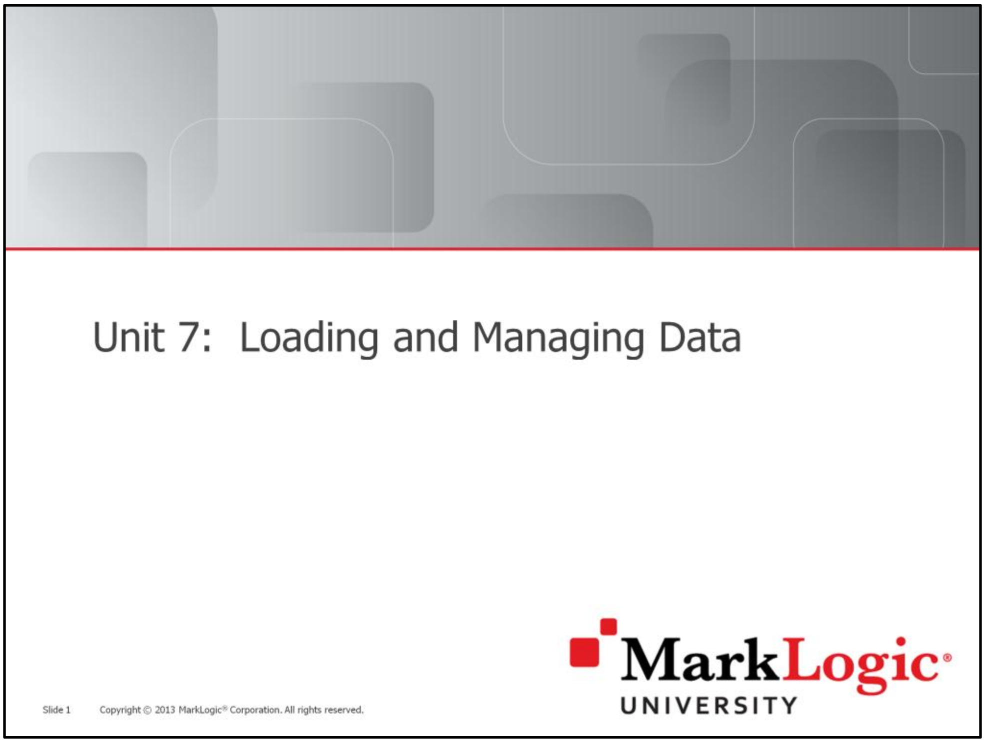
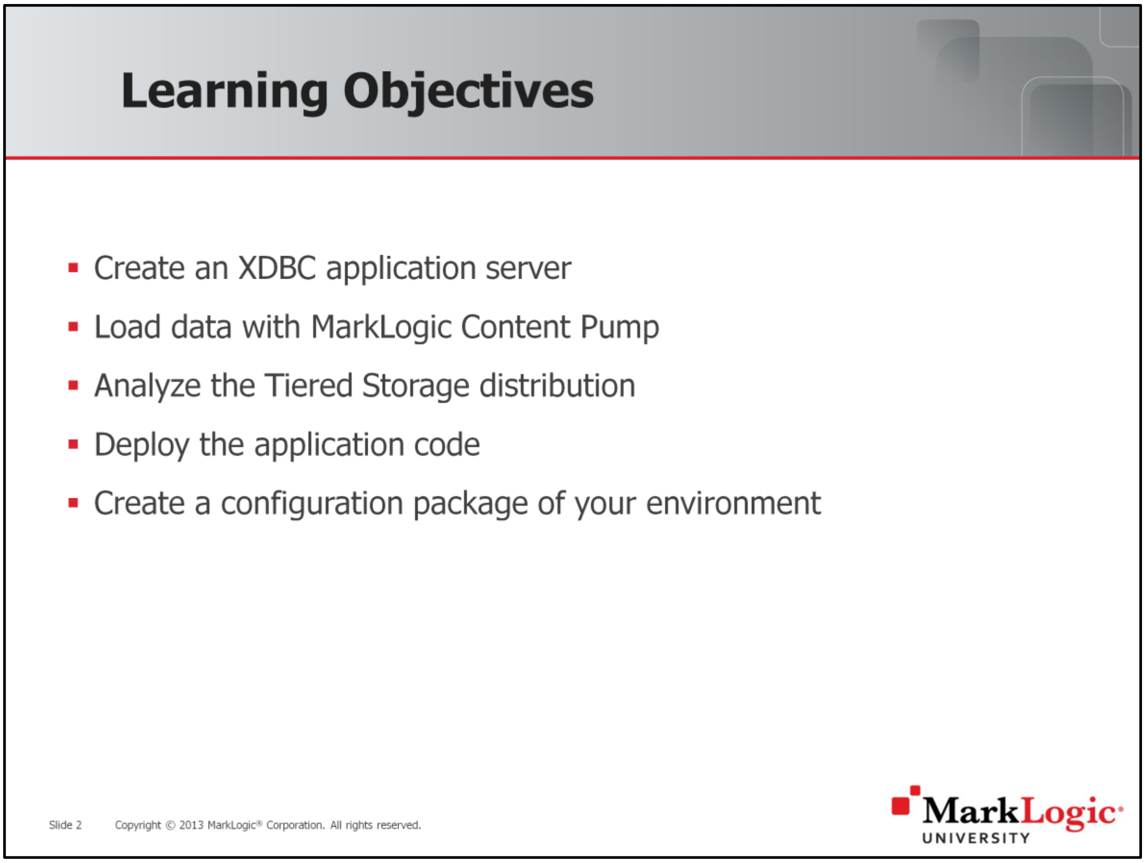


Unit 7: Loading and Managing Data

Slide 1

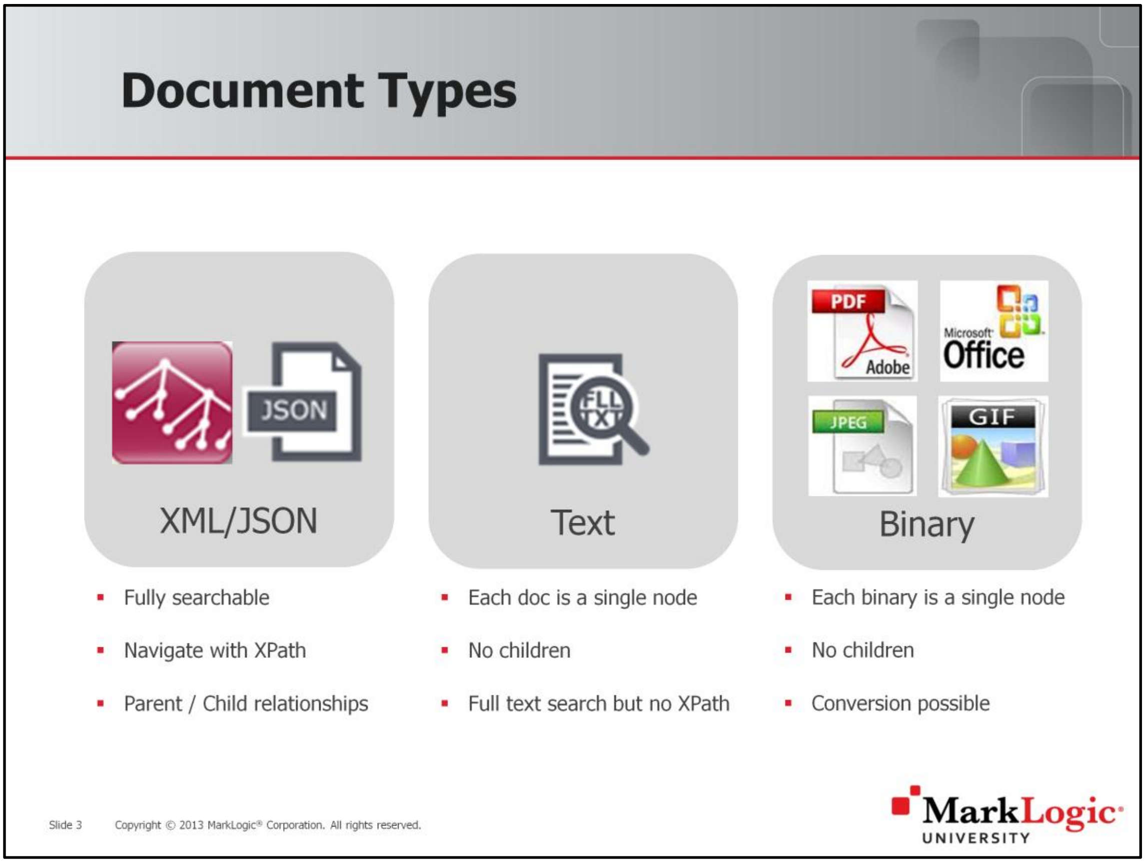
MarkLogic®
UNIVERSITY



Learning Objectives

- Create an XDBC application server
- Load data with MarkLogic Content Pump
- Analyze the Tiered Storage distribution
- Deploy the application code
- Create a configuration package of your environment

Slide 2 Copyright © 2013 MarkLogic® Corporation. All rights reserved.



Document Types

JSON
PDF
Adobe
Microsoft Office
JPEG
GIF

XML/JSON

- Fully searchable
- Navigate with XPath
- Parent / Child relationships

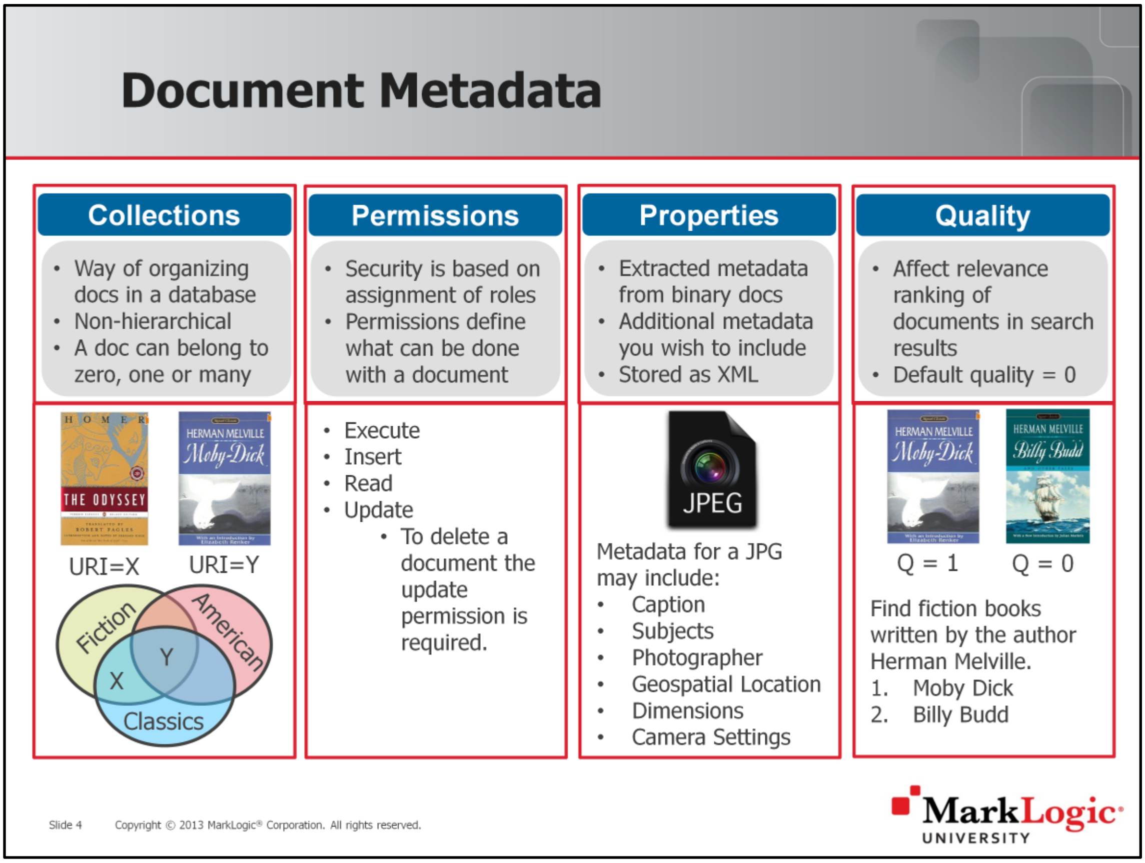
Text

- Each doc is a single node
- No children
- Full text search but no XPath

Binary

- Each binary is a single node
- No children
- Conversion possible

Slide 3 Copyright © 2013 MarkLogic® Corporation. All rights reserved.



Document Metadata

Collections

- Way of organizing docs in a database
- Non-hierarchical
- A doc can belong to zero, one or many

URI=X URI=Y

HOMER
THE ODYSSEY
HERMAN MELVILLE
Moby-Dick
Fiction
American
Y
X
Classics

Permissions

- Security is based on assignment of roles
- Permissions define what can be done with a document

- Execute
- Insert
- Read
- Update
  - To delete a document the update permission is required.

Properties

- Extracted metadata from binary docs
- Additional metadata you wish to include
- Stored as XML

Metadata for a JPG may include:

- Caption
- Subjects
- Photographer
- Geospatial Location
- Dimensions
- Camera Settings

Quality

- Affect relevance ranking of documents in search results
- Default quality = 0

Q = 1 Q = 0

Find fiction books written by the author Herman Melville.

