercase
- LOWER - changes all data to lowercase

Example of the table:

```
!      1                    2        3   4          5     6    …7     8
      9     10        11
!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!-!!!!!!!!!!-!!!!!-!!!!!!…!!!!
!!!-!!!!!-!!!!!!!!!!-!!!!!
SUBLIBRARY               s          N b          LOC## b
COLLECTION                c          N c          LOC## c
Z30-CATALOGER             r          N
Z30-BARCODE               b          N
Z30-MATERIAL              m          N
Z30-ITEM-STATUS           i          N
CALL-NO                              Y hi         090## ab         050## ab
```

tab_hol_item_map

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Mapping bibliographic values into the relevant holdings and/or items values

This table is an example of a table that can be used by the p-manage-50 batch process for the mapping of the supplied bibliographic values into the relevant holdings and/or items values.

The list of valid fields is as follows. Note that the convention being followed is to use the Z30 field name, except that if a field is valid in both the holdings and item record, the ‘Z30-' prefix is omitted.

Field Name	**HOL Equivalent**
Z30-BARCODE	none
SUBLIBRARY	852$
Z30-MATERIAL	none
Z30-ITEM-STATUS	none
Z30-CATALOGER	none
Z30-ALPHA	none
COLLECTION	852$c
CALL-NO-TYPE	852 first indicator
CALL-NO	852$hijklm
Z30-CALL-NO-2-TYPE	none
Z30-CALL-NO-2	none
Z30-DESCRIPTION	none
Z30-NOTE-OPAC	none
Z30-NOTE-CIRCULATION	none
Z30-NOTE-INTERNAL	none
Z30-INVENTORY-NUMBER	none
Z30-PAGES	none
Z30-ITEM-STATISTIC	none

Field Name	HOL Equivalent
Z30-ITEM-PROCESS-STATUS	none
COPY	852$t
Z30-TEMP-LOCATION	none

Following are the fields specific to holdings records:

Field Name
866-IND (866 indicators)
866-ALPHA
866-TEXT
867-IND (867 indicators)
867-ALPHA
867-TEXT
868-IND (868 indicators)
868-ALPHA
868-TEXT
HOL-OWN (for the mapping of the holdings record OWN value)

Structure of the table:

col. 1 HOL/Z30 Field name

col. 2 Case-sensitive matching

col. 3 Bibliographic value;

col. 4 HOL/Z30 value

Example of the table:

```
!          1                2           3                        4
!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!
SUBLIBRARY                Y SBL1                        UEDUC
SUBLIBRARY                Y SBL2                        ULINC
SUBLIBRARY                Y MAIN                        UEDUC
SUBLIBRARY                Y main                        MAIN
CALL-NO-TYPE               N 0                           0
CALL-NO-TYPE               N #                           5
```

tab_hol_mapping

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definition of mapping values when creating records with p-manage-500

Related table(s):

1) tab01.<lng> in the HOL library

This table is an example of a table that can be used by the p-manage-500 batch process for the creation of items and/or holdings records. The table defines where in the bibliographic record item and holdings data elements are to be found.

Structure of the table:

- Col. 1 Holding Code. Value in 049 (OCLC code). Note that this value acts as a filter. If an 049 value is not found in this table, no HOL or Items will be created for that bib. The value can be wildcarded. However, you cannot setup defaults at the top of the table and specifics at the bottom
- Col. 2 HOL/Z30 Field name. Z30 always refers to location data and HOL refers to tags in the HOL record
- Col. 3 source tag/arguments. The arguments can vary. When a value is enclosed in quotation marks, the value is taken as is and mapped to the locations. When mapping 852 $$h and $$I the first set of values is subfields in BIB record and the second set is subfield in the HOL record. There can be many arguments.

Example of the table:

```
!       1                     2                    3
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!>
XXXX                 Z30-SUBLIBRARY      WID
                     Z30-COLLECTION       GEN
                     Z30-CALL-NO-TYPE     0
                     HOL-852              082##,ab,hi          086##,ab,hi
    090##,ab,hi          050##,ab,hi          099##,ab,hi          098##,ab,h
        097##,ab,hi          095##,ab,hi
```

tab_hold_request

Location of the table: tab directory of the **ADM** library

Purpose of the table: checks for hold requests

Related table(s):

1) tab15.<lng>
    a. col. 8 =N, item cannot be requested or booked (check type a)
    b. col. 12=Y, the item can have only one request (check type c)
    c. col. 8 value (Y,N,C) (check type k)
2) tab37
    d. if sublibrary and item status are not listed, item cannot be requested (check type b)
    e. if combination of sublibrary, item status and patron status are not listed, item cannot be requested (check type e)
3) tab37_booking_pickup and tab37_booking_delivery returns empty pickup list (check type c)

4) tab16
    f. col. 13, maximum number of hold requests (check type h)
    g. col. 26, maximum number of booking requests per Sublibrary & Item Status (check type h)
    h. col. 13 check for maximum number of hold requests for the sublibrary ("99" line) (check type q)
    i. col. 13 check for maximum number of hold requests for the ADM library (check type t)
5) tab100 – check type o - if there is a "like" copy available (same year, volume, part, issue, processing status, AND depending on tab100 setup
6) tab17 and tab24 – check type v - if the item belongs to a remote storage and it is available, the expected delivery time is calculated from these tables.
7) tab_booking – col. 8. Limits how far into the future booking can be made (check type l)

This table is used to define which checks should be performed by the system when a hold request is placed. The checks can influence:
display (or not) of the Request link on the list of items
display of information regarding the group of items and the patron's place in the request queue
display of meaningful messages concerning the reason why a request is refused.

There are a number of separate instances defined for when a check is made:
PRE is used when the list of items is displayed in the Web OPAC
INFO is used when displaying the hold request fill-in form
POST is used when the operator sends the request to the server
CIRC is used in the Circulation client.
CIRC-INFO is used in the Circulation client. The errors reported by this instance are always overrideable.
TITLE-REQ is used for title level requests.
FWD-TITLE is used when the active stop in a title request is changed. Queue stops whose items do not pass FWD-TITLE checks will be considered not available.
BK-PRE is used for the 'Booking' link when the list of items is displayed in the Web OPAC
BK-CIRC is used in the Circulation client
BK-POST is used when submitting the booking request fill-in form
REALLOC-BK is used when a booking request is reallocated to another like item
XML-CIRC is used when a hold request is created from a X Service
HL-GRP-PRE is used for title level hold requests ('BIB Request'). This check is performed on each ADM that has items that are related to the selected BIB record. If the ADM does not pass the check, its items will not be included in a group.
HL-GRP is used for title level hold requests ('BIB Request'). Only items that pass the check are regarded as requestable.

AB-PRE is used when the list of items is displayed in the Web OPAC. Passing this check is required to show the 'request' link next to advance booking items.
AB-POST is used when submitting an advance booking request from the OPAC or from the GUI.
REALLOC-AB is used when an advance booking request is reallocated to another like item.
UE06-PREF and **UE06-LIKE** are used by the ue_06 request handling daemon. They are relevant only when the preferred items mechanism is applied and are used in order to define whether a preferred item is available or not. UE06-LIKE is used when a like item of the original requested item is checked. UE06-PREF is used when item which is not a like item is checked. Check routines that check hold request limits (e.g. check_hold_request_h) should be defined only in UE06-PREF and not in UE06-LIKE.

NOTE: If a check that requires patron information is used as a PRE check, then the patron must first be identified, and **check_hold_request_l** must be set as a PRE check as well. In this scenario, **if the patron is not signed-in no request links will display.**
Note that some of the checks relate to the item, some to the patron and some to a combination of item-patron. Some checks are relevant only to PRE, POST or INFO.

If a check is listed for PRE, it need not be listed for INFO or POST, since the INFO/POST scenario will not happen.

When setting which checks to use, take into account the system resources required for checks that are made when the items list displays (PRE checks). If the checks require comparing item records and existing requests, and there are many item records, this can create an undesirable load on the server. If a check is listed for PRE, it need not be listed for INFO or POST, since the INFO/POST scenario will not happen.

The following checks are available:

Check	Action	Heading	Suggested use
a	if tab15.<lng> col. 8 =N, item cannot be requested	1100	POST, CIRC
b	if sublibrary and item status are not listed in tab37, item cannot be requested.	1105	PRE, POST, CIRC
c	if tab15.<lng> col. 12=Y, the item can have only one request	1120	PRE, POST, CIRC
d	Z305 hold-permission	1110	POST, CIRC

d1	checks whether the item is unavailable (requested or loaned to any patron) Item statuses that are excluded from this check are configured as parameters.
	
	For example:
	`! 1 2 3`
	`!!!!-!!!!!!!!!!!!!!-!!!!!!!!!`
	`PRE check_hold_request_d1 01,02`
		1185	PRE, POST, CIRC

e	if combination of sublibrary, item status and patron status are not listed in tab37, item cannot be requested	1115	POST, CIRC
f	Z305 multi-hold allowed	1125	POST, CIRC
g	Z305 hold-on-shelf (item availability)	1130	POST, CIRC
h	tab16 col. 13, maximum number of hold requests	1135	POST, CIRC
h_1	tab16 col. 13, maximum number of hold requests. This routine is similar to check_hold_request_h, but also includes in the check existing loans that were originally hold requests.	1135	POST, CIRC
i	if the item is on loan, displays due date and information on whether lost or recalled	6000-6002	INFO
j	if item has requests, displays on hold, patron's place in queue	6005,6006	INFO
k	tab15.<lng> col. 8 value (Y,N,C)	6010-6012	INFO
k1	Z30-description value	6016	INFO
l	user-id has not been entered	1140	PRE
m	patron has item on loan and is attempting to request a like copy	1145	POST, CIRC
n	patron tries to reserve an item that was declared as 'Lost' or 'Claimed Returned'	1095	POST, CIRC
o	checks if there is a "like" copy available (same year, volume, part, issue, processing status, AND depending on tab100 setup	1150	INFO
p	checks if all "like" copies are on loan (not available) and informs the patron that the request cannot be placed, that he must apply to the circ desk. This check should be used by a library that wants library staff to decide which copy should be requested (because recall is made for the copy requested, and not all copies)	1155	POST, CIRC

q	tab16 col. 13 check for maximum number of hold requests for the sublibrary (“99” line)	1160	POST, CIRC
q_1	tab16 col. 13 check for maximum number of hold requests for the sublibrary (line 99) This routine is similar to check_hold_request_q, but also includes in the check existing loans that were originally hold requests.	1160	POST, CIRC
r	patron expiry date has been reached	1165	POST, CIRC
s	patron has the exact same item on loan	1170	POST, CIRC
t	tab16 col. 13 check for maximum number of hold requests for the ADM library (ADM line(s) in tab16)	1175	POST, CIRC
t_1	tab16 col. 13 check for maximum number of hold requests for the ADM library (ADM line(s) in tab 16) This routine is similar to check_hold_request_t, but also includes in the check existing loans that were originally hold requests.	1175	POST, CIRC
u	Z305 reading room permission	1176	POST, INFO, CIRC
v	If the item belongs to a remote storage and it is available, the expected delivery time is calculated (using tab17+tab24). The message displays in INFO in place of the “Your place in queue” message (j)	6003	INFO
w	items that are actually loaned to a specific patron status or that have a specific item status cannot be requested. The patron and item status that are checked in combination are defined in col. 3. For example: BOR=02,08;ITEM=03		
x_1	tab16 col.13 check for maximum number of hold requests for the sublibrary's group (line 98) This routine is similar to check_hold_request_x, but also includes in the check existing loans that were originally hold requests.	1161	POST, CIRC

y	Patron with a specific status cannot request sepcific items (by item status and process status). Set up using parameters in column 3, for example: BOR=03,06;ITEM=03;PROC=MI,NA	6030	POST, CIRC
z	the hold request that is being submitted requires a recall but will not be first in the queue. This is intended to give an indication that although a recall will be made, the requesting patron will not be the first to receive the item.	1177	CIRC-INFO
sh	Applies to tab_service_hour. Checks the request submission time (day+hour) against the pickup location service hours. The request is allowed only within the service hour's timeframe.	1235	POST, CIRC, XML-CIRC
1_a	Checks patron global blocks. The error message includes block codes and a description from the tab_delinq.lng table.	6041	PRE, POST, CIRC
1_b	Checks patron local blocks. The error message includes block codes and a description from tab/tab_delinq.lng table	6051	PRE, POST, CIRC
1_d	Checks if patron has overdue loans. A loan is considered overdue if the due date has passed and "n" or more notices have been sent, as defined in the library's tab/tab100 OVERDUE-LETTER-NO.	6070	PRE, POST, CIRC
1_e	checks patron block date	6060	PRE, POST, CIRC
4_a	checks if patron has exceeded the loan limit for the item status (tab16 column 12)	6111	PRE, POST, CIRC
4_b	checks if patron has exceeded the loan limit for the sub library (tab16 column 12)	6120	PRE, POST, CIRC
4_c	checks if patron has exceeded the loan limit for the ADM library (tab16 column 12)	6130	PRE, POST, CIRC
4_d	checks if patron has exceeded the loan limit for the sub library group (tab16 column 12)	6140	PRE, POST, CIRC

5	checks if the patron has exceeded the cash limit allowed in z305_cash_limit - uses O(pen) and T(ransferred) transactions This check retrieves list of charges to include/exclude as parameter. For example: CHARGES=-0001,0006,0008,0009 Means that all charges except of 0001,0006,0008,0009 will be counted. CHARGES=+0001,0006,0008,0009 Means that only 0001,0006,0008,0009 charges will be counted. If no parameter was retrieved, all cash charges will be taken into account.	6100	PRE, POST, CIRC
5_b	Checks if the patron has exceeded the cash limit allowed. But only for O(pen) transactions. This check retrieves list of charges to include/exclude as parameter. For example: CHARGES=-0001,0006,0008,0009 Means that all charges except of 0001,0006,0008,0009 will be counted. CHARGES=+0001,0006,0008,0009 Means that only 0001,0006,0008,0009 charges will be counted. If no parameter was retrieved, all cash charges will be taken into account.	6100	PRE, POST, CIRC

9_a	checks the limits defined in tab/tab_block_circ (Maximum number of overdue loans, Maximum number of recalled overdue loans, Maximum cash owing). This check retrieves 2 parameters: • block circ table name. If not specified, the default table tab_block_circ is consulted. • list of charges to include/exclude For example: CHARGES=-0001,0006,0008,0009 Means that all charges except of 0001,0006,0008,0009 will be counted. CHARGES=+0001,0006,0008,0009 Means that only 0001,0006,0008,0009 charges will be counted. If CHARGES parameter was not retrieved, all cash charges will be taken into account. The 2 parameters should be separated by ; For example: tab_block_test;CHARGES=+0001,0006,0008,0009	6081	PRE, POST, CIRC
9_b	checks the limits defined in tab/tab_block_circ as 9_a but at item sub library level. This check retrieves block circ table name as parameters. If not specified, the default table tab_block_circ is consulted.	6091	PRE, POST, CIRC

10_a	As 9_a but counts all types of loans, i.e. active loans and claimed return. This check retrieves two parameters: • block circ table name. If not specified, the default table, tab_block_circ, is consulted. • list of charges to include/exclude For example: CHARGES=-0001,0006,0008,0009 This means that all charges except for 0001,0006,0008,0009 are counted. CHARGES=+0001,0006,0008,0009 This means that only 0001,0006,0008,0009 charges are counted. If the CHARGES parameter is not retrieved, all cash charges are taken into account. The two parameters should be separated by a semicolon (;). For example: tab_block_test;CHARGES=+0001,0006,0008,0009	6081	PRE, POST, CIRC
10_b	As 9_b but counts all types of loans, i.e. active loans and claimed return. This check retrieves block circ table name as parameters. If not specified, the default table tab_block_circ is consulted.	6091	PRE, POST, CIRC

The following checks are for booking requests only:

Check	Action	Heading	Suggested use
a	if tab15 col.16=N, item cannot be booked	1200	BK-PRE
b	checks if tab37_booking_pickup and tab37_booking_delivery returns empty pickup list	1106	BK-PRE
c	Item cannot be booked if the same item is already booked by the same patron on the same day when Z305-MULTI-HOLD=N.	1221,1222	AB-POST

Check	Action	Heading	Suggested use

d	checks whether the item is unavailable (requested or loaned to any patron) Item statuses that are excluded from this check are configured as parameters.
	
	For example:
	`! 1 2 3`
	`!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!`
	`PRE check_hold_request_d1 01,02`
		1180	BK-PRE

e	Booking span is longer than the defined loan period for this item. This check is hardcoded in requests	1250	BK-CIRC
h	tab16 col.26, maximum number of booking requests per Sublibrary & Item Status.	1135	BK-POST
h_1	tab16 col.26, maximum number of booking requests per Sublibrary & Item Status. This routine is similar to check_booking_request_h, but also includes in the check existing loans that were originally booking requests.	1135	BK-POST
k	checks if the booking period complies with tab15 policy.	1300	BK-POST
l	tab_booking, Col 8: max_booking_start_date Limits how far into the future booking can be made.	1140	BK-POST
q	tab16 – Limit of Bookings per Sublibrary	1160	BK-POST
q_1	tab16 – Limit of Bookings per Sublibrary This routine is similar to check_booking_request_q, but also includes in the check existing loans that were originally booking requests.	1160	BK-POST
r	patron expiration date will be reached	1320	BK-POST,AB-POST
t	tab16 col.13 check for maximum number of hold requests for the ADM library (ADM line(s) in tab16)	1175	BK-POST

Check	Action	Heading	Suggested use
t_1	tab16 col.26 check for maximum number of booking requests for the ADM library (ADM line(s) in tab16) This routine is similar to check_ booking _request_t, but also includes in the check existing loans that were originally booking requests.	1175	BK-POST
x_1	tab16 col.26 check for maximum number of hold requests for the sublibrary's group ("98" line) This routine is similar to check_ booking _request_x, but also includes in the check existing loans that were originally booking requests.	1161	POST, CIRC

Structure of the table:

col. 1 defines when check is performed:
PRE, POST, CIRC, INFO…

col. 2 defines check procedure to perform – as listed in the table above. The check code should be entered as per the following format
check_hold_request_a/b/c..

col. 3 Program arguments

Example of the table:

```
! 1                    2                    3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!…!!!!!!!!!
PRE         check_hold_request_a
PRE         check_hold_request_b
PRE         check_hold_request_d
PRE         check_hold_request_c
PRE         check_hold_request_l
PRE         check_hold_request_n
```

tab_hold_request_form

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defining hold request forms for use in the OPAC.

Related table(s):

1) tab100 - AVAILABILITY-ROUTINE
2) tab_sub_library.<lng>
3) tab15.<lng>
4) tab31

This table sets the option to configure the system to use different hold request forms when the patron places a request through the Web OPAC. When a request is placed, the system will look for a matching line in the table. The correct line will be selected according to one of the available parameters in this table:
the item's sublibrary, -
item status,
item process status,
patron status and
item's availability that is determined according to the -AVAILABILITY-ROUTINE flag in tab100.

If no match is found, or the table does not exist, the system will display the regular hold request forms. The same happens if the table does not exist.

Note that specific lines should come before the general ones. For example:

```
##### 32 DP ## # depo32
##### ## DP ## # depo
```

The forms that can be changed are:
item-hold-request-body
item-hold-request-head
item-hold-request-tail
booking-page-delivery-form-head
booking-page-pickup-form-head
booking-page-return-pickup-head
booking-page-return-delivery-head
booking-page-form-body
booking-page-return-delivery-tail
booking-page-delivery-form-tail
booking-page-pickup-form-tail
booking-page-return-pickup-tail

All of these forms need to be set with the defined suffix for the system to use them if the request placed match the settings in the table.

Structure of the table:

Col. 1 SubLibrary. Use ## to denote any (all) sublibrary

Col. 2 Item status. Use ## to denote any (all) item status

Col. 3 Item Process status

Col. 4 Patron status. Use ## to denote any (all) patron status

Col. 5 Item Availability Status (Using tab100AVAILABILITY-ROUTINE)

- Y = Available item
- N = Non-available item

Col. 6 WWW Form suffix

Example of the table:

```
! 1   2  3  4  5    6
!!!!!-!!-!!-!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
##### 70 DP ## # depo
##### 71 DP ## # depo
```

tab_photo_request_form

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defining photocopy request forms for use in the OPAC.

Related table(s):

1) tab100 - AVAILABILITY-ROUTINE
2) tab_sub_library.<lng>
3) tab15.<lng>
4) tab31

This table sets the option to configure the system to use different photocopy request forms when the patron places a request through the Web OPAC. When a request is placed, the system looks for a matching line in the table. The correct line is selected according to one of the following available parameters in this table:

- The item's sublibrary
- Item status
- Item process status
- Patron status
- The item's availability that is determined according to the - AVAILABILITY-ROUTINE flag in tab100.

If no match is found or the table does not exist, the system displays the regular photocopy request forms.

Note that specific lines should come before the general ones. For example:
32 DP ## # depo32
DP ## # depo

The following forms can be changed:
- item-photo-request-body
- item-phot-request-head
- item-photo-request-tail
- item-photo-request-tail-an

Set these forms with the defined suffix so the system can use them if the request placed matches the settings in the table.

Structure of the table:

Col. 1 SubLibrary. Use ## to denote any (all) sublibrary

Col. 2 Item status. Use ## to denote any (all) item status

Col. 3 Item Process status

Col. 4 Patron status. Use ## to denote any (all) patron status

Col. 5 Item Availability Status (Using tab100AVAILABILITY-ROUTINE)

- Y = Available item
- N = Non-available item

Col. 6 WWW Form suffix

Example of the table:

```
! 1   2  3  4  5    6
!!!!!-!!-!!-!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
##### 70 DP ## # depo
##### 71 DP ## # depo
```

tab_ill_bib_info

Location of the table: tab directory of the ILL library

Purpose of the table: Defintions for ILL document tags that builds the ISO ILL Request

This table defines the ILL document tags that builds the ISO IllRequestItemId xml tags. Note that default values for building the ItemId tag are set in the code (e.g.: field tag 260##$$a is directed to ItemIdPlaceOfPublication).

Edit this table only if you want to override these hard-coded values and define your preferred set.

For each XML tag (e.g., AuthorOfArticle) you may define up to two sets of tag, subfield and filing procedure. In case the first set (columns 2-4) fails (i.e. there is no such tag in the ILL document), the system uses the second set (coumns 5-7).

Structure of the table:

Col. 1 XML tag
col. 2 1st tag
col. 3 1st subfields
col. 4 1st filing
col. 5 2nd tag
col. 3 2nd subfields
col. 4 2nd filing

Example of the table:

```
   5             6          7
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!-
!!!!!!!!!!!!!!!!!!!!-!!-!!!!!-!!!!!!!!!!!!!!!!!!!-!!
ItemType                                            592## a
Author                                              100## a
Title                                               245## a
SubTitle                                            245## b
PlaceOfPublication                                  260## a
23
Publisher                                           260## b
23
PublicationDate                                     260## c
24 ISS## y
```

tab_ill_bib_key

Location of the table: tab directory of the ILL library

Purpose of the table: Defines ILL Server Library document tags

Structure of the table:

Col. 1 Well known no.
col. 2 Well known code
col. 3 SysNo Type
col. 4 xml tag
col. 5 xml subfields
col. 6 xml filing
col. 7 db tag
col. 8 db subfields
col. 9 db filing

Example of the table:

```
!1      2        3     4   5 6    7   8 9
!!-!!!!!!!!!!-!!!!!-!!!!!-!-!!-!!!!!-!-!!
01 LC          SYSNO 010   a
06 RLIN        NBN   035## a
09 ALEPH_SYS   SYSNO SID   c    SID   b
```

tab_ill_change_status_out

Location of the table: alephe/tab directory

Purpose of the table: Define for which borrowing request status the action 'Change Status' is allowed, to which status the request may be changed, and possible additional actions that will be carried out when the status change is done.

Related table(s):

1) tab_ill_change_status_in

This table defines for which outgoing request status (col.1) the action 'Change Status' is allowed. The statuses to which the request can be changed to are listed in Col.2. The procedure that is activated is defined in Col.3.
In case there are a few statuses to which you want to allow updating a specific request status, define a few lines for the same status. For example, if you want to allow update of NEW request to CA (Cancelled) and DEL (Deleted), set two separated lines as follows:
NEW CA
NEW DEL

Col 1. The original status of the changed request.
Col 2. The status to which the request may be changed
Col 3. The additional action that will be performed. The ill_update_out_sup_status procedure may be used to update the active Z411 record with the same status as the Z410 record.

Example of the table:

```
!1   2           3
!!!-!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
NEW CA
NEW DEL
SV  CA  ill_update_out_sup_status
WPR SV  ill_update_out_sup_status
```

tab_ill_change_status_in

Location of the table: alephe/tab directory

Purpose of the table: Define for which lending request status the action 'Change Status' is allowed, to which status the request may be changed, and possible additional actions that will be carried out when the status change is done.

Related table(s):

1) tab_ill_change_status_out

This table defines for which outgoing request status (col.1) the action 'Change Status' is allowed. The statuses to which the request can be changed to are listed in Col.2. The procedure that is activated is defined in Col.3.
In case there are a few statuses to which you want to allow updating the specific request status, define a few lines for the same status. For example, if you want to allow update of NEW request to CLS (Closed) and CBR (Cancelled By Requester), set two separated lines as follows:
NEW CLS
NEW CBR

Col 1. The original status of the changed request.
Col 2. The status to which the request may be changed
Col 3. The additional action that will be performed.

Example of the table:

```
!1   2           3
!!!-!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
AUF CLS
AUF DEL
NEW CLS
NEM DEL
```

tab_ill_held_media_map.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Translates the ISO IllRequest.ItemId.HeldMediumType code for an incoming ILL request, to a textual description.

Structure of the table:

Col. 1 IllRequest.ItemId.HeldMediumType code

Col. 2 Textual Description

Example of the table:

```
!-!!!!!!!!!!!!!!!!!!!!!!!!!
1 Printed
3 Microform
4 Film-or-Video
5 Audio
6 Machine-Readable
7 Other
```

tab_ill_partner_conf

Location of the table: alephe/tab directory

Purpose of the table: Defines user names and passwords for ILL suppliers that require password in the Prompt1 extension, such as OCLC.

Structure of the table:

Col. 1 Supplier code.

col. 2 User ID

col. 3 Password

Example of the table:

```
!        1               2                   3
!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!
U15ED                U15EDOCLCUSER        U15EDPW123
```

tab_ill_preferred_media

Location of the table: tab directory of the ILL library

Purpose of the table: Possible media types for each ILL format

Related table(s):

1) ill_format_type

This table is used to define the possible media types that can be requested for each format. The format is the first occurrence of the 592 field in the ILL record. The 592 field is determined by the ill_format_type table.

The list of preferred media is displayed in the Web OPAC ILL Request Form in the 'Preferred Media' Field.

Note: The codes which are listed in Col.2 through 16 CAN ONLY be one of the following:

- PRINTED
- COPY
- MICROFORM
- VIDEO
- AUDIO
- MR
- OTHER

And can be used only with the prefix L-, C- or E-.

Structure of the table:

Col. 1 Format Type Code, as defined in ill_format_type

Col. 2-16 Code of Preferred Media Type

Example of the table:

```
!1    2               3               4               5               6
7
!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!
!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!
!!!!!!!!!!-!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!
MN L-PRINTED       C-PRINTED       L-COPY          C-COPY          L-
MICROFORM     C-MICROFORM     E-COPY
SE C-COPY          C-MR            L-MR            E-COPY          L-
PRINTED
## L-PRINTED       C-PRINTED       L-COPY          C-COPY          L-AUDIO
C-AUDIO         L-VIDEO         C-VIDEO         L-OTHER         C-OTHER
E-COPY
```

tab_ill_status_borrow

Location of the table: alephe/tab directory

Purpose of the table: Defines how outgoing requests will be grouped by status in the Summary View of the Borrowing tab of the ILL GUI

The textual description of the group and its statuses is derived from the BOR-REQ-STATUS section of xxx40/tab/pc_tab_exp_field.<lng>

Structure of the table:

Col 1 The group code

Col 2 -20 The statuses that belong to the group

Examle of table:

```
!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-
!!!-!!!-!!!-!!!-

GNW NEW NEM LOW

GWP WAP LOF DSF

GPD WSP SV  WPR HPL

GSP SHP

GUF UNF

GCO CRP RTY EST LCR

GRC LON LOP DMG

GRT RTP RT

GRN RNR RNA WRN

GRL RCL

GOD OVD

GLT LST

GCN WCR CA

GCS CLS

GEP EXP
```

tab_ill_status_lend

Location of the table: alephe/tab directory

Purpose of the table: Defines how incoming requests will be grouped by status in the Summary View of the Lemding tab of the ILL GUI

The textual description of the group and its statuses is derived from the LEND-REQ-STATUS section of xxx40/tab/pc_tab_exp_field.<lng>

Structure of the table:

Col 1 The group code

Col 2 -20 The statuses that belong to the group

Examle of table:

```
!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-!!!-
!!!-!!!-!!!-!!!-

GRV NEW NEM NEP LOC LPR MED MLC MLP REF

GIP AWS AHP

GSP SL  SC  RCV

GRT RT

GCN CRA ACN

GCA CBR

GOD OVD

GRL RCL

GLT LST

GDG DMG

GNP ART AUF ALC AES EXP

GCS CLS
```

tab_ill_supp_close_days

Location of the table: alephe/tab directory

Purpose of the table: Specify ILL suppliers' non-working days

This table is used to define the non working days of the system's ILL suppliers, which may be either internal ILL units or external suppliers. The schedule that is set up in this tables effects the calculation of the outgoing request's expiry date.

The table may be set to define weekdays independently of the date, dates independently of the weekday, or a combination of date and weekday.

Structure of the table:

Col. 1 ILL Supplier code
Col. 2 Date
Col. 3 Day of Week (00-Sunday - > 06-Saturday)
Col. 4 Status ("O" – Open or "C" Closed)

Example:

```
! 1                    2       3  4
!!!!!!!!!!!!!!!!!!!-!!!!!!!!-!!-!
#################### ######## 00 C
#################### ######## 06 C
ILL_LAW              ####1222    C
ILL_LAW              ####1223    C
ILL_LAW              ####1224    C
ILL_LAW              ####1225    C
ILL_LAW              ####1225    C
```

tab_ill_tree_bor_msg

Location of the table: alephe/tab directory

Purpose of the table: Defines how messages of outgoing requests will be grouped in the Summary View of the Borrowing tab of the ILL GUI

The textual description of the group and its statuses is derived from the BOR-REQ-MSG section of xxx40/tab/pc_tab_exp_field.<lng>

Structure of the table:

Col 1 The group code

Col 2 -21 The messages that belong to the group

Examle of table:

```
! 1    2  3  4  5  6  7  8  9 11 12 13 14 15 16 17 18 19 20 21
!!!!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!
G-MSG 14 20 29 30
```

tab_ill_tree_lend_msg

Location of the table: alephe/tab directory

Purpose of the table: Defines how messages of incoming requests will be grouped in the Summary View of the Lemding tab of the ILL GUI

The textual description of the group and its statuses is derived from the LEND-REQ-MSG section of xxx40/tab/pc_tab_exp_field.<lng>

Structure of the table:

Col 1 The group code

Col 2 -21 The messages that belong to the group

Examle of table:

```
! 1    2  3  4  5  6  7  8  9 11 12 13 14 15 16 17 18 19 20 21
!!!!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!-!!
G-MSG 19 13 23 28 31
```

tab_io_remote

Location of the table: alephe/tab directory

Purpose of the table: Remote library specifications

This table is used when a particular library (database) is in a remote server. For example, if the authorities library is on another server, there is a need to define to which server the authority library belongs.

Structure of the table:

Col. 1 Library code
col. 2 Hostname
col. 3 ALEPH version

Example:

```
!   1                2                              3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!
LOC01 ram43:6530
LOC50 ram43:6530
LOC60 ram43:6530
```

tab_item_history.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Item History trigger and description

Related table(s):

1) tab100 - CREATE-Z30H

This table defines in which cases a field change or an action will cause an item history record to be written. This information is accessible via the History button in the Items module.

In addition, the table is also used for setting the description of the action. This description is used for display in the item history window.

The actions which can trigger a history record are changes in:

ITEM-LOCATION	L Call No.
ITEM-CALL-NO-2	L Call No.2
ITEM-COLLECTION	L Collection
ITEM-SUB-LIBRARY	L Sublibrary
ITEM-PROCESS-STATUS	L Process Status
ITEM-BARCODE	L Barcode

```
BIND                          L Deleted (Bound)
DELETE                        L Deleted
ITEM-STATUS                   L Status
SHELF-REPORT-NUMBER           L Inventory Check
ITEM-MATERIAL                 L Material Type
ITEM-DESCRIPTION              L Item Desc.
ITEM-CIRC-NOTE                L Circ Note
ITEM-INT-NOTE                 L Internal Note
ITEM-OPAC-NOTE                L OPAC note
ITEM-LOCATION-TYPE            L Location Type
! ITEM-CALL-NO-2-TYPE         L Location2 Type
! ITEM-COPY                   L Copy Number
! ITEM-ENUM-A                 L Enum Level 1
! ITEM-ENUM-B                 L Enum Level 2
! ALL-NON-CIRC                L All non-circ
  L All fields
```

Use "ALL-FIELDS" in order to register a change in any other field. If using ALL-FIELDS, it should be listed last in tab_item_history.lng.
If ALL-FIELDS is listed before specific changes, changing a listed field will register both "ALL-FIELD" and the specific field.

Use "ALL-NON-CIRC" in order to register all changes except of item (Z30) changes caused as a result of item's circulation activities:
z30-date-last-return
z30-hour-last-return
z30-ip-last-return,
z30-no-loans
z30-maintenance-count

If using ALL-NON-CIRC, it should be listed last in tab_item_history.lng.
Make sure NOT to set both "ALL-FIELDS" and "ALL-NON-CIRC" in the table.
If ALL-NON-CIRC is listed before specific changes, changing a listed field will register both "ALL-NON-CIRC" and the specific field.

Structure of the table:

- Col. 1 Field changed or action that causes history record to be written
- Col. 2 ALPHA
- Col. 3 Description of change made to the item (15 characters only)

Example of the table:

```
!                 1                          2       3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!-
ITEM-SUBLIBRARY                             L Sublibrary
ITEM-BARCODE                                 L Barcode
ITEM-PROCESS-STATUS                          L Process Status
ITEM-LOCATION                                L Call Number
…
```

tab_item_list_order

Location of the table: tab directory of the **BIB** library

Purpose of the table: Display order of the items

Related table(s):

1) tab_sub_lib_sort– col. 2.
2) tab_item_list_order

This table determines the display order of the items in the Web OPAC and GUI Search functions according to the sublibraries for each base.

It is possible to define a different display order for each base. If you create a table for each base, the table name should include the base name as an extension of the table name (for example, tab_item_list_order.serials for the logical base "serials"). In the event that there is no table for a base, the program uses the "tab_item_list_order" table.

NOTE: The item order from this table will be consulted only when "02" is defined in tab_sub_lib_sort column 2.

It is possible to configure the display of PST/LOC fields according to the sublibrary in the custom brief display in the Web. Note that if there is only one sublibrary, tab_item_list_order is not consulted at all (the same as in ITM3).

To activate this, the following line should be added to tab_sub_lib_sort - WWW-FULL-LOC-PST 02.

Structure of the table:

col. 1 sublibrary

Example of the table:

```
!  1
!!!!!-!!!!!!!!

UHLTH
ULAW
ULINC
UMUSI
USCI
UARCV
```

tab_itm_mapping

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definition of mapping values when creating records with p-manage-500

Structure of the table:

Col. 1 Z30 Field name
Col. 2 Alpha
Col. 3 Source value
Col. 4 Target value

Example of the table:

```
!   1                      2       3                        4
!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!
Z30-MATERIAL                                          BOOK
Z30-ITEM-STATUS                                       01
Z30-ALPHA                                             L
Z30-DESCRIPTION                                       Z30-DESCRIPTION
Z30-NOTE-OPAC                                         Z30-NOTE-OPAC
Z30-NOTE-CIRCULATION                                  Z30-NOTE-CIRCULATI
Z30-NOTE-INTERNAL                                     Z30-NOTE-INTERNAL
Z30-PAGES                                             300
Z30-ITEM-STATISTIC                                    ITEM-STATISTIC
```

tab_label.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: label prefixes for spine labels

Related table(s):

1) tab_sub_library.<lng>
2) tab40.<lng>

This table defines the prefixes that will print on the spine label. The number of lines that can be defined in this table is 2000

Structure of the table:

Col. 1 Sublibrary code as defined in tab_sub_library.<lng>
Col. 2 Collection code as defined in tab40.<lng>
Col. 3 Prefix (or suffix) delimited by '**;**' (semi-colon) which denotes line feed

For information on how suffixes are handled refer to the document, “How To Set Up Label Printing.”

Example of the table:

```
! 1     2            3
!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!>
UGDOC GEN   UGDOC;General
UHLTH REF   UHLTH;Reference
UMUSI       UMUSI;
UEDUC GEN   UEDUC;General
```

tab_label_parse

Location of the table: tab directory of the **ADM** library

Purpose of the table: Parsing call numbers for label printing

This table specifies the routines used to parse (split) the call number and item description when generating item labels.

Column 1 specifies the field to be parsed.
C = call number
D = item description

Columns 2, 3, and 4 are used only if column 1 is set to C.

Column 2 specifies the call number type to match.

Columns 3 and 4 specify an optional subfield and contents to filter the call number for.

> For example:
> if '2' and '[mesh]' are listed in these columns, the parsing routine in column 6 will be used only if the call number type matches and the call number contains a subfield $$2 [mesh].
> Note that the subfield contents must be entered in lowercase.

Column 5 is used only if column 1 is set to D. It specifies the item material type to filter for. If left blank, acts as a default entry for parsing item descriptions.

Column 6 specifies the routine to use to parse the call number or item description.

Programs for parsing call numbers (type C):
parse_call_no_default
parse_call_no_lc_1
parse_call_no_lc_2
parse_call_no_lc_3
parse_call_no_dw_1
parse_call_no_852_form

The first routine is general, and breaks on a space or subfield i.
The other routines include breaking on space or subfield i, together will additional features.
The three ‘lc’ routines are intended to be used with LC (Library of Congress) and NLM (National Library of Medicine) call numbers.

Examples for the routines are given, using
$$hKJV444.21804 A7$$iL63 1805 and
$$hG635.H4$$iA3 1989^a

parse_call_no_default:
splits the call number at spaces and subfield i.

If the character ^ is present in the call number, converts it to a space but does not break at that point.

parse_call_no_lc_1:
In addition to parse_call_no_default, breaks between the letter and number components of the classification part of the call number (even if there is no space) and breaks on a decimal preceding a letter (even if it is not preceded by a space). For example:

KJV	G
444.21804	635
A7	.H4
L63	A3
1805	1989 a

parse_call_no_lc_2:
In addition to parse_call_no_lc_1, when a line would end up longer than eight characters, it will be split on a ".". For example:

KJV	G
444	635
.21804	.H4
A7	A3
L63	1989 a

parse_call_no_lc_3:
The same as parse_call_no_lc_2, except that no space is added to force a line break after the class code. For example:

KJV444	G635
.21804	.H4
A7	A3
L63	1989 a
1805	

parse_call_no_dw_1:
This is a site-specific parsing routine.
Break at the end of the first string of numerals terminated with a space or a full stop. The full stop remains on the first line.
Then the following rules apply :

- Non-periodicals: Start a new line after every three numeric characters or after every space (whichever occurs first).
- Periodicals: All call numbers that begin with P (other than those beginning with P/) and end with either a suffix consisting of a five-digit string or a five-digit string followed by a decimal point and a two-digit string. This suffix breaks at the decimal point, or after the five-digit string. All characters/digits preceding the suffix for these "P" call numbers are handled as non-periodicals.

Note: if no matching program is specified, or if the program does not exist, parse_call_no_default will be used.

parse_call_no_852_form:
This parse routine is relevant for versions 21 and later.
This is a site-specific parsing routine to handle multiple consecutive $$i fields as one $$i field. It splits the call number for each subfield, space, and "^".

Programs for parsing item descriptions (type D):

parse_desc_default:
splits the item description at spaces.

parse_desc_niso_8
parse item descriptions formatted according to ANSI/NISO Z39.71, ("Holdings Statements for Bibliographic Records"). It also attempts to limit each line of the item description to 8 characters.
This routine uses the following algorithm:
remove any trailing information in parentheses, square brackets, or angle brackets
break at "+" or "&", which indicate bibliographic units, retaining the "+" or "&"
within a bibliographic unit, break on ":" but do not retain the colon in the label
within a level, break on "," and ";", retaining the punctuation
if a segment produced by (d) is longer than 8 characters, break up to two times on the "-" and "/" characters, retaining them on the label
if more than ten lines of item description would be produced based on (b)-(e), append the rest of the item description to the tenth line, even if it ends up longer than 8 characters.

Example: the item description
v.120:no.1:pt.A-B,D + Index + Supplement (1998)

would be parsed into
v.120
no.1
pt.A-B,
D
+ Index
+ Supplement

parse_desc_semicolon
If a semicolon is present in the description, split on semicolons. Otherwise, split it on commas.

parse_desc_semi_no_parens:
same as parse_desc_semicolon except that except that only the part of the item description up to the first (, [, or < is used.

Note: if no matching program is specified, or if the program does not exist, parse_call_no_default will be used.

Structure of the table:

Col. 1 Item field
Col. 2 Call number type to match
Col. 3 Filter subfield
Col. 4 Filter subfield contents
Col. 5 Item material type filter
Col. 6 Parsing program

Example of the table:

```
! 2 3   4          5     6
!-!-!-!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
C 0                    parse_call_no_lc_1
C 8                    parse_call_no_default
C 1                    parse_call_no_dw_1
```

tab_late_return

Location of the table: tab directory of the **ADM** library

Purpose of the table: Time and rate for fine method "F"

Related table(s):

1) tab16 – col. 14
2) tab_sub_library.<lng>
3) tab15.<lng>
4) tab31

This table defines late return periods and amounts for late return fines, when the fine method **F** is set in col. 14 in tab16. Using method "F" a set rate is charged for a time period.

For example:
A fine is set for 5 Euro for the first 10 days and 50 Euro for the next 5 days. If an item is returned any time within the first 10 days, the fine charged will be 5 Euro. If the item is returned any time from day 11 through 15, the fine charged will be 50 Euro.

This number of lines in the table is limited to 1500.

Structure of the table:

Col. 1 Sublibrary as defined in tab_sub_library.<lng>
Col. 2 Item status as defined in tab15.<lng>. Use ## to denote any (all) item status.
Col. 3 Patron status as defined in tab31.Use ## to denote any (all) patron status.
Col. 4 Number of days since due date
Col. 5 Sum: left-aligned

Example of the table:

```
! 1   2  3   4       5
!!!!!-!!-!!-!!!!-!!!!!!!!!!
##### ## ## 0000 1.00
##### ## ## 0011 2.50
##### ## ## 0021 5.00
##### ## ## 0031 10.00
```

tab_library_group

Location of the table: alephe/tab directory

Purpose of the table: Grouping of libraries to share the same batch processes.

The table defined a group of libraries to share the same lib_batch and UE processes.

Note: if your library is managed by another library using a tab_library_group definition, the utilities will not allow you to start or stop jobs.

A message will appear, for example in UTIL/C/2:

*** util_c_02 - start USM10 batch queue ***
lib_batch for USM10 is managed by USM01, aborting...

Structure of the table:

Col. 1 Hosting library;

Col. 2- 9

Member libraries

Example of the table:

```
! 1     2     3     4     5     6     7     8     9
!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!>
USM01 USM10
USM11 USM12 USM14 USM15 USM19
USM30
USM50 USM51 USM52 USM53
USM60
```

tab_loader

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines processing and policy for the loaders services. It refers to the creation of the holding records, items, orders, budget transactions, and load information.

tab_loader is used by the followings load services:

- Advanced Generic Vendor Records Loader (file-90)
- Load OCLC Records (file-93)

- Load MARCIVE Records (file-99)
- OCLC server

The maximum number of lines is 1000.

Related table(s):

1) tab_loader_def

Structure of the table:

Col 1. 049 Code – Holding code in tag 049$$a of incoming BIB record. If 049$$a is missing from the BIB input, the value entered in the batch service is taken into account.

Col 2. OWN code for the bibliographic record.– Optional – The data entered in this column will be used for the OWN field of records added/updated by the OCLC server processes.

Col 3. OWN Group code – Optional, this column is used in order to group different OWN codes to a common code. This will be used when a library uses different 049 codes, and it creates a different OWN code for each 049 code. If no OWN group is defined, a separate BIB record will be created for each separate OWN code. If several OWN codes are assigned to the same OWN group, each OWN code is added to a single shared BIB record. The OWN group code is not written in the BIB record, it is used only in this table for grouping.

Col 4. Item creation method. Possible values:

0 – Do not create items. This is the default value.

1 – Create item for new records only. The system will generate the barcode.

2 – Create first and additional items using the definitions in 949/852 fields. If field is missing (or barcode is missing) the system will generate a barcode.
Create items even though BIB already has items.

3 – Create first items using the definitions in 949/852 fields. If field is missing or barcode is missing) the system will generate a barcode.
Do not create item if BIB already has items.

4 – Create first and additional items using the definitions in 949/852 fields. If field is missing (or barcode is missing) don't create item. In this case ADM record will be Created.
Create items even though BIB already has items.

5 – Create first items using the definitions in 949/852 fields. If field is missing (or barcode is missing) do not create items. In this case ADM record will be created.
Do not create items if BIB already has items.

Item creation in the load is based on two fields in the BIB record:

- Field 949: $$a (for barcode), $$c (for Shelf list)
- Field 852: $$p (for barcode), $$a (for Shelf list)

An item is created for every 949/852 field+Shelf list occurrence.
For the system generated barcodes, the program looks for CSCR-OCLC-Z30-BARCODE in tab_checksum

Col 5. HOL record creation method.

0 – Do not create HOL record. This is the default value.

1 – Create the first HOL for new records only.

2 – Create HOL records (check uniqueness with 852 $$b + $$c -if matching HOL record exists, a new one will not be created). Add information that is in brackets in 049 to 852 subfields k,m

3 – Create HOL records (check uniqueness with 852 $$b + $$c –if matching HOL record exists, a new one will not be created). Do not add information that is in brackets in 049 to 852;

4 - Used if the tab100 UNION-LIBRARY variable is set to 3

5 – Always create new HOL record.

The information for the HOL record is located in the ADM library table tab_mapping.

Col 6. – Order record creation method

0 – Do not create Order record. This is the default value.
1 – Create the first Order for new ADM record, or additional order for existing ADM record.

Col 7. – Budget transaction creation method
0 – Do not create a Budget transaction record. This is the default value.
1 – Create new Budget transaction records.

Col 8. – Multi match bibliographical record load method
0 – Do not create bibliographical record. This is the default value.
1 – Create new bibliographical record.

Example of the table:

```
    1          2          3      4 5 6 7 8
!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!-!-!-!-!
NO-OWN                             3 3 1 1 0
WID        MASTER50                0 0 0 0 0
CUV7       MASTER50                1 1 1 0 1
```

The information for the holding records is located in tab_mapping table.
The information in the table cannot contain hashes (#).

tab_loader_def

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines default values for fields in the items (Z30), orders (Z68), and budget transaction records (Z601) that are created using the loaders services.

tab_loader_def is used by the followings load services:

- Advanced Generic Vendor Records Loader (file-90)
- Load OCLC Records (file-93)
- Load MARCIVE Records (file-99)
- OCLC server

Related table(s):

1) tab_loader

tab_loader defines processing regarding the creation of the holding records, items, orders, budget transactions, and load information. The default information for the holding record should be populated in tab_mapping table.

Structure of the table:

Col 1 Z30/Z68/Z601 field name.

Col 2. Match 049 value in the incoming bibliographic record or hashes to indicate that the value in column 3 will be used regardless of the bibliographic record.

Col 3. Value to place in the field defined in column 1

Example of the table:

```
        1                            2          3
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!>
Z30-SUB-LIBRARY                 BXMM       WID
Z30-SUB-LIBRARY                 BXMA       MED
Z30-SUB-LIBRARY                 YBPDD      WID
Z30-SUB-LIBRARY                 ########## WID
Z30-COLLECTION                  BXMM       GEN
Z30-COLLECTION                  BXMA       GEN
Z30-COLLECTION                  ########## GEN
Z30-MATERIAL                    ########## BOOK
Z30-ITEM-STATUS                 ########## 01
```

```
Z30-CATALOGER                  ########## MASTER
Z30-CALL-NO-TYPE               ########## 0
Z30-ITEM-PROCESS-STATUS        ########## OR

Z68-SUB-LIBRARY                BXMM       WID
Z68-SUB-LIBRARY                BXMA       MED
Z68-SUB-LIBRARY                YBPDD      WID
Z68-SUB-LIBRARY                ########## WID
Z68-ORDER-TYPE                 ########## S
Z68-ORDER-STATUS               ########## SV
Z68-ARRIVAL-STATUS             ########## P

Z601-VENDOR-CODE               ########## YBP
Z601-USER-NAME                 ########## MASTER
Z601-CURRENCY                  ########## USD
Z601-TYPE                      ########## ENC
Z601-ORIGINAL-SUM              ########## 0010
```

Column 1 in tab_loader_def can contain the following values (Bold fields are mandatory):

For Z30 ITEM creation:

Z30-SUB-LIBRARY
Z30-COLLECTION
Z30-MATERIAL
Z30-ITEM-STATUS
Z30-CATALOGER
Z30-NO-LOANS
Z30-ALPHA
Z30-CALL-NO-TYPE
Z30-CALL-NO
Z30-CALL-NO-KEY
Z30-CALL-NO-2-TYPE
Z30-CALL-NO-2
Z30-CALL-NO-2-KEY
Z30-DESCRIPTION
Z30-NOTE-OPAC
Z30-NOTE-CIRCULATION
Z30-NOTE-INTERNAL
Z30-INVENTORY-NUMBER
Z30-INVENTORY-NUMBER-DATE
Z30-LAST-SHELF-REPORT-DATE
Z30-PRICE
Z30-DOC-NUMBER-2
Z30-SCHEDULE-SEQUENCE-2
Z30-COPY-SEQUENCE-2
Z30-VENDOR-CODE
Z30-INVOICE-NUMBER
Z30-LINE-NUMBER
Z30-PAGES
Z30-ISSUE-DATE
Z30-EXPECTED-ARRIVAL-DATE

Z30-ARRIVAL-DATE
Z30-ITEM-STATISTIC
Z30-ITEM-PROCESS-STATUS
Z30-COPY-ID
Z30-HOL-DOC-NUMBER
Z30-TEMP-LOCATION
Z30-ENUMERATION-A
Z30-ENUMERATION-B
Z30-ENUMERATION-C
Z30-ENUMERATION-D
Z30-ENUMERATION-E
Z30-ENUMERATION-F
Z30-ENUMERATION-G
Z30-ENUMERATION-H
Z30-CHRONOLOGICAL-I
Z30-CHRONOLOGICAL-J
Z30-CHRONOLOGICAL-K
Z30-CHRONOLOGICAL-L
Z30-85X-TYPE

For Z68 ORDER creation:

Z68-ORDER-DATE
Z68-E-TERM-PERCENT
Z68-ORDER-TYPE
Z68-ORDER-STATUS
Z68-ARRIVAL-STATUS
Z68-VENDOR-CODE
Z68-RUSH
Z68-DELIVERY-TYPE
Z68-ORDER-DELIVERY-TYPE
Z68-TARGET-FLAG
Z68-E-CURRENCY
Z68-METHOD-OF-ACQUISITION
Z68-SEND-METHOD
Z68-MATERIAL-TYPE
Z68-AUTO-CLAIM
Z68-ORDERING-UNIT
Z68-VENDOR-NOTE
Z68-INVOICE-STATUS
Z68-ORDER-NUMBER-1
Z68-ORDER-NUMBER-2
Z68-ORDER-GROUP
Z68-LIBRARY-NOTE
Z68-VENDOR-REFERENCE-NO
Z68-QUANTITY-TEXT
Z68-E-NOTE
Z68-E-TERM-PERCENT
Z68-AUTO-CLAIM

For Z601 BUDGET TRANSACTION creation:

Z601-BUDGET-NUMBER
Z601-ORIGINAL-SUM
Z601-CURRENCY
Z601-SEQUENCE
Z601-VENDOR-CODE
Z601-INVOICE-NUMBER
Z601-LINE-NUMBER
Z601-USER-NAME
Z601-OPEN-DATE
Z601-TYPE
Z601-NOTE
Z601-CREDIT-DEBIT
Z601-CURRENCY-RATIO
Z601-PAID

tab_locate

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: Procedures for locating a record in other libraries through Web OPAC, Cataloging and ILL.

Related table(s):

1) tab_word_breaking – procedure 97
2) tab00.<lng>
3) tab01.<lng>

The *locate* function enables the user to search for similar records in other databases.

The locate query is built from the source document, using the target database lines in this table. The query includes all lines for which there is data in the source document, with a Boolean "and" between them.

Multiple lines can be set up for one library, in which case ALL lines will be taken with an AND condition between them. The tab_locate table should include both the source and the target library.

Extract Functions:
locate_str_0 - Takes subfield content as is.
locate_str_1 - Runs "build_filing_key" on subfield and takes 2 longest words. A word must be at least 2 characters in order to be considered to be a "word". If the subfield has only one word, the one word will be taken.
locate_str_2 - Takes year from 008## Position 8 Length 4
locate_str_3 - Works similarly to locate_str_1, but takes as many longest words as specified in Col. 6 (Parameters).
locate_str_no_locate - Does not perform locate (implemented for ILL for "last resort" supplier)

locate_str_sys_no - Uses a doc number in a specified field to perform an exact match.

The locate query is built from the source document, using the target database lines in this table. The query includes all lines for which there is data in the source document, with a Boolean "and" between them.

The word breaking procedure "97" is used when creating the 'FIND' query, and therefore must be defined in the tab_word_breaking configuration table.

