Voyager®
Primo® Integration User's Guide

August 2013

(Voyager Version 8.2.2)

CONFIDENTIAL INFORMATION

DISCLAIMER

TRADEMARKS

Document released: August 2013

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

Contents

Contents

Contents

Contents

Contents

Contents

About This Document

Purpose

The purpose of the *Voyager® Primo® Integration User's Guide* is to provide you with the Voyager-related information that you need to do the following:

- Set up/modify configuration files
- Run batch files
- Set up the system for processing user requests

Intended Audience

This document is written for a technical audience responsible for the following:

- Configuration file setup
- Initial upload of Voyager data into Primo
- Writing scripts and scheduling batch jobs for running updates
- User authentication setup
- Review/analysis of message logs as needed

Reason for Reissue

This guide incorporates and is being reissued for the following reason:

- Added the page.myAccount.chargedItem.dueDate.indef= Parameter section on page 5-11

Document Summary

Chapter 1	"Getting Started" This chapter describes the prerequisites for getting started with the Voyager Primo integration setup.
Chapter 2	"Data Extraction" This chapter provides information needed to process the initial data extraction from Voyager and ongoing updates.
Chapter 3	"Course Reserves" This chapter provides information regarding the extraction of course reserves data from the Voyager database.
Chapter 4	"Real-Time Availability" This chapter provides information about real time availability processing with Voyager Primo integration.
Chapter 5	"Patron Services - OPAC via Primo" This chapter provides information about patron services such as user authentication and processing user requests with OPAC via Primo.
Chapter 6	"Patron Services - OPAC via Link" This chapter provides information about patron services such as user authentication and processing user requests with OPAC via Link.
Chapter 7	"Apache mod_deflate Module" This chapter provides the instructions for enabling the Apache `mod_deflate` module in the Voyager environment.
Appendix A	"Verify Voyager Processes/Services" This appendix provides the information and steps to verify successfully running Voyager processes/services.
Index	The Index is an alphabetical, detailed cross-reference of topics.

Conventions Used in This Document

The following conventions are used throughout this document:

- Names of commands, variables, stanzas, files, and paths (such as `/dev/tmp`), as well as selectors and typed user input, are displayed in `constant width` type.
- Commands or other keyboard input that must be typed exactly as presented are displayed in **`constant width bold`** type.
- Commands or other keyboard input that must be supplied by the user are displayed in ***`constant width bold italic`*** type.
- System-generated responses such as error messages are displayed in `constant width` type.
- Variable *portions* of system-generated responses are displayed in *`constant width italic`* type.
- Keyboard commands (such as **Ctrl** and **Enter**) are displayed in **bold**.
- Required keyboard input such as “Enter **`vi`**” is displayed in **`constant width bold`** type.
- Place holders for variable portions of user-defined input such as **`ls -l`** ***`filename`*** are displayed in ***`italicized constant width bold`*** type.
- The names of menus or status display pages and required selections from menus or status display pages such as “From the **Applications** drop-down menu, select **System-wide**,” are displayed in **bold** type.
- Object names on a window’s interface, such as the **Description** field, the **OK** button, and the **Metadata** tab, are displayed in **bold** type.
- The titles of documents such as *Acquisitions User’s Guide* are displayed in *italic* type.
- Caution, and important notices are displayed with a distinctive label such as the following:

NOTE:
Extra information pertinent to the topic.

IMPORTANT:
Information you should consider before making a decision or configuration.

CAUTION:
Information you must consider before making a decision, due to potential loss of data or system malfunction involved.

TIP:
Helpful hints you might want to consider before making a decision.

RECOMMENDED:
Preferred course of action.

OPTIONAL:
Indicates course of action which is not required, but may be taken to suit your library's preferences or requirements.

Document Reproduction/Photocopying

Photocopying the documentation is allowed under your contract with Ex Libris (USA) Inc. It is stated below:

> All documentation is subject to U.S. copyright protection. CUSTOMER may copy the printed documentation only in reasonable quantities to aid the employees in their use of the SOFTWARE. Limited portions of documentation, relating only to the public access catalog, may be copied for use in patron instruction.

Comment on This Document

To provide feedback regarding this document, use the Ex Libris eService or send your comments in an e-mail message to docmanager@exlibrisgroup.com.

Getting Started

1

Getting Started

1

Introduction

The *Voyager® Primo® Integration User's Guide* is designed to provide you with the information that you need to successfully utilize Voyager data in a Primo user environment.

The following components are a part of the Voyager Primo integration for all Primo configurations:

- Configuration files for data extraction
- Batch files to run jobs
- Real-time availability

The following components are available using either OPAC via Link or OPAC via Primo:

- User authentication with Patron Directory Services (PDS)
- User account information
- Item display and request processing

NOTE:
For Voyager versions 6.5.4 through 7.1, OPAC via Link is the only implementation available of Voyager with Primo. Starting with Voyager 7.2, OPAC via Primo is available and the OPAC via Link implementation is available as a legacy offering.

Purpose of this Chapter

The purpose of this chapter is to identify the skills and preparations necessary for setting up your environment for Voyager Primo integration.

Prerequisite Skills and Knowledge

To use this document effectively and set up your systems for Voyager Primo integration, you need to have a general understanding of the following.

- Microsoft® Windows-based applications
- XML and XML-format data as well as editors and related software
- UNIX® operating system commands and file system
- At least one UNIX-based text editor such as `ed` or `vi`
- Local procedures

This user's guide assumes you have a working knowledge of Primo setup and access to Primo documentation.

Before You Begin

Before you can begin your Voyager Primo integration, you need to have the following software installed:

- Voyager (the base product)
- Primo

Checklist

In preparation for setting up Voyager Primo integration, gather the following information:

- Determine if you are going to use OPAC via Primo or OPAC via link (consult all available Primo documentation for help with this decision)
- Identify the elements needed to customize configuration files
- Identify the schedule/times for processing updates
- Identify excluded happening locations
- Identify excluded operator IDs

- Identify institution ID
- Identify statuses to be made available to Primo

Port Settings

Voyager port settings are defined one time at installation.

NOTE:
The Real Time Availability (RTA) port is different from the OPAC port. The RTA port is also defined one time at installation.

Refer to the *Primo Interoperability Guide* for additional information regarding port settings.

Data Extraction

2

Data Extraction

2

Introduction

Primo discovery takes advantage of the rich data stored in the Voyager database. To do this, bibliographic data is extracted from the Voyager database for an initial load and then, subsequently, kept current via regularly scheduled updates with Primo. This chapter discusses the data extraction process.

Purpose of this Chapter

The purpose of this chapter is to describe the components needed for extracting Voyager data for Primo to harvest.

Data Extraction Process Overview

The Voyager Primo integration data extraction tools are used for the following activities:

- Initial load of Voyager data for a new installation of Primo
- Ongoing updates to the Primo database for an existing installation of Primo harvesting Voyager data
- Other special purposes

Initial Primo Data Load

For the initial Primo database load, you need to do the following:

- Tailor the configuration file to your environment/needs.

  Use the `PrimoExp-Publishing.ini` as a starting point. See Publishing Configuration File on page 2-14

- Run the `Pprimoexp` batch job.

  Using the parameters set in the configuration file, this step identifies the Voyager records to be uploaded to the Primo database that are bundled into one or more `tar.gz` files.

  IMPORTANT:
  Do not include more than 10,000 records in a single tar file. There is a 10MB maximum, file-size limit for Primo throughput that is estimated to be 10,000 MARC records.

- Permit Primo to securely access the directory in which the generated files from Voyager are placed. This is the output location for the `tar.gz` files.

NOTE:
A number of options exist for moving the data to the Primo server for the initial load such as using FTP, a shared NFS mount, manually copying the data, or utilizing a cron/script to copy the data. For additional information, see your implementation support contact to discuss the alternatives for Primo to access Voyager data.

Initial Load Generation - Excluded Records

When identifying bibliographic records for the initial data load, the following criteria is used to identify records to be excluded:

- If a bibliographic record is suppressed from the OPAC, a MARCXML record is not generated.
- If a bibliographic record was most recently modified at one of the excluded happening locations as identified in the configuration file, a MARCXML record is not generated.
- If a bibliographic record was most recently modified by one of the excluded operators as identified in the configuration file, a MARCXML record is not generated.

See Stanza Definitions for Voyager Primo Integration Configuration Files on page 2-29 for more information about customizing the configuration file.

Ongoing Updates to Primo Database

You need to evaluate how frequently to process updates to the Primo database. Consider the following:

- Level of activity within your Voyager database
- Frequency of bibliographic record changes
- Frequency of item status changes
- Time of day when system usage is greatest
- Time of day when system usage is the least

Pipes

Primo uses separate pipes for harvesting Voyager data.

One pipe is used to harvest Voyager data with bibliographic record changes. This is the publishing pipe.

The other pipe is used to harvest Voyager data with holdings availability changes. This is the availability pipe.

Given this approach for harvesting data, separate and customized versions of configuration files are provided to you for publishing and availability purposes as follows:

- `PrimoExp-Publishing.ini`

  See Publishing Configuration File on page 2-14 for more information.

- `PrimoExp-Availability.ini`

  See Availability Configuration File on page 2-6 for more information.

These are located in `/m1/voyager/yyydb/ini` where `yyydb` is the database name.

These configuration files are the same except for the parameter setting differences required for each purpose. See Stanza Definitions for Voyager Primo Integration Configuration Files on page 2-29 for more information about the parameters that affect which records are generated for harvesting.

NOTE:
The type of content in these generated records is the same. However, different records get generated for Primo harvesting as identified by the triggers (parameters) specified in the configuration files.

Corresponding copies of `Pprimoexp` batch files for publishing and availability purposes can be created in order to automate the harvesting process using a cron. See Pprimoexp Batch Job on page 2-46 for more information about `Pprimoexp` batch processing.

Having separate functional configuration files, batch files, and directories for generated files enables your flexibility in scheduling production processing of data changes in the Voyager Primo integrated environment. This allows you to schedule Primo harvesting of bibliographic record changes from Voyager at an appropriate time and separate from scheduling harvesting of Voyager data with holdings availability changes.

Special Purpose/Selective Database Extractions

Aside from the initial Primo data load and ongoing updates, you may want to set up the Voyager Primo integration tools for other special purposes that meet your institution's specific requirements.

For example, you may want to set up the Voyager Primo integration tools to create a testing subset before doing a complete initial load of the database. This can be done using parameters like `BibRangeBegin` and `BibRangeEnd` or `BibsFromFile`.

The `PrimoExp-Selective.ini` configuration file is provided as one method for creating a testing subset. It is located in `/m1/voyager/yyydb/ini` where `yyydb` is the database name.

For Selective record processing and storing `tar.gz` files, the following directory is provided where `yyydb` is the database name:

`/m1/voyager/yyydb/primo/Selective/bundled`.

Insure that Primo has access to this directory.

Primo Transaction / Record Normalization

Primo originates an FTP transaction to Voyager to harvest the records selected by the `Pprimoexp` processing for inclusion in the Primo database. Subsequently within Primo, these records are normalized into the Primo Normalized XML (PNX) format.

NOTE:
A number of options exist for moving the data to the Primo server such as using a shared NFS mount, manually copying the data, or utilizing a cron/script to copy

the data. See your implementation support contact to discuss alternatives for harvesting data.

Configuration Files

Primo uses data extracted from the Voyager database. Configuration files are used to define what is extracted. The following customized configuration files are provided with your installation:

- `PrimoExp-Availability.ini`

  See Availability Configuration File on page 2-6.

- `PrimoExp-Publishing.ini`

  See Publishing Configuration File on page 2-14.

- `PrimoExp-Selective.ini`

  See Selective Configuration File on page 2-22.

They are located in `/m1/voyager/yyydb/ini` where `yyydb` is the database name.

These configuration files are provided as examples for publishing, availability, and selective requirements with parameters set to what many customers use. However, copies of the configuration files may be made and tailored to meet your unique needs.

CAUTION:
Do not change Availability, Publishing, or Selective in the file naming convention when making copies. You may, however, modify the file name as in PrimoExp-Publishing.ini.sample_extract, for example.

The configuration files define the following elements:

- Item statuses to check for initial load and ongoing updates to the Primo database
- Happening locations for exclusion

  Records associated with these happening locations are excluded from the initial load and/or updates for Primo harvesting.

- Records to be excluded from the initial load and/or ongoing updates determined by operator ID that processed the record

  See [Excluded Operator Ids] on page 2-31.

- Institution ID to be included in the availability tag

- Location definitions for consistency within the Primo database

  See [Location to Primo] on page 2-36.

- Tags to identify unique field usage within your institution's database specific to the database extract

  See [Enrichment Tags] on page 2-36.

- Extract processing considerations such as date/time, record ID(s), filters, and so forth

  See [PrimoExp] on page 2-38.

- Additional processing considerations for uploading the Voyager records identified by the configuration file processing

  See [XMLWriterProtocol] on page 2-45.

See Stanza Definitions for Voyager Primo Integration Configuration Files on page 2-29 for a complete description of configuration file options.

Availability Configuration File

The following is an example of the availability configuration file:

```
# Configuration for Voyager Export for Primo -- Availability

# Voyager status maps to (A)vailable or (U)navailable for Primo
[Item Statuses]
Not Charged=A
Charged=U
Renewed=U
Overdue=U
Recall Request=U
Hold Request=U
On Hold=U
In Transit=U
In Transit Discharged=U
In Transit On Hold=U
Discharged=A
Missing=U
Lost--Library Applied=U
```

Figure 2-1. PrimoExp-Availability.ini Example

```
Lost--System Applied=U
Claims Returned=U
Damaged=U
Withdrawn=U
At Bindery=U
Cataloging Review=A
Circulation Review=A
Scheduled=U
In Process=U
Call Slip Request=U
Short Loan Request=U
Remote Storage Request=U

[Excluded Happening Locations]
# Items whose most recent activity took place in one of these locations will not be
          exported.
#ACQ
#ACQ TEMP
#illcirc

[Excluded Operator Ids]
# Items whose most recent activity was recorded by one of these operators will not be
          exported.

[LocationCodesForBibInclusion]
# If any location codes are listed here, only bib records that are linked to
# holdings at the specified locations will be included in the extract.

[LocationCodesForAVA]
# If any location codes are listed here, availability statements will only be generated
          for items
# and e-items that are linked to these locations. (Items via perm or temp location, e-
          items by
# holdings location.) Also, Course Reserve information will only be generated if the
          course reserve
# list location is among these locations.

[Institution Id]
# This value will be included in the Availability tag
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
# as subfield a
Id=qa810db

[Location to Primo]
# This section maps Voyager locations to Primo locations.

[Enrichment Tags]
# Tags added to marc record with additional data
#
# WARNING: You must change the orderInfoHolTag=960 in vxws.properties if you use the 960
           tag in here.
#
# AVA tag contains the availability information
# AUT tag contains name cross-reference information
# SUB tag contains subject cross-reference information
# CR tag contains course reserve information
# DPS tag contains Digital Preservation System (Product Rosetta) information
# EITM tag contains E-Item availability data (New for 8.0.0)
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
#Inserted by voy821 patch
# InsertPrimoBrowseTags, if set to Y, will include see references for
# subjects and authors that will allow Primo to construct browse searches
# from them. When InsertPrimoBrowseTags is Y, then the InsertAUTTag and
# InsertSUBTag will not be created since Primo can enrich the keyword search
# from the included Browse tags.
InsertPrimoBrowseTags=N

#======================================================================
# Preservation related object link definition
# This includes the label for the URL link, the URL link, and parameter name.
# The only supported dynamic parameter at this time is id (i.e. bib id).
# The parameter is passed to the third-party application that processes and retrieves
# intellectual entities from the Preservation repository.
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
#========================================================================
[DPS]
DPSLinkLabel=Rosetta Link:
DPSLinkAddress=http://somecomponent.com/content-aggregator/getIEs?system=ilsdb&
DPSLinkParm=id=



[PrimoExp]
# note that stanza.name can be used with -- on the command line
# argv(case insensitive) override
# ini(case insensitive) override
# env (all uppercase)
#
# these parameters provide defaults for this run of the job
# each of them can be overridden using a command-line parameter
#
# BibRangeBegin and BibRangeEnd are the beginning and ending bibliographic ID numbers to
          export.
# -B overrides this parameter on the command line
# BibRangeBegin
#BibRangeBegin=1
# -E overrides this parameter on the command line
#BibRangeEnd=200
#
# BibsFromFile reads the file at the supplied path for a list of bibliographic record ids
          to export.
# The file should contain one bibliographic record ID per line.
# -F overrides this parameter on the command line
# BibsFromFile=/m1/voyager/2008.1.0/qa810db/local/PrimoBibs.txt
#
# ChangedSince exports records changed since the supplied date.
# ChangedSince may be either
#     a timestamp in YYYYMMDD.HHMMSS format
#    or a path to a file containing a timestamp in YYYYMMDD.HHMMSS format on a single line
# If a path to a file is supplied, the current run time will be saved in the file when
# the job finishes.
# -C overrides this parameter on the command line
# Example:
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
# ChangedSince=YYYYMMDD.HHMMSS
#  or
# ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Availability/ChangedSince.txt
ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Availability/ChangedSince.txt
#
#
# HeadingChanges determines whether to export bibliographic records
# whose cross-references have changed
# -H Y|N overrides this parameter on the command line
HeadingChanges=N
#
# AvailFilter detemines whether to export records whose availability
# fields have not changed since the last time the job ran.
# If AvailFilter is Y, only records whose availability has changed are exported;
# if AvailFilter is N, all changed records are exported.
# -A Y|N overrides this parameter on the command line
AvailFilter=Y
#
# for filenames
# note that $str$ will be passed to date format so that you can
# specify date in the these files
#
# LogFile is the location of the log file.
# Note that $str$ will be passed to Java's Date.Format()
# function to control the datestamp embedded in the file name.
# -L overrides this parameter on the command line
LogFile=primo.export.$yyMMdd$.log
LogFileDir=/m1/voyager/2008.1.0/qa810db/primo/Availability/logs
#
#
# -v
# the logging level
LogLevel=10
# LogToStdOut determines whether or not to write the log output to standard output,
# as well as the log file.
LogToStdOut=N
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
# Database and the other Database parameters determine which database to connect to.
# Ordinarily these are set by the wrapper script based on the environment.
# -d
# DataBase=
#
# DatabaseHost
DatabaseHost=10.100.2.112
#
# DatabasePort
# -e
# UserPass
# The credentials needed to connect to the database.
# Optional parameter, override on command line with -u
# STRONG RECOMMENDATION: do not set this value here, use
# the command line override instead.
# Special note: Oracle TNS Alias (@DB) is _not_ used here
#UserPass=qa810db/qwrite0
#
Database=VGER
Protocol=com.endinfosys.voyager.extract.OAIPMHXmlWriterProtocol

# must do items, then mfhds, then bibs
Task=com.endinfosys.voyager.extract.ExtractCourseReservesPrimo
Task=com.endinfosys.voyager.extract.ExtractItemsPrimo
Task=com.endinfosys.voyager.extract.ExtractMFHDsPrimo
Task=com.endinfosys.voyager.extract.ExtractBibsPrimo
DoItems=Y
DoMfhds=Y

# The Del* Dir and File are used to determine when records are deleted.
# These are standard Voyager locations.
DelBibsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelBibFile=deleted.bib.marc
DelMFHDsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelMFHDsFile=deleted.mfhd.marc
#Import delete file has the form delete.imp.YYYYMMDD.HHMM
DelMFHDsImpFileBase=delete.imp
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
DelItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelItemsFile=delete.item
# NEW FOR 8.0.0
DelEItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelEItemsFile=deleted.e-items

# Set to Y to include course reserve data.
CourseReserves=N

# NEW FOR 8.1.0
# Set to Y to export suppressed bibs that are linked to reserve list items.
#  This flag will have no effect on suppressed bibs that are not linked to
#  reserve list items, and will be ignored if the CourseReserves flag is
#  set to N.
IncludeReserveListItemsSuppressedBibs=N

# NEW FOR 8.0.0
# Set to Y to include E-Item data.
#  If this flag is set to Y and the CourseReserves flag is set to N,
#  then this flag overrides the value of the CourseReserves flag,
#  as E-Items are linked to course reserves.
ExportEItemsWithCourseReserves=N

# NEW FOR 8.0.0
# Set to Y to export suppressed bibs that are linked to E-Items.
#  This flag will have no effect on suppressed bibs that are not
#  linked to E-Items, and will be ignored if the ExportEItemsWithCourseReserves
#  flag is set to N.
IncludeEItemSuppressedBibs=N

# NEW FOR 8.0.0
# The EItemAvailability flag determines the availability status value for all
#  exported E-Items. By default, this will be 'C' for 'check_availability', but
#  can also be set to 'A' for 'available' or 'U' for 'unavailable'.
EItemAvailability=C

# NEW FOR 8.0.0
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

```
# The destination directory where the *.tar.gz files will be placed
BundleDir=/m1/voyager/2008.1.0/qa810db/primo/Availability/bundled/


# NEW FOR 8.0.0
# The format of the name of the .tar and .tar.gz files - the group id and suffix will be
          appended
BundleFile=primo.$yyyyMMddHHmmss$


[XMLWriterProtocol]
# OAIPMHXmlWriterProtocol will include the bib id
#  as the second-to-last element in the file name before
#  the group id and the .xml extension.
# NEW as of 8.0.0 the PrimoExport application will write the *.tar.gz files
#  (see BundleDir= and BundleFile= above)
#  and the bundlePrimo.ksh script will no longer need to be run.
#  Therefore this variable will now only define the export file names,
#  and not the directory.
File=primo.export.$yyMMddhhmmss$.xml
XSL=
# max number of records to group in a single tar file
#  all in single group if 0 or undefined
recsPerGroup=1000
# prefix for record identifiers, must be present, may be empty
idPrefix=
#
# Set to Y to enable pretty printing of XML output.
# Caution: may affect data with embedded spaces
UsePrettyPrint=N


#
# EVERYTHING BELOW THIS LINE IS INTERNAL DO NOT MODIFY
```

Figure 2-1. PrimoExp-Availability.ini Example (Continued)

Publishing Configuration File

The following is an example of the publishing configuration file:

```
# Configuration for Voyager Export for Primo -- Publishing


# Voyager status maps to (A)vailable or (U)navailable for Primo
[Item Statuses]
Not Charged=A
Charged=U
Renewed=U
Overdue=U
Recall Request=U
Hold Request=U
On Hold=U
In Transit=U
In Transit Discharged=U
In Transit On Hold=U
Discharged=A
Missing=U
Lost--Library Applied=U
Lost--System Applied=U
Claims Returned=U
Damaged=U
Withdrawn=U
At Bindery=U
Cataloging Review=A
Circulation Review=A
Scheduled=U
In Process=U
Call Slip Request=U
Short Loan Request=U
Remote Storage Request=U


[Excluded Happening Locations]
# Items whose most recent activity took place in one of these locations will not be
          exported.
```

Figure 2-2. PrimoExp-Publishing.ini Example

```
#ACQ
#ACQ TEMP
#illcirc


[Excluded Operator Ids]
# Items whose most recent activity was recorded by one of these operators will not be
          exported.


