USER DOCUMENTATION

How to Set Up an ALEPH Union Catalog
Table of Contents

1 INTRODUCTION

1.1 What is a Union Catalog

1.2 How is a Union Catalog Different From a Standard Catalog?
   1.2.1 Duplicate Records
   1.2.2 Embedded Holdings
   1.2.3 Ongoing Loads

1.3 About This Document

2 EQUIVALENT RECORDS

2.1 What are Equivalent Records?

2.2 Storing Equivalent Records

2.3 Equivalency Algorithm
   2.3.1 Phase One - Candidate Selection
   2.3.2 Server Tables
   2.3.3 Library Tables
   2.3.4 Direct Indexes
   2.3.5 Keyword Indexes

2.4 Phase Two – Record Matching
   2.4.1 Server Tables
   2.4.2 Library Tables
   2.4.3 Assigning Weights for Record Matching
   2.4.4 List of Common Serial Titles
   2.4.5 Additional Table Settings

2.5 Phase Three - Selecting the Preferred Record
   2.5.1 Server Table
   2.5.2 Library Tables

2.6 Building Equivalent Record Tables
   2.6.1 Batch Processes
   2.6.2 Workflow
   2.6.3 Differences between p_union_02 and p_union_04
   2.6.4 Miscellaneous Tables
   2.6.5 Triggers

3 OPAC FUNCTIONALITY

3.1 Enabling Union Catalog Bases In Your OPAC

3.2 Configuring Your OPAC Interface for Union Catalog
   3.2.1 Union Catalog Settings in www_server.conf
   3.2.2 Merged Display
   3.2.3 Displaying Individual Records
4 EMBEDDED HOLDINGS

4.1 Location Table
4.1.1 Tab_locations [data_tab]
4.1.2 Oracle Tables
4.1.3 Indexing and Displaying Locations

4.2 Displaying Embedded Holdings
4.2.1 expand_doc_union_holding
4.2.2 expand_doc_union_holdings_cdl

4.3 Circulation Status
4.3.1 Using z39.50
4.3.2 Using the X-server
4.3.3 Union Catalog Setup
4.3.4 Local Library Setup

5 LOADING RECORDS INTO A UNION CATALOG

5.1 Setup and Workflow
5.1.1 Directory Structure
5.1.2 Configuration Tables

5.2 Record Conversion and Pre-processing
5.2.1 Workflow
5.2.2 Mandatory Fields
5.2.3 Record Validation
5.2.4 Batch Processes

5.3 Record Loading

5.4 Batch Processes
5.4.1 Other Processes
1 Introduction

1.1 What is a Union Catalog

A Union Catalog is a library catalog that contains the holdings of more than one library. The Union Catalog often serves as a utility for resource discovery, allowing users to search the catalogs of multiple institutions. Frequently, the Union Catalog will offer some form of interlibrary loan so that users can request the resources they identify.

1.2 How is a Union Catalog Different From a Standard Catalog?

In ALEPH, a Union Catalog differs from a standard library catalog in a number of significant ways. A Union Catalog:

- Contains the holdings of multiple libraries and thus contains multiple records for titles that are held by multiple institutions.
- Does not contain item records.
- Has embedded holdings instead of linked holdings records.
- Characterized by frequent loads of records from contributing libraries.

1.2.1 Duplicate Records

ALEPH uses a sophisticated algorithm that it developed in conjunction with the California Digital Library to identify duplicate records. In the OPAC, result sets are de-duplicated using pre-constructed record equivalency tables and records are merged on the fly for display.

1.2.2 Embedded Holdings

Bibliographic records in the Union Catalog contain embedded holdings statements. These embedded holdings generate a location display that includes Campus/Location, Library/Sublocation and call number. In addition, the Union Catalog offers linking to the functionality for displaying up-to-date circulation status information from the local library. Circulation status can be obtained using Z39.50 or the Ex Libris X-server.

1.2.3 Ongoing Loads

Libraries that contribute records to a Union Catalog need to constantly update their libraries holdings in the Union Catalog. The result is that they send frequent batches of new and updated records. These batches need to be processed efficiently. ALEPH has a set of batch processes that convert the incoming records and load them into the Union Catalog.
1.3 About This Document

This document describes in detail how to set up the various components of an ALEPH Union Catalog including the special features mentioned above. The document focuses on configuration and functionality that is available from version 16.02 and later. Please see earlier versions of the document to configure earlier versions of the product.

2 Equivalent Records

2.1 What are Equivalent Records?

Equivalent records are bibliographic records that describe the same bibliographic entity. Typically, equivalent records will have been contributed by different Union Catalog members and thus will have identical or very similar bibliographic data but very different holdings data. [In the Union Catalog, holdings data is embedded in the bibliographic record]. ALEPH uses an algorithm that was developed in conjunction with Melvyl (California Digital Library), to identify equivalent records. Equivalent records are not physically merged - instead, lists of pre-built equivalencies are stored in Oracle tables and these tables are used to de-duplicate retrieval sets and merge equivalent records on the fly.

2.2 Storing Equivalent Records

There are two Oracle tables that are used for storing equivalent records, the Z120 record and the Z127 record. Each record in the catalog has its own Z120 and Z127 record. For each record in the database, the Z120 record stores a complete list of records that are equivalent to it, as well as a single record that is considered the preferred record. Batch processes create the record equivalency table and maintain it so that whenever a bibliographic record is updated its equivalencies are rebuilt – both the list of records that it is considered equivalent to and the list or records that contain it as an equivalent record are updated.

The Z120 table contains the following data elements: system number, total number of equivalent records, system numbers of all equivalent records, system number of preferred record, update flag. Z120 records are built after the initial data load (using p_union_01 and p_union_02) and then updated each time the record is updated. The process that updates Z120 records is called p_union_04.

Illustration 1 – Z120 table definitions for versions 14.2 –15.5

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 Z120-REC-KEY.</td>
<td></td>
</tr>
<tr>
<td>03 Z120-DOC-NUMBER</td>
<td>PICTURE 9(9).</td>
</tr>
<tr>
<td>02 Z120-REC-KEY-1.</td>
<td></td>
</tr>
<tr>
<td>03 Z120-PREFERRED-DOC-NUMBER</td>
<td>PICTURE 9(9).</td>
</tr>
<tr>
<td>02 Z120-UPDATE-FLAG</td>
<td>PICTURE X(1).</td>
</tr>
<tr>
<td>02 Z120-SAME-NO-LINES</td>
<td>PICTURE 9(3).</td>
</tr>
<tr>
<td>02 Z120-SAME OCCURS 100.</td>
<td></td>
</tr>
<tr>
<td>03 Z120-SAME-DOC-NUMBER</td>
<td>PICTURE 9(9).</td>
</tr>
</tbody>
</table>
The Z127 record, on the other hand, stores only the system number of a single record and its preferred record.