1. Moby Dick
2. Billy Budd

Slide 4 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

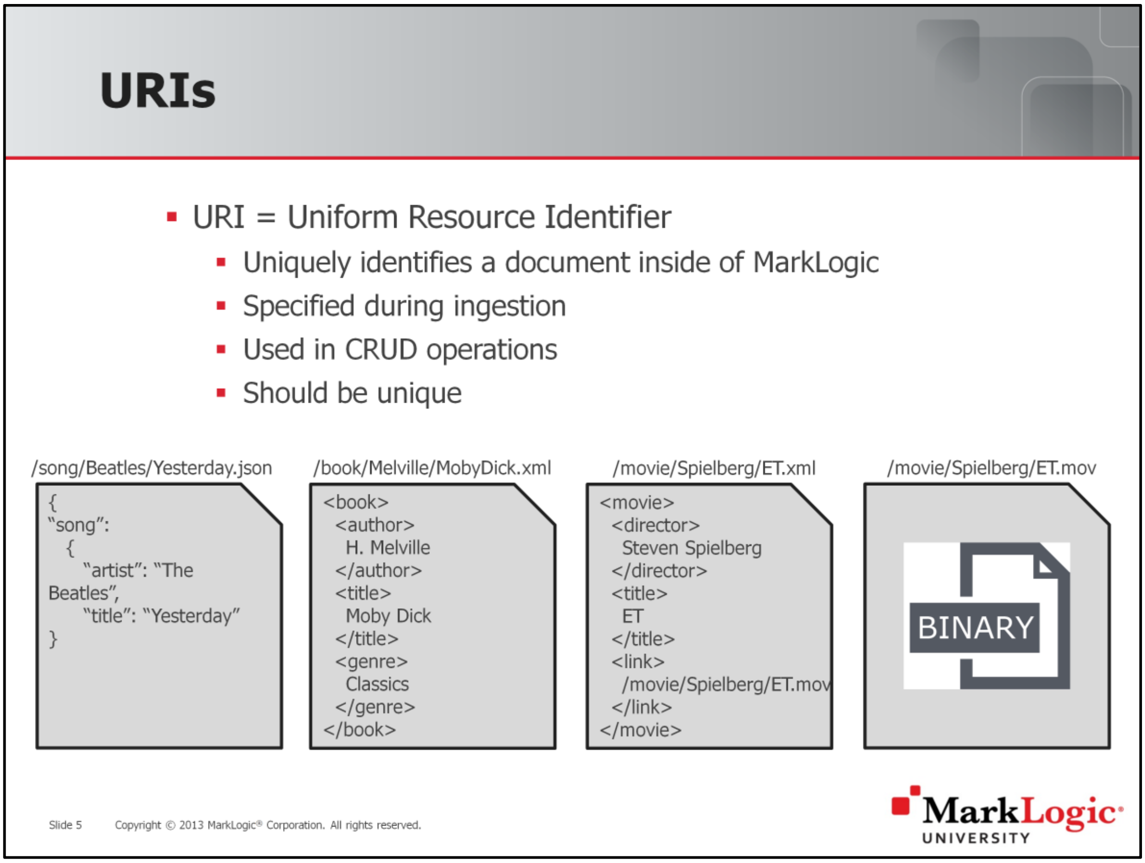
MarkLogic UNIVERSITY

There are many different types of metadata that can be managed and utilized hand in hand along with your documents standard content.

Collections are a way to organize documents in a database. A collection defines a set of documents in the database. You can set documents to be in any number of collections either at the time the document is created or by updating a document. Searches against collections are both efficient and convenient.

Permissions on documents control who can access a document for the capabilities of read, update, insert, and execute. To perform one of these operations on a document, a user must have a role corresponding to the permission for each capability needed.

The quality metadata affects the ranking of documents for use in searches my creating a multiplier for calculating the score for that document, and the default value for quality in the Java API is 0.



URIs

- URI = Uniform Resource Identifier
  - Uniquely identifies a document inside of MarkLogic
  - Specified during ingestion
  - Used in CRUD operations
  - Should be unique

/song/Beatles/Yesterday.json

```
{
"song":
   {
      "artist": "The
Beatles",
      "title": "Yesterday"
}
```

/book/Melville/MobyDick.xml

```
<book>
  <author>
    H. Melville
  </author>
  <title>
    Moby Dick
  </title>
  <genre>
    Classics
  </genre>
</book>
```

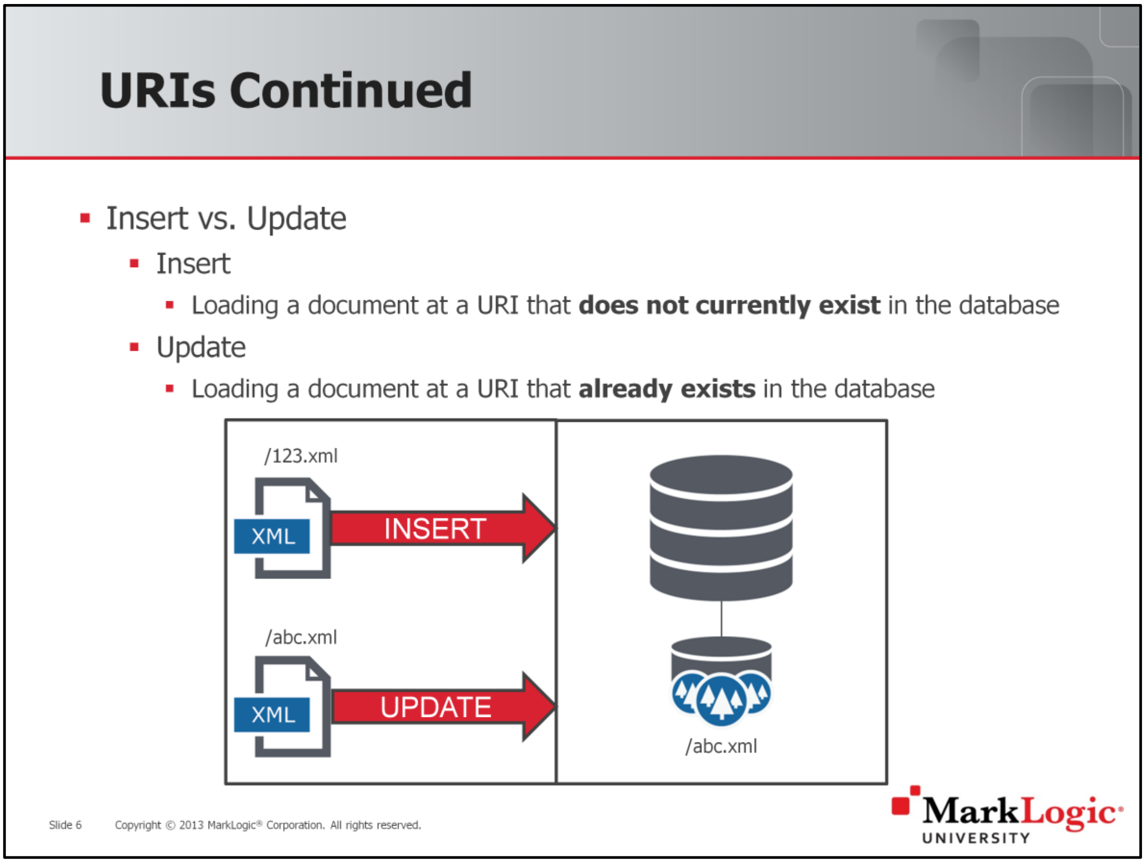
/movie/Spielberg/ET.xml

```
<movie>
  <director>
    Steven Spielberg
  </director>
  <title>
    ET
  </title>
  <link>
    /movie/Spielberg/ET.mov
  </link>
</movie>
```

/movie/Spielberg/ET.mov

BINARY

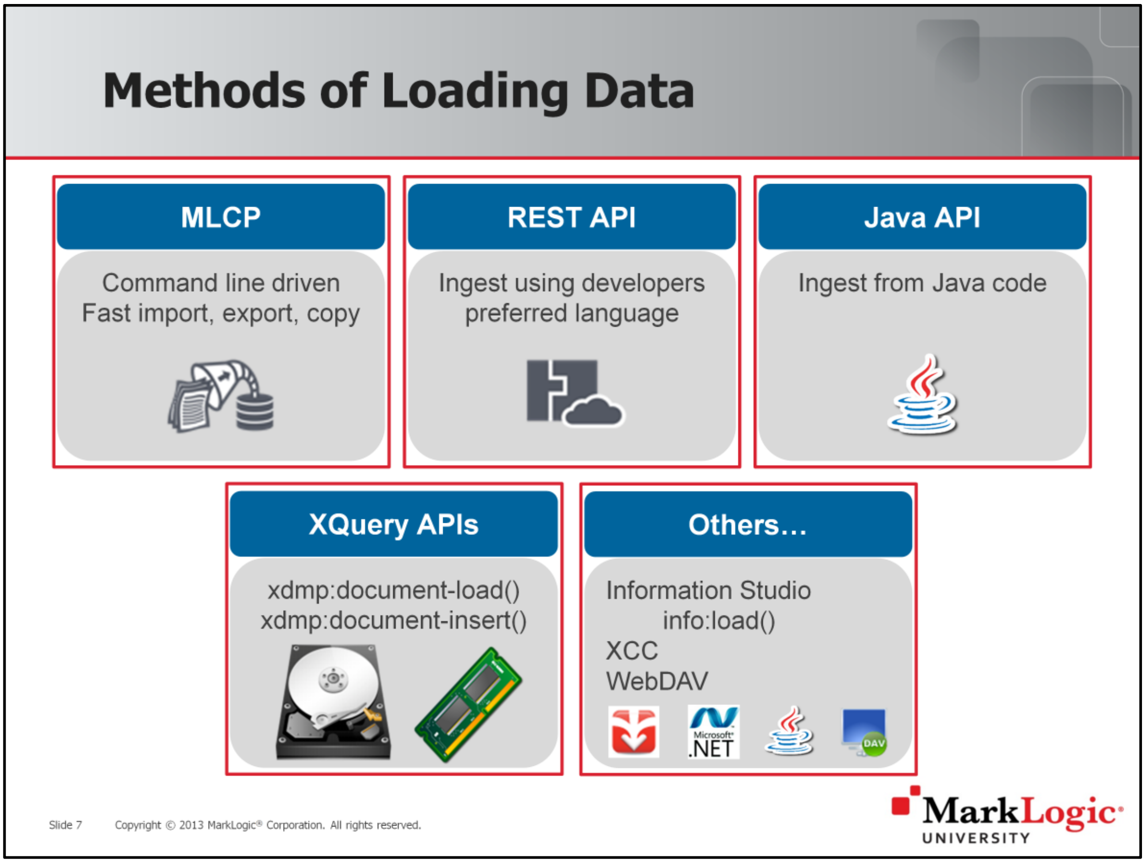
MarkLogic UNIVERSITY



URIs Continued

- Insert vs. Update
  - Insert
    - Loading a document at a URI that **does not currently exist** in the database
  - Update
    - Loading a document at a URI that **already exists** in the database

Slide 6 Copyright © 2013 MarkLogic® Corporation. All rights reserved.



Methods of Loading Data

MLCP

Command line driven
Fast import, export, copy

REST API

Ingest using developers preferred language

Java API

Ingest from Java code

XQuery APIs

xdmp:document-load()
xdmp:document-insert()

Others…

Information Studio
info:load()
XCC
WebDAV

Microsoft .NET
DAV

Slide 7

There are a host of tools and methods available for ingesting content into MarkLogic. Your requirements will help dictate which tool is best for you.

One approach is to create documents in the database using XQuery code. The function xdmp:document-load allows you to load documents from a filesystem, and provides for the customization of the load using specific options. The function xdmp:document-insert allows for a document to be built and inserted on the fly from memory.

Another approach is to use a tool such as Information Studio. Developed by MarkLogic, Information Studio provides a transactional, wizard based approach to loading content, while also providing flexibility for customized actions via external Xquery or XSLT code.

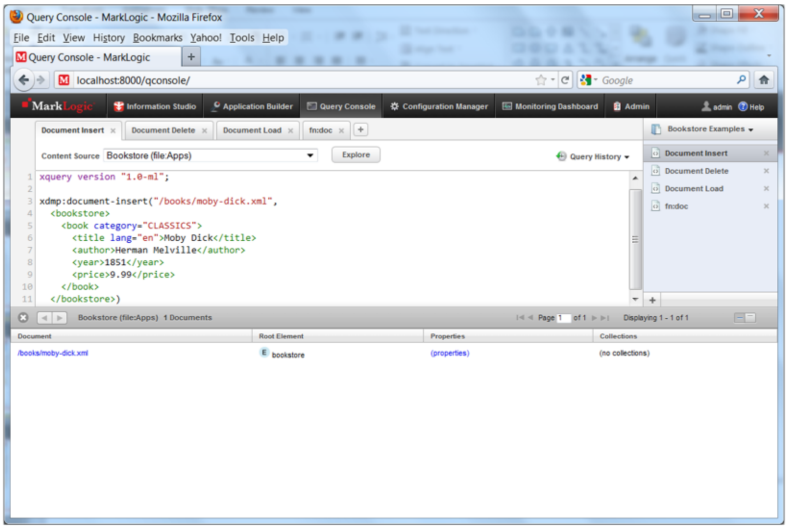
WebDAV provides a drag and drop way to load content, and is a common way to store application code in a modules database.

