Study Unit 6

Semantic Data

Semantic Data Outline

- Semantic Data
- Semantic Web and Linked data
- Semantics Modelling and Ontologies
- eCRF as a FAIR Tool
- CEDAR as a FAIR Tool

Study Unit Duration

This Study Session requires a 3-4 hours of formal study time.

You may spend an additional 2-3 hours for revision

Introduction

This Study Unit covers the basic concepts of semantic web, linked data, the semantic web stack and technologies like SKOS, RDF, OWL and SPARQL. It also explains sematic modelling and compare it with other data models. Other topics covered includes how to use eCRF and CEDAR to create and explore metadata and how to use them as FAIR tool.

Learning Outcomes of Study Unit 6

Upon completion of this study unit, you should be able to:

6.1 Describe the Semantic Web, it's goals and benefit

6.2 Explain Semantic Web basic building blocks such as RDF, SKOS, OWL etc.

6.3 Describe the concept of structure of Linked Data

6.4 Explain the concept of Semantic Modelling, Ontology and data models.

6.5 Use the eCRF Wizard to create and explore metadata

6.6 Use the CEDAR workbench to create and explore metadata.

6.1 Introduction to Semantic Web

The World Wide Web was invented because Tim Berners-Lee, a scientist at the CERN nuclear research laboratory in Switzerland, wanted a way for him and his colleagues to share documents over the Internet. The Semantic Web is a collaborative movement led by international standards body the World Wide Web Consortium (W3C). The standard promotes common data formats on the World Wide Web. By encouraging the inclusion of semantic content in web pages, the Semantic Web aims at converting the current "web of documents", dominated by unstructured and semi-structured documents into a "web of data". The Semantic Web provides meaning to the information contained in Web documents. The Semantic Web also adds the linking data and documents to the original web. Semantic Web describes and link web content in a manner that's meaningful to machines. The link itself have a specific meaning. The many formal definitions of semantic web are presented in Box 6.1

Box 6.1: Definitions of Semantic Web

1. Semantic Web refers to World Wide Web Consortium (W3C's) vision of the Web of linked data.
2. Essentially, the semantic web marks a move from a global web of human readable documents (web pages), to a global web of machine-readable documents; so that machines can automatically interpret the meaning of data on the web; what to do with it, and how to represent it.
3. Principally, the Semantic Web is a Web 3.0 web technology. It is a way of linking data between systems or entities that allows for rich, self-describing interrelations of data available across the globe on the web.
4. The goal of the Semantic Web is to make Internet data machine-readable.

A Semantic Web can be built by linking information that exists within documents, and by allowing data itself to be on the Web as shown by Figure 6.1. Semantic Web technologies is popular in the library, scientific arena, in particular in biomedical research, and in the realm of government data.

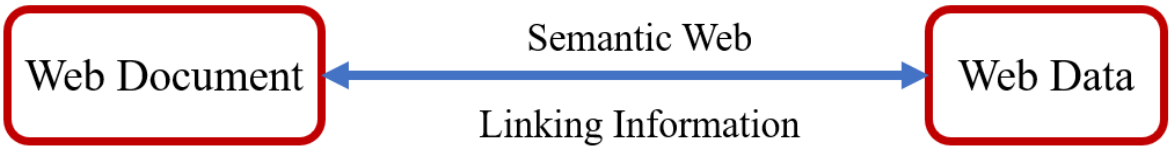


Figure 6.1 Semantic Web

6.1.1 Benefits of Semantic Web

By enabling machines to understand data, we can benefit in many ways:

- **Automation**: We can avoid doing mundane stuff such as booking tickets or rescheduling appointments. These can be efficiently handled by virtual assistants or agents.
- **Personalization**: Content on the web is growing daily. It's impossible for us to follow everything. Agents can personalize or curate content for us.
- **Information Retrieval**: Within enterprises or via web search engines, Semantic Web can give us more relevant answers.
- **Data Reuse**: Because Semantic Web enables linking of data from a variety of sources, data can be reused. Data that was previously stored in isolated databases can now be shared in a standard manner.
- **Knowledge Discovery**: By linking data across the web, new knowledge can be discovered. Semantic Web enables machines to apply logic on existing relationships and infer new ones. For example, this could be useful in discovering new drugs.

6.1.2 Semantic Web and Machine Learning

Since semantic web, Artificial Intelligence (AI) and Machine Learning (ML) uses data, it is important to compare the manner they use the data. It therefore does appear that AI/ML has gone ahead and enabled machines to see, hear and speak. The mid-2010s have seen the arrival of voice assistants, chatbots, computer vision applications, and more. This has been possible because of the availability of data to train ML algorithms. The comparison is illustrated in Figure 6.2 and Table 6.1.

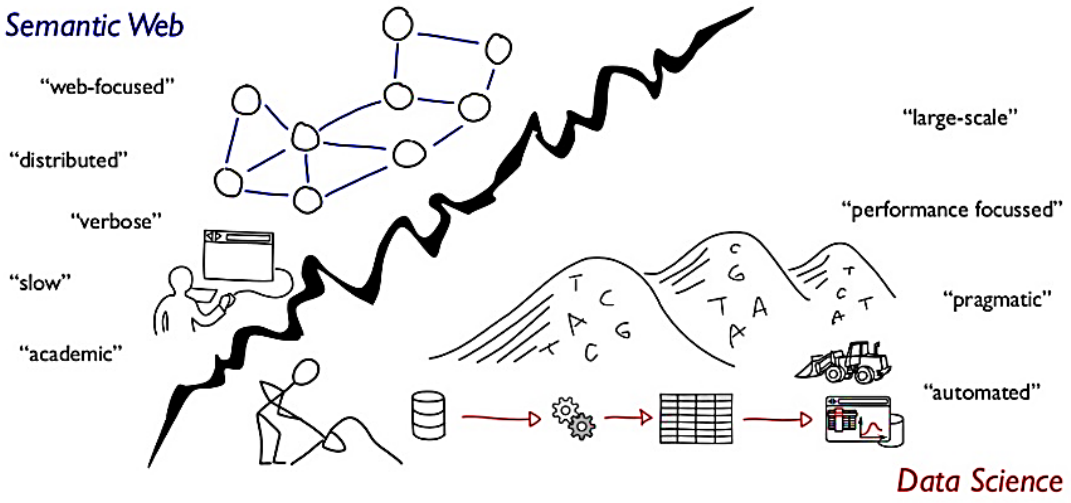


Figure 6.2 comparing Semantic Web and Data Science. Source [4]

Table 6.1 Comparing Semantic Web and Data Science

Semantic Web	Machine Learning
It aids machines to solve well-defined problems on well-defined data through well-defined operations.	Artificial Intelligence and Machine Learning makes machines intelligent. They ask machine to understand human.
Semantic web adds semantic metadata to all data.	Though, ML algorithms require data to be tagged or labelled for training, there is no need to add semantic metadata to all data in the manner of the Semantic Web.
However, Semantic Web is complementary to AI/ML approaches. Chatbots and intelligent assistants will use them.	
Semantic Web can add background knowledge to AI/ML systems, particularly in areas where data is scarce.	
AI/ML being applied to conceptualize domain knowledge for the Semantic Web.	