### 2.3 Equivalency Algorithm

The equivalency-building algorithm goes through three phases in the process of determining equivalent records and building the equivalency table (the Z120 table). They are:

- Candidate Selection
- Record Matching
- Preferred Record Selection

Each of these phases is controlled by a combination of programs and configuration tables. There is one central configuration table that controls which combination of programs and tables will be used – this table is located in alephe_tab and is called union_global_param.

Illustration 3: Union_global_param [alephe_tab]

| USM90  | B candidate_prog | union_candidate_cd1 | 100 |
| USM90  | B match_prog    | union_match_cd1     |     |
| USM90  | B preferred_prog| union_preferred_cd1 |    |
| USM90  | B normalize_prog| union_normalize_cd1 |   |
2.3.1 Phase One - Candidate Selection

During the Candidate Selection phase of the equivalency building process, candidates are retrieved up to a set number - if more than the set number of records is retrieved, adding the year of publication or place of publication to the search refines the set. In versions 14.2 through 15.5 candidate limit cannot be larger then 100, in version 16.02 and later, it can be up to 500. The exact number is set in the union_global_param table, in Column 5 of the line that calls the candidate program.

The candidate phase of the equivalency process utilizes three direct indexes - LCCN, ISSN and ISBN and three keyword indexes – WTL, Normalized Title, WYR, year and WPL place of publication – there is more information about setting up these indexes in section 2.3.

The following indexed fields are used to retrieve candidate records:

- LCCN – 010 subfields a and z
- ISSN or ISBN – 020 $$a and $$z and 022 $$a $$y and $$z
- Normalized Main title – 245 $$a,b,n,p
- Year of Publication or Place of Publication – 008 position 7-10 or 260 $$c/260 $$a

Note that there is an “or” relationship between the first three search terms (LCCN or ISSN/ISBN or title). There is an and relationship between the first set and the fourth term ((LCCN or ISSN/ISBN or title) and year of publication/place of publication). Year of publication is used to refine the candidate search for non-serial records; place of publication is used to refine the candidate search for serial records.

2.3.2 Server Tables

The union_global_param table controls the individual components of the equivalency algorithm and is located in the alephe_tab directory with other shared tables. In the table, programs and parameters are set for each of the phases of the equivalency building process. Currently there is only one program that can be called for the candidate retrieval phase of the equivalency-building program, it is union_candidate_cdl. Thus, the following line should be entered in union_global_param for your Union Catalog library.

Illustration 4 – union_global_param [alephe_tab]

| USM90 B candidate_prog | union_candidate_cdl | 500 |

The fifth parameter, number of candidates, can be set anywhere between 100 and 500. The number sets the limit for number of candidates that the system will check.
2.3.3 Library Tables

The candidate retrieval phase of the record equivalency algorithm uses library indexes to retrieve potential duplicates. Thus, the Union Catalog library must have the appropriate indexes, direct and keyword, set up in its indexing tables.

2.3.4 Direct Indexes

The ISSN, ISBN, and LCCN fields need to be indexed as direct indexes. They should go through expand_doc_extract so that each subfield is indexed separately. They must also be normalized so that prefixes and suffixes are deleted and hyphens are removed. It is especially important that parentheses be removed by the filing routine. Failure to remove parentheses can result in errors during the p_union_02 and p_union_04 processes.

Here are direct index specific settings that should be found in all Union Catalog libraries.

Illustration 5 – tab11_ind [data_tab]

| a010 | 010 |
| z010 | 010 |
| a020 | 020 |
| z020 | 020 |
| a022 | 022 |
| y022 | 022 |
| z022 | 022 |

Illustration 6 – tab_expand [data_tab]

| INDEX | expand_doc_extract |

Illustration 7 – tab_expand_extract [data_tab]

| 010## a a010 |
| 010## z z010 |
| 020## a a020 |
| 020## z z020 |
| 022## a a022 |
| 022## y y022 |
| 022## z z022 |

Illustration 8 – tab00.eng [data_tab]

| H 010 IND 70 00 00 LCCN |
| H 020 IND 71 00 00 ISBN |
| H 022 IND 72 00 00 ISSN |
2.3.5 Keyword Indexes

The main title (245) must be sent to a keyword index called NTL. This index is used for title searching in the Candidate phase of equivalency building. Before indexing a title, an expand program called expand_doc_ntl normalizes the title by stripping initial articles (using non-filing indicators), stripping punctuation and diacritics and compressing spaces. Finally, the first twenty-five characters are indexed as a single keyword. The expand_doc_ntl program uses filing routine 75 to normalize data in the bibliographic record – make sure that this routine is defined in your tab_filing.

Illustration 9 – tab_filing

```plaintext
! * 70 - LCCN normalization for LCCN index to be used by
! * Equivalency building program and union_match_cdl_mo
70 del_subfield
70 compress
70 compress_blank
70 non_numeric
! * 71 - ISBN normalization for ISBN index to be used by
! * Equivalency building program and union_match_cdl_mo
71 isbn
! * 72 - ISSN normalization for ISNN index to be used by
! * Equivalency building program and union_match_cdl_se
72 issn
```

Illustration 10 – tab11_word [data_tab]

```
| NTL## | 03 | NTL |
```

Illustration 11 – tab_expand [data_tab]

```
| WORD          | expand_doc_ntl |
```

Illustration 12 – tab00.eng

```
| H NTL | W-035 | 00 | 06 | Normalized Title |
```

Illustration 13 – tab_filing

```plaintext
! * Reserved for equivalency building program – used to
create the normalized title (NTL)
74 del_subfield
74 non_filing
74 to_blank          ! @$%^&*()_+-={\[\]:"<>?,./~`
74 char_conv          FILING-KEY-01
74 compress_blank
74 first_25
74 to_lower
```
In addition to the NTL index, libraries must have two additional key word indexes defined: Word from Publisher and Year of Publication. The former should be called WPL and the latter WYR.