[LocationCodesForBibInclusion]
# If any location codes are listed here, only bib records that are linked to
# holdings at the specified locations will be included in the extract.


[LocationCodesForAVA]
# If any location codes are listed here, availability statements will only be generated
          for items
# and e-items that are linked to these locations. (Items via perm or temp location, e-
          items by
# holdings location.) Also, Course Reserve information will only be generated if the
          course reserve
# list location is among these locations.


[Institution Id]
# This value will be included in the Availability tag
# as subfield a
Id=qa810db


[Location to Primo]
# This section maps Voyager locations to Primo locations.


[Enrichment Tags]
# Tags added to marc record with additional data
#
# WARNING: You must change the orderInfoHolTag=960 in vxws.properties if you use the 960
          tag in here.
#
# AVA tag contains the availability information
# AUT tag contains name cross-reference information
# SUB tag contains subject cross-reference information
# CR tag contains course reserve information
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
# DPS tag contains Digital Preservation System (Product Rosetta) information
# EITM tag contains E-Item availability data (New for 8.0.0)
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
#Inserted by voy821 patch
# InsertPrimoBrowseTags, if set to Y, will include see references for
# subjects and authors that will allow Primo to construct browse searches
# from them. When InsertPrimoBrowseTags is Y, then the InsertAUTTag and
# InsertSUBTag will not be created since Primo can enrich the keyword search
# from the included Browse tags.
InsertPrimoBrowseTags=N

#=========================================================================
# Preservation related object link definition
# This includes the label for the URL link, the URL link, and parameter name.
# The only supported dynamic parameter at this time is id (i.e. bib id).
# The parameter is passed to the third-party application that processes and retrieves
# intellectual entities from the Preservation repository.
#=========================================================================
[DPS]
DPSLinkLabel=Rosetta Link:
DPSLinkAddress=http://somecomponent.com/content-aggregator/getIEs?system=ilsdb&
DPSLinkParm=id=



[PrimoExp]
# note that stanza.name can be used with -- on the command line
# argv(case insensitive) override
# ini(case insensitive) override
# env (all uppercase)
#
# these parameters provide defaults for this run of the job
# each of them can be overridden using a command-line parameter
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
#
# BibRangeBegin and BibRangeEnd are the beginning and ending bibliographic ID numbers to
          export.
# -B overrides this parameter on the command line
# BibRangeBegin
#BibRangeBegin=1
# -E overrides this parameter on the command line
#BibRangeEnd=200
#
# BibsFromFile reads the file at the supplied path for a list of bibliographic record ids
          to export.
# The file should contain one bibliographic record ID per line.
# -F overrides this parameter on the command line
# BibsFromFile=/m1/voyager/2008.1.0/qa810db/local/PrimoBibs.txt
#
# ChangedSince exports records changed since the supplied date.
# ChangedSince may be either
#     a timestamp in YYYYMMDD.HHMMSS format
#    or a path to a file containing a timestamp in YYYYMMDD.HHMMSS format on a single line
# If a path to a file is supplied, the current run time will be saved in the file when
# the job finishes.
# -C overrides this parameter on the command line
# Example:
# ChangedSince=YYYYMMDD.HHMMSS
#  or
# ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Availability/ChangedSince.txt
ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Publishing/ChangedSince.txt
#
#
# HeadingChanges determines whether to export bibliographic records
# whose cross-references have changed
# -H Y|N overrides this parameter on the command line
HeadingChanges=Y
#
# AvailFilter detemines whether to export records whose availability
# fields have not changed since the last time the job ran.
# If AvailFilter is Y, only records whose availability has changed are exported;
# if AvailFilter is N, all changed records are exported.
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
# -A Y|N overrides this parameter on the command line
AvailFilter=N
#
# for filenames
# note that $str$ will be passed to date format so that you can
# specify date in the these files
#
# LogFile is the location of the log file.
# Note that $str$ will be passed to Java's Date.Format()
# function to control the datestamp embedded in the file name.
# -L overrides this parameter on the command line
LogFile=primo.export.$yyMMdd$.log
LogFileDir=/m1/voyager/2008.1.0/qa810db/primo/Publishing/logs
#
#
# -v
# the logging level
LogLevel=10
# LogToStdOut determines whether or not to write the log output to standard output,
# as well as the log file.
LogToStdOut=N

# Database and the other Database parameters determine which database to connect to.
# Ordinarily these are set by the wrapper script based on the environment.
# -d
# DataBase=
#
# DatabaseHost
DatabaseHost=10.100.2.112
#
# DatabasePort
# -e
# UserPass
# The credentials needed to connect to the database.
# Optional parameter, override on command line with -u
# STRONG RECOMMENDATION: do not set this value here, use
# the command line override instead.
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
# Special note: Oracle TNS Alias (@DB) is _not_ used here
#UserPass=qa810db/qwrite0
#
Database=VGER
Protocol=com.endinfosys.voyager.extract.OAIPMHXmlWriterProtocol

# must do items, then mfhds, then bibs
Task=com.endinfosys.voyager.extract.ExtractCourseReservesPrimo
Task=com.endinfosys.voyager.extract.ExtractItemsPrimo
Task=com.endinfosys.voyager.extract.ExtractMFHDsPrimo
Task=com.endinfosys.voyager.extract.ExtractBibsPrimo
DoItems=Y
DoMfhds=Y

# The Del* Dir and File are used to determine when records are deleted.
# These are standard Voyager locations.
DelBibsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelBibFile=deleted.bib.marc
DelMFHDsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelMFHDsFile=deleted.mfhd.marc
#Import delete file has the form delete.imp.YYYYMMDD.HHMM
DelMFHDsImpFileBase=delete.imp
DelItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelItemsFile=delete.item
# NEW FOR 8.0.0
DelEItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelEItemsFile=deleted.e-items

# Set to Y to include course reserve data.
CourseReserves=N

# NEW FOR 8.1.0
# Set to Y to export suppressed bibs that are linked to reserve list items.
#  This flag will have no effect on suppressed bibs that are not linked to
#  reserve list items, and will be ignored if the CourseReserves flag is
#  set to N.
IncludeReserveListItemsSuppressedBibs=N
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
# NEW FOR 8.0.0
# Set to Y to include E-Item data.
#  If this flag is set to Y and the CourseReserves flag is set to N,
#  then this flag overrides the value of the CourseReserves flag,
#  as E-Items are linked to course reserves.
ExportEItemsWithCourseReserves=N

# NEW FOR 8.0.0
# Set to Y to export suppressed bibs that are linked to E-Items.
#  This flag will have no effect on suppressed bibs that are not
#  linked to E-Items, and will be ignored if the ExportEItemsWithCourseReserves
#  flag is set to N.
IncludeEItemSuppressedBibs=N

# NEW FOR 8.0.0
# The EItemAvailability flag determines the availability status value for all
#  exported E-Items. By default, this will be 'C' for 'check_availability', but
#  can also be set to 'A' for 'available' or 'U' for 'unavailable'.
EItemAvailability=C

# NEW FOR 8.0.0
# The destination directory where the *.tar.gz files will be placed
BundleDir=/m1/voyager/2008.1.0/qa810db/primo/Publishing/bundled/

# NEW FOR 8.0.0
# The format of the name of the .tar and .tar.gz files - the group id and suffix will be
          appended
BundleFile=primo.$yyyyMMddHHmmss$

[XMLWriterProtocol]
# OAIPMHXmlWriterProtocol will include the bib id
#  as the second-to-last element in the file name before
#  the group id and the .xml extension.
# NEW as of 8.0.0 the PrimoExport application will write the *.tar.gz files
#  (see BundleDir= and BundleFile= above)
#  and the bundlePrimo.ksh script will no longer need to be run.
#  Therefore this variable will now only define the export file names,
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

```
#  and not the directory.
File=primo.export.$yyMMddhhmmss$.xml
XSL=
# max number of records to group in a single tar file
#  all in single group if 0 or undefined
recsPerGroup=1000
# prefix for record identifiers, must be present, may be empty
idPrefix=
#
# Set to Y to enable pretty printing of XML output.
# Caution: may affect data with embedded spaces
UsePrettyPrint=N

#
# EVERYTHING BELOW THIS LINE IS INTERNAL DO NOT MODIFY
```

Figure 2-2. PrimoExp-Publishing.ini Example (Continued)

Selective Configuration File

The following is an example of the selective configuration file:

```
# Configuration for Voyager Export for Primo -- Selective


# Voyager status maps to (A)vailable or (U)navailable for Primo
[Item Statuses]
Not Charged=A
Charged=U
Renewed=U
Overdue=U
Recall Request=U
Hold Request=U
On Hold=U
In Transit=U
In Transit Discharged=U
In Transit On Hold=U
Discharged=A
Missing=U
Lost--Library Applied=U
Lost--System Applied=U
Claims Returned=U
Damaged=U
Withdrawn=U
At Bindery=U
Cataloging Review=A
Circulation Review=A
Scheduled=U
In Process=U
Call Slip Request=U
Short Loan Request=U
Remote Storage Request=U


[Excluded Happening Locations]
# Items whose most recent activity took place in one of these locations will not be
          exported.
```

Figure 2-3. PrimoExp-Selective.ini Example

```
#ACQ
#ACQ TEMP
#illcirc


[Excluded Operator Ids]
# Items whose most recent activity was recorded by one of these operators will not be
          exported.


[LocationCodesForBibInclusion]
# If any location codes are listed here, only bib records that are linked to
# holdings at the specified locations will be included in the extract.


[LocationCodesForAVA]
# If any location codes are listed here, availability statements will only be generated for
          items
# and e-items that are linked to these locations. (Items via perm or temp location, e-items
          by
# holdings location.) Also, Course Reserve information will only be generated if the
          course reserve
# list location is among these locations.


[Institution Id]
# This value will be included in the Availability tag
# as subfield a
Id=qa810db


[Location to Primo]
# This section maps Voyager locations to Primo locations.


[Enrichment Tags]
# Tags added to marc record with additional data
#
# WARNING: You must change the orderInfoHolTag=960 in vxws.properties if you use the 960
          tag in here.
#
# AVA tag contains the availability information
# AUT tag contains name cross-reference information
# SUB tag contains subject cross-reference information
# CR tag contains course reserve information
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
# DPS tag contains Digital Preservation System (Product Rosetta) information
# EITM tag contains E-Item availability data (New for 8.0.0)
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
#Inserted by voy821 patch
# InsertPrimoBrowseTags, if set to Y, will include see references for
# subjects and authors that will allow Primo to construct browse searches
# from them. When InsertPrimoBrowseTags is Y, then the InsertAUTTag and
# InsertSUBTag will not be created since Primo can enrich the keyword search
# from the included Browse tags.
InsertPrimoBrowseTags=N

#=========================================================================
# Preservation related object link definition
# This includes the label for the URL link, the URL link, and parameter name.
# The only supported dynamic parameter at this time is id (i.e. bib id).
# The parameter is passed to the third-party application that processes and retrieves
# intellectual entities from the Preservation repository.
#=========================================================================
[DPS]
DPSLinkLabel=Rosetta Link:
DPSLinkAddress=http://somecomponent.com/content-aggregator/getIEs?system=ilsdb&
DPSLinkParm=id=



[PrimoExp]
# note that stanza.name can be used with -- on the command line
# argv(case insensitive) override
# ini(case insensitive) override
# env (all uppercase)
#
# these parameters provide defaults for this run of the job
# each of them can be overridden using a command-line parameter
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
#
# BibRangeBegin and BibRangeEnd are the beginning and ending bibliographic ID numbers to
          export.
# -B overrides this parameter on the command line
# BibRangeBegin
#BibRangeBegin=1
# -E overrides this parameter on the command line
#BibRangeEnd=200
#
# BibsFromFile reads the file at the supplied path for a list of bibliographic record ids
          to export.
# The file should contain one bibliographic record ID per line.
# -F overrides this parameter on the command line
BibsFromFile=/m1/voyager/2008.1.0/qa810db/local/PrimoBibs.txt
#
# ChangedSince exports records changed since the supplied date.
# ChangedSince may be either
#     a timestamp in YYYYMMDD.HHMMSS format
#    or a path to a file containing a timestamp in YYYYMMDD.HHMMSS format on a single line
# If a path to a file is supplied, the current run time will be saved in the file when
# the job finishes.
# -C overrides this parameter on the command line
# Example:
# ChangedSince=YYYYMMDD.HHMMSS
#  or
# ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Availability/ChangedSince.txt
ChangedSince=/m1/voyager/2008.1.0/qa810db/primo/Selective/ChangedSince.txt
#
#
# HeadingChanges determines whether to export bibliographic records
# whose cross-references have changed
# -H Y|N overrides this parameter on the command line
HeadingChanges=N
#
# AvailFilter detemines whether to export records whose availability
# fields have not changed since the last time the job ran.
# If AvailFilter is Y, only records whose availability has changed are exported;
# if AvailFilter is N, all changed records are exported.
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
# -A Y|N overrides this parameter on the command line
AvailFilter=N
#
# for filenames
# note that $str$ will be passed to date format so that you can
# specify date in the these files
#
# LogFile is the location of the log file.
# Note that $str$ will be passed to Java's Date.Format()
# function to control the datestamp embedded in the file name.
# -L overrides this parameter on the command line
LogFile=primo.export.$yyMMdd$.log
LogFileDir=/m1/voyager/2008.1.0/qa810db/primo/Selective/logs
#
#
# -v
# the logging level
LogLevel=10
# LogToStdOut determines whether or not to write the log output to standard output,
# as well as the log file.
LogToStdOut=N

# Database and the other Database parameters determine which database to connect to.
# Ordinarily these are set by the wrapper script based on the environment.
# -d
# DataBase=
#
# DatabaseHost
DatabaseHost=10.100.2.112
#
# DatabasePort
# -e
# UserPass
# The credentials needed to connect to the database.
# Optional parameter, override on command line with -u
# STRONG RECOMMENDATION: do not set this value here, use
# the command line override instead.
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
# Special note: Oracle TNS Alias (@DB) is _not_ used here
#UserPass=qa810db/qwrite0
#
Database=VGER
Protocol=com.endinfosys.voyager.extract.OAIPMHXmlWriterProtocol

# must do items, then mfhds, then bibs
Task=com.endinfosys.voyager.extract.ExtractCourseReservesPrimo
Task=com.endinfosys.voyager.extract.ExtractItemsPrimo
Task=com.endinfosys.voyager.extract.ExtractMFHDsPrimo
Task=com.endinfosys.voyager.extract.ExtractBibsPrimo
DoItems=Y
DoMfhds=Y

# The Del* Dir and File are used to determine when records are deleted.
# These are standard Voyager locations.
DelBibsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelBibFile=deleted.bib.marc
DelMFHDsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelMFHDsFile=deleted.mfhd.marc
#Import delete file has the form delete.imp.YYYYMMDD.HHMM
DelMFHDsImpFileBase=delete.imp
DelItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelItemsFile=delete.item
# NEW FOR 8.0.0
DelEItemsDir=/m1/voyager/2008.1.0/qa810db/rpt
DelEItemsFile=deleted.e-items

# Set to Y to include course reserve data.
CourseReserves=N

# NEW FOR 8.1.0
# Set to Y to export suppressed bibs that are linked to reserve list items.
#  This flag will have no effect on suppressed bibs that are not linked to
#  reserve list items, and will be ignored if the CourseReserves flag is
#  set to N.
IncludeReserveListItemsSuppressedBibs=N
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
# NEW FOR 8.0.0
# Set to Y to include E-Item data.
#  If this flag is set to Y and the CourseReserves flag is set to N,
#  then this flag overrides the value of the CourseReserves flag,
#  as E-Items are linked to course reserves.
ExportEItemsWithCourseReserves=N

# NEW FOR 8.0.0
# Set to Y to export suppressed bibs that are linked to E-Items.
#  This flag will have no effect on suppressed bibs that are not
#  linked to E-Items, and will be ignored if the ExportEItemsWithCourseReserves
#  flag is set to N.
IncludeEItemSuppressedBibs=N

# NEW FOR 8.0.0
# The EItemAvailability flag determines the availability status value for all
#  exported E-Items. By default, this will be 'C' for 'check_availability', but
#  can also be set to 'A' for 'available' or 'U' for 'unavailable'.
EItemAvailability=C

# NEW FOR 8.0.0
# The destination directory where the *.tar.gz files will be placed
BundleDir=/m1/voyager/2008.1.0/qa810db/primo/Selective/bundled/

# NEW FOR 8.0.0
# The format of the name of the .tar and .tar.gz files - the group id and suffix will be
          appended
BundleFile=primo.$yyyyMMddHHmmss$

[XMLWriterProtocol]
# OAIPMHXmlWriterProtocol will include the bib id
#  as the second-to-last element in the file name before
#  the group id and the .xml extension.
# NEW as of 8.0.0 the PrimoExport application will write the *.tar.gz files
#  (see BundleDir= and BundleFile= above)
#  and the bundlePrimo.ksh script will no longer need to be run.
#  Therefore this variable will now only define the export file names,
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

```
#  and not the directory.
File=primo.export.$yyMMddhhmmss$.xml
XSL=
# max number of records to group in a single tar file
#  all in single group if 0 or undefined
recsPerGroup=1000
# prefix for record identifiers, must be present, may be empty
idPrefix=
#
# Set to Y to enable pretty printing of XML output.
# Caution: may affect data with embedded spaces
UsePrettyPrint=N