6.2 Semantic Web Basic Building Blocks

Semantic Web gives meaning to the information contained in Web documents. This description and its interpretation are supported by a stack of technologies that have been designed and recommended by the World Wide Web Consortium (W3C) since 1999. This stack is often called the Semantic Web stack or Semantic Web cake as presented by Figure 6.3. The Semantic Web Stack is an illustration of the hierarchy of languages, where each layer exploits and uses capabilities of the layers below. It shows how technologies that are standardized for Semantic Web are organized to make the Semantic Web possible. It also shows how Semantic Web is an extension (not replacement) of classical hypertext web.

The illustration was created by Tim Berners-Lee. The stack is still evolving as the layers are concretized.

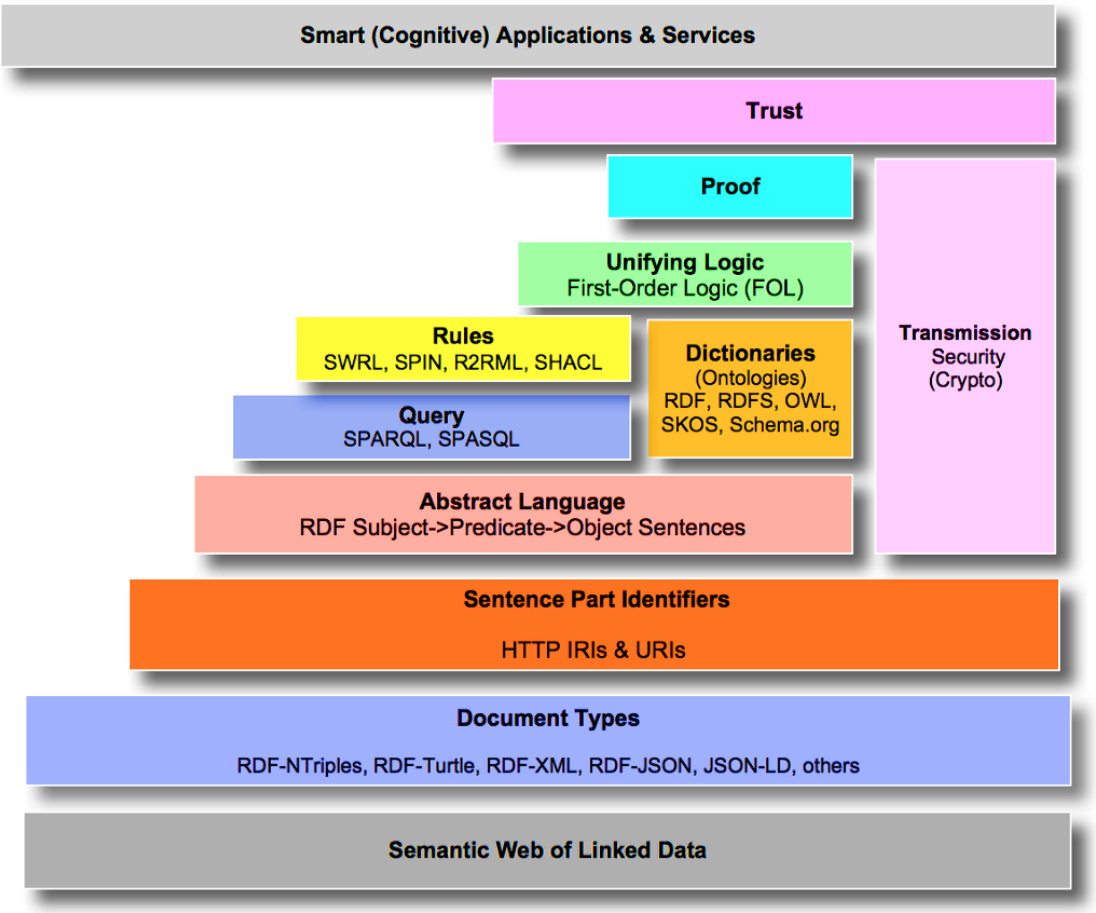


Figure 6.3 The Semantic Web Stack [5]

Semantic Web builds upon the foundations of the original web. Data, both old and new must be described with metadata. This metadata will identify data, interlink data and relate data to concepts so that machines can understand them. Data must be uniquely identified and this is done using Uniform Resource Identifier (URI) or Internationalized Resource Identifier (IRI). Resource Description Framework (RDF) provides the data model. Meaning is added at a higher layer with ontologies. In other words, RDF specifies the syntax while ontologies specify the semantics. Just as HTML was the building block of the original web, RDF is the building block of the Semantic Web. Web content can expose their semantics by embedding RDF statements within webpages. There are many ways to do this: RDFa, RDF-XML, RDF-JSON, JSON-LD, Microdata, etc. Semantic data already processed and stored in RDF format can be queried. Just as MySQL exists to query relational databases, SPARQL is a language to query RDF stores. Given the semantics, rules can help in applying logic and reasoning.

6.2.1 Semantic Web Technologies

As illustrated by the Semantic Web Stack, the following languages or technologies are used to create Semantic Web. The technologies from the bottom of the stack up to Web Ontology Language (OWL) are currently standardized and accepted to build Semantic Web applications. The scope of this course spans only the five bottom layers. Figure 6.3 presents the stack of Semantic Web Cake.