2.4 Phase Two – Record Matching

The second phase in the equivalency building process is the record matching phase. During this phase, all candidate records are compared and those found to be equivalent are recorded in the Z120 record. While much of this process is hard-coded, there are a number of configuration tables involved and some configuration. For more information about the matching algorithm see the document “Matching Algorithm”.

2.4.1 Server Tables

The following line should be entered in the server table union_global_param. This table controls the individual components of the equivalency algorithm. At this time, union_match_cdl is the only program that can be called for this phase in the equivalency-building algorithm.

Illustration 14 – union_global_param [alephe_tab]

| USM90 B match_prog | union_match_cdl |

2.4.2 Library Tables

The program union_match_cdl uses three configuration tables, all three must be located in the data_tab directory of your Union Catalog library:

- tab_cdl_mo_weights
- tab_cdl_se_weights
- tab_com_tit_cdl

Two of these tables set the record equivalency threshold as well as points that are assigned for each phase of the record matching process. These are set separately for serial and non-serial records. The third table lists common serial titles for the library, common titles receive fewer points for title matches than other titles.

Note that the equivalency algorithm permits there to be matches between all formats except Serials. Serial records can only match other Serial records, however, records with the format book can match records with the format music, computer file, etc.

2.4.3 Assigning Weights for Record Matching

Each step in the record matching phase of the equivalency-building algorithm compares data in the base record with data in every potential match or candidate. A complete list of fields that are compared is available in the document “Matching Algorithm”. A partial list is shown below. The tab_cdl_mo_weights and tab_cdl_se_weights tables contain a brief description of each step in the serial and non-serial algorithm. Each step is accompanied by a number, the number is the number of points that is assigned. Text in the first column is fixed. Both tables are
located in the data_tab directory of the Union Catalog database. Record equivalency building processes need to be rerun after changes are made to either of these tables.

Illustration 15 - tab_cdl_mo_weights [data_tab]

<table>
<thead>
<tr>
<th>treshold</th>
<th>875</th>
</tr>
</thead>
<tbody>
<tr>
<td>010aa match</td>
<td>200</td>
</tr>
<tr>
<td>010az match</td>
<td>100</td>
</tr>
<tr>
<td>010zz match</td>
<td>050</td>
</tr>
<tr>
<td>010aa mismatch</td>
<td>-470</td>
</tr>
<tr>
<td>010az mismatch</td>
<td>-050</td>
</tr>
<tr>
<td>020aa</td>
<td>085</td>
</tr>
<tr>
<td>020az</td>
<td>030</td>
</tr>
<tr>
<td>020z#</td>
<td>010</td>
</tr>
<tr>
<td>020mismatch</td>
<td>-225</td>
</tr>
<tr>
<td>date exact match</td>
<td>200</td>
</tr>
<tr>
<td>date within 2</td>
<td>-025</td>
</tr>
<tr>
<td>date mismatch</td>
<td>-250</td>
</tr>
</tbody>
</table>

Illustration 16 - tab_cdl_se_weights [data_tab]

<table>
<thead>
<tr>
<th>treshold</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>010aa match</td>
<td>200</td>
</tr>
<tr>
<td>010az match</td>
<td>100</td>
</tr>
<tr>
<td>010zz match</td>
<td>050</td>
</tr>
<tr>
<td>010aa mismatch</td>
<td>-470</td>
</tr>
<tr>
<td>010az mismatch</td>
<td>-050</td>
</tr>
<tr>
<td>022aa match</td>
<td>200</td>
</tr>
<tr>
<td>022ay match</td>
<td>100</td>
</tr>
<tr>
<td>022az match</td>
<td>050</td>
</tr>
<tr>
<td>022yy match</td>
<td>050</td>
</tr>
<tr>
<td>022yz match</td>
<td>030</td>
</tr>
<tr>
<td>022zz match</td>
<td>010</td>
</tr>
<tr>
<td>022aa mismatch</td>
<td>-250</td>
</tr>
</tbody>
</table>

2.4.4 List of Common Serial Titles

The list of common serial titles is consulted during the matching stage of the CDL equivalency building process. Title matches in serial records that have titles that are considered “common” receive significantly fewer points than title matches for other serial records (135 instead of 600). The table lists “common” titles in a given database or Union Catalog. If the contents of the table change, p-union-02 should be rerun.

Illustration 17 - tab_com_tit_cdl [data_tab]

<table>
<thead>
<tr>
<th>ANNUAL REPORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANNUAL REPORT FOR</td>
</tr>
<tr>
<td>ANNUAL REPORT FOR THE FISCAL YEAR ENDED MARCH</td>
</tr>
<tr>
<td>BIENNIAL REPORT</td>
</tr>
<tr>
<td>BULLETIN</td>
</tr>
<tr>
<td>CALENDAR</td>
</tr>
</tbody>
</table>
2.4.5 Additional Table Settings

There are two additional settings that must be in place for the equivalency building processes to work. The first setting is in tab_expand. Tab_expand has a section that lists expand programs that are to be called by the two equivalency building programs, p_union_02 and p_union_04. Note that both p_union_02 and p_union_04 use the UNION-02 section. These settings should not be changed.

Illustration 18 - tab_expand [data_tab]

| UNION-02 | expand_doc_extract |
| UNION-02 | expand_doc_ntl |

The second table that needs to be set up is tab_filing. Five filing routines are used by the equivalency building processes, they are: 70-75. Routine 74 is used to normalize the short title (first 25 characters) for the candidate phase of the routine, routine 75 is used to normalize all text fields during the match phase of the routine, routine 70-73 are used to normalize numeric fields (010,020 and 022) for indexing and during the match phase of the routine.
Illustration 19 - tab_filing [data_tab]

```plaintext
!* The following six routines (70, 71, 72, 73, 74, 75) are needed for !* Union Catalog and Union Catalog
!* 70 - LCCN normalization for LCCN index to be used by Equivalency
!* building program and union_match_cdl_mo
70  del_subfield
70  compress
70  compress_blank
70  non_numeric
!* 71 - ISBN normalization for ISBN index to be used by Equivalency
!* building program and union_match_cdl_mo
71  isbn
!* 72 - ISSN normalization for ISNN index to be used by Equivalency
!* building program and union_match_cdl_se
72  issn
!* SID index
73  del_subfield
73  compress
73  compress_blank
73  cut_prefix
!* Reserved for equivalency building program - used to create the
!* normalized title (NTL)
74  del_subfield
74  non_filing
74  to_blank  !@#$%^&*()_+-={}[\]^`~`
74  char_conv  FILING-KEY-01
74  compress_blank
74  first_25
74  to_lower
!* Reserved for equivalency building program - used to normalize
!* all text fields in the record during the match phase of
!* equivalency building
75  del_subfield
75  to_upper
75  suppress
75  numbers
75  compress
75  to_blank  !@#$%^&*()_+-={}[\]^`~`
75  expand_num
75  non_filing
75  pack_spaces
75  char_conv  FILING-KEY-01
```

2.5 Phase Three - Selecting the Preferred Record

The third phase of the equivalency building process is selecting the preferred record. The program for selecting preferred records is identified in the union_global_param table in alephe_tab and the routine for selecting preferred records is configured in a single configuration table, union_preferred. Currently there is only one program available for preferred record selection, it is union_preferred_cdl.