#
# EVERYTHING BELOW THIS LINE IS INTERNAL DO NOT MODIFY
```

Figure 2-3. PrimoExp-Selective.ini Example (Continued)

Stanza Definitions for Voyager Primo Integration Configuration Files

This section defines the configuration file stanzas that need to be customized for Voyager Primo integration.

The following is a list of the stanzas:

- [Item Statuses]
- [Excluded Happening Locations]
- [Excluded Operator Ids]
- [LocationCodesForBibInclusion]
- [LocationCodesForAVA]
- [Institution Id]
- [Location to Primo]
- [Enrichment Tags]
- [DPS]
- [PrimoExp]
- [XMLWriterProtocol]

[Item Statuses]

The `[Item Statuses]` stanza identifies which item statuses in Voyager are defined as Available (A) or Unavailable (U) for Primo's purposes. A change in status causes a record to be selected for the availability pipe.

See Table 2-1 for a list of the item statuses and default settings stored in the `[Item Statuses]` stanza of the configuration file.

Table 2-1. [Item Statuses] parameters

Parameter	Default Setting
Not Charged=	A
Charged=	U
Renewed=	U
Overdue=	U
Recall Request=	U
Hold Request=	U
On Hold=	U
In Transit=	U
In Transit Discharged=	U
In Transit On Hold=	U
Discharged=	A
Missing=	U
Lost--Library Applied=	U
Lost--System Applied=	U
Claims Returned=	U
Damaged=	U
Withdrawn=	U
At Bindery=	U
Cataloging Review=	A
Circulation Review=	A
Scheduled=	U
In Process=	U
Call Slip Request=	U

Table 2-1. [Item Statuses] parameters

Parameter	Default Setting
Short Loan Request=	U
Remote Storage Request=	U

CAUTION:
Changes to these parameters after your institution has an established ongoing update/upload process may cause a significant data upload to the Primo database utilizing system resources as needed to complete the process.

[Excluded Happening Locations]

The `[Excluded Happening Locations]` stanza is used to identify the records associated with these locations to be excluded from consideration for uploading to the Primo database. These are location codes as established in Voyager System Administration.

The locations identified in the default configuration file are there as an example. Remove the commenting to use these examples and/or delete any that do not apply to your institution and enter your own location codes.

NOTE:
Only one location code may be entered per line.

[Excluded Operator Ids]

The `[Excluded Operator Ids]` stanza is used to identify the records associated with these operator ID(s) to be excluded from consideration for uploading to the Primo database. Enter one operator ID per line.

[LocationCodesForBibInclusion]

The `[LocationCodesForBibInclusion]` stanza is used to identify the records of specific locations for extraction. This may be useful for institutions that share a single Voyager database with one instance of Primo installed or institutions with a single, multiclustered Voyager database.

For this stanza, you provide a list of MFHD location codes (as identified in Voyager System Administration) whose attached bibliographic records should be extracted for Primo. This is based on the 852‡b of any attached MFHDs.

All location codes, including those defined as Circulation or Acquisitions happening locations, may be specified in the `[LocationCodesForBibInclusion]` stanza.

NOTE:
The extraction process, however, does not check if the bibliographic records can be used, that is, circulated, for example, by a happening location. The extraction process is only checking the location code in the 852‡b of the MFHD.

If a bibliographic record has, at least, one attached MFHD whose 852‡b matches a location in the `[LocationCodesForBibInclusion]` stanza list of included MFHD locations, the record is extracted. Otherwise, the record is not extracted.

NOTE:
There may be cases where the bibliographic record has no locations. In this situation, you may need to first run an extract without using the limitation of location codes.

Other considerations when using the `[LocationCodesForBibInclusion]` stanza:

- If a bibliographic record has multiple MFHDs, where one or more but not all location codes match a location in the `[LocationCodesForBibInclusion]` stanza list, the bibliographic record is extracted regardless of whether the owning library of the included MFHD matches the owning library of the bibliographic record. To limit which holdings are extracted for this bibliographic record, see the [LocationCodesForAVA] section.
- Item permanent and temporary locations are not part of the check, only the location in the MFHD 852‡b.
- When there is more than one 852‡b fields in a valid record, only the first occurrence is checked for the extraction.

The location codes listed with this stanza should be entered one per line/row. You may also choose not to specify any location codes and leave this stanza blank. When no location codes are specified, the extraction process pulls all bibliographic records taking into account any other specified limitations in the PrimoExp-*.ini files such as `BibRangeBegin=` and `BibRangeEnd=` and so forth.

[LocationCodesForAVA]

The [LocationCodesForAVA] stanza is used in combination with the [LocationCodesForBibInclusion] stanza. Once the list of extracted bibliographic records has been identified, the process also needs to determine what information should be included in the extracted record. The [LocationCodesForAVA] stanza serves that purpose.

For location codes listed in the [LocationCodesForAVA] stanza, the following is extracted:

- Availability (AVA/949)

  This includes MFHD/item availability statements whose location codes (852‡b or 949/AVA‡j) are included in the list of MFHD locations in the [LocationCodesForAVA] stanza.

- E-Items (EITM/954)

  This includes MFHD/item availability statements whose location codes (852‡b or 954/EITM‡j) are included in the list of MFHD locations in the [LocationCodesForAVA] stanza.

- Course information (CR/952)

  This includes the course reserve information if the course reserve list location (952/CR‡d) is included in the [LocationCodesForAVA] stanza.

- Digital Preservation/Rosetta (DPS/953)

  This includes all DPS tags. Since there is no way to determine location, all are included.

- Authority enrichments (AUT/950 and SUB/951 tags)

  Since there is no way to determine location, all are included.

For the [LocationCodesForAVA] stanza component of the extraction process, the item permanent and temporary locations are taken into account since they are included in a separate AVA/949‡j tag when present.

Example

When the Voyager MARC record contains:

- MFHD location = Main
- Item 1 permanent location = Main
- Item 2 permanent location = Reserve
- Item 3 temporary location = Cataloging repair

The following AVA/949 fields are generated (when nothing is specified in the `[LocationCodesForAVA]` stanza):

- ‡j Main
- ‡j Reserve
- ‡j Cataloging repair

In the final analysis of the Voyager Primo extraction process, when the `[LocationCodesForAVA]` stanza specifies the MFHD location code of Main, only the AVA/949 with Main in ‡j is extracted and sent to Primo. The holdings/item statements for ‡j Reserve and ‡j Cataloging repair are not sent in the final extract to Primo.

[LocationCodesForBibInclusion]/ [LocationCodesForAVA] Usage Examples

The ability to specify a different set of locations for extract versus included information is necessary for different consortial situations. This section provides two comparative examples, Situation A and Situation B.

Situation A

For this example, the institution belongs to a consortium and wants only its bibliographic records and holdings to show in its discovery program. To do this, the institution includes a limited list of location codes in the `[LocationCodesForBibInclusion]` stanza and the same limited list of location codes in the `[LocationCodesForAVA]` stanza.

For situation A, location code X is specified for both the `[LocationCodesForBibInclusion]` stanza and the `[LocationCodesForAVA]` stanza. See results in Table 2-2.

Table 2-2. Situation A Results

Voyager Database Contains	Exported	Primo
Bib 1 MFHD: X, Y, Z	Bib 1 MFHD: X	Bib 1 MFHD: X
Bib 2 MFHD: Y, Z		
Bib 3 MFHD: X	Bib 3 MFHD: X	Bib 3 MFHD: X

Situation B

For this example, the institution belongs to a consortium and has purchased Primo as a consortium. The intent is to only extract bibliograhic records separately because of availability update differences or other circumstances. If the bibliographic record gets sent to Primo, all locations for the consortium should be included on the bibliographic record. In this case, a limited list of locations is included for the `[LocationCodesForBibInclusion]` stanza and the consortium-wide list of locations for the `[LocationcodesForAVA]` stanza.

For situation B, location code X is specified for the `[LocationCodesForBibInclusion]` stanza and X, Y, and Z for the `[LocationCodesForAVA]` stanza. See results in Table 2-3.

Table 2-3. Situation B Results

Voyager Database Contains	Exported	Primo
Bib 1 MFHD: X, Y, Z	Bib 1 MFHD: X, Y, Z	Bib 1 (overwritten on import) MFHD: X, Y, Z
Bib 2 MFHD: Y, Z		Bib 2 (not overwritten; this was present before) MFHD: Y, Z
Bib 3 MFHD: X	Bib 3 MFHD: X	Bib 3 (overwritten on import) MFHD: X

If you are going to replace the bibliographic records that belong to X, you need to be sure to include all of the MFHDs that everyone searches for in Primo.

Changed Since in a Shared Database Environment

If you are in situation B, where the same database is exporting records in different location groupings, you may need to avoid using a single `ChangedSince.txt` file that is consulted for each separate run. As an alternative, you can submit the `ChangedSince` date using the command line parameters.

[Institution Id]

The `[Institution Id]` stanza is used to specify your institution. The institution ID is included as a subfield in the Availability tag of records prepared for the Primo database and is used by Primo to determine ownership of the records.

The Voyager institution ID is set one time at system installation. Verify this setting to insure its accuracy.

[Location to Primo]

The `[Location to Primo]` stanza is used to maintain consistency within Primo for item location codes. Primo is capable of retrieving records from many sources which may not be consistent in naming item location codes. The `[Location to Primo]` stanza can also be used by libraries to collapse locations or split locations for display in Primo.

Enter the Voyager item location code first and the Primo equivalent using the following format:

[Voyager Location Name]*=*[Primo Location Name]

See the following for an example:

Music Reserve=Reserve

Science Reserve=Reserve

Each Voyager/Primo item location code entry should be on a separate line in the configuration file.

NOTE:
Locations that are not listed in the `[Location to Primo]` stanza pass to Primo unchanged.

[Enrichment Tags]

The `[Enrichment Tags]` stanza is used to specify the tags used to include the availability, author enrichment, subject enrichment, course reserves, and Rosetta information in the extract.

By default, the configuration file makes the following assignments:

```
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
InsertPrimoBrowseTags=N
```

The InsertAVATag identifies the placement of availability information in the MARCXML output.

The InsertAUTTag identifies the placement of name/author cross-reference information.

The InsertSUBTag identifies the placement of subject cross-reference information.

The InsertCRTag is provided for course reserves data extraction. See InsertCRTag=952 Enrichment Tag on page 3-3 for more information.

The InsertDPSTag (953) identifies when a Voyager bibliographic record is stored in Rosetta as in the following example:

```
953‡a Exists in Rosetta
```

The InsertEITMTag contains (course reserves) e-item availability data.

The defaults in the `[Enrichment Tags]` stanza assume that your institution is not already using 949, 950, 951, 952, 953, and 954 in the MARC records of your database.

If, for example, your institution currently uses MARC 949 for another purpose, you can specify a different, unused 9XX number for the `InsertAVATag` in the `[Enrichment Tags]` stanza. The same logic applies to the other Enrichment Tags. Changing the default requires a matching change in the Primo configuration for harvesting.

CAUTION:
The orderInfoHolTag= in the vxws.properties file uses the 960 tag as its default. You may not duplicate tag numbers between the [Enrichment Tags] stanza in the PrimoExp.ini files and the orderInfoHolTag= setting in the vxws.properties file. However, you may change the tag number in either location to keep it unique. See orderInfoHolTag=960 on page 5-10 for more information.*

The `InsertPrimoBrowseTags=` option, when set to `Y`, directs the system to construct tags in the exported bibliographic record that allows Primo to generate browse indexes when the record is imported to Primo. When set to `Y`, additional tags that contain see references for subjects and authors are included in the exported record. The see references are based on local Voyager authority record information.

NOTE:
Since Primo can enrich the keyword search from the included browse tags, the InsertAUTTag and InsertSUBTag are not created, when `InsertPrimoBrowseTags=` is set to Y.

The default setting for `InsertPrimoBrowseTags=` is `N`.

[DPS]

The `[DPS]` stanza is used to identify the link to Rosetta information when it exists.

See Table 2-4 for a description of the parameters used to construct the 856 tag with Rosetta link information.

Table 2-4. [DPS] parameter descriptions

Parameter	Description
DPSLinkLabel	Use this parameter to specify the label text. The default text is as follows: `Rosetta Link:`
DPSLinkAddress	Use this parameter to identify the link address for Rosetta information. The following is an example of the link format: `http://somecomponent.com/content-aggregator/getIEs?system=ilsdb&`
DPSLinkParm	Use this to specify the link parameter. The default is as follows: `id=`

[PrimoExp]

The `[PrimoExp]` stanza is used to identify database extract processing considerations such as number of records to process, where to log messages, whether to use filters and so forth.

The parameters identified in the `[PrimoExp]` stanza may be entered on the command line or set within the configuration file. Parameters specified on the command line override any settings within the configuration file.

IMPORTANT:
When you use the command line, the –f option for specifying the `*.ini` file must be listed first in the command line string. See Figure 2-4 for an example.

```
Pprimoexp.pub -f /m1/voyager/<yyydb>/ini/PrimoExp-Publishing.ini.test1 -B1 -E10
(where yyydb is your database name)
```

Figure 2-4. Example Command Line Entry

See Table 2-5 for a description of these parameters.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
BibRangeBegin= -B	Use this parameter to specify the bibliographic record ID number for the beginning of a range of bibliographic record IDs. Use this with -E. Used in combination with -E, you are able to process a subset of records from the entire database. **IMPORTANT:** *HeadingChanges must be set equal to N when BibRangeBegin is used.*
BibRangeEnd= -E	Use this parameter to specify the bibliographic record ID number for the end of a range of bibliographic record IDs. Use this with -B. Used in combination with -B, you are able to process a subset of records from the entire database. **IMPORTANT:** *HeadingChanges must be set equal to N when BibRangeEnd is used.*
BibsFromFile= -F	Use this parameter to identify the name of a file that contains bibliographic records to be processed/stored in a file that is separate from the database. This assumes that the file is formatted with one bibliographic record ID per line/row.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
ChangedSince= -C	Use this parameter to indicate which Voyager records to evaluate for export to Primo based on a date/timestamp format as follows: YYYYMMDDHHMMSS where ... YYYY=year MM=Month DD=Day HH=Hours MM=Minutes SS=Seconds If no time is specified, 12:00AM is the default setting. Alternatively, store a date such as ***20070711*** and time in a file called since.txt with a path of your choosing and reference the since.txt file in [PrimoExp]. For example: ChangedSince=./test/work/since.txt **NOTE:** The system logs the following warnings/errors with the use of the ChangedSince= parameter: • If the ChangedSince= date is 8 or more days in the past, a warning is logged; but the extract runs. • If the ChangedSince= date is 181 or more days in the past, this causes a fatal error. The extract logs an error and stops. **NOTE:** The ChangedSince for bibliographic records looks at BIB_HISTORY where ACTION_DATE is later than the date entered and where ACTION_TYPE_ID is 1, 2, 3, or 4 (create, update, merge, or replace). For MFHD (holdings records), it looks to MFHD_HISTORY where ACTION_DATE is later than the date entered and where ACTION_TYPE_ID is 1, 2, 3, or 4 (create, update, merge, or replace). For items, it looks at ITEM where the CREATE_DATE or MODIFY_DATE is later than the date entered. It also looks at ITEM_STATUS where the ITEM_STATUS_DATE is later than the date entered.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
HeadingChanges= -H	Use this parameter to indicate the following: • Yes (Y), find changed headings and include all records affected. • No (N), do not look for changed headings. This parameter should be set to Y when the configuration file is used to harvest records for the Primo publishing pipe. This parameter should be set to N when the configuration file is used to harvest records for the Primo availability pipe. **IMPORTANT:** *The HeadingChanges parameter requires that a setting is specified for the ChangedSince parameter. HeadingChanges can only be processed if there is a ChangedSince date.* **NOTE:** If an entry in HEADING has a CREATE_DATE later than the most recent extract and the OPACREFS field contains an S in any position, then any bibliographic records, in BIB_HEADING whose HEADING_ID matches AUTH_HEADING.HEADING_ID_POINTEE for AUTH_HEADING entries that match the headings HEADING_ID field, will need to be resent. (The S indicates that the heading has a See Reference. If the heading was updated, it may have an updated See Reference, and the See References are included in the record for the heading information in tags 950 and 951.)
AvailFilter= -A	Use this parameter to indicate which Voyager records to evaluate for export to Primo based on availability changes. Y=Yes (filter for availability changes) N=No (do not filter for availability changes) This parameter should be set to Y when the configuration file is used to harvest records for the Primo availability pipe. This parameter should be set to N when the configuration file is used to harvest records for the Primo publishing pipe.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description

LogFile=
-L
	Use this parameter to set the file name for the log file.
	For example:
	`primo.export.$yyMMdd$.log`
	The system automatically updates the `$yyMMdd$` with the date/time component of the file name.
	To specify that no log file is to be created, use the following parameter setting:
	`LogFile=/dev/null`
	  **TIP:**
	Use a naming convention that reflects whether the configuration file is being used for Voyager records to be exported to the Primo publishing pipe or the Primo availability pipe.

LogFileDir=	Use this parameter to set the path for the log file. **NOTE:** This is automatically set at installation.
LogLevel= -v	Leave the `LogLevel` parameter setting as specified in the configuration file unless directed otherwise by Ex Libris support staff.
LogToStdOut=	Leave the `LogToStdOut` parameter setting as specified in the configuration file unless directed otherwise by Ex Libris support staff.
Database= -d	Use this parameter to set the Oracle database name (SID). The default is **`VGER`**. For example: `-d VGER`
DatabaseHost= -s	Use this parameter to set the database host IP address (the same as in `voyager.env`). The `DatabaseHost` parameter is set one time at system installation and, in most cases, should not be changed. Verify this setting to insure its accuracy. For example: `-s 208.178.237.40`

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
DatabasePort= -p	Use this parameter to set the database port. The `DatabasePort` parameter is set one time at system installation and, in most cases, should not be changed. Verify this setting to insure its accuracy. For example: `-p 1521`
UserPass= -u	Use this parameter to specify the Oracle database user name and password. The `UserPass` parameter is set one time at system installation and, in most cases, does not need to be changed. Verify this setting to insure its accuracy. For example: `-u user/pass`
DelBibsDir=	Use this parameter to specify the directory in which the deleted.bib.marc files can be found. **NOTE:** This is automatically set at installation.
DelBibFile=	Leave this parameter setting as specified in the configuration file for deleted.bib.marc files.
DelMFHDsDir=	Use this parameter to specify the directory in which the deleted.mfhd.marc files can be found. **NOTE:** This is automatically set at installation.
DelMFHDsFile=	Leave this parameter setting as specified in the configuration file for deleted.mfhd.marc files.
DelItemsDir=	Use this parameter to specify the directory in which the deleted.items files can be found. **NOTE:** This is automatically set at installation.
DelItemsFile=	Leave this parameter setting as specified in the configuration file for delete.item files.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
DelEItemsDir=	Use this parameter to specify the directory in which the deleted.e-items files can be found. **NOTE:** This is automatically set at installation.
DelEItemsFile=	Leave this parameter setting as specified for the name of the deleted.e-items files.
ExportEItemsWith CourseReserves=	Set this parameter value to `Y` (Yes) to include e-item data. **NOTE:** If the `CourseReserves=` parameter is set to `N`, a `Y` value for the `ExportEItemsWithCourseReserves=` parameter overrides the `CourseReserves=` setting. The default value is `N`.
IncludeEItem SuppressedBibs=	Set this parameter value to `Y` (Yes) to export suppressed bibliographic records that are linked to active e-items. **NOTE:** This parameter has no effect on suppressed bibliographic records that are not linked to e-items. This parameter is ignored if `ExportEItemsWithCourseReserves=` is set to `N`. The default value is `N`.
EItemAvailability=	Use this parameter to set the status value for all exported e-items. The setting options are: • `C` (check availability) • `A` (available) • `U` (unavailable) The default value is `C`.
IncludeReserve ListItems SuppressedBibs=	Set this parameter value to `Y` (Yes) to export suppressed bibliographic records that are linked to reserve list items. **NOTE:** This parameter has no effect on suppressed bibliographic records that are not linked to reserve list items. This parameter is ignored if `CourseReserves=` is set to `N`. The default value is `N`.

Table 2-5. [PrimoExp] parameter/command descriptions

Parameter/Command	Description
BundleDir=	Use this parameter to specify the directory where the bundled `*.tar.gz` files (see `BundleFile=`) are to be placed. **NOTE:** This is automatically set at installation.
BundleFile=	Use this parameter to identify the naming convention for the `.tar` and `.tar.gz` files that contain the files/records being exported in groups of records determined in the `[XMLWriterProtocol]` stanza. **NOTE:** The naming convention for the `.xml` files that are bundled is defined in the `[XMLWriterProtocol]` stanza. The bundled file names are appended with the group ID and suffix

[XMLWriterProtocol]

The `[XMLWriterProtocol]` stanza defines the storage location of the XML records, number of records in the tar.gz file, and so forth for the XML records targeted for export to the Primo database.

See Table 2-6 for a description of the parameters used in the `[XMLWriterProtocol]` stanza.

Table 2-6. [XMLWriterProtocol] parameter descriptions

Parameter	Description
File=	Use this parameter to identify the file naming convention for the XML files generated from harvesting Voyager records for export to the Primo database. For example: `primo.export.$yyMMddhhmmss$.xml`
XSL=	Leave this parameter blank. No setting required.
recsPerGroup=	Use this parameter to identify the number of records you prefer to be saved in each `tar.gz` file. **NOTE:** Do not include more than 10,000 records in a single `.tar` file. There is a 10MB maximum, file-size limit for Primo throughput that is estimated to be 10,000 MARC records.

Table 2-6. [XMLWriterProtocol] parameter descriptions

Parameter	Description
idPrefix=	Leave this parameter blank. No setting required.
UsePrettyPrint=	Use this parameter, set to Y (Yes), for more readable XML files. **NOTE:** Using the pretty print Y option risks altering data with leading and trailing spaces. Use the N (No) option for machine-readable files when human review is not a requirement. The N (No) option is the default.

Pprimoexp Batch Job

The `Pprimoexp` batch job creates XML files for export that conform to the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) schema and places them in the location identified by the ***File=*** parameter set in the `[XMLWriterProtocol]` stanza. See Table 2-6.

The `Pprimoexp` batch job is located in `/m1/voyager/yyydb/sbin` where `yyydb` is the database name.

`Pprimoexp` runs the `PbundlePrimo` script that handles, for example, bundling individual records into `tar.gz` files and cleans up the exports directory.

Command Line Options

See Table 2-5 on page 2-39 for a list of available commands and the RequiredBib (-R) on page 2-47 and File Path (-f) on page 2-47 commands that are provided specifically for `Pprimoexp` batch job processing.

IMPORTANT:
When you use the command line, the –f option for specifying the `*.ini` file must be listed first in the command line string. See Figure 2-5 for an example.