Table 6.2 Layers of the Semantic Web Cake

Three Layers	Layers	Components	Description
	Layer 1	IRIs / URIs	This layer provides a global identification solution for the resources found on the Web to allow provable manipulation with resources in the top layers.
		Unicode	serves to represent and manipulate text in many languages. Semantic

The bottom layers contain hypertext web technology and that without change provide basis for the semantic web.			Web should also help to bridge documents in different human languages, so it should be able to represent them
	Layer 2	XML XML Namespaces	This layer supports the definition of a syntax based on XML, is a mark-up language that enables creation of documents composed of semi-structured data. Semantic web gives meaning (semantics) to semi-structured data with his associated technologies.
Middle layers contain standardized semantic web technologies by W3C to enable building semantic web applications.	Layer 3	RDF	Semantic Web journey begins at this layer RDF language. It is a framework for creating statements in a form triple. It exchanges data among agents
	Layer 4	RDF Schema (RDFS)	This layer provides provides basic vocabulary for RDF. Using RDFS it is for example possible to create hierarchies of classes and properties.
	Layer 5	OWL	This layer extends RDFS by adding more advanced constructs to describe semantics of RDF statements. It allows stating additional constraints, such as for example cardinality, restrictions of values, or characteristics of properties such as transitivity. It is based on description logic and so

			brings reasoning power to the semantic web.
This is not a new layer but SPARQL and RIF spread across layers 4 and 5.		SPARQL	It is a RDF query language - it can be used to query any RDF-based data (i.e., including statements involving RDFS and OWL). Querying language is necessary to retrieve information for semantic web applications.
		RIF	It is a rule interchange format. It is important, for example, to allow describing relations that cannot be directly described using description logic used in OWL.
Top layers contain technologies that are not yet standardized or contain just ideas that should be implemented in order to realize Semantic Web.	Layers 6, 7 and 8 are beyond the scope of this course	Cryptography	is important to ensure and verify that semantic web statements are coming from trusted source. This can be achieved by appropriate digital signature of RDF statements.
		Trust	Trust to derived statements will be supported by (a) verifying that the premises come from trusted source and by (b) relying on formal logic during deriving new information.
		User interface	User interface is the final layer that will enable humans to use semantic web applications.

6.2.2 RDF and the Triple

The Resource Description Framework (RDF) is a formal language that defines the basic structure of the linked data that makes up the Semantic Web. It is a model for relationship (Hyperlink) and interchange like (*S, P, O*) triplet where *P* is the naming relationship between *S* and *O*. RDF is to the Semantic Web as data packets are to the Internet. Both provide a basic, underlying structure that services can be built upon. They both are designed for use by computer programs, not by humans. Simply put, RDF defines the basic unit of the Semantic Web as a three-part structure, commonly referred to as a triple. This structure is analogous to a very simple sentence, and each triple has this same set of subject, predicate, object components as shown in Figure 6.4:

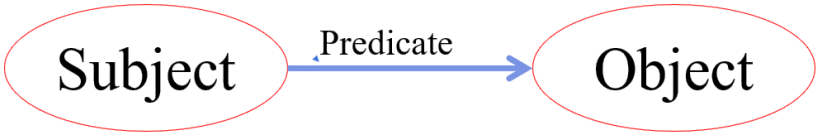


Figure 6.4 Basic RDF triple

The subject is what you are talking about, the object is what you are saying about it, and the predicate is a verb-like connector that states meaningfully what links the subject and object. While the structure is called a triple, a triple of information is often referred to as a statement because it states some information about the subject. Since RDF is a set of relationships, it can be presented as a graph. Its triple components (*S, P, O*) can then be presented as a directed labelled graph. So,

- a set of RDF statements is a directed, labeled graph as shown in Figure 6.4
  - the nodes represent the resources that are bound (Subject, Object)
  - the labelled edges are the relationships with their names (Predicate)

RDF can be presented in several common serialization formats as illustrated by Table 6.4.

Table 6.4. Serialization formats for RDF

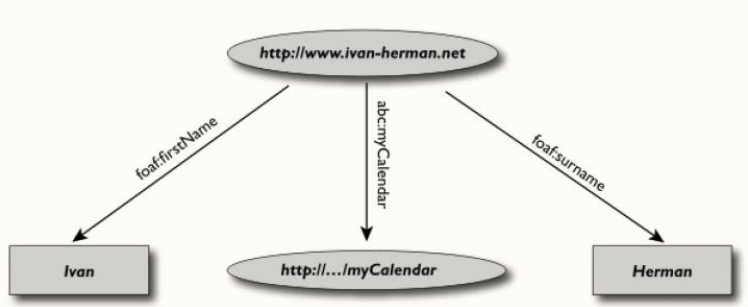
Format	**Description**
Turtle	a compact, human-friendly format.
N-Triples	a very simple, easy-to-parse, line-based format that is not as compact as Turtle.
N-Quads	a superset of N-Triples, for serializing multiple RDF graphs
JSON-LD	a JSON-based serialization.

N3 or Notation3	a non-standard serialization that is very similar to Turtle, but has some additional features, such as the ability to define inference rules.
RDF/XML	an XML-based syntax that was the first standard format for serializing RDF.
RDF/JSON	an alternative syntax for expressing RDF triples using a simple JSON notation.

Example 1: Ivan Herman's calendar

When someone says that data has been made available "in RDF," that is usually shorthand for saying that the data follows Semantic Web standards. You will often see references to RDF/XML. Data in that format uses the standard RDF schema, also shortened to "RDFs." As shown by Table 6.5.

Table 6.5: RDF/XML representation of Ivan Herman




```
<rdf:Description rdf:about="http://www.ivan-herman.net">
    <foaf:name>Ivan<</foaf:name>
    <abc:myCalendar rdf:resource=http://.../myCalendar/>
    <foaf:surname>Herman</foaf:surname>
</rdf:Description>
```

Example 2:

The following example is taken from the W3C website describing a resource with statements "**there is a Person identified by http://www.w3.org/People/EM/contact#me, whose name is Eric Miller, whose email address is em@w3.org and whose title is Dr.**" as represented as the RDF graph in Figure 6.5:

Figure 6.5: An RDF Graph describing Eric Miller

The resource "http://www.w3.org/People/EM/contact#me" is the subject.

The objects are:

- "Eric Miller" (with a predicate "whose name is"),
- mailto: em@w3.org (with a predicate "whose email address is"), and
- "Dr." (with a predicate "whose title is").

The subject is a URI.

The predicates also have URIs. For example, the URI for each predicate:

- "whose name is" is http://www.w3.org/2000/10/swap/pim/contact#fullName,
- "whose email address is" is http://www.w3.org/2000/10/swap/pim/contact#mailbox,
- "whose title is" is http://www.w3.org/2000/10/swap/pim/contact#personalTitle.

In addition, the subject has a type (with URI http://www.w3.org/1999/02/22-rdf-syntax-ns#type), which is person (with URI

http://www.w3.org/2000/10/swap/pim/contact#Person) .

This example is represented in three different formats shown in Tables 4.6, 6.6 and 4.8.

