UC3Merritt

University of California Curation Center

Merritt Inventory Service

Rev. 1.00 – 2012-11-27

1 Introduction

Information technology and resources have become integral and indispensable to the pedagogic mission of the University of California. Members of the UC community routinely produce and utilize a wide variety of digital assets in the course of teaching, learning, and research. These assets represent the intellectual capital of the University; they have inherent enduring value and need to be managed carefully to ensure that they will remain available for use by future scholars. Within the UC system the UC Curation Center (UC3) of the California Digital Library (CDL) has a broad mandate to ensure the long-term usability of the digital assets of the University. UC3 views its mission in terms of *digital curation*, the set of policies and practices aimed at maintaining and adding value to authentic digital assets for use by scholars now and into the indefinite future [Abbott].

In order to meet these obligations UC3 is developing Merritt, an emergent approach to digital curation infrastructure [Merritt]. Merritt devolves infrastructure function into a growing set of granular, orthogonal, but interoperable micro-services embodying curation values and strategies [Foundations]. Since each of the services is small and self-contained, they are collectively easier to develop, deploy, maintain and enhance [Denning]; equally as important, since the level of investment in and commitment to any given service is small, they are more easily replaced when they have outlived their usefulness. Yet at the same time, complex curation functionality can emerge from the strategic combination of individual, atomistic services [Fisher].

The Merritt Inventory micro-service provides a comprehensive catalog of information known about the objects, and their subsidiary versions, and aggregated collections and owners, managed in the repository. The initial implementation of the Inventory service is based on the 4store semantic database. Unfortunately, 4store does not exhibit appropriate operational stability or performance. To address these concerns, the Inventory service will be re-implemented using proven relational database technology.

2 Schema

[*See attached diagram and SQL DDL*]

3 Queries

The following queries are necessary to support the current function of the Merritt UI.

3.1 Welcome page (merritt.cdlib.org/home/choose_collection)

The welcome page displays a list of all collections to which the user has read, write, or download privileges.

This page is dependent on an LDAP query that returns the collection names, user privileges, and ARK identifiers. The equivalent, but unnecessary, SQL query is:

```
SELECT name, mnemonic, ark
  FROM collections
 ORDER BY name ASC;
```

It may be useful to cache the selected collection ARK (*collection-ark* = collections.ark) and primary key (*collection-id* = collections.id) in the user session state.

3.2 Collection landing page (merritt.cdlib.org/m/*collection*)

The collection landing page displays a paged list of collection objects by ARK, creator, and title, along with summary information on the collection's total number of objects, versions, and files, and total size. It is assumed that the collection ARK is known (*collection-ark* = collections.ark).

3.2.1 List of all collection objects

```
SELECT o.ark, erc_who, erc_what
  FROM collections_objects co
  JOIN collections c ON c.id = co.collection_id
  JOIN objects     o ON o.id = co.object_id
 WHERE c.ark = 'collection-ark'
 ORDER BY o.created DESC;
```

3.2.2 Total number of objects

```
SELECT count(*)
  FROM collections_objects co
  JOIN collections c ON c.id = co.collection_id
  JOIN objects     o ON o.id = co.object_id
 WHERE co.ark = 'collection-ark';
```

Note that this can be simplified if the collection primary key (*collection_id* = collections.id) is cached.

```
SELECT count(*)
  FROM collections_objects co
  JOIN objects     o ON o.id = co.object_id
 WHERE co.collection_id = collection-id;
```

3.2.3 Total number of versions

```
SELECT count(*)
  FROM collections_objects co
  JOIN collections c ON c.id        = co.collection_id
  JOIN versions    v ON v.object_id = co.object_id
 WHERE co.ark = 'collection-ark';
```

or

```
SELECT count(*)
  FROM collections_objects co
  JOIN versions    v ON v.object_id = co.object_id
 WHERE co.collection_id = collection-id;
```

3.2.4 Total number of files and full and billable size

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM collections_objects co
  JOIN collections c ON c.id        = co.collection_id
  JOIN files       f ON f.object_id = co.object_id
 WHERE c.ark = 'collection-ark';
```

or

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM collections_objects co
  JOIN files       f ON f.object_id = co.object_id
 WHERE co.collection_id = 'collection-id';
```

3.3 Object landing page (merritt.cdlib.org/m/*object*)

The object landing page displays ERC metadata with additional administrative properties: creation and modification time, total size, and number of versions; and an enumeration of all versions with the files of the current version. It is assumed that the object ARK is known (*object-ark* = objects.ark).

It may be useful to cache the object ARK and primary identifier (*object-id* = objects.id) in the user session state.