Structure of the table:

col. 1 base code of the base where you wish to locate records

col. 2 tag

Use the # sign to indicate any indicator. Note that this is always the local tag (for example, the 245## title field is the locate parameter when locating in a UNIMARC type library from a MARC 21 type library)

col. 3 Subfield used as a "locate" parameter. Syntax:

- ab - means the content of $$a and $$b.
- -ab - means the content of all subfields except $$a and $$b.

col. 4 Find command (WRD code) that is used in the target base for searching similar records

col. 5 extract function that defines in which way are the contents of the field going to be treated

col. 6 Parameters to be sent to Extract function (Col. 5) Current programs disregard the parameters.

Example of table:

```
!         1            2            3               4                   5
                        6
!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!!!!!…
!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
USM01                245## b                    wti=       locate_str_0
!USM01                100## a                    wau=       locate_str_0
!USM01                008##                      wyr=       locate_str_2
USM10                650## a                    wsu=       locate_str_0
USM11                650## a                    wsu=       locate_str_0
USM12                650## a                    wsu=       locate_str_0
!UNI01                100## a                    wau=       locate_str_0
UNI01                245## a                    wti=       locate_str_1
!UNI01                008##                      wyr=       locate_str_2
LOC                  245## a                    wti=       locate_str_1
```

tab_location_name.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Location names for Z39.50 server

Related table(s):

2) tab_sub_library.<lng>

The tab_location_name table defines name and codes for the **ALEPH** global library (for example, USM50).

This definition is required for the Z3950 Holdings Schema, which includes location information. Location information can include institution name, institution code and isilcode (ISL code). The location can be made up of Institution and Sub-Institution Name and codes, which parallels the global library (for example, USM50) and sublibraries in **ALEPH**.

The sublibrary codes and names are defined in tab_sub_library.<lng>. This table is used to define the global library codes and names.

Structure of the table:

Col. 1 Code of the physical library, for example, USM50

Col. 2 Standard location code. This is the same as 852 $$a location, which identifies the institution or person holding the item or from which access is given. Standard codes are entered in this column, such as codes in the MARC Code List for Organizations or codes in Symbols and Interlibrary Loan Policies in Canada.

Col. 3 ISL code - isil - ISO 15511 code of the location

Col. 4 Location name - Name of the institution represented by the global library code

Example of the table:

```
! 1          2                       3                        4
!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!…!!!!!!>
USM50 HUJI                                      Hebrew University
XYZ50 DNLM                                      National Libary of Medic
```

tab_map_privileges

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defining the types of global ALEPH Z305 record

Related table(s):

1) tab31
2) tab100 - USER-SHARING set to Y
3) tab_sub_library.<lng>

This table defines the type of global ALEPH Z305 record that will be created when the patron locally registers in a sublibrary.

It will be functional only in cases where column 22 of tab31 is configured to Y, meaning that an ALEPH record is to be created.

Structure of the table:

Col.1 Sublibrary

Col. 2 Patron local status. This is the status with which the patron is registered in the sublibrary

Col. 3 Patron consortia status. This is the status with which the the ALEPH Z305 record will be created

Example of the table:

```
!!!!!-!!-!!
WID   01 08
WID   02 07
HIL   ## 04
##### 01 05
USM50 01 09
```

tab_mapping

Location of the table: tab directory of the **BIB** and **ADM** libraries

Purpose of the table Mapping values when creating records using the loaders services

tab_mapping is used by the followings load services:

- Advanced Generic Vendor Records Loader (file-90)
- Load OCLC Records (file-93)
- Load MARCIVE Records (file-99)
- OCLC server

The "Text Command" in column 6 enables the manipulation of mapped text. The following commands may be used in this column:

s – Swap text. Used to change a known input text with the required mapping. For example, the command s/BNGG/MAIN/ tells the system to substitute the text MAIN for the text BNGG in the mapped field.

S – Works in the same way as "s" (substitutes the input string by another), but returns an empty string if the input string is not found. When using this command, col 7 of tab_mapping must be set to “N”.

a – Append text. Used to append a required string to the input that is being mapped. For example, the command a/foo_/_bar tells the system to add "foo_" as a prefix to the mapped field and "_bar" as a suffix to the mapped field.

f – Find text. Used to condition the mapping action with the existence of a required text. For example, the command f/BNGG enables the mapping to take place only if the text BNGG is found in the mapped field.

F – Like "f", but works the other way round, i.e. conditions the mapping action with the required text NOT being found in the mapped field. For example, the command F/BNGG enables the mapping to take place only if the text BNGG is NOT found in the mapped field.

e – Like "f", but looks for the holding code in the mapped text.

Structure of the table:

Col.1 Holding Code
col. 2 Source Code
col. 3 Source Subfields
col. 4 Target Code
col. 5 Target Subfields
col. 6 Text Command
col. 7 Overlay Flag
col. 8 New Line Flag

- Y - for all occurrences
- N - for first occurrence of source tags

Example of the table:

```
!    1            2     3     4     5                  6                 7 8
!!!!!!!!!!!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!
UMUSI           950## l     852   b                                      Y N
UMUSI           950## l     852   c     s/UMUSI/GEN/                     Y N
UMUSI           090## a     8520  h     s/\//g                           Y N
UMUSI           090## b     8520  i                                      Y N
MAIN            050## a     8528  h                                      Y N
MAIN            050## b     8528  i                                      Y N

XZLA            050## ab    8520  hi                                     Y N
```

tab_mapping_mng500

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definition of mapping values when creating records with p-manage-500

Related table(s):

1) tab01.<lng> in the HOL library

Structure of the table:

Col. 1 Holding Code

Col. 2 HOL/Z30 Field name. Z30 always refers to location data and HOL refers to tags in the HOL record

Col. 3 Source tag/arguments, the arguments can vary

Col. 4 Second Source

Example of the table:

```
!          1                  2                        3                      4
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!…
!!!
XXXX                  Z30-SUBLIBRARY      ULINC
                      Z30-COLLECTION       GEN
                      Z30-CALL-NO-TYPE     0
                      HOL-852              050##,ab,hi          0900,ab,h
                      HOL-852              "$$aExL"
                      HOL-852              053,a,t
                      HOL-852              "$$k1234"
                      HOL-007              007
                      HOL-888              533
```

tab_match

Location of the table: tab directory of the library

Purpose of the table: Match specifications for importing files

Related table(s):

1) tab11_ind – col. 1 or col. 5
2) tab_match_acc
3) tab_match_script
4) tab_weights

Note that this table is used by the following services
p-manage-36 – Check input file against database
p-file-99 - Load MARCIVE Records
OCLC Loader

Structure of the table:

col. 1 Match routine code. Each routine performs a particular match.

Note the reserved match code:

- CAT: This match code is used to specify the matching routines performed by the check_doc_match program.

col. 2 Match program:

- **match_doc_uid**: Match is based on a direct index (Z11). The

parameters column (column 3) should contain either the index name (column 5) or the tag code (column 1) in tab11_ind.

For example, if tab11_ind has defined for the ISBN direct index:

020 ISBN az

the parameters for a match based on the ISBN can be defined as follows:

XXX match_doc_uid I-ISBN

or

XXX match_doc_uid T-020

Where I=<index code> and T=<tag code>. When using T-<tag code> there must be an exact match. If tab11_ind col.1 has 020##, this table must have T-020## as well.

- **match_doc_uid_2.** Match is based on a direct index (Z11). The parameters column (column 3) should contain the index name and the tag code as a single value. This will only work if they are the same (example tag 035 and index 035) in tab11_ind. For example, if tab11_ind is defined as follows for the 035 direct index:

035 035

the parameters for a match based on the 035 can be defined as follows:

XXX match_doc_uid_2 035

- **match_doc_acc**: Match is based on a headings (ACC) index. The argument defined in column 3 is a table name. This table lists the tags in the record that should be checked against the headings index.

col. 3 Program arguments

For **match_doc_uid** this column contains the index code or the tag code used for the direct match.

For **tab_match_acc** this column contains the table name of the table that contains the tags that should be checked against the headings index. Note that this table must be located in the library's tab directory.

Example of the table:

```
F99   match_doc_gen                TYPE=IND,TAG=909,SUBFIELD=a,CODE=909
YBP   match_doc_uid                I-ISBN
STIDN match_doc_uid                I-STIDN
OCLC  match_doc_uid                T-010
```

tab_match_acc

Location of the table: tab directory of the library

Purpose of the table: Fields to be checked against the headings index

Related table(s):

1) tab_match
2) tab01.<lng>

The tab_match_acc table is a sample table used to define the fields in the records that should be checked against the headings index when the Check Input File Against Database (p-manage-36) service is used or when the check_doc_match routine is performed.

The table name should be passed as a parameter for the match_doc_acc program in the tab_match table of the library's tab directory.

Note that this table is also used by the following services:
Generic Vendor Records Loader (p-file-96)
Load MARCIVE Records (p-file-99)
OCLC Loader

Structure of the table:

col. 1 Field Code of the fields that should be checked against the headings index.

Example of the table:

```
! 1
!!!!!
245##
240##
```

tab_match_script

Location of the table: tab directory of the **BIB** library

Purpose of the table: Match specifications for checking input file against the database

Related table(s):

1) tab_match

This table is used in order to find "candidate matching records" for record loading, or for union view setup of a catalog that holds multiple records for the same work.

This table's functionality can be carried out by a table of any name. The name is not set in the programs, it is set in the third column of the tab_match table when match_doc_script is set as the program in column 2.

There are two match programs available:
match_doc_gen
match_doc_filter_hvd

For the **match_doc_gen match** program there are 3 sections to the program arguments: TYPE, TAG + SUBFIELD, and CODE. The ACC type can have an additional TRUNCATION argument.

TYPE defines the search method for finding a match:
TYPE = SYS: searches against DB system number, which is expressed as CODE=001
TYPE = IND: searches against IND Z11 index
TYPE = ACC: searches against the filing text field of the ACC Z01 headings index
TAG + SUBFIELD relates to incoming record only. The tag content is normalized using the same filing routine that is used for IND or ACC code.
CODE index name defines the code of the index that is searched in order to find database record.
TRUNCATION=Y can be added to the ACC match type. If this argument is present, the match will be performed using a truncated search, that is, the incoming record's field will be considered a match if it is contained within the heading.

The **match_doc_filter_hvd** uses the program arguments:
SE-TABLE-NAME= and MO-TABLE-NAME=.
The match procedure will use the additional table(s) registered here for more specific matching arguments. The program automatically rejects all matches if the incoming record format is not SE or BK, and automatically rejects matches if there is a mismatch on the FMT field.

For each argument in this table you can define what will happen next, depending on the number of candidate records found in database.
In column 3 you set the parameter for the number of candidate records (for example, 20+ means more than 20, 20- means 20 or less, 0+ means more than none, 0 means none.
In column 4 you register the action:
skip (to skip to the next match set)
stop (to stop script execution)
goto <xx> (to jump forwards/backwards to a different match set <xx>
<any text> or blank acts in the same manner as skip.

Structure of the table:

Col. 1	Match set identifier
Col 2	Match program name.
Col. 3	Number of candidate records found in database. This refers to the number of records in the database that match the incoming record.
Col. 4	The action to be taken where the condition of number of candidate records is true.
Col. 5	Match Program arguments

Example of the table:

```
!1            2                        3      4                 5
!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!>
01 match_doc_gen                    1    stop       TYPE=SYS,TAG=035,CODE=001
01                                  0+   goto 02

02 match_doc_gen                    1    stop       TYPE=IND,TAG=035,CODE=035
02                                  0+   goto 03

03 match_doc_gen                    1    stop       TYPE=IND,TAG=024,SUBFIELD=a,C
ODE=024
03                                  0+   goto 04 …
```

tab_md_ns_info

Location of the table: tab directory of the **BIB** library

Purpose of the table: Name spacing information for Aleph publishing platform

Related table(s):

1) tab_publish

tab_md_ns_info enables usage of additional or modified name space for Aleph publishing platform. This optional table can be used to define namespace information for provided formats.

Structure of the table:

Col. 1 The publishing set of records to be extracted (should match the set defined in Col.1 of tab_publish).

Col. 2 The namespace information for the publishing set

Example of the table:

```
!        1                      2
!!!!!!!!!!!!!!!!!!!--!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!->
TEST1                 xmlns="http://www.loc.gov/MARCxx/slim"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.loc.gov/MARC21/slim
http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd"
```

tab_merge

Location of the table: tab directory of the library

Purpose of the table: Merge routines

Related table(s):

1) tab_fix – cols. 1, 2
2) fix_doc.<lng> - col. 3
3) tab_merge_overlay
4) tab_merge_adv_overlay
5) tab_preferred

The fix_doc_merge program is used to merge or overlay cataloging records according to the merging program defined in this table.

The connection from table to table in the fix/match/merge procedures is:

1) **Column 1 of the tab_fix table** identifies a section of tab_fix. The section is used to match the "fix_doc" parameter given in various batch services. **Column 2 of tab_fix** identifies the fix program to use. If the fix program is fix_doc_merge, tab_merge is used, in which case **column 3 in tab_fix** identifies the section of tab_merge to use.
2) **Column 1 of tab_merge** should match **column 3 of fix_doc**. Note that the following are reserved merge routines:

SID
UE11-1
UE11-2
OCLC

Column 2 of tab_merge defines the merge program to use:
merge_doc_overlay - Overlays records using the tab_merge_overlay table
merge_doc_adv_overlay - Overlays records using the tab_merge_adv_overlay table
merge_doc_replace - Replaces contents of first record with the second, but includes all CAT fields.

Note that when running file-99, file-93, manage-18 and oclc_server with merge_doc_overlay or tab_merge_adv_overlay, the preferred record is the record in the incoming load file.

In order to change direction add preferred_doc_switch to the tab_preferred table. It changes the direction and switches between the preferred and the non-preferred records.

Structure of the table:

col. 1 Routine name
col. 2 Program name or action
col. 3 Program arguments. Can contain the section identifier of tab_merge_overlay

Example of the table:

```
!   1                   2                                   3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
OVERLAY-01 merge_doc_overlay             01
OVERLAY-02 merge_doc_overlay             02
OVERLAY-03 merge_doc_overlay             03
…
HVD        merge_doc_adv_overlay         01
MRCV       merge_doc_overlay             02
```

tab_merge_adv_overlay

Location of the table: tab directory of the library

Purpose of the table: Advanced routines for the merging of document records

Related table(s):

1) tab_merge – col. 2, 3
2) tab_fix – col. 3
3) tab_preferred
4) tab11_acc
5) tab11_ind
6) tab01.<lng>

The tab_merge_adv_overlay table is used to define which fields are retained from which record when merging of two records occurs. The table and its section are called from the tab_merge table, when merge_doc_adv_overlay is the program defined in **column 2 in tab_merge**.

The table functions for the same purpose and in a similar manner to tab_merge_overlay, with added sophistication. The additional functionality is based on knowing which record is "preferred" when the merge is performed. When merge_doc_adv_overlay is chosen from tab_merge, the system first consults tab_preferred to set the "preferred" program that will be used (for example, preferred_doc_cdl) and the accompanying "weights" table (for example, union_preferred) union_preferred that is used to evaluate the two records. This evaluation decides which of the two records is "preferred".

The fact that one record is preferred over another record has no direct effect on the merge/overlay. The effect it has depends on the setup of this table. If the system cannot evaluate which record is preferred, the database record is set as the preferred record, by default.

Structure of the table:

Col. 1 Merge set: Two digits that define the lines that make up a single merge set. This merge set parameter is used in two tables:

- col.3 of tab_merge
- col.3 of tab_fix, when the fix_doc_merge is defined as the fix program to use

Col. 2 Merging direction:

- 1: refers to the database record
- 2: refers to the record in the incoming load file

Col. 3 Merging direction:

- Y: defines lines for the preferred record.
- N: defines lines for the non-preferred record

This is used in conjunction with use of the tab_preferred table, which is used as the starting point for setting which of the two records (database or incoming) is the preferred record.

Col. 4 Action:

Final form of document will include fields from the database record and the incoming record. Which fields are chosen and which are dropped depends on the following action values:

- **Y**

a. For database record (1) - retains field
b. For incoming record (2) - copies field

- **N** - Does not retain field
- **C** - Retains field only if the field does not occur in the other record
- **U** - Retains field only if the field data does not appear in the other record. This action can be used only for fields that are indexed in tab11_acc or tab11_ind. The text is normalized before comparison, using the filing procedure defined for the field.

Col. 5 Field tag and indicators.

This column is used to define the tag to compare.

Additional parameters, for subfield and contents or external table to match on, are defined in columns 6 and 7. Note that specifying subfield-content is optional;

Col. 6 Subfield code

Col. 7 Content parameters for the match. The parameters can be set as follows

"+|-,T,<table name>" , or

"+|-,V,<value1> [,<value2>] [,<value3>] ...

+ (occurs) or - (does not occur)

T (following value is a table name)

V (following value is an actual value).

Examples:

01 1 # Y ##### 5 +,V,usci

The above line defines that all fields in the database record that have subfield 5 with value "usci" are retained

01 # # N ##### 5 +,T,list_of_values

The above line defines that all fields that have subfield 5 containing a value listed in the "list_of_values" table are removed.

Example of the table:

```
!1 2 3 4   5   6    7
!!-!-!-!-!!!!!-!-!!!!!!!!!!>
01 # Y Y #####
! if db record has field with subfield 5=usci, always retain
01 1 # Y ##### 5 +,V,USCI
! prefer vernacular pair
01 # # Y ##### 6 +,V,01,02,03,04,05
! retain 001 from db
01 1 # Y 001
! if non-preferred has 006, and preferred does not have 006, take 006
01 # N C 006##
01 # N C 010##
01 # N C 5050#
01 # N U 035##
```

tab_merge_overlay

Location of the table: tab directory of the library

Purpose of the table: Fields to retain when overlaying (merging) catalog records

Related table(s):

1) tab_merge – col. 3
2) tab_fix – col. 3
3) tab_preferred
4) tab01.<lng>

The table defines the fields to be retained without change, when overlaying cataloging records, (copy record/paste record).

Example of the table:

```
01 1 Y #####
01 1 N OWN##
01 1 C 245##

01 2 Y 245##
```

In this example:

- all fields are taken from the first document except the OWN field.
- 245 is always taken from the copied record. If the copied record does not have a 245, the 245 of the original record is retained. Otherwise it is overlaid from the second to the first document

Note that the search for the code is sequential. For example:

```
01 1 N OWN##
01 1 Y #####
```

At first, the system will not take the OWN field (because of the N in col. 3 for the OWN field). Then, the system "sees" the next line, which says to take all fields. The result will be that the OWN field will be taken, too.

Structure of the table:

col. 1 Merge set: two digits that define the merging set. **Column 3 of the tab_fix** table is used to define the merging routine that is performed by the fix_doc_merge program. Merging routines are defined in the **tab_merge** table.

For example, if **column 3 of the tab_fix** table has OVERLAY-01 attached to the fix_doc_merge program, then the records are overlayed or replaced according to the definitions of the tab_merge table attached to, in this case, OVERLAY-01. **Column 3 of the tab_merge** table contains the merging set that is performed when the routine - OVERLAY-01 - is selected and it matches the values of the merging set defined in this column (for example, 01)

col. 2 Merge direction

- **1** - refers to the database record
- **2** - refers to the record in the incoming load file

- When setting a "preferred" record before merge is performed (for example, p-manage-38 with tab_preferred),

- **1** - refers to the preferred record
- **2** - refers to the non-preferred record

If there is no tab_preferred, or if the records are of equal weight, the database record is taken as the preferred record (1).

Using tab_merge_adv_overlay allows setting both database/incoming and preferred/non-preferred.

- When performing COPY/PASTE RECORD in the Cataloging GUI interface,

- **1** - refers to the record into which record is copied
- **2** - refers to the record from which record is copied

col. 3 Action: Y/N/C

Final form of document will include fields from the original record, together with fields from the copied record, depending on:

- **Y** - For original record (1) - retains field
- For copied record (2) - copies field
- **N** - Does not retain field
- **C** - Retains field only if it does not appear in the other document

col. 4 Tag code and indicators. This column can also be used to define subfield and contents to match on. Example:

01 2 Y 590##,5,*abc*

In this case tag 590 is disregarded if subfield $5 of the field does not contain the string "abc" as part of its contents. Note that specifying subfield-content is optional.

Example of the table:

```
!1 2 3          4
!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!
01 1 N #####
01 1 Y LDR
01 1 Y 001
01 1 Y 09###
01 1 Y 5####
01 1 Y 79###
01 1 Y 9####
01 1 Y OWN
```

tab_mime_type

Location of the table: alephe/tab directory

Purpose of the table: Extension definitions for use in save-function in the OPAC.

This table controls which file extension will be used if an OPAC user locally saves records in a certain document format. For example you may want that all documents in format 777 will get extension .end so that it later can be used in EndNote® or other citation management products. If a format is not defined in this table .sav will be used as an extension by default.
Sending the correct file extension to the client is important so it knows how to handle the content of the file.
Defined file extensions that are used in this table should also be defined in file mime.types (in $httpd_root/conf) so that the extension is known to the Apache Server.

Structure of the table:

Col. 1 Format number

Col. 2 Extension

Example of the table:

```
!1   2
!!!-!!!
777 end
```

tab_month

Location of the table: alephe/tab directory

Purpose of the table: Month codes and abbreviations

This table allows for the definition of the names of the months for use in the system in any environment where the *name* rather than the number of the month will be displayed.

Structure of the table:

col. 1 Language
col. 2 Month numeric code
col. 3 Month abbreviation
col. 4 Month name

Example of the table:

```
!1  2   3          4
!!!-!!-!!!-!!!!!!!!!!!!!!!!!!!
ENG 01 Jan January
ENG 02 Feb February
ENG 03 Mar March
ENG 04 Apr April
ENG 05 May May
ENG 06 Jun June
```

tab_move_record

Location of the table: tab directory of the **BIB** library

Purpose of the table: List of programs for moving records

This table is used to define the moving routines that are performed when records are moved through the Overview Tree in the Cataloging module.

Following are the available moving programs:
move_adm_to_adm: moves all instances under an ADM record to another ADM record. This move includes:
Items + history (Z30)
Holding requests + history (Z37)
Loans + history (Z36)
Photocopy requests (Z38)
Advance booking - time slots (Z320)
Short loans - status (Z321)
Serials claim (Z20)
Maintenance Records (Z328)
Linked HOL records
Subscription information (Z16)
Routing lists (Z18)
Members of routing lists (Z14)
Routing trace (Z22)
Orders (Z68)
Order and subscription log (Z71)
Arrival information (Z78)
Acquisition claims (Z501)

Transactions - Budget (Z601)
Invoice - Line item (Z75)

The move is not performed if one of the items is linked to an incoming or to an outgoing ILL request.

move_z30_to_adm: moves a selected item to another ADM record. This move includes:
Items + history (Z30)
Holding requests + history (Z37)
Loans + history (Z36)
Photocopy requests (Z38)
Advance booking - time slots (Z320)
Short loans - status (Z321)
Serials claim (Z20)
Maintenance Records (Z328)
The move is not performed if:
The item is linked to an incoming or to an outgoing ILL request.
The item is linked to an HOL record
The item is linked to a subscription record
The item is linked to an order record

move_z16_to_adm: moves a selected subscription to another ADM record. This move includes:
Subscription information (Z16)
Items + history (Z30)
Routing lists (Z18)
Members of routing lists (Z14)
Routing trace (Z22)
The move is not performed if:
The item is linked to an incoming or to an outgoing ILL request.
The item is linked to an HOL record
The item is linked to an order record

move_z68_to_adm: moves a selected order to another ADM record. This move includes:
Orders (Z68)
Items + history (Z30)
Order and subscription log (Z71)
Arrival information (Z78)
Acquisition claims (Z501)
Transactions - Budget (Z601)
Invoice - Line item (Z75)
The move is not performed if:
The item is linked to an incoming or to an outgoing ILL request.
The item is linked to an HOL record
The item is linked to a subscription record

move_items_to_adm: moves all items under the selected items node to another ADM record. The move includes all instances specified under move_z30_to_adm. This program should be used with ITEMS in column 1.

move_copies_to_adm: moves all subscriptions under the selected subscriptions node to another ADM record. The move includes all instances specified under move_z16_to_adm. This program should be used with COPIES in column 1.

mover_orders_to_adm: moves all orders under the selected orders node to another ADM record. The move includes all instances specified under move_z68_to_adm. This program should be used with ORDERS in column 1.

move_mex_to_hol: moves a selected hol item to another HOL record. This program should be used with MEX in column 1. This move includes:
Items (MEX)

move_hol_items_to_hol: moves all hol items under the selected HOL record to another HOL record. The move includes all instances specified under move_mex_to_adm. This program should be used with HOL_ITEMS in column 1.

move_hol_to_bib: moves HOL record from one BIB to another BIB. The move is not performed if an item is linked to the HOL.

move_adm_to_bib: moves ADM record from one BIB to another BIB. The move includes linked HOL records. It is not performed if an ADM record linked to the target BIB already exists.

move_bib_to_bib: moves one BIB to another BIB. The move includes linked HOL records (together with their linked items) and ADM records. This function can also be used for AUT libraries.

move_hol_items_to_bib: moves a HOL record (together with its items, if any) to another BIB record. Moving of items is performed according to the same guidelines as in move_z30_to_adm; naturally, in this case the restriction preventing the move of an item connected to a HOL record is irrelevant.

move_z30_to_hol: moves an item connected to a HOL record, to another HOL record connected to a different BIB record. Moving of the item is performed according to the same guidelines as in move_z30_to_adm; as in move_hol_items_to_bib, the HOL link restriction is not activated.

Note that whenever an item record (Z30) is moved from one ADM record to another, the item Maintenance Records attached to it (Z328) are also moved.

In addition, if a Maintenance Profile record (Z327) is attached to the source ADM record, and the target ADM record does NOT have a Maintenance Profile, a new Maintenance Profile record is created for the target ADM by copying all data from the source ADM's Z327 record.
If the target ADM record already has a Z327 record, it remains untouched.

Note that if the relevant program for a particular move is not listed in this table, then an error message is displayed in the GUI when trying to perform the selected move.

Structure of the table:

- Col. 1 Moving from Record Available options are: BIB, ADM, Z30, Z16, Z68, ITEMS, COPIES, COPIES, ITEMS, HOL HOL-ITEMS, MEX
- Col. 2 Moving to Record; available options are: ADM , BIB, HOL, HOL-ITEMS
- Col. 3 Moving Procedure

Example of the table:

```
!   1           2                   3
!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
ADM         ADM         move_adm_to_adm
Z30         ADM         move_z30_to_adm
Z16         ADM         move_z16_to_adm
Z68         ADM         move_z68_to_adm
ITEMS       ADM         move_items_to_adm
COPIES      ADM         move_copies_to_adm
ORDERS      ADM         move_orders_to_adm
```

Note that all the error messages related to tab_move_record are located in $aleph_root/error_lng/move_record.

tab_ncip_record_id

Location of the table: tab directory of the **ADM** library

Purpose of the table: Assignment of fields to Direct Indexes for NCIP Server

This table defines the record keys that will be used by the NCIP server for two purposes:
1. Searching for records based on bibliographic information.

If an incoming message, such as RequestItem, uses the UniqueBibliographicId element to request locating an item based on bibliographic information, the method for handling the request is specified in this table. The table specifies for each record id type (specified in the BibliographicItemIdentifierCode\value or BibliographicRecordIdentifierCode\value element of the incoming message) what bibliographic field should be searched.

2. Including bibliographic information in NCIP message responses.

When the NCIP response includes a BibliographicDescription element, be filled in with bibliographic information from the field that is assigned in tab_ncip_record_id for the access code that is set up in the bibliographic_item_id variable of tab_ncip.conf.

For example, if a LookupItem message is sent for an item, the server will look at the bibliographic_item_id variable to know what bibliographic information it should return.

Structure of the table:

Col. 1 Record id

Col. 2 Index code - code of the Direct index file;

Col. 3 Field code

Col. 4 Subfield(s). Blank indicates entire field. Use minus (-) sign followed by subfields to define subfields to be stripped or list the subfields to be indexed

Example of the table:

```
!    1        2     3             4
!!!!!!!!!!-!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!
OCLC        035   035## a
ISSN        022   022## a
ISBN        020   020## a
```

tab_oclc

Location of the table: alephe/tab directory

Purpose of the table: OCLC server configuration table

Related table(s):

1) tab_merge
tab_match
2) tab_loader – col. 2

This table allows for configuring defaults for loading data from the OCLC server.

Structure of the table:

col. 1 OCLC server port number

col. 2 Record type, AUThority or BIBliographic

col. 3 Destination library for incoming records

col. 4 Fix procedure - in addition to the OCLC procedure.

col. 5 Document insertion fix procedure. This column can include up to 5 special fixes. Currently there are 2 fixes:

- Creates OWN field, populated by the value set in col.10.
- 2. Modify/create 001 field with data from column 10 and the record system number in the following way:

c. <Col.10 value>-<system number>

d. If column 10 is empty, the above fixes will use column 2 of tab_loader. If the tab_loader col.2 is empty, no OWN field will be created.

col. 6 Item creation flag

col. 7 HOL creation flag

col. 8 Merge routine (specified in tab_merge)

col. 9 Match section (as specified in tab_match)

use only the first 5 positions of this column - because tab_match section is only 5 characters

col. 10 OWN field value.

The value in this column is used as a filter for record match. This might be needed for setup in which libraries share a common bibliographic database, but catalog separate records.

Example of the table:

```
! 1   2    3     4     5   6 7     8          9              10
!!!!!-!!!-!!!!!-!!!!!-!!!!!-!-!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!
5777  BIB USM01       12    Y Y OCLC       OCLC
5777  AUT USM10             N N
```

tab_own

Location of the table: tab directory of the library

Purpose of the table: Cataloging OWN permissions and filter

Related table(s):

2) tab100

tab_fix_notes

Location of the table: tab directory of the BIB library

Purpose of the table: Defines texts used for translation in the fix routine fix_doc_notes

Related table(s):

2) tab_fix (fix_doc_notes routine)

tab_fix_notes defines text to be translated with the fix routine fix_doc_notes. Text may be defined for different tags and subfields. For more information, see the **Automatic Translation of Bibliographic Note Fields** section of the *Aleph 21 System Librarian's Guide-Cataloging*.

Col 1. BIB Tag & indicator

Col. 2 Subfield

Col. 3 BIB Source text

Col. 4 Translation text

The following is a partial example of tab_fix_notes (the actual lengths of columns 3 and 4 length are longer than that what they appear in the following table structure example):

```
! 1   2                    3                          4
!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>-!!!!!!!!!!!!!!!!!!!!!!!->
245## h [electronic resource]              [ressource électronique]

310## a Annual                             Annuel
310## a Biannual                           Semestriel
310## a Bi-annual                          Semestriel
310## a Biennial                           1 no par 2 ans
310## a Bimonthly                          Bimestrie

500## a Adaptation of:                     Adaptation de :
500## a Added t.p. title                   Titre de la p. de t.
addit. :
500## a At head of title                   En tête du titre
500## a Caption title                      Titre de départ
500## a Cataloguer's title                Titre du catalogueur
500## a Colophon title                     Titre de l'achevé
d'imprimer
500## a Cover title                       Titre de la couv.
```

3) tab_fix_z103

This table assigns the group of OWN values of a cataloging record (BIB, AUT, ADM or HOL) that are allowed for a particular OWN authorization. Depending on the relevant tab100 flag an additional display filter based on the permitted OWN values is activated (only HOL or BIB).

If a record has an OWN field with a value of, for example, 'AA', only users with OWN permission of 'AA' can modify or delete the record.
A user must also have an OWN permission of 'AA' to add or remove an OWN field with the value 'AA'.

Using this table, the OWN permission can be more general, and can be authorized for up to 5 different OWN values.

Note that it is possible to assign more than 5 different OWN values of cataloging records to a user's OWN value by using the hash (#) character as a wildcard. Following is a sample of the table in which the # sign is used to cover more OWN values:

```
!     1          2          3          4          5          6
! !!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!
! CAT         ME###
! CAT1        #####
! CAT2        ##########
```

In the above sample:

ME### includes, for example, MEDUC, MELEC, and so on..
includes all OWN values that are up to five characters.
########## includes all possible OWN values (this is equal to the GLOBAL authorization).

Note that if a User's OWN value needs to be assigned more than 5 different record's own values, (without using the hash (#) character as a wildcard) it is possible to define multiple lines for the same User's OWN. For example:

```
!     1          2          3          4          5          6
! !!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!
! CAT       MED        HYL        HIL        LAM        LAW
! CAT       LIT        MUS        WID        HILR       BCU
```

The filter - if activated - affects the following displays:
ITM (HOL) and ITML in the full + link display
navigation tree (HOL)
tab "HOL records" in cataloging

Structure of the table:

Col. 1 User's OWN. This column contains the value of the cataloging OWN Permission field assigned to the user(s).

Cols. 2-6

Record's OWN. Columns 2 to 6 contain the record's OWN values which the user with the OWN permission defined in column one is allowed to update.

Example of the table:

```
!    1          2          3          4          5          6
!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!-!!!!!!!!!
MOZART     HAYDN      HAYDNJ     BACH       BACHJ
HAYDN      HAYDNJ     BACH       BACHJ
HAYDNJ     HAYDN      BACH
BACH       BACHJ      HAYDNJ
BACHJ      BACH
EDUC5
YOHANAN    UARCV      UEDUC      UELEC      ULINC      ULAW
```

fix_z103_sort_by_my_own in the table tab_fix_notes

Location of the table: tab directory of the BIB library

Purpose of the table: Defines texts used for translation in the fix routine fix_doc_notes

Related table(s):

3) tab_fix (fix_doc_notes routine)

tab_fix_notes defines text to be translated with the fix routine fix_doc_notes. Text may be defined for different tags and subfields. For more information, see the **Automatic Translation of Bibliographic Note Fields** section of the *Aleph 21 System Librarian's Guide-Cataloging*.

Col 1. BIB Tag & indicator

Col. 2 Subfield

Col. 3 BIB Source text

Col. 4 Translation text

The following is a partial example of tab_fix_notes (the actual lengths of columns 3 and 4 length are longer than that what they appear in the following table structure example):

```
! 1   2                    3                          4
!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>-!!!!!!!!!!!!!!!!!!!!!!!->
245## h [electronic resource]              [ressource électronique]

310## a Annual                             Annuel
310## a Biannual                           Semestriel
310## a Bi-annual                          Semestriel
310## a Biennial                           1 no par 2 ans
310## a Bimonthly                          Bimestrie

500## a Adaptation of:                     Adaptation de :
500## a Added t.p. title                   Titre de la p. de t.
addit. :
500## a At head of title                   En tête du titre
500## a Caption title                      Titre de départ
500## a Cataloguer's title                Titre du catalogueur
500## a Colophon title                     Titre de l'achevé
d'imprimer
500## a Cover title                        Titre de la couv.
```

tab_fix_z103. This program sorts records according to the value of tag OWN. All records with OWN tags assigned to the user according to tab_own are positioned at the beginning of the record list.

tab_patron_log.<lng >

Location of the table: tab directory of the **USR** library

Purpose of the table: List the logs for patron record changes

tab_patron_log.<lng> is used for setting-up the action codes (column 1) that trigger Z307 Patron Log records and their textual descriptions (column 3). This table is located in the tab directory of $usr_library. Only the activated actions (column 2 = Y) are triggered. Use column 2 to have Aleph ignore action triggers.

The following is a list of possible logs (numeric code–col.1 and text decription-col.3):

001 - Update of global patron record-Details
002 - Update of global patron record-Blocks
003 - Update of local patron record-Details
004 - Update of local patron record-Blocks
005 - Update of local patron record-Privileges
006 - Update of patron addresses
007 - Update of patron IDs
010 - Delete of global patron record
011 - Delete of local patron record
012 - Delete of patron address record
013 - Delete of patron ID
020 - New global patron record created
021 - New local patron record created
022 - New patron address record created
023 - New patron ID created

Structure of the table:
COL 1. The action type code that cause a Z307 patron change log record to be written
COL 2. Active/Non active log: Y/N
COL 3. Description of the action

Example of the table:

```
!1  2           3
!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!->
001 Y Update of global patron record-Details
002 Y Update of global patron record-Blocks
003 Y Update of local patron record-Details
004 Y Update of local patron record-Blocks
005 Y Update of local patron record-Privileges
006 Y Update of patron addresses
007 Y Update of patron IDs

010 Y Delete of global patron record
```

```
011 Y Delete of local patron record
012 Y Delete of patron address record
013 Y Delete of patron ID

020 Y New global patron record created
021 Y New local patron record created
022 Y New patron address record created
023 Y New patron ID created
```

tab_photo_request

Location of the table: tab directory of the **ADM** library

Purpose of the table: checks for photocopy requests

Related table(s):

1) tab15.<lng>
    a. col. 9=Y (photocopy request is available) (check type a)
2) tab38 – sublibrary and item status are listed (item can be requested) (check type b)

This table is used to define which checks should be performed by the system when a photocopy request is placed. It is similar in functionality to tab_hold_request. In some cases, the checks on the photo requests for the Web OPAC determine whether the ***photo*** link appears on the item line.

Structure of the table:

col. 1 defines when check is performed

col. 2 defines check procedure to perform; most procedures are relevant for POST

col. 3 defines parameters to the check routine

Types of checks:
PRE - Web OPAC, before showing request link, usually does not depend on patron identification
POST- Web OPAC, after filling request form
CIRC- GUI client (patron identification is always present.
UE06-AVAIL - UE_06, before printing photocopy request slip

The following checks can be performed:

Type of check	Check Procedure	Explanation	Heading
PRE	a	tab15.<lng> – column 9=Y (photocopy request is available)	1200
PRE	b	tab38 – sublibrary and item status are listed (item can be requested)	1205
POST	c	z305 - photo-permission Y/N	1210
POST	r	z305 - Patron expiry date check	1165
INFO	k	tab15 col.8 value (Y,N,C)	6010-6015
INFO	k1	Z30-description value	6016

CIRC and POST	sh	check the request submission time (day+hour) against the pickup location service hours, set in tab_service_hour	1235
PRE	d	Item availability Retrieves list of item statuses that are not checked. For example: PRE check_photo_request_d 01,02	1250
PRE	e	Item statuses and borrower statuses Retrieves list of item statuses and borrower statuses that are checked. For example: PRE check_photo_request_e ITEM=##;BOR=02,01	1260

Example of the table:

```
! 1                      2                                    3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!

PRE         check_photo_request_a
PRE         check_photo_request_b
PRE         check_photo_request_d          01,02
PRE         check_photo_request_e          ITEM=##;BOR=02,01
!*
PRE-GUI     check_photo_request_a
PRE-GUI     check_photo_request_b
!*
POST        check_photo_request_c
POST        check_photo_request_f
POST        check_photo_request_r
```

tab_pinyin

Location of table: tab directory of the **BIB** library

Purpose of table: Definition of fields for fix_doc_add_pinyin_check_sub9 and fix_doc_add_pinyin_insert_sub9

Related table(s):

1) tab01.<lng>

This table is consulted to determine the fields for which fix_doc_add_pinyin_check_sub9 and fix_doc_add_pinyin_insert_sub9 should be performed.
The fix_doc_add_pinyin programs act on the fields defined, if the content is CJK. The programs take the content of $$a and create a parallel $$9 subfield in pinyin, using chi_segmentation (Z113) and pinyin translation (Z114).

The _check program can only be used in the cataloging module, with cataloger intervention. In this program, in cases where a character has more than one pinyin option, the created subfield contains <option1,option2,...>. The cataloger can decide which to use, deleting the others.

The _insert program chooses <option1> if there is more than option for pinyin.

This table is limited to 200 lines.

Structure of the table:

Col. 1 Document record field code. # can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 245## for 2451, 2452, 24501)

Col. 2 Not used; Subfield code

For _sub9 only subfield $$a is relevant, and it does not have to be entered.

Example of the table:

```
!1    2
!!!!! !!!!!!!!!!!!!!!!!!!!
100##
245##
260##
440##
600##
700##
```

tab_preferred

Location of the table: tab directory of the library

Purpose of the table: Defines settings for preferred records when merging document records

Related table(s):

1) tab_merge
2) tab_merge_overlay
3) tab_merge_adv_overlay

This table is used with the tab_merge table when merging records.

The tab_preferred table is used to set the program and the weighting table to use in order to set which of two matching documents is the preferred document.

preferred_doc_switch changes the direction of merge_doc_overlay (in the tab_merge table) and tab_merge_adv_overlay: It switches between the preferred and the non-preferred records.

This program should be used when running file-99, file-93, manage-18 and oclc_server, if you want to change direction.

The first column is used for listing the identifier that is used to match the merge type parameter input in the p_manage_38 (Check Input File Against Database) service.

This parameter is also used in the first column of tab_merge.

Note: The name in col.1 should be the same as the routine name in col.1 of the table tab_merge.

Structure of the table:

Col. 1 Section name identifier. This identifier is referred to from the p_merge_type parameter in p-manage-38.

Col. 2 Program name. The programs currently available are:

preferred_doc_cdl.

preferred_doc_switch

preferred_doc_suppl_prio

Col. 3 Program arguments. Currently the name of the weighting table to be used.

Example of the table:

```
!1                    2                          3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!>
AD-OVERLAY preferred_doc_cdl                union_preferred
OCLC       preferred_doc_cdl                union_preferred
RLIN       preferred_doc_cdl                union_preferred
TEST       preferred_doc_cdl                union_preferred
MRCV       preferred_doc_cdl                union_preferred
F99        preferred_doc_cdl                union_preferred
```

tab_publish

Location of the table: tab directory of the **BIB** or **AUT** libraries

Purpose of the table: Specifications for extracting records

Related table(s):

1) tab_base.<lng>
2) fix_doc
3) tab_expand
4) tab_md_ns_info

This table contains the specifications for extracting ALEPH records for publishing purposes.

Structure of the table:

Col. 1 Publishing Set. This column contains the code of the set of

records to be extracted. . The Publishing Set code must be unique across all tab_publish tables of the library and cannot be repeated within tab_publish.

Note that when column 5 is in HTML format the publishing set in column 1 must be prefixed by WEB_PUB, for example: WEB_PUB1.

Col. 2 Base. A set can be the entire database or a section of the database as defined by a logical base. This column contains the code of the desired logical base from the tab_base.<lng> table. If the column is left blank, the entire database will be extracted for the set.

Col. 3 De-duplication (Currently not in use)

Col. 4 The fix (fix_doc) and expand (tab_expand) code of the routines that should be applied before the record is extracted

Col. 5 The format of the records in the repository. Supported formats: MAB_XML, MARC_XML, HTML, OAI_MARC21_XML, OAI_DC_XML

Example of the table:

```
!        1                        2           3   4          5
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!-!-!!!!!-!!!!!!!!!!!!!!!
PRIMO-FULL                                   N FULLP MARC_XML
PRIMO-AVAIL                                  N AVA   MARC_XML
TEST1                MONOGRAPHS              N FULLP MARC_XML
```

tab_rfid

Location of the table: tab directory of the **ADM** library

Purpose of the table: RFID configuration table

Related table(s):

4) tab_rfid_translate

This table defines which fields from the item record (Z30) are expanded into the transponder record (RFID).

Structure of the table:

Col. 1 Transponder field

Col. 2 Item Field Name

Col. 3 Defines if the code of the item field is to be translated to name (for example: sublibrary code to sublibrary name). This option is available for various tags. For example:

- LibraryCode tag – using the SUB-LIB section of tab_rfid_translate
- LogisticPartGroup tags – using the LOGISTIC section of tab_rfid_translate

Values are:

Y – Translate

N – Do not translate

X – Not applicable

Col. 4 Routine for character conversion – This name must match procedure identification in col.1 of /alephe/unicode/tab_character_conversion_line

Default value is UTF_TO_RFID

Col.5 For 3M usage only. It defines the field type: (I)Integer or (S)String

Example of the table:

```
!                1                                  2                    3           4
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!
Usage                             0
TypeEAS                           0
DocumentNumber                    z30-barcode
LibraryCode                       z30-sub-library                      Y
ItemNr                            z30-enumeration-a
TotalItems                        01
LogisticPartGroup1                z30-sub-library                      Y
LogisticPartGroup2                z30-collection                       Y
LogisticPartGroup3                z30-material                         Y
LogisticPartGroup4                04
LogisticPartGroup5                   00
```

tab_rfid_translate

Location of the table: tab directory of the **ADM** library

Purpose of the table: RFID translation table

Related table(s):

1) tab_rfid

The table defines the expansion for RFID fields.

Structure of the table:

Col. 1 Type of line:

MEDIA-TYPE - enumerated material type

SUB-LIB - enumerated sublibraries, used for the NEDAP LibraryCode tag

LOGISTIC - enumerated values used for the NEDAP LogisticPartGroupTransponder field tags

Col. 2 ALEPH definition

Col. 3 Description

Example of the table:

```
!    1            2       3          4
!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
SUB-LIB          01    WID        Main Library
SUB-LIB          02    MED        Medicine Library
SUB-LIB          03    HYL        East Asian Library

LOGISTIC         11    GEN        General Collection
LOGISTIC         02    ARC        Archives Collection
LOGISTIC         01    WID        Main Library
LOGISTIC         05    HYL        East Asian Library
```

tab_roster

Location of the table: tab directory of the **ADM** library **OR** the alephe/tab directory

Purpose of the table: Controls the order that the roster batch job (p-cir-79) moves requests from institute to institute.

Related table(s):

1) tab_sub_library.<lng>

This table sets the roster of potential suppliers that will be queued for the fulfillment of a title request. The table defines groups of potential suppliers per pickup location. The potential suppliers may be either ADM libraries or sublibraries within the ADM.

A potential supplier will be queued for a title request fulfillment only if it has a requestable item.

The table may be defined either centrally in alephe_tab or in the tab directory of each ADM.

Structure of the table:

Col. 1 Pickup location. This is the location to which the item will be sent

Col. 2 Group level. This is the priority of the group within the roster. The lower the number, the higher the priority that the group has.

Col. 3 Group randomiztion. This column sets whether the potential suppliers within the group should be sorted as in the table or sorted in a random order.

Cols. 4-13

Potential Supplier.The code of the potential supplier. This may be either the code of an ADM library or the code of a sublibrary.

tab_service

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: Table for defining services in the Web OPAC

Related table(s):

1) edit_doc_999.<lng>
2) tab11_acc
3) tab11_ind
4) tab00.<lng>
5) www_server.conf
6) tab100 - AUTO-TRUNCATE-Z01-FIND
7) tab_z121

This table defines the services available for the fields displayed through edit_doc_999.<lng> in the Web OPAC. The services are defined for a field and are:

scan_acc and scan_ind

activates display of a browse list. By default, all the browse lists defined for the field in tab11_acc and tab11_ind are displayed to choose from. This can be limited by entering the lists to include in the service in col. 3.

find

activates the find command. All the browse lists defined for the field in tab11_acc are displayed to choose from.

find_m_acc

activates find command on multiple databases. All the browse lists defined for the field in tab11 are displayed to choose from. The browse lists in the various databases must be defined with the same code. The find is performed on the databases defined in the "www_parallel_search_base" environment variable in the www_server.conf configuration file. In order to ensure that long titles are retrieved, the AUTO-TRUNCATE-Z01-FIND switch in tab100 should be set to "Y".

aut

activates display of the authority record to which the field heading is linked

aut_bib

used only for AUT library. This service offers the following functionality:

The patron can initiate a search in an AUT database to find a relevant record. From this record, the aut_bib service can be used in order to create a set of the relevant BIB records, that are indexed using the same text as one of the AUT fields.

This service initiates performing a FIND query from the AUT record on a SCAN (Z01) list in the BIB library. The parameters for the query (which BIB library and BIB SCAN list) are defined in the www_a_<lng>/service-aut-bib html page.

The FIND command is truncated at 50 characters and normalized using filing procedure 99. This is particularly useful for libraries which use UDC/DDC classification for subject indexing of BIB records, and an AUThority database in which each UDC/DDC is an authority record with textual descriptor.

When the FIND command is executed and the resulting display is presented to the patron, the database environment is automatically switched to the BIB database. In order to initiate a new AUT search, the patron will have to change database. Therefore, for libraries that use this service, it is recommended that a link to the AUT database be added to the HTML "banner".

engine
activates an external search engine, using details defined in the tab_z121 table.

Structure of the table:

Col. 1 Tag+Indicators of the record field

Col. 2 Service

Col. 3 Parameters for the service.

For SCAN and FIND, parameters can be the Z01 lists that can be used for the service.
For www_f_service_engine subfields that should be used for searching in the external search engine can be defined (if blank all subfields are used). It is possible to define more than one subfield or use a "-" to indicate all subfields but not....

Example of the table:

```
! 1   2                              3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1#### www_f_service_find_acc         AUT
1#### www_f_service_scan_acc         AUT
1#### www_f_service_find_m_acc       AUT
1#### www_f_service_aut              AUT
!*
7#### www_f_service_find_acc         AUT,TIT
7#### www_f_service_scan_acc         AUT,TIT
7#### www_f_service_aut              AUT
```

Example of the tab_service table as defined in usm10:

```
! 1   2                              3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1#### www_f_service_find_acc         PER,COR,MET,TOP,TIT,GEO
1#### www_f_service_scan_acc         PER,COR,MET,TOP,TIT,GEO
1#### www_f_service_aut_bib          PER,COR,MET,TOP,TIT,GEO
!
4#### www_f_service_find_acc         PER,COR,MET,TOP,TIT,GEO
4#### www_f_service_scan_acc         PER,COR,MET,TOP,TIT,GEO
4#### www_f_service_aut_bib          PER,COR,MET,TOP,TIT,GEO…
```

tab_service_hour

Location of the table: tab directory of the **ADM** library

Purpose of the table: Library service hours for hold and photocopy requests

Related table(s):

1) tab_hold_request
2) tab_photo_request

This table defines the service hours for hold and photocopy requests. The request submission time (day+hour) is checked against the pickup location service hours that are set in tab_service_hour. This table lists service hours for days of the week (00-Sun thru 06-Sat) or a specific date (YYYMMDD).
Note that the system applies tab_service_hour only when check_hold_request_sh and check_photo_request_sh are set in tab_hold_request and tab_photo_request respectively. Only days+hours defined in this table are considered service hours. Days+hours that are not set are labeled as non-service hours. If the tab_service_hour table is empty (with no lines set) or does not exist, request services are off (there are no available service hours).
If a request is submitted when the pickup location is out-of-service, the request is not accepted and a message is displayed indicating that the pickup location is currently out of service and that the request cannot be submitted. For setup and additional information, see the **Request Service Hours Checks (tab_service_hour)** section of the *System Librarian Guide-Circulation*.

Structure of the table:

COL 1. - Service type – H or P,#
H-Hold Request or P-Photocopy request or #.
COL 2. - Item Sublibrary
COL 3. - Item status;
COL 4. - Patron Status;
COL 5. - Pickup Location;
COL 6. - Date YYYYMMDD, can be also set with a specific date with 'match-all' year. E.g.####1226;
COL 7. – Day of the week - ,00,01,02,03,04,05,06
Day of Week. Leave empty if column 6 is set with a specific date;
00 = Sun, 01=Mon thru 06 = Sat.
COL 8. - Status O or C.
O = open;
C = closed, can be set for full day only. Cannot be applied for specific hours;
COL 9. Open Hour, not-relevant when column 8 is set with 'C' - Closed;
COL 10. Close Hour; not-relevant when column 8 is set with 'C' - Closed;
If the closing hour is after midnight, a +24 notation may be used (relevant only when column 7 is set with specific day). For example, 2 A.M. is configured as 2600;

The following is an example of the table:

```
!  2    3  4   5        6     7  8    9 10
!-!!!!!-!!-!!-!!!!!-!!!!!!!!-!!-!-!!!!-!!!!
# ##### ## ## ##### ####1226    C
# ##### ## ## ##### ######## 00 C
# ##### ## ## ##### ######## 06 C
H ##### ## ## LAW   ######## 01 O 0800 1700
H ##### ## ## LAW   ######## 02 O 0800 1700
H ##### ## ## LAW   ######## 03 O 0800 2600
H ##### ## ## LAW   ######## 04 O 0800 1700
H ##### ## ## LAW   ######## 05 O 0800 1700
```

```
H ##### ## ## ###   ########    O 1000 1400
P ##### ## ## WID   ######## 01 O 0800 1700
P ##### ## ## WID   ######## 02 O 0800 1700
P ##### ## ## WID   ######## 03 O 0800 2600
P ##### ## ## WID   ######## 04 O 0800 1700
P ##### ## ## WID   ######## 05 O 0800 1700
P ##### ## ## ###   ########    O 1000 1400
```

To configure the pickup location service hour check for a hold request, set the following in ./xxx50/tab/tab_hold_request:

```
  ! 1                      2
   !!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   POST       check_hold_request_sh
   CIRC       check_hold_request_sh
   XML-CIRC   check_hold_request_sh
```

Note: For hold requests, the only relevant instances for the service hour check are POST, CIRC, and XML-CIRC.

To configure the pickup location service hour check for photocopy requests, set the following in ./xxx50/tab/tab_photo _request:

```
   ! 1                      2
   !!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   POST       check_photo_request_sh
   CIRC       check_photo_request_sh
```

Note: For photocopy requests, the only relevant instances for the service hour check are POST and CIRC.

tab_sfx

Location of the table: tab directory of the **BIB** library

Purpose of the table: defines under which conditions the SFX button will appear in the Web OPAC.

Related table(s):

1) tab_base.<lng>

This table is used to set the icon that will be displayed for the SFX link in the Web OPAC. The table can be used to define what SFX services should be available for a record that was found on a specific base, and which icon will be displayed to signify that the services are available.

Structure of the table:

Col. 1 Base name as defined in tab_base.<lng>. The base on which the search was conducted

Col. 2 SFX button name. The icon that will be displayed if the record has the required SFX services available. The icon name must be a valid name

Col. 3 SFX button identifier. The index on the HTML page to which the button will be inserted.