```
Pprimoexp.pub -f /m1/voyager/<yyydb>/ini/PrimoExp-
         Publishing.ini.test1 -B1 -E10

(where yyydb is your database name)
```

Figure 2-5. Example command line entry

RequiredBib (-R)

When running the `Pprimoexp` batch job, you may optionally specify one or more Voyager bibliographic record IDs (similar to a selective extraction).

The format is as follows:

- For a single ID number where ### is the bibliographic record ID number.

  `-R #####`

- For multiple ID numbers where ### is the bibliographic record ID number commas need to separate the ID numbers.

  `-R #####,#####,#####,#####`

File Path (-f)

Use `-f` to specify the relative or full-path name to the PrimoExp*.ini file (* stands for Availability, Publishing, or Selective).

Threads4Range (-T)

Use `-T` to specify a number of threads/extract processes to run when using `BibRangeBegin=` and `BibRangeEnd=` such as `-T 3`.

The bibliographic range is subdivided by the number of threads specified with `-T`, and each thread works on one subset of the range. If the number of bibliographic records in the range is less than the number of threads specified, a single thread is run.

NOTE:
This command line option may only be used with `BibRangeBegin=` and `BibRangeEnd=`. If you attempt to use `-T` with `ChangedSince=`, `RequiredBib`, `BibsFromFile=`, or `HeadingChanges=`, you get a warning message and the extract runs a single thread.

Scheduling

The Voyager Primo integration setup is designed to give you the flexibility to establish more than one production schedule.

- One production job may be set to run once per day to allow for Primo harvesting of Voyager data with bibliographic record changes through the publishing pipe.

IMPORTANT:
This requires an update to the bibliographic index in the Primo environment.

- Another production schedule may be set to run several times per day such as once per hour to allow for exporting Voyager holdings records with availability changes.

NOTE:
Voyager availability data harvested through the Primo availability pipe only updates the availability information without changing the bibliographic information in Primo.

The `Pprimoexp` batch job schedule determines when Voyager records are examined for export based on the criteria set in the configuration file.

Separately, the Primo publishing and availability pipes are scheduled to FTP whatever `tar.gz` files are available for export. For more information, refer to Schedule Tasks in the *Primo Back Office Guide*.

Extract Log Summary Section

The summary section of the extract log contains a hierarchical structure for all the actions (see Table 2-7). For an example, the BIB-Changes adds a top-level filter and then the sub-actions (BIB-Changes SUPPRESSED, BIB-Changes UNSUPPRESSED, and BIB-Changes-AvailabilityHistory) add more filters. If the criteria matches, it performs the action.

Table 2-7. Hierarchical Structure in the Extract Log Summary Section

Component	Action
BIB-Changes	This includes changed bibliographic records since a specific date (ChangedSince date) with excluded locations and operators.

Table 2-7. Hierarchical Structure in the Extract Log Summary Section

Component	Action
BIB-Changes SUPPRESSED	This is called from the BIB-Changes method and includes only changed suppressed bibliographic records.
BIB-Changes SUPPRESSED-AvailabilityHistory ======== or BIB-Changes UNSUPPRESSED	This is called from the BIB-Changes SUPPRESSED and includes only changed suppressed bibliographic records with an entry in the `primo_avail` table. ======== This is called from the BIB-Changes method and includes changed unsuppressed bibliographic records.
BIB-Changes UNSUPPRESSED-AvailabilityHistory ======== or BIB-Changes-AvailabilityHistory	This is called from the BIB-Changes UNSUPPRESSED and includes only changed unsuppressed bibliographic records with an entry the in `primo_avail` table. ======== This is called from the BIB-Changes and includes only the changed records with an entry in the `primo_avail` table.
BIB-Deleted	This processes the deleted bibliograhic files looking for records that have been deleted since the last since time.
BIB-Deleted-AvailabilityHistory	This is called from the BIB-Deleted and includes only deleted records with an entry in the `primo_avail` table.
BIB-Required	This includes bibliographic records from either the `bib_heading` table since a specific date (ChangedSince date) or read from a file.
BIB-Required-AvailabilityHistory	This is called from the BIB-Required and includes required bibliographic records with an entry in the `primo_avail` table.
ITEM-ChangedSince	This includes changed item records since a specific date (ChangedSince date).
ITEM-Delete	This processes the deleted item files looking for records that have been deleted since the last since time.

Table 2-7. Hierarchical Structure in the Extract Log Summary Section

Component	Action
MFHD-Changes-OuterLoop	This includes changed MFHD records since a specific date (ChangedSince date) with excluded locations and operators.
MFHD-Deleted-Catalog	This processes the deleted MFHD files looking for records that have been deleted since the last since time.

Additional information that is provided in the extract log summary section is described in Table 2-8.

Table 2-8. Additional Information in the Extract Log Summary Section

Component	Description
Number of visits	Number of visits checks how many times the visit method was called. The visit method performs the required actions on the MARC record to update the transaction and record the error or success of the transaction.
Milisecs	Milisecs keeps track of how long it takes for each action.
Counts	Counts displays the total number of records processed for each action.

Course Reserves

3

Course Reserves

3

Introduction

The purpose of this chapter is to describe the characteristics of the course reserves data extraction capability and any Voyager configuration changes required to enable this capability.

Course Reserves Function Overview

With the addition of course reserves data to Voyager bibliographic record extraction, the user discovery experience within Primo is extended. In Primo, facets and search scopes can be created to allow course reserve searching for Voyager customers.

With the course reserves option enabled, extracted Voyager bibliographic records are enriched with course reserve information such as course number, course name, instructor name/title, effective date, expiration date, department name/code, and so forth. See Table 3-1 on page 3-3 for the extraction format and description.

To enable course reserves data extraction, the following is provided in the `PrimoExp<xxx>.ini` configuration file where `xxx` may be `Selective`, `Availability`, or `Publishing`:

- `CourseReserves=`.
- 952 tag in the `[Enrichment Tags]` stanza.

CourseReserves=

In order to specify that Voyager course reserves data should be included in the data extraction process, the `CourseReserves=` flag is provided in the `.ini` configuration file. See Figure 3-1.

```
# Set to Y to include course reserve data.
CourseReserves=N
```

Figure 3-1. CourseReserves= flag

Specify `Y` (Yes) for course reserves data to be evaluated for data extraction. The default specified in the configuration file is `N` (No).

When `Y` is specified for `CourseReserves=`, the value in the Voyager database `ITEM.on_reserve` flag is examined to determine if course reserves data is to be included in the bibliographic record extraction or excluded.

NOTE:
The standard Voyager procedures for keeping the course reserve flag current in the Voyager database must be followed for Primo results to be current, too. See Chapter 8 in the *Voyager Technical User's Guide* for more information regarding circulation batch job 34 (Place Items on Active Course Reserve List) and 36 (Take Items on Inactive Course Reserve List Off Reserve) that are the batch jobs used to maintain currency for course reserves.

If the `ITEM.on_reserve` flag is `Y` (Yes) for one of the bibliographic record's item records, the bibliographic record being generated for extraction includes course reserve data for every list to which the item is linked (when no `ChangeSince=` value has been specified). When there is a `ChangeSince=` value specified, an additional step is taken to evaluate the `RESERVE_ITEM_HISTORY` table to determine if course reserve data is to be extracted. See Pprimoexp Extract Processing on page 3-4 and Figures 3-3 and 3-4 for additional information regarding extract processing.

NOTE:
Item records can be linked to course reserves lists without being on reserve. This allows you to reuse lists without needing to recreate them every semester or term. Thus, the `ITEM.on_reserve` flag is the authoritative source for determining that an item is on reserve.

InsertCRTag=952 Enrichment Tag

For course reserves data extraction, the `InsertCRTag=952` is provided in the `[Enrichment Tags]` stanza of the `.ini` configuration file. See Figure 3-2.

```
[Enrichment Tags]
# tags added to marc record with additional data
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
```

Figure 3-2. 952 tag

The `[Enrichment Tags]` stanza assumes that your institution is not already using the 952 in the MARC records of your database. If, however, your institution currently uses MARC 952 for another purpose, you can specify a different, unused MARC field number for the `InsertCRTag`. See [Enrichment Tags] on page 2-36 for additional information.

When the course reserves extraction function is enabled, the `InsertCRTag=` option specifies the data extract format. See Table 3-1 for a description of the extracted course reserves subfield data format.

Table 3-1. 952 Extract

952 Subfields	Extract Description
952a	Reserve list name
952b	List effective date
952c	List expiration date
952d	List location display name
952e	Department name
952f	Department code
952g	Instructor last name
952h	Instructor first name

Table 3-1. 952 Extract

952 Subfields	Extract Description
952i	Instructor title
952j	Course name
952k	Course number
952l	Section

IMPORTANT:
Even though you may substitute a different unused MARC field number for 952, the subfield designations in Table 3-1 cannot be changed.

Some subfields may not be included in the extracted data. This is dependent on the course reserves data available in Voyager.

An item record may belong to multiple reserve lists. A 952 field is constructed for each reserve list to which the item belongs.

Pprimoexp Extract Processing

When the `Pprimoexp` batch job is run for data extraction, it checks for the `CourseReserves=` flag to determine if course reserves data extraction should be part of the extraction results. Other parameters set in the batch job may affect the extraction processing end results.

In general, specifying parameters such as `ChangedSince=` or `BibRangeBegin=` and `BibRangeEnd=` indicates that a subset of the entire Voyager database is being processed for extraction.

Specific to course reserves when `ChangedSince=` is specified in the extraction script, the system does a check of the `RESERVE_ITEM_HISTORY` table to determine if one of the bibliographic record's item records has an entry in the table with a date that falls after the `ChangeSince=` value specified before going on to check the `ITEM.on_reserve` value.

See Figure 3-3 on page 3-5 and Figure 3-4 on page 3-6 for illustrations of the extract process workflow.

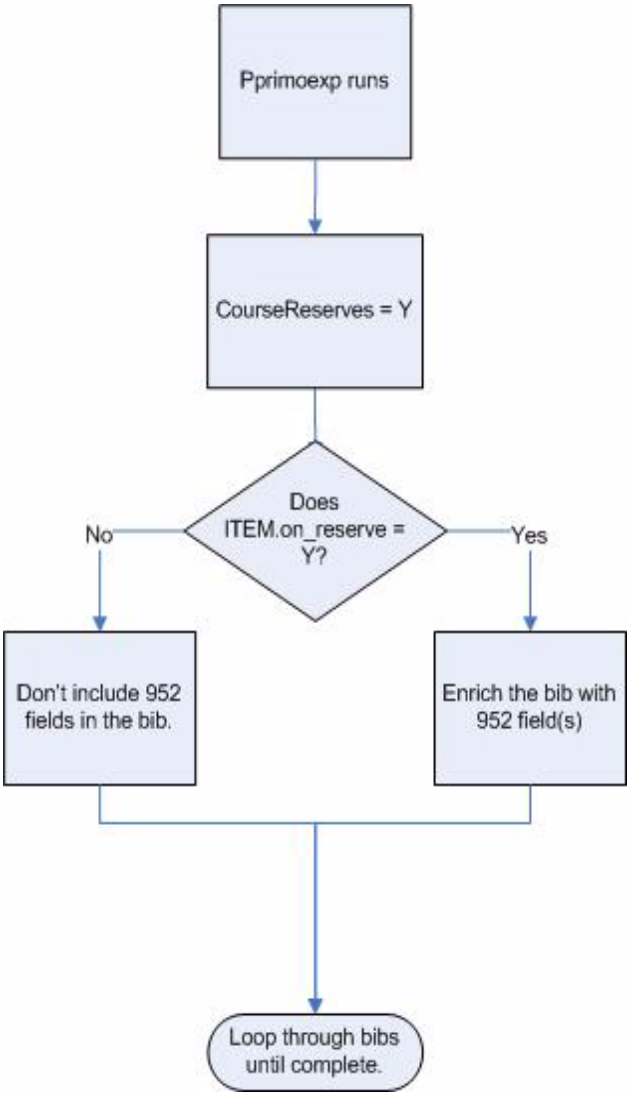

Figure 3-3. Extract processing without ChangedSince= specified

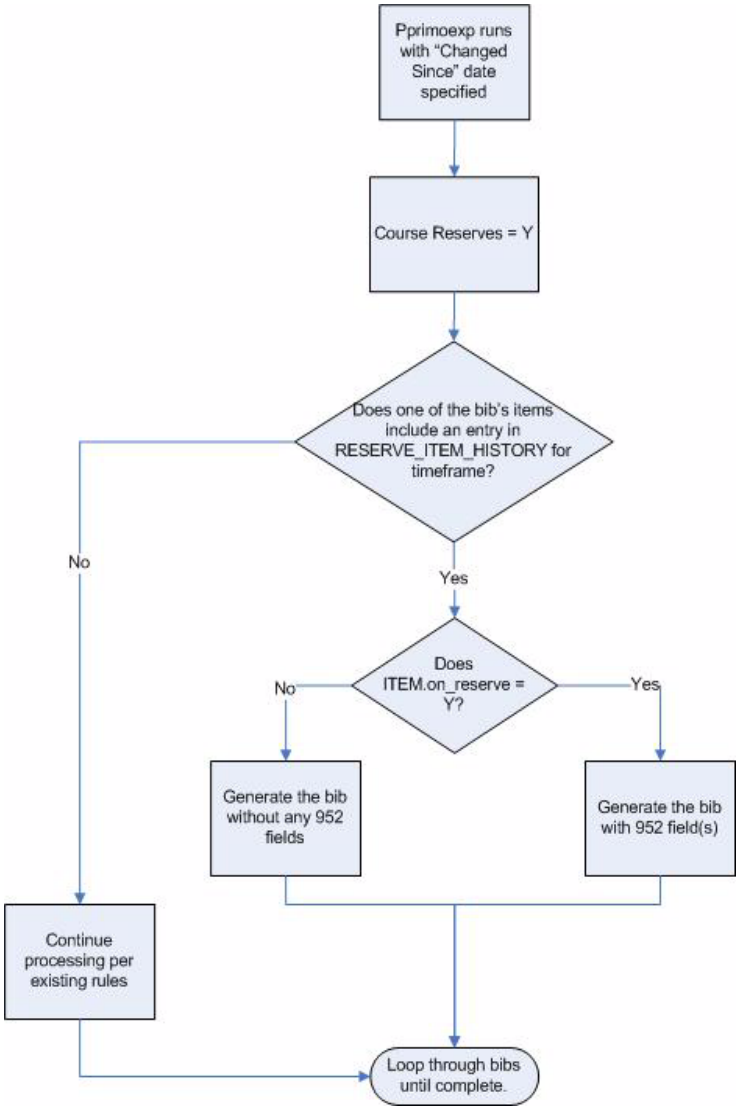

Figure 3-4. Extract processing with ChangedSince= specified

Course Reserves E-Items

E-items are a special type of item in Voyager created specifically for use with course reserves lists in Circulation. As a component of course reserves, e-items cannot exist by themselves. If a course reserves list is removed, access to the e-items are also removed.

Optionally, e-items may be attached to bibliographic records that are suppressed or not suppressed. When e-items are attached to bibliographic records that are suppressed, the e-items can only be accessed through the course reserves search in the OPAC. When e-items are attached to bibliographic records that are not suppressed, the e-items are available for search and viewing in the OPAC by everyone. Both of these options are supported by the Primo export files with the `IncludeEitemSuppressedBibs=` parameter. Specifically, these are configured in the `PrimoExp-Availability.ini`, `PrimoExp-Publishing.ini`, and `PrimoExp-Selective.ini` files (see the Data Extraction chapter starting on page 2-1).

Refer to the *Voyager Circulation User's Guide* for more information regarding course reserves and e-items.

Pprimoexp Extract

With the Primo extract, the system checks several places regarding the status of e-items to determine what is appropriate to export for patron viewing.

For the initial Primo export, e-items are included in the export that are attached to effective courses, that is, between the effective date and the expiration date.

For a Primo export that checks the `ChangedSince=` parameter for what has been changed since the last run, the system checks for:

- Deleted e-items
- Changed/edited e-items
- Reserve list changes (e-items added or removed)

  NOTE:
  E-items are deleted when removed from the last remaining Course Reserves list where they exist.

- Effective/expiration dates associated with course reserves lists (has a course gone off or on reserve)

IMPORTANT:
Do not expire a course reserves list earlier than the ChangedSince= date in the [PrimoExp] stanza when a ChangedSince= date is specified. If you do this, the expired course reserve list is not recognized as changed, and the associated e-items are not exported.

InsertEITMTag=954 Enrichment Tag

For course reserves data extraction, the `InsertEITMTag=954` is provided in the `[Enrichment Tags]` stanza of the `.ini` configuration file. See Figure 3-5.

```
[Enrichment Tags]
# tags added to marc record with additional data
InsertAVATag=949
InsertAUTTag=950
InsertSUBTag=951
InsertCRTag=952
InsertDPSTag=953
InsertEITMTag=954
```

Figure 3-5. 954 tag

The `[Enrichment Tags]` stanza assumes that your institution is not already using the 954 in the MARC records of your database. If, however, your institution currently uses MARC 954 for another purpose, you can specify a different, unused MARC field number for the `InsertEITMTag`. See [Enrichment Tags] on page 2-36 for additional information.

When the course reserves extraction function is enabled, the `InsertEITMTag=` option specifies the data extract format. See Table 3-2 for a description of the extracted e-item availability subfield data format.

NOTE:
When you export e-items, the system sends the 954 e-item tag content and the 952 tag content. The 952 tag contains the course reserves information (see InsertCRTag=952 Enrichment Tag on page 3-3).

Table 3-2. 954 Extract

954 Subfields	Extract Description
954a	Code supplied as a parameter to the job
954b	Display name of the Voyager owning library of the biblio-graphic record
954c	Display name of the Voyager location from the MFHD
954d	Voyager display call number from the MFHD

Table 3-2. 954 Extract

954 Subfields	Extract Description
954e	Availability/check holdings for all e-items You can specify what the availability statement is for all e-items with the option `EItemAvailability=` (see Table 2-5).
954f	Number of e-items associated with the availability field
954g	Empty
954h	Empty
954i	Empty
954j	Location code of the Voyager location from the MFHD

IMPORTANT:
Even though you may substitute a different unused MARC field number for 954, the subfield designations in Table 3-2 cannot be changed.

Some subfields may not be included in the extracted data. This is dependent on what is in the e-item.

An e-item record may belong to multiple reserve lists. A 952 tag is exported for each attached reserve list to which the item belongs.

ExportEItemsWithCourseReserves=

The `ExportEItemsWithCourseReserves=` parameter may be set to `Y` or `N` (see Table 2-5). The export job evaluates and processes the `ExportEItemsWithCourseReserves=` settings in the following manner:

- If the export job is run with `ExportEItemsWithCourseReserves=Y` and no `Changed Since=` date has been specified, the job includes all e-items that are defined in the e-item table and are on active courses.

  If an e-item exists and the linked course is past its effective date and before its expiration date then it is considered on reserve.

- If the job is run with the `ExportEItemsWithCourseReserves=Y` and there is a `Changed Since=` date, then

- The job queries the e-item update date field. If the date in this field falls after the `Changed Since=` date (the setting for when the previous extract was run) and after the date/time that the current extract job is run, then the bibliographic record associated with the e-item is included in the extract.
- The job queries the e-item reserve list update date field. If the date in this field falls after the `Changed Since=` date and the job date stamp, the bibliographic record associated with the e-item is included in the extract.
- The job also checks if the e-item has been deleted since the `Changed Since=` date. If the e-item has been deleted since the last run, the bibliographic record associated with the former e-item is included in the extract. However, the deleted e-item or course reserve information for this e-item in the bibliographic record is not included.

  NOTE:
  The history file is renamed after doing this.
- The job also checks if a course that is linked to all e-items has an on-reserve status that has changed since the `Changed Since=` date. In order to do this, all courses that are attached to e-items are identified, and checked for if either the expiration date or the effective date from the reserve list is later than the `Changed Since=` date, but not earlier than the time the extract job is run.

  If the effective date or the expiration date falls within the time period since the `Changed Since=` date, then the e-item is considered to have changed and should be included in the extract. The change is either that the e-item is on reserve (the effective date falls within the time period) or that the e-item is off reserve (the expiration date falls within the time period).

IMPORTANT:
Do not expire a course reserves list earlier than the ChangedSince= date in the [PrimoExp] stanza when a ChangedSince= date is specified. If you do this, the expired course reserve list is not recognized as changed, and the associated e-items are not exported.

IncludeEItemSuppressedBibs=

The `IncludeEItemSuppressedBibs=` parameter may be set to `Y` or `N` (see Table 2-5). The export job evaluates and processes the `IncludeEItemSuppressedBibs=` settings in the following manner:

- If the job is run with the `IncludeEItemSuppressedBibs=` parameter set to `N`, the bibliographic record and the corresponding e-items are not included in the extract. The suppressed flag on the bibliographic record stops the bibliographic record from being exported.
- If the job is run with the`IncludeEItemSuppressedBibs=` parameter set to `Y`, all bibliographic records attached to valid e-items, regardless of the suppressed flag status, are exported for inclusion in the Primo export. In this case, an additional tag is added to the exported bibliographic record as follows:

  ```
  Tag 999, no indicators, $a SUPPRESSED
  ```

IncludeReserveListItemsSuppressedBibs=

The `IncludeReserveListItemsSuppressedBibs=` parameter may be set to `Y` or `N` (see Table 2-5). The export job evaluates and processes the `IncludeReserveListItemsSuppressedBibs=` settings in the following manner:

- If the job is run with the `IncludeReserveListItemsSuppressedBibs=` parameter set to `N`, the bibliographic record and the corresponding reserve list items are not included in the extract. The suppressed flag on the bibliographic record stops the bibliographic record from being exported.
- If the job is run with the `IncludeReserveListItemsSuppressedBibs=` parameter set to `Y`, all bibliographic records attached to valid reserve list items, regardless of the suppressed flag status, are exported for inclusion in the Primo export. In this case, an additional tag is added to the exported bibliographic record as follows:

  ```
  Tag 999, no indicators, $a SUPPRESSED
  ```

Real-Time Availability

4

Real-Time Availability

4

Introduction

The Real-time Availability function in Primo provides dynamically-generated Voyager database availability information to the Primo user. Specifically, Primo communicates with Voyager to retrieve availability information for one or more bibliographic records that can be viewed on an individual record display or on a search results page with multiple records.

This chapter provides information regarding real-time availability processing with Voyager as a defined ILS in Primo and highlights other considerations like possible error conditions that you may encounter.

For information regarding Primo setup to trigger real-time availability communications with Voyager, refer to the *Primo Interoperability Guide*.

Real-Time Availability Processing

In general, the Primo/Voyager real-time availability processing occurs as follows:

1. Primo sends a request (URL format) to the Voyager ILS. This request includes the following:
    a. An `op` (operation) parameter that must be `publish_avail`.
    b. A `doc_num` parameter with a value of one or more bibliographic record ID numbers in a comma-separated list. The number of IDs is only limited by the practical limit of an http URI.

2. The Voyager availability service (VXWS) processes the request.

   Real-time availability processing utilizes the `PrimoExp.ini` configuration file when responding to real-time availability queries from Primo. This file is located in `/m1/voyager/yyydb/tomcat/vxws/ini/` where `yyydb` is the database name.

   See PrimoExp.ini on page 4-5 for additional information.

3. Successful processing returns bibliographic record information from the Voyager database in XML format using the Enrichment Tags (see [Enrichment Tags] on page 2-36) to determine the fields/data organization.

See Figure 4-1 for an example of the XML returned from a successfully processed request.