3.3.1 ERC metadata

The object creator, title, date, primary identifier, and local identifier are specified by the Dublin Kernel "who", "what", "when", and "where" elements.

```
SELECT element, value
  FROM objects o
  JOIN versions      v ON v.object_id  = o.id
  JOIN dublinkernels k ON k.object_id  = o.id
                      AND k.version_id = v.id
 WHERE o.ark    = 'object-ark'
   AND v.number = o.version_number
 ORDER BY seq_num;
```

or

```
SELECT element, value
  FROM objects o
  JOIN versions      v ON v.object_id  = o.id
  JOIN dublinkernels k ON k.object_id  = o.id
                      AND k.version_id = v.id
 WHERE o.id     = 'object-id'
   AND v.number = o.version_number
 ORDER BY seq_num;
```

3.3.2 Administrative metadata

The object creation and modification times and number of versions are retrieved directly from the "objects" table.

```
SELECT created, modified, version_number
  FROM objects o
 WHERE o.ark = 'object-ark';
```

or

```
SELECT created, modified, version_number
  FROM objects o
 WHERE o.id = object-id;
```

3.3.3 Number and size of files

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM objects o
  JOIN files   f ON f.object_id = o.id
 WHERE o.ark = 'object-ark';
```

or

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM objects o
  JOIN files   f ON f.object_id = o.id
 WHERE o.id = object-id;
```

3.3.4 List of all versions and creation timestamps

```
SELECT number, v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.ark = 'object-ark'
 ORDER BY number ASC;
```

or

```
SELECT number, v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.id = object-id
 ORDER BY number ASC;
```

3.3.5 Files in current version

```
SELECT pathname
  FROM objects o
  JOIN versions v ON v.object_id = o.id
  JOIN files    f ON f.object_id = o.id
 WHERE o.ark    = 'object-ark'
   AND v.number = o.version-number
   AND source   = 'producer'
 ORDER BY pathname ASC;
```

or

```
SELECT pathname
  FROM objects o
  JOIN versions v ON v.object_id = o.id
  JOIN files    f ON f.object_id = o.id
 WHERE o.id     = object-id
   AND v.number = o.version-number
   AND source   = 'producer'
 ORDER BY pathname ASC;
```

3.3.6 Additional metadata

Although currently not supported in the 4store implementation, the new table model provides the opportunity for additional metadata to be directly managed by the Inventory service.

```
SELECT value, md_schema, version, serialization
  FROM objects o
  JOIN versions  v ON v.object_id = o.id
  JOIN metadatas m ON m.object_id = o.id
 WHERE o.ark    = 'object-ark'
   AND v.number = o.version_number
 ORDER BY md_schema ASC;
```

or

```
SELECT value, md_schema, version, serialization
  FROM objects o
  JOIN versions  v ON v.object_id = o.id
  JOIN metadatas m ON m.object_id = o.id
 WHERE o.id     = 'object-id'
   AND v.number = o.version_number
 ORDER BY md_schema ASC;
```

3.4 Version landing page (merritt.cdlib.org/m/*object*/*version*)

The version landing page displays (object) ERC metadata with additional administrative properties: version number and creation time, total size, and number of files; an enumeration of all version files with MIME type and (full) size; and an enumeration of all versions. It is assumed that the object ARK and version number are known.

It may be useful to cache the version identifier (*version-id* = versions.id) in the user session state.

3.4.1 (Object) ERC metadata

```
SELECT element, value
  FROM objects o
  JOIN versions      v ON v.object_id  = o.id
  JOIN dublinkernels k ON k.object_id  = o.id
                      AND k.version_id = v.id
 WHERE o.ark    = 'object-ark'
   AND v.number = version-num
 ORDER BY seq_num;
```

or

```
SELECT element, value
  FROM objects o
  JOIN versions      v ON v.object_id  = o.id
  JOIN dublinkernels k ON k.object_id  = o.id
                      AND k.version_id = v.id
 WHERE o.id = object-id
   AND v.id = version-id
 ORDER BY seq_num;
```

3.4.2 Administrative metadata

```
SELECT v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.ark    = 'object-ark'
   AND v.number = version-num;
```

or

```
SELECT v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.id = object-id
   AND v.id = version-id;
```

3.4.3 Number and size of files

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.ark    = 'object-ark'
   AND v.number = version-num;
```

or

```
SELECT count(*), sum(full_size), sum(billable_size)
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.id = object-id
   AND v.id = version-id;
```

3.4.4 List of files and MIME types and size

```
SELECT source, pathname, mime_type, full_size
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.ark    = 'object-ark'
   AND v.number = version-num
 ORDER BY source ASC, pathname ASC;
```

or

```
SELECT source, pathname, mime_type, full_size
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.id = object-id
   AND v.id = version-id
 ORDER BY source ASC, pathname ASC;
```

3.4.5 List of all versions and creation timestamps

```
SELECT number, v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.ark = 'object-ark'
 ORDER BY number ASC;
```

or

```
SELECT number, v.created
  FROM objects o
  JOIN versions v ON v.object_id = o.id
 WHERE o.id = object-id
 ORDER BY number ASC;
```

3.4.6 File download information

To initiate a file download, the Merritt UI needs to construct the appropriate Storage service URL based on the storage node, object identifier (or ARK), version number, and pathname.