Therefore, the following "subject, predicate, object" RDF triples can be expressed:

Table 6.6: RDF triples of Eric Miller

```
<http://www.w3.org/People/EM/contact#me> <http://www.w3.org/2000/10/swap/pim/contact#fullName> "Eric Miller".

<http://www.w3.org/People/EM/contact#me> <http://www.w3.org/2000/10/swap/pim/contact#mailbox> <mailto: em@w3.org >.

<http://www.w3.org/People/EM/contact#me> <http://www.w3.org/2000/10/swap/pim/contact#personalTitle> "Dr.".

<http://www.w3.org/People/EM/contact#me> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://www.w3.org/2000/10/swap/pim/contact#Person>.
```

Table 6.7: Turtle format of Eric Miller

```
@prefix eric:    <http://www.w3.org/People/EM/contact#>.
@prefix contact: <http://www.w3.org/2000/10/swap/pim/contact#>.
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

eric:me contact:fullName "Eric Miller" .
eric:me contact:mailbox <mailto:e.miller123(at)example> .
eric:me contact:personalTitle "Dr." .
eric:me rdf:type contact:Person .
```

Table 6.8: RDF/XML Describing Eric Miller

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#"
xmlns:eric="http://www.w3.org/People/EM/contact#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
```

```
  </rdf:Description>
  <rdf:Description rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:mailbox rdf:resource="mailto:e.miller123(at)example"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:personalTitle>Dr.</contact:personalTitle>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.w3.org/People/EM/contact#me">
    <rdf:type rdf:resource="http://www.w3.org/2000/10/swap/pim/contact#Person"/>
  </rdf:Description>
</rdf:RDF>
```

Example 3: A Wikipedia article about Tony Benn

Given that "http://en.wikipedia.org/wiki/Tony_Benn" identifies a particular resource to say that the title of this resource is "Tony Benn" and its publisher is "Wikipedia" would be two assertions that could be expressed as valid RDF statements.

Table 6.9 N-Triples form of RDF of Tony Benn

```
<http://en.wikipedia.org/wiki/Tony_Benn> <http://purl.org/dc/elements/1.1/title> "Tony Benn" .

<http://en.wikipedia.org/wiki/Tony_Benn> <http://purl.org/dc/elements/1.1/publisher> "Wikipedia" .
```

Table 6.10 English structure of Tony Benn

The title of this resource, which is published by Wikipedia, is 'Tony Benn'

Table 6.9 Turtle form of RDF of Tony Benn

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc:   <http://purl.org/dc/elements/1.1/> .
```

```
<http://en.wikipedia.org/wiki/Tony_Benn>
   dc:publisher "Wikipedia" ;
   dc:title "Tony Benn" ;
   foaf:primaryTopic [
      a foaf:Person ;
      foaf:name "Tony Benn"
   ] .
```

Uniform Resource Identifier (URI)-s are used as universal *naming* tools. URI-s can make merging possible during data integration, grounds RDF into the Web which make it Semantic Web as shown in Figure 6.6. Data A is colored blue while Data B is colored yellow

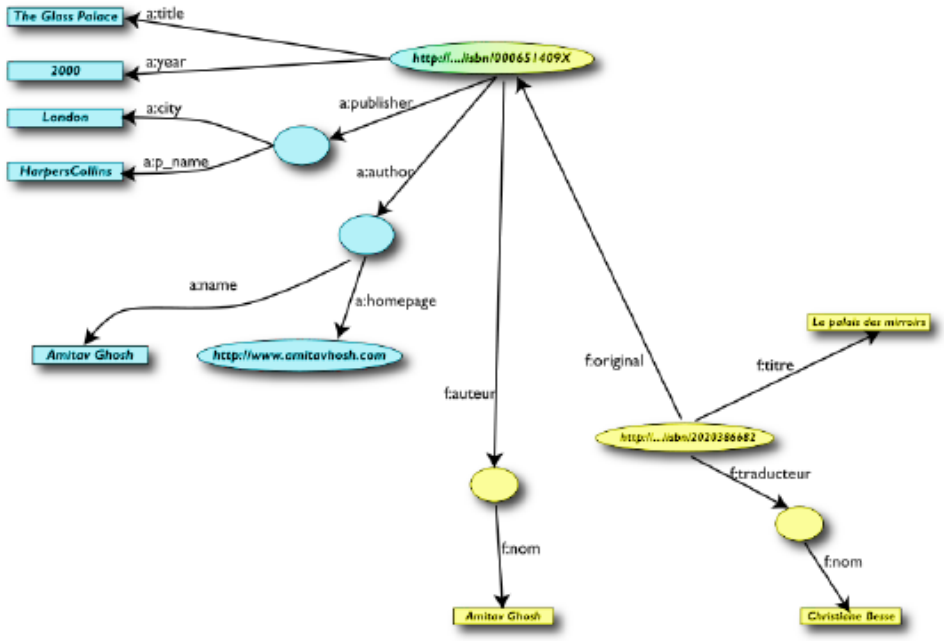


Figure 6.6 Simple data integration using RDF

6.2.3 SKOS

The Simple Knowledge Organization System (SKOS) is one of the first structures built on top of RDF. SKOS is a common data model for knowledge organization systems such as thesauri, classification schemes, subject heading systems and taxonomies. Using SKOS, a knowledge organization system can be expressed as machine-readable data. It can then be exchanged between computer applications and published in a machine-readable format in the Web.

SKOS data are expressed as RDF triples and may be encoded using any concrete RDF syntax (such as RDF/XML or Turtle. The SKOS data model views a knowledge organization system as a **concept scheme** comprising a set of **concepts**. These SKOS concept schemes and SKOS concepts are identified by URIs, enabling anyone to refer to them unambiguously from any context, and making them a part of the World Wide Web.

SKOS concepts can be **labeled** with any number of lexical (UNICODE) strings, such as "romantic love" or "れんあい", in any given natural language, such as English or Japanese (written here in hiragana). One of these labels in any given language can be indicated as the preferred label (shortened to prefLabel in encoded SKOS) for that language, and the others as alternative labels. Labels may also be "hidden", which is useful where a knowledge organization system is being queried via a text index. SKOS data are then expressed as RDF triples. Table 6.10 illustrates a RDF graph in TURTLE format that expresses some facts about a thesaurus.

Table 6.10 RDF graph in about a thesaurus.

```
<A> rdf:type skos:Concept ;
  skos:prefLabel "love"@en ;
  skos:altLabel "adoration"@en ;
  skos:broader <B> ;
  skos:inScheme <S> .

<B> rdf:type skos:Concept ;
  skos:prefLabel "emotion"@en ;
  skos:altLabel "feeling"@en ;
  skos:topConceptOf <S> .

<S> rdf:type skos:ConceptScheme ;
  dct:title "My First Thesaurus" ;
  skos:hasTopConcept <B> .
```

6.2.4 OWL

Ontology allows the *contextual relationships* behind a defined vocabulary to be defined. It is the cornerstone of defining a knowledge domain. Web Ontology Language (OWL) is a standard that extends RDF to RDFS (RDF Schema) and is formal syntax used to define specific Semantic Web metadata vocabularies (also called ontologies) as shown in Figure 6.7.