Col. 4 Direct link - The window type that will be opened for the link

Col. 5 Service name. The service name that must be present in order for the link to be active. Note that if a few services are specified under the same identifier (column 1), at least one of them can be present in order the link to be active. Below is a list of valid service names (all case-sensitive):

- getWebService
- getWebSearch
- getTOC
- getSubject
- getStatus
- getReference
- getPayPerView
- getMessageNoFullTxt
- getHolding
- getFullTxt
- getDOI
- getDocumentDelivery
- getCitedReference
- getCitedRecord
- getCitedJournal
- getCitedGenome
- getCitedBook
- getCitedAuthor
- getBookReview
- getAuthorEmail
- getAuthor
- getAlikeRecord
- getAbstract

Example of the table:

```
      1                    2              3  4    5
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!-!-!!!!!!!!!!!!!!!!!>
USM01                www_f_sfx_cj          11 Y getCitedJournal
USM01                www_f_sfx_ft          12 N getFullTxt
USM01                www_f_sfx_ae          12 Y getAuthorEmail
```

In the above example, three different buttons are defined
www_f_sfx_cj,
www_f_sfx_ft and

www_f_sfx_ae)
for three different services:
getCitedJournal,
getFullTxt and
getAuthorEmail.
Each one will appear only when its corresponding service is available via SFX. In addition, in the HTML pages, each one will be identified differently:
"getCitedJournal" button place-holder is "$1100", getFullTxt button is "$1200" and so on.

tab_sfx_denied

Location of the table: tab directory of the **BIB** library

Purpose of the table: Controls under what conditions the sfx button will not appear

Related table(s):

1) formats.<lng>
2) tab01.<lng>

Note that this table can be defined in any BIB ***type*** of library – for example, Course Reserves.

Structure of the table:

Col. 1 Doc format
Col. 2 Tag (as defined in tab01.<lng>)
Col. 3 Tag subfield
Col. 4 Exists (Y/N)
Col. 5 Boolean operator
Col. 6 Tag (as defined in tab01.<lng>)
Col. 7 Tag subfield
Col. 8 Exists (Y/N)

Example of the table:

```
!1   2   3 4  5    6   7 8
!!-!!!!!-!-!-!!!-!!!!!-!-!
BK 1001# a Y AND 245## a Y
MU 1001# a Y AND 245## a Y
SE 022## y Y AND 022## a N
```

tab_sip2_alert

Location of the table: tab directory of the **ADM** library

Purpose of the table: Self-check field alert definitions

Related table(s):

1) tab15.<lng>tab_sub_library.<lng>

The table defines the values for the self check fields alert and alert type for items returned at a self check station. Alert and alert type are defined dependent on item process status, sublibrary of item and pickup location of hold request (if the item has a hold).

Example of a setup for a self check machine located in sublibrary ULINC:

```
CT ##### ##### Y 99
## ULINC       N 00
## #####       Y 04
## ##### ULINC Y 01
## ##### ILLDT Y 03
## ##### ##### Y 02
```

All items with process status CT will get an alert with alert type 99.
The second and third line define the handling of items without holds (col. 3 is blank): No alert for items from sublibrary ULINC, all items from other sublibraries will get an alert with alert type 04 (sent to other branch).
Lines four to six define the handling of items with holds:
If the pickup location is ULINC the item will get an alert with alert type 01 (hold for this branch),
for pickup location ILLDT the alert type is 03 (hold for ILL).
For all other pickup locations except ULINC or ILLDT the item will get an alert with alert type 02 (hold for other branch).

Note:
If you are using self check machines in several branches you have to set up this table separately for every branch. Add the port number of the SIP2 server used for the branch to the name of the table: tab_sip2_alert.<port>

Structure of the table:

Col. 1 Item process status as defined in tab15.<lng>
Col. 2 Sublibrary of the item;
Col. 3 Pickup location of a requested item
Col. 4 Alert
Col. 5 Alert type:

- 00 - undetermined
- 01 - hold for this branch
- 02 - hold for other branch
- 03 - hold for ILL
- 04 - send to other branch
- 99 - other

tab_sip2_conf

Location of the table: tab directory of the **ADM** library

Purpose of the table: Self-Check configuration

This table defines configuration elements for the SIP2 Self Check.

Example of the table:

```
[HOST]
error_correction = N
institution_id = ALEPH
online_status = Y
checkin_ok = Y
checkout_ok = Y
stat_update_ok = Y
offline_ok = N
renew_policy = Y
alert = Y
sc_language = 001
pin_required = N
match_id_type = 01
item_transfer = N


[ALERT]
serveradmin = <mail-address>

[DEBUG]
verbose_msg = N
level = 0

[TIMEOUT]
retries_allowed = 010
timeout_period = 030

[SIP]
field_delimiter = |
protocol_version = 2.00
return_message = N
extensions = N

address_type = 1
shelf_count = N
use_sub_library_code = Y
item_identifier = REC-KEY
its_extensions = N
use_adm_privileges = Y

[CHAR-CONV]
input = 8859_1_TO_UTF
output = UTF_TO_8859_1


[MESSAGES]
checkout_message = Y
checkin_message = Y
checkin_fee = Y

[STATUSES]
missing = MI,MS
```

tab_sip2_sort_bin

Location of the table: tab directory of the **ADM** library

Purpose of the table: Self-Check sorting bin setup

This table is used to set the sort bin value when an item is returned.
First line that matches will be used.

Structure of the table:

Col. 1 Sort bin
Col. 2 Status of loan, A = Active, L = Lost, C = Claimed lost
Col. 3 Hold request, Y = Is requested, N=Not requested, S=On Hold Shelf
Col. 4 Photocopy request, Y = Is requested, N=Not requested
Col. 5 Sublibrary code
Col. 6 Collection code
Col. 7 Item status
Col. 8 Item call number (truncated)
Col. 9 Field tag from the title record
It is possible to use virtual or expanded fields. For this purpose you have to use the relevant expand-routines in section SIP2SERVER in ./xxx01/tab/tab_expand.
Col. 10 Subfield of tag defined in col. 9
Col. 11 Truncated text to compare with contents of subfield defined in col. 9
Example of the table:

```
!1 2 3 4   5     6   7                                     8
9    10                        11
!!-!-!-!-!!!!!-!!!!!-!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
-!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
30 # # # ##### ##### 11
02 # # # ##### ##### 02
09 # S # WID   ##### 02
21 # Y # LAW   ##### ##
23 # S # LAW   ##### ##
11 # S # ##### ##### ##
26 # Y # ##### ##### ##
```

tab_sip2_translate

Location of the table: tab directory of the **ADM** library

Purpose of the table: Self-Check field definitions

This table defines the expansion for Self-Check fields:

Structure of the table:
Col. 1   Type of line:

- MEDIA-TYPE - enumerated material type
- STATUS - circulation status

Col. 2 Self-Check definition

Col. 3 ALEPH definition

Col. 4 Description

Example of the table:

```
!    1            2       3          4
!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
MEDIA-TYPE      000              other
MEDIA-TYPE      001   BOOK       book
MEDIA-TYPE      002   ISSUE      magazine
MEDIA-TYPE      003              bound journal
MEDIA-TYPE      004              audio tape
MEDIA-TYPE      005              video tape
MEDIA-TYPE      006              CD/CDROM
…!
ITEM-STATUS     01               other
ITEM-STATUS     02               on order
ITEM-STATUS     03               available
```

tab_sort

Location of the table: tab directory of the library

Purpose of the table: Defines fields for sorting

Related table(s):

1) pc_tab_sear.<lng>
2) www_tab_short.<lng>
3) tab00.<lng>
4) pc_server_defaults
5) tab_filing
6) tab01.<lng>

The tab_sort table defines the field and subfield(s) assigned to a sort key. Up to 5 alternative field/subfield combinations can be defined for each sort key. In this table, each of the alternatives is defined in a group of three columns. The subfields can be defined as "use" or "remove", where a minus sign is used to denote "remove".
The last two columns (18 and 19) are used for defining positions from a fixed field.
This table, used in conjunction with:

1) www_tab_short.<lng> (for WEB OPAC) sort option in www_server.conf sort option in pc_server_defaults

tab00.<lng> - Access file codes and names (for building z101 sort keys according to filing procedure).

Structure of the table:

Col. 1 Sort key number (99 different sort options may be defined)

Col. 2 Filing procedure as defined in tab_filing. Used for building the filing key for sort key (Z101). The code entered here should match the desired filing procedure.

Col. 3 Field to be taken for sorting. Wildcard may be used.

Col. 4 Subfield to be taken for sorting. If left blank the entire field is taken.

Col. 5 Non-filing indicator for field (1/2)

Col. 6-17

2^{nd} – 4^{th} alternative field codes + subfield + non-filing indicator

Col. 18 For fixed fields, starting position for sorting information; enter 00 for variable fields

Col. 19 For fixed fields, number of characters to take for sorting of field; enter 00 for variable fields

Example of the table:

```
!1 2    3     4   5   6     7   8   9    10   1  12    13   4  15    16   7 18 9
!!-!!-!!!!!-!!!!!-!-!!!!!-!!!!!-!-!!!!!-!!!!!-!-!!!!!-!!!!!-!-!!!!!-!!!!!-!-!!-!
!
01 95 008           260## c                                                 08 0
4
02 01 1#### a                                                               00 0
0
03 11 245## a     2 240##       2                                           00 0
0
04 22 050##         LOC## hjl                                               00 0
0
05 11 440## av    2                                                         00 0
```

tab_sub_lib_group.<lng>

Location of the table: tab directory of the **ILL** library

Purpose of the table: Define groups of sublibraries within the same ADM. The group can be used as the Circ-Sublibraries of ILL Unit, if more than 20 sublibraries need to be defined for a specific ILL Unit.

Structure of table:

Col 1 Group Code. The code of the group

Col 2 Group Name. A readable string that is the group's name.

Col 3-12 Sublibrary Code. The code of the sublibrary that belong to the group.

In case where more than 10 sublibraries are assigned to a group, define another line for the same group code. No need to repeat the group name in Col. 2.

Example of the table:

```
!  1                 2                  3     4     5     6     7     8     9     10    11
12
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-
!!!!!-!!!!!
GRP01 Campus sublibraries             WID   LAW   MED   HIL   HYL
```

tab_sub_lib_sort

Location of the table: tab directory of the **BIB** library

Purpose of the table: Sublibraries list sort types

Related table(s):

1) edit_doc_999.<lng>
2) tab_sub_library.<lng>
3) tab_item_list_order

This table controls the order of the list of sublibraries in the following ALEPH functions:
Web edit_doc_999.<lng> display for ITM3 and ITM4 lines:
WWW-FULL-ITM3
WWW-FULL-ITM4
Web Holdings screen - The list of sublibraries in the combo-box:
www_tab_short.<lng>
- columns ITM,ITM+, LOC, LOC+, PSTS
WWW-SHORT
list of item locations in GUI SEARCH function's full record display:
PC-FULL-ITM
list of cash owing in Web OPAC->USER->Cash Transactions:
WWW-BOR-CASH
Cash list divided by sublibraries in the GUI:
CASH-LIST
Vendor list, order units in the GUI:
VENDOR-LIST
Web Customize Brief View screen - For LOC and PST lines
WWW-FULL-LOC-PST

The sort identifiers are:
00 - by sublibrary name, taken from tab_sub_library.<lng>
01 - by profile. This sets which sublibrary will display first in the list, after which the sort is by sublibrary name. Not relevant for GUI: PC-FULL-ITM, VENDOR-LIST, CASH-LIST
02 - by profile to determine the first sublibrary, and then by the tab_item_list_order table. The table can have .<base extension>, in which case there can be multiple tables. The system will choose the table that matches the logical base that is in effect. If there is no matching .<base extension> table, the system will use the generic tab_item_list_order table. Relevant only for : WWW-FULL-ITM3, WWW-FULL-ITM4, WWW-

SHORT, WWW-ITEM-LIST-FILTER, PC-FULL-ITM, WWW-FULL-LOC-PST

Structure of the table:

col. 1 Function name:

col. 2 Sort: option:

- 00 - by library name
- 01 - by library name, with "preferred library" (determined by profile) in first position
- 02 - by order of tab_item_list_order, with "preferred library" (determined by profile) in first position.

Example of the table:

```
!    1                2
!!!!!!!!!!!!!!!!!!!-!!
WWW-FULL-ITM3         00
WWW-FULL-ITM4         00
WWW-SHORT             00
WWW-BOR-CASH          01
WWW-ITEM-LIST-FILTER 00
PC-FULL-ITM           00
VENDOR-LIST           00
CASH-LIST             00
```

tab_sub_library.<lng>

Location of the table: alephe/tab directory

Purpose of the table: List of library codes and sublibrary codes

Related table(s):

1) tab27
2) tab37
3) tab38
4) tab15.<lng> (col. 6)
5) tab16 (col. 7)
6) tab17 (col. 8)
7) tab_www_item_desc.<lng>.

This table defines the sublibraries of all the ADM libraries on the server. In the list of items display in the ALEPH Web OPAC, it is possible to *gather together* all the sublibraries in one list, regardless of the connection to separate ADM libraries. For all other ALEPH functions (USER, CIRC, ITEMS, SERIALS, ILL, ACQ) it is required that the user *connect* to a specific ADM library (for example, USM50).

The number of sublibraries in the table is not limited. However, there are places in the system where the total number of sublibraries relevant to the situation cannot exceed 200. For example, when sorting a list of items or list holding libraries for a single bibliographic record.

Connect in GUI updating modules is to an ADM library (which is a group of sublibraries). This is a physical group, and one sublibrary can be assigned to one ADM library ONLY. Each sublibrary code must be unique in this table (unless a special setup for testing conversions has been installed by **Ex Libris**).

Library names (col. 5):
Names of the sub_libraries display in Web OPAC and GUI modules. For libraries that use more than one language interface, each language is a separate tab_sub_library.<lng> table. All rows and columns should be exactly the same, except for **column 5 (library name).** Different library names for Web and GUI can be assigned by adding the extension PC or WWW to the table name.
For example:
tab_sub_library.eng.PC
tab_sub_library.eng.WWW

Note! In you wish to define define alternative display text for Web OPAC use tab_www_item_desc.eng.

Library types (col .2):
In addition to actual sub_libraries (library type 1), some of the sub_libraries are defined for
patron records (type 2), in which case the code can be ALEPH or the ADM library code),
delivery of requested material (type 3, must be coordinated with tab27, tab37 and tab38), and for
transfer of material to a reading room (type 4, must be co-ordinated with tab27).
It is possible to define administrative units for acquisitions orders activities (type 5 - Ordering Unit).

Link to tab15.<lng> (col. 6):
The code registered in this column identifies the lines in tab15.<lng> (Item statuses) that are relevant for this sublibrary. The code here must match the code in column 1 of tab15.<lng>.
This allows for the definition of statuses in tab15.<lng> that are used in common by a number of sublibraries.

Link to tab16 (col. 7):
The code registered in this column identifies the lines in tab16 (Fines, due dates and loan limits) that are relevant for this sublibrary. The code here must match the code in column 1 of tab16.
This allows for the definition of lines in tab16 that are used in common by a number of sublibraries.

Link to tab17 (col. 8):
The code registered in this column identifies the lines in tab17 (Library hours) that are relevant for this sublibrary. The code here must match the code in column 1 of tab17. This allows for the definition of lines in tab17 that are used in common by a number of sublibraries that share a set of open/close hours.

Link to patron record (col. 9-13):
Columns 9 through 13 are used to define the preferred patron record, when matching the patron and the item to a line in tab16 (loan due dates, loan, request and renew limits, and so on). The five columns are alternatives, and the system checks one after the other, in an attempt to find a matching patron record.

For example, if the first of these 5 columns has sublibrary XYZAB, but the patron does not have an XYZAB patron record, the next of the 5 columns will be consulted, and so on. If, in the end, no matching patron record is found, the patron does not have privileges for this item, and he cannot take it on loan.

Therefore, at least one sublibrary code is mandatory for sublibraries type 1, and sublibrary type 2 must have one sublibrary code.
For example, for **library type 1**:

ULINC USM50 ALEPH

- checks for ULINC patron record, if not found checks for
- USM50 (general library patron) and lastly checks for
- **ALEPH** (general library patron)

UHLTH

- patron can loan items from UHLTH library only

For example, for **library type 2**:

ALEPH

- the system uses the patron status in the **ALEPH** patron record in order to find the matching line in tab16 for global loan and hold limits.

Note that this table is limited to 200 libraries; 1000 sublibraries and 1000 bases.

Structure of the table:

Col. 1 sublibrary

Col.2 type

- 1 - full sublibrary (patron and items)
- 2 - patron sublibrary only (no items)
- 3 - pickup location that is not a library
- 4 - reading room sublibrary (1 + special request handling)
- 5 - Ordering unit

Col. 3 ADM library

Col. 4 ALPHA

Col. 5 Sublibrary name

Col. 6 Group ID from tab15.<lng>

Col. 7 Group ID from tab16

Col. 8 Group ID from tab17 (Only for Item Sublibraries)

Col. 9 First Patron privileges record (Z305 sublibrary or **ALEPH**). This field is mandatory. If you do not want to give alternatives put the sublibrary code as listed in column 1.

Col. 10-13

Second to fifth Patron privileges record

Example of the table:

```
! 1   2   3   4             5                     6     7     8     9    10   11
!    12   13
!!!!!-!-!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!
!!-!!!!!-!!!!!
ALEPH 2       L General Patron                        ALEPH       ALEPH
!*
USR00 2       L USR00 Library                         ALEPH       ALEPH
!*
HOME  3       L Home address
!*
BOX   3       L In-house mailbox
!*
ILLDT 3       L ILL Department                        ALEPH       ALEPH
USM50 2 USM50 L Exlibris Demo Libraries               USM50       USM50 ALEPH
UARCV 1 USM50 L Archives Library                15A   UARCV UARCV UARCV ALEPH
UEDUC 1 USM50 L Education Library               15A   16A   17A   UEDUC USM50 ALE
PH
UELEC 1 USM50 L Electronic Resources            15A   UELEC UELEC UELEC USM50 ALE
PH
URLEC 4 USM50 L UELEC Reading Room              15A   UELEC UELEC UELEC USM50 ALE
PH
```

tab_sub_library_address.<lng>

Location of the table: alephe/tab directory **OR** tab directory of the library

Purpose of the table: Addresses of sublibraries for printouts

Related table(s):

1) form_sub_library_address
2) tab_sub_library.<lng>

This table defines addresses of branch libraries for printouts (for example, orders to vendors, overdue letters to patrons, and so on.)

It is used in conjunction with the library's /tab/ form_sub_library_address table, using column 2 as the link between them. Each address can be up to 50 characters per line, and up to 10 lines in length.

Note:
This table can be defined in both alephe/tab and in the tab directory of each of the ALEPH_LIBS libraries. This means that the table can be split to subtables.

The system checks in the following order:

1. library/tab table is searched for an address with the type that is defined in form_sub_library_address.
2. alephe/tab table is searched for an address with the type that is defined in form_sub_library_address.
3. library/tab table is searched for an address with the default type ("1").
4. alephe/tab table is searched for an address with the default type ("1").

Each address can be up to 50 characters per line, and up to 10 lines in length.

Structure of the table:

Col. 1	Sublibrary code (##### can be used as wildcards). Alternatively, the code of the ADM library can also be used to denote an address common to all branches.
Col. 2	Address type – if not defined defaults to type 1. • 1 - default address • 2 - Address for Circulation forms • 3 - Address for Acquisition forms • 4 - Address for ILL forms
Col. 3	Address line – up to 10 lines, 50 characters each line

Example of the table:

```
! 1   2                        3
!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
USM50 1 Ex Libris University Libraries
        222 Aleph Causeway
        Chicago, IL 60614
        thechoice@exlibris.co.il
        Tel# 773.404.5527
UARCV 1 Archives Department
        Lincoln Library
        808 Log Lane
        Chicago, IL 60614
        thechoice@exlibris.co.il
```

tab_sublibrary_grouping

Location of the table: tab directory of the **ADM** library

Purpose of the table: Define groups of sublibraries within the same ADM. The table is used for determining the trigger's department when a Rush Cataloging request is submitted. The department is set to be the code equivalent of the requested item's sublibrary.

Structure of table:

Col 1 Sublibrary Code.
Col 2 Group Code. The group code of the sublibrary

Example of the table:

```
!1         2
!!!!!-!!!!!!!!!!
WID   DEP1
LAW   DEP1
HIL   DEP2
##### DEP3
```

tab_subfield_punctuation

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: End-subfield punctuation definitions

Related table(s):

1) tab01.<lng>

This table is used to define subfield punctuation for fields. Punctuation for fields is necessary when the system automatically updates the bibliographic record from a linked authority record.

When the bibliographic record is updated from the authority database the system always uses the preferred term (1XX) from the authority record. Originally the bibliographic record may have more data then the authority record. This data should be retained. In MARC, authority records do not have end punctuation while bibliographic records do.

The tab_subfield_punctuation table is used to add end punctuation to the updated field. The table can be also used to add punctuation between the end of the preferred term from the authority record and the additional subfields retained from the bibliographic record (for example, between subfield $a – personal name - and subfield $t - title of MARC 21 600 field).

Structure of the table:

Col. 1 Program code

- A - Punctuation for fields updated from the authority database

Col. 2 Tag + Indicators

Col. 3 Subfield code

Col. 4 Following subfield code

Col. 5 Punctuation to add

Col. 6 If punctuation: if the field already ends with one of these punctuation signs, punctuation from the previous column will not be added.

Example of the table:

```
! 2     3 4 5          6
!-!!!!!-!-!-!!!!!!!!!!-!!!!!!!!!!
A 1#### a   .          .
A 1#### d   .          -.
A 110## b   .          .
A 600## a   .          ).
A 600## d   ,          ,
```

tab_suf_bor_stat

Location of the table: alephe/tab directory

Purpose of the table: Shared patron statuses

Related table(s):

1) tab31

This table defines the shared patron statuses

Structure of the table:

Col. 1 Patron status

Example of the table:

```
!1
!!
01
03
04
05
```

tab_suf_default

Location of the table: alephe/tab directory

Purpose of the table: Shared user definitions

This table defines two things:

which fields are shared, and which fields are local. Shared fields are registered in the USR01 record, and the local record field is updated from the USR01 record. Local fields are not written in the shared record.
the default values for the fields that are not shared (that is, local fields).

Structure of the table:

Col. 1 Internal Z30X code

Col. 2 Shared field status

- S = for shared
- L = for local

Col. 3 values for the local variations. In the USR01 library server, this column is blank. In the ADM library server this column contains the required value.

Example of the table:

```
! 1                   2   3
!!!!!!!!!!!!!!!!!!!-!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
z303-id              S
z303-primary-id      L
z303-proxy-id        L
z303-proxy-id-type   L
z303-open-date       S
z303-profile-id      L
```

tab_suf_lib

Location of the table: alephe/tab directory

Purpose of the table: Shared user definition

Related table(s):

1) tab31

The condition for the participation of an ADM library in the shared user mechanism is the presence of a definition in this table. Whenever the library is defined in the table, the SUF mechanism is applied. If the library is not defined the system assumes that the library does not participate.

The default of fields defined as local (according to the table alephe/tab_suf_default, cols. 2 + 3) will only be applied when a local record is created for the first time in the replication process. Updates will not consult this table for defaults. The contents of local fields will not be overwritten by the replication and original contents will be kept.

Structure of the table:

Col. 1 Library code

Col. 2 Alias code

Col. 3 Host name

Example of the table:

```
! 1     2     3
!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
USR01 USR01 metalib:6530
USR50 USR50 metalib:6530
USR51 USR51 metalib:6530
```

tab_suf_loc_borid

Location of the table: alephe/tab directory
Purpose of the table: Shared user definition of which patron ID types are to be replicated
Related table(s):

There are two options how to define the behavior of a local Aleph system:

- For which patron ID types the ID information has to be sent to the central SUF system (col.2=Y). If col.2 is set to N, the ID is only known in the local system.
- For which patron ID types the ID information received by the central SUF system is stored in the local system (col.3=Y). If col.3 is set to N, the information is ignored.

By default col.2 and col.3 are set to Y.
Col.1 can contain a single ID type. The hash (#) character can be used as a wildcard to define a group of ID types. The first line that matches col.1 with the current ID type is taken as the relevant line.

COL 1. Patron ID type (Z308-KEY-TYPE)
COL 2. Send ID information to central SUF;
COL 3. Accept ID information from central SUF;

tab_tag_text

Location of the table: tab directory of the library

Purpose of the table: Default tag values when creating records

Related table(s):

1) tab01.<lng>
2) formats.<lng>

This table allows for the defining of default tag values when creating records.

Structure of the table:

Col. 1	Tag
Col. 2	Record type (for example, BK, SE, CF)
Col. 3	Default Value

Example of the table:

```
!1   2                    3
!!!-!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
LDR BK ^^^^^nam^a22^^^^^^a^4500
LDR SE ^^^^^nas^a^^^^^^^^a^^^^^

008 BK ^^^^^^s2000^^^^^^^^^^^^^^^^^^000^^^eng^d
008 SE ^^^^^^c19009999^^^^r1^^^^^^^^0^^^^0eng^d
```

tab_type_config.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: Specifications for expand_doc_type

Related table(s):

1) expand_doc_type
2) tab01.<lng>

The tab_type_config table is used to define the specifications for the creation of the new field created by the expand_doc_type expand/fix program.

This program - together with the table - can be used, for example, to create a field that contains the format of the record based on the contents of other field(s) present in the record (for example, a combination of the LDR and the 008 field).

In the following example, the new field will contain the string 'FILM' according to a match performed on the values of both the LDR and the 008 field:

```
TYP      Film                LDR   F06-01     EQUAL      g
                             008   F33-01     EQUAL      m
```

In the above example:
the TYP field ($aFilm) is created when:
position 06 of the LDR contains a 'g' and
position 33 of the 008 field contains an 'm'.

Following is the structure of the new field:

```
TYP   L $$aFilm
```

Note: the tab_type_config table is a ***sample table***. If necessary, similar tables can be added and passed as parameters.

Note that in the tab_expand table, the name of the configuration file - for example, tab_type_config - should be added as a parameter in column 3. Following is a sample of the setup needed in the tab_expand table to use the new program:

```
!   1                   2                              3
!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
WORD       expand_doc_type                tab_type_config
```

Note in addition that this program can be also used as a fix program to actually add the new fields to the record.
In this case, the tab_fix table should be used instead of the tab_expand table.
Following is a sample of the setup needed in the tab_fix table to use this program:

```
! 1                 2                              3
!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
TYP   expand_doc_type                tab_type_config
```

Structure of the table:

col. 1 Target tag

col. 2 Format code

The value entered in this column is expanded into subfield $a of the new field created by expand_doc_type. If the column is left blank, then the format name (value of column 3) is added to subfield $a of the new field. For example, if the table contains:

TYP BK Book LDR F06-01 EQUAL a

then a new TYP field with the following content will be added when position 06 of the LDR field contains an 'a':

TYP L $$aBK$$bBook

If this column is left blank, the new field will be created/expanded as follows:

TYP L $$aBook;

col. 3 Format name

If a format code is present (column 2), then the format name is added/expanded into subfield $b of the new field. If no format code is defined, then the format name is added/expanded into subfield $a of the new field. For example,

TYP BK Book LDR F06-01 EQUAL a

a new TYP field with the following structure will be added when position 06 of the LDR field contains an 'a':

TYP L $$aBK$$bBook

If the format code column is left blank, the new field will be created/expanded as follows:

TYP L $$aBook

col. 4 Field tag

Field from the record used for determining the material type that is expanded into the new field. In the following line, the LDR (position 06 with 'a') is used to define that the record is for a book:

TYP BK Book LDR F06-01 EQUAL a

col. 5 Subfield(s)/position

The subfield codes or the fixed field position and length (of the field defined in column 4) to be checked

For for example, the program checks position 06 of the LDR for one position

TYP BK Book LDR F06-01 EQUAL a

Or the program checks subfield $a of the 490 field:

TYP Thesis 4901# a MATCH masters*

Note that for multiple subfield occurrences, set the third character to "A" (all)

TYP SHB 042## SbA MATCH SHB

col. 6 Match criteria

The match criteria in relation to the contents of the subfield or the fixed field positions defined in column 5. The available options are:

- EQUAL
- N-EQUAL
- EXIST
- N-EXIST
- MATCH
- N-MATCH

col. 7 Contents of the field or of the fixed field position that are used to match on (according to the match criteria defined in column 6). Use [ ] (square brackets) to enclose multiple values to match on. The relationship between the values is of type OR.

In the following line, the match is based on values 'e' or 'f' of position 06 of the LDR field:

TYP Map LDR F06-01 EQUAL [e,f]

col. 8 Case-sensitive matching flag

The table is limited to 500 lines.

Example of the table:

```
! 1   2            3                          4       5         6             7
!!!!!-!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!-!
TYP      Mixed Materials                FMT              EQUAL      MX
                                        008   F29-01     EQUAL      1
TYP      Conference                     111##            EXIST
TYP      Conference                     711##            EXIST
TYP      Conference                     811##            EXIST
TYP      Conference                     006   F00-01     EQUAL      a
                                        006   F12-01     EQUAL      1
TYP      Serial                         006   F00-01     EQUAL      s
TYP      Computer file                  007   F00-01     EQUAL      c
TYP      Microform                      007   F00-01     EQUAL      h
TYP      Videorecording                 007   F00-01     EQUAL      v
```

tab_vat_percent.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Codes for VAT percentages (for ACQ invoices)

This table defines codes, code texts and VAT percentages for VAT calculation in general invoices (Z77), line items (Z75) and budget transactions (Z601).

Note:
To use the VAT code mechanism, set the variable UseVATCode (under section "[invoice]") to "Y" in the file .\Acq\Tab\Acq.ini:

[Invoice]
UseVATCode=Y

When UseVATCode is set to "Y", a VAT Code can be assigned to a general invoice by an edit field equipped with a drop-down list (appearing instead of the "VAT Percent" edit field), in the General Invoice pane.

Structure of the table:

Col. 1 Code;

Col. 2 VAT Percent

Col. 3 Description

Example of the table:

```
! 1     2                    3
!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
NOVAT 00.00 No VAT
BOOKS 18.00 Books VAT
SERIA 16.50 Serials VAT
```

tab_vendor_sub_lib

Location of the table: tab directory of the **ADM** library

Purpose of the table: Sublibrary-specific and generic vendor fields

Tab_vendor_sub_lib is applicable for libraries that work with a two-level vendor. When creating/updating a sub-level vendor record, this table defines which group of fields of the Z70 table will be taken from the generic (USM50) vendor record and will be read-only for the sub-level record and which group of fields can be set specifically for the sub-level vendor record.

If ***Y*** is defined in Col.2:
the group of fields can be defined specifically for the sub-level vendor record.
If ***N*** is defined in Col.2:

the group of fields is controlled by the generic vendor record and can be read-only in the sub-level vendor record.

Following are the group definition of Z70 fields:

Group ID: GENERAL
Z70-COUNTRY
Z70-MATERIAL-TYPE
Z70-STATUS

Group ID: CONTACTS
Z70-CONTACT-n (1 to 5)
Z70-NOTE

Group ID: ACCOUNTING
Z70-TERMS-SIGN
Z70-TERMS-PERCENT
Z70-ACCOUNT-M
Z70-ACCOUNT-S
Z70-VENDOR-ACCOUNT

Group ID: ORDER-DISPATCH
Z70-DEFAULT-ORDER-DELIVERY
Z70-LE-LETTER-TYPE
Z70-LI-LETTER-TYPE
Z70-LE-SEND-METHOD
Z70-LI-SEND-METHOD

Group ID: EDI
Z70-EDI-VENDOR-CODE
Z70-EDI-VENDOR-CODE-TYPE
Z70-ED-SEND-METHOD
Z70-ED-LETTER-TYPE

Group ID: MATERIAL-DELIVERY
Z70-DELIVERY-TYPE-n (1 to 5)
Z70-DELIVERY-DELAY-n (1 to 5)

All other Z70 fields are either system set or taken from the generic vendor.

Structure of the table:

col. 1 Group ID of Z70 field

col. 2 The group's fields are/are not read from the vendor sublibrary/order unit record

Y=read from vendor sublibrary/order unit Z70 record.

N=read from generic vendor Z70 record.

Example of the table:

```
! 1                              2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!
GENERAL                         N
CONTACTS-NOTE                   Y
ACCOUNTING                      Y
ORDER-DISPATCH                  N
EDI                             Y
MATERIAL-DELIVERY               N
```

tab_version

Location of the table: alephe/tab directory

Purpose of the table: Client definitions to access the pc_server

Defines which clients are allowed to access the pc_server.
Note that definitions such as SER1.0 and other constants are used for non-ALEPH GUI clients, for example, Self-Check and so on.
These constants are defined by the developers of the external product and not by ***Ex Libris.***

Structure of the table:

Col. 1 Version

Col. 2 Format Type

Col. 3 Compress

Col. 4 Encryption

- 0 - none
- 1 - ALEPH
- 2 - DES
- 3 - Kerberos
- 4 - LDAP (Authentication only);
- 5 - Aleph Encryption and LDAP Authentication;

Col. 5 Encryption key

Col. 6 Service Prefix

Example of the table:

```
!  1     2   3 4    5          6
!!!!!!-!!!!!-!-!-!!!!!!!!-!!!!!!!!!!
SER1.0 ALEPH Y 0
16.02  ALEPH Y 0
…
```

tab_weights

Location of the table: tab directory of the **BIB** library

Purpose of the table: Weight factors for determining candidate records

Related table(s):

1) tab_match_script
2) tab01.<lng>

The tab_weights table is called from the tab_match_script table as a parameter for the match_doc_filter program.
It performs checks on the match candidate record, and as a result of these checks, can accept or reject the candidacy.
Each check can be assigned a positive or a negative value (weight).

The sum of these values is compared to the threshold value. If the sum is less than the threshold value, the record is removed from the pool of candidates. If the sum is more than the threshold value, it remains in the pool of candidates.

The match_doc_filter_hvd program automatically rejects all matches if the incoming record format is not SE or BK, and automatically rejects matches if there is a mismatch on the FMT field.

First, a threshold weight is defined, with TRSLD in the first column. If the record does not meet the threshold weight, it is no longer regarded as a candidate.

The following lines define various programs, for checking a field, a subfield, or a position range in a fixed field. The outcome of each program check assigns a positive or negative value. Each program line can end in a script command for continuation. If no action is listed, the next line is performed.

Script commands for continuation are:
empty, the default command is to skip to the next tag
skip one line
skip two lines
skip three lines
skip to next tag
unmatch (remove record from match candidate consideration)
continue (to next line);

Currently available programs are:
match_year_2: checks that the date is within +/-2 years, or is exact match if date is less than the date given in parameters (col.4)
match: for a given tag defined in col.1, the content of any occurrence of the field in the incoming record is the same as the content of any occurrence of the same tag in the database record.
mismatch: content of incoming record tag and db tag are different.
collat_msmch: checks for the presence of "v." in the field/subfield. If present in one and not the other, the collat_msmch check succeeds. If present in neither, or present in both, this condition fails.
keywords: compares the percentage of matching keywords to the threshold percentage given as parameter in col.4. If the percentage of matching keywords passes the threshold, a weight is assigned. The weight assigned

is the percentage of matching keywords multiplied by the weight parameter in col.5.
1st_wd_mch - matches on the first word to occur before punctuation. For use with the imprint field. For example, If the 260$$a contained the text Washington, D.C. the match would be on Washington only.
order: program is the same as "keywords", with an additional check of word order (that is, the keywords found are in the same order). If the keywords are not in the same order, this weight check fails. Therefore, this check should be performed BEFORE the "keywords" check.
edition: examines the edition (250) field, using the following parameters in col.4:
BOTH_MISSING - both records lack fields
ONE_MISSING_1 - field is missing from one of the two records and contains no numeric characters
ONE_MISSING_2 - field is missing in one record and contains "first" or "1st" in the other record
ONE_MISSING_3 - field is missing in one record and contains text other than "first" or "1st" in second record
FIRST - both records have field that contains "first" or "1st"
NUMERIC_MATCH - both records have fields that contain numbers - and the numbers match
NUMERIC_MISMATCH - both records have fields that contain numbers but the numbers do not match

Structure of the table:

Col. 1 Tag for comparison, or TRSLD. The table MUST begin with a TRSLD line, which defines the threshold weight that a record must pass in order to be considered a match. The TRSLD line has only col.5 (weight) defined.

Col. 2 Subfield code, or FNN-NN for fixed fields

Col. 3 Action – as described above

Col. 4 Action Parameters

Col. 5 Weight assigned, according to the success or failure of the action. The weight is expressed as plus or minus (+020, -010)

Col. 6 Script command for continuation:

- empty, the default command is to skip to the next tag
- skip one line
- skip two lines
- skip three lines
- skip to next tag
- unmatch (remove record from match candidate consideration)
- continue (to next line)

Example of the table:

```
! 1     2           3                                      4           5    6
!!!!!-!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!-!!!!!!>
TRSLD                                                                 +050
100## aqd    match                                                    +055
100## abc    mismatch                                                 -055
```

Note: use the keyword "NORMALIZE" in the 4th column of "match"/"mismatch" line in order to compare normalized data. In the "order"/"keywords" lines, specify the word breaking routine to use after the percent sign, separated by a comma. For example, to use normalization of the 100 field and the word breaking routine "01":

```
100## aqd    match                          NORMALIZE            +055
100## abc    mismatch                                            -055

245## abn    order                          33%                  +005
245## abn    keywords                       33%,01                +011
```

tab_word_breaking

Location of the table: tab directory of the **BIB** and **AUT** libraries

Purpose of the table: Defining word-breaking procedures.

Related table(s):

1) tab_character_conversion_line
2) tab_locate
3) unicode_to_word_gen
4) tab11_word
5) tab_expand (related to Acronymic titles)

A ***word*** is a group of characters between blanks, from beginning of field to first blank and from last blank to end of field. Most of the following procedures either create blanks, or change blanks into something else (so they will not split). In addition, after the text has been broken into words, a character conversion table is used to change texts (for example, u with umlaut to ue). The system uses the character conversion table that is listed for the WORD-FIX line in the tab_character_conversion_line table. Conversion is used for both parsing the FIND query and for word building.

For some of the procedures, characters to be considered are defined in column 4 (for example, in **to_blank** and **compress**). The character can be entered from your keyboard or can be in Unicode notation, by entering U+<hexa value> (for example, U+002E).

Note: Word routine "90" is reserved for the system. It is used when parsing a FIND query.

Word routines “93” and “94” are used if the searched text is CJK. The “93” routine is used if the search is without adjacency. The “94” routine is used is the search routine is with adjacency.

Word routine "97" is used when parsing a FIND query for the "locate" function (defined in tab_locate).

In the default setup, word routine 92 has been defined for Acronymic Titles. Note that for this to work correctly, tab_expand must also have the following line defined:

```
WORD       expand_doc_acronym_title       AA,BBB##C,DD
```

Where AA is the format, BBB is the code, C is the subfield and DD is the equivalent word breaking routine number from tab_word_breaking.

The table can be up to 500 lines long.

A list of available procedures that can be used:

Procedure	**Action**	**Example**
2_hyphen	changes 2 adjacent hyphens to blank. Particularly necessary for 505 (contents) field, which uses -- as a separator.	
abbreviation	compress a dot between single characters	
blank_to_caret	change blanks to caret	to ^
compress	compress (that is, strip) the characters listed in col. 4	
compress_blank	delete blanks	
compress_h_quote	strips the Hebrew Gershayim sign (" - inverted commas) between two Hebrew letters, in words used as initialisms or abbreviations	`Tana"kh - The Bible`
del_subfield	delete subfield sign	$$x
force_delimiter	change subfield sign ($$x) to blank-z-blank. This prevents words across subfields being considered adjacent	
marc21_041	for separating languages in MARC 21 field 041	engfreger changes to: eng fre ger
numbers	compress a comma and a dot between numbers	100,000 or 100.000 changes to 100000
subf_to_sign	change 2nd and subsequent subfield signs to the single character listed in col. 4	
to_blank	change characters listed in col. 4 to blank	
to_blank_2	change characters listed in col.4 to blank, if the character is followed by a blank. For example: 01 # to_blank_2 .,:;	Schiller, Friedrich" will be changed to "Schiller Friedrich" if comma is listed in col. 4.

Procedure	Action	Example
2_hyphen	changes 2 adjacent hyphens to blank. Particularly necessary for 505 (contents) field, which uses -- as a separator.	
	Note that the above is relevant only if the actual value of these characters is retained in the /alephe/unicode/ unicode_to_word_gen table. for example, Jones, Arthur - comma is changed to blank one,two,three - commas are retained XI.1.1.3 - dots are retained. This can be particularly useful for shelf or class numbers.	
The system automatically performs: to_lower:	change case to lower	BOOK to book

The following procedures are used in case of CJK:

Procedure	Action
split_cjk	The segmentation is performed according to the Z113 table (Chinese dictionary) from left to right. In addition, the text is split, character by character
cjk_to_word	The text is divided into words from right to left according to the longest word principle by using the Z113 table (Chinese dictionary).
cjk_split1	Each CJK character is considered as a word when defining segmentation routines that are performed on the search string.
cjk_split3	Words are determined according to a pre-defined dictionary (z113).
cjk_simplified	Characters are translated to a simplified form.
cjk_input_adj	Each CJK character is considered as a word when searching with adjacency.
cjk_input	Text is divided into words from right to left according to the longest word principle by using the Z113 table (Chinese dictionary). The next word starts after last character of the previous word.

cjk_2gram_lng	This routine is used with a parameter. Allowed values are CONC and NO-CONC: • cjk_2gram_lng with parameter: CONC Words are: o All Bi-Gram segments of the concatenation of consecutive CJK substrings of the same writing system (Chinese, Hangul or Kana) o All space delimited non CJK substrings ("Latin Words") o All space delimited single CJK characters o All Chinese characters o All Hangul characters that are a one character Korean word (defined in the /alephe/unicode/tab_cjk_single_char_word table) o All normalized Katakana characters that are the normalized form of one Kana character Japanese word (defined in the /alephe/unicode/tab_cjk_single_char_word table) • cjk_2gram_lng with parameter: NO-CONC Words are all Bi-Gram segmentation of the CJK space delimited substrings. In other words, no concatenation of the CJK substrings is done before the Bi-Gram segmentation. In addition, all "Latin words" are considered words.
cjk_2gram_all	Words are all Bi-Gram segments of the concatenation of the whole text (including non CJK text).
cjk_add_single	Every CJK character is added as a word in the index.
cjk_add_space	Insert space between characters of different writing systems.
morpheme_index	To be used only for segmentation during the indexing process. Creates all possible substrings that are concatenation of successive words in the given field When this routine is used for segmentation during the indexing process, the morpheme_search routine must be used during the segmentation of the FIND string.
morpheme_search	**To be used when morphem_index is used in the indexing process.**

Note that the system automatically carries out triple posting for hyphens and apostrophes:
as separate words
as is (with hyphen/apostrophe)
with hyphen/apostrophe compressed.

For example, twenty-five is indexed as:
twentyfive
twenty
five
twenty-five

Both the hyphen and the apostrophe MUST NOT be included in any of the word breaking procedures defined in this table.
In addition, the "hyphen" (solidus) and the apostrophe must be left with their actual value (002F) in alephe/unicode/unicode_to_word_gen

Structure of the table:

col. 1	ID - Two-digit identifier of the word breaking routine. This identifier is used in column 6 of tab11_word
col. 2	ALPHA (not used)
col. 3	Name of the word breaking procedure
col. 4	Parameters for the word breaking procedure (when relevant)

Example of the table:

```
!1 2       3                                   4
!!-!-!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
01 # del_subfield
01 # abbreviation
01 # numbers
01 # 2_hyphen
01 # compress_h_quote
01 # to_blank              -!@#$%^()_={}[]:";<>,.?|\
01 # split_CJK
01 # to_lower
!
03 # del_subfield
03 # abbreviation
03 # numbers
…
```

Note: If you wish to allow patrons to search for a term such as ***C++*** the plus sign (+) should **NOT** be defined in the list of characters that converts the character to a blank.

tab_www_item_desc.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: defines display text for Web OPAC

Related table(s):

1) tab15.<lng> - col. 5
2) tab40.<lng> - col. 4
3) tab_sub_library.<lng>- col. 5

This table defines alternative display text for Web OPAC.

Currently it is only available for tab15.<lng>, tab40.<lng> and tab_sub_library.<lng>.

Translation text for the original text in tables tab15.<lng>, tab40.<lng> or tab_sub_library.<lng> must be added

If the alternative display is not defined in these tables, then the text from the original table will be displayed.

Note that one-to-many translation is not allowed. The same text may appear only once in Col.2 of the table.

Structure of the table:

Col. 1 Code fromor original table:
ITEM-STATUS for tab15.<lng>
COLLECTION for tab40.<lng>
SUB-LIBRARY for tab_sub_library.<lng>
Col. 2 Original text
Col. 3 Text to be displayed in Web OPAC

Example of the table:

```
CHARACTER_CONVERSION=8859_8_TO_UTF
!COLLECTION      MORSE MUSIC
Morse Music - Web
COLLECTION      MORSE MUSIC AUDIOCASSETTE
Morse Music - Web
COLLECTION      MORSE MUSIC RESERVES
Morse Music - Web
COLLECTION      MORSE MUSIC RESERVES AUDIOCASSETTE
Morse Music - Web
COLLECTION      Depository
Web Depository test
```

tab_xyz

Location of the table: tab directory of the **BIB** library

Purpose of the table: defines the "XYZ" index that should be used to strip subject subdivisions.

Related table(s):

1) tab00.<lng>

This table is used to define the "XYZ" index that should be used to strip subject subdivisions (subfields $$v, $x, $y and $z) from the bibliographic subject heading in order to find the more general term in cases where there is no match between the original subject heading from the bibliographic library and the headings in the authority library.

Structure of the table:

Col. 1 Headings index code as defined in tab00.<lng>

Col. 2 Stripped headings index used to find the more general term in cases where there is no match between the original subject heading from the bibliographic library and the headings in the authority library

Example of the table:

```
! 1     2
!!!!!-!!!!!
SUB   XYZ
SUL   SULX
SUM   SUMX
```

tab_yankee_def

Location of the table: tab/import directory of the **BIB** library

Purpose of the table: Default values the Generic Vendor Records loader (file-96)

This table defines the default values for fields in the Item, Order and Budget records that are created using the Generic Vendor Records loader (file-96).