2.5.1 Server Table

The following line should be entered in the server table union_global_param. This table controls the individual components of the equivalency algorithm.
2.5.2 Library Tables

The table union_preferred, located in the data_tab directory of your library, configures the basis for selecting a preferred record from a set of equivalent records. It does so by assigning points for field presence, and/or subfield or fixed field values. After each record in a set of equivalent records gets a weight, the record with the greatest weight becomes the preferred record.

Illustration 21 – union_preferred [alephe_tab]

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDR F05-01</td>
<td>EQUAL</td>
<td>d -10</td>
</tr>
<tr>
<td>LDR F17-01</td>
<td>NOT-EQUAL</td>
<td>1,2,3,4,5,7,8,u,z</td>
</tr>
<tr>
<td>040 a</td>
<td>EQUAL</td>
<td>DLC</td>
</tr>
<tr>
<td>100##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>110##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>111##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>130##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>24###</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>6####</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>700##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>710##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>711##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>730##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>800##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>810##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>830##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
<tr>
<td>880##</td>
<td>PRESENT</td>
<td>001</td>
</tr>
</tbody>
</table>

2.6 Building Equivalent Record Tables

The three steps described above, candidate record selection, record matching and preferred record selection, are performed during equivalency building by the batch processes p_union_02 and p_union_04. In addition to those two basic processes, there are a number of other processes that must be run.

2.6.1 Batch Processes

There are a total of five processes for building and maintaining record equivalencies.

1. p_union_01 – Run after initial conversion and data load. Builds a Z120 record for each bibliographic record. Parameters are the database name. After p_union_01 is run, each record is equivalent to itself but not to any other record.

   $aleph_proc/csh –f p_union_01 USM90
2. **p_union_02** – Run after database has been indexed. Can be run with multiple processes. Parameters are database name, start number, end number, number of processes, mode - batch/test. This process locks the library. P_union_02 should always be run on the entire database - it should not be run on parts of a database. Doing so will result in inconsistencies in the record equivalency tables.

Prior to version 16.02 there was a parameter - rebuild links

```bash
$aleph Proc/csh -f p_union_02 USM90,000000000,999999999,B
```

3. **p_union_04** – Ongoing process, should be run in daily mode or each time records have been loaded using p_union_13. This process rebuilds record equivalencies for records that have been updated [identified by Z120 record flagged N for update.] From version 16.02 and later, p_union_04 is started by p_union_13 unless it is already running, it no longer has to be restarted each night.

4. **create_z127** – From version 15.2 and later, it is necessary to run this process to create Z127 records after running p_union_02 to create Z120 records. The process does not have any parameters – the syntax for running it is: csh –f $aleph Proc/create_z127

5. **load_z127_to_mem** – After Z127 records have been created, they need to be loaded into the memory. The syntax for running this job is: csh –f $aleph Proc/ load_z127_to_mem

### 2.6.2 Workflow

After bibliographic records have been loaded, run p_union_01 to create empty equivalency tables. Next, build indexes for the database (two processes are required - p_manage_01, p_manage_05), and after indexes have been built, run p_union_02 and p_union_04.

### 2.6.3 Differences between p_union_02 and p_union_04

P_union_04 runs all the time and maintains equivalencies for new and updated records. There are a number of important differences between p_union_02 and p_union_04. They are:

- P_union_02 rebuilds all equivalences for the range of records entered
- P_union_04 only rebuilds equivalency records that are flagged N
- P_union_02 builds equivalency records one by one and thus does not maintain symmetry between equivalency tables
- P_union_04 builds an equivalency record and updates all related records, thus maintaining symmetry between records
- P_union_02 can run with multiple processes
- P_union_04 has a single process
- P_union_02 only attempts to build equivalencies for records with fewer than 100 candidates. If a record has too many candidates, the Z120 is flagged N and p_union_04 attempts to build it
• P_union_04 attempts to build equivalencies for all records – even those with many candidates

Note that libraries should never run p_union_02 on part of a database specifically because it does not maintain symmetry. Only the p_union_04 process should be used to update a portion of the database.

2.6.4 Miscellaneous Tables
The variable in prof_library that controls the p_union_02 loop length is:

Illustration 22 – prof_library [data_root]

| 15.2 | setenv union_loop_length 5000 |
| 16   | setenv p_union_02_loop_length 5000 |

For more information on setting loop lengths see the document entitled, “How to Run Index Jobs”.

2.6.5 Triggers
Union Catalog libraries need to have a trigger for the Z127 table set up and active. The trigger ensures that the contents of the Z120 table and the Z127 table stay in synch. Note that the contents of the Z127 table are derived from the Z120 table. There are two processes that create a load to the Z127 table after a library has been loaded and the Z120 table built. The Oracle trigger continues to update existing Z127 records and add new Z127 records as required. Use the UTIL/A/17 menu to check the status of your Z127 trigger.

3 OPAC Functionality
ALEPH libraries that are defined as Union Catalog libraries can offer bases that are Union Catalog-enabled and bases that are not. Union Catalog-enabled bases are bases that use the record equivalency tables to de-duplicate result sets and merge records - non-Union Catalog enabled bases do not provide de-duplication or record merging. Union Catalogs wishing to offer individual campus views can set up logical bases comprised of records from a single campus. If the base is not Union Catalog-enabled, only that campus’s records will display. For more information on logical bases, see the How to Set Up Bases and Filters in ALEPH document.