```
<?xml version="1.0" encoding="UTF-8" ?>
 <publish-avail>
 <OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.openarchives.org/OAI/
        2.0/ http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
 <ListRecords>
 <record>
 <header>
 <identifier> 000010000</identifier>
 </header>
 <metadata>
 <record xmlns="http://www.loc.gov/MARC21/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">
 </metadata>
 </record>
 </ListRecords>
 </publish-avail>
```

Figure 4-1. Example real-time availability XML returned to Primo

The value in the `<identifier>` key is the Voyager bibliographic record ID number.

Error Conditions

Error conditions have been identified for the following:

- Missing bibliographic record ID number (`doc_num`) in the URL request

  See Figure 4-2 for an example of the error response provided.

- URL request that contains an operation parameter other than `publish_avail`

  See Figure 4-3 for an example of the error response provided.

- Requested bibliographic record ID number does not exist in the Voyager database

  See Figure 4-4 for an example of the error response provided.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<publish-avail>
<error>doc_num must be included in parameters</error>
</publish-avail>
```

Figure 4-2. Example error response for missing bibliographic record ID number

```
<?xml version = "1.0" encoding = "UTF-8"?>
<login>
<error>Unrecognized op</error>
</login>
```

Figure 4-3. Example error response for incorrect operation parameter

```
<OAI-PMH xmlns=http://www.openarchives.org/OAI/2.0/
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.openarchives.org/OAI/
        2.0/ http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
<ListRecords>
<record>
<header>
<identifier> 999910001</identifier>
</header>
<error>
<error_code>21</error_code>
<error_message>Error reading document</error_message>
</error>
</record> </ListRecords> </OAI-PMH>
```

Figure 4-4. Example error response for record ID that doesn't exist

web.xml

Values in `web.xml` are typically set one time during installation setup. Specfic to real-time availability, this file identifies the path for the the `PrimoExp.ini` file that is used in the process. You may want to check to confirm that it is currently accurate.

See Figure 4-5 for an example of `web.xml` content. Any reference to `yyydb` in the example indicates where you would specify your database name.

```
<servlet>
    <servlet-name>VoyagerAvailabilityService</servlet-name>
    <display-name>Voyager Availability Web Service</
     display-name>
    <servlet-
     class>com.endinfosys.voyager.websvc.VoyagerAvailabilit
     yService</servlet-class>
    <init-param>
        <param-name>PrimoExpIniFile</param-name>
        <param-value>
            /m1/voyager/yyydb/ini/PrimoExp.ini
        </param-value>
    </init-param>
</servlet>
```

Figure 4-5. web.xml example

PrimoExp.ini

The real-time availability process uses settings below the line ("EVERYTHING BELOW THIS LINE IS INTERNAL DO NOT MODIFY") in the `PrimoExp.ini` file to generate a successful response to Primo.

IMPORTANT:
Do not make changes to this section of the PrimoExp.ini.

The `UserPass=` option in the `[PrimoExp]` stanza must be specified for real-time availability to work. This cannot be commented out. See Table 2-5 on page 2-39 for a description of `UserPass=`.

Patron Services - OPAC via Primo

5

Patron Services - OPAC via Primo

5

Introduction

The following patron services are provided by Voyager Primo integration:

- Patron authentication with Patron Directory Services
- User account information
- Item display and request processing

These services are available with either OPAC via Primo or OPAC via Link. This chapter describes the OPAC via Primo implementation. For information regarding the legacy OPAC via Link implementation, see Patron Services - OPAC via Link on page 6-1.

Patron Directory Services (PDS)

Consistent with other Ex Libris products, Primo utilizes Patron Directory Services (PDS) to process patron authentication.

PDS does not have a patron database of its own. Instead, it provides the flexibility to pass authentication credentials to a designated target or redirect the user to an external authentication page.

Primo customers can use PDS to allow their users to authenticate against the Voyager database. When you use PDS to authenticate against the Voyager database, you get:

- Patron authentication against the Voyager database using credentials sent from PDS
- Borrower information sent from Voyager back to PDS
- PDS Single Sign-on (SSO) support for patrons using Primo, Voyager, and other Ex Libris products

Some sites may want to use something other than the Voyager database to authenticate their users. Common user authentication systems are LDAP or Shibboleth and are often provided by the campus bursar's office. For Voyager sites that do not provide Primo user authentication from the Voyager user database, Primo must still have information from Voyager about the user. This can be accomplished in two different ways.

First, the information that Primo must have about all Voyager users is:

- Voyager Patron ID (such as barcode or institution ID)
- Voyager Patron Group ID (such as faculty, staff, undergrad, and so on)
- UBID (the patron home DB)

Option 1: The easiest way to ensure that Primo has this Voyager information is to configure PDS so that bor-auth is authenticated against the campus information system (LDAP or other system) and bor-info is retrieved from Voyager. The above three values can be found in the `<bor-info>` tags of `<id>`, `<group_id>`, and `<ubid>`, respectively. See the *Patron Directory Services Guide* for more information on how to configure PDS.

Option 2: If you must use the campus information system for both bor-auth and bor-info, you must provide mappings from the campus information system values to Voyager values. For example, your campus information system may know that a patron is an undergrad, but that value must be mapped to the Voyager value of a patron group ID which is a number. To find the Voyager group ID numbers plus the patron ID and the UBID, refer to KB 16384-22994.

Configuration

The following server configuration components are needed to enable user authentication with PDS:

- The main PDS configuration table called tab_service.<institute> needs to be configured for communicating with the Voyager Web service application.

  More information about PDS configuration is available in the *Patron Directory Services Guide* available in the Ex Libris Documentation Center.

- The Voyager server XML needs to be configured with database connection information such as username, password, host, SID (system ID), and so forth. This can be set up once and is handled by the Ex Libris Voyager installation team.

See Appendix A for steps to verify your configuration information.

PDS Invoked

Primo invokes PDS when users attempt to log on.

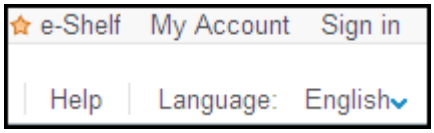

Figure 5-1. Sign-in link

Sign in using the link (Figure 5-1) provided.

Full PDS

For information about the full implementation of PDS with Single Sign On (SSO) capabilities, see the *WebVoyage Architecture Overview and Configuration Models* guide.

My Account

In the Voyager Primo environment, patrons are provided the following capabilities through the Primo My Account function without being redirected to a Voyager-provided screen:

- Display of patron information
- Option to renew charged items

- Option to cancel requests

This functionality is accessed through the My Account link in Primo.

Item Display and Request Information

With OPAC via Primo, the following capabilities are provided through the Primo interface without being redirected to a Voyager-provided interface:

- Display of holdings and items
- Placing requests (Primo-reserved OPAC request options)
  - Hold
  - Recall
  - Call Slip
  - UB

  (See Configuring OPAC Request Forms on page 5-6 for more information about the reserved OPAC request options provided by Primo.)

This is accomplished through Get Holdings and Get Requests in Primo with Voyager as the back office component.

For the other Voyager OPAC requests (not reserved in Primo) such as short loan, media booking, remote storage, ILL, blank form, and so on, users are able to make these requests from Primo through a Voyager screen embedded on the Primo Request tab (see Figure 5-9 on page 5-12). For set-up considerations, see Skins on page 5-12.

My Account Loans Paging in Primo

OvP – Loans Paging in Primo is a feature that can be used to improve performance between Voyager and Primo for the display of large numbers of loans. Review the loan limit settings you have made in Voyager System Administration in the matrices in Circulation and UB Policy Definitions (if applicable), and any global borrowing limits that have been set before customizing the OvP – Loans Paging bulk definition in Primo.

The OvP – Loans Paging bulk definition is an important setting for configuring connections with Voyager UB databases. Optimal performance needs to balance local loans (likely the most) with each remote database in the UB environment being queried. Primo retrieves and displays the number of loans set in OvP – Loans Paging from each database.

To illustrate this feature, refer to the following examples.

Example 1:

A patron has 25 loans at each of 3 databases and the OvP – Loans Paging (the setting to indicate how many loans from each institution to show/retrieve for a patron at a time) is equal to 5. In this example, Primo displays:

- Local database – Loan 5
- Remote database 1 – Loan 5
- Remote database 2 – Loan 5

The maximum number of loans that display to a patron in My Account in Primo is 15 (5 loans from each of the 3 databases).

If there are more loans to display, a Show More Loans link displays to page/request the next set of loans from each database.

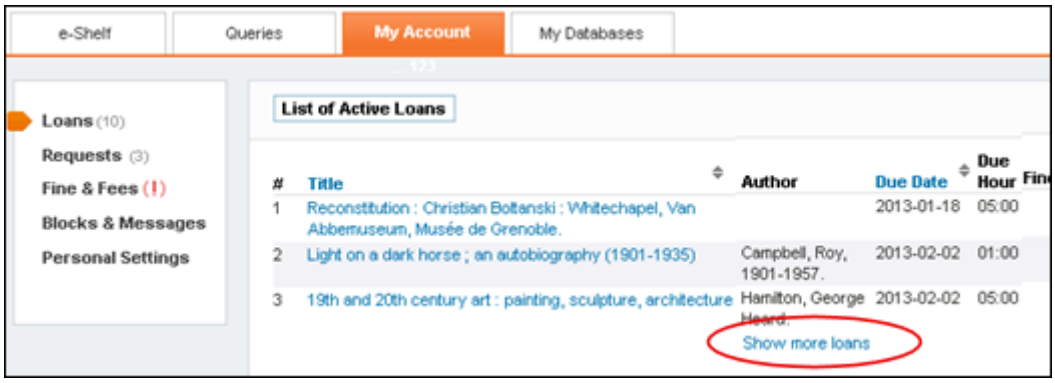

Figure 5-2. My Account Loans Paging Example

Example 2:

A patron has 25 loans at each of 3 databases and OvP – Loans Paging is equal to 20. In this example, Primo displays:

- Local database - Loans 1-20
- Remote database 1 - Loans 1-20
- Remote database 2 – Loans 1-20

A Show More Loans option displays to page/request the next set of loans from each database.

NOTE:
The default value for OvP – Loans Paging is 100. For UB environments, you may need to adjust this setting to take into account the multiple databases. One approach to consider is to try to set the OvP – Loans Paging to the probable number of loans possible for the patron group.

NOTE:
OvP – Loans Paging is set in the Bulk Definition mapping table in the front-end subsystem.

For additional information regarding configuring My Account loans, refer to the *Primo Interoperability Guide* (version 4.x or later).

Configuring OPAC Request Forms

For Voyager, the following default request forms are currently provided/reserved by Primo:

- Hold
- Recall
- Call Slip
- UB

These are defined in Primo as:

- `default.HoldRequest` with a description of Hold
- `default.Recall` with a description of Recall
- `default.CallSlip` with a description of Call Slip
- `default.UBRequest` with a description of UB

For other Voyager request forms, see Additional Request Forms You Define on page 5-6.

Additional Request Forms You Define

Additional Voyager request forms may be defined in the Primo Code Tables such as item level request, bib level request, course reserve list, ILL, and so on.

When you define additional OPAC request forms that are not currently identified and reserved in the Primo Code Tables, the `<request code>` in `<view code>.<request code>` needs to match the Codes name in the Voyager System Administration OPAC Configurations - Request Forms definitions (see Figure 5-3).

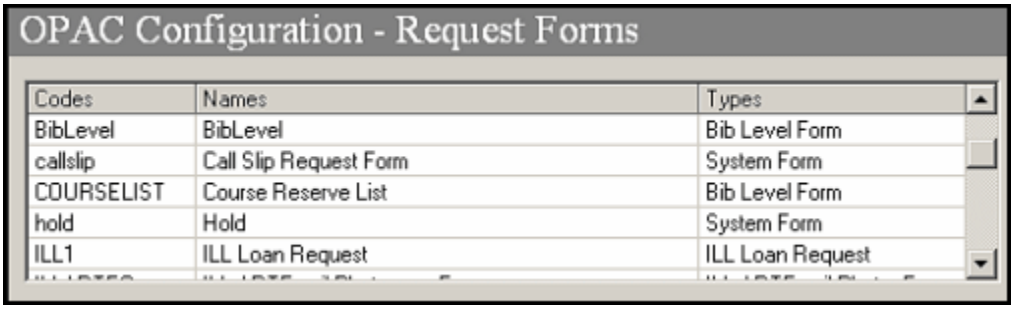

Figure 5-3. Voyager System Administration OPAC Configuration - Request Forms

Voyager transfers the System Administration Codes to Primo in all uppercase. For example, nobibdat is sent as `NOBIBDAT` (for the Blank Form). Since the Primo Code Table entries are case sensitive, all definitions for Voyager request forms that you add in Primo need to use the uppercase version of the Codes name from Voyager System Administration.

IMPORTANT:
Primo has reserved default.ILL in the Code Tables for Aleph interlibrary loan OPAC requests. As a result, you may need to change your Codes definition for interlibrary loan requests in Voyager System Administration if it is in any combination of ILL, ill, or Ill such that when it is sent in uppercase to Primo it results in ILL. You may, for example, change it to ILL1 and define it in the Primo Code Tables as <view code>.ILL1 with a description of ILL.

If you do not define the other OPAC request forms in the Primo Code Tables, they display in their uppercase form as sent from Voyager. See Figure 5-4 for an example of OPAC request form options that display in all uppercase as sent from Voyager.

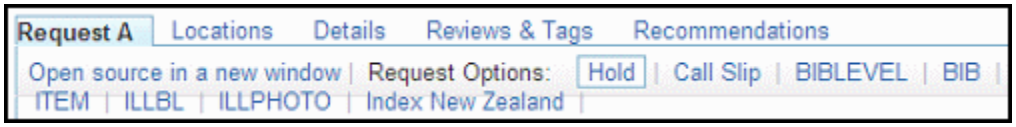

Figure 5-4. Request tab OPAC request options example

For more information regarding OPAC request form setup, refer to the *Primo Interoperability Guide*.

Configuring OPAC Request Forms in Primo Code Tables

Primo defines default request forms in the Code Tables for Aleph and Voyager that are unique and reserved for that purpose. See Figure 5-5.

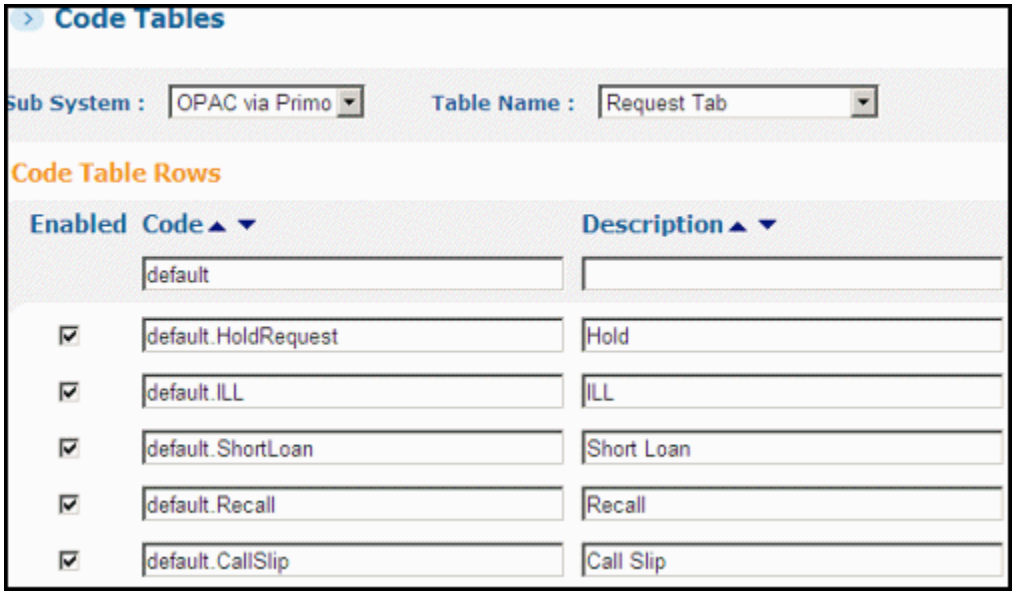

Figure 5-5. Primo Code Tables

The request form definition contains two key components:

- Code definition in the following format:

  `default.<request code>` or `<view code>.<request code>`

- Description definition that is viewable to users on the Request Tab such as Hold or Call Slip (see Figure 5-4)

NOTE:
Ex Libris development uses `default` for the `<view code>` name for reserved Primo Code Table definitions.

Configuration in Voyager for OPAC via Primo

Configuration for OPAC via Primo is done primarily in the `vxws.properties` file that is located in `/m1/voyager/xxxdb/tomcat/vxws/context/vxws/ui/eng` where `xxxdb` is your database name and `eng` is the three-letter ISO code for a language.

The `vxws.properties` file is used to define the following:

- Status text to be displayed
- Order information field (960)
- Skins
- Demerit status text and label
- Voyager databases for Universal Borrowing

Status Information

The `vxws.properties` file is used to define status information such as patron status, request status, and so on that is displayed in Primo for Voyager-originating materials. It is located in `/m1/voyager/xxxdb/tomcat/vxws/context/vxws/ui/`. The default directory is the `eng` (English) language directory as in `/m1/voyager/xxxdb/tomcat/vxws/context/vxws/ui/eng/`.

The `vxws.properties` file is built at installation by the Ex Libris installation team from the status information in your `webvoyage.properties` file.

Subsequent to the initial installation, you may customize `vxws.properties` as you prefer to display different status texts.

See Figure 5-6 for an example of status information that may be specified.

```
page.patronRequests.message.requestblocked-1=No items are available for Hold
        requests.
page.patronRequests.message.requestblocked-2=No items are available for CallSlip
        requests.
page.patronRequests.message.requestblocked-4=There are no circulating items
        available to satisfy your UB Borrowing Request.
page.patronRequests.message.requestblocked1=No holdings are available.
page.patronRequests.message.requestblocked10=No hold policy is defined for this
        item.
page.patronRequests.message.requestblocked11=This item has been scheduled
        through media booking.
page.patronRequests.message.requestblocked12=This item is not available for
        hold.
page.patronRequests.message.requestblocked2=The item is on order.
page.patronRequests.message.requestblocked3=The item is not charged.
page.patronRequests.message.requestblocked4=The item is missing.
page.patronRequests.message.requestblocked5=The item is lost.
```

Figure 5-6. vxws.properties file example

orderInfoHolTag=960

The `vxws.properties` file defines the field to be used to transport/dynamically display order information for an item in OPAC via Primo. The default for `orderInfoHolTag=` is the 960 tag. If 960 is already in use, you may specify another tag number.

IMPORTANT:
The tag number used for orderInfoHolTag= may not duplicate a tag number specified for the enrichment tags (see [Enrichment Tags] on page 2-36).

The `vxws.properties` file is located in `/m1/voyager/xxxdb/tomcat/vxws/context/vxws/ui/`. The default directory is the `eng` (English) language directory as in `/m1/voyager/xxxdb/tomcat/vxws/context/vxws/ui/eng/`.

page.myAccount.chargedItem.dueDate.indef= Parameter

The `page.myAccount.chargedItem.dueDate.indef=` parameter in the `vxws.properties` file is provided for customzing the display of indefinite loan information on the patron's My Account page in Primo.

Figure 5-7. Indefinite Loan Message on the Patron's My Account Page

In order for the indefinite loan message to display properly in the Due Date column on the patron's My Account page in Primo, the value specified for `page.myAccount.chargedItem.dueDate.indef=` must be a single term followed by a space.

```
################################################################################
# Item due date substitution tokens to display the codes' corresponding
# information in the Due Date column on the Charged Item section of My Account
         page
# \\d -> the item's due date
# \\e -> the items's due date, if the due date is not today
# \\t -> the item's due time
# \\u -> the item's due time, if the due date is today
################################################################################
page.myAccount.chargedItem.dueDate=\\d \\t
page.myAccount.chargedItem.dueDate.indef=Indefinite

date.format=yyyy-MM-dd

time.format=HH:mm
```

Figure 5-8. page.myAccount.chargedItem.dueDate.indef= Parameter

Skins

For request types that are not provided through OPAC via Primo such as ILL, ShortLoan, or the Blank request form, Primo displays a link to users that, when selected, embeds the WebVoyáge request screen into the Primo interface. Patrons are prompted to log on to WebVoyáge from the Primo interface (see Figure 5-9) to make these requests in the native Voyager interface.