Structure of the table:

Col. 1 Z30 (item) /Z68 (order) /Z601 (budget) field name

Col. 2 Value to match in the incoming bibliographic record or hashes to indicate that the value in column 3 will be used regardless of the bibliographic record

Col. 3 Value to place in the field defined in column 1

Example of the table:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!
ORDER-SEQ                       ########## ybp

Z30-SUBLIBRARY                 BXMM       UMUSI
Z30-SUBLIBRARY                 BXMA       UEDUC
Z30-SUBLIBRARY                 ########## UEDUC
Z30-COLLECTION                  BXMM       GEN
Z30-COLLECTION                  BXMA       GEN
Z30-COLLECTION                  ########## GEN
Z30-MATERIAL                    ########## BOOK
Z30-ITEM-STATUS                 ########## 42
Z30-CATALOGER                   ########## YOHANAN
Z30-CALL-NO-TYPE                ########## 0
```

tab_z30_sort

Location of the table: tab directory of the **BIB** and **ADM** libraries

Purpose of the table: Sort options for items

Related table(s):

1) pc_tab_exp_field.<lng>
2) tab_attr_sub_library– type 7 when sorting by sublibrary
3) tab_sub_lib_sort– col. 1
4) fix_doc
5) tab_filing sort routine 98

This table allows for the defining of sort options for items. Different sort routines can be used in different parts of the system.

Note that the instances that are related to Web OPAC and GUI Search should be listed in the BIB library's tab_z30_sort. Other instances should be listed in the ADM library's tab_z30_sort.

The default item sort is registered for the Cataloging and Circulation modules in the GuiSys files in the "ItemSortType" line, and for the Acquisitions/Serials module in the GuiSys files under the "ItemSortType" and "SerialSortType" lines.

In order to view the list in a different order than the default, the Items List pane includes a drop-down list for choosing a different sort type.

The drop-down list is defined in the pc_tab_exp_field.<lng> table under the following menus:

SERIAL-SORT-TYPE: lists the sort options for the Items List displayed by selecting the Items node under the Subscription root of the Serials tab (Acquisitions/Serials module)
SERIAL-CHK-SORT-TYPE: lists the sort options for the Items List displayed by selecting the Check In node of the Serials tab (Acquisitions/Serials module)
SERIAL-GRP-SORT-TYPE: lists the sort options for the Items List displayed by selecting the Group node of the Serials tab (Acquisitions/Serials module)
ITEM-SORT-TYPE: lists the sort options for the Items List displayed by selecting the Item List root (for Cataloging, Acquisitions and Circulation modules).

The last sort option that was selected by the user serves as the default for the next time the list is displayed.

The sort types are system-set and identified by a two-digit number. The sort and identification is separate for "issue" items and all other item types.

The issue/non-issue sort types are not completely parallel, although the following is a guideline:
Issue 01 -> Non-issue 00 (enum)
Issue 02 -> Non-issue 04 (HOL sublib + enum)
Issue 03 -> Non-issue 05 (sublib + item sequence)

The choice might be whether or not to put the sublibrary as the first sort factor, where: library is first sort - ISSUE 02, 03, 04, 06 | NON-ISSUE 04, 05, 06

library is not first sort - ISSUE 00, 01, 07, 08 | NON-ISSUE 00, 01, 02 or 03

Sort identifiers for ISS.. "issue" items

00 - by numbering, ignores location, reverses enum b/c

if chronological-i(year) is spaces and
enumeration-a(volume) is spaces then
description+
item-sequence.
if chronological-i(year) **not** spaces then
chronological-i(year)+
enumeration-a(volume)+
enumeration-c(part)+
enumeration-b(issue)+
item-sequence

01 - by numbering, ignores location

if chronological-i(year) is spaces and
enumeration-a(volume) is spaces then
description+
item-sequence.
if chronological-i(year) **not** spaces then
chronological-i(year)+
enumeration-a(volume)+
enumeration-b(issue)+
enumeration-c(part)+
item-sequence

02 - by location (using HOL records), then numbering

if chronological-i(year) is spaces and
enumeration-a(volume) is spaces then
hol-doc-number+
description+
item-sequence.
if chronological-i(year) **not** spaces then
hol-doc-number+
chronological-i(year)+
enumeration-a(volume)+
enumeration-b(issue)+
enumeration-c(part)+
item-sequence

03 - by location (using sublibrary), then sequence

sub_library+
item-sequence

04 - by location (using sublibrary), then numbering

sub_library+

if chronological-i(year) is spaces and
enumeration-a(volume) is spaces then
description+
item-sequence.
if chronological-i(year) not spaces then
chronological-i(year)+
enumeration-a(volume)+
enumeration-b(issue)+
enumeration-c(part)+
item-sequence

06 - by sublibrary (preferred sublibrary by IP)

sublibrary name+
collection code+
chronological-i(year)+
chronological-j(year)+
chronological-k(year)+
description+
copy-id+

Note: ! * tab_attr_sub_library type 7 defines which sublibrary will be first in the list.

07 - by 85x

85x-type+
sublibrary+
collection+
linking-number+
if 85x-type is 4 or 5 then
supp-index-o+
if chronological **not** spaces and
enumeration **not** spaces then
chronological+
enumeration.
if chronological is spaces then
description.

08 - by 85x [like 07, with addition of Z30-COPY-ID]

85x-type+
sublibrary+
collection+
linking-number+
copy-id+
if 85x-type is 4 or 5 then
supp-index-o+
if enumeration not spaces then
enumeration+
if chronology not spaces then
chronology+
if enumeration not spaces or
chronology not spaces then

description.

09 - by vendor code

1. vendor code + sublibrary

10 - by numbering, ignores location, reverses enum b/c – with the numbers in the description expanded for numeric sort

2. same as 00, only that the numbers in the description are expanded for numeric sort

12 - by location (using sublibrary), then numbering

3. sublibrary+
4. collection+
5. **if** chronological-i(year) is spaces and enumeration-a(volume) is spaces then
6. description+
7. if chronological-i(year) not spaces then
8. chronological-i(year)+
9. enumeration-a(volume)+
10. enumeration-b(issue)+
11. enumeration-c(part)+
12. copy

13 - by barcode (using filing routine 98)

13. barcode

15 – by item's chronology, enumeration and order number:
Chronology level 1 'Year' (Z30-CHRONOLOGICAL-I)
Chronology level 2 (Z30-CHRONOLOGICAL-J)
Chronology level 3 (Z30-CHRONOLOGICAL-K)
Enumeration level 1 'Volume' (Z30-ENUMERATION-A)
Enumeration level 2 (Z30-ENUMERATION-B)
Enumeration level 3 (Z30-ENUMERATION-C)
Order number (Z30-ORDER-NUMBER).

16 - by item's chronology, enumeration and copy ID:
Chronology level 1 'Year' (Z30-CHRONOLOGICAL-I)
Chronology level 2 (Z30-CHRONOLOGICAL-J)
Chronology level 3 (Z30-CHRONOLOGICAL-K)
Enumeration level 1 'Volume' (Z30-ENUMERATION-A)
Enumeration level 2 (Z30-ENUMERATION-B)
Enumeration level 3 (Z30-ENUMERATION-C)
Copy ID (Z30-COPY-ID).

The sort identifiers for non "issue" items are:

00 - by volume (enum or description) and location

if enumeration-a(volume) **not** spaces then
enumeration-a(volume)+
enumeration-b+

enumeration-c(part)+
sublibrary+
collection.
if enumeration-a(volume) is spaces then
description+
sublibrary+
collection

01 - by volume (enum + description) and location

enumeration-a(volume)+
enumeration-b+
enumeration-c(part)+
description+
sublibrary

02 - by volume (enum) and location

if enumeration-a(volume) **not** spaces then
enumeration-a(volume)+
enumeration-b+
enumeration-c(part)+
sublibrary.
else
description+
sublibrary

03 - by volume (description) and location and item status

if description is blank then
enumeration-a(volume)+
chronological-i(year)+
enumeration-b+
enumeration-c+
sublib+
item status
else
description+
sublib+
item status

04 - by HOL and volume (enum or description) and location

if enumeration-a(volume) **not** spaces then
hol-doc-number +
enumeration-a(volume)+
enumeration-b+
enumeration-c+
sublibrary+
collection.
else
hol-doc-number +
description+

sublibrary+
collection

05 - by location (using sublibrary), then sequence

sub_library+
item-sequence;

06 - by sublibrary (preferred sublibrary by IP)

sublibrary name+
collection code+
description+
copy-id

Note: ! * tab_attr_sub_library type 7 defines which sublibrary will be first in the list.

07 - by 85x

85x-type+
sublibrary+
collection+
linking-number+
if enumeration not spaces then
enumeration.
else
description.

09 - by vendor code

14. vendor code + sublibrary

10 - by volume (enum or description) and location - with the numbers in the description expanded for numeric sort

15. same as 00, only that the numbers in the description are expanded for numeric sort

11 - by by volume (description) and location and item status with the numbers in the description expanded for numeric sort

16. same as 03, only that the numbers in the description are expanded for numeric sort

12 - by location (using sublibrary), then numbering

17. sublibrary+
18. collection+
19. **if** chronological-i(year) is spaces and enumeration-a(volume) is spaces then
20. description+
21. if chronological-i(year) not spaces then
22. chronological-i(year)+
23. enumeration-a(volume)+
24. enumeration-b(issue)+

25. enumeration-c(part)+
26. copy

13 - by barcode (using filing routine 98)

27. barcode

Structure of the table:

Col. 1 Function code:

- **WWW-A** - Web OPAC (should be present in BIB library)
- **WWW-R** - Web Course reading (should be present in BIB library)
- **WWW-X** - "circ status" of X-Server
- **SEARCH** - Search function (should be present in BIB library)
- **SERIAL** - Serials client
- **CIRC** - Circulation client
- **ITEM-no.** - Items client
- **ITEM-BIND** - Items for binding
- **ACQ** - Acquisitions client
- **TREE** - Navigation window
- **BATCH** - Services
- **CIR-16** - Lost Item Report
- **HII** - Holdings Item Information (87x)
- **AHS** - Automatic Holdings Summary (86x)
- **86x** - for use by fix_doc_create_86x procedure
- **Z39** - Z39.50 server

Col. 2 Sorting order - ascending/descending for issue type items

Col. 3 Issue sort

Col. 4 Sorting order - ascending/descending for non-issue type items

Col. 5 Non-issue sort

Col. 6 Sublibrary sort. This column is a function name, and should point to column 1 of tab_sub_lib_sort (found in the BIB library's tab directory). In this way, different sublibrary sort methods can be defined for this table's sort functions. If no value is defined here, sublibraries will be sorted alphabetically by their code.

Example of the table in the ADM library:

```
!    1               2 3  4 5
!!!!!!!!!!!!!!!!!!!-!-!!-!-!!-!!!!!!!!!!!!!!!!!!!
HOLDING-STMT         A 07 D 00

WWW-R                A 90 D 00
SERIAL-0             A 03 D 00
SERIAL-1             A 01 D 01
SERIAL-2             A 00 D 03
SERIAL-3             A 07 D 04
SERIAL-4             A 06 D 06
```

Example of the table in the BIB library:

```
!    1               2 3  4 5
!!!!!!!!!!!!!!!!!!!-!-!!-!-!!-!!!!!!!!!!!!!
WWW-A                A 01 A 05
SEARCH               D 00 D 00
WWW-X                A 01 A 05
Z39-1                A 06 A 06
```

tab_z31_sort

Location of the table: tab directory of the **ADM** library

Purpose of the table: Definition of cash (Z31) sort types in various ALEPH functions

Related table(s):

1) pc_tab_exp_field.<lng>

The tab_z31_sort table is used to define the sort procedure and sort order (ascending or descending) for lists of cash records in various instances.

The default cash sort is registered for the Circulation module in the GuiSys file under the "CashSortType" line.

In order to view the list in a different order than the default, the Cash List pane includes a drop-down list for choosing a different sort type.

The drop-down list is defined in the pc_tab_exp_field.<lng> table under CASH-SORT-TYPE. This lists the sort options for the Cash List displayed by selecting the Patron Activity root in the Circulation module.

The last sort option that was selected by the user serves as the default for the next time the list is displayed.

The cash list in the Web OPAC is sorted using the entry WWW-USER in this table. The sort types are system-set and identified by a two-digit number.

The sort routines are:

00 - By the sort order defined as type "03" in tab_sort for the transaction's BIB record. This can be set to be the title of the BIB that is connected to the cash transaction.
01 - By the transaction type.
02 - By whether the transaction is a credit or a debit transaction.
03 - By the transaction date.
04 - By the transaction status ('C'losed, 'O'pen', 'W'aived or 'T'ransferred).
05 - By the sort order defined as type "02" in tab_sort for the transaction's BIB record. This can be set to be the author of the BIB that is connected to the cash transaction.
06 - By the Call Number (Z30-CALL-NO-KEY) of the item record related to the cash transaction.
07 - By the Barcode (Z30-BARCODE) of the item record related to the cash transaction.
08 - By the transferred number - transaction type. Group all cash transaction of the same transferred number (i.e. all cash transactions of the same shipment), and inside one transfer number - all cash transaction with the same type.
09 - By the transferred number - open date - requester name. Group all cash transaction of the same transferred number (i.e. all cash transactions of the same shipment), and inside one transfer number - group all cash transaction of the same date, and in the same date - by requester name.
10 - By the requester name.

Structure of the table:

Col. 1	Function name for Circulation client
Col. 2	Sorting order: Ascending/Descending
Col. 3	Sort routine as described above

Example of the table:

```
!    1                2 3
!!!!!!!!!!!!!!!!!!!-!-!!
CASH-1                A 00
CASH-1-D              D 00
CASH-2                A 01
CASH-2-D              D 01
CASH-3                A 02
CASH-3-D              D 02
CASH-4                A 03
CASH-4-D              D 03
CASH-5                A 04
CASH-5-D              D 04
CASH-6                A 05
CASH-6-D              D 05
CASH-7                A 06
CASH-7-D              D 06
CASH-8                A 07
CASH-8-D              D 07
CASH-9                A 08
CASH-9-D              D 08
CASH-10               A 10
CASH-10-D             D 10
WWW-USER              A 00
```

tab_z36_sort

Location of the table: tab directory of the **ADM** library

Purpose of the table: Definition of loan (Z36) sort types in various ALEPH functions

Related table(s):

1) pc_tab_exp_field.<lng>

The tab_z36_sort table is used to define the sort procedure and sort order (ascending or descending) for lists of loan records in various instances.

The default loan sort is registered for the Circulation module in the GuiSys file under the "LoanSortType" line.

In order to view the list in a different order than the default, the Loan List pane includes a drop-down list for choosing a different sort type.

The drop-down list is defined in the pc_tab_exp_field.<lng> table under LOAN-SORT-TYPE. This lists the sort options for the Loan List displayed by selecting the Patron Activity root in the Circulation module.

The last sort option that was selected by the user serves as the default for the next time the list is displayed.

Note that the loan list in the Web is sorted using the entry WWW-USER in this table.

The sort types are system-set and identified by a two-digit number.

The sort routines are:
00 - by Due-date
01 - by SubLibrary/Item status/Collection and Due-date
02 - Recalled items/Due date
03 - Recall items/Sublibrary/Status/Collection/Due date
04 - Rush recalled items/Recalled items/Due date
05 - Rush recalled items/Recalled items/Sublibrary/Status/Coll/Due date
06 - Loan Fine
07 - Letter Number/Letter Date
08 - Loan Status/Due date
09 - Loan Title

Structure of the table:

Col. 1	Function name LOAN-no. - Circulation client
Col. 2	Sorting order: Ascending/Descending
Col. 3	Sort routine as described above

Example of the table:

```
!    1                 2 3
!!!!!!!!!!!!!!!!!!!-!-!!
LOAN-1                 A 00
LOAN-2                 A 01
LOAN-3                 D 00
LOAN-4                 D 01
WWW-USER               D 09
DEFAULT                A 00
LOAN-5                 A 02
LOAN-6                 D 02
LOAN-7                 A 03
LOAN-8                 D 03
LOAN-9                 A 04
```

tab_z38_sort

Location of the table: tab directory of the **ADM** library

Purpose of the table: Definition of photocopy (Z38) sort types in patron's photocopy list

Related table(s):

1) pc_tab_exp_field.<lng>

The tab_z38_sort table is used to define the sort procedure and sort order (ascending or descending) for list of patron's photocopy requests.

The default photocopy sort is registered for the Circulation module in the GuiSys file under the "[PhotoList]" line.

In order to view the list in a different order than the default, the patron's photocopy List pane includes a drop-down list for choosing a different sort type.

The drop-down list is defined in the pc_tab_exp_field.<lng> table under PHOTO-SORT-TYPE. This lists the sort options for the patron's photocopy list in the Circulation module.

The last sort option that was selected by the user serves as the default for the next time the list is displayed.

The sort types are system-set and identified by a two-digit number.

The sort routines are:
00 - by photocopy number
01 - Sub library / requester name / photocopy number
02 - Pickup Location / Sub library / requester name / photocopy number
03 - Pickup Location / Status / requester name / photocopy number
04 - Sub library / Status / requester name / photocopy number
05 - Pickup Location / Sub library / Status / requester name / photocopy number
06 - Pickup Location / Status / Sub library / requester name / photocopy number
07 - Pickup Location / requester name / photocopy number

Structure of the table:

Col. 1	Function name PHOTO-no. - Circulation client
Col. 2	Sorting order: Ascending/Descending
Col. 3	Sort routine as described above

Example of the table:

```
!    1                2 3
!!!!!!!!!!!!!!!!!!!-!-!!
PHOTO-1               A 00
PHOTO-1-D             D 00
PHOTO-2               A 01
PHOTO-2-D             D 01
PHOTO-3               A 02
PHOTO-3-D             D 02
PHOTO-4               A 03
PHOTO-4-D             D 03
PHOTO-5               A 04
PHOTO-5-D             D 04
PHOTO-6               A 05
PHOTO-6-D             D 05
PHOTO-7               A 06
PHOTO-7-D             D 06
PHOTO-8               A 07
PHOTO-8-D             D 07
```

tab_z0101

Location of the table: tab directory of the **BIB** library

Purpose of the table: Brief record structure based on format, index and field of origin

Related table(s):

1) tab_filing
2) tab_z0101_text
3) formats.<lng>
4) tab00.<lng>
5) tab01.<lng>

This table is used to define for each particular heading, depending on the record's format and field of origin, the structure of the brief record (Z0101). The Z0101 record has both display text and filing text fields. The filing text is constructed using filing procedure 98 from tab_filing.
Structure of the table:

col. 1 Format code (for example, BK for book). Enter a specific record format, or use ## as a wildcard to indicate that the definitions are appropriate for any format.

col. 2 Headings code

Specify the headings code for which the brief specifications are relevant. For example, if the tab_z0101 table has the following line:

```
## AUT   100## TITLE-MAIN
```

Then the TITLE-MAIN text code defined in tab_z0101_text is the base for the brief record created for AUT headings (originated from the 100 field and for all record formats).

col. 3 Field tags and indicators for which the brief specifications are relevant. For example, if the tab_z0101 table has the following line:

```
## AUT   100## TITLE-MAIN
```

Then the TITLE-MAIN text code defined in tab_z0101_text is relevant for headings from the AUT only if they originated from a 100 field.

col. 4 The text code is used to define in which way the brief is constructed. Text codes are defined in the tab_z0101_text table.

Note that the brief is actually constructed from the text codes defined in columns 4, 5 and 6.

col. 5-8 Text code (see explanation for column 4).

col 9 AUT match code. Used to define the text that is used for searching a match in the authority database. Note that the system searches for a match in the Z0101 index of the authority library. Note that the match text is actually constructed from the match codes defined in columns 7, 8 and 9;

col. 10-11

As col. 9

Example of the table:

```
!1   2     3       4           5          6          7          8          9
     10         11
!!-!!!!!-!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!
-!!!!!!!!!!-!!!!!!!!!!
## AUT   100## TITLE-MAIN DATE                                          AUTHOR-REF
 TIT-M-REF
## AUT   110## TITLE-MAIN DATE
## AUT   111## TITLE-MAIN
## AUT   800## SERIES-ADD VOL800     DATE
## TIT   245## ME-ONLY    IMP-ONLY   DATE       TYPE                   AUTHOR-REF
 TTL-M-REF
## TIT   246## ME-OR-TI   DATE       TYPE                              AUTHOR-REF
 TTL-M-REF
## TIT   247## ME-OR-TI   DATE       TYPE                              AUTHOR-REF
 TTL-M-REF
!## TIT   130## TITLE-MAIN DATE
```

tab_z0101_text

Location of the table: tab directory of the **BIB** library

Purpose of the table: Text codes list and definitions for brief record (Z0101)

Related table(s):

1) tab_z0101_text
2) tab01.<lng>

The tab_z0101_text table is used to define the text codes that are used by the tab_z0101 table to create brief records (Z0101).
Note that the relation between the lines (for various occurrences of the text code) is an ***OR*** type of relation.
The text code defined in column 1 of the table can be repeated to determine that if the field tag (column 2) defined for the first occurrence of the text code is not present, then the text code can be built from other fields of the record.

Structure of the table:

col. 1 Text codes that are used in the tab_z0101 table to define the structure of the new brief record (Z0101).

Example:

If the tab_z0101 table has the following line:

```
## AUT   100## TITLE-MAIN
```

(the line defines that the TITLE-MAIN text code is the base of the brief record -Z0101- created for AUT headings from all bibliographic formats and created from the 100## field)

and if this table - tab_z0101_text - has the following lines:

```
TITLE-MAIN      100## t
TITLE-MAIN      240##
```

then the brief record for this type of headings will contain the title from subfield $t of the 100 field if present. If the subfield is not present, then the brief record will be created from the 240 field.

col. 2 Field from which the text code is created. If the field is not present in the record, the system continues reading the table until it finds a line with a matching field definition.

For example, if this table has the following lines:

```
TITLE-MAIN      100## t
TITLE-MAIN      240##
```

then brief records based on the TITLE-MAIN definition, will be built from subfield $t of the 100 field if present. If the subfield is not present, the system builds the brief record from the 240 field if present (and so on). Note that it is possible to add the following line as a place-holder in case none of the specified fields are present:

```
TITLE-MAIN      ^^^                     a
```

col. 3 Filter subfield. For example, if this table has the following lines:

```
TITLE-MAIN      100## t
TITLE-MAIN      240##
```

brief records created based on the TITLE-MAIN text code will be based on the 100 field only if subfield $t is present in the field. If the subfield is not present, the brief will be created from the 240 field if present.

col. 4 Subfields to include in the brief record (Z0101). If this column is left blank, the Z0101 record is built from all subfields present in the field defined in column 2.

col. 5 New subfield. If this column is left blank the original subfields will remain otherwise they will be stripped.

Example of the table:

```
!   1         2   3          4            5
!!!!!!!!!!-!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!
TITLE-MAIN 100## t t                      t
TITLE-MAIN 240##   adfhlmnopr             t
TITLE-MAIN 243##   adfhlmnopr
TITLE-MAIN 245##   ahnp
TITLE-ADD  700## t t
TITLE-ADD  240##
TITLE-ADD  243##
TITLE-ADD  245##
AUTHOR-REF 100## t -
AUTHOR-REF 100##
```

tab_z103

Location of the table: tab directory of the library

Purpose of the table: Defines which program will run for the building of links between records.

Related table(s):

1) edit_doc_999.<lng>

Structure of the table:

Col. 1 Program name

Col. 2 Program arguments

Example of the table:

```
!            1                          2
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!->
update_z103_lkr_extended       1way-par
```

The library's tab_z103 table defines the procedure(s) which are used to create links between records. The linking information is held in the document record, and the links themselves are held in the library's **Z103** table. Retaining the linking information in the document record acts as a safeguard, so that the Z103 table can be re-built from scratch if necessary.

Note: After any change in the definitions of tab_z103, links must be re-created using the Create Links Between Records (p_manage_12) batch procedure.

Links can be created based on:
the **ALEPH** LKR field,

MARC 21 linking fields,

UNIMARC linking fields,
the 5xx reference fields of the AUT library.

The **ALEPH** structure of BIB, ADM and HOL records requires an LKR field in order to link between these records. The MARC 21, UNIMARC and AUT links depend on various fields and subfields.

Links between BIB records can be displayed in the Web OPAC and GUI Search functions in the edit_doc_999.<lng> display. When a link field displays in the full record, clicking on the field invokes display of the linked record.

In addition to the LKR field there are some special programs for building links. The possible programs:

Name of Program	Explanation
Update_z103_lkr	ALEPH link
Update_z103_lkr_extended	ALEPH link, including additional filters
Update_z103_aut	for thesaurus NT,BT,RT - based on field 550, subfield w. Should be used in an AUT type library *only*.
Update_z103_dan	for DANMARC - based on 001 (Faust number) and 015 which contains the UP link. This requires that 001 be assigned to the FST index.
Update_z103_ids	based on MARC 21 linking fields, for creation of UP/DN, PAR and ANA links. Although based on MARC 21 links, there are departures from the MARC 21 standard.
Update_z103_mab	similar to 'dan' for MAB (German MARC) - based on tag 010 which contains the UP link and should be assigned to the FST index.
Update_z103_mab_ana	For MAB format: - to be run for building analytical links between records. To support this, two fields, ANU (analytical up) and AND (analytical down) may be added to edit_doc_999.<lng> .
Update_z103_uni	based on UNIMARC linking fields, as developed for Italy
Update_z103_uni_2	based on UNIMARC linking fields, as developed for Latvia

Structure of the LKR field

Subfield	Content	Explanation
$$a	UP	"up" link to another BIB type record. A record can have only one link of this type.
	DOWN	"down" link to another BIB type record. Multiple links are possible. This link is rarely used, since the link should be made in the UP direction
	PAR	parallel link from BIB record to BIB record
	HOL	link from BIB record to HOL record
	ADM	link from ADM record to BIB library
	ITM	link to items - see below

Subfield	Content	Explanation
	ANA	analytic links - see below
$$b	<sysno>	system number of linked record
$$l	<library>	library (default is current library and is not mandatory)
$$r	<MARC tag>	Reason for link. From $$r the system will create a caption to display before $$n and $$m. The caption table is defined in tab05 located in the tab directory of the library
$$n	UP link text	Text explaining up link
$$m	DN link text	Text explaining down link
$$y	<year>	
$$v	<volume>	
$$p	<part>	
$$I	<issue>	
$$d	<Fourth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$e	<Fifth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$f	<Sixth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$g	<Alternative Volume link>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$h	<Alternative Issue link >	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$j	<Second level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$w	<Third level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$o	<Fourth level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$q	<Alternative chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.

Structure of LKR/ITM field

Subfield	Content	Explanation
$$a	ITM	
$$b	<sysno>	system number of linked record – should be the system number of the ADM library, where the

Subfield	Content	Explanation
		physical item is registered.
$$n	UP link text	Text explaining up link (if necessary)
$$m	DN link text	Text explaining down link (if necessary)
$$y	<year>	as registered in ITEM record
$$v	<volume>	
$$p	<part>	
$$I	<issue>	
$$k		Filter for pages
$$d	<Fourth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$e	<Fifth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$f	<Sixth level of enumeration>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$g	<Alternative Volume link>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$h	<Alternative Issue link >	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$j	<Second level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$w	<Third level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$o	<Fourth level of chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.
$$q	<Alternative chronology>	applicable only when the update_z103_lkr_extended program is used in tab_z103 table and Z103X is defined.

Links of type ITM build Z103 records that link the cataloging records:
Z103 to closest "up" ($$y$$v$$p$$i is truncated until nearest match is found)
Z103 "up" to root, and creation of "dn" from root to all ADM's
Z103 "dn" to closest "up"

Note that this link type is used from a BIB record to an ADM record. An example of use is in the case of items being bound together. It does not create a link to the bibliographic record, only to the ADM. Therefore there will normally be no need to register subfields n and m for up and down notes.

LKR/ANA field

ANA links are between two BIB records and are intended to be used when linking an analytic record to the main series records.
The ANA link is made from the lower record to the higher one and the system automatically builds an UP/DOWN link between the two records as well as an ADM link to the items linked to the higher record.
If there are more than two tiers the UP/DOWN link will be from the lowest to the middle and from the middle to the highest. The ADM link will be to the ADM of the highest - that is, the third tier.

tab_z105

Location of the table: tab directory of the library

Purpose of the table: Record update messages between libraries

In some instances, updating of a record will trigger update of a record in a different database. For example:
update of a ***central*** BIB record should cause the same update in a ***local*** BIB record;
update of an AUT records should cause an update of a Z01 heading record.
The system accomplishes this by sending a message (Z105) as controlled by the tab_z105 table.

Structure of the table:

Col. 1 action that triggers a message (system set)

- UPDATE-DOC = Update of records (Z00)
- UPDATE-Z30 = Update of items (Z30)
- SEND-ILL-MSG = ILL transactions (SLNP-server)
- SUF-LOC = Update of patron information (for SUF)
- UPDATE-SDI = Update Z324 records for SDI
- INDEX-DOC = Indexing of records by UE-01(Z00)
- PERIODIC-INDEX = Update BIB LKR based on z30 item modification (enumeration/chronologic information)
- SEND-ORDER = Update BIB record when monograph order is sent to vendor
- UPDATE-ORDER = Update BIB STA field when first order record is created/arrives

Col. 2 message type

- 1=central to local message; local record is updated as a result of central record update
- 2=update of central HOL record from local HOL record
- 4=update of bibliographic z01 record because of authority database record update
- 5=update of central BIB record from local Z30 (item) record
- 6=ILL message
- 7 = Shared User File (SUF) - used to announce to the ADM library that an update has occurred in the USRxx lib
- 8 = Update of Z324(SDI-ready)record as a result of Z30 (item)
- 9 = update of union catalog from local BIB record
- a = replication of digital objects between DigiTool-3 and ADAM
- b = central to local message. Local record is updated as a result of central record update only if data is changed
- c = local to central message. Replication of digital objects
- d = update of BIB brief record (z0101) because of authority database record update
- e = update of Z13/Z13U(short doc) of ADM/HOL/CRS-libraries as a result of BIB-record update
- i = Update BIB hierarchy (LKR ANA and Z103 ITM) after item modification (enumeration/chronologic information)
- j = update BIB record 906$$b with monograph order’s sent date
- k = update of the BIB record STA field, as a result of the first order creation or first order's arrival. The STA values are defined in the ADM library table order_sta_update.

Col. 3-12 libraries to which message is sent

Example of the table:

```
!      1         2   3     4     5     6     7     8     9    10    11    12
!!!!!!!!!!!!!!!-!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!..!
UPDATE-DOC       4 USM01
UPDATE-SDI       8 USM01
```

tab_z105_filter

Location of the table: tab directory of the library

Purpose of the table: Fields to retain when replicating Z105 messages

Structure of the table:

Col. 1 Message type:

- 1=central to local message. Local record is updated as a result of central record update
- 2=update of central HOL record from local HOL record
- 4=update of bibliographic Z01 record because of authority database record update
- 5=update of central BIB record from local Z30 (item) record

Col. 2 Action

- Y - Retains field
- N - Does not retain field

Col. 3 Tag code

tab_z105_z30

Location of the table: tab directory of the **ADM** library

Purpose of the table: defines when a message from one library to another will **not** be created following a Z30 update.

Related table(s):

1) tab15.<lng>

Structure of the table:

Col. 1 Item status

Col. 2 Process status

Col. 3 Material type

tab_z121

Location of the table: tab directory of the **BIB** library

Purpose of the table: Definitions for the Web service search engine

This table defines the URL and search query string for the search engine in the Web service window.

Structure of the table:

col. 1 Search engine name. The search engine name must be the same as the value in the engine select box in the service-engine form (alephe/www_f_eng)

col. 2 URL and query string

Example of the table:

```
!      1                                        2
!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
EXCITE                http://search.excite.com/search.gw?search=$0100
GOOGLE                http://www.google.com/search?q=$0100&meta=lr%3D%26hl%3Den
NORTHERNLIGHT         http://www.northernlight.com/nlquery.fcg?cb=0&qr=$0100
BRITANICA             http://search.britanica.com/search?query=$0100
```

tab_z311

Location of the table: tab directory of the **ADM** library

Purpose of the table: Programs for building the Z311 table

The Z311 can now be used to store counters for z30-call-no/z30-call-no-2 and z30-inventory-number. This function is also available for items that are part of the HOL record. In order to define which kind of counter should be activated and which program can be used to create those counters tab_z311 has been implemented.

Structure of the table:

Col. 1 Item field name. Note that this is available for:

- call-no,
- z30-call-2-no,

Col. 2 Z311 Record type

- 0 - location
- 1 - inventory-number

Col. 3 Program name

Example of the table:

```
!  1                                  2     3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!
z30-call-no                         0 create_z30_location_i
z30-call-no-2                       0 create_z30_location_i
!z30-inventory-number               1 create_z30_inv_number
Z30-call-no                         0 create_z30_call_no_i_bib_090
```

tab_z403_directory.<lng>

Location of the table: alephe/tab directory **or** tab directory of the **BIB** library

Purpose of the table: Maps directories of OBJECT files between libraries

This is a table that is used by ADAM (ALEPH Digital Asset Management Module). Note that it is possible to define the file to be library specific. It can be configured in the directory ./xxx01/tab. If the file is not located under the ./xxx01/tab directory it will be loaded to $alephe_tab.

Structure of the table:

- Col. 1 Sublibrary code
- Col. 2 Usage type
- Col. 3 Description
- Col. 4 Indicates the name of the directory into which the object is to be placed.

Example of the table:

```
! 1       2                     3                                    4
!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!->
ALEPH VIEW                 Main repository (General)      $aleph_dev/demo_object
s/usm01/view
ALEPH INDEX                Index (General)                $aleph_dev/demo_object
s/usm01/index
ALEPH THUMBNAIL            Thumbnail (General)            $aleph_dev/demo_object
s/usm01/thumbnail
ALEPH BACKUP               Backup (General)               $aleph_dev/demo_object
s/usm01/backup
```

tab_z403_extension.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Maps extensions to mimetypes and icons for the Web OPAC

This is a table that is used by ADAM (ALEPH Digital Asset Management module).

It maps filename extensions of OBJECT files to mimetype and icon. It is used to define both the delivery system of an object and the icon that displays on the FULL view and the Copyrights page.

Structure of the table:

- Col. 1 Filename extension

Col. 2 Description
Col. 3 Icon
Col. 4 Open a new window – Y/N
Col. 5 Immediate Select of object – Y/N
Col. 6 File size
Col. 7 Create Thumbnail
Col. 8 Indexing
Col. 9 Viewer

Example of the table:

```
! 1       2                          3               4 5  6       7          8      9
!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!-!-!-!!!!-!!!!!!!!!!-!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!
!!!!!!!!!->
asf   asf                  f-tn-movie.jpg
avi   avi                  f-tn-movie.jpg
bmp   bmp                  f-tn-image.jpg         Y 100k convert
djv   djv                  f-tn-image.jpg
djvu  djvu                 f-tn-image.jpg
doc   doc                  f-tn-doc.jpg           Y 100k            gen
ead   ead                  f-tn-text.jpg
```

tab_z403_info.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Defines which fields from the object record (Z403) are expanded into the object info.

This is a table that is used by ADAM (ALEPH Digital Asset Management module).

Structure of the table:

Col. 1 Z403 Field Name
Col. 2 Defines if the code of the object field is to be translated. Values are:
- DATE
- SIZE

Col. 3 Description

Example of the table:

```
!    1                                    2           3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!->
z403-f-title                                        Title:^
z403-f-filename                                     Filename:^
z403-object-extension                               File Extension:^
z403-f-original-filename                            Original Filename:^
z403-object-size                        SIZE        Object Size:^
z403-note-1                                         File type:^
z403-update-date                        DATE        Last Update:^
```

tab_z701.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Determines default potential suppliers list for an outgoing IL request.

Potential suppliers can be set manually using the ILL GUI interface or automatically, based on the tab_z701 table. If you select to create the potential suppliers records using tab_z701, you must run the GUI Service 'Load Potential Suppliers Default' (ill-52) after setting up tab_z701.

Structure of the table:

- Col. 1 The requested media.
- Col. 2 The sequence of the group to which the potential supplier belongs.
- Col. 3 Whether or not the order of the potential suppliers in the group should be randomized (Y\N).
- Col. 4 The base in which the search should be made when the Locate process checks if the supplier can fulfill the request
- Col. 5 The supplier code
- Col. 6 The average number of days it takes for the supplier to supply requested material
- Col. 7 The number of days after which a request that is sent to the supplier will expire
- Col. 8 The numbr of days it takes for the library to return material to the supplier.

Example of the table:

```
!!!!!!!!!!!!!!!!!!!-!!-!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!-!!!-!!!
L-PRINTED            01 N                      ILL_LAW
L-PRINTED            01                        LAW_LT               010 003 002
L-PRINTED            02 Y                      ILL_MED
L-PRINTED            99                        LAW_LT
C-COPY               01 Y                      ILL_LAW
```

tab00.<lng>

Location of the table: tab directory of the library

Purpose of the table: Field codes and names for indexes. There should be one such table for each language defined.

Related table(s):

1) pc_tab_sear.<lng> - col. 5
2) tab11_acc
3) tab11_aut
4) tab11_ind
5) tab11_word
6) tab_filing

ALEPH allows various definitions of access paths (indexes) to document records, to suit the needs of each application. Indexes can be based on specific fields, a combination of fields, and individual words from specific document fields.

For example, the Titles table can contain titles, additional titles, uniform titles, and so on. The Authors table may contain authors, co-authors, additional authors, translators, and so on.

There are three types of indexes:

Headings (ACC) table
Examples of headings tables: authors, subjects, titles, publishers, and so on. Headings are added to the table automatically from a document field during cataloging.

Each heading entry is unique has a list of documents associated with it (that is, records in which the entry appears). A heading can be linked to an AUThority database record, which will enable the user to display references linked to the heading.

Index (IND) file
Examples of Index tables: ISBN, ISSN, call number, Library of Congress, and so on. Indexes serve as pointers to documents. Each Document field defined as Index opens an entry in the INDex table. The index entry is not necessarily unique, and the system number of the related documents sequences multiple occurrences of the same entry.

An index record is created automatically by the system from a document record field, during cataloging.

Note that the system always provides SCAN and FIND access by system number (SYS) and FIND access by barcode (BAR). Therefore, although they do not need to be defined in the indexing tables (tab11_ind or tab11_word), nor created by an indexing service, they should be defined here (in the IND section) in order to define the index name in column 11.

Words (W-) table:

Users select fields that serve as sources for entries in the Word tables. The system extracts each unique word from the specific fields of the document records, stores it in the Word table, and maintains pointers to the document in which it appears. A document will be counted as one in the count of occurrences, no matter how many times the word appears in it.

The default definition of *word* is: a character string from blank to blank, or from beginning of line to the first blank, or from last blank to the end of line.

Words are assigned to specific word *groups*. Thus, all words from the various types of title can be assigned to the words from titles group. Words from subjects can be assigned to a different words group. OPAC searches can apply to the general table or to any specific *group* table.

Users must define a tab00 line for a general Word list (WRD, W-001). Additional word groups can be added, using any code. The word group internal codes (W-nnn) must be unique, and should not jump large ranges of numbers.

Note that in the Web OPAC the filter by year option is hard coded to WYR. It is therefore advised that you use this particular code for indexing the year fields for keyword searches.

Note that the following Course library (XXX30) indexes must not be renamed: SID, CNO and CNO-S.

Structure of the table:

- Col. 1 Not in use, should always be H
- Col. 2 Heading code. Code of heading/word/index file. Can be up to 5 characters.
- Col. 3 Index type. Use ACC for headings, IND for direct indexes and W-nnn for word indexes. For word indexes, each index **must** have a unique W- number; you must start counting from W-001, and you should not jump large ranges of numbers;
- Col. 4 Not implemented
- Col. 5 Filing procedure (specified in tab_filing)
- Col. 6 Not implemented
- Col. 7 Not implemented
- Col. 8 Not implemented
- Col. 9 Location weight (last 2 digits in the W section are for weighting as used in the Rank command of the Web OPAC)
- Col. 10 Not in use, leave blank
- Col. 11 Name of heading/word/index file

Example of the table:

```
!   2     3   4 5  6  7 8   9   10            11
!-!!!!!-!!!!!-!-!!-!!-!-!---!!-!!!!!-!!!!!!!!!!!!!!!!!!
H SYS   IND     21 00       00       System Number
H BAR   IND     21 00       00       Barcode
H 010   IND     22 00       00       LCCN
H ISBN  IND     21 00       00       ISBN
H ISSN  IND     21 00       00       ISSN
…
H TIT   ACC     11 00       00       Titles
H TITC  ACC     61 00       00       Titles Chinese
H TITJ  ACC     62 00       00       Titles Japanese
H TITK  ACC     63 00       00       Titles Korean
H TRT   ACC     11 00       00       Titles/Related Title
H AUT   ACC     01 00       00       Authors
…
H WRD   W-001     00        01       Words
H WTI   W-002     00        02       W-titles
H WAU   W-003     00        02       W-authors
H WSU   W-004     00        03       W-subjects
H WUT   W-005     00        02       W-Unif .Titles
```

tab01.<lng>

Location of the table: tab directory of the library

Purpose of the table: List of tags for records

Related table(s):

1) tab00.<lng>
2) tab02
3) tab_filing

The tab01 table defines the tags in the library's record. It should be repeated, with language extension, for every interface language set up for **ALEPH**.
The table is used for:
defining which heading list (Z01) is used for the update of a full record
display formats 001 and 002 in OPAC
record from authority control
defining the filing procedure for sort keys based on a field (for example, Z101 table, p-ret-21 (sort of retrieved BIB records))
defining the non-filing indicator for the "non_filing" filing procedure
setting the order of fields for the sort_docx procedure

Structure of the table:

Col. 1 D

Col. 2 Internal codes or tags to be used for cataloging. Tags as defined in this table can be added, deleted, and changed as necessary. Each field code (tag) must be unique in the D table. Changes to

field codes (tags) are reflected in any subsequent additions of records to the database, but do not affect existing records.

Col. 3 Filing procedure for sorting records as defined in (tab_filing). Used when building filing key for sort keys (Z101) and for the batch services p-ret-21 (sorts retrieved docs) and p-manage-27 (update sort index). If changes are made to this column, p-manage-27 should be rerun.

If nothing is entered defaults to 99.

Col. 4 Do not use (always 00)

Col. 5 Do not use (always 0000)

Col. 6 Non-filing indicator- 1/2

Indicates which indicator to look at for non-filing characters.

Only used by p-ret-01 and p-ret-21 (col. 9 – (non filing indicator). tab11_acc is used for defining non-filing elements in indexed headings.)

Col. 7 The code of the headings list that is used for automatic correction of the document record field. If the heading which matches the document field is a "see" cross reference, and the heading is set to "UPDATE Y", the field will be updated to the correct form.

Col. 8 Input/output code. Repeat definitions in col. 2.

Col. 9 Field name. Used in format 002 for display and printing. Always prefix the text with "L".

Example of the table:

```
!   2   3  4   5   6   7     8      9
!-!!!!!-!!-!!-!!!!-!-!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!
D FMT       00 0000          FMT   LFormat
D LDR       00 0000          LDR   LLeader
D 001       00 0000          001   LControl No.
D 003       00 0000          003   LControl No. ID
D 005       00 0000          005   LDate & Time Last Tr
D 006       00 0000          006   LFixed field Add MatL
D 007       00 0000          007   LPhysical Descr. FF
D 008       00 0000          008   LFixed-Length Field
D 010       00 0000          010   LLC Control No.
D 013       00 0000          013   LPatent Ctrl Info
D 015       00 0000          015   LNational Bib. No.
D 016       00 0000          016   LNat.Bib.Agcy.Ctrl.No.
…
D 100   01  00 0000    AUT   100   LME-Personal Name
D 110   01  00 0000    AUT   110   LME-Corporate Name
D 111   01  00 0000    AUT   111   LME-Meeting Name
D 130   11  00 0000 1  TIT   130   LME-Uniform Heading
D 210       00 0000          210   LAbbreviated Title
D 222       00 0000 2        222   LKey Title
D 240       00 0000 2        240   LUniform Title
D 242       00 0000 2        242   LTitle Trans.by CatAgency
D 243       00 0000 2        243   LColl. Uniform Title
D 245   11  00 0000 2  TIT   245   LTitle
```

tab02

Location of the table: tab directory of the library

Purpose of the table: Text to use for setting non-filing indicator value

Related table(s):

1) tab_fix

This table defines initial text for the *fix_doc_non_filing_ind* routine. The routine is used for setting the value of the non-filing indicator for the relevant fields, by checking the initial text of a field.
It is activated in the Cataloging module via Fix Record, and the tab_fix routines (which are defined in the tab_fix table).
If, for example, the cataloger enters:

245 a The book of the dead

The *tab_fix* routine will add as the 2nd indicator the number 4 (suppression of the word *the* plus one space).

245 4 a The book of the dead

The *tab_fix* routine is dependent on two elements:
definition of which indicator to use for stopwords in indexing (defined in tab01.<lng>)
presence of a language code in fixed field 008 pos. 35-37

Structure of the table:

col. 1 alpha

col. 2 language code

col. 3 word to be ignored - Initial text to count for setting indicator value

Example of the table:

```
L eng the
L eng a
L eng an
L fre el
L fre il
L fre la
L spa lo
L spa un
L ger das
L ger Der
L ger die
L spa gli
```

tab04

Location of the table: tab directory of the library

Purpose of the table**:** Conversion of field codes between formats

Related table(s):

1) tab_fix
2) tab_expand
3) tab01.<lng>

Tab04 allows for converting one set of cataloging tags to another. It is used for tab_fix and for expand_doc_bib_tab04.

tab_fix

Different conversion routines can be defined and linked to the corresponding fix program in the tab_fix table. A conversion set can be used when importing records from databases with different cataloging systems (for example, from a UNIMARC type of library to a MARC 21 library).

expand_doc_bib_tab04

This expand is primarily intended for the Z39_SERVER instance in tab_expand, in order to translate alphabetic tags to numeric values (for example, LOC to 952), because the Z39 protocol does not recognize non-numeric tags.

Note that tags not defined in this table will be deleted from the record when activating the tab_fix or expand_doc routine.

The maximum number of lines is 5000.

Structure of the table:

Col. 1 Fix procedure.

Col. 2 Document field code of library from which record is copied. Use # for none or one character default field code of source library.

Col. 3 subfield codes of the field of the library from which record is copied (up to 10, or blank for entire field)

Col. 4 document field code of local library to which record is copied

Col. 5 indication to strip subfield codes when copying

- Y=strip,
- N=do not strip

Col. 6 subfield codes to be used instead of original ones

Col. 7 text to be appended to contents of field

In example below the table is defined to translate UNIMARC into MARC 21

Example of the table:

```
!    2       3        4   5   6          7
!!-!!!!!-!!!!!!!!!!-!!!!!-!-!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
01 FMT              FMT   N
01 LDR              LDR   N
01 001              001   N
01 005              005   N
01 010## abdz       020## N abcz
01 011##            022## N
01 012##            024## N
01 020##            015## N
01 021##            017## N
01 022## bza        086## N az2
01 040##            030## N
01 100##            008## N
```

tab05.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: Captions for links between records

Related table(s):

1) tab01.<lng>

This table defines captions for the links between records, when the links are defined using the LKR field with subfield 'r'.
In the LKR field, the MARC tag that defines the reason for linking two records can be registered in subfield 'r'. Using the content of subfield 'r', tab05 defines the caption to display in the OPAC before $$n (up link) and $$m (down link).

This is a language-sensitive table.

Structure of the table:

Col. 1 MARC tag – can be indicator sensitive

Col. 2 Caption for $$n (up link)

Col. 3 Caption for $$m (down link)

Note that the 'tag' NONE should be defined for linking fields where no $$r has been defined.

Example of the table:

```
! 1                        2                                3
!!!!!------!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
NONE       Click to view related record
760        Main series:                    Subseries of:
762        Has subseries:                   Main series:
765        Translation of:                  Translated as:
…
775        Other edition available:         Other edition available:
776        Available in other form:         Available in other form:
777        Issued with:                     Issued with:
78000      Continues                        Continued by
78001      Continues in part                Continued in part by
78002      Supersedes                       Superseded by
…
```

tab06

Location of the table: tab directory of the **BIB** library

Purpose of the table: Defines non-high lighted words for full record display in the Web OPAC .

Related table(s):

1) edit_doc_999.<lng>

Words listed in this table will not be highlighted in the full doc (edit_doc_999.<lng>) display in the Web OPAC.

Structure of the table:

Col. 1 ALPHA

Col. 2 Word to be ignored

Example of table:

```
!  2
!-!!!!!!!!!!!!!!!!!!
L of
L the
L a
L an
```

tab07

Location of the table: tab directory of the **BIB** library

Purpose of the table: Defines the links between Authority Headings

Related table(s):

1) tab00.<lng>
2) tab01.<lng>

This table defines the hierarchical links between Authority Headings for building Z103 link records. It is valid for an AUT type of library only.

NOTE: tab07 is limited to 40 lines.

Building z103 links between AUT records is ***required*** for multilingual AUT libraries. When cataloging in a multilingual AUT library it suffices to add 5XX references in one language only. Using this table, the system builds Z103 links, to "add" the 5XX in all its language manifestations. The additional 5XX fields are not actually present in the AUT record, but when the record is displayed, they are included (through the Z103 links).

Libraries can ***opt*** to use tab07 and Z103 links in order to build reciprocal 5XX references. According to table setup, if there is a 5XX in one record, the Z103 can create the reciprocal reference in the other record. In this way, if "broader term" references are entered, the "narrower term" references will be automatically generated.

For example:
LCSH authority records include 550 fields for Broader terms only ($$wg).
If the library wants to show references in the opposite direction (to Narrower terms), this can be done using this table.

Structure of the table:

Col. 1 field
Col. 2 subfield to strip or include when matching
Col. 3 access list used for searching for the heading
Col. 4 subfield which contains link type
Col. 5 link type code
Col. 6 type of ALEPH link to build
Col. 7 reciprocal link

Example of the table:

```
! 1        2        3   4     5      6   7
!!!!!-!!!!!!!!!!-!!!!!-!-!!!!!!!!!!-!!--!!

550## axyz568    GEN   w a          ET  LT
550## axyz568    GEN   w b          LT  ET
551## axy568     GEN   w h          NT  BT
551## axy568     GEN   w g          BT  NT
500## adqt568    GEN   w f          MC  MC
```

tab11_acc

Location of the table: tab directory of the library

Purpose of the table: Assignment of fields to headings indexes

Related table(s):

1) tab00.<lng>
2) tab01.<lng>
3) tab_filing
4) formats.<lng>

The tab11_acc table is used to assign fields to headings indexes.

The table is limited to 1500 lines.

Structure of the table

col. 1 Field tag. # (wildcard) can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 245## for 2451, 2452, 24501).

col. 2 TYP or FMT code. This column can be used to index fields by the format of the record. The program first looks for a match on the TYP field (created by the expand_doc_type program) and then for a match on the FMT field. For example, if the column contains CF then the system will look through all TYP and FMT fields to see if there is a match on CF. If there is a match then the field will be indexed.

col. 3 Subfield to filter. It is used together with column 4. For example

if this column contains a **2** and column 4 contains **usm50**, then only records with **usm50** in subfield 2 will have this field indexed.

col. 4 Subfield contents filter. It is used together with column 3. For example:

if column 3 contains a **2** and this column contains **usm50,** then only records with **usm50** in subfield 2 will have this field indexed.

If the contents is prefixed by a minus (-), the content is considered negative.

For example, if column 3 contains a **2** and this column contains **-usm50**, then only records that **do not have usm50** in subfield 2 will have this field indexed.

NOTE!! filter text must be entered in lowercase (irrespective of case in the record itself). Use ***** to indicate truncated text; use **#** to indicate single wildcard for text match

col. 5 Code of the headings index file as defined in tab00.<lng>

col. 6 Subfield(s) of the field for indexing. Blank indicates the entire field. Use minus (-) sign followed by subfields to define the subfields to be stripped or list the subfields to be indexed.

col. 7 Group. Values are: A, B or C. This column may be used to define a particular group of fields that will be used to create a ***new*** headings index. This option cannot be used for an index that already exists. Note that this option is available only when the program is run from the Unix command line. It is not available from the Headings index batch service.

The following is an example of the way in which the program should be run for fields that belong to group B:

from ../aleph/proc

csh -f p_manage_02
USM01,0,000000000,999999999,B,1,N,00,

col. 8 Non-filing indicator for non_filing procedure as defined in the tab_filing table.

Example of the table:

```
! 1    2    3     4         5             6            7 8
!!!!!-!!!!!-!-!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!-!-!
050##                    LCC   a
090##                    LCC   a
060##                    NLM   a
082##                    DDC   a
086##                    SUD   a
086##                    SUD   z
100##                    AUT   -e468
100##       9 chi        AUTC  -e468
100##       9 jpn        AUTJ  -e468
100##       9 kor        AUTK  -e468
AU100                    AWT
AT100                    AWT
110##                    AUT   -e468
110##       9 chi        AUTC  -e468
110##       9 jpn        AUTJ  -e468
110##       9 kor        AUTK  -e468…
```

tab11_aut

Location of the table: tab directory of the **BIB** library

Purpose of the table: Link back from the AUT record

Related table(s):

1) tab00.<lng> in the BIB library
2) tab01.<lng> in the AUT library

The tab11_aut table is used to defines the headings files that the system uses to create hypertext links to FIND and BROWSE from the authority record. This allows the user to navigate the bibliographic database using the authority record fields.

Structure of the table:

col. 1	Authority record field. Note that if an authority term is linked to more than one headings file (personal names, 100, can be used for both authors and subjects), several lines can be created for the field. The system runs the search from the current headings file. # can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 100# for 1000, 1001, 1003).
col 2	Not in use
col 3	Not in use
col 4	Bibliographic headings file on which Browse/Find is activated
col 5	Subfields to include/strip from authority record field for Search. The system looks for the exact text of the field in the headings file. Blank indicates entire field. Use the minus (-)sign followed by subfields to define subfields to be stripped or list the subfields to be indexed.
col 6	Not in use
col 7	Not in use

Example of the table:

```
! 1   2     3        4           5            6 7
!!!!!-!-!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!-!-!
100##              AUT
100##              AWT
100##              SUB
100##              SUL
100##              TIT
110##              AUT
110##              AWT
110##              SUB
…
150##              SUB
150##              SUL
150##              SUM
150##              MLT
151##              SUB
151##              SUL
151##              SUM
151##              MLT
400##              AUT   -wi5
```

tab11_ind

Location of the table: tab directory of the library

Purpose of the table: Assignment of fields to direct indexes

Related table(s):

1) tab00.<lng>
2) tab01.<lng>
3) tab_filing

The tab11 table is used to assign fields to Direct indexes. Direct indexes enable the user to retrieve a specific record. A direct index is suited to unique or almost unique identifiers of the record, and as such serves as a quick access to a record.

Direct indexes are also used by record loader and union view tables, in order to find possible matching records (candidates).

The table is limited to 500 lines.

Structure of the table

col. 1 Field tag. # (wildcard) can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 245## for 2451, 2452, 24501).

col. 2 Not in use.

col. 3 Subfield to filter. It is used together with column 4. For example if this column contains a **2** and column 4 contains **usm50**, then

only records with **usm50** in subfield 2 will have this field indexed.

col. 4 Subfield contents filter. It is used together with column 3. For example:

if column 3 contains a **2** and this column contains **usm50,** then only records with **usm50** in subfield 2 will have this field indexed.

If the contents is prefixed by a minus (-), the content is considered negative.

For example, if column 3 contains a **2** and this column contains **-usm50**, then only records that **do not have usm50** in subfield 2 will have this field indexed.

NOTE!! filter text must be entered in lowercase (irrespective of case in the record itself). Use ***** to indicate truncated text; use **#** to indicate single wildcard for text match

col. 5 Code of the headings index file as defined in tab00.<lng>

col. 6 Subfield(s) of the field for indexing. Blank indicates the entire field. Use minus (-) sign followed by subfields to define the subfields to be stripped or list the subfields to be indexed.

col. 7 Not in use

col. 8 Non-filing indicator for non_filing procedure as defined in the tab_filing table.

Example of the table:

```
! 1    2    3     4         5            6           7 8
!!!!!-!!!!!-!-!!!!!!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!-!-!
010##                     010   -bz
050##                     TST
010##                     STIDN -bz
020                       ISBN  az
020                       STIDN az
022                       ISSN  ayz
022                       STIDN ayz
028##                     028   -b6
028##                     STIDN -b6
0242#                     024   -26
…
```

tab11_word

Location of the table: tab directory of the library

Purpose of the table: Assignment of fields to word indexes

Related table(s):

1) tab00.<lng>
2) tab01.<lng>

3) tab_word_breaking

The tab11_word table is used to assign fields to word indexes. Note that for the "general words file" the library should use W-001 WRD, and there should not be large gaps in the numbers of the specific word groups.

The table is limited to 10000 lines.

Structure of the table:

Col. 1 Field tag. Document record field code. # can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 245## for 2451, 2452, 24501)

Col. 2 TYP or FMT code. This column can be used to index fields by the format of the record. The program first looks for a match on the TYP field (created by the expand_doc_type program) and then for a match on the FMT field. For example, if the column contains CF then the system will look through all TYP and FMT fields to see if there is a match on CF. If there is a match then the field will be indexed.

Col. 3 Subfield filter, used together with column 4. For example, if this column contains a '2' and column 4 contains 'usm50', then only records with usm50 in subfield 2 will have this field indexed.

Col. 4 Subfield contents filter. It is used together with column 3. For example, if column 3 contains a '2' and this column contains 'usm50', then only fields with usm50 in subfield 2 will be indexed. Note the following:

- the filter text must be entered in lowercase (irrespective of case in the record itself)
- the filter text must not include special characters (only 7 bit Latin)
- the filter text does not undergo filing (only unicode_case) and for this reason it should contain the exact string
- use * to indicate truncated text; use # to indicate a single wildcard for text match;

Col. 5 Subfield(s) of the field for indexing:

- subfield(s) of the field (blank indicates entire field).
  OR
- minus (-) sign followed by subfields to be stripped
  OR
- **F** followed by fixed field position (counted from base 00, taking subfield code into account),

followed by hyphen (-) and the
no. of characters.

Col. 6 Word building procedure - see tab_word_breaking for more details

Col. 7-8 Not in use

Col. 9-18 Word index code as defined in tab00.<lng>. Up to 10 codes can be assigned

Note the following:

When word proximity (and adjacency) is taken into account, the text string being indexed is used. If the text string is a selection of subfields, this will cause some problems with adjacency.

For example, if you are indexing field 246, which might have subfields a, b, c, d and so on., but you have defined indexing as:

```
11 W 246##              abnp                  03      WTI
```

This leaves out subfields c,d,e,f,g,h,i; and the last word of $$b will be considered adjacent to the first word of $$n.

There might be instances where this is desirable. But, if you do not want this to happen, you should set indexing to separate strings, for example,

```
11 W 246##              ab                    03      WTI
11 W 246##              np                    03      WTI
```

Example of the table:

```
! 1     2   3      4            5             6  7 8  9   10   11   12
!!!!!-!!!!!-!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!-!-!-!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!
008                     F07-04               01 A   WRD  WYR
008                     F35-03               01 A   WRD  WLN
SBL                                          01     WSL
SYS                                          90     SYS
LOC##                   -o                   03     WRD  WLC
041##                   abdefg               41 A   WRD  WLN
043##                                        01     WRD  WGA
100##                   -6                   01     WRD  WAU
110##                   -6                   01     WRD  WAU
111##                   -6                   01     WRD  WAU  WTI
130##                   -6                   03     WRD  WTI  WUT
210##                   a                    03     WRD  WTI
222##                   a                    03     WRD  WTI
240##                   -6                   03     WRD  WTI  WUT
242##                   abnp                 03     WRD
243##                   -6                   03     WRD  WTI
245##                   c                    03     WAU
245##                   -c6                  03     WTI
```

tab12

Location of the table: tab directory of the **BIB** library

Purpose of the table: Translation of ALEPH codes to Dublin Core XML elements

This table defines the translation of ALEPH fields and subfields to unqualified Dublin Core XML elements.

The table is used by z39_server to return XML records according to Bath profile definition.

Note: different subfield combinatations of the same field can be translated to different DC elements. For example:

260 b publisher
260 c date

Structure of the table:

col. 1 ALEPH field code

col. 2 Subfields list:

1) subfield(s) of the field (blank indicates the entire field).
OR

2) minus (-) sign followed by subfields to be stripped

OR

3) F followed by fixed field position (counted from base 00,

taking the subfield code into account), followed by hyphen (-) and the no. of characters;

col. 3 DC element. Possible values:

title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coverage, rights.

Example of the table:

```
!1               2                          3
!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
008 F35-03                         language
020 a                              identifier
022 a                              identifier
050 ab                             subject
090 ab                             subject
610 ab                             subject
852 hi                             subject
653 a                              subject
600 aqd                            subject
611 acd                            subject
```

tab14

Location of the table: tab directory of the **ADM** library

Purpose of the table: Item reshelving time

Related table(s):

1) tab_sub_library.<lng>
2) tab40.<lng>
3) tab15.<lng>

Tab14 allows the library to define the time it takes for a returned item to get back on the shelf. The definition can be on the level of the branch library, collection and item status. When an item is returned, the *due date column* will display *reshelving* for the time defined in this table.

Structure of the table:

col. 1 sublibrary (use ##### to signify all sublibraries)

col. 2 collection (use ##### to signify all collections)

col. 3 item status (use ## to denote all items)

col. 4 item process status (use ## to denote all items)

col. 5 reshelving time (HHMM)

Example of the table:

```
! 1     2   3  4   5
!!!!!-!!!!!-!!-!!-!!!!

