Ch 06c: Omeka-S to Heurist

Omeka S is a configurable database (there is an older version Omeka Classic). It is much more complex to set up and much more limited, although it does have some functions in the semantic web area which we don't yet address and extensive tech documentation, having been defined from scratch after a decade of Omeka Classic, and is therefore easier for programmers to extend with add-on modules. There is also an Omeka (either version) to Datacrate conversion and Heurist to Datacrate conversion developed in Python by Peter Sefton at UTS - you can find Datacrate on github - which might form the basis for an alternative pathway.

Please note that the migration from Omeka S to Heurist was developed before 2020 and may not operate 'out of the box'/

Converting from Omeka S to Heurist

The following table shows the correspondences between structures defined in Omeka S and structures defined in Heurist:

Omeka S	Heurist
Resource_class	defRecTypes
Resource_template_property	defRecTypeStructure (order, altlabel, requirements and data_type?)
Property	defDetailTypes
Resource	Records
Value	recDetails

Conversion

1. Since data_type is not defined in Resource_template_property (it was empty in def19 databases), it is necessary to detect type for every property.
   - ++Resources++: where value.value_resource_id IS NOT NULL
   - ++Terms++: look at tables with the same name as property and number of distinct values <100
   - ++Blocktext++: where number of long values is considerable length(value.value)>100
2. Get all properties in use

```
SELECT p.id,  p.local_name, count(\*) FROM value v, property p

where v.property_id=p.id group by p.id,  p.local_name order by p.id
```

3. Get properties in use by record class

```
SELECT distinct r.resource_class_id, p.id,  p.local_name FROM value v,
property p, resource r  where v.resource_id = r.id  and
v.property_id=p.id
```

4. Order by r.resource_class_id, p.id
5. As a result, you need to create following CSV tables.

For terms

- Property id: list of enum properties uses the same vocabulary
- Table name: takes terms from this table, don't worry if value is missed in this table it will be added to target vocabulary
- Vocab name: name of vocabulary to be added to heurist
- Resource class ID: check properties for these class only. (in OMEKA some fields are inconsistent for its types for different classes)

Property ID	Table Name	Vocab Name	Resource Class ID
202	fonctions	fonctions	155
"223,245,325"	pays	pays	
283	causes-fin-brevets	brevet cause fin	
291	genres	genres	
329	types-adresses	types de adresses	
346	typesdeproces	types de proces	
"290,383"	roles	roles	

For all fields:

```
$config = <<<'EOD'
```

rty	id	local_name	dty_Type	dty_ID	ptr/vocab Explanation
		7	date	date	9
		252	birthdate	date	
		35	isReferencedBy	blocktext	
		131	nick	freetext	
95,110,111	143	surname	freetext	1	map property 143 to heurist 1 for classes 95..
150	143	surname	resource	16	map property 143 to heurist 16 for class 150
		230	parrain	resource	95
		125	gender	enum	20
		202	agent	enum	6255

Classes by records

```
SELECT resource.resource_class_id, rc.local_name,count(\*) FROM
resource, resource_class rc

where  resource_class_id=rc.id group by
resource.resource_class_id,rc.local_name
```

Conversion notes (for developers)

I will do mapping their ResourceClass/Property to Heurist Rectypes/Fields
Enumeration types are vague in their system. If some of properties have table of the same name (for example property genre has table genres this property considered enumerated)

Import Resource/Values to Records/recDetails

DEFINITIONS: Map existing Heurist record types/fields to Omeka resource classes/properties.Omeka database does not keep any information about its database definitions just two tables that refers to resource/properties of RDF models (url of xml that describes these models are in Vocabulary table).
Example:
Resource class Agent (id 95, vocab_id=4) refers to Agent in http://xmlns.com/foaf/0.1/

Property Genre (vocab #6) refers to http://dbpedia.org/ontology/genre

Manual matching Omeka->Heurist: Resource class->Rectypes Property->Field type

Store RDF name (like foaf:Person OR dbo:Genre) in some field of defRectype, defDetailTypes tables OR keep matching in external file Omeka ID->Heurist ID, or RDF name->Heurist concept code I believe it is much cleaner to store such data in the database, this then allows us to use it directly in a future RDF export. Every time we use files we end up with problems eg. of synchronisation, referential integrity etc.

DATA: Import Omeka resource/value tables into Heurist Records/recDetails

---

Revision #5

Created 2026-07-01 12:10:28 UTC by IanJohnson

Updated 2026-07-04 12:34:02 UTC by IanJohnson