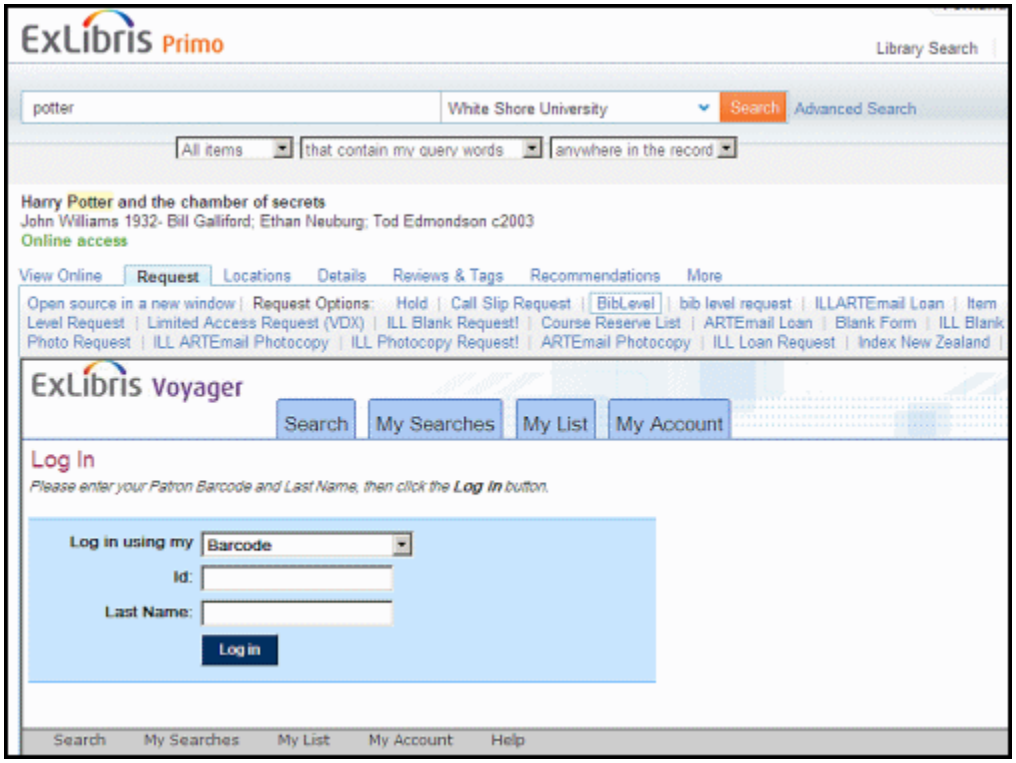

Figure 5-9. Prompted WebVoyáge Log In

You may specify the skin to be used for that link by identifying the skin name in the variable, `vwebv.skin`, in the `vxws.properties` file. You need to specify which VWEBV host IP and port is to be used for the link.

See Figure 5-10 for an example.

```
vwebv.host=10.100.2.36
vwebv.port=14004
vwebv.skin=en_US
```

Figure 5-10. Skin definition example in vxws.properties

Demerits

There is a customized label used for the demerit output in response to the retrieval of a patron's fines and demerits data. See the *Demerits: 10.00* example in Figure 5-11.

Use the `demerit.label=` variable in the `vxws.properties` file to define the label that you prefer. `Demerits:` is the default.

Figure 5-11. Demerits example

Universal Borrowing (UB)

For consortiums using Universal Borrowing, Voyager database codes need to be identified in the `vxws.properties` file. A translation of database names to Voyager database codes needs to be specified using the `connectDb.yyydb=` variable. This needs to be specified for every Voyager database. See Figure 5-12.

```
connectDb.yyydb=LOCAL
connectDb.aaadb=AREACOLL
connectDb.bbbdb=BIXUNIV
connectdb.cccdb=COMMCOLL
```

Figure 5-12. Example of database codes defined in vxws.properties

Using `connectDb.yyydb=LOCAL` as an example, the `yyydb` represents the name given to your Voyager database and is part of the directory structure when you navigate on the Unix server. This term is put into the `PrimoExp*.ini` files in the `[Institution Id]` stanza and is placed in the 949‡a of your Primo export. If you modified the code in that stanza, put that name here in place of the `yyydb`.

The variable, `LOCAL`, in the example, is the Code defined for the database in Voyager System Administration Search - Database Definitions. The database Code where the `vxws.properties` file is located always translates as `LOCAL`.

In Figure 5-12, there are four institutions in the example Universal Borrowing consortium. LOCAL is the home site for the VXWS service. The other three definitions represent the other Universal Borrowing sites/databases for your VXWS service as defined in `yyydb's` Voyager System Administration.

Patron Services - OPAC via Link

6

Patron Services - OPAC via Link

6

Introduction

The following patron services are provided by Voyager Primo integration:

- Patron authentication with Patron Directory Services
- User account information
- Item display and request processing

These services are available with either OPAC via Primo or OPAC via Link. This chapter describes the OPAC via Link implementation. For information regarding the OPAC via Primo implementation, see Patron Services - OPAC via Primo on page 5-1.

Patron Directory Services (PDS)

Consistent with other Ex Libris products, Primo utilizes Patron Directory Services (PDS) to process patron authentication.

PDS does not have a patron database of its own. Instead, it provides the flexibility to pass authentication credentials to a designated target or redirect the user to an external authentication page.

The Voyager Primo environment provides the following PDS authentication functions against the Voyager patron database.

- Patron authentication using credentials sent from PDS
- Borrower information sent to PDS
- PDS Single Sign On (SSO) support for Primo

Configuration

The following server configuration components are needed to enable user authentication with PDS:

- The main PDS configuration table called tab_service.<institute> needs to be configured for communicating with the Voyager web service application.

  More information about PDS configuration is available in the *Patron Directory Services* manual.

- The Voyager server XML needs to be configured with database connection information such as username, password, host, SID (system ID), and so forth. This can be set up once and is handled by the Ex Libris Voyager installation team.

PDS Invoked

Primo invokes PDS when users attempt to sign in. See Figure 6-1.

Figure 6-1. Primo Sign In / PDS Authentication

If a patron has not signed in prior to selecting **Get Avail Requests** (see Figure 6-2), clicking this option displays the Login dialog box. See Figure 6-3.

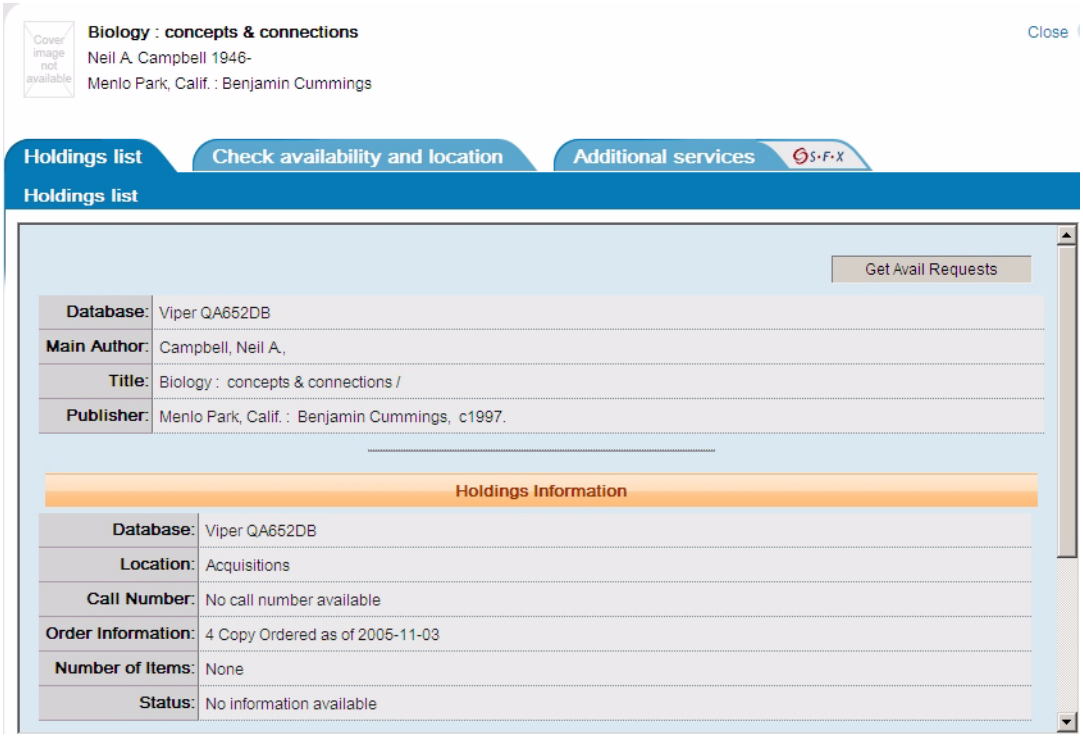

Figure 6-2. Get Avail Requests (prompting sign in)

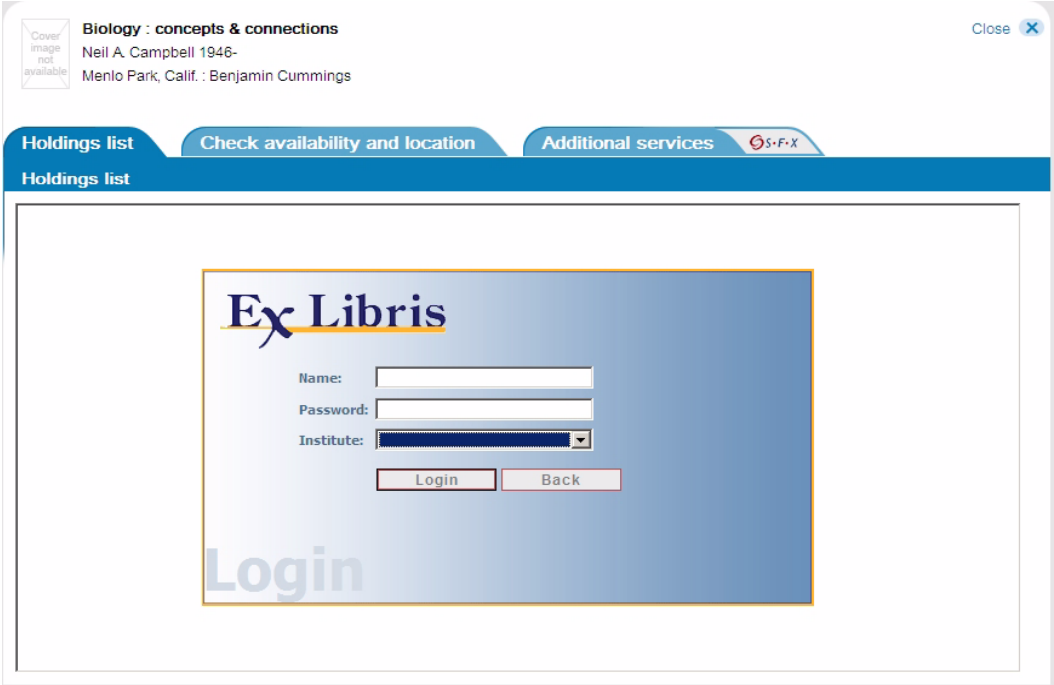

Figure 6-3. Login dialog box

Full PDS

For information about the full implementation of PDS with Single Sign On (SSO) capabilities, see the *WebVoyage Architecture Overview and Configuration Models* guide.

My Account

In the OPAC via Link environment, the Primo My Account function provides patrons with the following:

- Display of patron information
- Option to renew charged items
- Option to cancel requests

This functionality is accessed through the My Account link in Primo.

Item Display and Request Information

When the patron identifies a record of interest in Primo (from the Voyager database), clicking the GetIt! link displays the **Holdings List** tab in Primo.

The **Holdings List** tab display contains the Voyager bibliographic, holdings, and item information. There is also the **Select a Request** drop-down list (see Figure 6-4) providing the patron with a number of request options if the patron has signed in.

If the patron has not signed in, **Get Avail Requests** displays. See PDS Invoked on page 6-2 for the sign in and PDS authentication steps.

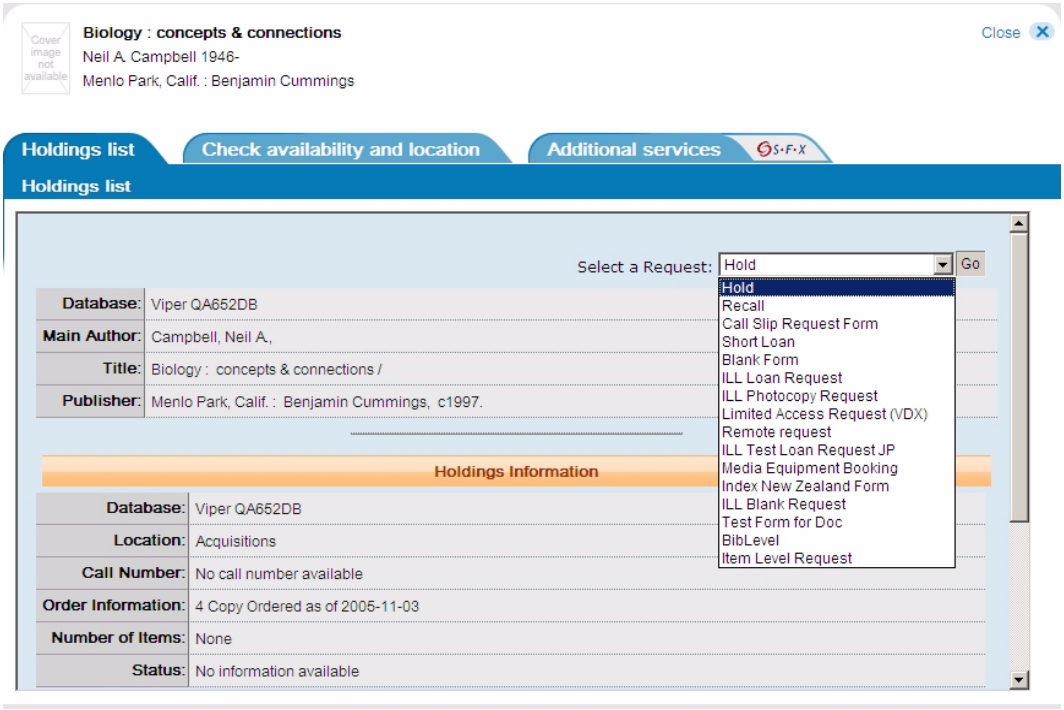

Figure 6-4. Select A Request options

The GetIt! link is configured by the Primo system administrator with the necessary linking information to the Voyager system. See the *Primo Back Office Guide* for details regarding how to configure GetIt!. This provides a deep link that enables the patron to confirm availability and process requests.

NOTE:
Requests in Voyager are only processed after PDS has successfully processed patron authentication.

Holdings Files

There are several files that may be tailored to customize the display of Voyager holdings in Primo. See Table 6-1 and Table 6-2 for a list, location, and description of these files.

Table 6-1. Files in /m1/voyager/yyydb/tomcat/vprimo/context/vprimo/xsl/html/default/

File Name	Description
`configs/104X_chronValues.xml`	This file is used to convert numeric values defined in MARC for month and season into chronological names. The element/attribute names in this file are arbitrary, but must match the XPath expression used to reference them from the translateChron template in holdingsDisplay.xsl. Additional languages may be added by creating new language elements and populating them with the appropriate chron elements: • lang id attribute = {language code from 008 field} • chron id attribute = {01-12 for months, 21-24 for seasons} • chron value = {chronological name to be displayed}
`configs/104X_config.xml`	This file is used to define serials format (NISO/Standard) punctuation, pairs of fields to display/not display, and captions.
`configs/104X_display.xsl`	This is the style sheet that pulls together compressed serials information from the 85X/86X fields.
`configs/displaycfg.xml`	This file is used to identify the display fields on a page.
`configs/displaycfg1.xml`	This file is used to identify the display fields on a page.

Table 6-1. Files in /m1/voyager/yyydb/tomcat/vprimo/context/vprimo/xsl/html/default/

File Name	Description
configs/displayHoldings.xml	This file is used to identify display fields such as database, location, call number, and order information for the holdings display record view.
configs/requestBlockCodes.xsl	This style sheet controls the display of a patron block message.
configs/requestBlockMsgs.xsl	This is the style sheet for request block code variables.
configs/shortLoanBlockCodes.xsl	This is the style sheet for request block code variables specific to short loans.
blocks.xsl	This style sheet controls building the block messages that display in MyAccount.
buildHtml.xsl	This style sheet is used to build the html page.
buildMarcDisplay.xsl	This style sheet controls the assembly of MARC records from Voyager (not Primo).
constants.xsl	This style sheet defines the location of style sheets, help, images, and Java script.
constantStrings.xsl	This style sheet manages constant text/labels for requests, copies, forms, UB requests, and so on.
currency.xsl	This style sheet handles the formatting of currency.
display.xsl	This is the style sheet that pulls everything together for display.
getAvailRequests.xsl	This style sheet controls getting available requests from the Voyager database.
marc21slim.xsl	This style sheet is used to build the MARC21 record.
mediaBooking.xsl	This style sheet controls the processing of the media booking request page.
messages.xsl	This style sheet defines a template called by multiple other files like sendRequest.
myAccount.xsl	This style sheet controls the top-level display of My Account page. This is linked to other my*.

Table 6-1. Files in /m1/voyager/yyydb/tomcat/vprimo/context/vprimo/xsl/html/default/

File Name	Description
myAccountCancel.xsl	This style sheet controls the My Account page when there is a cancel request.
myAccountChargedItems.xsl	This style sheet controls the display of charged items.
myAccountConstants.xsl	This style sheet manages the constants used for My Account page.
myAccountLinks.xsl	This style sheet controls the links for My Account, search preferences, personal information, and so on.
myAccountRenew.xsl	This style sheet controls the My Account display for charged items and so on.
myAccountRequests.xsl	This style sheet controls the display of requests.
personalInfo.xsl	This style sheet controls the display of the personal information page.
requestForm.xsl	This style sheet controls the processing of the request form.
sendRequest.xsl	This style sheet controls request processing.
shortLoanReqForm.xsl	This style sheet is used to build the short loan request form.
show-detail.xsl	This is a JavaScript debugging tool. **IMPORTANT:** *Do not edit.*
statusStrings.xsl	This style sheet is used to build the item status string.
UBReqForm.xsl	This style sheet controls building the UB (Universal Borrowing) request page.

Table 6-2. Files in /m1/voyager/yyydb/tomcat/vxws/context/vxws/xsl/

File Name	Description
`configs/104X_chronValues.xml`	This file is used to convert numeric values defined in MARC for month and season into chronological names. The element/attribute names in this file are arbitrary, but must match the XPath expression used to reference them from the translateChron template in holdingsDisplay.xsl. Additional languages may be added by creating new language elements and populating them with the appropriate chron elements: • lang id attribute = {language code from 008 field} • chron id attribute = {01-12 for months, 21-24 for seasons} • chron value = {chronological name to be displayed}
`configs/104X_config.xml`	This file is used to define format (NISO/ Standard) punctuation, pairs of fields to display/not display, and captions.
`configs/104X_display.xsl`	This is the style sheet that pulls together punctuation, caption prefixes, indicator filters, and so on.
`configs/emailcfg.xml`	This file is used to identify the display fields for e-mail.
`configs/emailholdingscfg.xml`	This file is used to identify the holdings display fields for e-mail.
`configs/requestcfg.xml`	This provides the XML for all the display tags such as author, title, subjects, description, publisher and so on.
`configs/requestholdingscfg.xml`	This provides the XML for the holdings details such as call number and shelving title.
`biblevelrequest.xsl`	This style sheet controls bibliographic-level requests.
`constants.xsl`	This style sheet controls cascading style sheets, help files, images, and javascripts.

Table 6-2. Files in /m1/voyager/yyydb/tomcat/vxws/context/vxws/xsl/

File Name	Description
`constantstrings.xsl`	This style sheet manages constant text/labels for requests, copies, forms, UB requests, and so on.
`emailbibrecord.xsl`	This style sheet handles e-mailing bibliographic record requests.
`marc21slim.xsl`	This style sheet is used to build the MARC21 record.

Universal Borrowing with Primo via Link

With Primo versions 2.1.3 and higher and Voyager versions 7.1 and higher, support is provided for Universal Borrowing (UB) capability to patrons using Primo as their user interface for discovery. The Universal Borrowing function enables patrons from a library within a consortium to place direct requests on materials from other libraries within the consortium.

Primo provides patrons the following access points for placing UB requests from the discovery display:

- GetIt! link
- Item in the catalog link
- Available location link

Regardless of the access method used, Universal Borrowing policies, patron blocks, and delivery rules are enforced per the policies of the Voyager institution. Also options like Supply Default Pickup Location are also supported.

UB Process Overview

With the Universal Borrowing with Primo feature, Primo and Voyager work together to provide the patron access to Universal Borrowing resources when the home Voyager database does not have the requested holdings. See Figure 6-5 for a high-level illustration of the process.

Refer to the other sections in this chapter for a more-detail description of the setup required to implement this process.