UARCV RARE  ## ## 0015
UARCV ##### ## ## 0030
!*
UELEC ##### ## ## 0045
!*
UGDOC ##### ## ## 0030
!*
UUPTN ##### ## ## 0130
```

tab15.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines the item statuses and various parameters connected to these statuses.

Related table(s):

1) tab_sub_library.<lng>
2) tab_hold_request

tab_patron_log.<lng >

Location of the table: tab directory of the **USR** library

Purpose of the table: List the logs for patron record changes

tab_patron_log.<lng> is used for setting-up the action codes (column 1) that trigger Z307 Patron Log records and their textual descriptions (column 3). This table is located in the tab directory of $usr_library. Only the activated actions (column 2 = Y) are triggered. Use column 2 to have Aleph ignore action triggers.

The following is a list of possible logs (numeric code–col.1 and text decription-col.3):

001 - Update of global patron record-Details
002 - Update of global patron record-Blocks
003 - Update of local patron record-Details
004 - Update of local patron record-Blocks
005 - Update of local patron record-Privileges
006 - Update of patron addresses
007 - Update of patron IDs
010 - Delete of global patron record
011 - Delete of local patron record
012 - Delete of patron address record
013 - Delete of patron ID
020 - New global patron record created
021 - New local patron record created
022 - New patron address record created
023 - New patron ID created

Structure of the table:

COL 1. The action type code that cause a Z307 patron change log record to be written

COL 2. Active/Non active log: Y/N

COL 3. Description of the action

Example of the table:

```
!1  2           3
!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!->
001 Y Update of global patron record-Details
002 Y Update of global patron record-Blocks
003 Y Update of local patron record-Details
004 Y Update of local patron record-Blocks
005 Y Update of local patron record-Privileges
006 Y Update of patron addresses
007 Y Update of patron IDs

010 Y Delete of global patron record
011 Y Delete of local patron record
012 Y Delete of patron address record
013 Y Delete of patron ID

020 Y New global patron record created
021 Y New local patron record created
022 Y New patron address record created
023 Y New patron ID created
```

3) tab_photo_request
4) tab16 – cols. 19 and 20
5) www_server.conf - setenv www_z37_recall_type

Structure of the table:

Col. 1 Group ID for group of definitions as defined in the tab_sub_library.<lng>. Note that there is a limitation of 1000 lines for the drop-down list. #### (wildcard) ***cannot*** be used in this column.

col. 2 item status. ## is used to denote all item statuses

col. 3 process status - ## is used to denote all process statuses

col. 4 L

col. 5 description

col. 6 can be loaned Y/N

col. 7 can be renewed Y/N

col. 8 Availability check for hold request:

- **Y** - item can be requested.. In addition, item availability is checked when hold request is placed. If a copy is available, and patron does not have permission

for requesting available item, message displays.

- **N** – item cannot be requested
- **C** - item availability is not checked when hold request is placed, because hold requests are always allowed (for example, closed stack).
- **T** - item availability is not checked when hold request is placed, because hold requests are always allowed (for example, short-term loan, lost, on order, and so on.). However, for p-cir-12 (print request slips) and ue_06 (online printing of request slips), item is not considered "requestable" and does not print.
- **R** - Rush Cataloging item. When a privileged patron requests the item, a cataloging trigger is created, prompting the Cataloging department to prioritize the cataloging process. Item availability issues are managed just like T, except that if there are available like items that are not "R" the request is blocked.
- **U** - Unbarcoded items. When a privileged patron requests the item, a new temporary item will be created. The request is placed on the new temporary item.

col. 9 Availability check for photo requests;

- **Y** – Photocopy request can be placed
- **N** - photocopy request cannot be placed for the item.
- **T** - item availability is not checked when hold request is placed, because photocopy requests are always allowed (for example, short-term loan, lost, on order, and so on.). However, for p-cir-22 (print photocopy request slips) and ue_06 (online printing of request slips), item is not

considered "requestable" and
does not print.

In general, note that apart from the availability check in tab15.<lng>, the checks that are made when a photocopy request is made is controlled by tab_patron_log.<lng>

col. 10 Display item in Web OPAC Y/N. Even if set to N – will always display in GUI Search functions.

col. 11 if set to Y, the request will be for the specific item, not another like copy, regardless of whether the request is initiated from the Web OPAC or from the GUI.

col. 12 limit hold requests to the number of available copies - there can only be a single request per item (Y/N).

col. 13 allow recall of item (Y/N). Note that tab16 cols.19 and 20 define the no. of days for recall notice and the minimum guaranteed loan period. Recall notices are generated by the Recall Notices service (p-cir-13)

col. 14 allow rush recall of item (Y/N).

col. 15 Reloaning limit: The value denotes the number of hours during which a patron is not allowed to reloan the same copy of an item, The count is from the time he returned it. The values "00" or " " (blanks) should be used for unrestricted reloaning.

col. 16 Booking permission

- Y - Booking is possible
- N - Booking is not possible
- O - Booking is possible only during open hours
- C - Booking is conditional on the patron's local booking permission (Z305-BOOKING-PERMISSION)

col. 17 Booking hours.

- A - Booking is possible at all times (both open and closed hours)
- O - Booking is possible only during open hours
- C - Booking is possible only during closed hours.

With regard to cols. 13 and 14 – recall/rush recall: note the variable in www_server.conf setenv www_z37_recall_type, which defines the default recall type when placing a hold request in the Web OPAC.
(The recall type is 01, 02 or 03, where 01 is regular, 02 is rush and 03 is no recall.)

Note: Both the requests daemon ue_06 (UTIL E/6) and the batch process p_cir_12 (Call Slips for Hold Requests) ignore the value of column 6 of tab15.<lng> (item can

be loaned Y/N). That is, the decision whether to process a hold request or not does not depend on the value of that column.

Example of the table:

```
! 1   2  3  4        5                       6 7 8 9 1 1 1 1 1 1
!                                               0 1 2 3 4 5
!!!!!-!!-!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!-!-!-!-!-!-!-!-!-!!

15A   ## OR L On Order                        Y N C N Y N N N N
15A   ## OI L OrderInitiation                 N N N N Y N N N N
15A   ## CL L Claimed                         N N Y N Y N N N N
15A   ## BP L Binding preparation             N N N N Y N N N N
15A   ## BD L Bound                           N N N N Y N N N N
15A   ## IP L In process                      N N C N Y N N N N
15A   ## SC L Send to Cataloging              Y Y Y N Y N N N N
15A   ## CT L Cataloging                      N N N N Y N N N N
15A   ## TE L Technical Department           N N N N Y N N N N
…
15A   10 ## L One Hour Loan                   Y Y Y Y Y N N Y Y
15A   11 ## L One Day Loan                    Y Y Y N Y Y N Y Y
15A   12 ## L One Week Loan                   Y Y Y Y Y Y N Y Y
15A   22 ## L Two Week Loan                   Y Y Y Y Y N N Y Y
15A   31 ## L Three Day Loan                  Y Y Y Y Y N N Y Y
15A   32 ## L Three Week Loan                 Y Y Y Y Y N N Y Y
```

tab16

Location of the table: tab directory of the **ADM** library

Purpose of the table: Due dates, fines & limits

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31
4) aleph_start - fine_factor_fix, fine_rounding
5) tab_late_return
6) tab_check_circ

Special note: When a change is made to tab16, in order for the changes to take immediate effect - the due dates table, (Z301) should be dropped and recreated (UTIL A/17/1) Note, however, that if the table is not dropped and recreated in this way, the changes automatically take effect the next day.

This table allows for various definitions per patron/item status – for for example,
computation of due dates
grace period
fine rate
maximum number of items that can be loaned to a patron
maximum number of hold requests

fine method, and so on.
The computation of the due dates for each combination of copy/patron status is performed by the system daily, according to the rules defined in tab16. The result of the computation displays in the *Due Dates* option in the Circulation GUI. The operator can change the dates if necessary.

Fine rates.
The library can opt to set fines for selected statuses only (for example, professors do not have a fee rate, students with patron status nn have a higher fee rate than students with patron status xx.)
Note the switch in tab100, (ZERO-FINE-HANDLING) which defines whether late returns should be registered in the Cash Management record, even if there is no fine for the late return.

Note:
The groups in tab16 (16A, 16B, and so on.) serve only to reduce the size of the table when there is a library policy which is the same for more than one sublibrary. The group itself has **no** functionality as a group. All checks are made on the sublibrary level. All checks of limits are for the items of the same sublibrary as the item in hand. For example, If there is a 99 line for "total limit", then this is the **total limit for each of the sublibraries individually,** and not for the sublibraries as a group.

The maximum number of lines per sublibrary or group is 500.

Item status 99 and 98
Defines limits for the total number of loans/holds/bookings. There is no item status check.
You can use ## in col.4 (patron status) to denote all patron statuses or enter a specific patron status in col.4 (patron status) if you want to define the global loan/hold limit per patron status.
Where 99 the total check is for every sublibrary that relates to the group ID.
Where 98 is used the total check is for the entire group that relates to the group ID.
The limit that is set in the 98 line means that the total number of loans/holds/bookings in the entire group may not exceed this limit.
It is possible to define a total check for the entire ADM library.
Define a separate line with a different group ID for the ADM library. Define a separate line with a different group ID for the ADM library and use the 99 item status.
Enter this group ID in column 7 in tab_sub_library.<lng> in the line that defines the ADM library.
This must be the LAST line in every group ID.

Hours are rounded to a single hour, days are rounded to a single day. The default is 2.

If no parameters are defined for cols. 15-18, then the regular date/hour parameters are used (cols. 5-6, 8-9)

Structure of the table:

col. 1	Group ID for group of definitions as defined in tab_sub_library.<lng>
col. 2	Item status. 2 numeric digits. (## denotes any/all status(es).

col. 3 Process status (## for all)

Note: The item process status is not taken into consideration when creating due dates and due hours (cols. 6, 9, 16 and 18) but is taken into consideration for all other columns. Due dates (Z301) will only be built from tab16 lines which have "##" in the item process status column.

col. 4 Patron status. ## denotes all patron statuses. Enter a specific patron status if you want to define the global loan/hold limit per patron status.

col. 5 Date operator for interpretation of the next field:

- **A** The date in col. 6 is an absolute date to be assigned as is.
- **+** The date in col. 6 is a relative time to be added to the current date.

col. 6 Date parameter - DDDDDDDD or YYYYMMDD

This date is treated according to the operator in col. 5:

If **+** in col. 5 then no. of days to add. Only the last **three** digits are relevant – for example, if you enter **1950** days, the system will compute **950** days. Therefore, the maximum number of days that can be defined is 999

If **A** in col. 5 then actual date – for example, 20101231.

col. 7 Grace days. – that is, number of days within which late return is "forgiven". Note however, that if an item is returned after the grace period, the fine rate will be for the total number of overdue days/hours, taking the grace period into account.

col. 8 Hour operator for interpretation of the next field:

- **A** - The hour in col. 9 is the absolute hour to be assigned as is.
- **+** - The hour in col. 9 is the relative hour to be added to the current hour. This is especially useful for short term loans.

NOTE: If there is a + in this column the dates field (col. 6) should be zeroes.

col. 9 HHMM

Hour and minutes for due date. This date is treated according to the flag in col. 8

col. 10 Grace time - that is, the hours and minutes within which a late return is “forgiven”.

col. 11 Fine multiple, that is, the factor by which the number of days overdue is multiplied to compute the fee owing for items returned late or overdue (3 digits + 2 decimal).

NOTE: the sum refers to a day. In order for it to refer to an hour, multiply the amount for an hour by 24 and register the resulting amount in this column.

Note that in /alephe/aleph_start it is possible to set the

multiplication factor and rounding (Y/N).

Example of fine definition in /alephe/aleph_start (UTIL Y/14):

```
Setenv fine_factor_fix          000
Setenv fine_rounding            N
```

col. 12 Maximum number of items for patron/item matrix – 3 digits

Remember that item status 99 defines total number of loans for defined patron status (no item status check). It is not possible to set to unlimited number of loans. If you have a patron status that you want to set up to allow an unlimited number of loans, set “check loan" (col.7) in tab31 to N (z305_loan_check=N). (for for example, for institutional patrons, like ILL departments.)

col. 13 Maximum number of hold request – 3 digits.

The maximum number of hold requests a patron is allowed to have of a specific item status or of all item statuses when relating to the 99 line

col. 14 fine method

- 0 - no fines charged
- A - open minutes
- 1 - open hours
- 2 open days
- B - total minutes
- 3 - total hours
- 4 - total days
- 5 - overlapping block date
- 6 - cumulative block date
- C - include recall - behaves like type 1 (open-hours)
- D - include recall - behaves like type 2 (open-days)
- E - include recall - behaves like type 3 (total-hours)
- 7 - include recall - behaves like type 4 (total days)
- R - fixed 1st day - behaves like type 2 (open-days)
- F - total days by intervals (using tab_late_return)
- 8 - overlapping block date +fine calculated according to total days
- 9 - cumulative block date +fine calculated according to total days
- Y - special for Denmark
- Z - special for Denmark

Note the following:

Fine method 5 - overlapping block date

A patron who returns one item late is blocked from borrowing items or renewing loans (this depends on the setup of tab_check_circ) for a period equal to the number of days the item was overdue.

A patron who returns two or more items late is blocked from borrowing items or renewing loans for a period equal to the highest number of days an item was overdue.

For example, a patron who returns two items late, one 5 days overdue and the other 10 days overdue is blocked from borrowing items for 10 days.

Fine method 6 - cumulative block date

A patron who returns one item late is blocked from borrowing items or renewing loans (this depends on the setup of tab_check_circ) for a period equal to the number of days the item was overdue.

A patron who returns two or more items late is blocked from borrowing items or renewing loans for a period equal to the sum of the number of days each item was overdue.

For example, a patron who returns two items late, one 5 days overdue and the other 10 days overdue is blocked from borrowing items for 15 days.

Fine method R - First day fixed, then open days

The fine for the first overdue day is a fixed sum of 2.00 (hardcoded). No other fines are added for the first day. From the second day onward the calculation is like fine method 2 (open days).

NOTE: hours are rounded to a single hour; days are rounded to a single day. The default set by the system is fine method 2.

col. 15 Date operator for loan of items with holds (+ or A as in col. 5)

col. 16 Date parameter for loan of items with holds. The value entered here is either an offset or an actual date depending on the value of column 15

col. 17 Hour operator for loan of items with holds (+ or A)

col. 18 Hour parameter for loan of items with holds. The value entered here is either an offset or an actual hour depending on the value of column 17.

NOTE: if no parameters are defined for cols. 15-18, then the regular date/hour parameters are used (cols. 5-6, 8-9)

col. 19 Number of days from recall notice to new due date for recalled item

col. 20 Number of days for minimum guaranteed loan period (when recalled)

col. 21 Maximum no of renewals. Values: 00-09 or blanks. Renewals can be limited up to 9; leave blank to define that renewals are unlimited.

To limit renewals by a time period rather than by a number, leave this column blank, and define the time period in column 22.

col. 22 Maximum renewal period

D days/W weeks/M months
nnn - number of units

example:

D007 - renew up to a maximum of 7 days

W004 - renew up to 4 weeks

Col. 23. Maximum fine (up to 2 decimals); the field should be left-aligned

Col . 24 Minimum fine. Fines below this amount will be treated as zero amount fines. The field should be left-aligned

Col. 25 Fixed Fine Addition. Fixed amount to be added to any fine. The field should be left-aligned.

Col 26 The maximum number of booking requests a patron is allowed to have. 999 to indicate unlimited number of requests.

Col. 27 Adjust Due Date and Hour. column is sensitive to sublibrary and item status only. Therefore, use '##' for both item process status and patron status.

When calculating the due date:

0 - Adjust a due hour which is after closing time to the closing time (if the due date is an open day), or to the closing time of the NEXT open day (if the due date is a closed day).

1 - Adjust a due hour which is after closing time to the closing time (if the due date is an open day), or to the closing time of the PREVIOUS open day (if the due date is a closed day).

2 - Do NOT adjust due hour if the due date is an open day; if the due date is a closed day, find the NEXT open day. This is the default value in case tab16 cannot be read or this column has an invalid value (that is, other than 0, 1, 2 or 3).

3 - Adjust a due hour which is after closing time to the NEXT open day's opening hour.

In any setting, if the due hour is EARLIER than the opening hour, it is adjusted to the opening hour.

Col. 28 Item requested threshold. If an item has this number of active hold requests it will be considered requested for the purpose of using columns 15-18 for setting its due date. Default (blank) acts as "01". Do not set this column to "00" as this will cause all loans to be treated as recalls.

Example of the table:

```
! 1   2  3  4  5    6     7  8  9    10   11   12  13  4 5    16    7  18   19
20 21  22       23        24        25       26 2728
!!!!!-!!-!!-!!-!-!!!!!!!!-!!-!-!!!!-!!!!-!!!!!-!!!-!!!-!-!-!!!!!!!!-!-!!!!-!!!-!
!!-!!-!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!-!-!!

USM50 99 ## ## + 00000000 00 + 0000 0000 00000 020 020
!*
!* UEDUC UHLTH ULAW ULINC UMUSI USCI
!*
16A   30 ## 03 + 00000000 00 + 0300 0000 01200 003 005 1
16A   ## ## 20 A 20201231 00 + 2000 0000 00000 099 015 2
16A   ## ## 21 + 00000007 00 A 2000 0000 00100 099 005 2
   09
!*
16A   11 ## 10 + 00000001 00 A 2000 0000 00100 002 005 2
   09
16A   11 ## 01 + 00000001 00 A 2000 0000 00100 002 005 7
   09 D003
16A   11 ## 02 + 00000001 00 A 2000 0000 00100 002 005 2
   09 D003
16A   11 ## 03 + 00000001 00 A 2000 0000 00100 004 006 2
   09 D003
```

tab17

Location of the table: tab directory of the **ADM** library

Purpose of the table: Library open hours

Related table(s):

3) tab16 - cols. 11 and 14
4) tab_sub_library.<lng> - col. 8

Tab17 defines the days and times that the library is open. It serves 2 purposes:
sets due date to next open date
skips "closed" days when computing fines, depending on tab16 col. 14 (fine method) setup

A library does not have to utilize tab17 for fines. If it does, fines will be calculated based on the open hours of the library and not on a 24 hour day (as per definition of col. 14 in tab16.

Note that tab16 column 11 still defines the daily fine based on a 24 hour day and tab17 should be defined with this in mind.

Tab17 is also used to calculate due dates. Due dates will not be assigned days that are listed as C (closed) in tab17.

Tab17 is read from the top down, this means that exceptions should appear first, followed by the schedule of a typical week. The valid values for 'Day of Week' column (col. 3) are either numbers to specify the exact day, or spaces that stand for any day.

Structure of the table:

Col. 1	Group ID for group of definitions as defined in tab_sub_library.<lng> (col. 8). A Z30_DEPOSITORY_ID code can be used in this column to set a remote storage's opening hours disregarding the fact that a Z30_DEPOSITORY_ID code is not a sublibrary code.
col. 2	date YYYYMMDD. Use # as wildcard
col. 3	day of week (00 Sun - 06 Sat)
col. 4	O(pen)/C(losed)
col. 5	open hour
col. 6	closing hour

Example of the table:

```
17A   ####1230    C
17A   ####1231    C
17A   ####0101    C
17A   ####0102    C
17A   ######## 01 O 0900 2000
17A   ######## 02 O 0900 2000
17A   ######## 03 O 0900 2000
17A   ######## 04 O 0900 2000
17A   ######## 05 O 0900 2000
17A   ######## 06 C
17A   ######## 00 C
…
```

tab18.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Cash transaction types

Related table(s):

1) tab16
2) tab34
3) ill_bor_charge
4) tab100 - OVERDUE-RECALL-RATIO
5) tab27
6) tab_sub_library.<lng>
7) tab31

A table which defines cash control options. It is branch library/patron status sensitive.

Note that for some lines the value in col. 4 is Y or N (meaning that the system should or should not register a cash transaction), or a ratio. The actual amount is set in other tables. In the same case for some lines the value in col.5 is not relevant because the actual amount is set in other tables. In these cases col.5 should be left empty in order not to create confusion, except for specific cases mentioned below.

0003,0014 - late return fee, dependent on tab16
040,0041,0042 - lost material, dependent on tab34 If tab34 is not present or if there is no value in tab34, the default amount is taken from tab18.<lng> col.5. In any case the GUI operator can change the value for 0041 (material replacement cost) when registering an item as lost. The amounts for 0040 (handling cost) and for 0042 (notice cost) are not influenced by a manual change of the value for 0041 by the GUI operator
0015 - ILL material arrival, dependent on ill_bor_charge
0050-0055 - ratio, actual amount or additional daily amount charged for late return of recalled/rush recalled loan. The parameter OVERDUE-RECALL-RATIO in tab100 defines whether the number is a ratio, a set amount or a daily amount added to the fine as calculated in tab16.

General note: Note that the possibility to calculate a fine will depend on the existence of the loan history (Z36H), which will be used to locate the exact cash transactions related to the return.

List of transactions and their description:

Transaction No.	**Description**
0000	General
0001	Photo request
0002	Hold request
0003	Late return - fine is defined in tab16
0004	Loan
0005	Renewal
0006	Photocopy request processing
A3-6	Photocopy processing (A3 page size) – example
A4-6	Photocopy processing (A4 page size) – example
B2-6	Photocopy processing (B2 page size) – example
0007	Photocopy request not filled
0008	Photocopy request home delivery
0009	Photocopy request pickup
0010	Claim return
0011	ILL request
0012	SLNP cash transaction (cash transaction for the sum 0.00 (in col. 5) will be created with the status C = closed)
0013	Routing list
0014	Late return of routed issue
0015	ILL material arrival

Transaction No.	Description
0016	Library charge for incoming ILL request
0017	Printing patron's label card (bor-label-card)
0021	Local patron registration
0022	Local patron renewal
0023	New Patron
Charges 24-28 are activated when the hold is filled using "Print letter - hold request filled" from the Requests option in the Main Menu. See also charges 1024, 1026, 1027, 1028)	
0024	Hold request filled – tab27 process option type 01 (hold shelf)
0025	Hold request not filled
0026	Hold request filled - tab27 process option type 02 (home delivery)
0027	Hold request filled - tab27 process option type 03 (mailbox)
0028	Hold request filled - tab27 process option type 04 (reading room)
0040	Lost material – Handling amount defined in tab34, col. 9
0041	Lost material – Replacement amount defined in tab34, col. 10
0042	Lost material – Notice - amount defined in tab34, col. 8
0050	Recall late return fine If tab100 OVERDUE-RECALL-RATIO is "Y", then the fine amount in tab16 is multiplied by this number. If tab100 OVERDUE-RECALL-RATIO is "N", then this number is the actual fine amount. If tab100 OVERDUE-RECALL-RATIO is "B", then this number is added to the fine amount in tab16.
0051	Rush Recall late return fine See tab100 switch as above
0052	Recall fine limit ratio See tab100 switch as above
0053	Rush Recall fine limit ratio See tab100 switch as above
0054	Recall lost letter (ratio, actual amount or additional daily amount, see the explanation for 0050) added to the notice charge defined (Col.9) in tab34.
0055	Rush Recall lost letter (ratio, actual amount or additional see the explanation of 0050) added to the notice charge defined (Col. 10) in tab34
0072	SMS messaging charges
0073	Rush Cataloging Request

Transaction No.	Description
0080	1st warning – Overdue
0081	2nd warning – Overdue
0082	3rd warning – Overdue
0083	4th warning – Overdue
0084	5th warning – Overdue
0085	6th warning – Overdue
0090	Overdue summary
0091	Booking request
0092	Booking request not filled
The following charges (0093,0094,0095,0096) are activated when the booking is filled from the Circ Main Menu ("Booking request filled")	
0093	Booking request filled - tab27 process option type 01 (hold shelf)
0094	Booking request filled - tab27 process option type 02 (home delivery)
0095	Booking request filled - tab27 process option type 03 (mailbox)
0096	Booking request filled - tab27 process option type 04 (reading room)
The following charges (0097,0098,0099,0100) are activated when the booking is filled during a "Return Item Session"	
0097	Booking request filled - tab27 process option type 01 (hold shelf)
0098	Booking request filled - tab27 process option type 02 (home delivery)
0099	Booking request filled - tab27 process option type 03 (mailbox)
00100	Booking request filled - tab27 process option type 04 (reading room)
Charges 1024,1026,1027,1028 are activated when the hold is filled when the item is returned and the "Letter" button on list of holds is activated. See also charges 0024,0026,0027,0028	
1024	Hold request filled - tab27 process option type 01 (hold shelf)
1025	Hold request not filled
1026	Hold request filled - tab27 process option type 02 (home delivery)
1027	Hold request filled - tab27process option type 03 (mailbox)
1028	Hold request filled - tab27 process option type 04 (reading room)
9000	Deposit
9001	Return deposit
9996	Local charge for disruption
9997	Damaged material
9998	Donation

Transaction No.	Description
9999	Payment

Note that system activated transactions are assigned numbers from 0000 to 8999 while manually activated transactions are assigned numbers from 9000-9999.

Structure of the table:

Col.1 Transaction number – note that these numbers are system-defined and cannot be modified.

Col. 2 Sublibrary (use ##### to signify all sublibraries)

Col. 3 Item status (use ## to signify all statuses)

Col. 4 Patron status (use ## to signify all statuses)

Col. 5 Y/N/Z –

- Y – activate cash control
- N - do not activate. In this case cash transactions of this type will be registered with the value 0.00 without regard to tab18.<lng> col.5 or the value set in the relevant table.
- Z - register cash transactions only when cash rate is not zero. Note that the Z parameter affects only new transactions.

Col. 6 rate

Col. 7 VAT rate: left-aligned. This number is the rate of the VAT that will be added to the sum that is set in column 5. For example, if a transaction has the number 10.00 in column 5 and 15.00 in column 8, the net sum will be 10.00, the VAT sum will be 1.50 (15 percent of 10) and the sum of the transaction will be 11.50.

If the column is left empty, no VAT is assumed and the net value of the transaction will equal the total sum. Note that transactions which are calculated based on tab16 (such as 0003) or tab34 (such as 0040-0042) also have their VAT rates set in this table.

Col. 8 Charge for e-mail. Refers to the following cash transactions only: 0015,0024,0026,0027,0028,0042,0080,0081,0082,0083,0090, 1024,1026,1027,1028

- Y = Cash transactions are always charged, even if the letter is sent by e-mail
- N = If an e-mail-address is present in the patron's valid address record, no cash transaction is charged.

Please note, that in this case the line

```
form-print-method              EMAIL     S
```

has to be present in the matching translation file and column 4 in print.ini has to be set to "M" to ensure that notifications are sent via e-mail if possible.

Free of charge e-mail messages:

A library that uses certain cash transactions only in order to charge postal fees does not want to charge the patron if the letter is sent by e-mail. Set this column to **N** in order not to charge for postal fees.

Col. 9 description of function (up to 40 characters)

Note that in col. 1 it is possible to define a photocopy request fee differentiation according to paper type, for example,: if paper type is A4 then the entry should be defined with the key

A4-6 (A4 superimposed on 0006 key)

If paper type is B2 then define the as B2-6

If no special entry is defined defaults to 006.

Note that it is possible to define cash transactions for hold/photo requests based on the accompanying letter, different send-actions and/or or pickup locations.

Note that a catch-all line MUST be defined – as line 0000 in the example below.

Example of the table:

```
!1     2   3  4  5      6      7         8  9
!!!!-!!!!!-!!-!!-!-!!!!!!!!!!-!!!!!!!!!!-!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0000 ##### ## ## Y            10.00        General
0001 ##### ## ## Y 1.00                     Photocopy request
0002 ##### ## ## Y 3.00                     Hold request
0003 ##### ## ## Y                          Late return
0004 ##### 01 ## Y 5.00       10.00        Loan Item status 01
0004 ##### ## ## Y 2.00       10.00        Loan
```

tab20

Location of the table: tab directory of the **BIB** library

Purpose of the table: Links from BIB Headings to AUT Records

Related table(s):

1) tab_aut
2) tab00.<lng> - in the BIB library
3) tab01.<lng> - in the AUT library

This table defines the headings (including the see references) that will be built in the ACC table of the bibliographic base, based on the fields of the authority record that "matches" the BIB ACC heading. The headings added might be the result of cross-

reference information, or of headings in other languages in a multilingual authority database.

The table can be up to 2000 lines long.

Note that the definition of the relationship of the library's ACC headings table to an authority library is set in tab_aut. This table defines:
the authority database that is "linked" to a headings (ACC table,
and the headings list in the authority database that is used for searching for the authority record.

Tab20 defines:
the code of the source BIB ACC heading list (that is, the list from which ue_08 (UTIL E/8) picked up a "new" heading),
and the code of the target BIB ACC heading list (that is, the list on which new headings are opened)

When procedure UTIL E/8 is running, for all "-NEW-" Z01 (headings records), the system searches for the authority record that matches it.

This is done by searching the authority library's headings list, to find the appropriate record. The match between BIB heading and AUT record depends on the setup of the library's tab_aut table, which defines (for each BIB HEADINGS list) the AUT database(s) to search for a matching record for the list,
and the "heading use" appropriateness) position in the 008 field in the AUT library (for MARC 21 AUT databases).

There are two types of lines in this table,
the first line defining how to find a matching AUT record,
and the second line defining the AUT fields that create cross-references in the BIB HEADINGS list.
When the record is found, headings are added to the library's headings (ACC) table, based on the definitions in tab20.

NOTE :
If the AUT library uses the fix_doc_preferred_usm fix_doc procedure, ALEPH automatically copies the 1xx field in an authority record to COR field when the 1xx field is updated.
In order to have this correction also trigger correction of the bibliographic library's heading and document records, field code "COR" should be added to tab20, formatted in the same way as actual "see" references (4xx).

```
1 AUT   AUT   1 100##              -wi                    0
2               COR##              -wi                    0 SEEF
```

Note that the fix procedure fix_doc_preferred_usm is valid also for UNIMARC headings.

Multilingual Authorities Links:
When setting up multilingual links, if the library wants separate browse lists

for each language, the setup should be:
BIB ACC browse list for ALL the languages together,
plus individual lists for each language (for example, SUB for all subjects,
SUBE for English subjects, SUBF for French subjects, and so on.).
filter of the AUT records by language subfield.
The example following shows the setup for multilingual SUBJECTS,
where
SUB is the BIB library's general subjects list, and includes subjects in all languages.
SUBG includes subjects in German form,
SUBE includes subjects in English form, and
SUBF includes subjects in French form.

"SUB" is searched on the AUT library's GEN headings; when a matching record is found, it is linked back to the BIB SUB, SUBG, SUBE and SUBF headings lists.
The AUT record has multiple 150 fields, with $$9 differentiating between the English, German and French forms of the subject.
In some instances the subject is the same in all forms, in which case the 150 field does not have $$9, which is indicated here as 9 -.

```
!  !-!!!!!-!!!!!-!!!!!-!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!-!!!!!
!  1 SUB   SUB   150##              -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBG  150## 9 ger        -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBE  150## 9 eng        -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBF  150## 9 fre        -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBG  150## 9 -          -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBE  150## 9 -          -wi                    0
!  2             450##              -wi                    0 SEEF
!  1 SUB   SUBF  150## 9 -          -wi                    0
```

Structure of the table:

- Col. 1 Line type (1 or 2);
- Col. 2 Source. Used for line type 1 only. Code of the heading list in the BIB library whose records are used for finding an AUT record.
- Col. 3 Target. Used for line type 1 only. Code of the heading list in the BIB library whose records are updated with a link to the AUT record
- Col. 4 Field tag. For line type 1:

  AUThority record field tag to check for match against the BIB HEADING record.

  For line type 2:

  AUThority record field tag for building cross reference.
- Col. 5 Subfield filter. Together with the next column, the subfield and

text that must be present in the AUT record field, in order for the AUT record field be considered a match for the BIB heading.

Col. 6 Subfield text. Together with the previous column, this is the text (in the subfield defined in the previous column) that must be present in the AUT record's 4XX field, in order that the field be used for enriching the BIB library's HEADINGS list. In MARC 21, subfield $w, 4th position (that is, $wxxx4) contains a code that enables the generation or suppression of a cross-reference from 4XX or 5XX fields. If the fourth position of subfield $w contains a blank or contains an 'n' (or the subfield does not exist), the intention is that cross references will display. If the fourth position of subfield $w contains a different character, the intention is that the cross-reference is suppressed. This MARC 21 principle is applied in this column of this tab20 table.

Col. 7 Subfields to take or strip from the authority record field as a basis for the match with the bibliographic heading.

Col. 8. Indicator. Used for line type 1 only.

Indicator (first or second) to use for stripping initial non-filing text.

Col. 9 SEEF. Used for line type 2 only.

Example of the table:

```
!   2     3     4   5     6                7           8  9
!-!!!!!-!!!!!-!!!!!-!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!-!!!!!
1 AUT   AUT   100##                                    0
2             400## w !!!n       -wi5                 0 SEEF

1 AUT   AUT   100##                                    0
…
1 TIT   TIT   130##                                    2
2             430## w !!!n       -wi5                 2 SEEF

1 TIT   TIT   130##                                    2
2             430## w !!!        -wi5                 0 SEEF
```

tab22

Location of the table: tab directory of the **BIB** library

Purpose of the table: Z13 fields definition

Related table(s):

1) edit_paragraph.<lng>
2) tab_expand_join (optional)
3) tab_expand. (optional)

tab22 defines from where to take data for the "fixed fields" of the Z13 (short-doc) record. Z13 is used for SQL reports, "quick" filter of set in OPAC, and various instances of print and display of bibliographic information in the ADM environment.

Some instances are:
Web OPAC: BASKET and PATRON (ILL, loans, holds, SDI)
Brief display when browsing by system number
CIRC list of loans

Some of the Z13 codes are system-defined. In addition, 15 additional codes can be set by the library.

For system codes
YEAR is 4 positions,
CALL-NO-KEY is 80 positions,
and all other fields that are taken from a document record field are up to100 positions.
For library-defined codes, up to 500 characters are taken from document record fields.

System codes must be based on field tags, whereas library-defined codes can be based on field tags, or on "paragraphs" defined in edit_paragraph.<lng>.

The Z13 record automatically includes OPEN-DATE and UPDATE -DATE.
If you do not want to set a bibliographic record field for one of the lines in this table, leave cols. 3-13 blank.

The system-defined Z13 fields are:
YEAR = year
CALL-NO = call number
CALL-NO-K = call number key (not currently implemented)
AUTHOR = author
TITLE = title
IMPRINT = imprint
ISBN-ISSN = ISBN/ISSN

The library-defined Z13 fields 1-5 can be:
USER-DEF-N = user defined n
or
USER-N
Fields 6-15 are defined as:
USER-N or
USER-NN

Each group of columns (tag + subfield + position) is used to define an alternative field, if the tag in the previous group of columns is not present. The system takes the first of the alternative fields that it finds when creating the Z13 record.

Use tag 'SYS' for the BIB record's system number, if you want to map it to Z13.

Virtual fields, created by tab_expand_join, can be used, in which case expand_doc_join must be listed under CREATE-Z13 in tab_expand.

When Col. 2 is set to 1, the data is taken from bib record's tag + subfield + position. When Col. 2 is set to 2, the data is taken from the bibliographic record, using edit_paragraph. In this case col 3 is used to list the paragraph number and the remaining columns are left blank.

The system takes the first of the alternative fields that it finds when creating the Z13 record.

Structure of the table:

Col. 1 Z13 tag:

Col. 2 Function code:

- 1=data taken bib record's tag + subfield + position

2=data taken from the bib, using

- edit_paragraph.<lng>.

Col. 3 Field tag + indicators

Col. 4 subfield

Col. 5 Defines the position from which to take data from a fixed field.

For example, if column 1 is YEAR 1, the year might be taken from a fixed field. In this case, define the position in the fixed field from which to commence taking 4 positions, counting from base 01. If the fixed field has a subfield code, add 3 to the starting position in order to take it ($$x) into account. for example,

0008 to define the 8th position of the 008 field in MARC 21,

0013 to define the 9th position of the 100 field in UNIMARC.

Col. 6 alternative tag if the first tag is not present

Col. 7 subfield

Col. 8 starting position

Col. 9-11 as cols. 6-8

Col. 12-14 as cols. 6-8

Col. 15 –17 as cols. 6-8

Example of table:

```
!   1      2   3   4  5     6   7  8     9  10  11   12  13  14   15  16  17
!!!!!!!!!!-!-!!!!!-!-!!!!-!!!!!-!-!!!!-!!!!!-!-!!!!-!!!!!-!-!!!!-!!!!!-!-!!!!
YEAR       1 008##   0008 260## c
CALL-NO    1 050## a      LOC## n
CALL-NO-K  1 LOC## n
AUTHOR     1 1####
TITLE      1 245## a
IMPRINT    1 260##
ISBN-ISSN  1 020##        022##
USER-DEF-1 2 109
USER-DEF-2 2 009
```

tab22_arc

Location of the table: tab directory of the **BIB** library

Purpose of the table: fields for ARC extractions

Related table(s):

edit_paragraph.<lng>

tab22_arc defines from where to take bibliographic data for library-defined "fixed fields" for ARC extractions.

Note:
This configuration table is used in the process of ETL only - Extract Aleph information to ARC (arc-01).

There are 70 library-defined fields that can be set by the library. The fields are defined in column 1 as:
ARC-DEF-1 = user defined 1
ARC-DEF-2 = user defined 2
ARC-DEF-3 = user defined 3
ARC-DEF-4 = user defined 4
ARC-DEF-5 = user defined 5
And so on until ARC-DEF-70.

Each line should be used to define a unique alternative library-defined field.
Note that if a similar line of tag + subfield + position + length repeats, the system takes the first of the alternative fields that it finds (group of columns of: tag + subfield + position + length).

When Col.2 is set to 2, the data is taken from the Bib record, using edit_paragraph. In this case, col.3 is used to list the paragraph number, and the remaining columns are left blank.

The following is the structure of the table:

Col. 1. field code;
ARC-DEF-1, ARC-DEF-2, ARC-DEF-3, ARC-DEF-4, ARC-DEF-5 ... till ARC-DEF-70
Col. 2. Function code:
1=data taken from the bib record's tag + subfield + position
2=data taken from the bib, using edit_paragraph;
Col. 3. Field tag + indicators or Paragraph number;
If col.2 is set to 1, enter Field Tag.
If col.2 is set to 2, enter Paragraph number from the edit_paragraph table.
Col. 4. Subfield;
Col. 5. Starting position:
Defines the position from which to take data from a fixed field. For example, if column 1 is YEAR 1, the year might be taken from a fixed field. In this case, you will define the position in the fixed field from which to commence, taking 4 positions, counting from base 01. If the fixed field has a subfield code, add 3 to the starting position in order to take it into account.
e.g. 0008 to define the 8th position of the 008 field in MARC21, 0013 to define the 9th position of the 100 field in UNIMARC;
Col. 6. Length: Number of bytes from Starting position.
Defines the number of bytes taken from starting position.

The following is an example of the table:

```
!   1      2   3   4  5    6
!!!!!!!!!!-!-!!!!!-!-!!!!-!!!!
ARC-DEF-1  1 LDR     0006 0001
ARC-DEF-2  1 LDR     0007 0001
ARC-DEF-3  1 FMT
ARC-DEF-4  1 STA   a
ARC-DEF-5  1 007     0000 0001
ARC-DEF-6  1 007     0001 0001
ARC-DEF-7  1 007     0004 0001
ARC-DEF-8  1 008     0006 0001
ARC-DEF-9  1 008     0007 0003
ARC-DEF-10 1 008     0015 0002
ARC-DEF-11 1 008     0021 0001
ARC-DEF-12 1 008     0022 0001
ARC-DEF-13 1 008     0028 0001
ARC-DEF-14 1 008     0035 0002
ARC-DEF-15 1 020   a
ARC-DEF-16 1 020   z
ARC-DEF-17 1 020   c
ARC-DEF-18 1 022   a
ARC-DEF-19 1 041   a
ARC-DEF-20 1 084   2
ARC-DEF-21 1 084   a
ARC-DEF-22 1 090   a
ARC-DEF-23 1 100   a
ARC-DEF-24 1 110   a
ARC-DEF-25 1 111   a
ARC-DEF-26 1 245   a
ARC-DEF-27 1 245   b
ARC-DEF-28 1 245   c
ARC-DEF-29 1 245   p
ARC-DEF-30 1 245   n
ARC-DEF-31 1 246   a
ARC-DEF-32 1 250   a
ARC-DEF-33 1 260   a
ARC-DEF-34 1 260   b
ARC-DEF-35 1 260   c
ARC-DEF-36 1 300   a
ARC-DEF-37 1 300   b
ARC-DEF-38 1 300   c
ARC-DEF-39 1 310   a
ARC-DEF-40 1 563   a
ARC-DEF-41 1 700## a
ARC-DEF-42 1 700## d
ARC-DEF-43 1 700## e
ARC-DEF-44 1 700##   0000 9999
ARC-DEF-45 1 740## a
ARC-DEF-46 1 740## h
ARC-DEF-47 1 740##   0000 9999
ARC-DEF-48 1 710## a
ARC-DEF-49 1 711## a
ARC-DEF-50 1 949## #
ARC-DEF-51 1 5#### a
ARC-DEF-52 1 5#### b
```

tab24

Location of the table: tab directory of the **ADM** library

Purpose of the table: Delivery times from remote storage

Related table(s):

1) tab17
2) tab_sub_library.<lng>

This table defines the maximum amount of time from the current time that is required for delivery from the remote storage, and the set times that items sent from the Remote storage arrive at the library.

Note that the expected delivery time from tab24 is checked against tab17, to determine that both the Remote storage and the sublibrary are open at the relevant time. If neither is open, the expected delivery time is set to the next time and the first tab24 available time. If the end-of-day is reached, the next time is the first time as defined for the next day. This new time is re-checked against tab17, an so on.

Structure of the table:

col 1 Sublibrary of the item

col. 2 Type of request that this line refers to:

- R = Hold Request
- P = Photo Request

col. 3 Delay from request time to delivery time (in minutes)

col. 4-9 Time of day to expect material

Example of the table:

```
! 1   2  3    4   5    6    7    8    9
!!!!!-!-!!!-!!!!-!!!!-!!!!-!!!!-!!!!-!!!!
HYL   # 120 1000 1400 1500
LAW   # 480 1100
```

tab25.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Possible values for item material types

Related table(s):

1) tab_z30_sort

This table defines the list of possible values for Z30-MATERIAL (item material type).

Material types that begin with ISSXX have special functionality.

Items marked ISSXX will sort according to cols. 2 and 3 of tab_z30_sort. All other items sort according to cols. 4 and 5 of tab_z30_sort.

Structure of the table:

Col. 1 Code

Col. 2 ALPHA

Col. 3 Name

Col. 4 Magnetic media definition. N/Y. This is used for identification of magnetic media in order to stop magnetization or demagnetization when using 3M equipment.

Note: In order to manage magnetization, a section [MagneticMedia966] has been added to the file circ.ini in the online clients.

The only line in this section is Port=nn

When set to 0 this option is turned off. This should be set to the serial port the 966 module is attached to (if there is such a module).

Example of the table:

```
! 1   2               3                 4
!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!
BOOK  L Book                            N
ISSUE L Serial                          N
ISSBD L Bound Volume                    N
VIDEO L Video Cassette                  Y
AUDIO L Audio Cassette                  Y
DIGIT L Digital Form                    Y
MANUS L Manuscript                      N
```

Note: Material type ISSBD for bound issues. The material-type is changed from ISSUE to ISSBD after the binding process. In all the usual functions (SORT, DISPLAY and so on..) bound issue will be treated as a regular issue.

tab27

Location of the table: tab directory of the **ADM** library

Purpose of the table: Hold requests send method configuration

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>

tab27 defines:

- For hold and ILL requests - whether the item is assigned to the "hold shelf" or immediately loaned when it becomes available.
- For booking requests : how the item is to be handled when it is delivered.

Structure of the table:

Col. 1 Request type. (regular "**H**"olds or "**B**"ooking)

Note that 01, 02, 03, 04, 05 are the valid values for hold requests while 06, 07 are the valid values for booking requests

Col. 2 sublibrary of the item

Col. 3 item status

Col. 4 Pickup location. Location to which item will be sent (either to be picked up by the patron, or actually sent to the patron)

Col. 5 process option. These are the actions that will result from operator clicking "Letter" on the Holds Requests display (when returning item,) or performing Print letter - Hold request filled in GUI CIRC.

- **01** generates a pickup letter for the patron and a hold shelf slip for the item. The loan will be performed in a regular manner when the item is picked up. This should be used for the normal library "hold shelf" functionality.
- **02** generates a delivery slip for the item, and automatically loans the item to the patron. This can be used for home delivery.
- **03** generates a pickup letter for the patron and a delivery slip for the item, and automatically loans the item to the patron. This can be used if the library puts the requested material in the patron's mailbox, and also wants to inform the patron that the material is now available.
- **04** reading room: performs option 02 and creates a Z310 record for Reading Room control.
- **05** for items received through ILL: loan to the pickup sublibrary, taking the pickup sublibrary from Z410. If there is no pickup sublibrary, loans to the ILL Library of the requesting patron. This requires that the sublibrary code is registered as a patron with a Z303 and a Z305 record
- **06** for booking requests. When the item is delivered it will be automatically loaned to the patron

- **07** for booking requests. When the item is delivered it is put in transit to the pickup location.
- **08** for booking requests. When the item is delivered it is put in the Reading Room. The item is automatically loaned to the patron and a Z310 record for Reading Room control is created

Example of the table:

```
!   2   3    4   5
!-!!!!!-!!-!!!!!-!!
H #####  73 #####  03
H #####  72 #####  02
H #####  75 #####  05

H #####  ## HILR#  04
H #####  ## WIDR#  04
H #####  ## HOME   02
H #####  ## #####  01
B #####  ## HILR   08
B #####  ## WIDR   08
B #####  ## #####  06
```

tab30

Location of the table: tab directory of the **ADM** library

Purpose of the table: Patron registration renewal

Related table(s):

1) tab_check_circ
2) tab100 - BOR-EXPIRY-DUE-DATE
3) tab_sub_library.<lng>
4) tab31

Tab30 allows the library to define defaults for two elements with regard to expiry of patron registration:
How many days before the expiry date should the system warn the operator of an approaching expiry date (in CIRC). If a patron borrows or renews an item within the defined period, the system gives a warning "expiry date approaching" together with the expiry date. This warning is only displayed if check_circ_1_c_b for LOAN or RENEW is defined in tab_check_circ. Note that dependent on the switch BOR-EXPIRY-DUE-DATE in tab100 the due date of the item can be set so that it does not fall beyond the expiry date.
The default renewal period. This can be seen online when the renew button (on the Update Local Patrons screen) is activated.

Structure of the table:

Col. 1 patron status
Col. 2 sublibrary (use # for wildcard)
Col. 3 registration renewal warning (number of days before expiry)
Col. 4 registration renewal period: years
Col. 5 registration renewal period: months

Example of the table:

```
!    2    3  4  5
!!-!!!!!-!!!-!!-!!

12 UUPTN 014 01 00
14 UUPTN 014 01 00
15 UUPTN 014 01 00
16 UUPTN 014 01 00
10 ##### 028 01 00
01 ##### 028 01 00
02 ##### 028 01 00
08 ##### 028 01 00
```

tab31

Location of the table: tab directory of the **ADM** library

Purpose of the table: Patron statuses

Related table(s):

1) pc_tab_exp_field.<lng>
2) pc_tab_exp_field_extended.<lng>
3) tab_sub_library.<lng>

This table defines default privileges and restrictions for the patron based on the patron status. When a new patron is created online the system will use these defaults for creating patron privileges and other definitions.
Status MUST be defined here in order to be valid in the system. The ***translation*** of the codes to patron groups (for example, 01=under graduate) should also be registered under the BOR-STATUS in either pc_tab_exp_field.<lng> or pc_tab_exp_field_extended.<lng> so that they will appear in the patron status pull-down menu.

The Default Hold Priority values should also be registered under HOLD-PRIORITY in pc_tab_exp_field.<lng> so they will appear in the hold priority drop-down menu.

Maximum number of lines is 1000.

Structure of the table:

col. 1 sublibrary
col. 2 patron status (01-99)
col. 3 loan permission (Y/N)
col. 4 photo permission (Y/N)

col. 5 override permission: can an override on a loan transaction be invoked or is the loan blocked

col. 6 multiple hold permission: can the patron place multiple holds on the same record?

col. 7 check loan: should the system check if the patron can loan the item?

col. 8 hold permission (Y/N): can the patron place hold requests?

col. 9 renew permission: can the patron renew items in the OPAC?

col. 10 ignore late return: should items returned late be registered (for fining)?

col. 11 photocopy charge

- C-charge
- F-free

col. 12 expiry date operator

- +=add to current date
- A=actual date

col. 13 expiry date operator type (when col. 12 has '+')

- D=day, M=month, Y=year

col. 14 expiry date parameter

- if col. 12 has '+': no. of days to add
- if col. 12 has 'A' actual date

col. 15 cash overspend limit - last two digits are decimal

col. 16 request hold for item on shelf (Y/N)

col. 17 loan display – Y/N

allows for the display of patron information in the Web OPAC for an item loaned out – for example, a department library that has permanent loan items.

col. 18 reading room permission

col. 19 Default Hold priority

col. 20 Item Booking permission

col. 21 Booking - Ignore closing hours

col. 22 Automatically create ALEPH record

col 23. Rush Cataloging permission

Example of the table:

```
!                      1 1 1 1                     1 1 1 1  2 2 2 2
! 1   2  3 4 5 6 7 8 9 0 1 2 3   14        15      6 7 8 9  0 1 2 3
!!!!!-!!-!-!-!-!-!-!-!-!-!-!!!!!!!!-!!!!!!!!!!-!-!-!-!!-!-!-!-!
WID   02 Y Y Y N Y Y Y N F + M 00000006 9999999999 N Y Y 21 Y N Y Y
##### 01 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 10 N N Y N
##### 02 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 20 Y N Y Y
##### 03 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 30 Y N Y Y
##### 08 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 30 Y N Y N
```

```
!*
!* Students
##### 12 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 30 Y N N N
##### 13 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 30 Y N N N
##### 14 Y N Y N Y Y Y N C + Y 00000001 0000000000 N N Y 30 Y N N N
```

tab32

Location of the table: tab directory of the **ADM** library

Purpose of the table: Overdue letter intervals

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31
4) tab18.<lng> - col. 8 (or assumed)
5) tab34

Structure of the table:

col. 1 sublibrary (##### to signify all sublibraries)

col. 2 item status (## for all)

col. 3 patron status (## for all

col. 4 current letter number in the loan transaction record (Z36). If the letter number is '00', no previous letters have been sent, and therefore this is the first notice. If the letter number is '01', this is the second notice, and so on.

col. 5 Number of days since : if
4 = 00 - number of days since due date overdue
4 > 00 - number of days since last overdue letter

4 = 09 - number of days since due date for items which are "claimed returned". This column is used when p-cir-50 (Print Overdue and Lost Billing Notices) is run for "Bills for lost material that was claimed to have been returned".

col. 6 Number of days for requested items: if

4 = 00 - number of days since due date overdue
4 > 00 - number of days since last overdue letter

4 = 09 - number of days since due date for items which are "claimed returned". This column is used when p-cir-50 (Print Overdue and Lost Billing Notices) is run for "Bills for lost material that was claimed to have been returned"

col. 7 action:
O - overdue letter; can incur overdue letter charge, using column 8 and tab18.<lng>

L - lost letter, can incur lost charges, using lines 0040, 0041, 0042, 0054 and 0055 in tab18, and tab34. The tab18 lines do not have to be defined in column 8, they are automatically assumed.

col. 8 transaction cost - line no. in tab18

Note: The system reads the table from the top down. Item status "##" is used to denote "all" item statuses.

Example of the table:

```
! 1   2  3  4   5   6  7  8
!!!!!-!!-!!-!!-!!!-!!!-!-!!!!
##### 11 ## 00 000 000 O 0080
##### 11 ## 01 000 000 O 0081
##### 11 ## 02 000 000 O 0082
##### 11 ## 03 000 000 L
##### 11 ## 09 021 021 L
!*
##### 31 ## 00 005 000 O 0080
##### 31 ## 01 007 000 O 0081
##### 31 ## 02 014 000 O 0082
##### 31 ## 03 014 014 L
##### 31 ## 09 021 021 L
!*
##### 12 ## 00 005 000 O 0080
##### 12 ## 01 007 000 O 0081
##### 12 ## 02 014 000 O 0082
##### 12 ## 03 021 021 L
```

tab33.<lng>

Location of the table: tab directory of the **BIB** and **ADM** libraries

Purpose of the table: Defines the format of the BIB-ADM record that displays in the navigation window in the various GUI modules

Related table(s):

1) edit_field.<lng> - Col. 4 – D line

This table should be present in each BIB and ADM library.
The table in the BIB library defines the fields from the BIB record. The table in the ADM library defines the fields from the ADM record.
Note that the order of the fields in the record, and not in the table, determines the display order of the fields.
NOTE that the "D" lines (D in col. 4) of the BIB library's /tab/ edit_field.<lng> table, set the formatting of the BIB field.

Structure of the table:

Col. 1 tag
Col. 2 ALPHA
Col. 3 caption

Example of the table in the ADM library

```
! 1   2     3
!!!!!-!-!!!!!!!!!!!!!!!!!!
FMT   L FormatADM
ACQ## L Acquisition note
Z68## L Order note
Z75## L Invoice note
Z78## L Arrival note
SER## L Serials note
Z16## L Subscription note
Z17## L Issue note
Z18## L Routing note
Z14## L Member note
ITM## L Items note
Z30   L Permanent loc.
```

Example of the table in the BIB library

```
! 1   2     3
!!!!!-!-!!!!!!!!!!!!!!!!!!
020## L ISBN
022## L ISSN
100## L Author
245## L Title
250## L Edition
260## L Imprint
300## L Collation
310## L Frequency
4#### L Series
70### L Added Author
74### L Added Title
```

tab34

Location of the table: tab directory of the **ADM** library

Purpose of the table: Lost material costs

Related table(s):

1) tab32
2) tab100 - LOST-LOAN-CREDIT-METHOD , REFUND-RATE and LATE-RET-FINE-WHEN-LOST
3) tab_sub_library.<lng>
4) tab15.<lng>
5) tab31
6) tab40.<lng>
7) tab18.<lng> -- transactions 40, 41, 42

Tab34 is an optional table that computes the charges to be added to the patron's cash record in case an item is changed to lost, (either through the LOST function in the

Circulation module, or as a result of batch services p-cir-50, -51 and -52, which use tab32 to declare an item as ***lost***).

Note that if the library does not define default charges in this table, the system will not be able to compute a **default** charge.
It is possible for the user to manually input the replacement cost. This is configurable from circ.ini:
[Lost]
ManualReplacement=Y/N

The charges are defined for each item-patron combination.
There are three types of charges:
notice,
replacement handling
price.

When a ***lost*** item is found, the different charges are waived depending on the tab100 variables LOST-LOAN-CREDIT-METHOD and REFUND-RATE

Structure of the table:

Col. 1	sublibrary – use # for wildcard
Col. 2	item status
Col. 3	patron status
Col 4	call number type
Col. 5	call number. This can be used to either specify an exact call number or specify a range. Specifying a range is done by using a # mark to separate the From call number from the To call number. For example, ABC#DEF means from call number ABC to call number DEF. For all call numbers, use only one # in the first position of the column and a space in the second position of the column.
Col. 6	collection
Col 7	item material type
Col. 8	notice cost. This amount will not be credited to the patron if the item is returned. Transaction 0042 in tab18.<lng> must be set to Y.
Col. 9	handling cost. This amount will be credited to the patron if the item is returned. Transaction 0040 in tab18.<lng> must be set to Y.
Col. 10	fixed price. This amount will be credited to the patron if the item is returned. Transaction 0041 in tab18.<lng> must be set to Y. You may also indicate that Z30-PRICE will be consulted prior to this column by using the prefix "I/" (for example, I/420.00). This applies only if Z30-PRICE is a decimal number other than zero before the first space in the field.
Col. 11	MARC price (placeholder for Danish implementation)
Col. 12	patron factor (Not yet implemented)

Example of the table:

```
!
!1     2  3 4                                  5                                  6
8         9          10         11       12
!!!!!-!!-!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!
!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!!!!!!-!!!!!>
##### ## 63 # #                                                                  ####
0.00       0.00       0.00
HIL   ## ## # $$hPS 123$$i.P12                                                   ####
0.00       0.00       77.00
HIL   ## ## # #                                                                 ####
0.00       0.00       50.00
HYL   ## 01 # #                                                                 ####
0.00       5.00       75.00
```

tab35

Location of the table: tab directory of the **ADM** library

Purpose of the table: Location numbers and other information for EDI orders

Related table(s):

1) tab_sub_library.<lng>

Every sublibrary may have different EDI ID numbers at different vendors. In addition, different sublibraries can have one account at the same vendor.
Tab35 allows for the defining of such numbers for each branch library. In addition the table allows for the registration of each sublibrary's VAT number.

Structure of the table:

Col. 1 sublibrary or order unit

Col. 2 Vendor code

Col. 3 Customer's EDI code

Col. 4 Customer's EDI code type

- 31B for US SAN,
- 14 for EAN-13
- 91 for ID assigned by supplier
- 92 for ID assigned by customer.

Col. 5 Not in use

Col. 6 Not in use

Col. 7 VAT number

Col. 8 E-mail address for reporting errors in order processing

Example of the table:

```
UARCV HAR                   IOWA-SER              091          IL
111 222 331                  yifat.lulav@exlibris.co.il
ORDU1 HAR                   IOWA-SER              091          IL
111 222 331                  yifat.lulav@exlibris.co.il

UARCV SWETS                 IOWA-SER              091          IL
111 222 331                  yossi.loss@exlibris.co.il
…
```

Note that if EDI communication is active for a vendor, the vendor's EDI code MUST be defined in the "vendor EDI code" field and the code type MUST be defined in "vendor EDI type" field. Both fields are part of the online ACQ vendor form.

tab36

Location of the table: tab directory of the **ADM** library

Purpose of the table: automatic opening of monograph-type items in Acquisitions

Related table(s):

1) tab100 - CREATE-ITM-FORM-ORDER-M
2) tab_sub_library.<lng>
3) tab15.<lng>
4) tab40.<lng>
5) tab25.<lng>
6) pc_tab_exp_field.<lng> - ITEM-LOCATION-TYPE menu

The system automatically opens items for monograph orders according to the number in the **Number of Units** field in the order record if **Create item records** on the monograph order form is selected. The default value for **Create Items Records** is determined by the tab100 variable CREATE-ITEM-FROM-ORDER-M.

tab36 defines per order sublibrary (col.1) and order material type (col.2) the default values that are assigned to the following automatically created item records: item statues, item material type, item collection, item call number, and item call number type.

Structure of the table:

Col. 1 Order sublibrary filter (use ##### for wildcard)
Col.2 Order material type filter (use ## for wildcard)
Col. 3 Item status
Col. 4 Item material type
Col. 5 Item collection
Col. 6 Item call number
Col. 7 Item call number type

The following is an example from the table:

```
  1    2  3     4    5                6                                   7
!!!!!-!!- !!-!!!!!-!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!
LAW   BK  01 BOOK  GEN                                                   0
LAW   DV  02 DVD   GEN                                                   0
LAW   MP  02 MAP   GEN                                                   0
LAW   ##  01 BOOK  GEN                                                   0
##### ##  01 BOOK  GEN                                                   0
```

Notes:

The item sublibrary is the same as the order sublibrary.

If the item's collection is not defined in the Quantity and Price tab of the Order form, tab36 is consulted.

The actual number of item records that are created is determined by the "number of units" entered in the Quantity and Price tab of the Order Form.

The item values entered in columns 3 thru 6 (inclusive) should match corresponding values as defined in the following Aleph configuration tables: tab25.lng (item material type), tab15.lng (item status), tab40.lng (item collection), and ITEM-LOCATION-TYPE menu of pc_tab_exp_field.lng (item call number type).

tab37

Location of the table: tab directory of the **ADM** library

Purpose of the table: Pickup list configuration for hold requests

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31

tab37 defines one aspect of hold requests: the drop-down list of pickup and delivery locations for hold requests (dependent on item and patron status, and on sublibrary of item).

NOTE the following:
If the item sublibrary, item status, item process status, patron status and pickup location are **not** defined in this table, the item **cannot** be requested.

If you are using a specific item status in col.2 (that is, not ##), make sure that the specific lines come before the general lines, since the table is read from top to bottom, and the first match is always taken. The same holds true for col. 3 (item process status) and col. 4 (patron status).

All item statuses are matched on ## in the first line, and the second line will never be used.