Figure 6.8 shows the subclass relationships between OWL and RDF/RDFS. Ontologies are a formal way to describe taxonomies and classification networks, essentially defining the structure of knowledge for various domains: the nouns representing classes of objects and the verbs representing relations between the objects. For example, if you wish to express your warehouse data as linked data, you would use OWL to explain in machine language what your metadata is and how it relates to other data in the Web of data. OWL is to be used by the developers of metadata formats for the Semantic Web; as such, it is quite complex. OWL has already been through its own development cycle and as a result exists in a small number of versions.

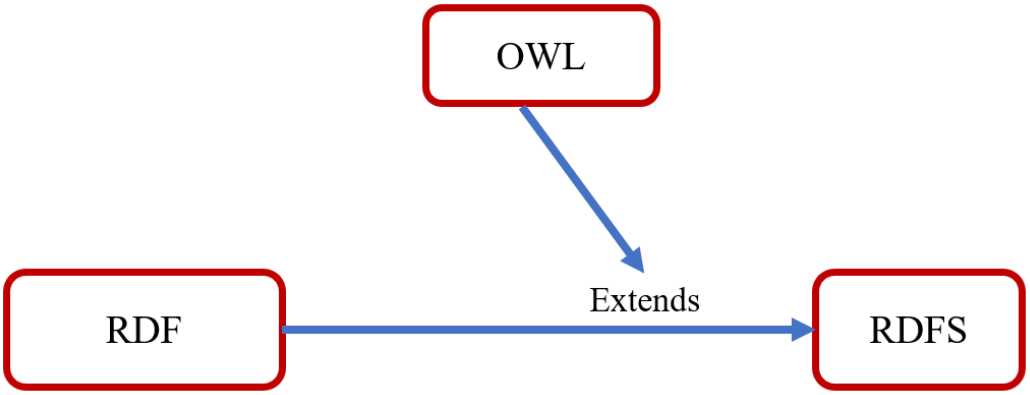


Figure 6.7: OWL

OWL ontologies consist of the following three different syntactic categories:

- *Entities*, such as classes, properties, and individuals, are identified by IRIs.
- *Expressions* represent complex notions in the domain being described.
- *Axioms* are statements that are asserted to be true in the domain being described.

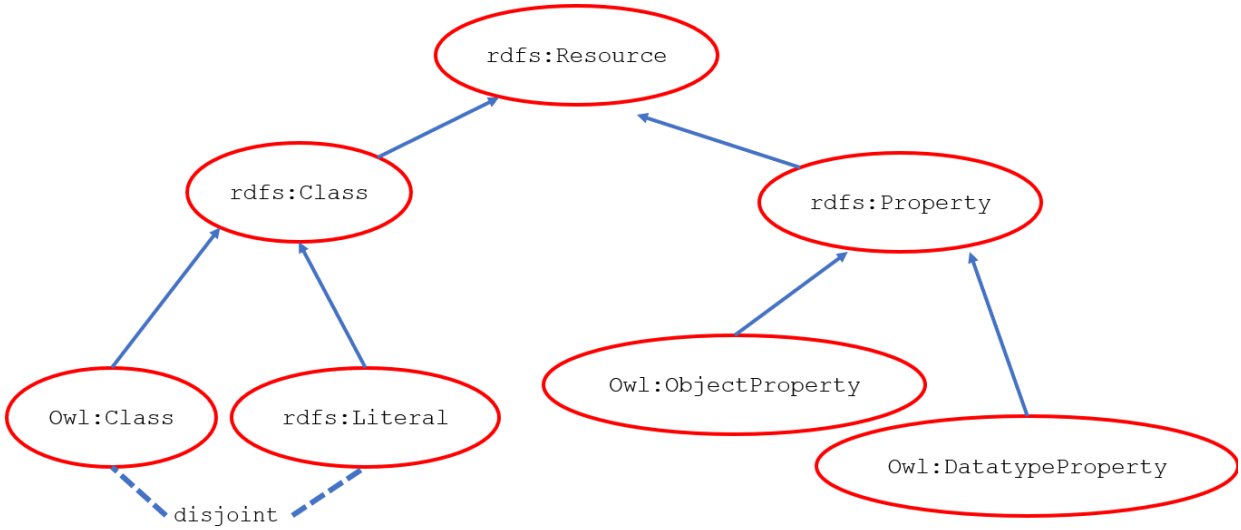


Figure 6.8: The subclass relationships between OWL and RDF/RDFS

6.2.5 SPARQL

One vision of the Semantic Web is that it is a huge web of data that uses the WWW as its database platform. In fact, it is expected that the Semantic Web will be queried much as a database is queried. A standard query language for that purpose, SPARQL (pronounced “sparkle”), is designed specifically to query the underlying triples of the Semantic Web using an SQL-like query format. Because SPARQL is designed to be run against Web resources, you must first point the SPARQL engine at a data graph/dataset. You can query on one, two, or all three elements of the triple. The query consists of two parts: the **SELECT** clause identifies the variables to appear in the query results (usually prefixed by **?**), and the **WHERE** clause provides the basic graph pattern to match against the data graph.

Example 1

The following example shows a SPARQL query to find the title of a book from the given data graph (RDF).

Table 6.11 Data to find the title of a book

/<http://example.org/book/book1> http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial”.

The basic graph pattern in this example consists of a single triple pattern with a single variable (?title) in the object position. A sample SPARQL query will be:

Table 6.12 SPARQL query to find the title of a book

```
SELECT ?title
WHERE
{
  <http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> ?title .
}
```

Table 6.13 SPARKLE query result

Title
"SPARQL Tutorial"

Example 2

The result of a query is a solution sequence, corresponding to the ways in which the query's graph pattern matches the data. There may be zero, one or multiple solutions to a query.

Table 6.14 Data solution sequence

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/>.