Record Loader is a Java based tool that is fast and light, and provides the ability to split documents at particular nodes.

Additional methods can be implemented by developing a RESTful web service or using the XCC to access MarkLogic from .Net or Java middleware.

It is important for large data loads to have the ability to break the load up into multiple transactions. But Information Studio and the Information Studio API provide for this requirement.

xdmp:document-insert()



Content Source: Bookstore (file:Apps)

```
xquery version "1.0-ml";

xdmp:document-insert("/books/moby-dick.xml",
  <bookstore>
    <book category="CLASSICS">
      <title lang="en">Moby Dick</title>
      <author>Herman Melville</author>
      <year>1851</year>
      <price>9.99</price>
    </book>
  </bookstore>)
```

Bookstore (file:Apps) 1 Documents — Page 1 of 1 — Displaying 1 - 1 of 1

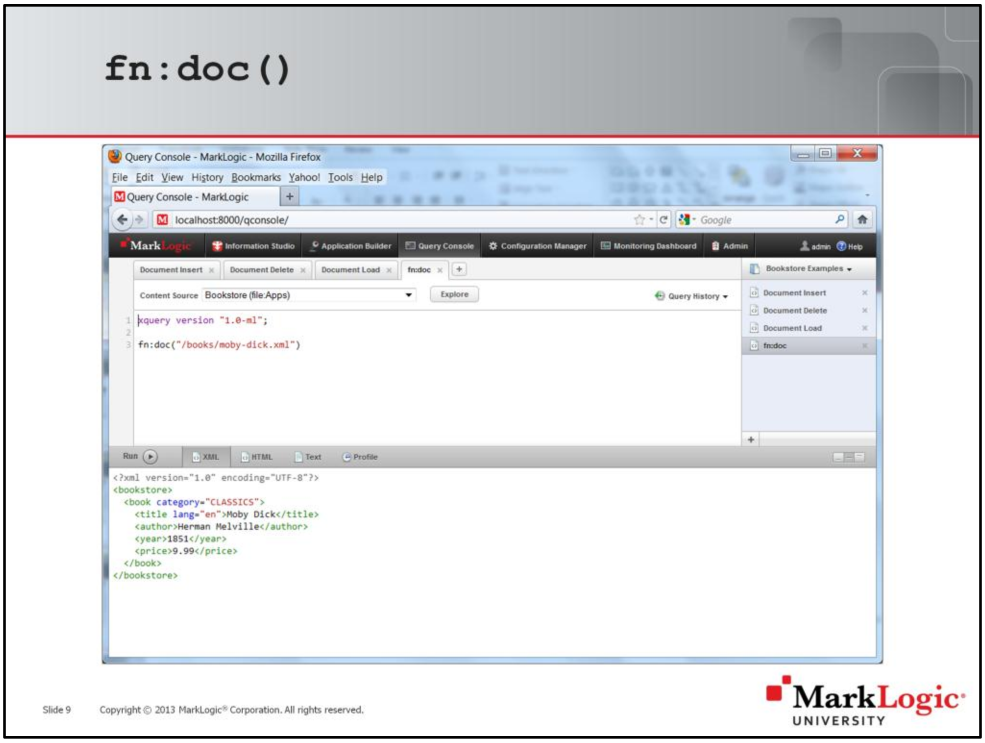
Document	Root Element	Properties	Collections
/books/moby-dick.xml	E bookstore	(properties)	(no collections)

Slide 8 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

Xdmp:document-insert loads content from **memory**. The function gets passed a sequence containing a URI and the XML content.

Be sure to select the correct database from the combo box, then execute the code by clicking the “XML” or “TEXT” button.

You can validate the document was inserted by clicking the “Explore” link.



fn:doc()

Content Source Bookstore (file:Apps)

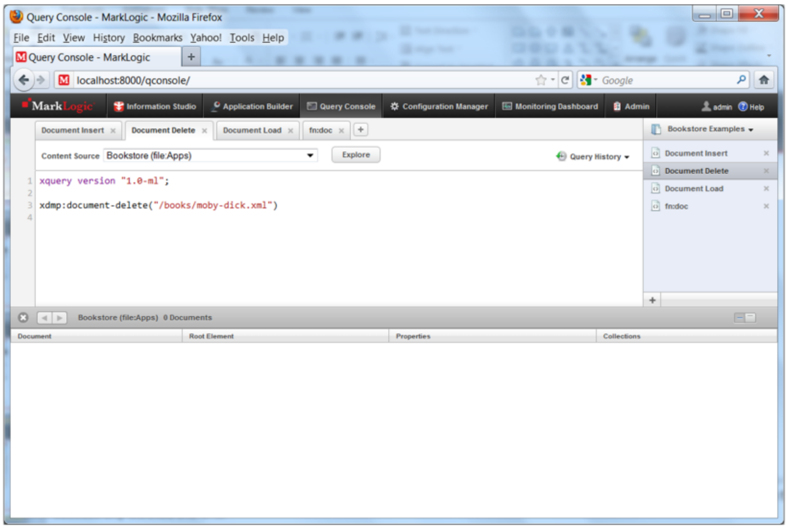
```
xquery version "1.0-ml";

fn:doc("/books/moby-dick.xml")
```

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="CLASSICS">
    <title lang="en">Moby Dick</title>
    <author>Herman Melville</author>
    <year>1851</year>
    <price>9.99</price>
  </book>
</bookstore>
```

Slide 9

This function gets passed a URI and displays a result from the database selected.

xdmp:document-delete()



Content Source: Bookstore (file:Apps)

```
xquery version "1.0-ml";

xdmp:document-delete("/books/moby-dick.xml")
```

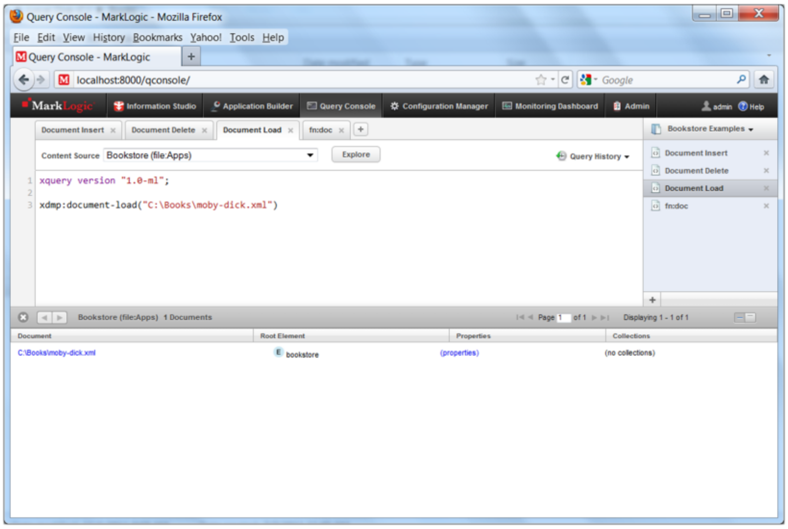
Bookstore (file:Apps) 0 Documents

Document	Root Element	Properties	Collections

Slide 10 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

This function gets passed a URI and deletes it from the database selected. Note the empty sequence returned, which in this case means we successfully deleted the URI.

xdmp:document-load()



Content Source Bookstore (file:Apps)

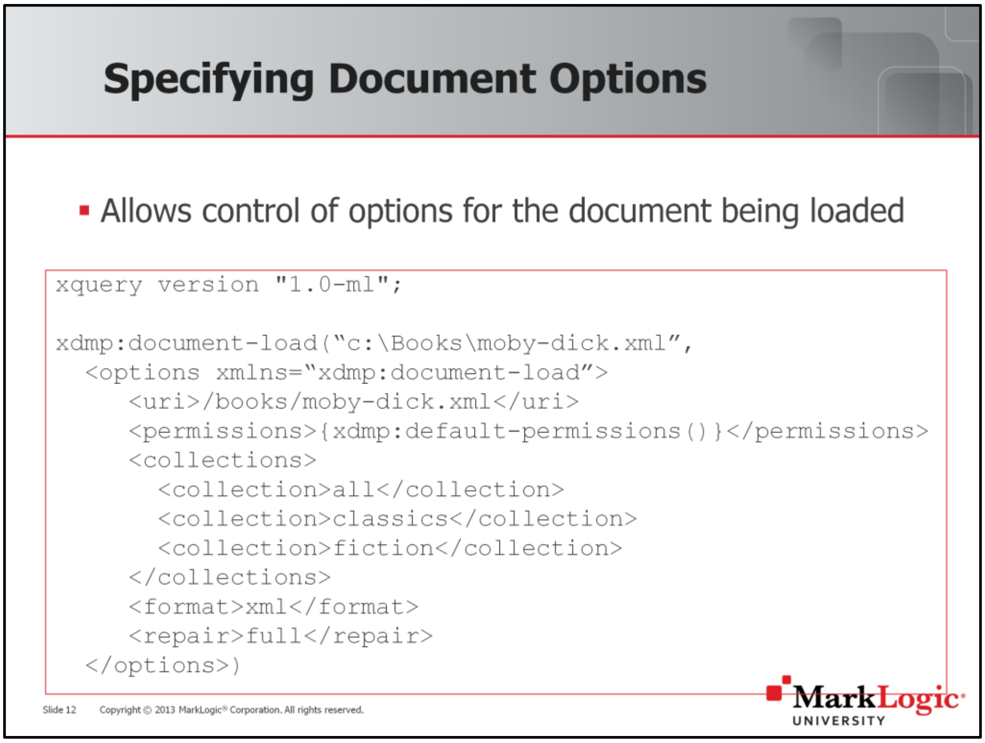
```
xquery version "1.0-ml";

xdmp:document-load("C:\Books\moby-dick.xml")
```

Bookstore (file:Apps) 1 Documents

Document	Root Element	Properties	Collections
C:\Books\moby-dick.xml	bookstore	(properties)	(no collections)

Slide 11 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

Xdmp:document-load inserts a document from the **filesystem**. The function gets passed a full path including filename and extension. This full path including the filename and extension becomes the documents URI.



Specifying Document Options

- Allows control of options for the document being loaded

```
xquery version "1.0-ml";

xdmp:document-load(“c:\Books\moby-dick.xml”,
  <options xmlns=“xdmp:document-load”>
     <uri>/books/moby-dick.xml</uri>
     <permissions>{xdmp:default-permissions()}</permissions>
     <collections>
       <collection>all</collection>
       <collection>classics</collection>
       <collection>fiction</collection>
     </collections>
     <format>xml</format>
     <repair>full</repair>
  </options>)
```

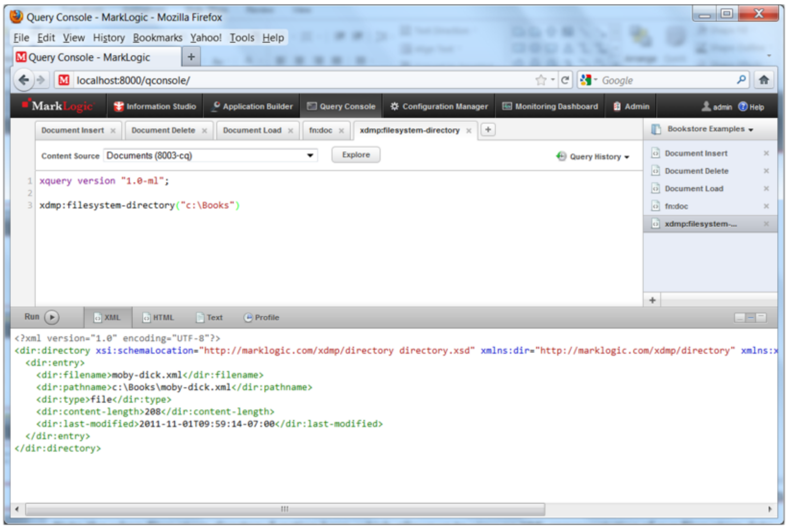
Slide 12   Copyright © 2013 MarkLogic® Corporation. All rights reserved.

There are specific options for each document that you can control at the time of ingestion when using xdmp:document-load.

This code sample is very simple, loading a single document and controlling its URI, permissions, format, and making it a member of certain collections.

We will expand upon this code to make it much more dynamic, allowing for the import of many XML files from directory locations. URI generation can also utilize other naming functions to create dynamic, unique URIs. For example, xdmp:random could be used to generate a random number for inclusion into a URI to ensure it is unique.

Bulk Loading Using FLWOR



Content Source Documents (8003-cq)

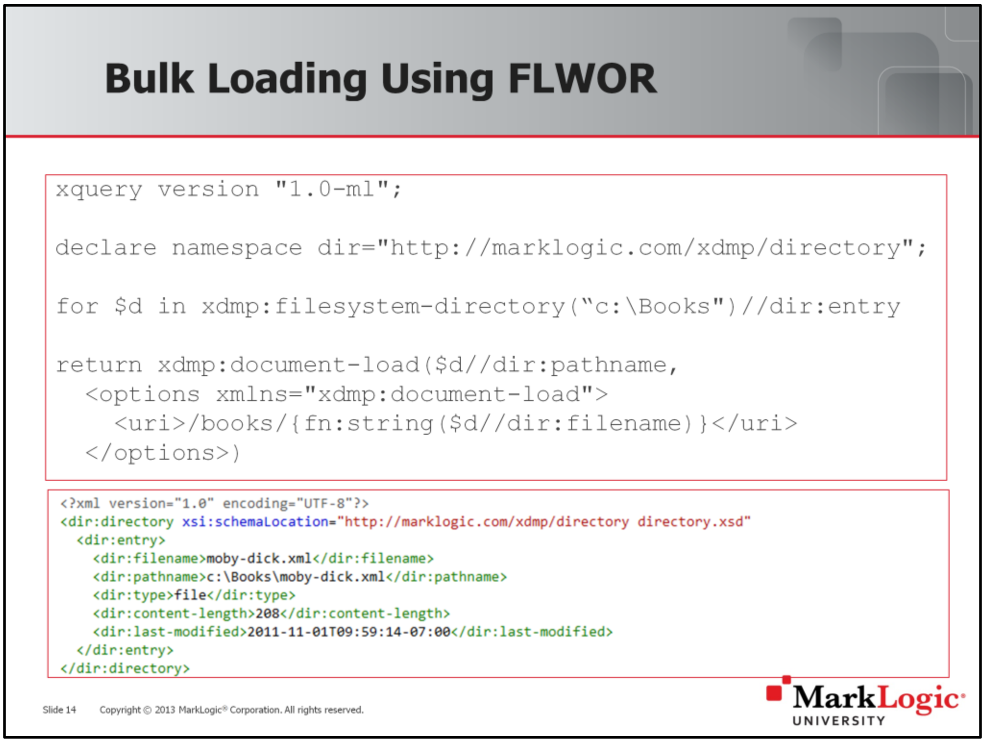
```
xquery version "1.0-ml";