```
Example of correct setup:
! ULINC 14 ## ## # UEDUC
! ULINC ## ## ## # UEDUC
!
Example of incorrect setup:
! ULINC ## ## ## # UEDUC
! ULINC 14 ## ## # UEDUC
```

If you have a combination of one specific item status with all patron statuses and a combination of all item statuses with a specific patron status, define these lines first.

For example:
You want all items with item status 14 except those requested by a patron with patron status 05 to be ready for pickup at UEDUC, and all items ! with patron status 05 to be ready for pickup at UEDUC, and all items requested by a patron with patron status 05 to be ready for pickup at UHLTH. In order to achieve this, specify lines for the combination of the two specific statuses:

Example:

```
ULINC 14 ## 05 # UHLTH
ULINC 14 ## ## # UEDUC
ULINC ## ## 05 # UHLTH
ULINC ## ## ## # ULINC
```

You can define up to 1000 different pickup locations for each combination. Up to ten different locations can be entered on each line. If there is more than one line per combination (to allow for more than ten pickup locations), leave columns 1-5 empty in the other lines.

For example :

```
UHLTH ## ## ## # XXX01 XXX02 XXX03 XXX04 XXX05 XXX06 XXX07 XXX08 XXX09 XXX10
                 XXX11 XXX12 XXX13 XXX14 XXX15 XXX16 XXX17 XXX18 XXX19 XXX20
                 XXX21 XXX22 XXX23
```

Display of the pickup sublibrary drop-down list
Only the most specific line matching sublibrary/item-status/item-process-status is taken for the drop-down list (that is, although ## lines would match they are not included in the drop-down list if a more specific line was found).

Note the following: Column 5 (Availability of item) can be used to filter out a sublibrary from the drop-down list of pickup locations.
If column 5 is set to "N", the pickup location will display only if the item is not available (that is, item is on loan or on hold shelf).

There are two correct ways to use this:
by exclusion:

```
UHLTH ## ## ## N HOME
UHLTH ## ## ## # UHLTH UEDUC
```

by inclusion:

```
UHLTH ## ## ## Y UHLTH UEDUC
UHLTH ## ## ## N UHLTH UEDUC HOME
```

Structure of the table:

Col. 1 Sublibrary of the item (can use #)

Col. 2 Status of item being requested (can use #)

Col. 3 Process Status of item being requested (can use #)

Col. 4 Status of the patron (can use #)

Col. 5 availability of item.

If "Y" this line only applies to available items (that is, on the shelf).

If "N" this line only applies to unavailable items (that is, on loan, hold waiting to be picked up or in transit).

Use # to indicate that the line is insensitive to availability of the item

Col. 6-15

pickup location. Location to which item will be sent (either to be picked up by the patron, or actually sent to the patron).

Example of the table:

```
! 1   2  3  4  5   6     7     8     9    10    11    12    13    14    15
!!!!!-!!-!!-!!-!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!…!!!!!

##### IL ## 03 # HOME  ULINC UMUSI USCI  UUPTN
##### ## ## 11 # ILLDT ULINC UMUSI USCI  UUPTN
UARCV ## ## ## # UARCV UEDUC ULAW  ULINC UMUSI USCI  UUPTN RLINC
…
ULINC 42 DP 02 # UMUSI
ULINC 42 DP ## # ULINC UUPTN UEDUC UHLTH ULAW
ULINC 73 ## ## # BOX   HOME  ULINC UEDUC UHLTH ULAW  UMUSI USCI  UUPTN RLINC
ULINC ## ## ## # ULINC UEDUC UHLTH ULAW  UMUSI USCI  UUPTN HOME  RLINC
```

NOTE that in tab_sub_library.<lng>, it is possible to define different *library types:*

1 - full sublibrary (patron and items)

2 - patron sublibrary only (no items)

3 - special

4 - reading room sublibrary (1 + special request handling)

5 – ordering unit

For the purpose of tab37, it is possible, for example, to define different *mailboxes* which need to *also* be defined in tab_sub_library:

Example from tab_sub_library.<lng>:

```
library - type
BOX1      3    USM50 L Pickup in Forum
```

Where *Box1* is the “library”, and the library type is 3=special

tab37_booking_delivery

Location of the table: tab directory of the **ADM** library

Purpose of the table: Booking Requests delivery configuration

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31
4) tab_delivery_locations.<lng>

Tab37_booking_delivery defines the drop-down list of delivery locations for Booking requests (dependent on item status, item process status, patron status and sublibrary of item).

These delivery locations ***must not*** be valid sublibraries in tab_sub_library.<lng>. They ***must be*** valid locations in the XXX50_tab/tab_delivery_locations.<lng> table.

You can define up to 1000 different delivery locations for each combination. Up to ten different locations can be entered on each line.
If there is more than one line per combination (to allow for more than ten pickup locations), leave columns 1-4 empty in the other lines.

Structure of the table:

Col. 1 Sublibrary of the item

Col. 2 Status of item being requested

Col. 3 Process Status of item being requested

Col. 4 Item material

Col. 5 Status of the patron

Col. 6 -15

Delivery location. Location to which item will be sent.

Example of the table:

```
! 1   2  3    4   5    6     7     8     9    10    11    12    13    14    15
!!!!!-!!-!!-!!!!!-!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!
HIL   ## ## ##### ## LOC1
HYL   ## ## ##### ## LOC1
LAM   ## ## ##### ## LOC1
LAW   ## HD ##### ## LOC1
LAW   ## ## ##### ## LOC1
LIT   ## HD ##### ## LOC1
LIT   ## ## ##### ## LOC1
MED   ## HD ##### ## LOC1
MED   ## ## ##### ## LOC1
MUS   ## HD ##### ## LOC1
```

tab37_booking_pickup

Location of the table: tab directory of the **ADM** library

Purpose of the table: Booking Requests pickup list configuration

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31

Tab37_booking_pickup defines the drop-down list of pickup locations for Booking requests (dependent on item status, item process status, patron status and sublibrary of item).

These delivery locations ***must be*** valid sublibraries in tab_sub_library.<lng>.

If the item sublibrary, item status, item process status, patron status and pickup location are not defined in this table, the item cannot be requested.

You can define up to 1000 different pickup locations for each combination. Up to ten different locations can be entered on each line. If there is more than one line per combination (to allow for more than ten pickup locations), leave columns 1-4 empty in the other lines.

Structure of the table:

Col. 1 Sublibrary of the item

Col. 2 Status of item being requested

Col. 3 Process Status of item being requested

Col. 4 Item material

Col. 5 Status of the patron

Col. 6-15

Pickup location. Location to which item will be sent

Example of the table:

```
! 1   2  3    4    5   6     7     8     9    10    11    12    13    14    15
!!!!!-!!-!!-!!!!!-!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!
HIL   ## ## ##### ## HILR  WID   LAW   MUS   HIL
!HYL   ## HD ## Y HYL
HYL   ## ## ##### ## HYL   WID
LAM   ## ## ##### ## HIL
LAW   ## HD ##### ## LAW
LAW   ## ## ##### ## LAW
LIT   ## HD ##### ## LIT
LIT   ## ## ##### ## LIT
MED   ## HD ##### ## MED
MED   ## ## ##### ## MED
MUS   ## HD ##### ## MUS
```

tab37_campus_filter

Location of the table: tab directory of the **ADM** library

Purpose of the table: Hold requests filter table

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31

This table is used in conjunction with p-cir-24, the hold request router program. Set this table up to remove holds from consideration that have certain characteristics.

For example, the following line will not allow you to recall a ULAW book from a faculty member (03).

ULAW ## ## ## 03 N N

The table is read from the top down, so put specific lines at the top and the more general lines at the bottom.

Structure of the table:

Col. 1 Sublibrary of the requested item
Col. 2 Status of the requested item
Col. 3 Processing status of the requested item
Col. 4 Patron status of the person who is requesting the item
Col. 5 Patron status of the person who currently has the item out
Col. 6 Allow hold (Y/N)
Col. 7 Allow recall (Y/N)

Example of the table:

```
! 1   2   3  4 5  6 7
!!!!!-!!-!!-!!-!!-!-!
##### ## ## ## ## Y N
UARCV ## ## ## ## Y N
UELEC ## ## ## ## Y N
ULAW  ## ## ## 03 N N
ULAW  ## ## ## 02 N N
UGDOC ## ## ## ## N N
##### ## ## 11 16 Y N
```

tab37_campus_preferred

Location of the table: tab directory of the **ADM** library

Purpose of the table: Preferred supplier list for hold requests

Related table(s):

1) tab_sub_library.<lng>

This table is used in conjunction with the holds request router job, p-cir-24. and should be used to determine the preferred supplier list for items being requested.

Column 1 defines the pickup location for the hold. So, for example, if the patron wants to pick up an item near his home at campus A (CMPA) the following line indicates that the system should try to supply this from CMPA, then CMPA2, etc, and through the list.

CMPA CMPA CMPA2 CMPA3 CMPB1

If no item is available from the local list, the system will try to supply it from the global list:

CMPA CMPB CMPC CMPD CMPE CMPF CMPG CMPF CMPH
CMPI CMPJ

Not that the above lines are concatenated.

Structure of the table:

Col 1 Pickup sublibrary of the hold group
Col 2-10 Preferred provider for the request

Example of the table:

```
! 1     2     3     4     5     6     7    8      9    10
!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!-
ILL01 UELEC UARCV
!EXL:2222
ILL02 ULAW
##### URLEC UEDUD UHLTH ULINC URDOC RLINK
ULAW  ULAW  ULINC UHLTH USCI  UEDUC
USCI  UHLTH USCI  UELEC ULINC
UEDUC UEDUC UUPTN ULINC USCI
##### UARCV ULAW UELEC UEDUC UGDOC
```

tab38

Location of the table: tab directory of the **ADM** library

Purpose of the table: configuration for photocopy requests

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31
4) tab18.<lng>

Tab38 defines the configuration for photocopy requests. This configuration includes:
target library for photocopy request
item status
patron status
pickup library
delivery method

NOTE: if no combination is defined for an item/patron, the photocopy request will not be allowed.
Note too that ***HOME*** is equal to *delivery*; a photocopy letter only will be printed
All other sublibrary codes are pickup. A photocopy letter and a pickup notice are printed. For this reason it is not possible to define HOME and a sublibrary code in the same line.

Note too that it is possible to charge a different fee for home delivery (cash transaction 0008 – tab18.<lng>).

Structure of the table:

Col. 1 source sublibrary
Col. 2 source item status
Col. 3 patron status
Cols. 4-13
pickup location

Example of the table:

```
! 1    2  3     4 … … …
UGDOC ## ## UGDOC
UEDUC ## ## UEDUC
UHLTH ## ## UHLTH
ULAW  ## ## ULAW
ULINC ## ## ULINC
ULINC ## ## HOME
UMUSI ## ## UMUSI
```

NOTE that in tab_sub_library.<lng>, it is possible to define different *library types:*
1 - full sublibrary (patron and items)
2 - patron sublibrary only (no items)
3 - special
4 - reading room sublibrary (1 + special request handling)
5 – order units
For the purpose of tab38, it is possible, for example, to define different *mailboxes* which need to *also* be defined in tab_sub_library.<lng>:

for example, from tab_sub_library.<lng>:

```
library – type
BOX1      3    USM50 L Pickup in Forum
```

Where *Box1* is the "library", and the library type is 3=special

tab39

Location of the table: tab directory of the **ADM** library

Purpose of the table: Hold request daemon configuration

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab40.<lng>

The tab39 table is used in conjunction with the hold request daemon (ue_06 – UTIL E/6). For this purpose tab39 specifies several parameters for processing hold requests. One of these parameters is a suffix or extension that is attached to each print file that is produced as a result of hold requests coming from the Circulation module and the Web OPAC. The suffix is identified by the print daemon which directs it to a specified printer.

Up to 1000 lines may be defined.

Structure of the table:

Col. 1 item sublibrary (##### for all sub libraries)
Col. 2 item status (## for all statuses)
Col. 3 Item process status (## for all item statuses);

Col. 4 collection (##### for all collections)

Col. 5 Call number range. This is the lower limit of the range. Leave empty if you want to limit the range only on the upper limit.

Note: if this field is filled in and the field in column 6 is empty, the request daemon ue_06 will only retrieve the item(s) with this exact call number

Col. 6 Call number range. This is the upper limit of the range. Leave empty if you want to limit the range only on the lower limit.

Col. 7 Print hold wait letters?

This switch determines whether "hold request not filled" letters (hold-wait-letter.xsl) will be printed or not.

Col. 8 How many hold slips?

This switch determines how many hold request slips (hold-request-slip.xsl) will be printed.

Col. 9 Format number of hold slip template (hold-request-slip.xsl)

Col. 10 Print ID. Hold slip Print ID; that is, the extension that allows the Print Daemon to identify the file [Note: Print ID must be entered in lowercase.]

Col. 11 Format number of "hold request not filled" letter template (hold-request-wait.xsl)

Col. 12 Print ID. Hold request not filled Print ID; that is, the extension that allows the Print Daemon to identify the file [Note: Print ID must be entered in lowercase.]

Col. 13 Request type

Filter daemon according to the type of request :

- R - Rush request
- N - Normal request
- blank - all requests

Example of the table:

```
##### ## DP #####

                    Y 1 00 depo.slip  01 depo.wait
!##### ## ## ##### 2000
                    7000
                     Y 1 00 holdslip   01 holdwait
##### ## ## #####

                    Y 1 00 holdslip   00 holdwait
```

tab40.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Collection codes and names

Related table(s):

1) tab_sub_library.<lng>

This table defines the *expansion* of the collection code as entered in the Z-30-COLLECTION FIELD in items, to the expanded form to be displayed in the OPAC.

Structure of the table:

Col. 1 Collection code as entered in the Z30-COLLECTION field of the item record.

Col. 2 Sublibrary (may use #)

Col. 3 ALPHA

Col. 4 Collection name as will display in the OPAC (maximum 80 characters)

Example of the table:

```
! 1     2   3     4
!!!!!-!!!!!-!-!!!!!!!!!!!!!!!!!!!!!>
ABST  ULINC L Abstracts
ASIAN ULINC L Asian Collection
AV    ULINC L Audio-Visual
GEN   ULINC L General
GRANT ULINC L Grant Collection
LEE   ULINC L Lee Collection
SER   ULINC L Serials/Microforms
MICR  ULINC L Microforms
REF   ULINC L Reference
RESV  ULINC L Reserves
```

tab41

Location of the table: tab directory of the **ADM** library

Purpose of the table: Photocopy request daemon configuration

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab40.<lng>

The tab41 table is used in conjunction with the photocopy request daemon (ue_06 – UTIL E/6). For this purpose tab41 specifies several parameters for processing photocopy requests. One of these parameters is a suffix or extension that is attached to each print file that is produced as a result of hold requests coming from the Circulation module and the Web OPAC. The suffix is identified by the print daemon which directs it to a specified printer.

Structure of the table:

Col. 1 item sublibrary (##### for all sub libraries)

Col. 2 item status (## for all statuses)

Col. 3 item process status. (## for all item statuses). Leaving this column empty means 'No Process Status'.

Col. 4 collection (##### for all collections)

Col. 5 Call number range. This is the lower limit of the range. Leave empty if you want to limit the range only on the upper limit.

Note: if this field is filled in and the field in column 6 is empty, the request daemon ue_06 will only retrieve the item(s) with this exact call number

Col. 6 Call number range. This is the upper limit of the range. Fill in this column only if you want a range of call numbers.

Col. 7 Print photocopy wait letters? (Y/N) determines whether "photocopy request not filled" letters (photo-wait-letter.xsl) will be printed or not.

Col. 8 Print photocopy slips? (Y/N) determines whether photocopy request slips (photo-request-slip.xsl) will be printed or not.

Col. 9 Number of slips - how many copies of each photocopy request slip will be printed.

Col. 10 Format number of photocopy slip template (photo-request-slip.xsl)

Col. 11 Photocopy slip Print ID - the extension that allows the Print Daemon to identify the file [Note: Print ID must be entered in lowercase.]

Col. 12 Format number of "photocopy request not filled" letter template (photo-request-wait.xsl)

Col. 13 Photocopy request not filled Print ID – the extension that allows the Print Daemon to identify the file [Note: Print ID must be entered in lowercase.]

Col. 14 Format number of "photocopy request filled" letter template (photo-request-letter-p.xsl or photo-request-letter-d.xsl)

Col. 15 Photocopy request filled Print ID – the extension that allows the Print Daemon to identify the file [Note: Print ID must be entered in lowercase.]

Example of the table:

```
##### ## #####

                Y Y 1 00 photoslip  00 photowait   00 photofilled
```

Tab_req_preferred

Location of the table: tab directory of the ADM library

Purpose of the table: preference order of material for ue_06 handling of hold and photocopy requests.

Related table(s):

4) Tab_req_preferred_loc

tab_req_preferred is used by the ue_06 request handling daemon to determine the order of preferred items that should be used to fulfill the request.

Items of the requested title that are preferred according to tab_req_preferred will be checked:

- If they match the items enumeration and chronology values of the item the request was made on?
- Are they available?

The order in which the items are checked is according to the order of the tab_req_preferred lines. The most preferred items are checked first.

The first available preferred item that is found is used – this is the item that is printed on the slip.

In case of hold request, the request moves to this item.

In case of photocopy request, PHOTO-USE-EXACT is used in order for the request to move to this item.

If no preferred item is available, the request is filled by the existing Aleph request fulfillment mechanisms.

Note:

The preferred items mechanism should be used only when PHOTO-REQUEST-STYLE is 3.

Structure of the table:

Col. 1	request type (H – hold requests, P – photocopy requests, # - both hold and photocopy requests)
Col. 2	item sublibrary (##### for all sub libraries)
Col. 3	item status (## for all statuses)
Col. 4	item process status. (## for all item statuses). Leaving this column empty means 'No Process Status'.
Col. 5	collection (##### for all collections)

Example of the table:

```
!  2    3  4   5
!-!!!!!-!!-!!-!!!!!
# LAW   01 ## GEN
# LAW   01 ## NBS
# LAW   ## ## #####
# HYL   01 ## GEN
# HYL   01 ## NBS
```

```
# HYL   01 ## #####
```

In the above example, items of the LAW sublibrary with status 01 from the GEN collection are the most preferred. If these items are not available, items of the LAW sublibrary with status 01 from the NBS collection are used. If they are also not available, then the rest of LAW items is considered.

Only if all LAW similar items are not available are HYL items considered.

Note that in the above example, the definitions are the same for both hold requests and photocopy requests. It is possible to define a different setup for hold and photocopy requests by setting column 1 accordingly.

Tab_req_preferred_loc

Location of the table: tab directory of the ADM library

Purpose of the table: pickup locations for preferred items mechanism

Related table(s):

5) Tab_req_preferred

The preferred items mechanism are applied by the request handling daemon (UE_06) only for requests that have a pickup location that is listed in tab_req_preferred_loc.

Note:
If the table does not exist or is empty, the preferred item mechanism is applied for any pickup location.

Structure of the table:
Col. 1 pickup location

Example of the table:

```
!  1
!!!!!
HOME
LAW
```

In the above example, the preferred items mechanism are applied to requests that are requested for pickup at HOME or LAW pickup locations.

tab42

Location of the table: tab directory of the **ADM** library

Purpose of the table: Automatic update of the processing status field in the item record.

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>

This table controls automatic update of the processing status field in the item record. Automatic update is triggered by various actions or changes in the Acquisitions, Serials and Items modules, and dependent on sublibrary and action.
If the action trigger is listed in this table, the item processing status will be updated as defined.

ACQUISITIONS

Changing the order status acts as the trigger. The order statuses are:
NEW - Newly created order.
WP - Waiting for processing.
PS - Processing started.
WB - Waiting for budget confirmation.
QSV - Query before sending order.
CNB - Cancelled, no budget.
DNB - Delayed, no budget.
RSV - Ready to send to vendor.
SV - Order sent to vendor.
SV+ - This is not an order status. It is the "action" when material arrival is registered.
VC - Vendor cancellation of order.
LC - Library cancellation of order.
CLS - Closing of order.

SERIALS

OP - Item opened
CL - Item claimed
AR - Item arrival registered
NP - Item not published (this is not a trigger, it is a manually entered processing status.
NO - Item opened, when Z16 is not active
UAR - Unarrive item

ITEMS

BD - Binding (occurs when issues are processed for binding)
PL - Print label utility
DP - Remote Storage functionality

Note: The process status of the new bound issue, will be taken automatically from tab42, section **ITEM** and status **BD**. If the line doesn't exist in tab42, then the process status will be taken from the manual input of the user.

Structure of the table:

Col. 1 Module

- ACQ = Acquisitions
- SERIAL= Serials
- ITEM= Items

Col. 2 Sublibrary (Use # for wildcard)

Col. 3 Action code that will trigger the change of the item processing

status. Use **DP** as the trigger for an ITEM when you want the processing status updated because the item has a remote storage-id.

Col. 4 The processing status code. Enter blanks to remove the processing status code. The processing status codes are not system controlled, but the following has functionality in the system:

- **BD** (bound item - does not display in list of items)

Col 5 Previous Item processing status:

Only applicable when column 1. is **ACQ**. The current Item Process Status is checked, and the process status will be changed only if there is a match.

This can be used in order to ensure that a Process Status entered manually will be retained.

Use "##" to indicate that the current Item Process Status is irrelevant, and the change is to be made in all cases.

Example of the table:

```
! 1          2   3   4  5
!!!!!!!!!!-!!!!!-!!!-!!-!!
ACQ         ULINC NEW OR ##
ACQ         ULINC RSV OR ##
ACQ         ULINC SV  OI ##
ACQ         ULINC CLS    ##
ACQ         ##### NEW OI ##
ACQ         ##### WP  OI ##
ACQ         ##### PS  OI ##
…
SERIAL      ##### UAR NA ##
SERIAL      ##### NP  NP ##
SERIAL      ##### OC  NP ##
…
ITEM        ##### PL     ##
ITEM        ##### BD  SB ##
```

tab43

Location of the table: tab directory of the **ADM** library

Purpose of the table: Advanced booking schedule

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>

Tab43 is an expanded table for defining advanced booking parameters. The table allows for the definition of start day and hour, as well as end day and hour.
The table allows for defining "closed" times, independent of tab17 (library hours).

It is possible to define days where the slots are different from the schedule as a whole. To do so, use the E line type to define Exact day.

The definition can be of a slot different from the one already set for the schedule period.

For example:
The regular slot is:

P ##### 60 20040101 01 0900 01 1100;

The P stands for settings of a period and will include the date until which the definition is valid.

Note: In order to see periodic slots for days before the E lines, the P lines must come before the E lines.

The settings for the irregular slot are:

E ##### 60 20030930 00 0900 00 1700

- The E indicates that the following line is a definition of an exact date and not an end of period as in lines of type P.
- The 00 in an E type of line, indicates that the definition is for the date listed in column 4.
- 00 in a P type of line still stands for Sunday.

The slot in this example starts on 09:00 and ends at 17:00 on the same day.

It can also be defined to end the next day in the following way:

E ##### 60 20030930 00 0900 01 0900;

In this case the slot starts at 09:00 on the current day and ends at 09:00 one day after.

Another option is to define a date that is included in the schedule period and on which no slots will be open.
To do so, leave columns 5-8 empty in the following way:

E ##### 60 20030930

This indicates that on September 30th, there are no slots available for item status 60.
To define two close days in a row, set a line for each day.

It is possible to use ## instead of the year, month or day listed in column 4 of the E type of lines.
For example:

E ##### 60 2003##01

Indicates that there are no open time slots for the 1st of every month on 2003.

E ##### 60 ####0930

Indicates that there are no open time slots on September 30th of each year.
It is NOT possible to use ## as part of the date in the P type of lines.

Note: There is a limit of 100 time slot lines (columns 5 to 8) for each group of columns 1 to 4 (Line Type, Sublibrary, Item Status, and End of Period Date).

Structure of the table:

Col. 1 Line Type:

- P - Slots for period date
- E - Slots for exact date

Col. 2 Sublibrary (use #### for wildcards)

Col. 3 Item Status

Col. 4 End Period Date

Col. 5 Beginning of time slot - Day (00=Sunday, 06= Saturday)

Col. 6 Beginning of time slot - Hour

Col. 7 End of time slot (00-10) - Day (number of days that have elapsed since Sunday)

Col. 8 End of time slot - Hour

NOTE: Every row should end with a semi-colon (;). See following example.

Note that the number of advanced booking days that will *display* in the Web OPAC (that is, the number of days ahead that a person may book an item), is defined in www_server.conf – setenv www_item_schedule.

Example of the table:

```
!   2     3     4   5   6   7   8
!-!!!!!-!!-!!!!!!!!-!!-!!!!-!!-!!!!
P ##### 60 20040101 01 0900 01 1100;
                   01 1100 01 1300;
                   01 1300 01 1500;
                   01 1500 01 1700;
                   01 1700 01 1900;
                   01 1900 02 0900;
                   02 0900 02 1100;
                   02 1100 02 1300;
                   02 1300 02 1500;
                   02 1500 02 1700;
…
E ##### 60 20031015
```

tab44

Location of the table: tab directory of the **ADM** library

Purpose of the table: Defines the period that requested items will be held on the hold shelf

Related table(s):

1) tab_sub_library.<lng>
2) tab15.<lng>
3) tab31

Tab44 allows for defining the period for which requested items will be kept on the hold shelf. A differentiation may be made depending on whether the hold is triggered by the **return** of the item, or triggered by the Patron Request utility "**Print Letter -**

Hold Request Filled". Many libraries use the utility for closed stack management and want a shorter hold period for items in closed stacks than for other items. Two date options can therefore be defined:

Date 1- ***Return***: hold request status is changed to "S" through the return screen
Date 2- ***Utility***: hold request status is changed to "S" through requests/print letter hold request filled

Structure of the table:

Col. 1 Sublibrary
Col. 2 Item Status
Col. 3 Patron status (use # for wildcard)
Col. 4 Date operator -1
(D=day, M=month, W=week)
Date 1 - Item is put on hold using the Return functionality
Col. 5 Date parameter -1
Number of days/weeks/months to add
Col. 6 Date operator - 2
(D=day, M=month, W=week)
Date 2 - Item is put on hold using the Print letter-Hold request filled functionality
Col. 7 Date parameter -2
Number of days/weeks/months to add

Example of the table:

```
! 1   2  3  4  5  6  7
!!!!!-!!-!!-!-!!!-!-!!!
##### 11 ## D 004 D 004
##### 21 ## D 006 D 006
##### 31 ## D 007 D 007
##### 41 ## D 007 D 007
##### 12 ## D 008 D 008
##### 22 ## D 010 D 010
##### 32 ## W 002 W 002
##### 42 ## W 002 W 002
…
```

tab45.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Order transaction types for the order log

The ***order log***, is part of the online Acquisitions**.** It is dependent on definitions defined in tab45. An entry is made in the log when an order is created, when the order status

or item status is created or changed, when the next claim date changes, or when the user manually adds his own log notes.

Definitions in tab45 sets whether the transaction creates an order log record or not, and defines the description for the action (which is written in the order log text field).

Note: It is recommended to set Col. 3 (Order Log can/cannot be entered manually) to Y only for the following transactions:
00 – General
13 - General Invoice Note
95 - Subscription history note
98 - Reply from vendor (no claim)
99 - Note to vendor
If Y is set for other transaction types, the log record is created, the text that is entered is written on the Oracle table, but it does not display correctly in the GUI Order Log window.

Structure of the table:

col. 1	Transaction number
col. 2	Y/N • Y=activate logger • N=do not activate
col. 3	Order Log • Y = Order Log can be entered manually • N = Order log cannot be entered manually
col. 4	ALPHA plus function description

Example of the table:

```
!
!1 2 3            4
!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

00 Y Y LGeneral note
01 Y N LOrder status
02 Y N LNext claim date
03 Y N LOrder created
04 Y N LItem(s) created
05 Y N LItem process status
06 Y N LEstimated price
08 Y N LArrival
09 Y N LBudget
10 Y N LInvoice
11 Y N LNo budget
```

tab46.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: ILL request log transaction types

Tab46 defines whether
a transaction creates an ILL request log record,
the transaction is system generated, or can be entered manually.

The ILL request log is part of the online ILL module. It depends on definitions defined in tab46. For example, an entry can be made
when an ILL Outgoing request (Z410) is created,
when a supplier request is created (Z411),
when Incoming ILL request is created (Z416)
or when the user manually adds his own log notes.

Note: It is recommended to set Col. 3 (Request Log can/cannot be entered manually) to Y only for the following three transactions:
00 - General
20 - Message to Supplier
21 - Supplier Response

If Y is set for other transaction types, the log record is created, the text that is entered is written on the Oracle table, but it does not display correctly in the GUI Request Log window.

Structure of the table:

col. 1 Transaction number

col. 2 Active/non active log

- Y=activate logger
- N=do not activate

col. 3 Manually log

- Y = ILL Log can be entered manually
- N = ILL log can not be entered manually

col. 4 ALPHA +function description

Example of the table:

```
!1 2 3                 4
!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
00 Y Y LGeneral
01 Y N LSupplier Request Status Changed
02 Y N LILL request created
03 Y N LSupplier request created
04 Y N LVouchers allocation
05 Y N LVouchers de-allocation
06 Y N LLetter was sent to the patron
```

tab47.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: ILL Borrowing - Patron's copyright agreement policy

Tab47 determines the default copyright policy for the patron's copyright agreement and copyright block.This table allows for the coded definition of materials requested by the library. The coded definitions can be supplier-specific.

For each combination of
- supplier code (Col.1) and
- requested material type (Col.4),

there is a definition as
- to whether the patron's copyright cgreement is required (Col.2)
- and whether to block the dispatch of the request to the supplier if the copyright agreement has not been signed (Col.3).

The borrowing request media code (Col.4) must be one of the following:
- PRINTED,
- COPY,
- MICROFORM,
- VIDEO,
- AUDIO,
- MR,
- OTHER

with one of the following prefixes L-, C- or E-.
For example: L-PRINTED, L-VIDEO, E-MR, C-PRINTED.

Structure of the table:

col. 1	Supplier code – use **#** as a wildcard
col. 2	Copyright agreement needed – Y/N
col. 3	Block request – Y/N The request will be blocked if there is no copyright agreement.
col. 4	ILL material type
col. 5	ILL material description

Note the material type **L-NORETURN.**
It is treated by the system in the same way as any other loan type (L-) except that the expected return date will be always set to 31-Dec-2099.

This allows for the creation of a Loan (Z36) and Item (Z30) for material that does not need to be returned to the ILL supplier. In those cases where it is important for the library to create an item and a loan record, this option will be useful.

Example of the table:

```
!  1                  2 3       4                 5
!!!!!!!!!!!!!!!!!!!!-!-!-!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!…!!!!!
ILL-TEST              N N L-BOOK               Book (loan)
ILL-TEST              N N C-BOOK               Book (photo)
ILL-TEST              N N L-SERIAL             Serial (loan)
ILL-TEST              Y N C-SERIAL             Serial (photo)
ILL-TEST              N N L-CONFERENCE         Conference (loan)
ILL-TEST              Y N C-CONFERENCE         Conference (photo)
…
ILL-TEST              N N L-NORETURN           No return media
…
##                    N N L-BOOK               Book (loan)
##                    Y N C-BOOK               Book (photo)
```

tab48.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: General invoice payment status

Related table(s):

1) user_function.<lng>

This table allows for a more flexible way of defining the Z77-P-STATUS in relation to the invoice approval process, checks, etc.

1. Payment status
It defines all statuses for Z77-P-STATUS. It also defines the default payment status when the general invoice is created.

2. Freeze on invoice
Defines in which statuses the invoice is frozen. This means that all operations except change in status are not allowed.

3. Permissions
The table also defines which user_function routine should be used over and above the standard checks for different statuses. To accommodate this, the following changes have been made to the table user_function.<lng> .

ACQ/INVOICE-HEAD-P-S-R
Update "Ready to be paid" general invoice and/or its line invoices.

ACQ/INVOICE-HEAD-P-S-Y
Update "Payment authorization given" general invoice and/or its line invoices.
ACQ/INVOICE-HEAD-P-S-P
Update "Paid" general invoice and/or its line invoices.

This replaces the old line:
ACQ/INVOICE-HEAD-UNPAID
Update "PAID" general invoice and/or its line invoices.

Structure of the table:

Col. 1 Invoice payment status.

Note that a line with status "P" (Paid) **MUST** exist in this table. All other statuses can be library defined.

Col. 2 Default

Defines if the status of Col.1 will be the default value for a new general invoice

- Y = Default
- N = No default

When creating a new invoice, the system opens the invoice with the payment status as blanks. Refresh/Update replaces it with the assigned default. For a single line item invoice, the default value is taken automatically. There is no Z77-P-STATUS in the single line invoice form window.

Note: Only one line in this table should have the value "Y";

Col. 3 Freeze invoice

Freeze all related invoice records for this payment status. Records cannot be updated.

- Y = Freeze
- N = Do not freeze

Col. 4 user_function authorization to use for this status, in addition to the standard checks. If left blank, no additional check is made, and authorization for ACQ Update Invoice Payment Status suffices. The code in this line does not have to be the same as the in column 1. It is used only to identify the user authorization function line in the user_function table. Therefore, multiple invoice payment statuses can use the same authorization. Note! in addition to preventing the operator from changing the invoice 'from' the given pay-status, the lack of the specified permission prevents the operator from changing the invoice 'to' that status

Col. 5 Payment status text

Example of the table:

```
! 2 3          4                        5
!-!-!-!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
N Y N                      Not ready to be paid
R N N INVOICE-HEAD-P-S-R   Ready to be paid
Y N Y INVOICE-HEAD-P-S-Y   Payment authorization given
P N Y INVOICE-HEAD-P-S-P   Paid
U N N INVOICE-HEAD-P-S-Y   User invoice
```

tab50

Location of the table: tab directory of the **ILL** library

Purpose of the table: Determining automatic processing for outgoing ILL requests

In order to activate tab50, you must activate the background job: "ILL Automatic Processing (ue-19)". The daemon will use the table for processing both ISO ILL requests as well as non ISO requests.

The table definines how the "ILL Automatic Processing (ue-19)" daemon will manage outgoing ILL requests. An action can be set for each combination of:

- ILL Unit that manages the outgoing request.
- The potential supplier
- The requsted media
- The request status
- The sequence of the active potential supplier in the request's list of potential suppliers

For each such combination, one of two actions may be set:

- SEND – If no list of potential suppliers exist, create one using the Locate action. If a list exists already, send the request to the currently active potential supplier.
- CANCEL – Send a Cancel notice to the currently active potential supplier.

Structure of the table:

Col. 1	ILL Unit
Col. 2	Active potential supplier code
Col. 3	Requested media.
Col. 4	Supplier request status (Z411-STATUS)
Col. 5	Sequence of the active potential supplier in the request's list of potential suppliers.
Col. 6	Action to carry out.

Example of the table:

```
!1                    2                     3                     4   5         6
!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!-!!!-!!!!!!!!!-!!!!!!!!!
ILL_LAW               ILL_MED               L-PRINTED             NEW 000000001 SEND
#################### #################### #################### CRP 000000001 CANCEL
```

tab100

Location of the table: tab directory of the library

Purpose of the table: Switch settings for different aspects of the system

Related table(s):

1) tab_sub_library.<lng> - tab100 USE-ORDER-UNIT
2) tab_own - tab100 OWN-FILTER
3) tab31 - tab100 CHECK-BOR-DEFAULTS
4) tab37 - tab100 HOLD-BY-ITEM-GROUP
5) tab18.<lng> – tab100 OVERDUE-RECALL-RATIO
6) tab_z311 - tab100 INVENTORY-NUMBER
7) edit_doc_999.<lng> - tab100 ITMG-DISPLAY
8) pc_tab_exp_field.<lng> section USER_ADDRESS_TYPE for switch DEFAULT-ADDRESS
9) www_f_heading - message no. 9982 for switch BROWSE-NO-MATCH
10) tab_bor_address – tab100 DEFAULT-ADDRESS
11) column 1 of tab_bor_id.<lng> - tab100 ADDITIONAL-ID-TYPE

It is possible to create tab100 or tab100_<server_type> under $alephe_tab or $data_tab of a library. For example, there can be an extension _pc or _www, in order to specify different values for pc_server and for www_server. If values are common, they can be entered in the tab100 that does not have extension.
When a program calls tab100 of a certain library/server type, the following is the reading order of tab100:

1) initialization with default values from TAB100.source (always)
2) reading $alephe_tab/tab100
3) reading $alephe_tab/tab100_<server_type>
4) reading $data_tab/tab100
5) reading $data_tab/tab100_<server_type>

At each stage (from 2 to 5), if tab100 exists, the values defined in it **override** the values from previous readings, for example,
if in $alephe_tab/tab100, ADDRESS-ZIP-STYLE=2,
and in $data_tab/tab100 of USM50, ADDRESS-ZIP-STYLE=3,
the final value of ADDRESS-ZIP-STYLE will be "3".

The names of the available variables, their types (text or number), their maximum length, their default value and permitted values are defined in ../alephm/source/copy/TAB100.source, and CANNOT be changed by the user.

Lines beginning with an asterisk ("*") contain category names, which are not part of tab100; they are used only for display purposes when ./alephm/source/copy/TAB100 and the tab100 header are created automatically out of TAB100.source.
Note the following:
A tab100 variable name can contain no more than 23 characters - (Column 1 of TAB100.source).
Column 2, Variable Maximum Length, is a two-digit number from 01 to 99 for textual variables. If the variable is a number, column 2 is set to "N ".
All numbers in tab100 can be up to 9 digits
Column 3, Permitted in tab100 of library - Y/N, indicates whether the variable can be defined in tab100 located under the $data_tab directory of a library.
Column 4, Permitted in tab100_<server_type> - Y/N, indicates whether the variable can be defined in tab100_<server_type> (for example, tab100_pc) located under $alephe_tab, or tab100_<server_type> located under the $data_tab directory of a library. This column has been set to "N" for all variables transferred from tab10.
Column 5, Default Value, is maximum 20 characters long; it is a string or a number with which the variable is initialized whenever tab100 is loaded, BEFORE reading the settings of tab100 under $alephe_tab or $data_tab.
Column 6, Permitted Values, is a space-delimited list of all possible values for the variable; for numeric variables, a range of numbers is provided, for example, 10-90.

The values of tab100 can be displayed via UTIL G/4/100. This option is particularly useful for two reasons:

1) ALL tab100 variables (as defined in TAB100.source) are shown together with their values, regardless of whether they have been set in the specific tab100; unset variables are given a default value.
2) Full validation is performed on all tab100 values before display; The checks are::
    a) the tab100 line contains a "=" sign (for example, CREATE-Z30H=Y); a line without a "=" sign is ignored.
    b) the tab100 variable is allowed for the specific tab100 that is being loaded for example, variables which are not allowed for tab100 of $data_tab must NOT be set there, only in tab100 of $alephe_tab.
    c)the value of a variable does not exceed its maximum length.
    d) the value of a variable receives one of the permitted values only (if defined).
    e) numeric variables indeed receive a numeric value, which is also in the permitted range (if defined).

Whenever an invalid value is encountered, an error message is displayed and the value is replaced with the predefined default value, for example,

Note that an asterisk * identifies lines that have limited special use.

ACQUISITIONS

BUDGET-PER-ORD-UNIT
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Budgets will be listed only if they are in the user's permitted ordering units. The regular GUI subbrary filter will not be used.
N = Budgets will be listed according to the regulat GUI filter.

CALC-INV-LINE
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = If the balance between the Total Amount Line Items (Z75) and the Total Amount of General Invoice (Z77) is less than 0.05 then the balance is distributed among the preceding invoice line-items.
N = The balance between the Total Amount Line Items (Z75) will not be recalculated.

CHECK-INVOICE-CURRENCY
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = currency on invoice must match one of the vendor currencies. At least one currency must be entered in the vendor record**.**

CHECK-ORDER-BUDGET
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = for "P" (purchase) orders, does not allow 0.00 in the estimated price of order, when the order is sent, and requires a budget encumbrance. Performs budget checks (validity and balance) and changes the order status to DNB if errors are detected.
N = "P" order is allowed 0.00 in estimated price, and budget assignment is optional. If a budget is assigned, when the order is sent -- budget checks are performed, errors are reported, but the order status is not changed to DNB, and the order is sent. If no budget was assigned to the order, an invoice line item can be registered without a budget.

CHECK-ORDER-ISBN-ISSN
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = ISBN/ISSN entered on the Acquisition order is checked for validity

CHECK-ORDER-PERMISSION

Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Check whether the staff user is allowed to see the order information according to the order sublibrary/order unit permissions (either when seeing an order or an invoice line item).
N = No check is done, the order information is always displayed.

CREATE-ITM-FORM-ORDER-M
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = check box for automatic item creation (on tab 3 of order form) is set to "checked".

CLS-ORDER-UPON-ARRIVE
Values:0 1 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = Upon completion of monograph or standing order arrival, the order status is not automatically changed to Closed. Automatic closure of the order occurs when the order is marked as Completely Arrived, Completed Invoiced, or Paid.

1 = Upon reporting a complete arrival of a monograph or standing order, the order status is automatically changed to Closed. This occurs regardless of the order's invoice information.

DEFAULT-GEN-INV-TYPE
Values: Default:
Type: Text; Max Length: 03
tab100 of library: Yes; tab100_<server_type>: No.

Default value for Type of a new General Invoice being produced.

DEFAULT-GEN-INV-STATUS
Values: Default:
Type: Text; Max Length: 03
tab100 of library: Yes; tab100_<server_type>: No.

Default value for Status of a new General Invoice being produced.

EDI-OUT-LIBRARY-NOTE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = (Default) The note will be added to an outgoing EDI message when such a message is sent to a vendor.
N = The note will not be sent to the vendor.

EDI-OUT-PRICE-NOTE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = (Default) The note will be added to an outgoing EDI message when such a message is sent to a vendor.
N = The note will not be sent to the vendor.

EDI-OUT-QUANTITY-NOTE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = (Default) The note will be added to an outgoing EDI message when such a message is sent to a vendor.
N = The note will not be sent to the vendor.

EDI-OUT-VENDOR-NOTE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = (Default) The note will be added to an outgoing EDI message when such a message is sent to a vendor.
N = The note will not be sent to the vendor.

ISSN-CODE
Values: Default: ISSN
Type: Text; Max Length: 20
tab100 of library: Yes; tab100_<server_type>: No.

This code is used to build the search request (for example, "ISSN=12345") when searching for an issue by its ISSN code via the Serials search Bar in the Acquisitions GUI. The code should match column 5 of tab11_ind of fields containing an ISSN code (for example, field 022## in MARC 21). The default value is "ISSN". Maximum length is 20 characters.

ORDER-ISBN-ISSN-ADD-TAG
Values: Default:
Type: Text; Max Length: 06
tab100 of library: Yes; tab100_<server_type>: No.

Up to now Z68-ISBN was populated from the BIB record using the standard MARC codes (dependant on TAB100-MARC-TYPE, for example 020 $$a or 022 $$a for MARC 21

This parameter allows you to define an additional tag and subfield.
For example:

ORDER-ISBN-ISSN-ADD-TAG=021##a

Will look for tag 021, any indicator and subfield a to populate Z68-ISBN.

OVER-EXP-INCLUDE-ENC
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = Max over expenditure of a budget is checked in relation to the encumbrance. The system checks the max over expenditure against the free balance. The free balance is the total allocation - (paid + unpaid invoices + encumbrances).
N = Max over expenditure of a budget is not checked in relation to the encumbrance but in relation to the expenditure. The system checks the max over expenditure against the actual balance of the budget. The actual balance is the total allocation - paid + unpaid invoices).

SERIAL-ITEM-LABEL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

If value is N (Default) the form serial-item-label is used.

If value is Y, system will use the forms item-issue-label or item-copy-label as already in use in Items or Circulation modules.

TWO-LEVEL-VENDOR
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = The system uses two levels of vendor.
N = Only one level of vendor is managed.

USE-OBJECT-CODE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

N = When registering the Invoice, there is no check against the budget's object code and filling the Invoice's object code is not mandatory. Object code may be used for reporting only.
Y = Object code must be assigned to the Invoice and must match the assigned budget's object codes.

USE-ORDER-UNIT
Values:Y N Default: N

Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This switch determines in which method the system works:
Ordering units or sublibraries.
It concerns the Budget's (Z76) order unit, Vendor's (Z70) order unit, Acq order's (Z68) order unit and user passwords (Z66).
N = each sublibrary is an order unit.
Y = ordering units are defined in tab_sub_library.<lng> using sublibrary type 5.

VENDOR-SHARING
Values:0 1 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = for sites that do not have multi-ADM, or are not sharing a single vendor table in a multi-ADM environment. This is the default if this parameter is not defined.
1 = for sites that are sharing a single vendor table in a multi-ADM environment. The Parameter must be "Y" in order to filter the vendor list to display only vendors that have either a relevant sublevel vendor record, or a relevant sublibrary/order unit defined.

0 = this is the default if this parameter is not defined;

1) when a filter is active on the vendor list, vendors that have not been assigned a sublibrary/order unit are considered as "general use" vendors and therefore display in the list.
2) when vendor is assigned to an order, the vendors that do not have any sublibrary/order unit are considered as "general use" vendors and therefore can be assigned to the order; if the vendor has even one sublibrary/order unit, it must have a sublibrary/order unit that matches the sublibrary/order unit of the order.

1 = this can be useful for sites that have many vendor records that are irrelevant to a particular library. This can happen when the site is sharing a single vendor table for multiple ADM libraries, or the site has done a batch load of general vendor records.

1) when a filter is active on the vendor list, vendors that have not been assigned a relevant sublibrary/order unit are not displayed in the list.
2) when vendor is assigned to an order, the vendor must have same sublibrary/order unit as the order. Vendors that do not have any sublibrary/order unit cannot be used on orders.

VIEW-NON-AUTHORIZED
Values: Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

N = The budget information in the lower pane of the budget list (Balance, Transactions, etc.) is displayed for non-authorized budgets even when the budget list is not filtered for authorized budgets only.

Y = The budget information in the lower pane of the budget list (Balance, Transactions, etc.) is displayed for all budgets listed in the upper pane.

CATALOGING

ADAM-INDEX-CHAR-SET
Values: Default: ISO 8859-1
Type: Text; Max Length: 20
tab100 of library: Yes; tab100_<server_type>: No.

Default character set used by ADAM when creating Indexing objects

ADM-OWN-CHECK
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = OWN in the bibliographic record is used to distinguish between ADM's in a multi-ADM enviroment. Then when pushing the BIB record to an ADM, the current staff user's OWN permission will be checked against the pushed BIB record.
N = Do not do a check on the bibliographic OWN field

CREATE-852-HOL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

*** Applicable only to the HOL library (USM60).**
Y = automatic generation of 852 subfields from call number fields in the BIB record (099, 098, 090, 092, 096, 050, 055, 060, 070, 082, 086)

CREATE-CAT-FIELD
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = "CAT" fields are created.

N = "CAT" fields are not created for records in this library in main flow activities, but might be created by some actions such as moving one record information to another record.

CREATE-SCAN-JOB
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

CREATE-Z00H
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = transfer a deleted BIB record to a history file. This is for statistical purposes only, and does not imply that there is capability to restore.

CREATE-Z00R
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = create a Z00R record for each Z00 record. Suitable for BIB, HOL and AUT libraries, but not for an ADM library.

CREATE-Z106
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This switch determines whether a Z106 record is created automatically each time a cataloging record is created or updated.
N = record updates will not automatically generate Z106 records. In this case, the Z106 records can be created by running the Create/Update Z106 Table for "CAT" Field (p-manage-19) service available from the Catalog Maintenance Procedures option of the Services menu in the Cataloging module.
Y = each time a record is created or updated a Z106 record will be created.

CREATE-Z115
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

N = The system does not create z115 records for this library

DOC-BLANK-CHAR
Values:^ - | Default: ^
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Defines what sign is used to denote a blank in MARC 21 fixed fields. This should not be confused with the fill character |.

FORCE-USE-Z07
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = A Z07 record will be created in the library (for example, ADM or HOL) although the document being updated does NOT belong to it (for example, a Z07 will be created in an ADM library when a record in a BIB library to which it is linked, is updated).

FRENCH-AUT

Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Setup of this variable is only in $alephe_root/tab100. Only for libraries using SUDOC: affects the correction of headings and word indexes.
Y = UE_11 will generate z07 in the Bibliographic library for updated Authority records.

HOL-008-LNG
Values:0 1 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This flag defines how the LNG in the 008 field is determined.

0 - From tab_tag_text without any change
1 - Derived from the BIB record's language based on standard system rules (that is, 008, 041 and so on...).

The flag should be defined in the HOL library. The default is 1.

INDEX-AND-LINK
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = If an article is linked by field 590i to a serial and the call number of the serials item is updated, the index for the call number will be updated for the linked article.

This is only relevant for MAB libraries.

INDEX-DN-LINK
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This flag can be used to suppress creating Z07 for linked (DN) BIB records. The default "Y" indicates that the system writes Z07 records for all DN-linked sons if the father record is changed.

INDEX-ITM-LINK
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

When updating docx, if this var is set to "Y" (while the default is "N") the attached document using ITM links will also be indexed.

MARC-EXP-BLANK-CHAR

!Values:Y N Default: Y
!Type: Text; Max Length: 01
!tab100 of library: Yes; tab100_<server_type>: No.

Determines whether to replace the tab100 variable DOC-BLANK-CHAR by a blank in non-fixed fields when exporting records in MARC format.
Y = the character defined by DOC-BLANK-CHAR will be replaced by a blank
N = the character defined by DOC-BLANK-CHAR will remain as is.

MARC-TYPE
Values:1 2 3 4 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Defines that type of MARC record
(1=USMARC, 2=UNIMARC, 3=DANMARC, 4=MAB)

OWN-FILTER
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = The display filter based on tab_own is activated (only for HOL or BIB)
N = The filter is not active. (Default)

SPLIT-HANGUL-CHARS
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Hangul words are split character by character for word indexing and searching.

N = Hangul words are NOT split, i.e. each cluster of characters separated by a space is indexed. The same logic is applied also for search queries.

UNION-IGNORE-MATCH
Values: Default: deleted,circ-created, suppressed
Type: Text; Max Length: 99
tab100 of library: Yes; tab100_<server_type>: No.

A record will not be found as an equivalent to other records if subfield 'a' of the "STA" field will match one of the values in this variable. More than one value can be defined here, separated by a comma.

USE-ACC-TEXT
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = using CTRL+F3/F4 in cataloging the system copies the content of the chosen GEN heading into the BIB.
N = using CTRL+F3/F4 in cataloging the systems takes the preferred term (MARC = 1XX;MAB = TMP01) from the AUT record and copies it to the BIB.
Default = N; this switch should only be defined for AUT libraries

CIRCULATION

ADDITIONAL-ID-TYPE
Values: Default:
Type: Text; Max Length: 02
tab100 of library: Yes; tab100_<server_type>: No.

This is a two-digit number, representing column 1 of tab_bor_id.lng (ID Code). It is used to populate the Z308-KEY-TYPE of the Additional ID filled in the "Fast Patron Registration" form in the Circulation GUI and in ill_77 (Synchronize Patron Data with Partner Data). If Additional ID is used by a library, this variable MUST have a value, but it CANNOT be set to "00" or "01" - these types are reserved for System ID and Barcode, respectively.

Note that this variable must be defined in tab100 of the usr_library.

AVAILABILITY-ROUTINE
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

1 = item is non-available if it is on loan, or on hold, or can fulfill a request even if the request was placed on another "like item". This is the default if the switch is left blank.
2 = (site specific) item is non-available if it is on loan, or on hold, or this same item has been requested. The item will not be considered non-available if another "like item" has been requested.

BARCODE-DELETE-SPACES
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

***** Y = deletes spaces when barcode data is transferred to the system. This refers to the item barcode only.

BLOCK-RATIO
Values:1 2 3 4 5 6 7 8 9 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Defines the factor by which the system multiplies the late days for a borrower that has returned an item late. This is applicable for fine methods 5, 6, 8, 9 as defined in tab16 (col. 14).

BOOKING-DELIVERY-TIME
Values: Default: M000
Type: Text; Max Length: 04
tab100 of library: Yes; tab100_<server_type>: No
Defines the delivery time that will be added to a booking request that is required for delivery. The value is a three digit number that can express minutes, hours, days or weeks, depending on the variable prefix.

BOR-EXPIRY-DUE-DATE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = sets the due date for loan and renew to the local patron expiration date if the local patron expiration date is earlier than the computed due date.

BOR-STATUS-FOR-RENEWAL
Values:L C Default: L
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

L = when the item is renewed, the patron status at the time of the original loan is used. This is the default if the switch is left blank.
C = when the item is renewed, the patron's current status (at the time of renewal) is used, and the Z36-BOR-STATUS is updated accordingly.

CHANGE-ITM-STS-LOAN

Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Allow change of item status if item is on loan
N = Block change of item status if item is on loan

CHECK-BARCODE
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

***** N = disables (the default) check for barcode on the item record, thereby allowing item records without barcodes. If library opts to disable barcode check for items, the z30_id1 entry in the file_list must be adjusted accordingly, since it assumes that the barcode is an index, and will not allow a non-unique index key ("blank" is an index key).

CHECK-BOR-DEFAULTS
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = when updating the Z305 (local patron record) in the GUI CIRC client, the values set in the record are checked against the values set in tab31 for the patron status. If there is a discrepancy, an overridable warning displays.

CHECK-UNIQUE-NAME-BIRTH
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = in a patron Z303, record the combination of name and birth date must be unique. Umlaut letters are considered to be identical to their non-umlaut value for this purpose. The non-umlaut value depends on the translation of the umlaut in the Unicode table.

CREATE-Z30H
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = create Z30H (ITEM HISTORY) record when item is updated or deleted, if /tab/tab_item_history.<lng> is set for creating history record. There can be multiple Z30H records for a single item; a Z30H record can be re-instated as current record.

CREATE-Z36H
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = transfer completed loan records to a history file

CREATE-Z37H
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = transfer completed hold records to a history file

CREATE-Z38H
Values: Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = transfer completed photocopy records to a history file

DEFAULT-ADDRESS

Values: Default: 01 030
Type: Text; Max Length: 06
tab100 of library: Yes; tab100_<server_type>: No.

Variable format - TT DDD, for example, 01 060; it is used for ADM libraries that do not use tab_bor_address.
TT - Two digits representing the default patron address type (detailed in pc_tab_exp_field.eng, section USER_ADDRESS_TYPE)
DDD - A three-digit number, representing the number of days for which the default address is to be valid after its creation date. Setting this variable to 999 is interpreted as valid forever.

DEFAULT-BOR-ID
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = create a default field value, using last-bor-id counter from Z52 (UTIL G/2).

DEFAULT-BOR-ID-1
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = create a default field value, using last-bor-id-1 counter from Z52 (UTIL G/2)

DEFAULT-BOR-VERIFY-1
Values:Y N K Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = create a default field value, using last-bor-verify-1 counter from Z52 (UTIL G/2)
K = the same as 'N', except in the P-FILE-20, where the empty verification field in the input file is populated with the z308-key-data.

DEFAULT-BOR-VERIFY-2
Values:Y N K Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = create a default field value, using last-bor-verify-2 counter from Z52 (UTIL G/2).
K = the same as 'N', except in the P-FILE-20, where the empty verification field in the input file is populated with the z308-key-data.

EMAIL-VALIDATION
Values: Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Possible values are: Y or N.
Determines whether or not patron's e-mail address check is performed.

Y = the system performs a validation check on the e-mail address. A notification is displayed if the e-mail address is not valid. The check is activated when the patron address is updated via GUI, OPAC, PLIF loader, DLF-API, and X-services and the e-mail field is populated.

N = No validation on the patron e-mail address (default).

ENCRYPT-Z308
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = indicates that the patron identifying numbers are encrypted

This flag affects only record creation online, and does not affect record export and import to/from an external file.
Exporting the records with p-file-03 will always export unencrypted records.
Importing with PLIF or p-file-06 can import with encryption, depending on the system settings.
Importing with p-file-04 does not import with encryption, regardless of system settings.

FAST-CAT-HOLD-ID
Type: Text; Max Length: 12
tab100 of library: Yes; tab100_<server_type>: No.
Determines the ID for which a hold request will be created when an item is created using the fast cataloging option.
If this entry does not exist then :
If the library uses user-sharing then the ID will be "CATALOGER".
If the library does not use user-sharing then the ID will be a concatenation of "CAT-" and the library name.

HOLD-BY-ITEM-GROUP
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = When a hold request is created from the Web OPAC, the list of pickup locations will be built according to the tab37 pickup locations of all items that are 'like copies' of the requested item. This value is mandatory for the 'BIB Request' functionality, but it affects also item level requests.
N = List of pickup locations is built only according to the tab37 line on the requested item.

HOLD-FULFILL-NOTIFY
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable sets the system behavior when an item is transferred from the owning library to the pickup library.

1 = Notify the patron about the request fulfillment only when the item arrives at the pickup location.
2 = Notify the patron about the request fulfillment when the item is sent from the owning library.

HOLD-REQ-PROCESS-STATUS
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = a hold request includes the process status of the copy chosen for hold, and only copies with same process status will satisfy the request.

HOLD-REQUEST-COLLECTION
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = a hold request includes the collection of the copy chosen for hold, and only copies with the same collection will satisfy the request.

HOLD-REQUEST-ITM-STATUS
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = a hold request includes the item status of the copy chosen for hold, and only copies with same item status will satisfy the request.

ILL-ITEM-SUB-LIBRARY
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

1 = (default). Always use patron home library if populated, otherwise use pickup location
2 = use pickup location if its a valid "item" sublibrary, otherwise use patron home library

ITEM-BOR-INFO-CIRC-ONLY
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Patron Info (such as Patron Name, ID) is shown only in the Circulation module, and NOT in Acquisitions and Cataloging modules, in:
Item Loan History list
Item Hold History list
Item Expand display
Item Circulation Summary display

KEEP-FILLED-PHOTO
Values: Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Photocopy request will be kept with "Filled" status, after "Filled" letter is printed.
N = Photocopy request will be deleted, after "Filled" letter is printed.

PHOTO-USE- EXACT
Values: Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = after photocopy request slip is printed (in ue_06 or in other place) the request moves to the item used to fulfill it. The relevant item's information is from the item printed on the slip.
N = information of the first like item will be used.

LATE-RET-FINE-WHEN-LOST
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable is used to set up the library's policy with regard to lost items overdue charges.

Y = When an item is declared lost, charge any overdue fines that have been accrued up to this point of time. In addition, charge the lost-item charges according to the regular tab34 definitions. If the item is later returned, refund the lost-item charges according to the regular LOST-LOAN-CREDIT-METHOD and REFUND-RATE tab100 variables policy. Do not create any new overdue fines at this point.

N = When an item is declared lost, do not charge any overdue fines that have been accrued up to this point of time. Do charge the lost-item charges according to the regular tab34 definitions. If the item is later returned, refund the lost-item charges according to the regular LOST-LOAN-CREDIT-METHOD and REFUND-RATE tab100 variables policy.
In addition, create overdue fines at this point for the entire overdue period, including the period of time in which the item was declared lost.

LOST-LOAN-CREDIT-METHOD
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

1 = when a lost loan is returned (that is, an item which has a Z36 record with status "L"), all associated cash transactions except "waived" are credited. This means "C"losed, "T"ransferred or "O"pen transactions are credited.
Also, partially waived transactions are not credited at all.
Closed / Transferred / Open transactions - Credited based on REFUND-RATE.

Waived / Partially Waived transactions - Not Credited at all.

2= when a lost loan is returned, all "T"ransferred or "O"pen transactions are credited. Closed transactions are not credited automatically.
Also, partially waived or partially closed transactions are fully credited disregarding REFUND-RATE:
Transferred / Open transactions - Credited based on REFUND-RATE.
Partially Waived / Partially Closed - Credited fully.
Closed / Waived transactions - Not Credited at all.

3 = like 1 but REFUND-RATE only applies for "C"losed transactions. Other transactions are fully refunded:
Closed transactions - Credited based on REFUND-RATE.
Transferred / Open transactions - Credited fully.
Waived / Partially Waived transactions - Not Credited at all.

4 = like 1 except that Open transactions are credited by waiving the lost bill debit transaction.

5 = like 2 except that Open transactions are credited by waiving the lost bill debit transaction.

6 = like 3 except that Open transactions are credited by waiving the lost bill debit transaction.

LOST-PROC-STATUS
Values: Default:
Type: Text; Max Length: 99
tab100 of library: Yes; tab100_<server_type>: No.

This is the list of process statuses that will be removed from an item when the barcode of the item is read. The format of the variable is 'P1 P2 P3', where 'P1', 'P2' and 'P3' are item process statuses.

OFFLINE-CASH-TRANS
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
Y = Offline Circulation transactions will record cash transactions.
N = Offline Circulation transactions will not record cash transactions.

ONLINE-RECALL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable sets the library policy regarding online recall. If the variable is set to Y any request that is created in the GUI and is required for recall will trigger a recall when the request is submitted. If the variable is set to N, the recall will be triggered only when the 'Recall Items on Loan (cir-13)' service is run.

OVERDUE-LETTER-NO
Values:0 1 2 3 4 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Defines "overdue" for tab_check_circ 1_d block. A loan is overdue if the due date has passed and "n" or more notices have been sent.

OVERDUE-LETTER-STYLE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

N = standard overdue notice, without special text based on letter number
Y = overdue notice text is sensitive to letter number

OVERDUE-RECALL-RATIO
Values:Y N B Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This switch defines whether lines 0050-0055 in tab18.<lng> (fines relating to recalled items) are to be treated as a ratio or as an actual amount
Y or blank = lines are ratio
N = lines are actual amount
B = Both - lines are actual amount but they are added to the tab16 amount

OVERLAP-BOOKING
Values: Default: 01
Type: Text; Max Length: 02
tab100 of library: Yes; tab100_<server_type>: No.

01 = Extend loan until end of Booking Period (default).
02 = Delete booking request; execute loan regularly (accept system- due date).
03 = GUI functionality: Allow staff to choose between options 01 and 02 Self check and X-Server: the same as option 01.
04 = GUI functionality: the same as option 03 (for GUI). Self check and X-Server: the same as option 02.

PATRON-LIST-ID-TYPE

Values: Default:
Type: Text; Max Length: 02
tab100 of library: Yes; tab100_<server_type>: No.

If left blank the Z353 type ID is built from Z303-ID as usual.
You may specify another Z308-KEY-TYPE to replace Z303-ID in the GUI patron list.

In addition, the type of ID that is added in the 'Additional ID' field in fast patron registration is set in this variable.

This variable **must** be defined in the alephe or the usr_library tab100 table.

Please note that any change requires running p_cir_25 to re-generate the index list.

PHOTO-REQUEST-STYLE
Values:H - Default: -
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

H= after the photocopy request slip is printed the request is deleted. The system does not check for availability of the item. If the slip is printed by ue_06 or by cir_22 the request is not deleted but its status is changed to 'H'. The system does not check for availability of the item.
blank=after the photocopy request is printed the system sets the status of the request to 'A' and the print status to 'P'.

PHOTO-SLIP-STYLE
Values:1 2 3 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

For printing photocopy request slips (cir-22). If a suitable item is available, a pickup slip is printed. Otherwise, a wait letter is printed. This switch relates to the list of items that are included in the print slip.
1 = print all "like" item records in the slip
2 = print only available "like" item records in the slip
3 = print the first of the available "like" items in the slip

PICKUP-LIST-METHOD
Values:1 2 3 4 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable controls how the pickup list will be created when a title request form is loaded.

1 = Create a joint pickup list that is based on the pickup locations of all of the requestable items that are linked to the requested record.
2 = Only the dispatch library that is defined for the patron is a valid pickup location.

Values 3 and 4 are reserved for future use.

PLAIN-ONLY
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

This variable enables shutting down the option to send e-mail letters with an attachment. Note that it is configurable **only** in the alephe table.
Y = Do not send an e-mail attachment, no matter how the patron or vendor records are defined. Only plain text in the e-mail body will be sent.
N = Send e-mail attachments depending on how the patron or vendor records are defined.
H = Do not send an e-mail attachment, regardless of how the patron or vendor records are defined. Only HTML format in the e-mail body is sent.

RECALL-METHOD
Values:1 2 3 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

"1": Let renewals update the z36_due_date and let recalls update it, if the recall-due-date is earlier. Use the z36_original_due_date as the "due-date-before-recall". (Let renewals update it.)
"2": Let renewals update the z36_due_date. Do not actually update the z36_due_date to the recall-due-date. But, if the latter is earlier than the z36_due_date, use the latter as the "effective due date". Leave the z36_original_due_date as the actual, original due date. (Do not let renewals update it).
"3": Same as "2", except always use the recall-due-date as the effective due-date - even if it is later

REFUND-RATE

Values: Default: 100 100 0
Type: Text; Max Length: 50
tab100 of library: Yes; tab100_<server_type>: No.

When an item that has been declared lost is returned by the patron, the charges that were made when the item was declared lost can be refunded.
This variable can be used to set the percentage of each charge that will be refunded. Possible values are three groups of digits, each group up to three digits long, up to value "100".
The first group sets the percentage that will be refunded for handling costs (transaction number 40),
the second group the replacement cost (transaction number 41)
and the third group the notice cost (transaction number 42).

A value of 0 means that no refund will be made, and a value of 100 means that full refund will be made. When one of the rates is set to zero, the flag ZERO-FINE-HANDLING is consulted whether to generate zero-sum credit transaction.

RENEW-DURING-LOAN
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = when a loan is performed on a loamed item, if the patron ID for the loan is the same as the patron ID in the loan record, the loan is treated as a renewal, and switch RETURN-DURING-LOAN, is ignored.
N = when a loan is performed on a loamed item, only RETURN-DURING-LOAN is in effect.

RESET-LETTER-NUMBER
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable sets whether an online recall of an item will reset the loan's Z36-LETTER-NUMBER field.

RETURN-DURING-LOAN
Values:0 1 2 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
0 = when an item is being loaned, and the system detects that the item is currently on loan, a message displays and the item must first be returned
1 = when an item is being loaned, and the system detects that the item is currently on loan, the item is automatically returned. No cash for overdue and no check for hold requests.
2= when an item is being loaned, and the system detects that the item is currently on loan, the item is automatically returned and a cash transaction is created if the item was returned late.

SHOW-USR-VERIFICATION
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = display patron verification fields when updating the Z303 (patron) in the GUI CIRC client.
N = mask out the view of patron verification fields when updating the Z303 (patron) in the GUI CIRC client.

STORE-CALL-NO-DOLLAR
Values: Default: U+0036
Type: Text; Max Length: 10
tab100 of library: Yes; tab100_<server_type>: No.

When an item's information is overridden by a temporary change, the original information is saved. This variable defines what string will replace a $ mark in the call number when the call number field is saved. It is suggested to use a string that will not appear as part of the call number data.

SUBLIBRARY-DIVISION
Values:Y N Default: N
Type: Text; Max Length: 01

tab100 of library: Yes; tab100_<server_type>: No.

Y = a hold/photo request includes the sublibrary of the copy chosen for hold/photo, and only copies with the same sublibrary will satisfy the request.

SUB-LIBRARY-DIVISION
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = a hold/photo request includes the sublibrary of the copy chosen for hold/photo, and only copies with the same sublibrary will satisfy the request.

TIMEZONE-ADJ
Values: Default: +00
Type: Text; Max Length: 03
tab100 of library: Yes; tab100_<server_type>: No.

This variable is used to set the "current time" of a library in relation to the server time. The format is a two digit number, prefixed by either a '+' sign or a '-'sign. The number is the difference in hours between the server time and the time zone of the ADM library, and the sign sets whether the ADM library is ahead (+) or behind (-) the server time. This adjustment affects circulation-related activities in an ADM library.

UPDATE-RECALL-DATE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This switch determines if, when an item is loaned, and there is a matching outstanding hold request, the loaned item should immediately be considered "recalled", or whether a loaned item becomes "recalled" only through the batch recall service (cir-13).

Y = item loan can trigger "recall" (Z36-recall-date and z36-recall-due-date are automatically set at time of loan if there is a matching "O" request). Note that a loan triggered recall is considered a rush recall.
N = recall of a loaned item (update of Z36-recall-date and z36-recall-due-date) is triggered only by p-cir-13.

USER-ADDR-PERMISSION
Values:0 1 2 3 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = No Check - Can update user Addresses
1 = Can update its Addresses only if it has any Z305 records.
2 = Can update its Addresses only if it has Z305 records other than ALEPH.

USER-HOME-PERMISSION
Values:Y N Default: N

Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

N = No Check - Can Update user information
Y = The staff user can update user (Z303,Z308.Z304) only if he is connected to a library that includes the patron's home library.

Note that this variable is configurable ONLY in the tab100 table of alephe/tab.

USER-IDS-PERMISSION
Values:0 1 2 3 4 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = No Check - Can view/update user ID's
1 = Can view/update user IDs only if the user has any Z305 records.
2 = Can view/update user IDs only if the user has Z305 records other than ALEPH.
4 = Can view/update user IDs only if there is a Z305 record for the user that matches the operator's sublibrary/library permission . This option is relevant for installations that share a common Z305 table, permission is allowed because the user has registered at the library.

USER-PERMISSION
Values:0 3 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = update of Z303 is based on standard permissions (that is, operator has been assigned privileges for update Z303
1 = Can update Z303 only if it has Z305 records.
2 = Can update Z303 only if it has Z305 records other than ALEPH.

USER-SHARING
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = The library participates in a shared user environment; patrons created by this library are NOT identified in the user record (Z303_USER_LIBRARY is blank), and will display in the Patron List when a staff operator is connected to ANY library that is defined as "Y"; patrons that are identified as "belonging" to a library (Z303_USER_LIBRARY=an ADM library code) are NOT included in the Patron List.
N = The library does not participate in a shared user environment; patrons created by this library are identified by the ADM library code in the patron record (Z303_USER_LIBRARY), and will display in the Patron List only when a staff operator is connected to the ADM library.

COURSES

NUM-REQUESTERS-ID
Values: Default: 000000010
Type: Number; Max Length: 09
tab100 of library: Yes; tab100_<server_type>: No.
Holds the number of Requester's ID to be displayed at "List of items for document" page.
There is more information to the "List of items" display in Course reading.
Copy Number and
Requester's ID columns.
Requester's ID column will display up to N ID's, each one in a new line. N will be determined by this value in tab100 (of the ADM library - for example usm50 NOT usm30) of this field. The minimum number is 1, maximum number 99.

GENERAL

ALPHA-NUMERIC-PASSWORD
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Defines whether or not the staff password must contain a combination of alpha and numeric characters.

Y = Password must have both alpha and numeric characters.
N = Password does not have to have both alpha and numeric characters.

ARC-URL
Values: Default:
Type: Text; Max Length: 99
tab100 of library: Yes; tab100_<server_type>: No.

A new GUI service : Retrieve the ARC path, for executing ARC through the "ALEPH" menu in the GUI..

The parameter ARC-URL in tab100 of the **ADM** library determines the URL for execution.

The entry "Browser" under "[Main]" in alephcom.ini defines the browser location in the GUI.

AUT-TYPE
Values:S M T Default:
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

S = regular AUT database
M = multilingual AUT database
T = Thesaurus

CORRECT-HEADING-ORDER
Values:0 1 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Defines the order of the subfields when adding or updating subfields of a heading via GUI-Search-Browse - Correct heading.

Note that this flag is relevant only when updating a single specific subfield. It has no affect when editing the full header string (all concatenated subfields).

1 – The subfields remain in the original order. The new or updated subfield is added at the end of the field.
0 – The new or updated subfield is added at the beginning of the field.

DOC-TYPE-ADM
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = indicates that the library is of type "ADM"

DOC-TYPE-AUT
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = indicates that the library is of type "AUT"

DOC-TYPE-BIB
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = indicates that the library is of type "BIB"

DOC-TYPE-HOL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = indicates that the library is of type "HOL"

DOC-TYPE-ILL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = indicates that the library is of type "ILL"

FIRST-LOGIN-PASS-CHANGE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Defines whether new staff should change the password when logging for the first time.

Y = password change is required when logging for the first time.
N = password change is not required when logging for the first time

MINIMUM-PASSWORD-LENGTH
Values: Default: 00
Type: Text; Max Length: 02
tab100 of library: No; tab100_<server_type>: No.

Defines the minimum length of a staff user's password. The maximum length is 10.

PASSWORD-CHANGE-PERIOD
Values: Default: 000
Type: Text; Max Length: 03
tab100 of library: No; tab100_<server_type>: No.

Defines the number of days after which password change is required

PASSWORD-FAIL-BLOCK
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Defines whether three failed login attempts will block the user.
Y = Three failed login attempts will block the user.
N = Three failed login attempts will not block the user.

PASSWORD-NO-USE-PERIOD
Values: Default: 000
Type: Text; Max Length: 03
tab100 of library: No; tab100_<server_type>: No.

Defines whether a staff should be blocked if the login has not been used for a defined number of days.

PW-EXPIRY-WARNING

Values: Default: 00
Type: Text; Max Length: 02
tab100 of library: No; tab100_<server_type>: No.

Defines how many days before expiration, the user will get a warning when logging into the system.

SUPPRESS-DECIMAL-DISP

Values: Default:

Type: Text; Max Length: 40

tab100 of library: No; tab100_<server_type>: No.

Allows reporting and displaying of currencies' amounts without a decimal point and the digits afterwards. This variable affects most amounts that are displayed in the GUI, OPAC, and printouts. Use SUPPRESS-DECIMAL-DISP to list all currency codes (separated by a comma) for which the decimal point and the digits afterwards should not be displayed. For example:

SUPPRESS-DECIMAL-DISP=JPY,KRW,VND

For printouts, this variable works in conjunction with the translate routine: SP-DECIMAL.

ILL

COPYRIGHT-MODE
Values:0 1 2 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = No handling of copyright for ILL
1 = US copyright mode for ILL
2 = Australian copyright mode for ILL

ILL-CHECK-DUP
Values:0 1 2 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

For a Web OPAC ILL request, define the routine for checking duplicate active requests that has been submitted by the same patron.
0 - no duplication check
1 - Reject duplicated requests: a duplication check is performed and in the event that a 'match' active request is located for the same patron, the request is blocked and the patron is prompted by a 'Duplicate Request' block message.
2 - Accept duplicate requests: a duplication check is performed and in the event that a 'match' active request is located for the same patron, an ILL request with status: 'New Staff Review' is created.

MULTI-ILL-UNIT-SELECT
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable controls the option of assigning an alternate ILL unit for a request:

N = The ill unit of the patron is used as default.
Y = A list will be displayed including all ILL units that belong to the patron's ADM. Upon selecting the alternate unit - the patron will have the ability to select the pickup location which does not belong to his original ILL unit.
P = The patron will be given a pickup locations list that is a joint list of all of the pickup locations of all the ILL units that belong to the patron's ADM.
When submitting the request, the system will direct the ILL request to the ILL unit that supplies the selected pickup location to the patron.

Note: The “Y” and “P” options are available only for Patrons entering a request in the WEB OPAC. Staff users submiting a request via GUI ILL will not have the option to select an alternate ILL unit and his assigned unit will be used (even in the case where MULTI-ILL-UNIT-SELECT=Y)

PAT-CANCEL-ILL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable defines whether the patron is permitted to cancel the ILL request (via the Web OPAC - Library Card - ILL Active and Total Request - Full View) which has already been sent to the supplier (request statuses: Sent to supplier-SV, Retry-RTY, Will be supplied-WSP, Conditional reply-CRP, Hold placed-HPL and Estimate reply-EST).

The variable is used only for the ISO ILL flow.

PAT-RENEW-ILL
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable defines whether the patron is allowed to renew an ILL loaned item (via the WEB Opac - Library Card - ILL Active and Total Requests - Full view)

The variable is used only for the ISO ILL flow.

PATRON-LOCAL-OWN
Values:0 1 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable controls how the system will response when patron submits an ILL request via Web OPAC and the request is detected as a 'locally owned'.

0 = Locally owned requests are blocked.
1 = Locally owned requesst are continued to be processed and they get the status 'Locally Owned' (LOW).

PRINT-COPY-ARRIVAL-SLIP
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y= the system will generate an arrival slip in addition to the arrival letter.
N=only the arrival letter will be printed.

RETURN-EMAIL-ADDRESS
Values: Default:
Type: Text; Max Length: 99
tab100 of library: Yes; tab100_<server_type>: No.

SHIP-TO-PICKUP
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable controls the option of selecting an alternate delivery address:

N = The delivery address quoted in the APDU is the ILL unit address.
Y = The delivery address quoted in teh APDU is the pickup location address. The 'ship to' address is displayed in the Lending request details and can be printed out on the lending slip.

Note: this function is only available for the ISO ILL flow.

SHIP-TO-PICKUP-ADDR
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = The address from tab_sub_library_address.lng marked as of type "X" populates the <DeliveryAddress> ILL tag according to specific formatting of the address.
N = Address populates the <ExtendedDeliveryAddress> ILL tag.

SUPPRESS-PATRON-INFO
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This Yes/No variable determine whether or not patron information (ID, Name, Status etc.) will be sent out with the ILL ISO Borrowing request.

ITEMS

BIND-USE-DESCRIPTION

Values:Y N Default: N

Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable is used to define what information ALEPH will return to external binding preparation software packages.

Y - Return Z30-DESCRIPTION in case Z30-CALL-NO is empty.
N - Always return the Z30-CALL-NO.

ITEM-BARCODE-OR-CALL-NO
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Item retrieval by call number is enabled ("Item Bar" in the Circulation GUI).
N = Item retrieval by call number is disabled (that is, there is an attempt to retrieve the item by barcode; if it fails, no further attempt is made to retrieve by call number).

ITEM-DATE-FROM-ORDER
Values:0 1 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = Upon registering a monograph order arrival, there is no automatic update of the related item's arrival date.

1 = Upon registering arrival of a monograph order, the arrival date of the order's related items are updated with the order's arrival date.

ITEM-FORM-STATISTIC

Values: Y N Default: N

Type: Text; Max Length: 01

tab100 of library: Yes; tab100_<server_type>: No.

N = The item's statistic field (Z30-ITEM-STATISTIC) is not mandatory. The Item record can be saved with null in the Statistic field.

Y = The item's Statistic is mandatory. If the Statistic field is left blank when an item record is saved, a message is displayed requiring the user to populate the Statistic field.

INVENTORY-NUMBER
Values:Z52 Z311 Z313 Default: Z52
Type: Text; Max Length: 04
tab100 of library: Yes; tab100_<server_type>: No.

Z52 - Z30-INVENTORY-NUMBER can have any value; it can be assigned by using Z52 sequences, defined by UTIL G/2.To do so, a question mark (?) followed by Z52-

sequence> must be entered in the GUI field "Inventory Number" (for example, ?inventory-number).
Z311 - Z30-INVENTORY-NUMBER is always assigned by using the Oracle table Z311 and the ALEPH table ./usm50/tab/tab_z311.
Z313 - Z30-INVENTORY-NUMBER is always assigned by locating a relevant Z313 record; A mere question mark (?), or a question mark followed by <Series> MUST be entered in the GUI field "Inventory Number" for the relevant Z313 record to be found and allocated to Z30-INVENTORY-NUMBER.

Note that any inventory number can be manually entered in this field (NOT beginning with "?"), to be saved ‘as is’ in the item record's field Z30-INVENTORY-NUMBER.

The Inventory Number is not a mandatory field, and therefore it can be left blank

ITEM-BARCODE-OR-CALL-NO
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Note that this switch is only relevant for the Item bar of the Circulation module.
Y = Item retrieval by call number is enabled
N = Item retrieval by call number is disabled (that is, there is an attempt to retrieve the item by barcode; if it fails, no further attempt is made to retrieve by call number).

MATCH-Z30-ISSBD
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = the system will ignore all analytical links except the ones to Z30-ENUMERATION-A (volume) and Z30-CHRONOLOGICAL-I (year) for the display of bound items (Z30-MATERIAL=ISSBD), which are linked to the bib-record via item link (LKR type ANA and ITM).

UPDATE-Z30-COPY
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = Z16-COPY-SEQUENCE is copied to Z30-COPY-ID when an item record (Z30) is opened for the subscription.
N = Z16-COPY-ID is copied to Z30-COPY-ID when an item record (Z30) is opened for the subscription.

X852-ITEM-OVERRIDE
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = item location subfields (sublibrary, collection, call no.) are updated from the 852 field of the linked HOL record. The line

setenv correct_852_subfields

in aleph_start controls which 852 subfields update the Z30 (item) location fields.
"Y" must be set in both the xxx50 and xxx60 libraries.

X852-ITEM-OVERRIDE-C

Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable is evaluated ONLY when X852-ITEM-OVERRIDE is set to "Y".
Y = HOL field 852 subfield c (Collection) overrides Z30-COLLECTION and Z16-COLLECTION.
N = HOL field 852 subfield c (Collection) does NOT override Z30-COLLECTION or Z16-COLLECTION, although X852-ITEM-OVERRIDE is set to "Y".

Note that in case of new item records (Z30), Z30-COLLECTION will be populated from subfield c of field 852 regardless of the value of this variable (again - only if X852-ITEM-OVERRIDE is set to "Y").

Z30-PRICE-FROM-ORDER
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = the item price field is controlled by and automatically updated from the acquisition price (estimated and invoiced).

UPDATE-BARCODE
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.
This variable is used to define whether the item barcode can be updated by the user in the GUI.

Y = Default. Enable barcode data update in GUI.
N = Disable barcode data update in GUI.

OPAC

ACC-COUNT-STYLE
Values:S R Default: S
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This switch determines how the counter of number of related BIB records displays in a browse list, when the heading is a "see" reference.

R = counts both the BIB records linked to this heading and also the BIB records linked to the heading pointed to.
Normally, there will be no records under the "see" heading, since it refers to another heading. However, this can occur in libraries that do not automatically update BIB records because of AUThority record "see" references (these libraries use UPD=N in the AUThority record).
S = count only the BIB records that are linked to this "see" heading. In libraries that that do not use UPD=N, and do automatic correction from cross references, this will normally be zero.

ACC-COUNT-LIMIT

Values: Default: 000000050
Type: Number; Max Length: 09
tab100 of library: Yes; tab100_<server_type>: No.

ACQ-REQUEST-OWN

Values: Default:

Type: Text; Max Length: 10

tab100 of library: Yes; tab100_<server_type>: No.

This variable is used in ADM libraries only. Its value is used to populate the "OWN" field of a new BIB record created as a result of an ACQ Request submitted in the Web OPAC. It can contain the ADM library code (for example, USM50) or any other value up to 10 characters. Note that the ADM library from which this Tab100 variable is read is the ADM library related to the patron's Home Library (Z303-HOME-LIBRARY) as specified in tab_sub_library.lng.

AUTO-TRUNCATE-Z01-FIND
Values:Y N S Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = when a FIND search is performed on an access (browse) list, the search term is automatically set as truncated.
S = same as Y, but the truncation occurs only after a space, not in the middle of a word. For example, when a search is performed on a browse list, will retrieve results that begin with the entered text as a whole word. For example, the search "TIT=law" will retrieve: "Law", "Law and apocalypse" and "Law and order". It will not retrieve "Lawyer", "Lawn care for your home" etc.

BASE-DEMO-LIB
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

1 = the system's demo data base is USMXX

2 = the system's demo data base is MABXX
This parameter is used for tables synchronization report (UTIL H/1), and for UTIL G/2 values.

BROWSE-NO-MATCH
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = When browsing in the Web OPAC, and an exact match is NOT found in the headings list/words list/direct index list and so on., a message such as:
[searched string] would have appeared here
(taken from message No. 9982 in ./$aleph_root/error_lng/www_f_heading) will appear immediately preceding the next best match in the list.
N = Default value. The message will not appear in case an exact match is not found.

DEFAULT-BOR-LIBRARY
Values: Default:
Type: Text; Max Length: 05
tab100 of library: Yes; tab100_<server_type>: No.

The login process to the Web OPAC determines the ADM enviroment in the following order:

a. The connected ADM library (if it is not "Shared Pool")
b. The Z303-HOME-LIBRARY if it exists
c. The first ADM library for which the user has local information.
d. TAB100-DEFAULT-BOR-LIBRARY

FILTER-LIBRARY-CARD

Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = whether to filter libraries presented in "My Library Card" to include only shared pool's libraries in which the patron is active.
N = display all libraries which are part of shared pool in "My Library Card".

ITMG-DISPLAY
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

In Web OPAC, for ITM1, ITM4 and ITM5:
Y = display the sublibrary in edit_doc_999 only if there are Z30 (item) records for the library.
N = display the sublibrary in edit_doc_999 if there are either Z30 (item) or HOLdings records for the library

MAX-CANDIDATES-MATCH
Values: Default: 100
Type: Text; Max Length: 03
tab100 of library: Yes; tab100_<server_type>: No.

Determines the maximum number of candidates to be presented as possible matches for documents in a Union Catalog for which no match was found.

PDS-AWARE
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Y = use the PDS as user authentication in the Web OPAC

PDS-KEY-TYPE
Values: Default:
Type: Text; Max Length: 02
tab100 of library: Yes; tab100_<server_type>: No.

Defines the Z308-KEY-TYPE against which all PDS authentications are verified.

Multiple key types may be defined by using a comma separator, such as 01,02,03.

PICKUP-SORT-BY-ITEM
Values:Y N Default: Y
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

When presenting pickup location lists for hold requests or photocopy requests, the patron's home library is always used as default. This variable defines what sublibrary will be used as default when no home library is defined.

Y = When Z303-HOME-LIBRARY is not defined, the item sublibrary will be the default pickup location.
N = When Z303-HOME-LIBRARY is not defined, the default pickup location will be the first sublibrary in the tab37 or tab38 setup.

THUMBNAIL-LINK
Values:1 2 Default: 1
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

This variable is used to determine what happens when selecting a thumbnail in ADAM.
1 = If the thumbnail is linked to an object, double clicking the thumbnail retrieves the object. If no object is linked, a message is displayed informing the user that the objects are available from the full view of the record.

2 = The thumbnails are not linkable and cannot be used to retrieve the objects.

UNION-LIBRARY
Values:0 1 2 Default: 0
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

0 = BIB database does not use "union" features
1 = Union View Catalog: BIB database uses Z120 to display equivalent BIB records as one record. The single record display is linked to the multiple related ADM records for items/holdings display
2 = Union Catalog: BIB database can use Z120 to display equivalent BIB records as one record. Holdings information is imbedded in the BIB record, and there is no item display.

X852-HOLDING
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = whether to use a special program that presents a list of holdings taken from 852 tag. For union catalogs.

Z39-LOGIN-BASE-CHECK
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: Yes; tab100_<server_type>: No.

Y = do not permit to non-signed users connecting via Z39.50 to access bases access limited to signed-in users.
XSS-VALIDATION
Values:Y N Default: N
Type: Text; Max Length: 01
tab100 of library: No; tab100_<server_type>: No.

Determines whether to activate the HTML validation to prevent an XSS breakthrough (a malicious insertion of HTML script by a user).

Example of the table in the BIB library

```
CENTRAL-LIBRARY=N
CREATE-852-HOL=Y
CREATE-Z30H=Y
X852-ITEM-OVERRIDE=Y
HOLD-REQ-PROCESS-STATUS=Y
BARCODE-DELETE-SPACES=Y
CHECK-BARCODE=N
MARC-TYPE=1
DOC-TYPE-BIB=Y
DOC-TYPE-HOL=N
DOC-TYPE-ADM=N
DOC-TYPE-AUT=N
DOC-TYPE-ILL=N
ITMG-DISPLAY=N
CHANGE-DUE-HOUR=Y
```

Example of the table in the AUT library

```
CREATE-852-HOL=T
MARC-TYPE=1
ADDRESS-ZIP-STYLE=A
AUT-TYPE=S
DOC-TYPE-BIB=N
DOC-TYPE-HOL=N
DOC-TYPE-ADM=N
DOC-TYPE-AUT=Y
```

Example of the table in the ADM library

```
HOLD-REQUEST-ITM-STATUS=Y
HOLD-REQUEST-COLLECTION=Y
CENTRAL-LIBRARY=N
CREATE-852-HOL=Y
CREATE-Z36H=Y
CREATE-Z37H=Y
CREATE-Z30H=Y
CHECK-INVOICE-CURRENCY=N
X852-ITEM-OVERRIDE=Y
HOLD-REQ-PROCESS-STATUS=N
RETURN-DURING-LOAN=0
CHECK-ORDER-BUDGET=Y
```

tag_text.dat

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Pre-defined text options for subfields

Related table(s):

1) tab01.<lng>

Structure of table:

col. 1	tag
col. 2	indicators. Use # as wildcard.
col. 3	ALPHA Direction of input in cataloging. L (Latin), H (Hebrew).
col. 4	subfield. Use # to indicate any subfield.
col. 5	Value // Description

Note that if changes are made to this table, UTIL M/7 must be activated in order for the changes to be reflected in the online catalog.

Example of the table:

```
!1  2  3 4      5
!!!-!!-!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
655 ## L a Biographies
655 ## L a Bird's eye views
655 ## L a Cartoons
655 ## L a Catechisms
…
LKR ## L a PAR
LKR ## L a ITM
STA ## L a SUPPRESSED
STA ## L a DELETED
```

Note that to activate the option in the online cataloging enter F8 (or defined hot key for the list), and the beginning text of the element.

For example for the 655 tag:
“b” + tab will present the user with a list of the entries beginning with B (there is more than one entry beginning with the letter ’b’.
When entering Q, however, as there is only one entry with the letter Q (*case-sensitive*), the text will automatically be placed in the field.

Note that the library can define any kind of text list for field fill in – as per the example of the **STA** field

tagonnew.dat

Location of the table: pc_tab/catalog directory of the library

Purpose of the table: Default fields for a new record

Related table(s):

1) tab01.<lng>

This table defines which are the default fields when “Create New Record” is activated in the online Cataloging Module.

Structure of the table:

Col. 1 ALPHA + field code (that is, default field(s))

Example of the table:

```
!  1
!!!!!!
L008
L245
L260
```

unicode_to_filing_01

Location of the table: alephe/unicode directory

Purpose of the table: Unicode character conversion equivalencies for sorting

Related table(s):

1) tab_filing
2) tab_character_conversion_line

This table defines equivalencies for characters, for the purpose of filing (sorting). All characters naturally sort by their unicode value. In order to force a different sort order, an equivalency for sorting can be set. The equivalency can be up to 5 characters.

For example:
in order to sort an accented lowercase **a** (for example, Latin small letter a with acute - **á**) as a regular capital **A**, set the equivalency of the accented character (00E1) to 0041 (the hexadecimal value of **A**).
Another example:
in order to sort **ü** as **ue,** set the equivalency of **u-umlaut** (00FC) to **U** + **E** (0055 + 0045).

Remember that some of the procedures in the library's tab_filing table also change characters (for example, compress, compress_blank, expand_num, and so on.)

When a browse query is parsed, the data is translated using the unicode_to_filing table, not tab_filing. Therefore, the two tables should be coordinated. In actual fact, "to_blank" in tab_filing is equivalent to setting a character to 0020 in unicode_to_filing
"compress" in tab_filing is equivalent to setting a character to 0000 in unicode_to_filing,
and it suffices to set their value in the unicode_to_filing table. However, for purposes of clarity, the following characters have been defined both in tab_filing (procedures 11,11,12,99), and in unicode_to_filing:
compress (set to 0000): '"`<>%
to_blank (set to 0020): ./-:;,{}[]()?!^_

The library may have special filing requirements for non-textual information (for example, UDC, DDC, call numbers), in which case a different unicode_to_filing table

should be set, in which the value of special characters such as those listed in "compress" and "to_blank" should be retained.

Also note that tab_filing can include "to_lower" which uses the unicode_case table to transfer all uppercase characters to lowercase.

Note:

- All characters are expressed in hexadecimal values.
- Characters which are not listed in this table, and characters which are listed in col.1, but do not have a value in col.2, are dropped from the filing key. In other words, all filing characters ***must*** be defined in the filing table.
- The last column in the table is a note field, giving a description of the character.
- There can be multiple **unicode_to_filing_nn_source** tables (where **nn** is replaced by two digits). The relevant table is called up by the FILING-KEY-nn line in the tab_character_conversion_line table. The library's tab_filing table points to the FILING-KEY-nn line.

In this sample file, unicode_to_filing_01, there is character conversion for
Basic Latin,
Latin-1 Supplement + Latin Extended-A (0021 through 017F),
Cyrillic,
Greek,
superscript,
subscript,
some other individual characters.

We have not provided character conversion for other characters and languages (for example, Danish, Czech).

In general, special characters in these languages have been left with their "natural" value.

In the 0021 through 017F range,
some special characters have been translated to 0000 (for example, PERCENT SIGN),
accented characters have been translated to the equivalent non- accented character,
and a few characters have been translated to two characters (for example, 00C6 (AE) to A + E).

Structure of the table:

Col. 1	Character as stored in the database, in its hexadecimal value
Col. 2	Equivalent – that is, character to which character will be changed
Col. 3-6	Equivalents – that is, 2nd – 5th characters in string to which characters will be changed
Col. 7	Description of the character. Must begin with #

Example of the table:

```
0020 0020                    #SPACE
0021 0000                    #EXCLAMATION MARK
0022 0000                    #QUOTATION MARK
0023 0023                    #NUMBER SIGN
…
00C0 0041                    #LATIN CAPITAL LETTER A WITH GRAVE
00C1 0041                    #LATIN CAPITAL LETTER A WITH ACUTE
00C2 0041                    #LATIN CAPITAL LETTER A WITH CIRCUMFLEX
00C3 0041                    #LATIN CAPITAL LETTER A WITH TILDE
00C4 0041                    #LATIN CAPITAL LETTER A WITH DIAERESIS
00C5 0041                    #LATIN CAPITAL LETTER A WITH RING ABOVE
```

unicode_to_word_gen

Location of the table: alephe/unicode directory

Purpose of the table: List of Unicode character conversion equivalencies, to be used when creating records in the WORDS file

Related table(s):

1) tab_word_breaking

This table defines equivalencies for characters, for the purpose of creating words in the words file. It is used in conjunction with the library's tab_word_breaking table. All characters naturally retain their unicode value, and are stored in the system in UTF encoding. In order to translate one character into another character (for example, translating an accented "e" to "e"), you can set an equivalency. The equivalency can be up to 5 characters.
For example,
to translate a lowercase "a" with acute accent to regular "a", set the equivalency of the accented character (00E1) to 0061 (the hexadecimal value of a).
to set an umlauted "u" as "ue", set the equivalency of u-umlaut (00FC) to "u" + "e" (0075 + 0065)..
Notes:
All characters are expressed in hexamdecimal values.
The last column in the table is a note field, giving a description of the character.
The unicode_to_word_gen table is used both for building words entries and for translating search query input.
The library's tab_word_breaking table can define different treatment for the same characters. In separate procedures specific characters can be set to compress or to be changed to blank. Characters dealt with in this manner should be left in their natural value, and not translated in this table.
Note that the system automatically carries out triple posting for hyphens and apostrophes:

(1) as separate words;
(2) as is (with hyphen/apostrophe);

(3) with hyphen/apostrophe compressed. For example, twenty-five is indexed as:

twentyfive
twenty
five
twenty-five

Both the hyphen and the apostrophe must be left with their actual value (hyphen - 002D and apostrophe - 0027), and not translated in this table.
The table does not have to include all characters. It suffices to register the exceptions.

Structure of the table:

Col. 1 Character as stored in the database, in its hexadecimal value

Col. 2 Character to which character will be changed

Col. 3-6 2nd – 4th characters in the string, to which the character will be changed

Col. 7 Description of the character. Must begin with #

Example of the table:

```
0041 0061                    #LATIN CAPITAL LETTER A
0042 0062                    #LATIN CAPITAL LETTER B
0043 0063                    #LATIN CAPITAL LETTER C
0044 0064                    #LATIN CAPITAL LETTER D
…
00C0 0061                    #LATIN CAPITAL LETTER A WITH GRAVE
00C1 0061                    #LATIN CAPITAL LETTER A WITH ACUTE
00C2 0061                    #LATIN CAPITAL LETTER A WITH CIRCUMFLEX
00C3 0061                    #LATIN CAPITAL LETTER A WITH TILDE
00C4 0061                    #LATIN CAPITAL LETTER A WITH DIAERESIS
```

union_global_param

Location of the table: alephe/tab directory

Purpose of the table: Union deduplication parameters

Structure of the table:

Col. 1 Library

Col. 2 T = for testing (UTIL/F/1/21)
B = for batch jobs

Col. 3 Code

Col. 4 Program

Col. 5 Table

Note that the codes used for performing a refined search can be written in col. 5.

There are two types of refine:

for records with the format "SE"
all the rest.

The default codes are:
"WPL" for "SE" records.
"WYR" for all the rest.

To define other code names, the codes should be entered in col. 5 in the following order: first the code for non "SE" records, a comma, then the code for "SE" records.

For example:

```
! 1   2           3                          4                      5
!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!
USM05 B candidate_prog       union_candidate_cdl            WID,
```

non "SE" = WID
SE = WPL (default)

```
! 1   2           3                          4                      5
!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-
!!!!!!!!!!!!!!!!!!!
USM05 B candidate_prog       union_candidate_cdl            ,WXX
```

non "SE" = WYR (default)
SE = WXX

Example of the table:

```
! 1   2           3                          4                      5
!!!!!-!-!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!…!!!!
USM90 B candidate_prog       union_candidate_cdl
          100
USM90 B match_prog           union_match_cdl
USM90 B preferred_prog       union_preferred_cdl
USM90 B normalize_prog       union_normalize_cdl
!
USM90 T candidate_prog       union_candidate_cdl
```

union_preferred

Location of the table: tab directory of the **BIB** library

Purpose of the table: Weighting table for determining which of two matching records is the preferred record

Related table(s):

1) tab_preferred
2) tab_merge_adv_overlay

3) tab01.<lng>

This table's functionality can be carried out by a table of any name. The name is not set in the programs, it is set in the **third column of the tab_preferred** table. It is the table to be used for weighing and giving points to a record in order to set which of two matching documents is the preferred document.

Setting the preferred document is an option for the merge_doc_adv_overlay program, used when merging records.

This program uses the tab_merge_adv_overlay table to decide which fields to retain and which fields to drop when merging records. Preferred record identification is an added parameter in the tab_merge_adv_overlay table.

Structure of the table:

Col. 1 Field tag. Document record field code. # can be used for the third to fifth positions to indicate truncation of numeric additions to the field code (for example,, 245## for 2451, 2452, 24501).

Col. 2 Subfield(s) of the field (blank indicates the entire field).

OR

F followed by fixed field position (counted from base 00, taking the subfield code into account if there is a subfield code), followed by a hyphen (-) and the number of characters.

Col. 3 Operation code - Code of the action to carry out on the data. Available actions are EQUAL, NOT-EQUAL and PRESENT.

Col. 4 Value of the subfield content, or of the data in the section of the fixed field. Multiple values separated by a comma are treated as "or".

Col. 5 Points assigned. Number of points to assign when the line is "true". A negative number of points can be assigned by placing a minus in the first position, followed by 2 digits.

Example of the table:

```
!!!!!-!!!!!!-!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!-!!!
LDR   F05-01 EQUAL      d                          -10
LDR   F17-01 NOT-EQUAL  1,2,3,4,5,7,8,u,z          010
LDR   F17-01 EQUAL      1                          009
LDR   F17-01 EQUAL      2                          008
```

user_function.<lng>

Location of the table: alephe/tab directory

Purpose of the table: User authorization definitions and codes

Related table(s):

1) aleph_start – pw_library variable

This is a list of the password authorizations that can be assigned to users, including all functions in the Web Services module (Lines which begin WWW-U, R, G, N, T).
A single password can be defined for several libraries. The passwords, however, are defined in one library which must be set up as the 'pw_library' in aleph_start.

NOTE: This table should not be modified except for translation purposes. When translating, ONLY the NAMES, should be translated, NOT the codes.

Structure of the table:

Col. 1 Internal code of function. This should NOT be translated.
Col. 2 ALPHA
Col. 3 Name of function (can be translated)
Col. 4 Internal code of subfunction. This should NOT be translated.
Col. 5 ALPHA
Col. 6 Name of subfunction (can be translated)

Example of the table:

```
!  1                 2         3                              4                 5  6
!!!!!!!!!!!!!!!!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!-!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

GLOBAL              L All Functions                GLOBAL              L All
 sub-functions
OVERVIEW-TREE       L Overview Tree                VIEW-TREE           L Vie
w Overview tree
ACQ                 L Acquisitions                 GLOBAL              L All
 subfunctions
ACQ                 L Acquisitions                 INDEX-LIST          L Ord
er Search - Order index
ACQ                 L Acquisitions                 LIST                L Ord
er Search - New/Cancelled index
ACQ                 L Acquisitions                 ORDER-LIST          L Ord
ers - Display list for single record
ACQ                 L Acquisitions                 ORDER-VIEW          L Ord
er - View summary information
ACQ                 L Acquisitions                 ORDER-GET           L Ord
er - Display input form
```

www_f_func

Location of the table: alephe/tab directory

Purpose of the table: Functional codes linked to HTML action commands

This table includes all functional codes that come from the HTML action command, and the correspondence program name and parameters (if needed). It is an internal table, which makes it easier to add programs for particular applications or projects.

Structure of the table:

Col. 1 Library code

Col. 2 function code

Col. 3 program name

Col. 4 parameters

Example of the table:

```
! 1             2                                3
!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!
>
##### FIND-A-0             www_f_find_a_0
##### FIND-A               www_f_find_a
##### FIND-B-0             www_f_find_b_0
##### FIND-B               www_f_find_b
##### FIND-B-LIST          www_f_find_b_list
##### FIND-B-PERMUTE       www_f_find_b_permute
```

www_f_heading

Location of the table: aleph/error_<lng>

Purpose of the table: Messages to the patron in the Web OPAC

Related table(s):

1) tab15.<lng> related to heading #8013

Note: In version 17.0 the directory error_<lng> was moved from alephe/error_<lng> to aleph/error_<lng>.

This table lists the messages sent to the patron in the Web OPAC interface.

Note the following:
Availability display in the "Due date" column - heading #350 ("On shelf") is sensitive to the Z30-ITEM-PROCESS-STATUS of the item. In order to take care of this, the following changes have been implemented:
There is a file containing all headings for an item's circulation status. The file is $aleph_root/error_lng/circ_status_holding. Codes 8000 - 8012 were removed from $aleph_root/error_lng/global and put into this table. The holdings display in the web now uses codes 8000 - 8013 in circ_status_holding instead of codes 0350 - 0359 in www_f_heading
A new heading was created - 8013. If Z30-ITEM-PROCESS-STATUS is blank, the heading #8012 (On Shelf) appears. If it is not, heading #8013 appears with the text taken from tab15.<lng>.

Structure of the table:

Col. 1 Message number

Col. 2 ALPHA

Col. 3 Text displayed to patron in Web OPAC.

Example of the table:

```
!!!!-!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!>
0019 L Reenter your password.
0095 L Enter patron ID / barcode.
0100 L Unrecognized operation code. Type ENTER.
0105 L Error in borrower ID/sublibrary.
0106 L Note is too long.
0107 L Note is empty.
0108 L Note cannot be added on to current notes.
0109 L Borrower doesn't have Items on Loan.
0110 L Your note has been delivered.
0120 L All items
0122 L not sorted.
0123 L sorted by :
0124 L Sort options :
0132 L Verification fields do not match or empty.
0133 L Verification field is empty.
```

Note that if the patron is allowed to update his/her address, the link for address update from the HTML page displays. If the patron is not allowed to update address, line 4052 from www_f_heading displays.

www_f_sort_heading.<lng>

Location of the table: tab directory of the **BIB** library

Purpose of the table: Sort options for Web OPAC brief formats

Related table(s):

1) www_tab_short.<lng>
2) tab_sort
3) www_f_heading

The brief display of bibliographic data in the Web OPAC allows for sorting of the different display elements as defined in www_tab_short.<lng>.

WWW_f_sort_heading defines the sort options, and their description. The description of the sort displays on this brief list of records. Note that the sort keys are set up in tab_sort.

The display includes the current sort -- Heading "Sorted by" (0123 in www_f_heading), and optional sorts – Heading "Sort options" (0124 in www_f_heading).

Structure of the table:

Col. 1 Primary sort key code.

Must match the sort key as defined in the tab_sort table

Col. 2 A-Ascending
D-Descending
Col. 3 Secondary sort key code
Must match the sort key as defined in the tab_sort
Col. 4 A-Ascending
D-Descending
Col. 5 Text that displays after "Sorted by"
Col. 6 Text that displays after "Sort options"

Example of the table:

```
!1   2 3   4       5                                  6
!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!
02---A01---D Author/Year (descending)       Author/Year(d)
02---A01---A Author/Year (ascending)        Author/Year(a)
01---D02---A Year (descending)/Author       Year(d)/Author
02---A03---A Author/Title                   Author/Title
03---A01---D Title/Year (descending)        Title/Year(d)
03---A01---A Title/Year (ascending)         Title/Year(a)
01---D03---A Year (descending)/Title        Year(d)/Title
```

Note that the linked pairs of primary and secondary sort options, should match the sort definitions as defined in www_tab_short.<lng>:

```
www_f_sort_heading:
02---A01---A Author, then Year …
01---D02---A Year, then Author …
www_tab_short:
1 L Author     02---A01---A 00 00 0100 S 9 -          1#### 700##
3 L Year       01---D02---A 08 04 0000 S              008
```

www_f_tab_short_999

Location of the table: tab directory of the **BIB** library

Purpose of the table: Custom brief display format for Web OPAC

Related table(s):

1) tab01.<lng>

This table defines the short format options (not short table view) for the Web OPAC. The selection of the format is through the ***preferences*** (option-display/option-display-nobor) screen.

The first column defines the format number. The format number should be between 951-998. 950 is reserved for the full edit_doc_999.<lng> view.

Columns 2-27 define the field codes which will be displayed.

Note: Format 999 must be defined. The system will use it if a format that does not exist in the table is chosen.

Col. 1 Format number

Cols. 2-27

Field code

Example of the table:

```
951 1#### 24### DIS   856## ITM5  ITMH
952 24### DIS
953 24### ITM3# DIS
954 24### ITM3# DIS
!
999 1#### 24### ITM5# DIS
```

www_server.conf

Location of the table: alephe_root directory

Purpose of the table: Default values for the Web

Related table(s):

1) aleph_start ADJACENCY relates to www_server.conf's setenv set_prox_limit
2) setenv www_parallel_search_base
3) tab100 - AUTO-TRUNCATE-Z01-FIND - setenv www_parallel_search_base
4) tab_sort – set env www_sort_field
5) tab15.<lng> - setenv www_z37_recall_type
6) www_f_heading - setenv www_short_max_no_ext_links

General note:
Note that there are some definitions that offer more than one "setenv" option. The non-active option does not have to be commented out – the system will always use the **last** line defined for that option.

Explanation of some of the elements related to path in the file:
setenv icon_path "&server_httpd/www"
It is possible to use an httpd daemon to load icon GIF files on to the **ALEPH** WWW directories. This will improve the efficiency of the main WWW server.

setenv security_path
"${ALEPH_MOUNT}/a${ALEPH_VERSION}_${ALEPH_COPY}
;/tmp;$TMPDIR;$FILE_TMPDIR"
In order to be able to use the "*save set"* option in the Web OPAC, the security path must include $TMPDIR and $FILE_TMPDIR:
Only the files under the defined sub-file-systems can be delivered by the server.

Explanation of some of the variables:

setenv allow_create_new_profile Y
If set to "Y", the system will create a new profile for the patron

setenv create_statistics N
The z34 records can be used for statistical analysis of WEB server use. Overrides the default that is set in aleph/proc/www_server; If set to Y the counter “last-z34-sequence” should be defined in the z52 table (UTIL G/2).

set_default_z110_create_order A
A variable that controls how documents are added into a set (z110).
The order can be "A" (ascending) or "D" (descending).
The default is "A".
This order is important in non-sorted sets as typically the last document numbers in a system are the most the recently cataloged records.

setenv direction_support Y
Text displayed in Web pages of the Web OPAC will be right-aligned and right-oriented (for Hebrew and Arabic)
Note the following: Add placeholders with numbers greater by 50 than the original, to Web pages for which right-to-left support is required.
For instance, if the title (placeholder $0400 in ./alephe/www_f_eng/short-a-body) has to be right-aligned and right-oriented, add a placeholder - $5400 - immediately following the cell's class definition:

- Without right-alignment:

<td class=td1 width="30%">$0400</td>

- With right-alignment:

<td class=td1 $5400 width="30%">$0400</td>

setenv due_date_format 1
For details see explanation for the same variable as defined in pc_server_defaults

setenv hol_filter_op BASE
setenv hol_sort_op BASE
The variables hol_filter_op and hol_sort_op can determine the filtering and sorting options of the holdings list on the holding information page.

setenv ill_borower_status 01
This is used for setting the borrower status when creating a patron from an X-Service.

setenv ill_total_limit_period 0814

setenv mark_match_words_prefix ""
setenv mark_match_words_suffix ""
Highlight words in full (999) format display (edit_doc_999.<lng> format) depending on the search request. The function compares each line with Z05-REQUEST.
The two parameters mark_match_words_prefix and mark_match_words_suffix defines what is to be added before and after the matched word.
In the example above, the word will be highlighted. In order, for example, to have the matched word appear in **bold** define as follows:

mark_match_words_prefix ""

mark_match_words_suffix ""

setenv server_default_charset iso-8859-1
If the WEB server detects that the Web OPAC browser is not UTF-8 enabled (that is, is not Netscape 6+ or Microsoft Internet Explorer 5+) the UTF-8 parameter in the HTML files is "naturalized", and the data is sent back from the server in a standard character set. The character set to be used is defined in this line.

setenv patron_notice_cash Y
When this parameter is set to Y, the system checks the patron's cash transaction after the patron logs into WWW OPAC and displays the “Pending Cash Transaction” notice if unpaid cash transactions are detected. (HTML notice page: ./alephe/www_f_eng/patron-notice).

setenv patron_notice_delinq Y
When this parameter is set to Y, the system checks the patron's delinquencies after the patron successfully logs into WWW OPAC and displays the “Delinquency” notice if delinquencies are detected. (HTML notice page: ./alephe/www_f_eng/patron-notice).

setenv patron_notice_overdue Y
When this parameter is set to Y, the system checks patrons loaned items and displays the “Overdue Items” notice if the patron currently is holding overdue items. (HTML notice page: ./alephe/www_f_eng/patron-notice).

setenv set_prox_limit 500
Sets the limit on the number of documents that will be checked for adjacency, if the aleph_start ADJACENCY is set to "0".

The limit is 500.

setenv set_result_set_limit
Even though the number of hits for a find command was unlimited the actual records in a set was limited to 1000. It is now possible to change this limit in the range of 1000-20000

Please note that this will affect performance for large sets.

Also the Web screen short-head-2 should be changed

From:
Records $0100 -$0200 of $0300 (maximum display and sort is 1000 records)

To:
Records $0100 -$0200 of $0300 (maximum display and sort is $4500 records)

setenv set_word_limit 01000
Limits the number of words that will be "collected" when truncation is used (for example, find a? will do a find on all words beginning with a) . If a find limit is defined, this is the number of Z97 record numbers (that is, distinct words) in a given search.

Maximum value is 9999. The system will stop the FIND if the number of documents in the set exceeds the defined limit.

setenv www_base USM01
Default database login

setenv www_browse_previous Y/N

- Y - show one line before scan start
- N - no preview. List will display from requested entry. For example:
  Patron requests scan of author “jones”:
  where set to N. The list displays from “jones”
  where set to Y. The list displays one entry before “jones”

setenv www_con_lng ENG
Default language of interface

setenv www_course_email_address
The default email address to which course reading material will be sent (from the Web OPAC) for addition into the course reading module.

Note that for the option for saving the basket contents from the Web OPAC into a server file plus notifying the course-reading administration, it is possible to specify inside the web-page the e-mail address of the administration. Only if the value is not an e-mail address, (for example, the default value of the check box being checked) , the variable www_course_email_address will be consulted.

This option can be used to build a drop-down menu of e-mails where there are several course reading administrations to choose from.

For example , setting the Web page "basket-course" in this manner :
...
<input type=checkbox name="SEND_MAIL" value="tsachm@exlibris-usa.com">
Check here in order to inform the Course Reading admini...

will send the notification e-mail to tsachm@exlibris-usa.com , while
this setting :
…
<input type=checkbox name="SEND_MAIL">
Check here in order to inform the Course Reading admini...

will send the e-mail to the value defined in "www_course_email_address".

setenv www_dn_link_display 03
Defines number of down links to display in the full view of the parent record. If there are more down links than the number defined in www_server defaults, the full view of the parent record will offer two options:
viewing the down links in a separate window
creating a set of down linked records

setenv www_dn_view_display

Y=

setenv www_item_schedule 14
Defines the number of days that should be accessible for advanced booking of short loan items. According to the example, a patron will be able to book an item up to 14 days in advance.

setenv www_lib_info_link N/Y
If set to Y, patron can click on the “About library” hypertext link in order to see more information about a branch library.
Note: does not influence **ITM4 .**

setenv www_myeshelf_limit 020
This variable sets the maximum number of documents an e-Shelf folder can contain – both the Basket or a patron-defined folder.
Note that the maximum number of documents in a folder cannot be more than 500. If the variable is not defined, or has value more than 500, it will automatically be set to 500.

When adding documents to "My e-Shelf" if the folder limit will be exceeded by adding documents, a warning is displayed and the documents will not be added.

setenv www_no_items_display 40
Defines the number of items that will display. The default is 40

setenv www_parallel_search_base USM01-CUN01
The service - www_f_service_find_m_acc in tab_service which activates find-m on bases which are defined in this variable.
Note that the browse lists in the various databases must be defined with the same code.
In order to ensure that long titles are retrieved, the AUTO-TRUNCATE-Z01-FIND switch in tab100 should be set to "Y".

setenv www_short_max_no_locations "05"
Defines the maximum number of locations that will display in the brief display. Note that the system limit is 2000 characters. If either this or the parameter in www_short_max_no_locations is exceeded, a message displays:

9564 L More Libraries (as defined in www_f_heading)

setenv www_short_max_no_ext_links "2"
Defines the maximum number of external links that will display in the brief display (using code EXTL. See www_tab_short.<lng> for more information). lines is limited either by this variable or to 2000 characters. In both cases a heading is displayed to indicate that there are more lines: heading 9568 (as defined inwww_f_heading)

setenv www_sort_field "01---D02---A"
Primary and secondary sort keys – as per tab_sort – for displaying keyword search results

setenv www_sort_field_aut "03---A01---D"
setenv www_sort_field_sub "01---D02---A"
setenv www_sort_field_srs "05---A01---D"
Primary and secondary sort keys - as per tab_sort – for an ACC listing – Author (AUT), Subjects, (SUB) and Series (SRS) as per the above lines..

setenv www_sort_limit 800
The sort limit is a number of up to 6 digits. Its default value is 999999 (that is, no sort limit). If the number of hits returned by the search query exceeds the sort limit, the result set will not be sorted.

Reinstatement of the sort limit has been made because result sets with a large number of hits were sorted by default, which affected performance by increasing the response time.

Note that the maximum number of documents a result set can contain is controlled by the environment variable "set_result_set_limit", either in www_server.conf or in pc_server_defaults. So it would make no sense to define a sort limit greater than "set_result_set_limit". The default value of "set_result_set_limit" is 1000, and its value can be in the range of 1000-20000

setenv www_update_session_id "UPDATE_ALEPH"
In order to update the current session default values from the WWW interface and not from the GUI.

When a non-identified patron accesses **ALEPH** through the Web OPAC, he is assigned the **ALEPH** profile values. **ALEPH** profile values are set in program code, which can be overwritten by a setting up an actual **ALEPH** profile through the GUI ADMIN module. By program definition, the **ALEPH** profile cannot be updated through Web OPAC “option” (preferences). However, if the library defines a setenv www_update_session_id value, when a patron signs in with this ID, he is able to change the **ALEPH** profile through “option” (preference).
The purpose of this environment variable is to provide a tool for changing **ALEPH** profile values, without having to use the GUI Administration function interface.

Sort routines for loans (z36), hold requests (z37), photocopy requests (z38): Sort order – (A)scending, or (D)escending
setenv www_user_z36_sort_routine 00
setenv www_user_z36_sort_order A
Possible routines:

- 00 -z36-due-date
- 01 - z36-sublibrary z36-status z30-collection z36-due-date

setenv www_user_z37_sort_routine 00
setenv www_user_z37_sort_order A
Possible routines:

- 00 - z37-priority z37-request-date z37-open-date z37-open-hour
- 01 - z30-sublibrary z30-item-status z30-collection z37-open-date z37-open-hour

- 02 - z30-sublibrary z30-item-status z30-collection z37-status z37-open-date z37-open-hour

setenv www_user_z38_sort_routine 00
setenv www_user_z38_sort_order A
Possible routines:

- 00 - z38-open-date

setenv www_z36h_max_record_limit 100
Defines the maximum number of Z36h records displayed in the Web OPAC - patron information - Loan history list

setenv www_z37_date_to M002
Defines the end of the time slot when creating hold request.

The variable definition should include one character that describes the time period (D- day, M- month, Y- year) and a 3 digit value for quantity – for example, M003.

This will create a hold request that starts from current date and ends in three months.

If the variable is not defined, or is defined incorrectly, the default "date to" will be three months.

setenv www_z37_recall_type 03
Defines the default recall type for requests placed in the Web OPAC. If the line is missing from www_server.conf, the system defaults the recall type to 01.

The recall type is 01, 02 or 03, where 01 is regular, 02 is rush and 03 is no recall.

Recall is related to item status in tab15.<lng>, col.13 and 14, according to which an item can be recalled or rush recalled. This relates only to the b-cir-13 recall service.

Not all libraries use the recall feature, and not all libraries want every request to automatically be recalled --- some libraries do not want recall at all; some libraries want the recall to be in effect only if the person placing the request comes to the CIRC desk and asks for recall.

Therefore, the need for the default variable.
For libraries that want all requests to automatically be recalls, this variable should be set to ***01***.
For libraries that want to use recall, but only for librarian initiated requests, should be set to ***03***.

setenv syndetics_client_code xxxx
Define client's code which has subscription for Syndetics.
xxxx stands for the customer's code.

www_tab_myshelf_short.<lng>

Location of the table: tab directory of the **ADM** library

Purpose of the table: Brief display format of records in My e-Shelf

Related table(s):

1) tab_expand
2) tab_sort

3) edit_field.<lng>
4) tab01.<lng>
5) tab_sub_lib_sort
6) tab_buf_z403

This table defines which information will be displayed in the columns of the short display in My e-Shelf.

The order of the columns defined in this table determines the order of the display, in conjunction with the myshelf-short-head-2, myshelf-short-head-2-no-course and myshelf-short-body Web OPAC files in the Web OPAC files directory.

These files should have placeholders in sequential order, and should have as many placeholders (from 0300) as the the number of columns defined in this table.

Refer to www_tab_short.<lng> for more details on the options related to this table.

Structure of the table:

Col.1	Group no.
Col. 2	ALPHA
Col. 3	Name of column (20 characters)
Col. 4	Sort key (as defined in tab_sort)
Col. 5	For fixed fields starting position in field from which to take information. 00 for entire field
Col. 6	For fixed fields, number of characters to take for display. 00 for entire field.
Col. 7	Length of field to display for variable fields. Enter 0000 for fixed fields
Col. 8	Identifier of the line to use from edit_field.<lng> when formatting the data. If the field code is "LOC" (for library location) and this position is "L", there is a link from the library name to the items list.
Col. 9	filter subfield code
Col. 10	filter text - together with previous column, defines whether the field should be taken for the column, dependent on: • presence of a particular subfield and content,

- absence of a subfield (where absence is identified as '-' in text)

Col. 11 document record format (FMT) code

Col. 12 first field code

Col. 13-16

Alternate field codes.

Example of the table:

```
! 2           3                4       5  6   7   8 9    10       11   12   13
!-!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!-!!-!!-!!!!-!-!-!!!!!!!!!!-!!-!!!!!-!!!!!-!
!!!!-!!!!!-!!!!!
1 L Author                02---A       00 00 0100 S 9 -          ## 1#### 700##

2 L Title                 03---A       00 00 0100 S              ## 245##

3 L Imprint                            00 00 0100 L              ## 260##

4 L Year                  01---A       00 00      S              ## YR
```

www_tab_self_check

Location of the table: alephe/tab directory

Purpose of the table: Web Self-Check functionality definitions

Related table(s):

1) tab_sub_library.<lng>

This table defines for a particular workstation, or group of workstations, whether or not Web Self-Check functionality can be used.

Only items of the listed sublibraries can be loaned or returned.

- 1: Use this type if you want to activate Web self check-out for items of the sublibrary at this station. If the sublibrary of the item is not listed as type 1 for this workstation, the item cannot be checked out.
- 2: Use this type if you want to activate Web self check-in for items of the sublibrary at this station. If the sublibrary of the item is not listed as type 2 for this workstation, the item cannot be checked-in.

General

You can enter up to 100 sublibraries of types 1 or 2 per IP address.
The sublibraries can be written on one line or on repeated lines, but the same type must be grouped together. The sublibrary code must be filled to 5 characters, and listed with one space between each sublibrary. In other words, if the sublibrary code is only 3 characters, it will then have 2 blanks for fill, and an additional blank for separator before the next sublibrary code.

Structure of the table:

Col.1 Attribute identification, for ip address identification

Col. 2 Attribute type:

- 1 - item can be checked out.
- 2 - item can be checked in.

Cols. 3-7

Sublibrary code

Example of the table:

```
!       1             2   3     4     5     6     7
!!!!!!!!!!!!!!!!!!!!-!-!!!!!-!!!!!-!!!!!-!!!!!-!!!!!>
*.*.*.*               1 #####
                      2
```

www_tab_short.<lng>

Location of the table: tab directory of the library

Purpose of the table: Brief display format in the Web OPAC

Related table(s):

2) tab_expand
3) tab_sort

4) edit_field.<lng>
5) www_server.conf - www_short_max_no_locations
6) www_f_heading related to
7) www_short_max_no_locations and to
8) www_short_max_no_ext_links
9) tab01.<lng>
10) tab_sub_lib_sort
11) tab_buf_z403

This table defines which bibliographic information will display in the columns of the short display in table format in the Web OPAC.

The order of the columns defined in this table determines the order of the display, in conjunction with the short-a-head and short-a-body Web files in the Web OPAC files directory.

These files should have placeholders in sequential order, and should have as many placeholders (from 0300) as the number of columns defined in this table.

Note that the field that is defined for a column can be dependent on the bibliographic record format (FMT). The alternative definitions are assigned the same identifying number in column 1 of this table.
Example:

```
1 L Author     02---A01---A 00 00 0100 S 9 -          BK 1#### 700##
1                                                     SE 260##
1                                                     ## 1#### 700##
```

In the above example the ***Author*** column in the Web OPAC will be taken from fields:
1#### or 700## if the document is a book,
from field 260## if the document is a journal,
and in any other case it will be taken from field 1#### or 700##.

A single column can be set to display more than one field of information, by:
repeating the column number in col. 1,
entering the name of the field in col. 3,
and the field code in col. 12.

For example:

```
2 L Title      03---A01---A 00 00 0100 S               BK 245## 240##
2                                                      ## 245## 240##
2 L Type                               S               ## TYP##
```

The second column (2) contains two fields
the Title field
and the TYP field.
It creates a virtual field TYP indicating the type of record - (Electronic Resource, Web Resource, Map, Kit and so on.), based on coding in
LDR (pos. 06,07),
006 (pos. 00), 007 (pos. 00,01) and
008 (pos. 23)
The ***TYP*** field can be used for indexing and display, depending on the setup of tab_expand. Note that the name "TYP" will ***not*** be displayed. Note too that this option will not work with the special types of fields like **ITM+** and **RANK**.

Note: ! There are special "field codes" for displaying and, in some cases, linking to item information:
EXTL - displays a linkable text for digital objects associated with the record (objects added through the ADAM functionality or through the cataloging record from fields such as the 856). This is similar to the MED functionality. The difference is in the way the digital object is displayed. In the case of MED, icons are displayed while in the case of EXTL a linkable text is displayed.
ITMG -displays message 9052 as link to global holdings display
ITM+ -displays each sublibrary+collection, with link, and with total no. of items / items on loan
ITMH - link to holdings and items, each sublibrary + collection listed on a separate line. Note that the holdings and items display is NOT sensitive to collection, and displays the data at the sublibrary level. Up to 200 sublibraries can be displayed. **Note:** Will only display if there is a holding (HOL) record
LOC+ -like LOC (displays the LOC field, created by expand_doc_bib_loc_disp and using edit_field), + link to relevant holdings.

LOC - displays the LOC field, created by expand_doc_bib_loc_disp and using edit_field
LOW+ - link to remote item information for non ALEPH local systems. The link based on the LOW tag which contains the sub library code of the local library.
MED - displays linkable icons for digital objects associated with the record (objects added through the ADAM functionality or through the cataloging record from fields such as the 856).
PST - like PSTS, for the PST field created by the expand_doc_bib_loc_n and expand_doc_sort_x group of programs (refer to ALEPH System Librarian's Guide/Indexing/Expand Routines chapter) for more information related to this set of programs).
PSTS - similar to LOC+ (displays the PSTS field, created by expand_doc_bib_psts_disp and using edit_field), + link to relevant holdings.
TN - displays a thumbnail associated with objects added through the ADAM functionality.

Note that LOC/PSTS/ITM+/LOC+ are limited either by 2000 characters or by the parameter defined in:

setenv "www_short_max_no_locations"

in www_server.conf. In both cases a message is displayed to indicate that there are more lines: The headings are 9564 for ITM+ and 9553 for LOC, LOC+, PST, PSTS in $aleph_root/error_lng/www_f_heading.

The same applies to **EXTL**: the number of lines is limited either by the variable:

"www_short_max_no_ext_links"

in www_server.conf or to 2000 characters. In both cases a heading is displayed to indicate that there are more lines: $aleph_root/error_lng/www_f_heading - heading 9568.

SYS - Displays the record's system number.

Structure of the table:

Col.1	Group no.
Col. 2	ALPHA
Col. 3	Name of column (20 characters)
Col. 4	sort key (as defined in tab_sort) The first 6 positions define the first sort key. The following 6 positions define the second sort key. Note that **A** and **D** stand for **A**scending and **D**escending
Col. 5	For fixed fields starting position in field from which to take information. 00 for entire field Note that if you want to display "year", you can identify this by code YR in column 12, if you have set expand_doc_yr in tab_expand section WEB-BRIEF.
Col. 6	For fixed fields, number of characters to take for display. 00 for entire field.

Col. 7 Length of field to display for variable fields. Enter 0000 for fixed fields

Col. 8 Identifier of the line to use from edit_field.<lng> when formatting the data. If the field code is "LOC" (for library location) and this position is "L", there is a link from the library name to the items list.

Col. 9 filter subfield code

Col. 10 filter text - together with previous column, defines whether the field should be taken for the column, dependent on:

- presence of a particular subfield and content,
- absence of a subfield (where absence is identified as '**-**' in text)

Col. 11 document record format (FMT) code

Col. 12 first field code

Col. 13-16

Alternate field codes.

Note that within the www_tab_short.<lng> table it is possible to define sort options for the table. For example, in the table example below, clicking on the *Title* (as defined in column 3), will sort the short display according to the options set in columns 4-7 for that line.

Order of Sublibraries with codes **ITM** and **ITM+**:
The first sublibrary to display will be the default sublibrary, set through the profile record, (Z61-BOR-SEARCH-SUBLIBRARY).
The order of the rest sublibraries will be alphabetical.
as set in the table tab_sub_lib_sort
Order of Sublibraries with codes **LOC, LOC+, PSTS**
as set in the table tab_sub_lib_sort

Example of the table:

```
! 2           3                4       5  6   7   8 9    10       11   12   13
!-!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!-!!-!!-!!!!-!-!-!!!!!!!!!!-!!-!!!!!-!!!!!-!
!!!!-!!!!!-!!!!!
1 L Author               02---A01---A 00 00 0100 S 9 -          ## 1#### 700##

2 L Format                                       S              ## TYP## FMT##

3 L Title                03---A01---A 00 00 0100 S              ## 245##

4 L Year                 01---D02---A 00 00      S              ## YR
```

www_x_func

Location of the table: alephe/tab directory

Purpose of the table: X-Server function definitions

Structure of the table:

Col. 1 client version code

Col. 2 function code

Col. 3 program name

Col. 4 parameters

Example of the table:

```
!    1             2                          3
!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!-!!!!!!!!!!!!!!!!!!!!!!!!!!!!!…!!!!!!>
########## BOR-AUTH             www_x_bor_auth
########## BOR-AUTHENTICATE     www_x_bor_auth
########## BOR-BY-KEY           www_x_bor_by_key
########## BOR-COURSE           www_x_bor_course
########## BOR-INFO             www_x_bor_info
```

xsl_creator.<lng>

Location of the table: alephe/tab directory

Purpose of the table: Define labels in any language for use with the online XSL Creator

The online XSL Creator has been enhanced to include labels in different languages. Previously the default label was taken from the XML tag by suppressing the znn and replacing the hyphen ("-") with spaces. For example:

z68_order_number was defaulted to "Order Number".

In order to allow translation to different languages, tab/xsl_creator.<lng> can be used.

Structure of the table:

Col. 1 Original value

Col. 2 Translated value

Z111_index table

Location of the table: alephe/tab directory

Purpose of the table: set for which fields Z111 keywordsa re created so that only required fields are indexed when p-manage-111 batch service is run (build keywords).

Structure of the table:

- Col. 1 field to index (30 characters).

When building the Z111 keywords for search table by activating the batch service p-mamage-111, the keywords are built only for the fields defined in the z111_undex table.

Column 1 can contain the following fields.

Z303-NAME
Z303-SALUTATION
Z303-PROXY-FOR-ID
Z303-PRIMARY-ID
Z308-KEY-DATA
Z304-ADDRESS
Z304-ZIP
Z304-EMAIL-ADDRESS
Z310-BARCODE
Z310-ID
Z13-TITLE
Z13-AUTHOR
Z70-VENDOR-NAME
Z70-VENDOR-CONTACT
Z70-COUNTRY
Z70-MATERIAL-TYPE
Z70-ADDITIONAL-VENDOR-CODE
Z70-CONTACT-5
Z700-REC-KEY
Z700-NAME
Z700-GEN-INFO
Z76-EXTERNAL-BUDGET
Z76-CURRENCY
Z76-NAME
Z76-DEPARTMENT
Z72-VENDOR-ADDRESS

The following is an example of z111_table:

```
!           1
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Z303-NAME
Z303-SALUTATION
Z304-ADDRESS
Z310-BARCODE
Z310-ID
Z70-VENDOR-NAME
Z70-VENDOR-CONTACT
Z70-COUNTRY
Z70-MATERIAL-TYPE
Z70-CONTACT-5
Z72-VENDOR-ADDRESS
Z700-REC-KEY
Z700-NAME
Z700-GEN-INFO
```

Z76-EXTERNAL-BUDGET
Z76-NAME

Index

Abbreviations to be used with numbering 19
Access definitions to WEB and GUI 134
Address types for forms 75
Addresses of sub-libraries for printouts 300
Advanced booking schedule 417
ALEPH environment for running scripts 130
aleph/error_<lng> 40
  check_circ 28
  circ_logger 46
  pc_tree_c1501 123
  www_f_heading 474
aleph/error_<lng>tab
  check_doc 31
aleph_start 102, 103, 460
alephadm 105
ALEPHADM 140
alephe/root
  ALEPH environment for running scripts 130
  aleph_startup 26
  alephe_start 20
  oclc_server_defaults 93, 94, 96
alephe/tab directory
  853_chrono 19
  853_numbering 19
  allowed_languages 26
  job_list 83
  job_list.conf 84
  library_relation 86
  license 89
  marc_country_codes 89
  marc_language_codes 91
  no_utf_mac 92
  no_utf_pc 92
  server_ip_allowed 134
  tab_alephadm.<lng> 140
  tab_base.<lng> 149
  tab_base_count 151
  tab_ccl_boolean_operators 162
  tab_days 174
  tab_export 191
  tab_io_remote 234, 235, 236, 237, 238
  tab_library_group 246
  tab_location_name<lng> 254
  tab_mime_type 267
  tab_month 267
  tab_oclc 272
  tab_roster 284
  tab_sub_library.<lng> 297
  tab_sub_library_address.<lng> 300
  tab_suf_bor_stat 303
  tab_suf_default 303
  tab_suf_lib 304
  tab_version 311
  tab_z403_directory.<lng> 344
  tab_z403_extension.<lng> 344
  tab_z403_info.<lng> 345, 346
  union_global_param 470
  user_function.<lng> 472
  www_f_func 473
  www_tab_self_check 485
  www_x_func 489
  xsl_creator.<lng> 490
alephe/unicode directory
  marc8_xxx_to_unicode 91
  tab_character_conversion_line 163
  unicode_to_filing_01 467
  unicode_to_word_gen 469
alephe_root directory
  pc_server_defaults 98
  www_server.conf 477
Allowed and denied tags for users in cataloging 124, 125
allowed languages *Related table:* pc_tab_expand
Assignment of fields to Direct Indexes for NCIP Server 271
AUT information on cross references 63
Authority lists accessed on defined cataloging tags 132
Automatic opening of monograph-type items in Acquisitions 401
Automatic update of the processing status 415
binding readiness parameters 153
Block codes and restrictions 175
Blocking loans 155
BNA loader 136
Booking Requests delivery configuration 405
Booking Requests pick up list configuration 406
Brief display format in the Web OPAC 486
Brief display format of records in My e-Shelf 484
Brief format of a record in the Search function 121
Brief record structure 333
build_filing_key 252
Captions for links between records 353
cash (z31) sort types in various ALEPH functions 328, 330
Cash transaction types 376
catalog.ini 48
Cataloging OWN permissions and filter 273
Cataloging tables
  check_doc 30
  check_doc (error messages) 31
  check_doc.<lng> 32
  check_doc_field_006 34
  check_doc_field_007 36
  check_doc_field_008 37
  check_doc_field_ldr 38
  check_doc_line 39
  check_doc_line_contents 41

check_doc_mandatory 42
check_doc_new_acc 44
check_doc_new_acc_aut 44
check_doc_tag_text 45
check_doc_unique_index 47
codes.<lng> 47, 49
fix_doc.<lng> 71
formats.<lng> 76
marc_exp.dat 90
pc_tab_acq_fast_cat.<lng> 107
pc_tab_circ_fast_cat.<lng> 109
permission.dat 124, 125
remote_catalog.dat 130
scancode.dat 131
tab_fix 203, 206
tab_merge 261
tab_merge_adv_overlay 263
tab_merge_overlay 265
tab_own 273
tab_preferred 280
tab_tag_text 305
tag_text.dat 465
tagonnew.dat 466
Character conversion tables 91
Charges for ILL incoming requests 82
Charges for ILL outgoing requests 78
Check of pre-defined texts for sub-fields 45
Check routines 30
Check routines for user authorization and payment related to digital rights 52
checks for hold requests 217
Checks on contents of a tag 41
Checks on forbidden errors in a document record 42
Checks on MARC21 tag 006 34
Checks on MARC21 tag 007 36
Checks on MARC21 tag 008 37
Checks on tags 39
Checks on the MARC ldr 38
Checksum programs 170
circ.ini 391, 399
circ_status_holding 474
Circulation tables
cash_09_translate 28
check_circ 28
check_circ_override 29
circ_logger 46
pc_tab_fast_circ 117
tab_block_circ 155
tab_booking 156
tab_bor_address 157
tab_bor_id.<lng> 46, 159, 277, 365
tab_check_circ 166
tab_fast_patron_register 193
tab_hold_request 217
tab_hold_request_form 227
tab_late_return 245
tab_map_privileges 254
tab_photo_request 278
tab_photo_request_form 229
tab_roster 284
tab_sip2_alert 290
tab_sip2_conf 291
tab_sip2_translate 293
tab_z31_sort 328
tab_z36_sort 330
tab14 364
tab15.<lng> 365
tab16 369
tab17 286, 375
tab18.<lng> 376
tab24 390
tab27 391
tab30 393
tab31 394
tab32 396
tab34 398
tab37 402
tab37_booking_delivery 405
tab37_booking_pickup 406
tab37_campus_filter 407
tab37_campus_preferred 408
tab38 409
tab39 410
tab41 412, 413, 415
tab43 417
tab44 419
www_tab_self_check 485
Client definitions to access the pc_server 311
Collection codes and names 411
columns in GUI screens 110
Conditions for the sfx button to not appear 290
Conditions the SFX button will appear in the Web OPAC. 288
Configuration for Advanced Library Statistics (DBS) 174
Controls the order that the roster batch job (p-cir-79) moves requests from institute to institute
tab_roster 284
Conversion of field codes 352
Country codes that may be used in MARC records 89
create links 183
Creation of a virtual field from two or more fields 187
creation of items and holdings 212
Custom brief display format for Web OPAC 476
Database definitions 149
Day codes and abbreviations 174
Default fields for a new record 466
Default tag values when creating records 305
Default values for GUI applications 98
Default values for p-file-96 (Vendor records loader) 136
Default values for p-file-98 (BNA loader) 136
Default values for p-file-99 (Marcive loader) 137
Default values for the Web 477

Default values the YBP loader (p-file-96) 320
Define labels in any laguage for use with the online XSL Creator 490
Defining ILL Unit groups for shared copyright policy 83
Defining the types of global ALEPH Z305 record 254
Delivery times from remote storage 390
Direct indexes 359
Display of bibliographic fields in the Acquisition's Order index 139
Display order of the items 240
DisplayTagInfo 48
Document format definition for GUI/WWW and printouts 56
Due dates, fines & limits 369
EDI incoming messages attributes 53
EDI orders 400
EDI outgoing messages attributes 54
edit_doc_999.<lng> 462
edit_doc_999_aut_<aut_lib>.<lng> 63
edit_field 66, 130
edit_field.<lng> 64
Editing of fields and headings for OPAC display and print 65
End-subfield punctuation 302
environment variables for the OCLC server 93, 94
Error messages linked to circ tables 28
Error messages linked to the circulation log 47
Events definitions 178
Expand object data into the bibliographic record 70
Expand programs 181
expand_doc 67
expand_doc_bib_loc_[n] 60, 122
expand_doc_bib_loc_disp 60, 65, 122, 487
expand_doc_bib_loc_usm 60, 65
expand_doc_bib_psts_disp 122
expand_doc_bib_tab04 352
expand_doc_hld_stmt 211
expand_doc_sort_loc 60
expand_doc_sort_x 122
expand_doc_type 356
expand_doc_type specifications 306
expand_join_simple 187
Extension definitions 267
Extraction of sub-fields for indexing 186
Fast circulation 117
Fast patron registration 193
Field codes and names for indexes 347
Field duplication 184
Field values for the GUI pull down menus 114
Fields for cataloging in Acqisitions 107
Fields for cataloging in Circulation 109
Fields for searching for a record in the ILL module 118
Fields from the object record (Z403) are expanded into the object info. 345
Fields not displayed in the Catalog Editor 162
Fields that should be ignored when checking for duplication 47
Fields that should be ignored when checking new headings in the headings list 44
Fields to be checked against the headings index 258
Fields to be ignored when checking new headings list with check in the AUT library 44
Fields to retain when overlaying (merging) catalog records 265
Fields to retain when replicating z105 messages 342
Filing and normalization routines 194
Filing routines for items 202
Fix_doc definitions 71
fix_doc programs 203, 206
fix_doc_add_pinyin_check_sub9 279
fix_doc_add_pinyin_insert_sub9 279
fix_doc_merge 262
fix_doc_preferred_usm 382
fixed fields 123, 295
form_<lng> 57
Format of the BIB-ADM record 397
Formats for bibliographic display in templates 27
Formats for Cataloging 76
Full document format 58
General invoice payment status 424
Grouping of libraries to share the same batch processes 246
GUI SEARCH search options 120
Headings indexes 356
Hold request daemon configuration 410
hold request forms 227
Hold requests filter table 407
Hold requests send method configuration 391
Holdings summary statement building 211
ILL Borrowing - Patron's Copyright agreement policy 423, 426
ILL request log transaction types 422
ILL tables
ill_bor_charge 78
ill_format_type 79
ill_formats_map.<lng> 81, 233
ill_lend_charge 82
ill_unit_group.<lng> 83
pc_tab_find.<lng> 118
pc_tab_scan.<lng> 119
tab_ill_bib_info 230
tab_ill_bib_key 231, 233
tab_ill_preferred_media 233
tab46.<lng> 422
tab47.<lng> 423, 426
ISO 5426 to Unicode 18
Item fields expanded into the bibliographic record 68
Item History trigger and description 238
Item material types 390
Item reshelving time 364

Item statuses 365
Job daemon scheduling of programs and procedures 83
Job list definitions 84
job_list.conf 84
Joining fields for indexing 189
Language code definitions 26
Language codes that may be used in MARC records 91
Language definitions of CCL Boolean operators 162
Language sensitive tables 58
check_doc.<lng> 32
codes.<lng> 47, 49
edit_doc.<lng> 56
edit_doc_999_aut_<aut_lib>.<lng> 63
edit_field.<lng> 65
edit_paragraph.<lng> 67
fix_doc.<lng> 71
form_description.<lng> 73
formats.<lng> 76
ill_formats_map.<lng> 81, 233
ill_unit_group_map.<lng> 83
pc_tab_acq_fast_cat.<lng> 107
pc_tab_circ_fast_cat.<lng> 109
pc_tab_exp_field.<lng> 114
pc_tab_field_extended.<lng> 116
pc_tab_find.<lng> 118
pc_tab_scan.<lng> 119
pc_tab_sear.<lng> 120
pc_tab_short.<lng> 121
tab_ www_item_desc.<lng> 318
tab_alephadm.<lng> 140, 490
tab_base.<lng> 149
tab_bor_id.<lng> 46, 159, 277, 365
tab_dbs.<lng> 174
tab_delinq.<lng> 175
tab_delivery_locations.<lng> 176
tab_events.<lng> 178
tab_exp_own.<lng> 180
tab_item_history.<lng> 238
tab_label.<lng> 241
tab_location_name<lng> 254
tab_sub_library.<lng> 297
tab_sub_library_address.<lng> 300
tab_tab_short.<lng> 130
tab_type_config.<lng> 306
tab_vat_percent.<lng> 309
tab_z403_directory.<lng> 344
tab_z403_extension.<lng> 344
tab_z403_info.<lng> 345, 346
tab00.<lng> 347
tab01.<lng> 349
tab05.<lng> 353
tab15.<lng> 365
tab18.<lng> 376
tab25.<lng> 390
tab33.<lng> 397
tab40.<lng> 411
tab45.<lng> 420
tab46.<lng> 422
tab47.<lng> 423, 426
tab48.<lng> 424
user_function.<lng> 472
www_f_heading 474
www_f_sort_heading.<lng> 475
www_tab_myshelf_short.<lng> 484
www_tab_short.<lng> 486
xsl_creator.<lng> 490
Library open hours 375
License parameters 89
Link back from the AUT record 358
linked records 181
Links between Authority Headings 355
Links from BIB Headings to AUT Records 381
list of browsers/versions (user agents) that do not fully support UTF-8. 92
List of library codes and sublibrary codes 297
List of remote catalogs for Z39.50 Update Service 130
List of tags 349
LKR field 353
Locating a record in other libraries 252
Location names for Z39.50 server 254
Logical bases (record counters) in Web and GUI browse list 151
Lost material costs 398
Mapping bibliographic values into the relevant holdings and/or items values 215
Maps directories of OBJECT files between libraries 344
Maps extensions to mimetypes and icons for Web OPAC 344
Marcive loader 137
Match specifications for checking input file against the database 259
Match specifications for importing files 257
Matching fields for p-serial-52 (import 85XX records) 135
Matching of BIB library's Headings (ACC) index to AUThority records 145
Merge routines 261
Messages between libraries 340
Messages to the patron in the WEB OPAC 474
Minimum user level for override 29
Month codes and abbreviations 267
Moving records 268
Multilingual authorities links 382
NACO Normalization Rules 194
Names of months and seasons 19
Navigation Window 123, 183
Node environment definitions 20
Non-filing indicator 351
Non-high lighted words 354
OCLC Loader 257, 259
OCLC server configuration table 272
of ALEPH codes to Dublin Core XML elements 363
Order of HOL records to be expanded into the BIB record 206

Order transaction types for the order log 420
Overdue letter intervals 396
OWN field values 180
p_file_96 136
p-acq-04 140
Paragraphs for displaying bibliographic information 67
Parsing call numbers for label printing 242
Patron address periods 157
Patron registration renewal 393
Patron statuses 394
p-bind-01 153
p-bind-02 153
pc_server_defaults 22, 103, 478, 482
pc_tab/catalog directory of the **BIB** library
remote_catalog.dat 130
pc_tab/catalog directory of the library
codes.<lng> 47, 49
fix_doc.<lng> 71
formats.<lng> 76
marc_exp.dat 90
permission.dat 124, 125
scancode.dat 131
tag_text.dat 465
tagonnew.dat 466
p-cir-08 166
p-cir-10 166
p-cir-12 367
p-cir-13 368, 450
p-cir-22 367
p-cir-24 407, 408
p-cir-50 396
p-cir-77 167
Period that requested items will be held on the hold shelf 419
p-export-01 191
p-file-08 77
p-file-96 320
Load YBP records 259
p-file-98 136
p-file-99 137
Load MARCIVE records 257, 259
photo request forms 229
Photocopy request daemon configuration 412, 413
Photocopy requests 278, 409
pick up list configuration for hold requests 402
Pinyin 279
p-manage-11 30, 43
p-manage-19 435
p-manage-25 43
p-manage-27 350
p-manage-31 86, 88
p-manage-33 30, 43
p-manage-36 257, 259
p-manage-38 281
p-manage-50 212, 215
p-manage-500 216, 240, 256
p-print-08 130
Pre-defined text options for sub-fields 465
Preferred supplier list for hold requests 408
Prefixes for budget, vendor and ILL supplier codes 173
prefixes for spine labels 241
p-ret-01 195, 350
p-ret-21 195, 349
Print method for different printouts 73
printing bib information in columnar form 130
Processing script for fix_doc_do_file_08 and p-file-08 77
Programs for building buf_z403 160
p-serial-52 135
pw_library 473
Re-direction of directories for sharing between libraries 98
Relation between libraries by library type 86
Remote conversion programs 108
Remote library specifications 236, 238
Retrieval definitions for p-export-01 191
RLIN Loader 255
sear_brief_edit 121, 122
Self-check configuration 292
Self-check field alert definitions 290, 291
Self-check field definitions 293
SFX 288
Shared user definition 304
Shared user definitions 303
Smart barcode export configuration 192
sort
hold list 100
loan list 100
photocopy requests 100
Sort options for items 320
Sort options for WEB OPAC brief formats 475
Sort sub-fields for expand 190
Sort tables
tab_expand_sort 190
tab_sort 294
tab_sub_lib_sort 295, 296
tab_z30_sort 320
www_f_sort_heading 475
Sorting 294
Sorting sub-libraries 296
Specifications for creation of ILL Record's format field 79
Specifications for extracting records 281
Split field definitions 190
Splits external fields 185
Startup definitions 26
Sub-fields for Open Form and List of fields (F5) 90
Subject for a form sent by e-mail 73
Sublibrary relationships based on station ID / IP address 142
Sub-library sensitive field values for the GUI pull down menus 116
Sublibrary specific and generic vendor fields 309
Sublibrary translation to Campuses 28

Switch settings for different aspects of the system 427
tab directory of the **ADM** library
bib_format 27
cash_09_translate 28
check_circ_override 29
edi_in_attr 53
edi_out_attr 54
form_sub_library_address 75
pc_tab_exp_field.<lng> 114
pc_tab_fast_circ 117
pc_tab_field_extended.<lng> 116
pc_tab_find.<lng> 118
pc_tab_scan.<lng> 119
tab_85x_import 135
tab_acq_index 139
tab_attr_sub_library 142
tab_binding 153
tab_booking 156
tab_bor_address 157
tab_check_circ 166
tab_checksum 170
tab_code_prefix 173
tab_dbs.<lng> 174
tab_delinq.<lng> 175
tab_delivery_locations.<lng> 176
tab_exp_own.<lng> 180
tab_filing_call_no 202
tab_hld_stmt 211
tab_hold_request 217
tab_hold_request_form 227
tab_item_history.<lng> 238
tab_label.<lng> 241
tab_label_parse 242
tab_late_return 245
tab_map_privileges 254
tab_ncip_record_id 271
tab_photo_request 278
tab_photo_request_form 229
tab_rfid 155, 207, 273, 276, 282
tab_rfid_translate 283
tab_roster 284
tab_sip2_alert 290
tab_sip2_conf 291
tab_sip2_translate 293
tab_vat_percent.<lng> 309
tab_vendor_sub_lib 309
tab_www_item_desc.<lng> 318
tab_z105_z30 342
tab_z30 246, 249
tab_z31_sort 328
tab_z311 343, 346
tab_z36_sort 330, 331
tab14 364
tab15.<lng> 365
tab16 369
tab17 286, 375
tab18.<lng> 376
tab24 390
tab25.<lng> 390
tab27 391
tab30 393
tab31 394
tab32 396
tab34 398
tab35 400
tab37 402
tab37_booking_delivery 405
tab37_booking_pickup 406
tab37_campus_filter 407
tab37_campus_preferred 408
tab38 409
tab39 410
tab40.<lng> 411
tab41 412, 413, 415
tab42 415
tab43 417
tab44 419
tab45.<lng> 420
tab46.<lng> 422
tab47.<lng> 423, 426
tab48.<lng> 424
www_tab_myshelf_short.<lng> 484
tab directory of the **BIB** and **ADM** libraries
pc_tab_acq_fast_cat.<lng> 107
pc_tab_circ_fast_cat.<lng> 109
tab_mapping 255
tab_z30_sort 320
tab33.<lng> 397
tab directory of the **BIB** and **AUT** libraries
pc_tab_cat_conv 108
pc_tab_sear.<lng> 120
tab_locate 252
tab_service 284
tab_subfield_punctuation 302
tab_word_breaking 314
tab directory of the **BIB** library
check_doc_field_006 34
check_doc_field_007 36
check_doc_new_acc_aut 44
drm_routines 52
edit_doc_999_aut_<aut_lib>.<lng> 63
expand_doc_bib_z30 68
expand_doc_bib_z403 70
form_description.<lng> 73
pc_tab_col.<lng> 110
tab_abbrev 138
tab_aut 145
tab_buf_z403 160
tab_cat_hidden_fields 162
tab_doc_publish 177
tab_fix_local_notes.conf 206
tab_fix_z103 208
tab_font_publish 210
tab_hol_mapping 216
tab_item_list_order 240
tab_itm_mapping 240
tab_mapping_mng500 256
tab_match_script 259
tab_move_record 268

tab_pinyin 279
tab_publish 281
tab_sfx 288
tab_sfx_denied 290
tab_sub_lib_sort 295, 296
tab_type_config.<lng> 306
tab_weights 311
tab_xyz 319
tab_z0101 333
tab_z0101_text 334
tab_z121 342
tab_z403_directory.<lng> 344
tab05.<lng> 353
tab06 354
tab07 355
tab11_aut 357
tab12 363
tab20 381
tab22 384
union_preferred 471
www_f_sort_heading.<lng> 475
www_f_tab_short_999 476
tab directory of the **ILL** library
ill_bor_charge 78
ill_format_type 79
ill_formats_map.<lng> 81
ill_lend_charge 82
ill_unit_group.<lng> 83
tab_ill_bib_info 230
tab_ill_bib_key 231
tab_ill_preferred_media 233
tab directory of the library
check_doc 30
check_doc.<lng> 32
check_doc_doc 33
check_doc_field_008 37
check_doc_field_ldr 38
check_doc_line 39
check_doc_line_contents 41, 146
check_doc_mandatory 42
check_doc_new_acc 44, 97
check_doc_tag_text 45
check_doc_unique_index 47
edit_doc.<lng> 56
edit_doc_777 57
edit_doc_999.<lng> 58
edit_field.<lng> 65
edit_paragraph.<lng> 67
form_print_method 73
path_convert 98
pc_tab_short.<lng> 121
tab_alephadm.<lng> 140, 490
tab_events.<lng> 178
tab_expand 181
tab_expand_duplicate_field 184
tab_expand_external 185
tab_expand_extract 186
tab_expand_join 187
tab_expand_join_simple 189
tab_expand_sort 190
tab_expand_split 190
tab_filing 194
tab_fix 203, 206
tab_match 257
tab_match_acc 258
tab_merge 261
tab_merge_adv_overlay 263
tab_merge_overlay 265
tab_own 273
tab_preferred 280
tab_sort 294
tab_sub_library_address.<lng> 300
tab_tab_short.<lng> 130
tab_tag_text 305
tab_z103 336
tab_z105 340
tab_z105_filter 342
tab00.<lng> 347
tab01.<lng> 349
tab02 351
tab04 352
tab100 427
tab11_acc 356
tab11_ind 359
tab11_word 360
www_tab_short.<lng> 486
tab directory of the **USR** library
ldap.conf 85
tab_block_circ 155
tab_bor_id.<lng> 46, 159, 277, 365
tab_fast_patron_register 193
tab/import directory of the **BIB** library
generic_fix 77
tab_96_def 135, 490
tab_98_def 136
tab_99_def 137
tab_hol_item_create 212
tab_hol_item_map 215
tab_yankee_def 320
tab_attr_sub_library 323
tab_base_count 150
tab_booking 104
tab_bor_id.<lng> 22
tab_character_conversion 205
tab_character_conversion_line 53, 54, 196
tab_filing 202, 348
tab_locate 252
tab_match 259
tab_own 208
tab_service 61
tab_sfx 288
tab_sfx_denied 290
tab_subfield_punctuation 133
tab_z311 459
tab100 22
tab11_ind 258
tab16 22
tab20 63
tab35 55
tab37 24

tab38 24
Text codes list and definitions for brief record (Z0101) 334
Time and rate for fine method "F" 245
Translation of ILL item type code 81, 233
two-level vendor 309
ue_01 184
ue_06 367, 410, 412
ue_08 382
Unicode character conversion equivalencies 469
Unicode character conversion equivalencies for sorting 467
unicode_to_filing 467
Union deduplication parameters 470
User authorization definitions and codes 472
UTIL A/17/1 369
UTIL E/11 23
UTIL E/3 255
UTIL E/6 24, 368
UTIL E/8 21, 145, 382
UTIL G/2 101, 441, 458, 462
UTIL H/1 462
UTIL H/1/10 150
UTIL M/7 124, 466
UTIL Y/6 98
UTIL/F/1/21 470
Valid tags and aliases for the Cataloging Client 48
Validation messages (system driven) for check_doc programs 31
VAT percentages 309
Web self check functionality definitions 485
web service search engine 343
Weight factors for determining candidate records 312
Weighting table 471
when a message from one library to another will **not** be created following a Z30 update 342
Word breaking procedures. 314
Word indexes 360
www_server.conf 21
X Server function definitions 489
XML_XSL 73
XYZ 319
YBP loader 320
Z01 145, 349, 382
Z103 355
Z30-MATERIAL 390
Z311 table 343