3.1 Enabling Union Catalog Bases In Your OPAC
Logical bases are set in the tab_base.lng table, which is located in the alephe_tab directory. In libraries that are set up as Union Catalog or Union View libraries in tab100, bases that begin with U- are Union Catalog bases, and bases that do not begin with a U- are standard bases. Note that the only difference between Union Catalog bases and all other bases is that the result set in Union Catalog bases uses record equivalency tables. They are identical in all other ways. Thus, make sure to set all
columns in tab_base.lng and be sure to run the same processes as are run for any logical base (p_manage_32). See ALEPH documentation for more information on Logical bases.

### 3.2 Configuring Your OPAC Interface for Union Catalog

#### 3.2.1 Union Catalog Settings in www_server.conf

The following setting in www_server.conf applies to Union Catalog bases in versions 14.2 and 15.5. This setting was removed from version 15.2, 16.02 and all subsequent versions:

Illustration 23 - www_server.conf [alephe_root]

```plaintext
setenv www_sort_limit 800
```

The sort limit is also used as the de-duplication limit. Only that number of records will be de-duplicated. This means that if a retrieval set is larger than the limit, only part of the retrieval set will be de-duplicated. The de-duplicated records should be the first to be displayed.

In versions without this setting [version 15.3 and later], the system has no limit on the retrieval set size, but there is a system-wide limit of 1,000 records for display and sorting. In a Union Catalog, up to 1,000 de-duplicated and merged records will display. However, because the entire set is not de-duplicated, only the number needed to reach a display of 1,000 records, the set size does not accurately reflect the number of de-duplicated records.

#### 3.2.2 Merged Display

The system uses the configuration table `tab_merge_union` located in the `data_tab` directory of the bibliographic library to configure the merged display. This table controls record merging for display and allows you to set, on a tag-by-tag basis, which fields will be displayed from the preferred and non-preferred records.

Illustration 24 - tab_merge union [data_tab]

```plaintext
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|90 |1 |Y |###|
|90 |1 |N |SID##
|90 |1 |N |852##
|90 |1 |N |856##
|90 |2 |Y |5050#,u,*
|90 |2 |Y |852##
|90 |2 |Y |856##
```

**Key to the table:**

- Column 1 – Merge set – will always be 90 for a Union Catalog merge
Column 2 – Merging direction (1 refers to preferred record, 2 to each equivalent record)

Column 3 – Action (Y, N, C) Y – retain field, N – delete field, C – retain field if it is original

Column 4 – Field tag.

Column 4 (continuation after ,) – subfield and contents to match on. Example: 01 2 Y 590##,5,*abc* (meaning “if there is a subfield 5 which contains ‘abc’)

Note: The program expand_doc_merge_union must be called in tab_expand for each instance where a merged display is required.

For example, a Union Catalog site might want to have a merged display in the Web OPAC but not the Z39.50. In this case, the WEB-FULL section of tab_expand would call expand_doc_merge_union but the Z39_SERVER would not.

Illustration 25 – tab_expand [data_tab]
WEB-FULL expand_doc_merge_union

### 3.2.3 Displaying Individual Records

In addition to displaying merged records that contain fields from the preferred record and from all equivalent records, it is also possible to display individual, unmerged records from each contributing library. Display of the individual unmerged record is activated by the SID2 link and the display itself is configured in the DIRECT-DOC section of tab_expand.

Illustration 26 - edit_doc_999.eng [data_tab]

```
## SID2 D LSource Record (SID2) Y E
```

Illustration 27 – tab_expand [data_tab]

```
WEB-DIRECT expand_doc_direct
WEB-DIRECT expand_doc_union_holding DIRECT=Y,SORT=1,JUMP-TO=Y
```

The link is hard-coded to display the Library name from the z124_campus_name field. For more information about the Z124 table, see section 4 below. Selecting the hyperlinked library name results in the display of that library’s record and holdings while selecting the hyperlinked “global view” returns you to the merged display of all records and all holdings.
Illustration 28 – OPAC full view of a merged record

4 Embedded Holdings

In a Union Catalog, holdings statements are stored in 8XX fields embedded in the bibliographic record. This is different from a non-Union Catalog library where location information is stored in MARC21 holdings records that are stored in a separate but linked library. A bibliographic record can have multiple holdings fields embedded in a single bibliographic record and all of the fields will display. In addition to displaying the holdings location, the Union Catalog also displays electronic links and links to real-time circulation status.

Real time circulation status is obtained from ALEPH libraries using the Ex Libris X-server. The X-server needs to be licensed on both the Union Catalog server and on the local library’s server. X-server functionality is available from version 14.2 and later.

4.1 Location Table

Typically, the embedded 852 field contains a campus or institution code as well as a library code and call number. The mechanism for translating what we will refer to as location codes into location statements is as follows: a configuration table, tab_locations, stores a list of campus and library codes and their corresponding names. This table is loaded into Oracle as the Z124 table. The system matches the contents of the SID $b and the 852 $a and $b against the key of the Oracle table. If a match is found, the location names display instead of the codes - if no match is found, the location does not display in the OPAC. Note that more information about the SID field is available in section 5.2.3.

4.1.1 Tab_locations [data_tab]

Enter location information in tab_locations table in the data_tab directory of the Union Catalog library (xxx90).

The table has ten columns, the first, the location key, is a 24-character field that contains a location code plus a sublocation code and matches on the combined
contents of subfields a and b of embedded 852 fields. The remaining nine columns contain data useful for display and/or indexing. The remaining fields are:

- campus type
- campus region
- suppress display
- campus code
- campus base
- primary location
- full campus name
- call number prefix
- notes.

All of the data elements and their lengths are listed in the table below. Data elements marked (I) are expanded into the bibliographic record for indexing. Data elements marked (D) are included in the location display on the full record display in the Web OPAC, and data elements marked (R) are included in the results list in the Web OPAC.

<table>
<thead>
<tr>
<th>Length</th>
<th>Data Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>Location key</td>
</tr>
<tr>
<td>1</td>
<td>Type (I)</td>
</tr>
<tr>
<td>1</td>
<td>Region (I)</td>
</tr>
<tr>
<td>1</td>
<td>Suppress display (values are Y or N or blank – blank is No)</td>
</tr>
<tr>
<td>5</td>
<td>Campus code (I)(R)</td>
</tr>
<tr>
<td>15</td>
<td>Campus base (I)</td>
</tr>
<tr>
<td>9</td>
<td>Primary location (D)</td>
</tr>
<tr>
<td>50</td>
<td>Campus name (D)</td>
</tr>
<tr>
<td>15</td>
<td>Call number prefix (D)</td>
</tr>
<tr>
<td>70</td>
<td>Notes (D)</td>
</tr>
</tbody>
</table>

Once the tab_locations table has been populated, it needs to be loaded into Oracle. This is done by running the process p_load_Z124. This process can only be run from the command line.