xdmp:filesystem-directory("c:\Books")
```

```
<?xml version="1.0" encoding="UTF-8"?>
<dir:directory xsi:schemaLocation="http://marklogic.com/xdmp/directory directory.xsd" xmlns:dir="http://marklogic.com/xdmp/directory" xmlns:x
  <dir:entry>
    <dir:filename>moby-dick.xml</dir:filename>
    <dir:pathname>c:\Books\moby-dick.xml</dir:pathname>
    <dir:type>file</dir:type>
    <dir:content-length>208</dir:content-length>
    <dir:last-modified>2011-11-01T09:59:14-07:00</dir:last-modified>
  </dir:entry>
</dir:directory>
```

Slide 13

Bulk loading using FLWOR allows for a very customized ingestion process utilizing all the options available to xdmp:document-load.

Note the xdmp:filesystem-directory function here, which allows us to view an XML representation of our filesystem data.

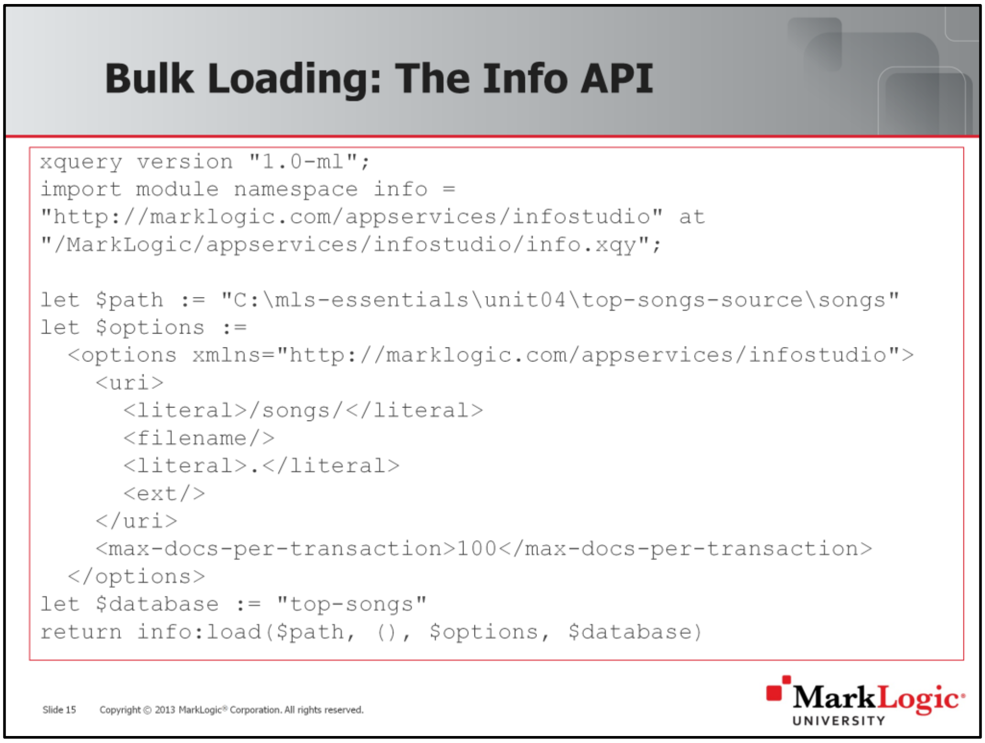


Bulk Loading Using FLWOR

```
xquery version "1.0-ml";

declare namespace dir="http://marklogic.com/xdmp/directory";

for $d in xdmp:filesystem-directory(“c:\Books")//dir:entry

return xdmp:document-load($d//dir:pathname,
  <options xmlns="xdmp:document-load">
    <uri>/books/{fn:string($d//dir:filename)}</uri>
  </options>)
```

```
<?xml version="1.0" encoding="UTF-8"?>
<dir:directory xsi:schemaLocation="http://marklogic.com/xdmp/directory directory.xsd"
  <dir:entry>
    <dir:filename>moby-dick.xml</dir:filename>
    <dir:pathname>c:\Books\moby-dick.xml</dir:pathname>
    <dir:type>file</dir:type>
    <dir:content-length>208</dir:content-length>
    <dir:last-modified>2011-11-01T09:59:14-07:00</dir:last-modified>
  </dir:entry>
</dir:directory>
```

Slide 14



Bulk Loading: The Info API

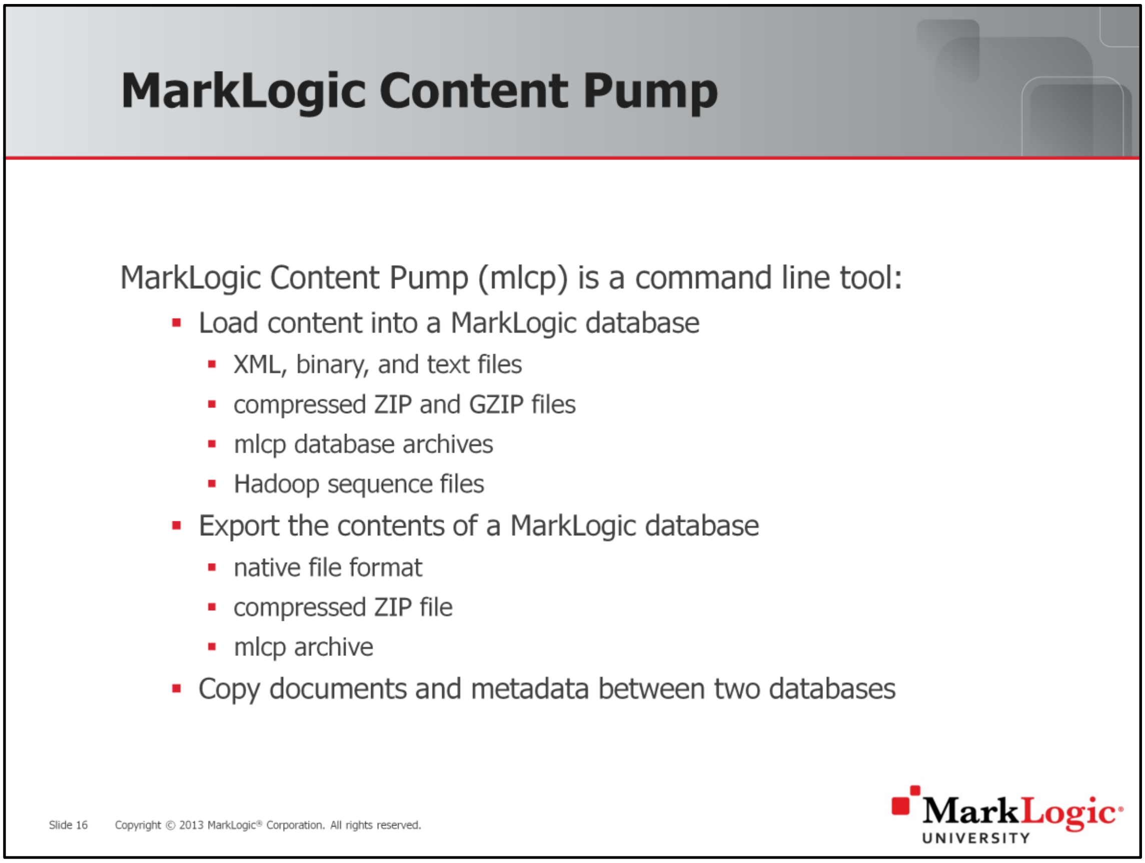
```
xquery version "1.0-ml";
import module namespace info =
"http://marklogic.com/appservices/infostudio" at
"/MarkLogic/appservices/infostudio/info.xqy";