_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.org>
```

Table 6.15 SPRQLE query on solution sequence

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
```

```
WHERE
  { ?x foaf:name ?name .
    ?x foaf:mbox ?mbox }
```

Table 6.16 SPARQL result of the query

name	Mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

The link to a video for more information - Querying Wikidata with SPARQL for Absolute Beginners - YouTube

Example 3

This example demonstrates a simple query that leverages the ontology definition foaf Where foaf (friend of a friend) is a machine-readable ontology describing persons, their activities and their relations to other people and objects

Table 6.17 SPARQL query example that returns names and emails of every person in the dataset

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
      ?email
WHERE
  {
    ?person  a         foaf:Person .
    ?person  foaf:name  ?name .
    ?person  foaf:mbox  ?email .
  }
```

Example 4

Another SPARQL query example that models the question "What are all the country capitals in Africa?"

Table 6.18 SPARQL query on all the country capitals in Africa

```
PREFIX ex: <http://example.com/exampleOntology#>
SELECT ?capital
    ?country
WHERE
 {
  ?x  ex:cityname     ?capital  ;
     ex:isCapitalOf   ?y       .
  ?y  ex:countryname   ?country  ;
     ex:isInContinent  ex:Africa  .
 }
```

Example 5: This is an example of a DBpedia SPARQL query for finding all cities with a population exceeding 5 million

Table 6.19 SPARQL query for DBpedia

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

PREFIX dbpedia: <http://dbpedia.org/resource>

PREFIX dbpprop: <http://dbpedia.org/property>

SELECT DISTINCT ?citylabel ?countrylabel ?pop
```

```
WHERE {
   ?city rdf:type dbpedia-owl:City.
   ?city rdfs:label ?citylabel.
   ?city dbpedia-owl:country ?country.
   ?country rdfs:label ?countrylabel.
   ?city dbpedia-owl:populationTotal ?pop .
   FILTER ( LANG(?countrylabel) = 'en' and LANG(?citylabel) = 'en' and ?pop>5000000)
}
```

Peer to Peer Interaction

Answer the SPARQLE query in example 3, 4 and 5

6.2.6 RDF Schema (RDFS)

RDFS is a set of classes with certain properties using the RDF extensible knowledge representation data model, providing basic elements for the description of ontologies. It uses various forms of RDF vocabularies, intended to structure RDF resources. RDF and RDFS can be saved in a triplestore, then one can entail some knowledge from them using a query language, like SPARQL.

RDF Schema summary

The Tables 4.19 and 4.20 provide an overview of the RDF Schema vocabulary.

Table 6.19 RDF classes

Class name	Comment
rdfs:Resource	The class resource, everything.
rdfs:Literal	The class of literal values, e.g. textual strings and integers.
rdf:langString	The class of language-tagged string literal values.
rdf:HTML	The class of HTML literal values.
rdf:XMLLiteral	The class of XML literal values.
rdfs:Class	The class of classes.
rdf:Property	The class of RDF properties.
rdfs:Datatype	The class of RDF datatypes.
rdf:Statement	The class of RDF statements.
rdf:Bag	The class of unordered containers.
rdf:Seq	The class of ordered containers.
rdf:Alt	The class of containers of alternatives.
rdfs:Container	The class of RDF containers.
rdfs:ContainerMembershipProperty	The class of container membership properties, rdf:_1, rdf:_2, ..., all of which are sub-properties of 'member'.
rdf:List	The class of RDF Lists.

Table 6.20 RDF properties

Property name	Comment	domain	range
rdf:type	The subject is an instance of a class.	rdfs:Resource	rdfs:Class
rdfs:subClassOf	The subject is a subclass of a class.	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	The subject is a subproperty of a property.	rdf:Property	rdf:Property
rdfs:domain	A domain of the subject property.	rdf:Property	rdfs:Class
rdfs:range	A range of the subject property.	rdf:Property	rdfs:Class
rdfs:label	A human-readable name for the subject.	rdfs:Resource	rdfs:Literal

rdfs:comment	A description of the subject resource.	rdfs:Resource	rdfs:Literal
rdfs:member	A member of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:first	The first item in the subject RDF list.	rdf:List	rdfs:Resource
rdf:rest	The rest of the subject RDF list after the first item.	rdf:List	rdf:List
rdfs:seeAlso	Further information about the subject resource.	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	The definition of the subject resource.	rdfs:Resource	rdfs:Resource
rdf:value	Idiomatic property used for structured values.	rdfs:Resource	rdfs:Resource
rdf:subject	The subject of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:predicate	The predicate of the subject RDF statement.	rdf:Statement	rdfs:Resource
rdf:object	The object of the subject RDF statement.	rdf:Statement	rdfs:Resource

Example 1

The following declares that 'Dog1 is an animal', 'Cat1 is a cat', 'zoos host animals' and 'Zoo1 hosts the Cat2' represented by ontology in Table 6. 21, the RDF graph in Figure 6.9 and Table 6.22 for Turtle format

Table 6.21 Ontology of an animal

```
ex:dog1          rdf:type          ex:animal
ex:cat1          rdf:type          ex:cat
ex:cat           rdfs:subClassOf   ex:animal
zoo:host  rdfs:range          ex:animal
ex:zoo1          zoo:host          ex:cat2
```

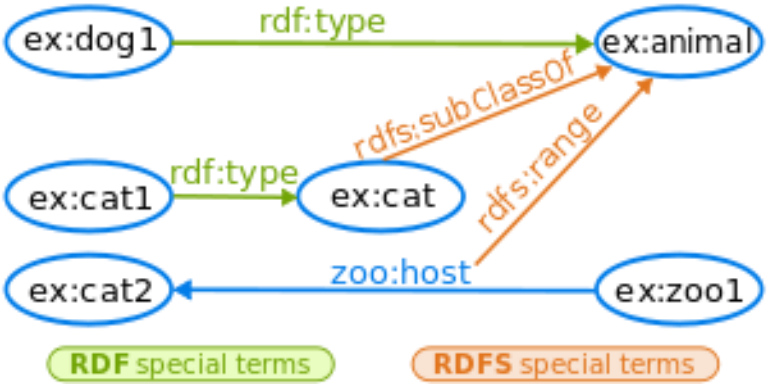


Figure 6.9 The RDF of animal

Table 6.22: RDF/Turtle format of animal

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix ex:  <http://example.org/> .
@prefix zoo:  <http://example.org/zoo/> .
ex:dog1  rdf:type  ex:animal .
ex:cat1  rdf:type  ex:cat .
ex:cat  rdfs:subClassOf  ex:animal .
zoo:host  rdfs:range  ex:animal .
ex:zoo1  zoo:host  ex:cat2 .
```

If your triplestore (or RDF database) implements the regime entailment of RDF and RDFS, the SPARQL query as follows (the keyword "a" is equivalent to rdf:type in SPARQL):

Table 6.23: SPARQL query of animal