### 4.1.2 Oracle Tables
The Z124 table stores data from the location table. The structure of the Z124 table is:

IIlustration 29 – Z124 table

<table>
<thead>
<tr>
<th>Field</th>
<th>Picture</th>
</tr>
</thead>
<tbody>
<tr>
<td>2124-REC-KEY</td>
<td>X(24). K</td>
</tr>
<tr>
<td>2124-TYPE</td>
<td>X(1).</td>
</tr>
<tr>
<td>2124-REGION</td>
<td>X(1).</td>
</tr>
<tr>
<td>2124-SUPPRESS</td>
<td>X(1).</td>
</tr>
<tr>
<td>2124-CAMPUS-CODE</td>
<td>X(5).</td>
</tr>
<tr>
<td>2124-AT</td>
<td>X(15).</td>
</tr>
<tr>
<td>2124-PRIMARY-LOCATION</td>
<td>X(9).</td>
</tr>
<tr>
<td>2124-CAMPUS-NAME</td>
<td>X(50).</td>
</tr>
<tr>
<td>2124-CALL-NUMBER</td>
<td>X(15).</td>
</tr>
<tr>
<td>2124-NOTES</td>
<td>X(70).</td>
</tr>
<tr>
<td>2124-SORT-KEY</td>
<td>X(50).</td>
</tr>
</tbody>
</table>
There is a single process, `p_load_z124`, which is used to load the Z124 table. It takes the `tab_locations` table and loads it into Oracle. Note that this must be done each time `tab_locations` is changed. The table can be located in `data_tab`, `data_files` or `data_scratch`. The second parameter of the process is table name. If the table is in `data_files`, run it with name only. If the table is located in `data_scratch` or `data_tab`, the process must include a directory relative to `data_files` as shown below:

Table in `data_files`:  
csh -f $aleph_proc/p_load_z124 usm90.z124,n,n,n,n

Table in `data_tab`:  
csh -f $aleph_proc/p_load_z124 usm90../tab/tab_locations.z124,n,n,n,n

Table in `data_scratch`:  
csh -f $aleph_proc/p_load_z124 usm90../scratch/tab_locations.z124,n,n,n,n

### 4.1.3 Indexing and Displaying Locations

Four of the ten location fields can be expanded into bibliographic records either for indexing or for display. They are: type, region, campus code, and campus base. These fields are particularly useful for creating logical bases. A single expand program, `expand_doc_cdl_location`, expands the index-able fields into the bibliographic record as four separate fields, CMP1-4. Mapping is as follows:

<table>
<thead>
<tr>
<th>Type</th>
<th>CMP1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td>CMP2</td>
</tr>
<tr>
<td>Code</td>
<td>CMP3</td>
</tr>
<tr>
<td>Campus Base (Z124_AT)</td>
<td>CMP4</td>
</tr>
</tbody>
</table>

To create keyword indexes on locations, add `expand_doc_cdl_location` to the WORD section of `tab_expand` and the following lines to `tab11_word`.

Illustration 30 - `tab_expand` [data_tab]

```plaintext
WORD expand doc cdl location
```

Illustration 31 - `tab11_word` [data_tab]

<table>
<thead>
<tr>
<th>CMP1#</th>
<th>03 B WRD WC1 WID</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMP2#</td>
<td>03 B WRD WC2 WID</td>
</tr>
<tr>
<td>CMP3#</td>
<td>03 B WRD WC3 WID</td>
</tr>
<tr>
<td>CMP4#</td>
<td>03 B WRD WC4 WID</td>
</tr>
</tbody>
</table>

### 4.2 Displaying Embedded Holdings

Display of embedded holdings in the Web OPAC is controlled by one of two programs: `expand_doc_union_holding` and `expand_doc_union_holding_cdl`. The two programs are very similar, however, `expand_doc_union_holding_cdl` has some institution-specific behavior programmed in.

#### 4.2.1 expand_doc_union_holding

This program is called in `tab_expand` and controls the holdings section of the Union Catalog full record display. The program generates a virtual field called HOL from
data in the Bibliographic record, in the Oracle location table (Z124) and in database configuration tables. An HOL field is created for each 852 or 856 field in a bibliographic record. The HOL field can then be configured to display in the Web OPAC as well as in the terminal mode client.

The program works as follows:

1. The program expand_doc_merge_union merges equivalent records, taking holdings fields (SID, 852 and 856) from each equivalent record
2. Each 852 $$a + $$b in a bibliographic record is matched against the Oracle location table (Z124), an HOL field is created for each field that has a matching location tables. Note that if no match is found, no HOL line is generated.
3. Fields from Oracle location table are concatenated with 852 field from the bibliographic record as follows:
   a. HOL $$a => Source tag – tag in the bibliographic record that data is taken from
   b. HOL $$b => Location, campus name - taken from z124_campus_name
   c. HOL $$c => Sub-location, library name – taken from Z124-primary_location
   d. HOL $$d => Call number - taken from the following places in this order
      i. 852 $$k
      ii. z124_call_number
      iii. 852 $$h $$i $$j
   e. HOL $$e => Availability – This section is further controlled by the parameter “JUMP-TO” as explained below. If the source tag is 852 then the availability is circulation status. There is more information about configuring this section below. If the source tag is 856 then the availability is online and 856 $$u is used to generate a link to.
   f. HOL $$f => Note – taken from the following list of subfields in the order listed. 852 $$g $$p $$3* $$m Z124_notes 852 $$z $$D**.
   g. HOL $$g => Campus code – taken from z124_campus_code
4. Three parameters control behavior of the HOL display. The three parameters are
   a. DIRECT – controls if the program is called when a single un-merged record is displayed. Parameters are Y and N.
   b. SORT – controls the sort of the HOL fields. One of two sort routines can be used
      i. 1 – sort using the location table sort key (z124_sort_key). The z124_sort key is filled in when the contents of tab_locations are loaded into Oracle and each line is assigned a running number based on position, the first line in the table is 1, the second 2 etc.
      ii. 2 – sort alphabetically by Campus Name
   c. JUMP-TO – controls creation of a link from the Holdings display for circulation status and for electronic locations. If Jump-to is set to Y then a link to circulation status is created for HOL fields created from 852 fields, and a link to electronic locations is created for HOL fields created from 856 fields.
852 $3 – subfield 3 of the 852 field can be used to store textual holdings statements for serials. Subfield 3 can repeat.