let $path := "C:\mls-essentials\unit04\top-songs-source\songs"
let $options :=
  <options xmlns="http://marklogic.com/appservices/infostudio">
    <uri>
      <literal>/songs/</literal>
      <filename/>
      <literal>.</literal>
      <ext/>
    </uri>
    <max-docs-per-transaction>100</max-docs-per-transaction>
  </options>
let $database := "top-songs"
return info:load($path, (), $options, $database)
```

Slide 15 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

Using the Information Studio API (namespace “info”) provides more control and flexibility to bulk load data than does a standard FLWOR statement using xdmp:document-insert()

For example, some advantages include:

- Ability to control transaction size
- Customization using an options node, such as URIs, collections, error handling, tickets and retention policy, and how to handle duplicate URIs



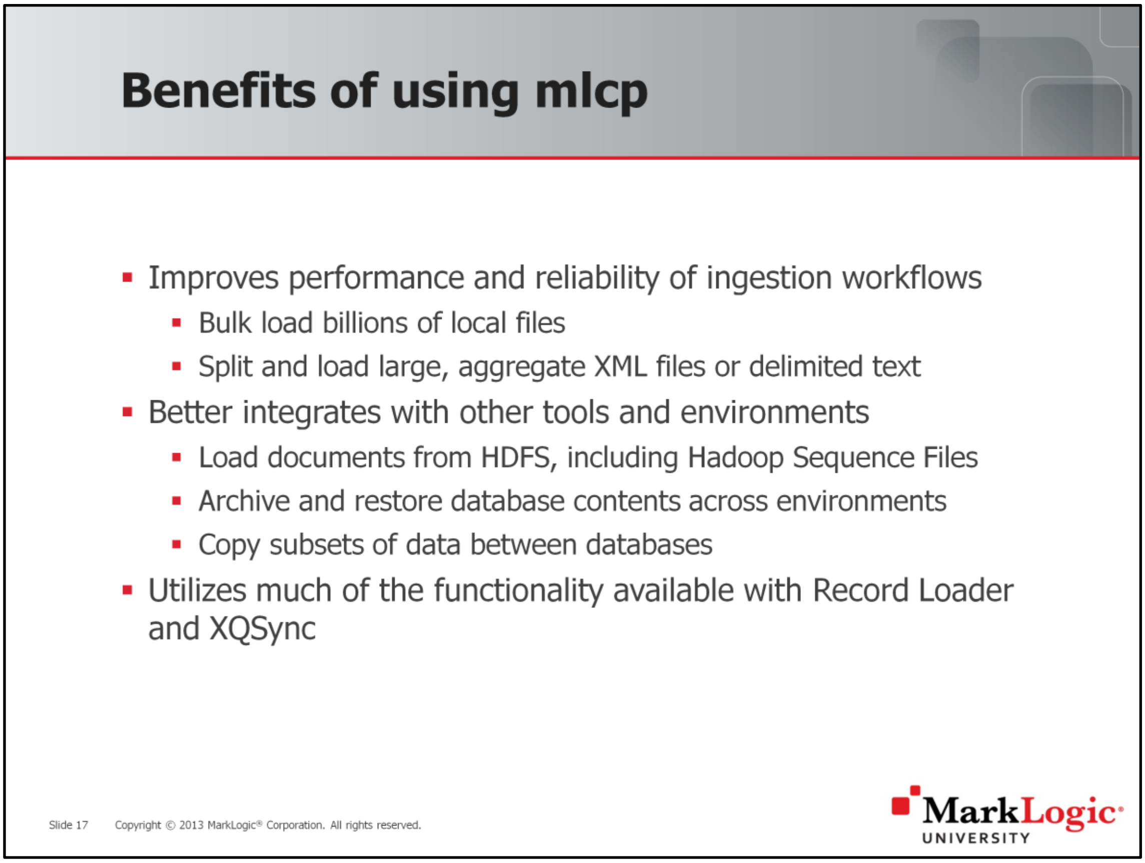
MarkLogic Content Pump

MarkLogic Content Pump (mlcp) is a command line tool:

- Load content into a MarkLogic database
  - XML, binary, and text files
  - compressed ZIP and GZIP files
  - mlcp database archives
  - Hadoop sequence files
- Export the contents of a MarkLogic database
  - native file format
  - compressed ZIP file
  - mlcp archive
- Copy documents and metadata between two databases

Slide 16 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

MarkLogic Content Pump (mlcp) is a command line tool for loading content into and exporting content out of a MarkLogic database.
The mlcp tool is shell script for Linux, Solaris, and OS X and a bat script for Windows.
It allows you to script importing and exporting data from a remote host, for example to bulk load data from a local file system or Hadoop file system, export data and metadata to a platform-independent archive, or copy content between live databases. This makes it ideal for data loads, migration, and database restores.



Benefits of using mlcp

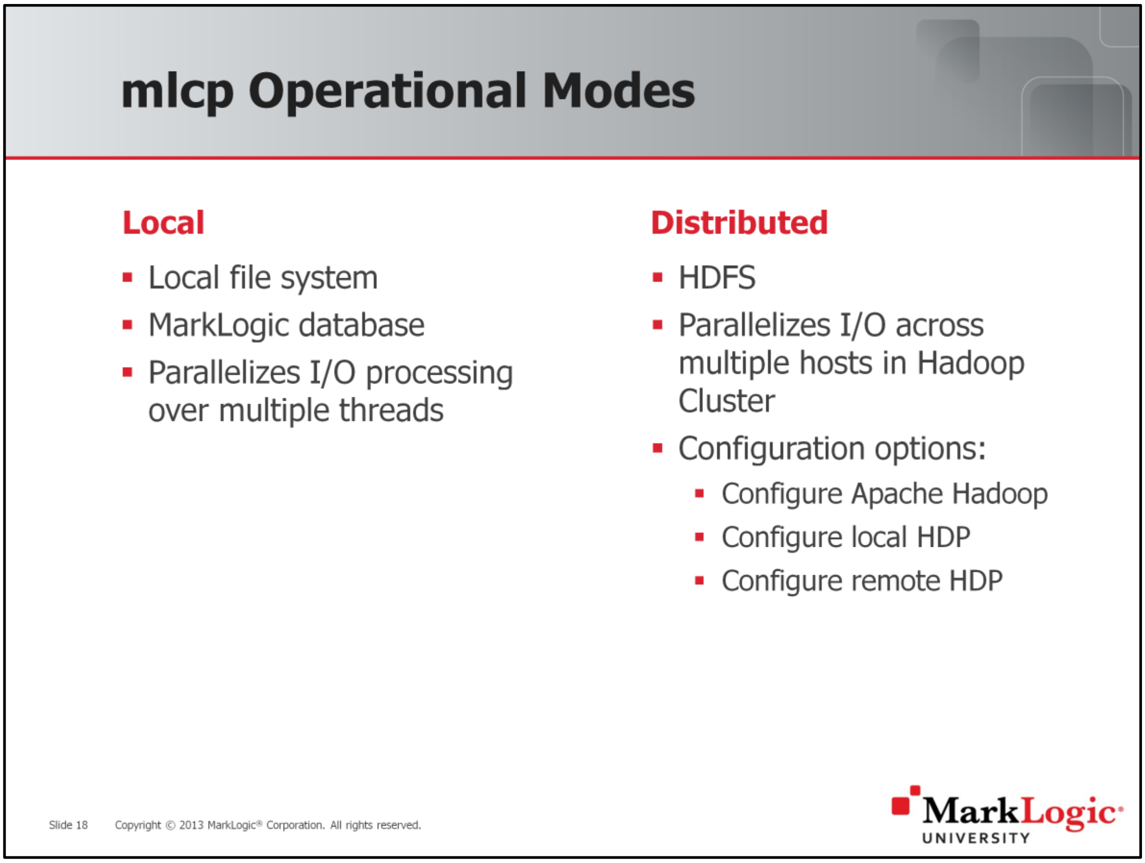
- Improves performance and reliability of ingestion workflows
  - Bulk load billions of local files
  - Split and load large, aggregate XML files or delimited text
- Better integrates with other tools and environments
  - Load documents from HDFS, including Hadoop Sequence Files
  - Archive and restore database contents across environments
  - Copy subsets of data between databases
- Utilizes much of the functionality available with Record Loader and XQSync

Slide 17

The mlcp tool is the fastest way to load data into MarkLogic. It is optimized for large, parallel document loads.
With mlcp you can split large jobs into batches and transparently parallelize I/O using Hadoop.
mlcp uses MarkLogic Connector for Hadoop, and XCC to communicate with a MarkLogic XDBC server.
mlcp is a client-side tool that has many configuration options.
Content Pump supports moving data between a MarkLogic database and any of the following:

- Local file system
- HDFS
- MarkLogic database archive
- Another MarkLogic database

Utilizes much of the functionality available with Record Loader and XQSync



mlcp Operational Modes

Local

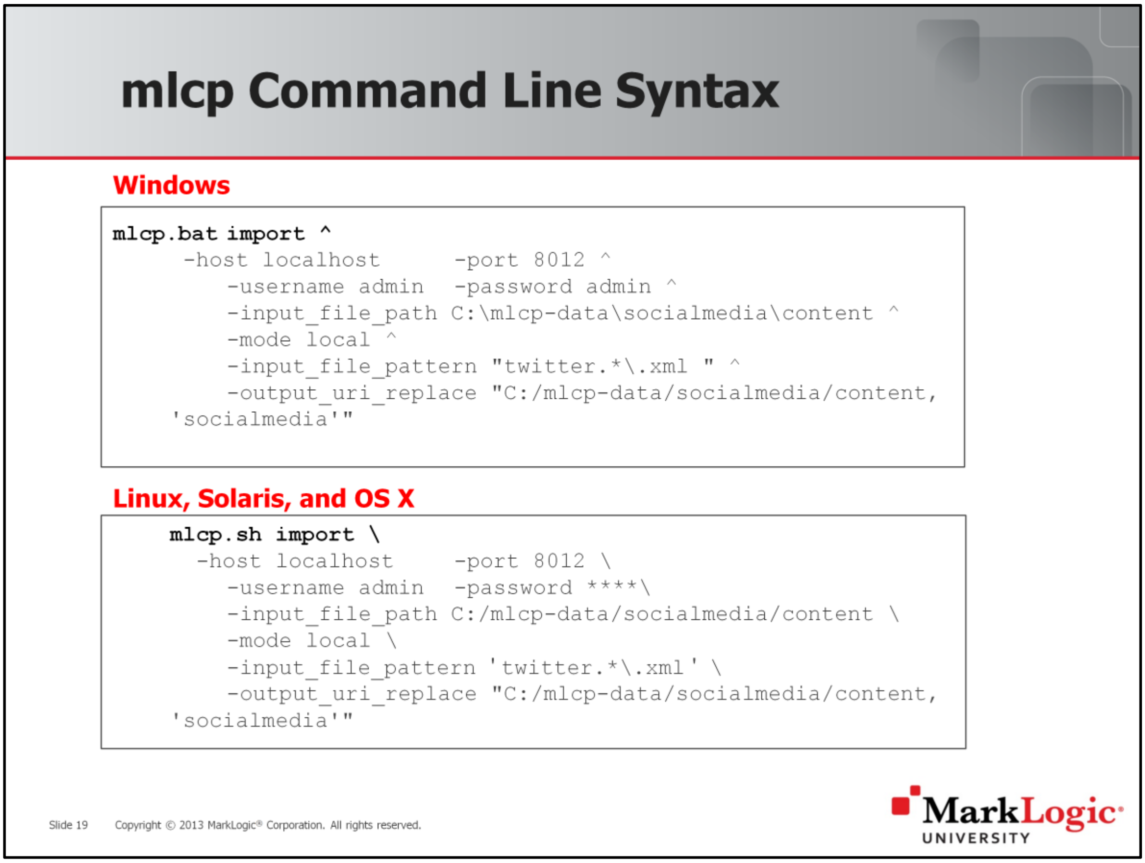
- Local file system
- MarkLogic database
- Parallelizes I/O processing over multiple threads

Distributed

- HDFS
- Parallelizes I/O across multiple hosts in Hadoop Cluster
- Configuration options:
  - Configure Apache Hadoop
  - Configure local HDP
  - Configure remote HDP

Slide 18 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

The mlcp tools has two modes of operation; local mode and distributed mode.
In local mode, mlcp defaults to the local file system on the host where it is invoked. Resources such as import data and export destinations must be reachable from that host. Local mode parallelizes I/O processing over multiple threads.
In distributed mode, an Apache Hadoop or Hortonworks Data Platform (HDP) installation is required with configuration files and libraries available to mlcp locally. mlcp distributes its workloads across the nodes in a Hadoop cluster. Resources such as import data and export destination must be reachable from the cluster, which usually means via HDFS. Distributed mode parallelizes I/O across multiple hosts.
In local mode, mlcp is supported on the same platforms as MarkLogic Server, including 64-bit Linux, Windows, Solaris; and Macintosh OS X. Distributed mode is only supported on 64-bit Linux.
This unit focuses on working with mlcp in local mode on a Windows environment.



mlcp Command Line Syntax

Windows

```
mlcp.bat import ^
     -host localhost      -port 8012 ^
        -username admin   -password admin ^
        -input_file_path C:\mlcp-data\socialmedia\content ^
        -mode local ^
        -input_file_pattern "twitter.*\.xml " ^
        -output_uri_replace "C:/mlcp-data/socialmedia/content,
    'socialmedia'"
```

Linux, Solaris, and OS X