```
PREFIX  ex: <http://example.org/>
SELECT ?animal
WHERE
  { ?animal a ex:animal . }
```

Gives the following result with *cat1* in it, because the Cat's type inherits of Animal's type:

Table 6.23 SPARQL query result

animal
<http://example.org/dog1>
<http://example.org/cat1>
<http://example.org/cat2>

6.3 Linked Data: Four Rules, Five Stars, and a Plan

Linked Data is a set of best practices for publishing and connecting structured data on the Web. It refers to the collection of interrelated datasets on the web (web of data). The Semantic Web is a Web of data — of dates and titles and part numbers and chemical properties and any other data one might think of. RDF provides the foundation for publishing and linking your data. Various technologies allow you to embed data in documents (RDFa, GRDDL) or expose what you have in SQL databases, or make it available as RDF files. It is not a single standard or format but, as Tim Berners-Lee says in the informal document that first stated the four rules of linked data, it is an "expectation of behaviour." In design terms, Berners-Lee defined that behaviour in this way is to use the four rules stated by Table 6.24

For more information on Linked Data, watch this additional video by clicking on the link
What is Linked Data? - YouTube

6.3.1 Principles of Linked Data according to Tim Berners-Lee

There are four principles of linked data, paraphrased along the following lines:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)

4. Include links to other URIs. so that they can discover more things.

The emphasis here is on the use of identifiers, and in fact that is a key element of linked data. RDF is designed for machine "understanding" and therefore does not use natural language in its statements. In fact, the subject and predicate must be identifiers, and the identifiers are in the form of a URI (called a URI because it identifies, not locates, as its function). These rules simply say that you should use identifiers for the things you are describing with your metadata and for your metadata itself. Those identifiers will be more precise than natural language, they will be language-neutral, and because the identifier takes the form **"http://"** it can also be used to provide information at that location about the thing it identifies.

The five stars in the Table 6.24 define linked data that adheres to the Semantic Web standards.

Table 6.24 W3C's five-star

S/N	Semantic Web standards	Stars
1	Available on the web (whatever format) *but with an open license, to be Open Data*	★
2	Available as machine-readable structured data (e.g. excel instead of image scan of a table)	★★
3	as (2) plus non-proprietary format (e.g. CSV instead of excel)	★★★
4	All the above plus, Use open standards from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff	★★★★
5	All the above, plus: Link your data to other people's data to provide context	★★★★★

This is a high-level view, and it needs some filling in before it can become a plan. One possible plan is laid out in the Singapore Framework for Dublin Core Application Profiles developed by the Dublin Core community. As described in the step-by-step document "Guidelines for Dublin Core Application Profiles," the steps are listed out in Table 6.24

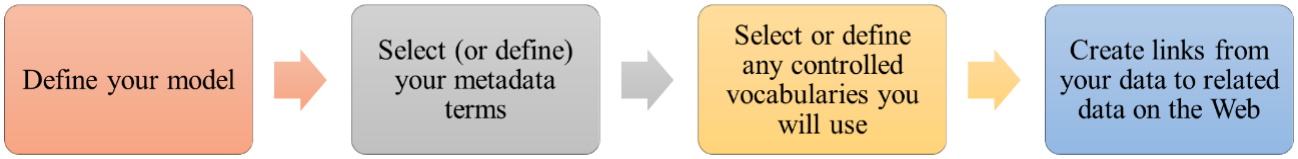


Figure 6.10 Steps to building framework

This process is significantly different from the way that metadata was created in the unlinked world. In the past, when you planned new metadata, you had to take into account only your known data-sharing partners and develop a standard that all could agree upon. In the linked data world, the scope of sharing has become the entire World Wide Web. This opens up your data for greater use, but it also means that you need to think broadly about how your data fits with such a large information base. It helps to think about universals in your metadata, things like *people*, *places*, and *physical description*. These are concepts that are not limited to any one community but will be useful in many data contexts. These common elements are obvious points for linking and should not be seen as internal to any one community's data. At the same time, the first community to define useful terms is contributing them to the general pool of useful terms and elements that anyone can take advantage of. In this sense, libraries, with their extensive set of controlled vocabularies (including those in authority files), have a lot to contribute to shared linked data.

6.3.2 The Cloud

The very first point on the five-star linked data cup is, "On the web, open license." For your data to participate on the Web, it has to be openly accessible and usable. This does not mean that you could not create a closed linked data system for your own purposes, and in fact there is considerable attention at this time to the creation of enterprise systems using linked data. But we presume that libraries and other cultural heritage institutions will wish to contribute to the open exchange of information on the Web and the knowledge-creation activities that the Web of data will foster. This open data Web is visualized in the linked data cloud, and you will often hear the expression *linked open data* (LOD) referring to data that is available for unfettered use on the Web. It is worth taking a short look at the cloud itself as it exists today. There is a picture of the cloud that can help us visualize the datasets that are there and the connections between them. Beginning