852 $D – subfield D of the 852 field can be used to store receipt status.

4.2.2 expand_doc_union_holdings_cdl

The expand_doc_union_holdings_cdl program functions very much like the expand_doc_union_holdings program with a few minor differences. The differences are hard-coded and therefore not configurable. The differences are as follows:

- There is special treatment for holdings locations that have a campus code that contains the RLF. These locations remote storage facilities and electronic links do not display from records that have these codes and no others.

- Suppressed locations – locations that are flagged for suppression but have an electronic link, display with a link to the electronic location.

4.3 Circulation Status

The Union Catalog supports the display of up-to-date circulation status information by querying the local catalog and displaying the results in the Union Catalog. The initial query is initiated by selecting the hypertext link marked “availability” from the Full display. The link displays from the full display of the merged record and the full display of the unmerged record. The link is activated by setting the JUMP-TO variable in tab_exp to Y. The link to will only work if the records in the Union Catalog point to specific records in the local catalog and if the local catalog is available. The pointer from the Union Catalog to the local catalog is the SID field. More information about the SID field is available in section 5.2.3. If the library is not available, location information from the embedded holdings will display with a message saying that the library is not available.

The link to circulation status can be configured to use the z39.50 server or the Ex Libris X-server.

4.3.1 Using z39.50

More information about the Z39.50 setup can be obtained directly from Ex Libris. Bear in mind that in order for the link to circulation status to work via Z39.50, both the Union Catalog and each of the local catalogs need to have a Z39.50 server and gate set up.

4.3.2 Using the X-server

Union Catalogs that target ALEPH libraries can use the Ex Libris X-server instead of Z39.50 to obtain circulation status information from the local libraries. In order to use the X-server to obtain circulation status information, it must be licensed and set up on both the Union Catalog and on the local library server.
4.3.3 Union Catalog Setup

The method of obtaining circulation status is set in the alephe_gate directory. There, a configuration table should be created for each campus that needs to be accessed. The convention for naming the configuration table is \(<\text{local library code}>.\text{conf}\) where local library code is the code that appears in the 852 field subfield a. An example of a configuration table name is bin.conf

Variables that need to be set in this table are:
1. Base – name of the base that is being targeted
2. Host name – IP of the server that is being targeted
3. Circulation program – name of the program used to display circulation status.
   There is currently only one program – \(\text{www_f_union_circ_www_x}\)
4. Access method – method of accessing the targeted server and base, for X-server should be set to X-SERVER.

Illustration 32 – usm.conf [alephe_gate]

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z58-BASE-REMOTE</td>
<td>USM01</td>
</tr>
<tr>
<td>Z58-HOST-NAME</td>
<td>10.1.235.11:8991</td>
</tr>
<tr>
<td>Z58-CIRC-PROGRAM</td>
<td>(\text{www_f_union_circ_www_x})</td>
</tr>
<tr>
<td>Z58-ACCESS-METHOD</td>
<td>X-SERVER</td>
</tr>
</tbody>
</table>

4.3.4 Local Library Setup

The following tables need to be configured on the server of the library being targeted in order for the X-server to work: www_x_func, user_function.eng, license.www_x, tab_z30_sort, www_server.conf

1. www_f_func [alephe_tab] needs to have a line added for the function CIRC-STATUS. The line appears as follows:

   `########## CIRC-STATUS             \(\text{www_x_circ_status}\)`

2. user_function.eng [alephe_tab] needs to have a line added for Circulation status. The line appears as follows:

   `WWW-X           L X-SERVER Interface          CIRC-STATUS      L Circ status`

3. License.www_x [alephe_tab] needs to have the X-server enabled for circulation status. The line appears as follows:

   `CIRC-STATUS              Y`

4. tab_z30_sort [data_tab in the bibliographic library] A sort routine needs to be defined for sorting items retrieved by the X-server. The line appears as follows:

   `WWW-X                      A 01 A 05`
5. `www_server.conf` [alephe_root] - an environmental variable needs to be added for limiting the number of items that can be retrieved by the X-server. The line appears as follows:

```
www_no_items_display     1000
```

6. In versions 16.02 and later, a staff user needs to be set up for the X-server. The user needs to be set up in the local library. The username and password for this user must be WWW-X-CIRC/WWW-X-CIRC and it must be assigned permission for the X-server interface and for displaying circulation status. Earlier versions do not check for user name and password.

5 Loading records into a Union Catalog

Union Catalogs typically receive regular batches of records from each of their member libraries, these records need to be loaded into the Union Catalog. The process of loading the records into the Union Catalog has two steps. The first step involves data conversion, enrichment and validation. The second step involves loading and indexing the records into the Union Catalog. While the second step, the process of loading records, is identical for all libraries, the first step, where records are converted from the local format into the Union Catalog format and character set, might be slightly different for each library.

Records are placed in the incoming directory in MARC 21 format and converted from MARC 21 format into ALEPH sequential. At the same time, they are converted into UTF8. Once in ALEPH sequential format, record conversion programs can be used to convert or enrich the data, finally, records are validated and either moved to the converted directory for loading or rejected.

5.1 Setup and Workflow:

5.1.1 Directory Structure:

ALEPH Union Catalogs [XXX90] contain a directory called load and this directory contains a subdirectory for each input stream. The data_load directory is configured in prof_library. An input stream is any contributing library or organization that had distinct conversion needs. Each input stream’s directory has the following five subdirectories: incoming, converted, rejected, loaded, and log. Note that an input steam name must be one to five characters.

Illustration 32 – directory structure for record loading

```
/load
 /abcd /efgh
 /incoming /converted /rejected /loaded /log /incoming /converted /rejected /loaded /log
```
5.1.2 Configuration Tables:
The ALEPH configuration tables that control record conversion and validation are all located in the data_tab directory of the Union Catalog. They are: tab_fix, check_doc, check_doc.eng and check_doc_mandatory. Each input stream can have a corresponding section in any one or all of these tables. The section name and the directory name are the same.