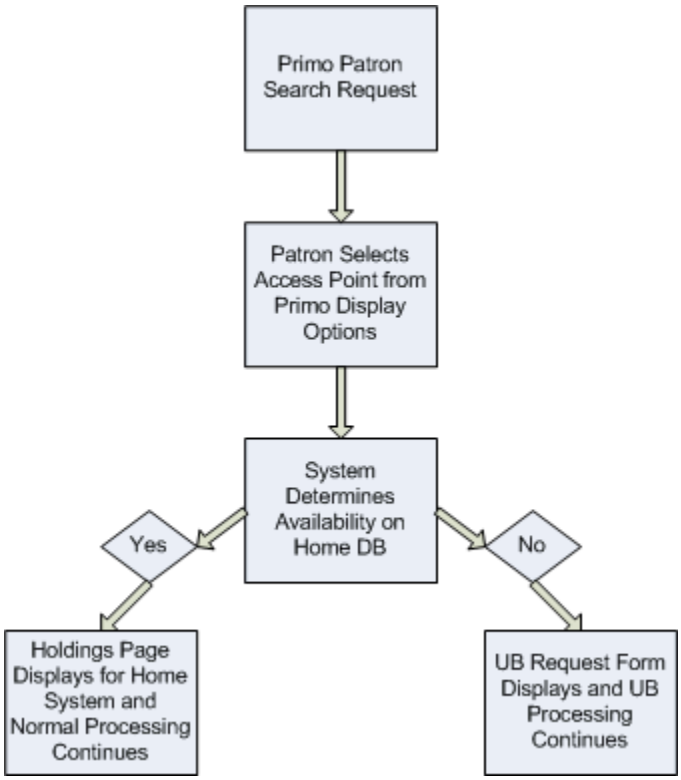

Figure 6-5. Voyager Primo UB process overview

Enabling Universal Borrowing with Primo via Link

The primary setup components for enabling Universal Borrowing with Primo are as follows:

- Configuration files on the Voyager server.

  Refer to the following sections for setup details:

  - vprimo.properties on page 6-12
  - requests.properties on page 6-17
  - messages.properties on page 6-17

- Additional Voyager database for mapping UB requests from Primo

  See Primo UB Connector (PUC) Database on page 6-19 for details regarding setup for this database.

- VPDS setup in Voyager System Administration (Search > Database Definitions)

  See the *Voyager System Administration User's Guide* for more information regarding database definitions setup.

For information regarding setup on the Primo server for configuration files and so on, refer to the *Primo Interoperability Guide*.

vprimo.properties

The `vprimo.properties` file located in `/m1/voyager/xxxdb/tomcat/vprimo/context/vprimo/ini` (where `xxxdb` is your database name) provides the following UB settings:

- `UBProcessing`
- `TurnOffLibraryDropDown`
- User-defined database key

See an example of this file in Figure 6-6.

The `vprimo.properties` file also contains the holdings status codes for the VPRIMO display status string (not to be modified).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties version="1.0">
    <comment>definitions for VPRIMO configuration</comment>
    <!-- set this option to Y to enable UB processing, if Voyager has UB      -->
    <entry key="UBProcessing">N</entry>
    <!-- Option to show only one randomly selected library for the item.      -->
    <!-- Default is N to show all libraries that would have the item.         -->
    <entry key="TurnOffLibraryDropDown">N</entry>
    <!--                                                                      -->
    <!-- User defined DB key to the DB code used to retrieve holdings records -->
    <!--                                                                      -->
    <entry key="connectDb.YYYDB">LOCAL</entry>
    <!-- additional example:
    <entry key="connectDb.SomeOtherLibrary">otherdb</entry>
    -->
```

Figure 6-6. Example of vprimo.properties configuration file

```
<!--                                                                          -->
<!-- Map "institute" values as returned by PDS to DB_CODE as found in the -->
<!-- VOYAGER_DATABASES table.                                                 -->
<!--                                                                          -->
<entry key="institute.YYYDB">LOCAL</entry>
<!-- additional example:
<entry key="institute.SomeOtherLibrary">otherdb</entry>
-->
<!--                                                                          -->
<!-- Holdings page message                                                    -->
<entry key="holdings.message.noAvailableItems">There are no circulating items available
       to satisfy your Borrowing Request</entry>
<!--                                                                          -->
<!-- Holdings status code to VPRIMO display status string. Please do not  -->
<!-- modify because the code is defined in the database.                      -->
<!--                                                                          -->
<entry key="holdings.item.status.1">Not Charged</entry>
<entry key="holdings.item.status.2">Charged</entry>
<entry key="holdings.item.status.3">Renewed</entry>
<entry key="holdings.item.status.4">Overdue</entry>
<entry key="holdings.item.status.5">Recall Request</entry>
<entry key="holdings.item.status.6">Hold Request</entry>
<entry key="holdings.item.status.7">On Hold</entry>
<entry key="holdings.item.status.8">In Transit</entry>
<entry key="holdings.item.status.9">In Transit Discharged</entry>
<entry key="holdings.item.status.10">In Transit On Hold</entry>
<entry key="holdings.item.status.11">Discharged</entry>
<entry key="holdings.item.status.12">Missing</entry>
<entry key="holdings.item.status.13">Lost--Library Applied</entry>
<entry key="holdings.item.status.14">Lost--System Applied</entry>
<entry key="holdings.item.status.15">Claims Returned</entry>
<entry key="holdings.item.status.16">Damaged</entry>
<entry key="holdings.item.status.17">Withdrawn</entry>
<entry key="holdings.item.status.18">At Bindery</entry>
<entry key="holdings.item.status.19">Cataloging Review</entry>
<entry key="holdings.item.status.20">Circulation Review</entry>
<entry key="holdings.item.status.21">Scheduled</entry>
```

Figure 6-6. Example of vprimo.properties configuration file (Continued)

```
    <entry key="holdings.item.status.22">In Process</entry>
    <entry key="holdings.item.status.23">Call Slip Request</entry>
    <entry key="holdings.item.status.24">Short Loan Request</entry>
    <entry key="holdings.item.status.25">Remote Storage Request</entry>
</properties>
```

Figure 6-6. Example of vprimo.properties configuration file (Continued)

UBProcessing

Use the `UBProcessing` parameter to turn on the Universal Borrowing with Primo feature on the Voyager server. Set this parameter to `Y` (Yes) to turn it on. The default setting is `N` (No).

TurnOffLibraryDropDown

By default, the system displays all libraries in the consortium that have the item. This is a function of the `TurnOffLibraryDropDown` parameter that has a default setting of `N` (No).

To have the system show only one randomly-selected library for the item, set the `TurnOffLibraryDropDown` parameter to `Y` (Yes).

User-Defined Database Key

The databases (UB libraries) participating in Universal Borrowing with Primo need to be identified in the `vprimo.properties` configuration file. See Figure 6-7 for an example of the format used to define these databases.

```
    <!-- User defined DB key to the DB code used to retrieve holdings records -->
    <!--                                                                      -->
    <entry key="connectDb.YYYdb">LOCAL</entry>
    <entry key="connectDb.qadma710db">DMA71DB</entry>
    <entry key="connectDb.qagwcc710db">GWCC71DB</entry>
    <entry key="connectDb.qarscc710db">RSCC71DB</entry>
```

Figure 6-7. Example connectDb database definitions

Using `<entry key=”connectDb.gadma710db”>DMA71DB</entry>` from the Figure 6-7 example, `qadma710db` represents the database name.

DMA71DB represents the database code as defined in Voyager System Administration. See Figure 6-8 and Figure 6-9.

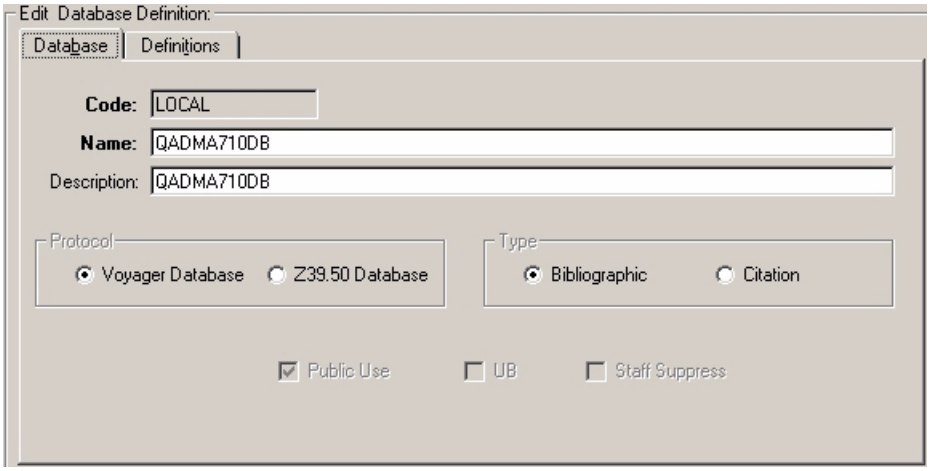

Figure 6-8. Voyager System Administration DB definition - Database tab

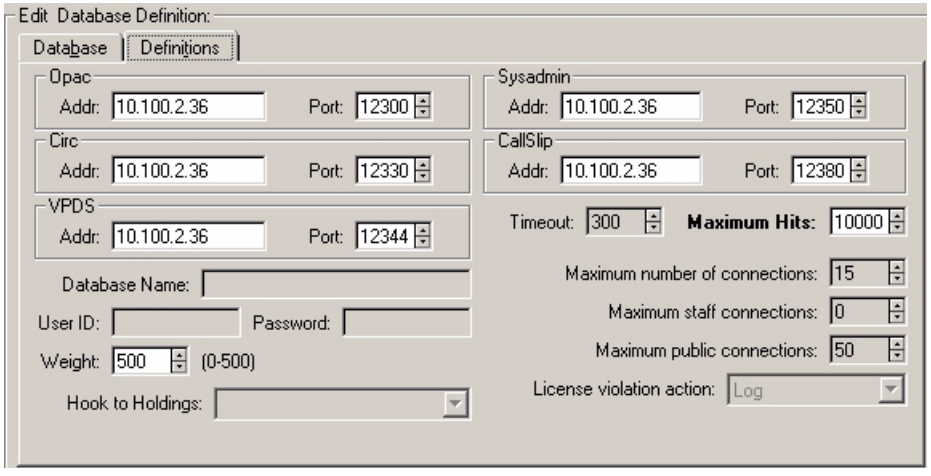

Figure 6-9. Voyager System Administration DB definition - Definitions tab

There needs to be one entry for each UB database participating in Universal Borrowing with Primo via Link.

If `qadma710db` is defined with the LOCAL database code in Voyager System Administration, the entry would be as follows:

```
<entry key=”connectDb.gadma710db”>LOCAL</entry>
```

NOTE:
There is no LOCAL entry in the PUC `vprimo.properties` configuration file. See Primo UB Connector (PUC) Database on page 6-19 for more information.

Institute Mapping

The institute mapping in the `vprimo.properties` configuration file is used for patron authentication.

Each UB database participating in Universal Borrowing with Primo needs to have an entry in the `vprimo.properties` configuration file where the Primo PDS (Patron Directory Services) code is mapped to the Voyager System Administration's database code. See Figure 6-10 for an example.

NOTE:
The institute definitions are case sensitive and need to be all uppercase.

The institute referenced in this section is the same as the institute identified in the [Institute Id] stanza discussed in [Institution Id] on page 2-35. See Figure 2-1 on page 2-6 for an example of this stanza. The institution ID is stored in Primo in the 949‡a of the Primo PNX.

```
<!-- Map "institute" values as returned by PDS to DB_CODE as found in the -->
<!-- VOYAGER_DATABASES table.                                              -->
<!--                                                                       -->
<entry key="institute.YYYDB">LOCAL</entry>
<entry key="institute.QADMA710DB">DMA71DB</entry>
<entry key="institute.QAGWCC710DB">GWCC71DB</entry>
<entry key="institute.QARSCC710DB">RSCC71DB</entry>
```

Figure 6-10. Example institute definitions

NOTE:
There is no LOCAL entry in the PUC `vprimo.properties` configuration file for institute definitions. See Primo UB Connector (PUC) Database on page 6-19 for more information.

requests.properties

The `requests.properties` file located in `/m1/voyager/xxxdb/tomcat/vxws/ini` (where `xxxdb` is your database name) provides the following UB settings:

- `SupplyDefaultPickupLocs`
- `SelectPickupLocsMsg`

See an example of this file in Figure 6-11.

```
<!-- the next 2 properties are for universal borrowing requests -->
<entry key="SupplyDefaultPickupLocs">Y</entry>
<entry key="SelectPickupLocsMsg">You must provide a Pick Up Library and Desk before your
          request is complete!</entry>
```

Figure 6-11. Example of UB parameters in request.properties

SupplyDefaultPickupLocs

Use the `SupplyDefaultPickupLocs` to indicate that you want a default pickup location to be displayed for the patron. The default is `Y` (Yes) to turn on this option. Specify `N` (No) to turn it off.

When this key is set to `Y`, the default pickup location is set to the first pickup location in the drop-down list.

When this key is set to `N`, the pickup locations are listed alphabetically in the drop-down list and no default is selected.

SelectPickupLocsMsg

Use the `SelectPickupLocsMsg` key to define the message that displays when no pickup location has been specified by the patron. The message shown in Figure 6-11 is the default that is provided.

messages.properties

The `messages.properties` file located in `/m1/voyager/xxxdb/tomcat/vprimo/context/vprimo/ini` (where `xxxdb` is your database name) provides the various block or error messages that may be returned by the Voyager Primo web services. See Figure 6-12 for an example of stored messages.

These messages are preset and do not require any additional customization. Use the `messages.properties` file as your reference for patron blocked messages.

```
<properties version="1.0">

    <comment>This properties file defines messages to be displayed for various
block or error messages that might be returned by the Voyager-Primo web
services.</comment>

    <!-- patron blocked message -->

    <entry key="Requests.patronblocked">You have no requests available in this
database.</entry>

    <!-- generic request blocked messages -->

    <entry key="Requests.requestblocked-4">There are no circulating items
available to satisfy your UB Borrowing Request.</entry>

    <entry key="Requests.requestblocked-3">No items are available for Remote
Storage requests.</entry>

    <entry key="Requests.requestblocked-2">No items are available for CallSlip
requests.</entry>

    <entry key="Requests.requestblocked-1">No items are available for Hold
requests.</entry>

    <entry key="Requests.requestblocked1">No holdings are available.</entry>

    <entry key="Requests.requestblocked2">The item is on order.</entry>

    <entry key="Requests.requestblocked3">The item is not charged.</entry>

    <entry key="Requests.requestblocked4">The item is missing.</entry>

    <entry key="Requests.requestblocked5">The item is lost.</entry>

    <entry key="Requests.requestblocked6">The item is at the bindery.</entry>

    <entry key="Requests.requestblocked7">No items are available for recall.</
entry>

    <entry key="Requests.requestblocked8">You have already placed a request for
this item.</entry>

    <entry key="Requests.requestblocked9">No recall policy is defined for this
item.</entry>

    <entry key="Requests.requestblocked10">No hold policy is defined for this
item.</entry>

    <entry key="Requests.requestblocked11">This item has been scheduled through
media booking.</entry>

    <entry key="Requests.requestblocked12">This item is not available for
hold.</entry>
```

Figure 6-12. Example from message.properties file

Primo UB Connector (PUC) Database

Specific to Universal Borrowing with Primo is the addition of the PUC (Primo UB Connector) database. For each Universal Borrowing consortium, an additional Voyager database (PUC) is created for the purpose of routing Universal Borrowing requests. Key to this capability are the `connectDb` and `institiute` definitions in the `vprimo.properties` file for the PUC system.

Specifically, the PUC `vprimo.properties` file needs to define all the UB databases in the consortium and omit any reference to itself as the LOCAL database. See Figure 6-13 for an example.

```
<!-- User defined DB key to the DB code used to retrieve holdings records -->
    <!--                                                                    -->
    <entry key="connectDb.qadma710db">DMA71DB</entry>
    <entry key="connectDb.qagwcc710db">GWCC71DB</entry>
    <entry key="connectDb.qarscc710db">RSCC71DB</entry>
<!--                                                                    -->
<!-- Map "institute" values as returned by PDS to DB_CODE as found in the -->
<!-- VOYAGER_DATABASES table.                                           -->
    <!--                                                                    -->
    <entry key="institute.QADMA710DB">DMA71DB</entry>
    <entry key="institute.QAGWCC710DB">GWCC71DB</entry>
    <entry key="institute.QARSCC710DB">RSCC71DB</entry>
```

Figure 6-13. Example PUC vprimo.properties connectDB and institute definitions

Non-Universal Borrowing Database Installations

For non-Universal Borrowing database installations using Primo, the following parameters still need to be set in the `vprimo.properties` file:

- `<entry key="UBProcessing">N</entry>`

  See UBProcessing on page 6-14 for more information.

- `<entry key="TurnOffLibraryDropDown">N</entry>`

  See TurnOffLibraryDropDown on page 6-14 for more information.

- `<entry key="connectDb.YYYDB">LOCAL</entry>`

  See User-Defined Database Key on page 6-14 for more information.

- `<entry key="institute.YYYDB">LOCAL</entry>`

  See Institute Mapping on page 6-16 for more information.

See Figure 6-14 where these parameters are highlighted.

These values need to be set in the `vprimo.properties` file to enable Patron Directory Services (PDS) to function properly.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties version="1.0">
    <comment>definitions for VPRIMO configuration</comment>
    <!-- set this option to Y to enable UB processing, if Voyager has UB      -->
    <entry key="UBProcessing">N</entry>
    <!-- Option to show only one randomly selected library for the item.      -->
    <!-- Default is N to show all libraries that would have the item.         -->
    <entry key="TurnOffLibraryDropDown">N</entry>
    <!--                                                                      -->
    <!-- User defined DB key to the DB code used to retrieve holdings records -->
    <!--                                                                      -->
    <entry key="connectDb.YYYDB">LOCAL</entry>
    <!-- additional example:
    <entry key="connectDb.SomeOtherLibrary">otherdb</entry>
    -->
    <!--                                                                      -->
    <!-- Map "institute" values as returned by PDS to DB_CODE as found in the -->
    <!-- VOYAGER_DATABASES table.                                             -->
    <!--                                                                      -->
    <entry key="institute.YYYDB">LOCAL</entry>
    <!-- additional example:
    <entry key="institute.SomeOtherLibrary">otherdb</entry>
```

Figure 6-14. Example of vprimo.properties file for non-UB installation

Apache mod_deflate Module

7

Apache mod_deflate Module

7

Introduction

The purpose of the Apache `mod_deflate` module is to compress the Web content before sending it over the network to the browser clients. The compression is implemented by the deflate filter that runs inside the Apache server with `mod_deflate` enabled and the proper configuration.

If your Primo installation has `mod_deflate` enabled, then you also need to enable `mod_deflate` in your Voyager environment using the instructions provided in this chapter.

For the Solaris, AIX, and Linux operating system environments, the `mod_deflate` module has been statically built in the Apache 2.2.6 version. For Windows, the Apache 2.2.6 software has the `mod_deflate` module built in dynamically as part of the standard distribution.

Prerequisites

The following are the prerequisites for the Voyager `mod_deflate` module implementation:

- `mod_module` (on your Apache server) or the Apache server 2.2.6 version that has `mod_deflate` compiled

- Ex Libris customized version of `mod_deflate` configuration installed on your server
- Updated `mod_deflate` configuration that is properly enabled

Implementing the mod_deflate Module

This section provides the instructions for implementing the `mod_deflate` module in the following operating system environments:

- SunOS, AIX, and Linux
- Windows

The implementation procedure has the following parts:

- Confirm that the mod_deflate module is installed on your server (if yes, no further action is required)
- Download the Apache server 2.26 version with the mod_deflate configuration
- Configure and enable the mod_deflate module on the Apache server

SunOS, AIX, and Linux Environments

The following procedures are for the SunOS, AIX, and Linux operating system environments:

Procedure 7-1. Part 1: Confirming that the mod_deflate Module is Installed

Use the following procedure to confirm that the `mod_deflate` module is installed on your server:

1. Login to the server as the root user.

   ```
   su - root
   ```

2. Change the directory to the `bin` directory.

   ```
   cd /m1/shared/apache2/bin
   ```

3. Check the version.

   ```
   ./httpd -version
   ```

4. Check the list of built-in modules.

   ```
   ./httpd -l
   ```

5. Verify that the standard output displays compiled in modules as show in Figure 7-1.

```
core.c
mod_deflate.c
prefork.c
http_core.c
mod_so.c
```

Figure 7-1. Standard Output Compiled in Modules

   If the Apache version is earlier than 2.2.6 or `mod_deflate.c` is not listed in the precompiled modules, an upgrade to the Apache server 2.2.6 version is required. Contact Ex Libris Customer Support if this is your situation.

6. Compare the content of your `mod_deflate.conf` configuration file (current and updated) with the content of Figure 7-3.

   ```
   cd /m1/shared/apache2/conf/modules.conf

   ls mod_deflate.conf

   more mod_deflate.conf
   ```

   If your `mod_deflate.conf` file content matches the content in Figure 7-3, you have the correct `mod_deflate` module for compression implemented and no further action is required. Otherwise, proceed to Procedure 7-2, Part 2: Downloading the Apache Server 2.2.6 Version with the mod_deflate Configuration.

Procedure 7-2. Part 2: Downloading the Apache Server 2.2.6 Version with the mod_deflate Configuration

The standard Apache server 2.2.6 version is contained in the shared package distributed for Voyager customers. You can download the shared package from the Ex Libris FTP server using the voyager user login. Use the following procedure to download this material.

NOTE:
Assume that `/m1/incoming` is the local directory.