```
    mlcp.sh import \
      -host localhost     -port 8012 \
        -username admin   -password ****\
        -input_file_path C:/mlcp-data/socialmedia/content \
        -mode local \
        -input_file_pattern 'twitter.*\.xml' \
        -output_uri_replace "C:/mlcp-data/socialmedia/content,
    'socialmedia'"
```

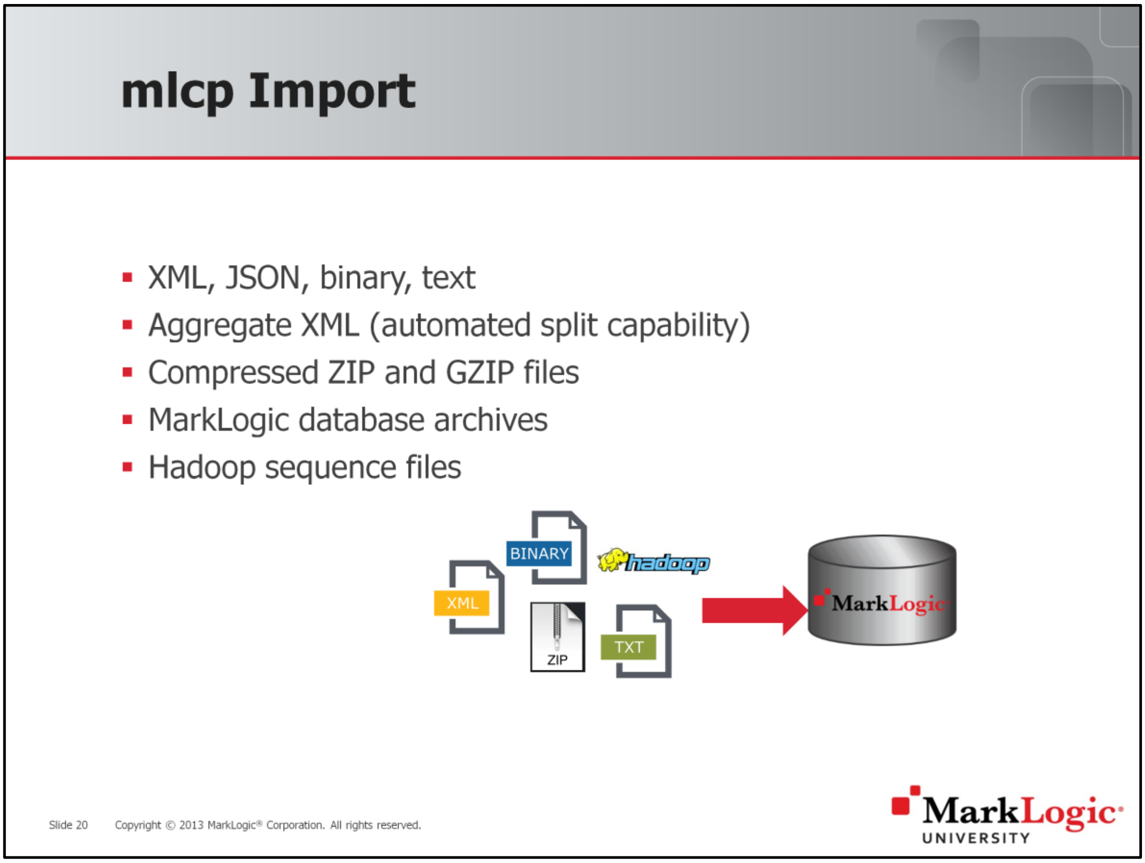
Slide 19

It is important to note that the syntax for scripts executed in Windows differs from Linux, Solaris, and OS X. If you write mlcp commands with Unix command line syntax, they need to be adapted to construct equivalent commands for the Windows command line interpreter to execute on Windows. For example, you replace single quotes with double quotes, and escape characters that have special meaning to the Windows command interpreter.

The first example shows a bat script for importing Twitter data, from the local file system into a MarkLogic database. Below this, is an example shell script that executes the same mlcp import for Linux, Solaris, and OS X environments.

In the examples, the line breaks are aesthetic for the purpose of displaying the code. When you write command line syntax, you do so as a single line from the client bin directory.

There are four mlcp commands, import, export, copy, and help. Each command comprises case-sensitive command options that are prefixed with a single hyphen (-); for example -host. Some command options are boolean while others require a value.



mlcp Import

- XML, JSON, binary, text
- Aggregate XML (automated split capability)
- Compressed ZIP and GZIP files
- MarkLogic database archives
- Hadoop sequence files

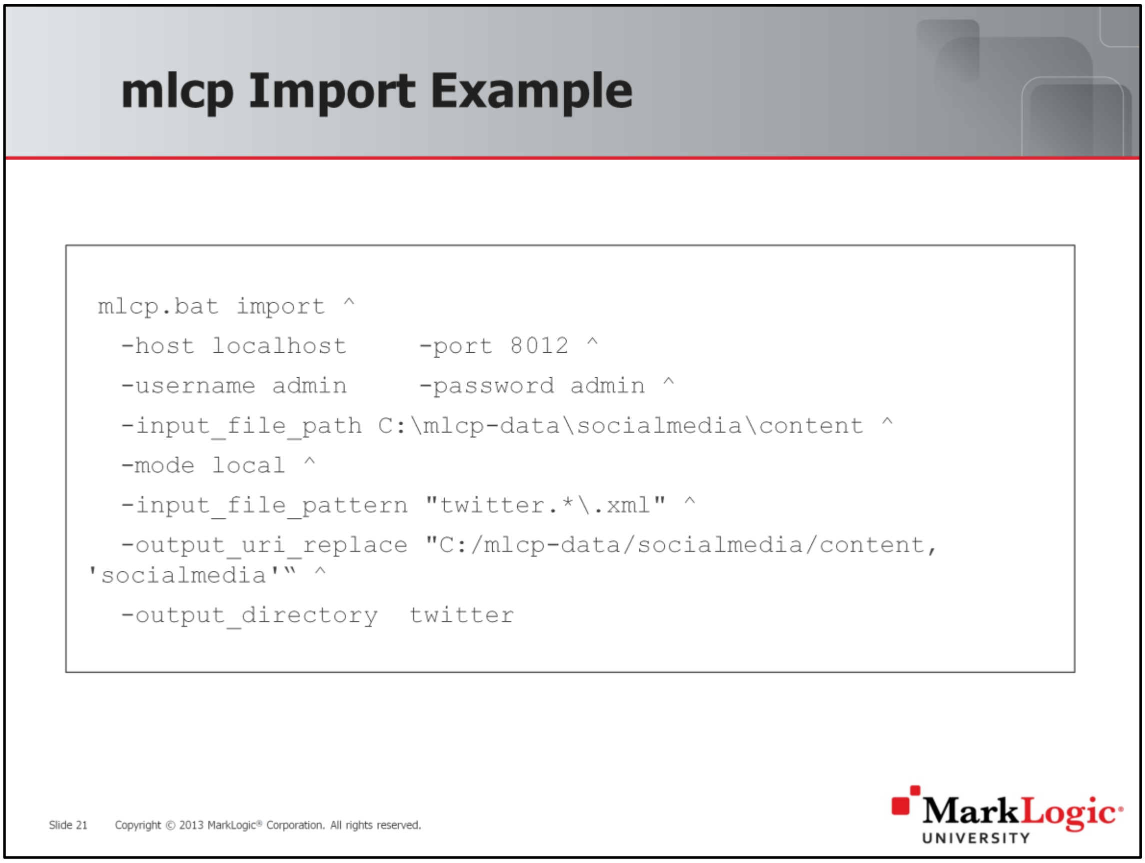
BINARY
hadoop
XML
ZIP
TXT
MarkLogic

Slide 20 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

You can import content from single native files, compressed files and archives, entire directories, and with built-in support for Hadoop, sequence files. The flexibility of mlcp allows you to load the following types of content into a MarkLogic database:

- XML, binary, and delimited text from flat files; for example, a CSV file.
- Aggregate XML, which is XML content that includes recurring element names and which can be split into multiple documents with the recurring element as the document root.
- Compressed files in ZIP or GZIP format. The URIs of documents in compressed files represent the directory hierarchy with the ZIP or GZIP when they are loaded.
- MarkLogic database archives. An archive is a compressed MarkLogic Server database archive created using the mlcp export command which we cover later.
- Hadoop sequence files. A Hadoop sequence file is a flat file consisting of binary key/value pairs that is extensively used in MapReduce as input and output formats.

You can import mixed content types from a directory, using the file suffix and MIME type mappings to determine document type. Any unrecognized or missing suffixes are imported as binary documents.



mlcp Import Example

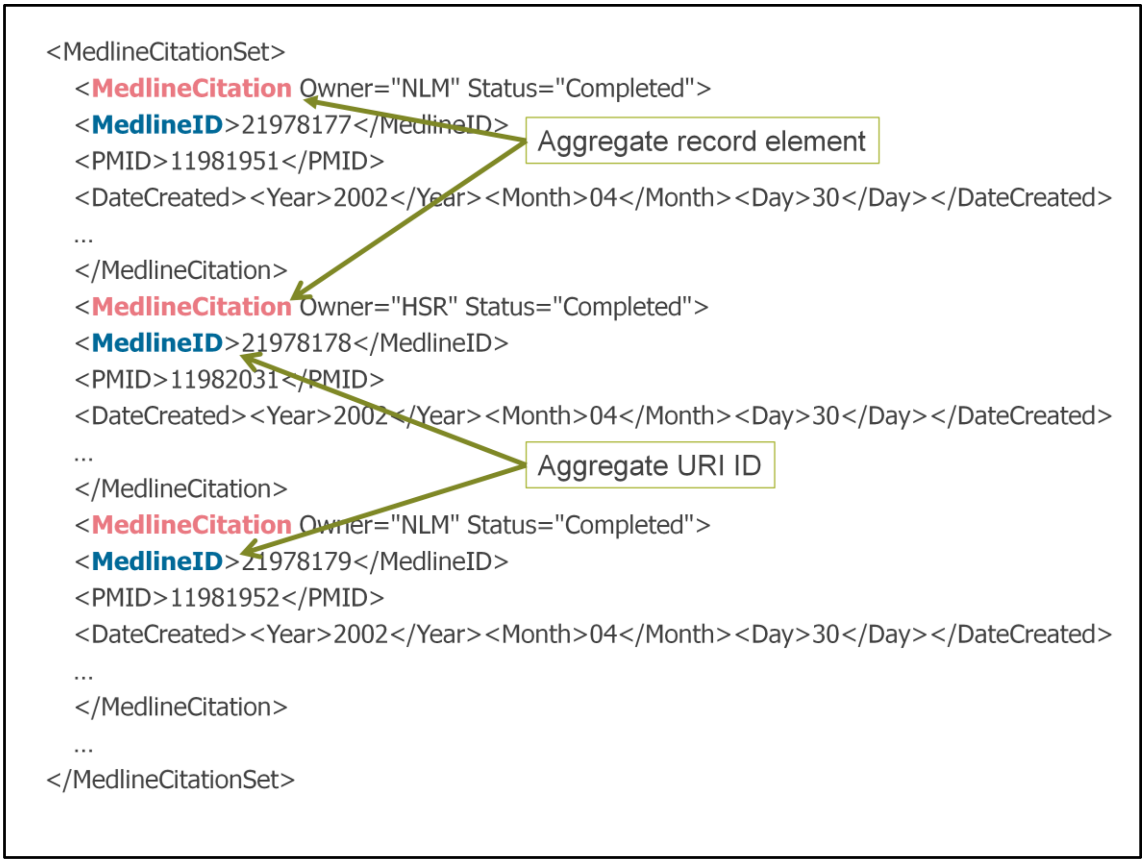
```
 mlcp.bat import ^
   -host localhost     -port 8012 ^
   -username admin     -password admin ^
   -input_file_path C:\mlcp-data\socialmedia\content ^
   -mode local ^
   -input_file_pattern "twitter.*\.xml" ^
   -output_uri_replace "C:/mlcp-data/socialmedia/content,
'socialmedia'" ^
   -output_directory  twitter
```

Slide 21 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

There are various import options that you can use with the mlcp import command for controlling what is imported and how it is stored in the database. For example, there are options specifying what content and metadata is loaded, the URI, collection, directory, and namespace of the documents.

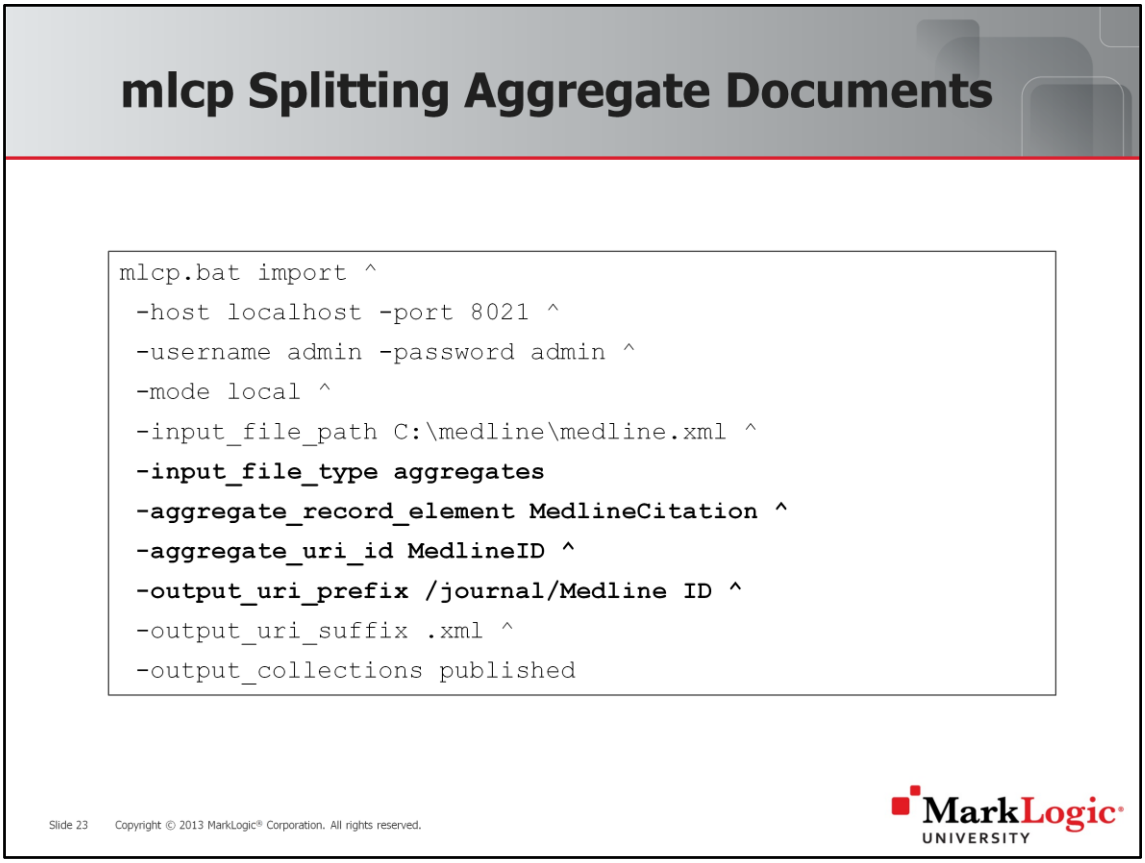
In the example, the bat script is for mlcp import with just some of the available options. The following describes the command options:

- The host machine and port are required, thus the database associated with port 8012 must be reachable.
- The user credentials are specified, thus adequate privileges must be provided to import to the database associated with port 8012.
- The *-input_file_path* is the source of the data import.
- The modes of operation is local mode, thus mlcp loads documents that are local to the mlcp client process.
- The *-input_file_pattern* is a filter that uses Java regular expression to find Twitter documents from a folder on the local file system.
- The *-output_uri_replace* is one of several options for transforming the URI. The default URI is the -input_file_path.
- The –*output_directory* automatically uses the –*fastload* option to force optimal performance by loading directly into the destination forest.