5.2 Record Conversion and Pre-processing

5.2.1 Workflow
Files of incoming records in MARC 21 format are placed in the incoming directory of the appropriate input stream. The p_union_03 process picks the records up from there, converts them into ALEPH Sequential format, validates them, and moves them to one of two directories: converted or rejected. The log file of the p_union_03 process is written to the log directory. Note that in addition to converting records from MARC format to ALEPH Sequential format, the p_union_03 process can also convert the character set of the records from MARC 8 to UTF8. The third parameter of the p_union_03 process controls character conversion. If the records need to be converted to UTF8, set the third parameter to Y. Otherwise, run it with N or blank.

5.2.2 Record Conversion:
There are three stages in record conversion. They are:

- Conversion from the incoming format into ALEPH sequential
- Field-by-field data conversion or enrichment
- Record validation.

During the record conversion stage a file of MARC records is converted into ALEPH sequential (p_file_01 and p_file_02). There are no configuration tables consulted at this stage. Once the records are in ALEPH sequential format, the records are “fixed” – this process is regulated by tab_fix, an ALEPH configuration table located in the data_tab directory of the Union Catalog. Tab_fix should include a section for each input stream and can be used to call “fix” programs that are specific to the input stream as well as generic ALEPH “fix” programs. Libraries can write their own fix programs or they can use any of the pre-existing ALEPH fix programs. In addition, they can create scripts for fixing records using the ALEPH generic fix mechanism. Finally, libraries can write their own fix programs and call them in tab_fix. A list of generic fix_doc programs can be found in the ALEPH User Guide (a.k.a. Web Guide), Cataloging System Librarian chapter. Note that fix_doc programs must be entered in the table in the order in which they are to be performed.

Illustration 33 - tab_fix [data_tab]

<table>
<thead>
<tr>
<th>UCLA</th>
<th>fix_doc_ucla</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCLA</td>
<td>fix_doc_sid_cdl</td>
</tr>
</tbody>
</table>
Key to the table:
    Column 1 – Routine name/input stream
    Column 2 – Program name
    Column 3 – Program arguments

5.2.2 Mandatory Fields
In order for records to be considered “valid” for the Union Catalog, they must have a
005 field and an SID field. The 005 field is the MARC 21 field for storing latest
transaction date, the programs that load records into the Union Catalog consults the
005 field to confirm that the record being loaded is more recent than the version of
that record already in the database. The SID field is an ALEPH proprietary field. It
serves as a pointer from the record in the Union Catalog to the same record in a local
catalog. In addition, the SID field is used to uniquely identify records in the Union
Catalog for the purpose of updating them. The structure of the SID field is $$b Local
library code $$c system number in local library. Here is an example of an SID field:
SID $$bUCSB$$c000034928.

If records do not already have these fields, tab_fix should be used to assign them.

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

Illustration 34 - check_doc [data_tab]

<table>
<thead>
<tr>
<th></th>
<th>check_doc_doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCLA</td>
<td>check_doc_doc</td>
</tr>
<tr>
<td>UCLA</td>
<td>check_doc_ucla</td>
</tr>
</tbody>
</table>

Key to the table:
    Column 1 – Routine name/input stream
    Column 2 – table name/program name

Illustration 35 - check_doc_doc section 1 [data_tab]

| OC XX 5002 01 01 245## |
| OC XX 5008 01 99 852## |
5.2.4 Batch Processes

The batch process that converts and validates records is called p_union_03. The process can be run with the following parameters: Active library, input stream,
convert to UTF8. If an input stream is specified then the process will only check the incoming directory for that stream, if no input stream is specified, the process checks all incoming directories. Note that the list of allowed input streams will vary from installation to installation.

5.3 Record Loading

Once records have been converted, enriched and validated, they are ready to be loaded into the Union Catalog. Valid records are picked up from the subdirectory converted and loaded into the Union Catalog. The process that loads records does the following:

Determines if the record is new or if it already exists in the database by searching the database for the SID of the incoming record.

1. If the record exists
   a. The dates of the two records are compared and the most recent record is retained. The date is taken from the 005 field.
   b. If the most recent version of a record is a deleted record, the record in the Union Catalog is deleted and all access points removed. Delete status is set in LDR position 6.
2. If the record is new, it is loaded into the database as a new record and indexed

Note that this entire process is hard-coded. Libraries must have an SID field in their incoming and database records and an SID index set up as a direct index in tab11_ind. The index should be called SID.

5.4 Batch Processes

The batch processes that perform the record loading and indexing are p_union_13 and p_union_14.

P_union_13 takes over from p_union_03 and loads records from the directory converted, records are placed in /converted by p_union_03. After loading the records, the input file is moved to the directory loaded and a suffix loadout is added to the file name. A log file is created in the log directory. It logs the number of records that were added as New, the number of records that replaced existing records in the database and the number of records that were rejected. Records are rejected if their date is earlier than the date of the same record in the database.

The p_union_13 has three parameters. They are: library name, input stream, and load equal. If no input stream is specified, the process will check all input streams. If load equal is set to Yes, incoming records that have a date equal to the date in the database record will be loaded. If it is set to No, incoming records with a date equal to the database record date will be rejected.

In addition to loading incoming records, p_union_13 partially indexes them and updates the record equivalency tables (Z120). The update to the equivalency tables works as follows:
• For new records, a new table is created, the table is flagged Z

• For updated records, the Z120 is emptied out so that the record itself is no longer equivalent to any other records in the database. In addition, the record is taken out of any record equivalencies that it occurs in, either as an equivalent or as a preferred record. Finally, the record’s record equivalency table is flagged Z.

The p_union_13 process starts two related processes, p_union_14 and p_union_04. Note that p_union_13 and p_union_14 can run simultaneously. P_union_04 cannot run with p_union_13 and will only start once p_union_13 has completed.

P_union_14 has only one parameter, library name. It retrieves all records that have a record equivalency flagged Z and builds keyword indexes for those records. After building or rebuilding keywords, p_union_14 updates the record equivalency from Z to N.

P_union_04 also has one parameter, library name. It retrieves all records that have a record equivalency flagged N and builds the equivalency table. Note that equivalencies are symmetric. Accordingly, when a record is added to the new record’s equivalency table, the new record is in turn added to its equivalency table.

5.4.1 Other Processes
In addition to the processes listed above, Union Catalog libraries also have to run p_manage_17 to sort long headings and util_e_08 to build links between the bibliographic library and the authority library. Union Catalog libraries do not have to run util_e_01.