1. Change the directory to the `incoming` directory.

   ```
   cd /m1/incoming
   ```

2. Begin the FTP.

   ```
   ftp downloads.exlibrisgroup.com
   ```

3. Log in as `voyager`.

4. Change the directory to the `2007.2.0` directory.

   ```
   cd <OS>/2007.2.0
   ```

   (where OS=SunOS, AIX, or Linux)

5. Specify binary mode.

   ```
   bin
   ```

6. To monitor the progress of the transfer, specify:

   ```
   hash
   ```

7. Get the shared files.

   ```
   get 2007.2.0.SHARED.tar.bz2
   ```

8. Exit.

   ```
   bye
   ```

Procedure 7-3. Part 3: Configuring/Enabling the mod_deflate Module

To implement the `mod_deflate` module, you need to stop your Apache server, copy the customized `mod_deflate.conf` to the appropriate directory, and restart the Apache server. Use the following steps:

1. Uncompress the downloaded shared package.

   ```
   cd /m1/incoming

   bzip2 -dc 2007.2.0.SHARED.tar.bz2 | tar xvf -
   ```

2. Stop the running Apache server.

   ```
   su - root

   /m1/shared/apache2/bin/apachectl stop
   ```

3. Copy mod_deflate to the appropriate directory.

   ```
   cp /m1/incoming/SHARED/httpd/2.2.6/conf/modules.conf/
   mod_deflate.conf /m1/shared/apache2/conf/modules.conf/.
   ```

4. Restart the Apache server.

   ```
   /m1/shared/apache2/bin/apachectl start
   ```

5. Verify that the log file, deflate_log, has been generated in the Apache logs directory.

   ```
   ls -l /m1/shared/apache2/logs/deflate_log
   ```

Windows Environment

The following procedures are for the Windows operating system environment:

Procedure 7-4. Part 1: Confirming that the mod_deflate Module is Installed

Use the following procedure to confirm that the `mod_deflate` module is installed on your server:

1. Open a Korn Shell session inside the Windows server.
2. Change the directory to the `bin` directory.

   ```
   cd D:/m1/shared/Apache2/bin
   ```

3. Check the version.

   ```
   ./httpd.exe -version
   ```

4. Check the list of built-in modules.

   ```
   cd D:/m1/shared/Apache2/modules

   ls mod_deflate.so
   ```

   If the Apache server version is earlier than 2.2.6 or `mod_deflate.so` is not listed, contact Ex Libris Customer Support.

5. Verify if the existing Apache `httpd.conf` contains the `mod_deflate` configuration.

   Search for `<IfModule !mod_deflate.c>`

6. If you find the string from step 5, compare the content of your `httpd.conf` configuration file with the content of Figure 7-4.

   If your `httpd.conf` file content matches the content in Figure 7-4, you have the correct `mod_deflate` module for compression implemented and no further action is required. Otherwise, proceed to Procedure 7-5, Part 2: Downloading the Script Package with the mod_deflate Configuration.

Procedure 7-5. Part 2: Downloading the Script Package with the mod_deflate Configuration

The customized `mod_deflate.conf` file is contained in a package called `scripts.zip`. You can download this package from the Ex Libris FTP server using `voyager` as the login user. Use the following procedure to download this material:

NOTE:
Assume that `D:/m1/incoming` is the local directory.

1. Change the directory to the `incoming` directory.

   ```
   cd D:/m1/incoming
   ```

2. Begin the FTP.

   ```
   ftp downloads.exlibrisgroup.com
   ```

3. Log in as `voyager`.

4. Specify binary mode.

   ```
   bin
   ```

5. To monitor the progress of the transfer, specify:

   ```
   hash
   ```

6. Change the directory to the `2007.2.0` directory.

   ```
   cd WIN/2007.2.0
   ```

7. Get the shared files.

   ```
   get scripts.zip
   ```

8. Exit.

   ```
   bye
   ```

Procedure 7-6. Part 3: Configuring/Enabling the mod_deflate Module

To implement the `mod_deflate` module, you need to stop your Apache server, incorporate the `mod_deflate.conf` content into the existing Apache `httpd.conf`, and restart the Apache server. Use the following steps:

1. Uncompress the downloaded scripts package.

   ```
   cd D:/m1/incoming

   unzip scripts.zip
   ```

2. Stop the running Apache server using the installed script by Voyager (or from the Windows service).

   ```
   C:/etc/inet.d/HTTPD.KSH stop
   ```

3. Archive the existing Apache httpd.conf file

   ```
   cd D:/m1/shared/Apache2/conf

   cp httpd.conf httpd.conf.pre-<YYYYMMDD>
   ```

4. Append the `mod_deflate.conf` file to the existing `httpd.conf` file.

   NOTE:
   Use >> for appending, not for overwriting.

   ```
   cat D:/m1/incoming/scripts/files/apache/mod_deflate.conf \
   >> D:/m1/incoming/scripts/files/apache/httpd.conf
   ```

5. Edit the `httpd.conf` file and move the lines in Figure 7-2 to the bottom of the `httpd.conf` file so that the `mod_deflate` module is loaded properly. You can locate the lines quickly by searching for `db.httpd.conf` in the file.

   ```
   cd D:/m1/shared/Apache2/conf /

   vi httpd.conf
   ```

```
#Voyager DBs
Include D:\m1\shared\Apache2\conf\*db.httpd.conf
```

Figure 7-2. Lines to Move

6. Restart the Apache server using the installed script by Voyager (or from the Windows service).

   ```
   C:/etc/inet.d/HTTPD.KSH start
   ```

7. Verify that the log file, `deflate_log`, is generated properly and in the Apache logs directory.

   ```
   ls -l D:/m1/shared/apache2/logs/deflate_log
   ```

mod_deflate.conf Configuration Content

See Figure 7-3 for the `mod_deflate.conf` file's configuration content for the SunOS, AIX, and Linux operating system environments.

```
# Load deflate_module
<IfModule !mod_deflate.c>
LoadModule deflate_module modules/mod_deflate.so
</IfModule>

<IfModule mod_deflate.c>

# Insert filter for output compression
SetOutputFilter DEFLATE

# Insert filter for input decompression
SetInputFilter DEFLATE

# Netscape 4.x has some problems...
BrowserMatch ^Mozilla/4 gzip-only-text/html

# Netscape 4.06-4.08 have some more problems
BrowserMatch ^Mozilla/4\.0[678] no-gzip

# MSIE masquerades as Netscape, but it is fine
# BrowserMatch \bMSIE !no-gzip !gzip-only-text/html

# NOTE: Due to a bug in mod_setenvif up to Apache 2.0.48
# the above regex won't work. You can use the following workaround
BrowserMatch \bMSI[E] !no-gzip !gzip-only-text/html

# Don't compress images
SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png)$ no-gzip dont-vary
```

Figure 7-3. mod_deflate.conf for SunOS/AIX/Linux

```
# Don't compress the zipped files
SetEnvIfNoCase Request_URI \.(?:exe|t?gz|zip|bz2|sit|rar)$ no-gzip
         dont-vary


# Don't compress the pdf files
SetEnvIfNoCase Request_URI \.pdf$ no-gzip dont-vary


# Make sure proxies don't deliver the wrong content
Header append Vary User-Agent env=!dont-vary


# DeflateFilterNote Directive & accurate logging
DeflateFilterNote Input input_info
DeflateFilterNote Output output_info
DeflateFilterNote Ratio ratio_info
LogFormat '"%r" %{output_info}n/%{input_info}n (%{ratio_info}n%%)'
         deflate
CustomLog /m1/shared/apache2/logs/deflate_log deflate


# If you decide to prevent compliant proxies from caching entirely,
         uncomment the line below
# Header set Vary *


# DeflateBufferSize Directive - Specifies the size in bytes of the
         fragments
# that zlib should compress at one time. Default value is 8096
# DeflateBufferSize 8096


# DeflateCompressionLevel Directive - Specifies what level of
         compression should be used,
# the higher the value, the better the compression, but more CPU time
         is required.
# A value between 1-9.  Default is 6
# DeflateCompressionLevel 6


# DeflateMemLevel Directive - Specifies how much memory should be used
         by zlib for compression
# (a value between 1 and 9). Default is 9.
# DeflateMemLevel 9
```

Figure 7-3. mod_deflate.conf for SunOS/AIX/Linux (Continued)

```
# DeflateWindowSize Directive - Specifies the zlib compression window
          size
# (a value between 1 and 15). Default is 15.
# DeflateWindowSize 15

</IfModule>
```

Figure 7-3. mod_deflate.conf for SunOS/AIX/Linux (Continued)

See Figure 7-4 for the `mod_deflate.conf` file's configuration content for the Windows operating system environment.

```
# Load deflate_module
<IfModule !mod_deflate.c>
LoadModule deflate_module modules/mod_deflate.so
LoadModule headers_module modules/mod_headers.so
</IfModule>

<IfModule mod_deflate.c>

# Insert filter for output compression
SetOutputFilter DEFLATE

# Insert filter for input decompression
SetInputFilter DEFLATE

# Netscape 4.x has some problems...
BrowserMatch ^Mozilla/4 gzip-only-text/html

# Netscape 4.06-4.08 have some more problems
BrowserMatch ^Mozilla/4\.0[678] no-gzip

# MSIE masquerades as Netscape, but it is fine
```

Figure 7-4. mod_deflate.conf for Windows

```
# BrowserMatch \bMSIE !no-gzip !gzip-only-text/html

# NOTE: Due to a bug in mod_setenvif up to Apache 2.0.48
# the above regex won't work. You can use the following workaround
BrowserMatch \bMSI[E] !no-gzip !gzip-only-text/html

# Don't compress images
SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png)$ no-gzip dont-vary

# Don't compress the zipped files
SetEnvIfNoCase Request_URI \.(?:exe|t?gz|zip|bz2|sit|rar)$ no-gzip
         dont-vary



# Don't compress the pdf files
SetEnvIfNoCase Request_URI \.pdf$ no-gzip dont-vary

# Make sure proxies don't deliver the wrong content
<IfModule headers_module>
        Header append Vary User-Agent env=!dont-vary
</IfModule>

# DeflateFilterNote Directive & accurate logging
DeflateFilterNote Input input_info
DeflateFilterNote Output output_info
DeflateFilterNote Ratio ratio_info
LogFormat '"%r" %{output_info}n/%{input_info}n (%{ratio_info}n%%)'
         deflate
CustomLog D:/m1/shared/Apache2/logs/deflate_log deflate

# If you decide to prevent compliant proxies from caching entirely,
         uncomment the line below
# Header set Vary *

# DeflateBufferSize Directive - Specifies the size in bytes of the
         fragments
```

Figure 7-4. mod_deflate.conf for Windows (Continued)

```
# that zlib should compress at one time. Default value is 8096
# DeflateBufferSize 8096


# DeflateCompressionLevel Directive - Specifies what level of
         compression should be used,
# the higher the value, the better the compression, but more CPU time
         is required.
# A value between 1-9.  Default is 6
# DeflateCompressionLevel 6


# DeflateMemLevel Directive - Specifies how much memory should be used
         by zlib for compression
# (a value between 1 and 9). Default is 9.
# DeflateMemLevel 9


# DeflateWindowSize Directive - Specifies the zlib compression window
         size
# (a value between 1 and 15). Default is 15.
# DeflateWindowSize 15


</IfModule>
```

Figure 7-4. mod_deflate.conf for Windows (Continued)

Verify Voyager Processes/Services

A

Overview

The purpose of this appendix is provide you with the steps to confirm successful running processes/services for:

- vpds
- vprimo
- vxws
- vwebv

The following sections step you through the verification process of identifying ports and URL testing:

Tomcat Processes

Before proceeding to any of the other steps, confirm that your Tomcat processes are running. See Figure A-1 for the commands that you can run to confirm that the following Tomcat processes are running:

- vpds
- vprimo
- vxws
- vwebv

```
[VGER] voyager@supzv721 : voyager/ => ps -fu voyager| grep xxxdb | grep tomcat
[VGER] voyager@supzv721 : voyager/ => ps -fu voyager| grep xxxdb | grep tomcat
 voyager 13908  6863   0   May 05 ?          47:50 /m1/voyager/xxxdb/tomcat/vprimo01/java/
          bin/java -server -Xms64m -Xmx512m -Xss1m
 voyager 14002  6863   0   May 05 ?         46:47 /m1/voyager/xxxdb/tomcat/vprimo/java/bin/
          java -server -Xms64m -Xmx512m -Xss1m -
 voyager 13791  6863   0   May 05 ?          47:37 /m1/voyager/xxxdb/tomcat/vpds/java/bin/
          java -server -Xms64m -Xmx512m -Xss1m -Xi
 voyager 21025  6863   0 12:36:45 ?          0:35 /m1/voyager/xxxdb/tomcat/vwebv/java/bin/
          java -server -Xms64m -Xmx512m -Xss1m -X
 voyager 20860  6863   0 12:36:24 ?           1:08 /m1/voyager/xxxdb/tomcat/vxws/java/bin/
          java -server -Xms64m -Xmx512m -Xss1m -Xi
```

Figure A-1. Verification of Tomcat processes example

Identify Ports for VPDS, VPrimo, VXWS, and VWEBV

You need the vpds, vprimo, vxws, and vwebv ports to build the URL for verification of Apache and other URL tests. See Procedure A-1, Locating VPDS, VPrimo, VXWS, and VWEBV Ports.

Procedure A-1. Locating VPDS, VPrimo, VXWS, and VWEBV Ports

To locate the ports that the vpds, vprimo, vxws, and vwebv virtual hosts are listening on, you need to:

1. Locate the file that contains the port information.

See Figure A-2 for an example of the commands to do this.

```
[VGER] voyager@supzv721 : voyager/ => cd /m1/shared/apache2/conf/ActivatedVirtualHosts/
[VGER] voyager@supzv721 : ActivatedVirtualHosts/ => ls -la | grep xxxdb | grep vpds
lrwxrwxrwx   1 root     root          47 Feb  9 16:44 xxxdb_vpds_httpd.conf -> ../
          ConfiguredVirtualHosts/xxxdb_vpds_httpd.conf
```

Figure A-2. Locate file containing vpds port information

2. Open the files to display the port.

   Figure A-3 shows an example of opening a file with vi, and Figure A-4 and Figure A-5 show an example of displaying the port information using the `cat` command where xxxdb is your database name.

```
[VGER] voyager@supzv721 : ActivatedVirtualHosts/ => vi xxxdb_vpds_httpd.conf
```

Figure A-3. Example of opening file with vi

```
[VGER] voyager@supzv721 : ActivatedVirtualHosts/ => cat xxxdb_vprimo_httpd.conf
```

Figure A-4. Example of cat command to display file contents for vprimo

```
[VGER] voyager@supzv721 : ActivatedVirtualHosts/ => cat xxxdb_vxws_httpd.conf
```

Figure A-5. Example of cat command to display file contents for vxws

3. Find the specific port number from the files that you have opened.

   Figure A-6 and Figure A-7 show examples of the section that contains the port information. In Figure A-6, 7044 is the port that has been configured for vpds; and in Figure A-7, 7024 is the port that has been configured for vprimo.

See Table A-1 for a list of the ports designated for these processes/services.

Table A-1. Ports

Process/Service	Port*
vpds	7x44
vprimo	7x24
vxws	7x14
vwebv	7x08
* where x is equal to a value from 0 through 9	

```
##########################################
###   VirtualHost for Voyager PDS xxxdb    ###
##########################################

Listen *:7044
<VirtualHost *:7044>
        #ServerName XXX.XXX.XXX.XXX
        DocumentRoot "/m1/voyager/xxxdb/tomcat/vpds/context/vpds/htdocs"
        JkMount /vpds/* ajp13_lb_xxxdb_vpds

        ErrorLog logs/xxxdb/error.log
        CustomLog logs/xxxdb/access.log common
</VirtualHost>
```

Figure A-6. Example of section with port information

```
##########################################
###   VirtualHost for VPRIMO xxxdb       ###
##########################################

Listen *:7024
<VirtualHost *:7024>
        #ServerName XXX.XXX.XXX.XXX
        DocumentRoot "/m1/voyager/xxxdb/tomcat/vprimo/context/vprimo/htdocs"
        JkMount /vprimo/* ajp13_lb_xxxdb_vprimo

        ErrorLog logs/xxxdb/error.log
        CustomLog logs/xxxdb/access.log common
</VirtualHost>
```

Figure A-7. Example of section with vprimo port information

VPDS, VPrimo, VXWS, and VWEBV Port/Apache Verification

You need to verify that Apache is listening on the correct port by displaying the virtual host's index page in your browser.

Using the ports that you located in Identify Ports for VPDS, VPrimo, VXWS, and VWEBV on page A-2, enter the URLs in your browser using the format in Figure A-8.

```
http://<ip>:<port>/
```

Figure A-8. URL format

The resulting display should look like the one in Figure A-9. The message displayed in Figure A-9 confirms that Apache is listening on the correct port.

Figure A-9. Example of successful Apache test for vpds

URL Test for VPDS - Authentication

To verify borrower authentication (bor-auth) for the vpds virtual host, enter a URL in your browser using the format in Figure A-10.

```
http://<server IP>:<port>/vpds/auth?bor_id=<lastname>&verification=<patron_barcode>
```

Figure A-10. URL format for authentication

A successful URL entry returns a display similar to Figure A-11.

Figure A-11. Successful auth URL result

URL Test for VPDS - Information

To verify full borrower information (bor-info) for the vpds virtual host, enter a URL in your browser using the format in Figure A-12.

```
http://<server IP>:<port>/vpds/info?bor_id=<lastname>&verification=<patron_barcode>
```

Figure A-12. URL format for information (bor-info)

A successful URL entry returns a display similar to Figure A-13.

```
<bor-info>
    <id>338</id>
    <institute>Voyager Database</institute>
    <group>Undergraduate Student</group>
    <group_id>4</group_id>
    <ubid>1@MAST20001DB20020910104124</ubid>
    <password>      </password>
    <passwordType>B</passwordType>
    <name>Bit, J</name>
    <lastname>Bit</lastname>
    <email_address>12345678901234567890123456789012345678901234567890</email_address>
    <expiry_date>23821231</expiry_date>
    <address_1>Ex Libris Group</address_1>
    <address_2>Customer Support</address_2>
    <address_3>1350 E. Touhy Ave, Suite 200 East</address_3>
    <city>Des Plaines</city>
    <stat_province>IL</stat_province>
    <zip>60018</zip>
    <country>USA</country>
    <telephone_1>           </telephone_1>
    <portal/>
</bor-info>
```

Figure A-13. Successful info URL result

URL Test for VPrimo - Circulation

To verify the vprimo service, enter a URL in your browser using the format in Figure A-14.

```
http://<server IP>:<port>/vprimo/getHoldings?bib_id=<bibliographic record number>
```

Figure A-14. URL format for authentication

A successful URL entry returns a display similar to Figure A-15.

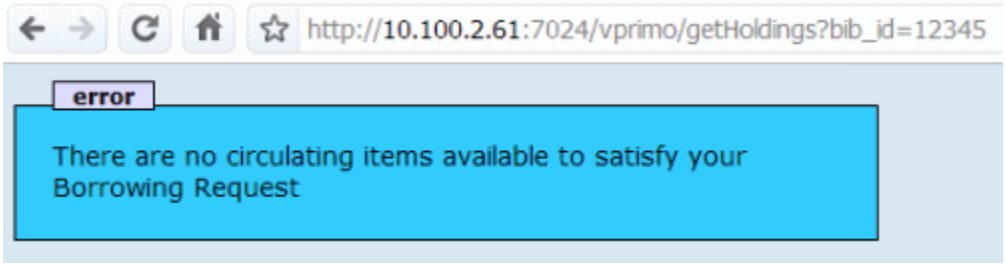

Figure A-15. Example of a successful URL test for vprimo

URL Test for VXWS and VWEBV - Search

To verify the vxws and vwebv virtual hosts, enter a URL in your browser using the format in Figure A-16.

NOTE:
If you are able to do a search in the Tomcat OPAC, then both vwebv and vxws are working.

```
http://<server IP>:<port>/vwebv/searchBasic?sk=en_US
```

Figure A-16. URL format for authentication

A successful URL entry returns a display similar to Figure A-17.

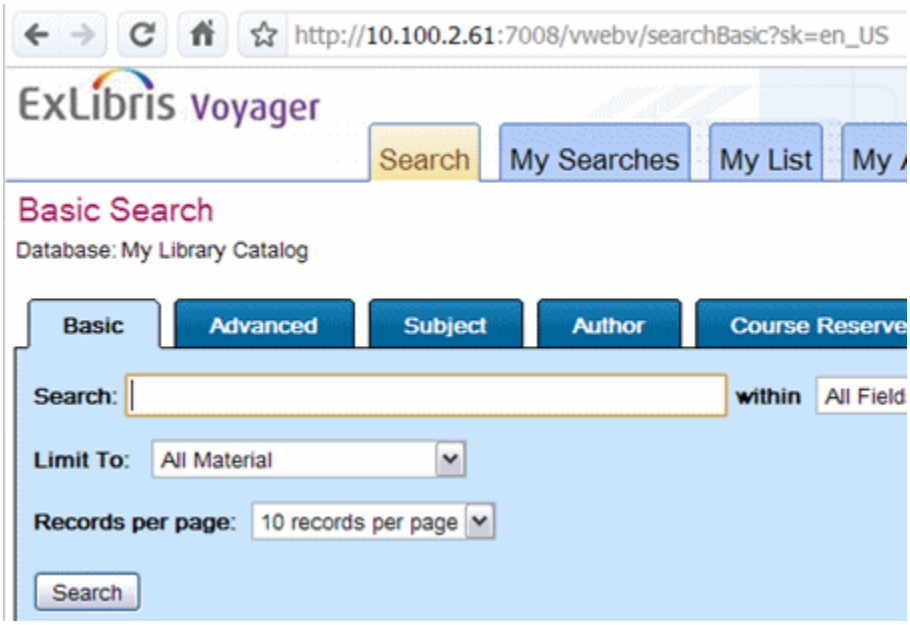

Figure A-17. Example of a successful URL test for vxws and vwebv

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

X