```
<MedlineCitationSet>
   <MedlineCitation Owner="NLM" Status="Completed">
   <MedlineID>21978177</MedlineID>
   <PMID>11981951</PMID>
   <DateCreated><Year>2002</Year><Month>04</Month><Day>30</Day></DateCreated>
   ...
   </MedlineCitation>
   <MedlineCitation Owner="HSR" Status="Completed">
   <MedlineID>21978178</MedlineID>
   <PMID>11982031</PMID>
   <DateCreated><Year>2002</Year><Month>04</Month><Day>30</Day></DateCreated>
   ...
   </MedlineCitation>
   <MedlineCitation Owner="NLM" Status="Completed">
   <MedlineID>21978179</MedlineID>
   <PMID>11981952</PMID>
   <DateCreated><Year>2002</Year><Month>04</Month><Day>30</Day></DateCreated>
   ...
   </MedlineCitation>
   ...
</MedlineCitationSet>
```

Often, many XML documents are presented as a single file, wrapped in an XML element. For example, MedLine citations is a single file comprising many thousands of XML documents in the same file, with a wrapper element around them. You can split aggregate XML in single, multiple, and compressed files.



mlcp Splitting Aggregate Documents

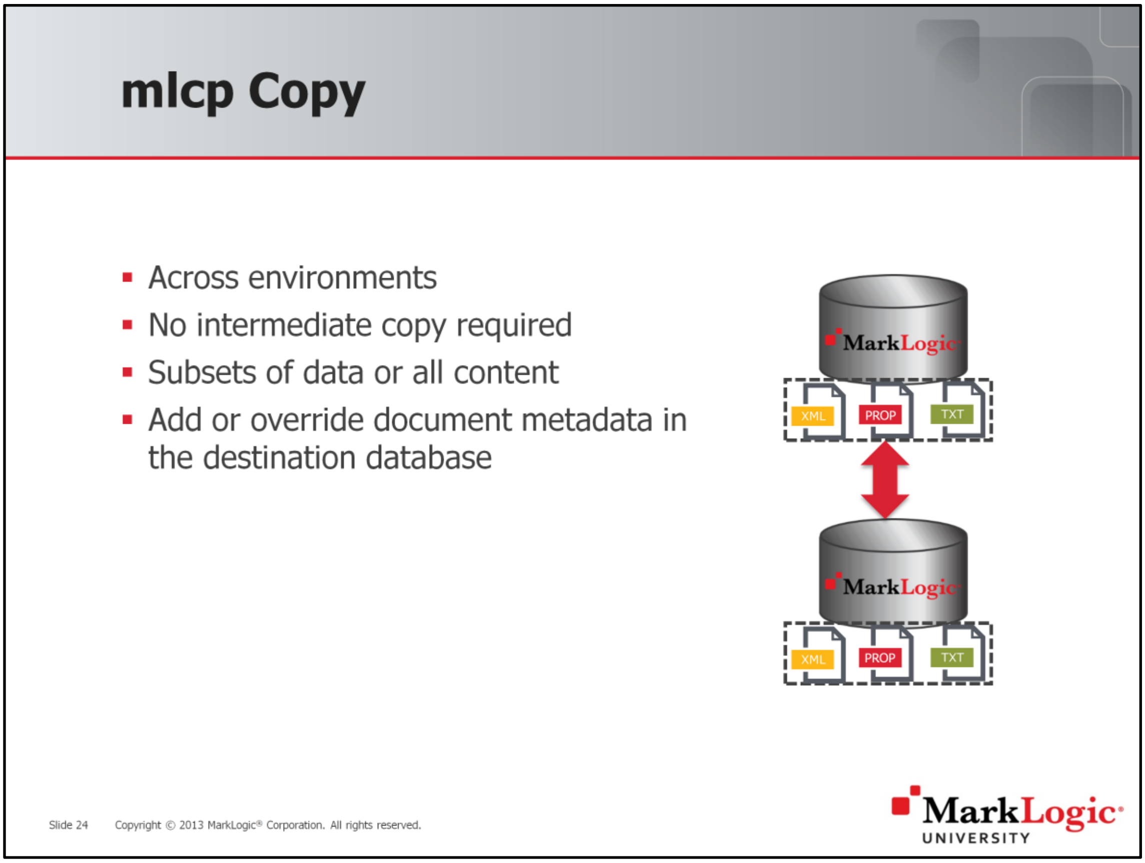
```
mlcp.bat import ^
 -host localhost -port 8021 ^
 -username admin -password admin ^
 -mode local ^
 -input_file_path C:\medline\medline.xml ^
 -input_file_type aggregates
 -aggregate_record_element MedlineCitation ^
 -aggregate_uri_id MedlineID ^
 -output_uri_prefix /journal/Medline ID ^
 -output_uri_suffix .xml ^
 -output_collections published
```

Slide 23

MarkLogic UNIVERSITY

In this example, we can split the Medline citation set with MedlineCitation is the root element and MedlineID is the URI ID. Each journal is loaded as a separate XML document and the original document remains whole on the file system.

The import options -*output_uri_prefix* and -*output_uri_suffix* transform the default URI. The -*output_collections* option assigns the documents to a collection.



mlcp Copy

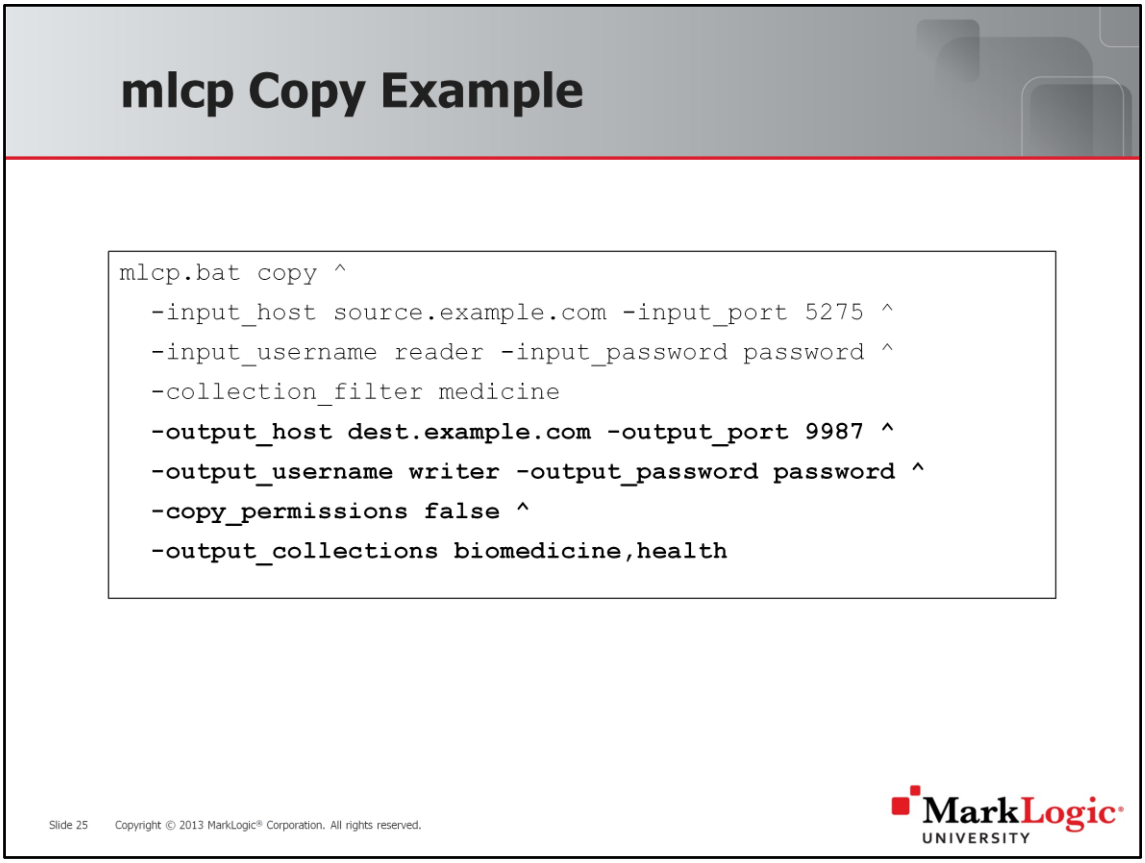
- Across environments
- No intermediate copy required
- Subsets of data or all content
- Add or override document metadata in the destination database

MarkLogic UNIVERSITY

Use the mlcp copy command to copy content and associated metadata from one MarkLogic Server database to another when both are reachable on the network. There is no need to copy to the file system first, the content is copied directly between databases.

When you use mlcp copy, by default all documents are copied, which automatically includes collection, permission, quality and document properties metadata.

There are a number of additional options you can use. For example, you can specify which documents are copied by using directory or collection filters. You can also specify the metadata to include or exclude during copy or override document metadata in the destination database.

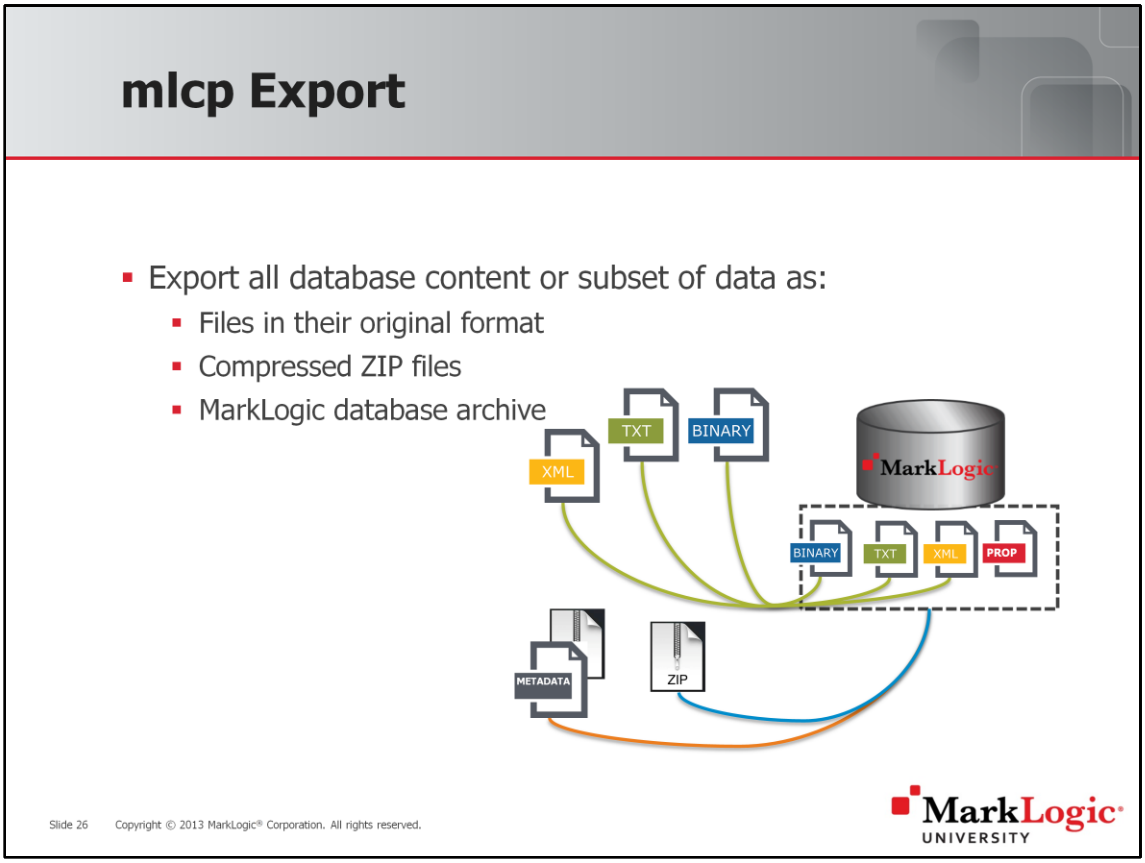


mlcp Copy Example