```
SELECT node_number, ark, version_number, pathname
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.ark      = 'object-ark'
   AND v.number   = version-num
   AND f.pathname = 'pathname';
```

or

```
SELECT node_number, ark, version_number, pathname
  FROM objects o
  JOIN versions v ON v.object_id  = o.id
  JOIN files    f ON f.object_id  = o.id
                 AND f.version_id = v.id
 WHERE o.aid      = 'object-id'
   AND v.id       = version-id
   AND f.pathname = 'pathname';
```

3.5 Data use agreements

It is assumed that the object ARK is known.

3.5.1 Object-level DUA

```
SELECT title, terms, template, accept_obligation, name_obligation,
       affiliation_obligation, email_obligation, applicability,
       persistence, notification
  FROM objects o
  JOIN duas    d ON d.object_id = o.id
 WHERE o.ark = 'object-ark';
```

or

```
SELECT title, terms, template, accept_obligation, name_obligation,
       affiliation_obligation, email_obligation, applicability,
       persistence, notification
  FROM objects o
  JOIN duas    d ON d.object_id = o.id
 WHERE o.ark = object-id;
```

3.5.2 Collection-level DUA

```
SELECT title, terms, template, accept_obligation, name_obligation,
       affiliation_obligation, email_obligation, applicability,
       persistence, notification
  FROM collections_objects co
  JOIN objects o ON o.id            = co.object_id
  JOIN duas    d ON d.collection_id = co.collection_id
 WHERE o.ark = 'object-ark';
```

or

```
SELECT title, terms, template, accept_obligation, name_obligation,
       affiliation_obligation, email_obligation, applicability,
       persistence, notification
  FROM collections_objects co
  JOIN objects o ON o.id            = co.object_id
  JOIN duas    d ON d.collection_id = co.collection_id
 WHERE o.id = object-id;
```

4 Population

The Inventory database SHALL be populated through two mechanisms:

1. Retrospectively, by requesting relevant object information from the Merritt Storage micro-service.
2. Prospectively, by receiving relevant object information from the Merritt Ingest micro-service.

4.1 Retrospective

The various tables of the Inventory relational model can be populated with information from the following object and version files:

1. Object collection membership: system/mrt-membership.txt
2. Object data use agreement: producer/mrt-dua.txt
3. Object DataCite metadata: producer/mrt-datacite.xml
4. Object Dublin Core metadata: producer/mrt-dc.xml
5. Object Dublin Kernel metadata: system/mrt-erc.txt
6. Object EML metadata: producer/mrt-eml.xml
7. Object owner: system/mrt-owner.txt
8. Version Ingest metadata: system/mrt-ingest.txt
9. File: manifest.txt

Additional metadata may be specified in the future:

10. Object FGDC CSDGM metadata: producer/mrt-csdgm.xml

Note that the manifest will need to be augmented with file-level MIME type.

4.2 Prospective

References

[Abbott] Daisy Abbott, *What is Digital Curation?* April 3, 2008 <http://www.dcc.ac.uk/resource/briefing-papers/what-is-digital-curation/>.

[ARK] J. Kunze and R. Rodgers, *The ARK Identifer Schema*, May 22, 2008.

[Denning] Peter J. Denning, Chris Gunderson, and Rich Hayes-Roth, "Evolutionary system development," *Communications of the ACM* 51:17 (December 2008): 29-31.

[Fisher] David A. Fisher, *An Emergent Perspective on Interoperation in Systems of Systems*, CMU/SEI-2006-TR-003, ESC-TR-2006-003, March 2006 <http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr003.pdf>.

[Ingest] UC Curation Center, *Merritt Ingest Service*, 2012.

[Merritt] UC3, *Merritt: An Emergent Approach to Digital Curation Infrastructure*, 2010.

[MOM] UC Curation Center, *Merritt Object Modeling*, 2012.

[MySQL] Oracle Corp., *MySQL 5.1 Reference Manual*, 2012 <http://dev.mysql.com/doc/refman/5.1/en/>.

[SQL] C. J. Date, *A Guide to the SQL Standard* (2nd ed.; Reading: Addison-Wesley, 1989).