```
mlcp.bat copy ^
  -input_host source.example.com -input_port 5275 ^
  -input_username reader -input_password password ^
  -collection_filter medicine
  -output_host dest.example.com -output_port 9987 ^
  -output_username writer -output_password password ^
  -copy_permissions false ^
  -output_collections biomedicine,health
```

Slide 25

In this example, mlcp copy copies selected documents, excluding the source permissions and adding the documents to 2 new collections in the destination database. Although the example shows different users for the two MarkLogic Server instances the user can be the same for the source and destination databases adequate privileges are provided.



mlcp Export

- Export all database content or subset of data as:
  - Files in their original format
  - Compressed ZIP files
  - MarkLogic database archive

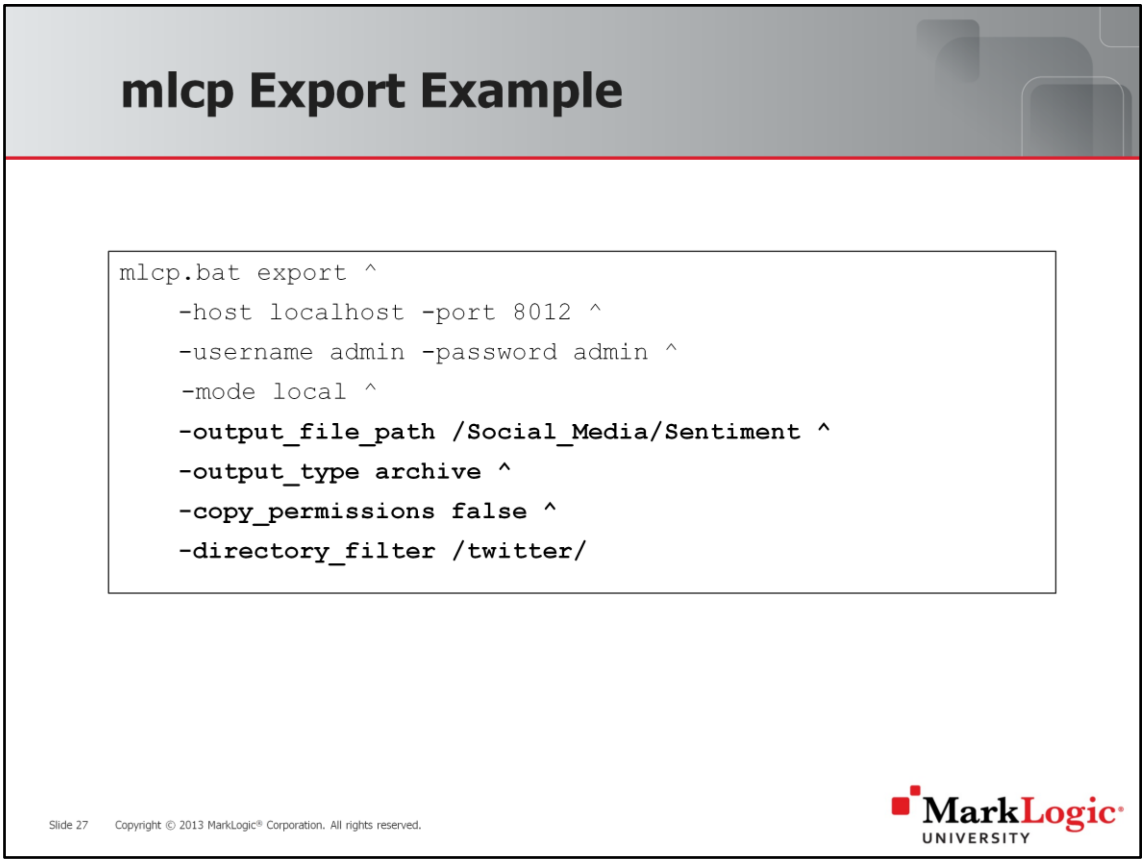
Slide 26 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

When you use the mlcp export command, you can export the entire contents of a MarkLogic Server database or a subset of data based on a collection or directory filter. The minimum commands that are required to export content as files (in their original format) are the MarkLogic Server instance, user, and -*output_file_path* to write the output to the native file system or HDFS.

To export as compressed ZIP files, you must also set -*compress* to true.

To export to a platform-independent database archive, you set -*output_type* to archive .

The MarkLogic database archive by default includes both the content and metadata.

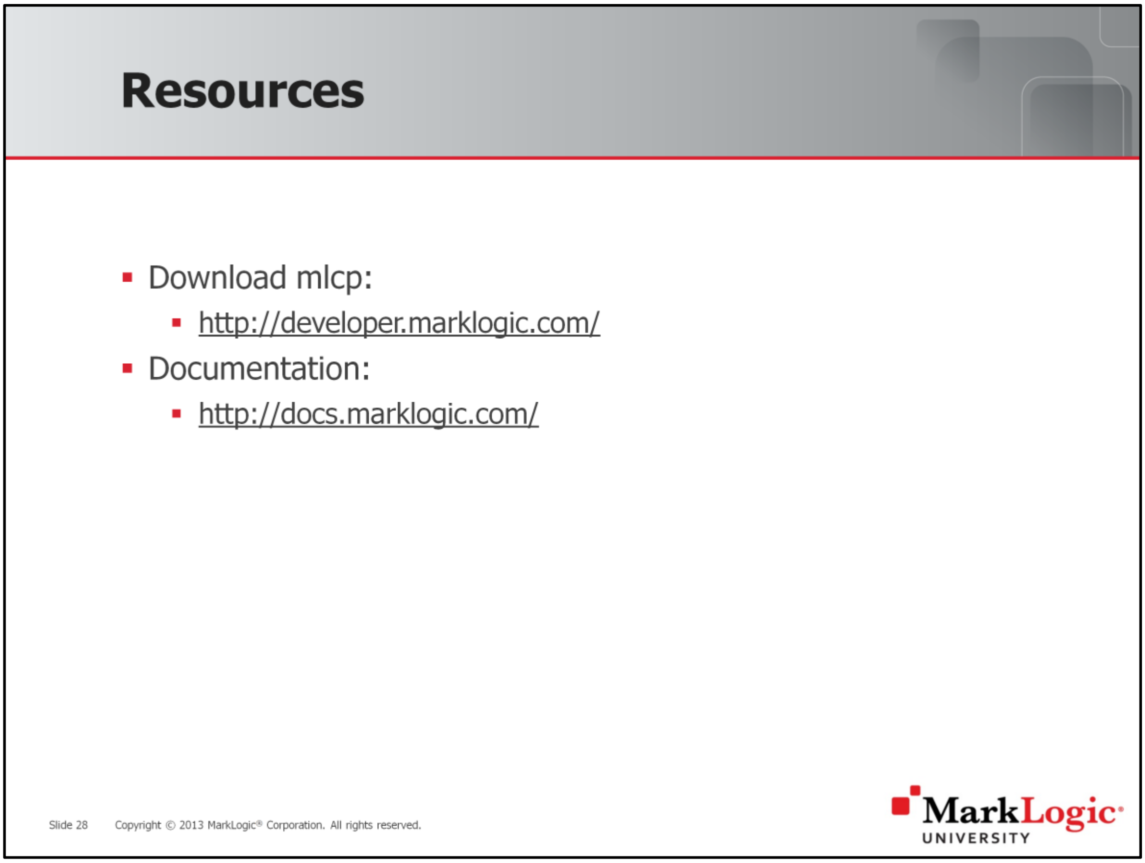


mlcp Export Example

```
mlcp.bat export ^
    -host localhost -port 8012 ^
    -username admin -password admin ^
    -mode local ^
    -output_file_path /Social_Media/Sentiment ^
    -output_type archive ^
    -copy_permissions false ^
    -directory_filter /twitter/
```

Slide 27 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

Just as with importing and copying, there are a number of options for exporting. You can export all documents and metadata to a local directory or exclude some or all document metadata from the archive. In the example, mlcp export creates an archive and exports only documents in the database directory "twitter", including their collections, properties, and quality, but excluding permissions.
The -*output_file_path* specifies the destination file or directory on the native filesystem or HDFS. When the export completes, the output directory contains one or more compressed archive files.



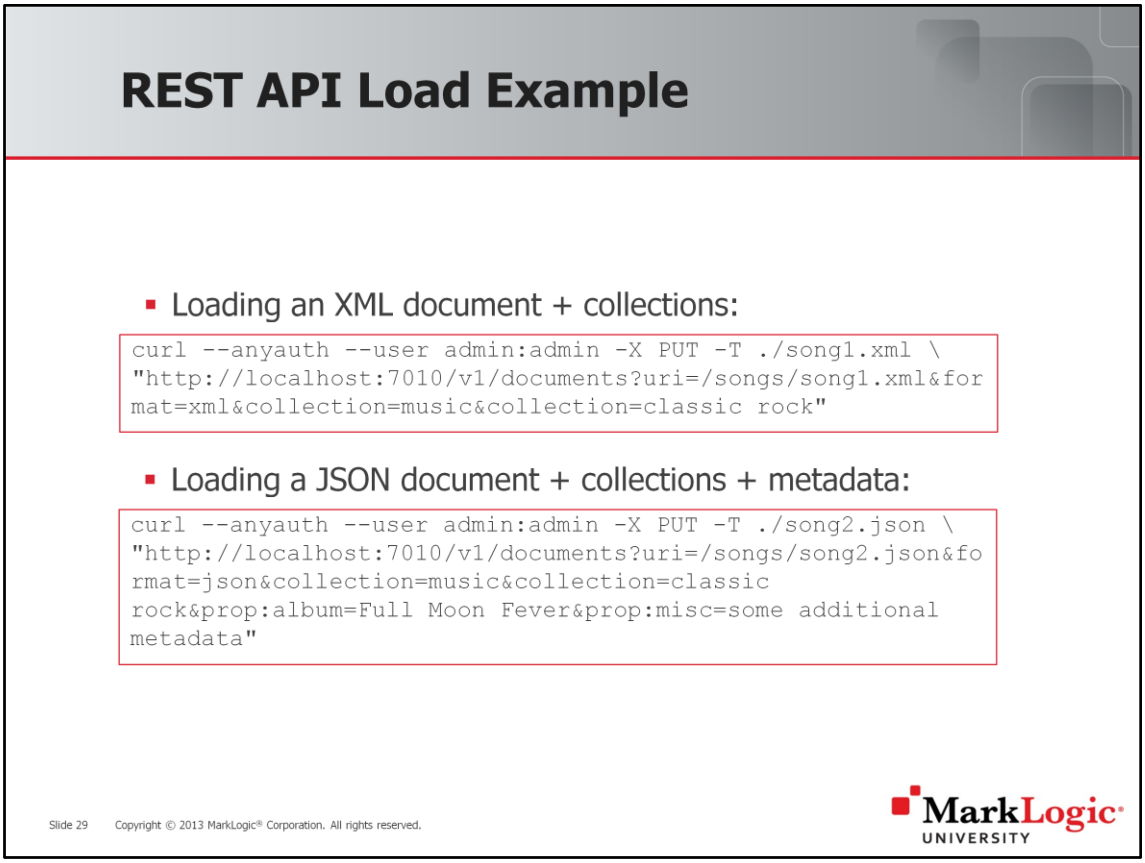
Resources

- Download mlcp:
  - http://developer.marklogic.com/
- Documentation:
  - http://docs.marklogic.com/

Slide 28 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

MarkLogic UNIVERSITY

Since mlcp is a java-based tool, you first install JDK 1.6 or later which is supported with MarkLogic Server 6.0-2 or later. Oracle's JDK is recommended, particularly if you plan to use Apache Hadoop.
You can download mlcp from http://developer.marklogic.com/products/mlcp and extract the contents to any directly on your local file system.
All the command options are described in the online documentation which can be downloaded from http://docs.marklogic.com/guide/ingestion/content-pump.
Additional installation and configuration steps are required if your planned mode of operation is distributed. Please see the "Configuring Distributed Mode" section in the documentation for more information about using Apache Hadoop or HDP installations with mlcp.



REST API Load Example

- Loading an XML document + collections:

```
curl --anyauth --user admin:admin -X PUT -T ./song1.xml \
"http://localhost:7010/v1/documents?uri=/songs/song1.xml&for
mat=xml&collection=music&collection=classic rock"
```

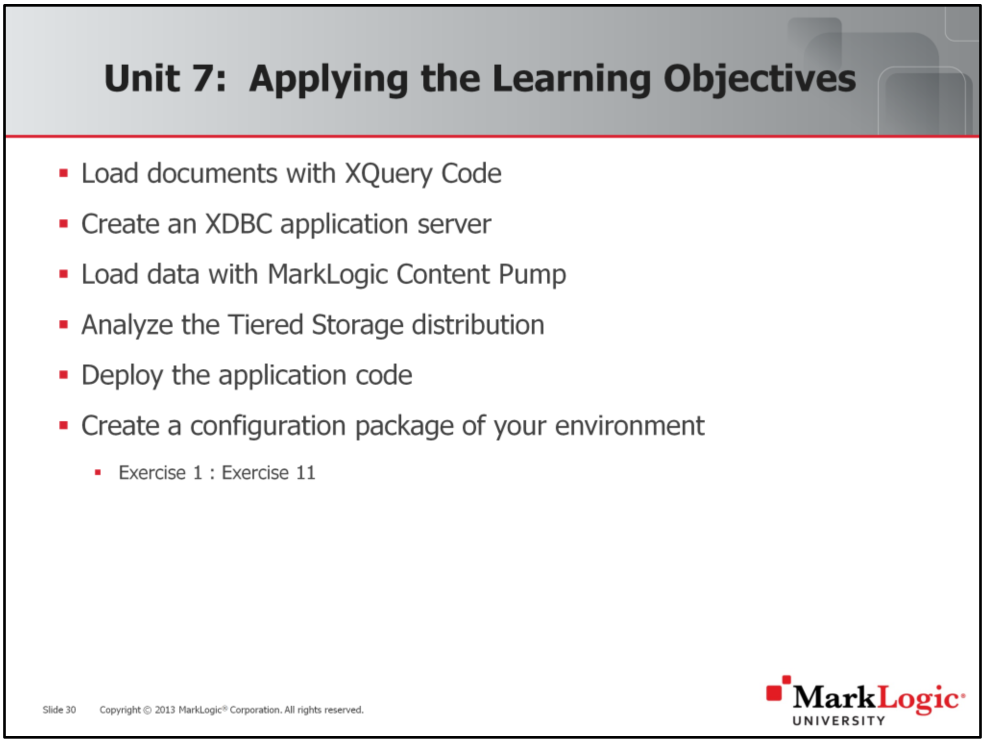
- Loading a JSON document + collections + metadata:

```
curl --anyauth --user admin:admin -X PUT -T ./song2.json \
"http://localhost:7010/v1/documents?uri=/songs/song2.json&fo
rmat=json&collection=music&collection=classic
rock&prop:album=Full Moon Fever&prop:misc=some additional
metadata"
```

Slide 29 Copyright © 2013 MarkLogic® Corporation. All rights reserved.

MarkLogic UNIVERSITY

- Loading a document (CREATE) uses the /v1/documents/ service and requires that you specify a document URI.
  - If the document URI already exists in the database, it becomes an UPDATE transaction.
  - Optional parameters include security permissions on the document, collections, metadata, etc. For full description of available optional parameters, please reference the product documentation.



Unit 7: Applying the Learning Objectives

- Load documents with XQuery Code
- Create an XDBC application server
- Load data with MarkLogic Content Pump
- Analyze the Tiered Storage distribution
- Deploy the application code
- Create a configuration package of your environment
  - Exercise 1 : Exercise 11