in 2007, when the cloud had only 12 datasets, Richard Cyganiak, of DERI in Ireland, and Anja Jentzsch, of the Freie Universität Berlin, have created a picture of the linked datasets and their connections. This graph now has over 300 members, and it is hard to take it in as a single picture. These are not all of the sets of linked data in the world; the creators select only those with a significant number of links between them. The Web of data is growing by leaps and bound, not gradually, because many large datasets are being added from existing applications. There also is no comprehensive search engine for this data, so discovering previously unknown data of interest is still problematic. As the cloud diagram has grown, it has become useful to gather the entries into categories based on the type of data and the community they serve. The categories and some examples from each are shown in Table 6.25

Table 6.25 The categories and examples cloud and community

Categories	Examples
Media	BBC ProgrammesNew York TimesMusic Brainz
Geographic	Ocean Drilling CodicesMetoffice Weather ForecastsGeoNames
Publications	Manchest Reading ListsSudocOpen LibraryLCSH
User-generated content	FlickrSemantic Tweet
Government	data.gove.ukTraffic ScotlandOpen Election Data Project
Cross-domain	FreebaseSearsLinked Open Numbers
Life sciences	PubMedChemBLGeneID

The centre of the cloud is DBpedia as shown in Figure 6.11.

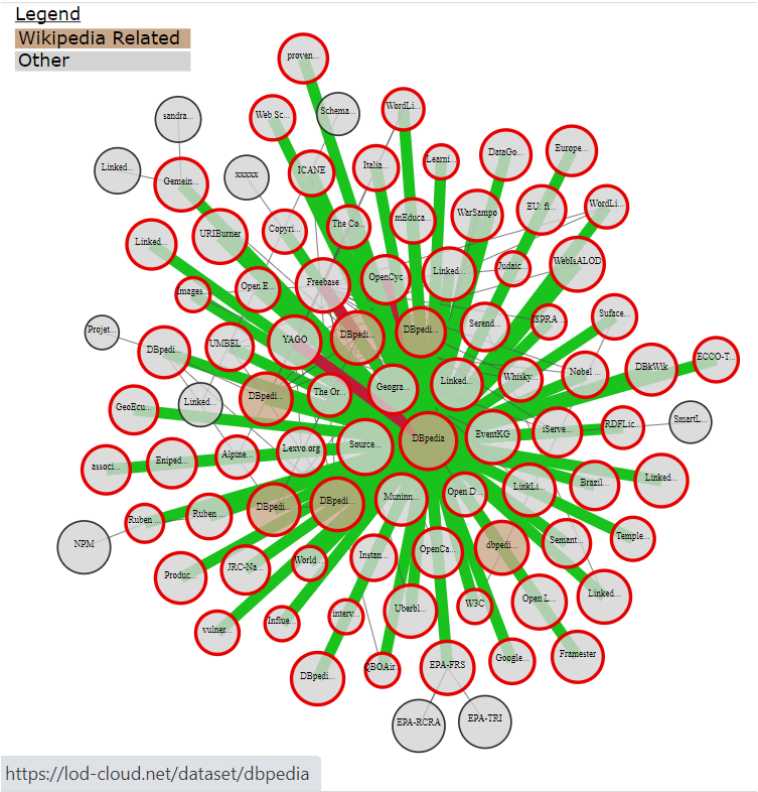


Figure 6.11 Centre of the LOD cloud diagram

DBpedia is an extraction of data from the information boxes of *Wikipedia*. You will have seen information boxes in the upper right of each *Wikipedia* page, but there are many that appear throughout a *Wikipedia* entry even though they may be less noticeable in the display. Each information box is a set of structured data, like dates, longitude and latitude, or the offices held by an elected official. The information boxes are specific to the type of data: those for people are different from those for places or events or technologies.

Anja Jentzsch, one of the creators of the linked data cloud diagram, has characterized DBpedia as "querying Wikipedia like a database." DBpedia allows structured queries that are more information-rich than the simple keyword search within Wikipedia itself. Because Wikipedia is encyclopedic in the information it contains, DBpedia is as well. This makes DBpedia an ideal meeting point for a wide variety of linked data.

Peer to Peer Interaction

Have you seen a simple RDF statement?

Have you seen a simple SPARQLE statement before?

What difference do you notice about the two?

6.4 Semantic Modelling

Semantic modelling shows the relationships that exist between classes or among specific values of data. It is a technique used to define the meaning of data within the context of its interrelationships with other data. It is also an abstraction which defines how the stored symbols relate to the real world. RDF offers a flexible, graph-based model for recording data that is interchangeable globally, it doesn't offer any means to record *semantics* or *meaning*. Data modelling is the process of creating a data model for the data to be stored in a database. This data model is a conceptual representation of Data objects, the associations between different data objects and the rules.

Table 6.26:	Importance of Semantic Modelling

1. It captures the "meaning" of your data with all its inherent relationships in a single enterprise Knowledge Graph for your entire organization;
2. It allows your data model to evolve at the pace of your Research/business demands so you can include additional business requirements, data sources and other models;
3. It makes your data more accessible to data scientists and business analysts by granting a unified access to knowledge from multiple sources;
4. It provides the ability to query the data and ask questions that you haven't anticipated while modelling your data;
5. It translates your data into usable information consumable for decision-making purposes.

6.4.1 Steps to data modelling

A data can be modelled with symbols. This can be done by identifying the object sets or entity of the problem, their relationships and semantics. The steps are shown by Figure 6.12 while the different symbols are presented by Table 27

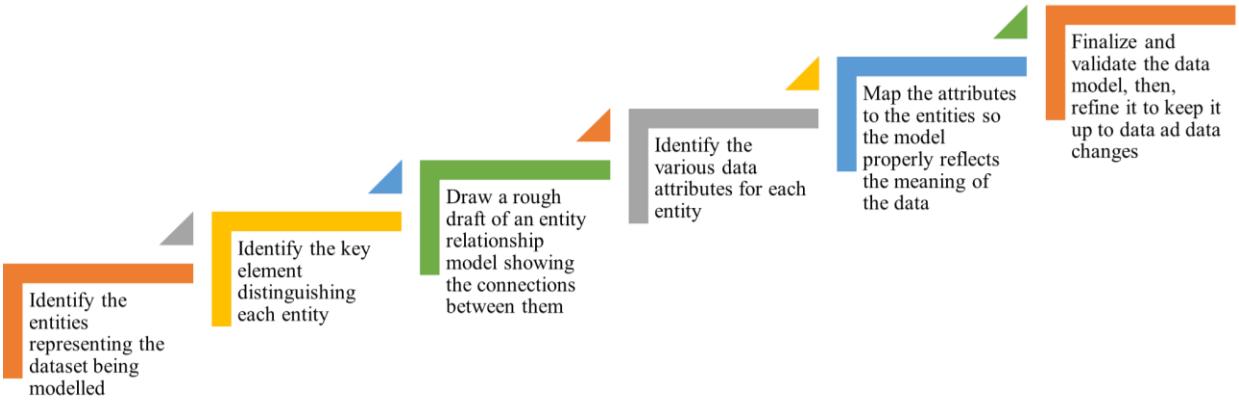


Figure 6.12 Data modelling steps

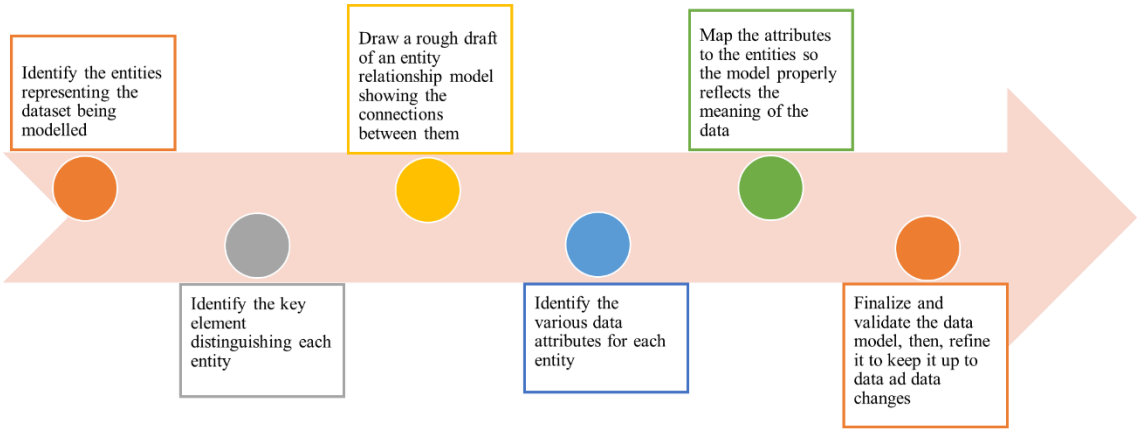
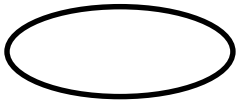
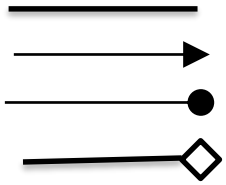


Table 4.27 Symbols used for modelling

Legend	Description
	Object or Entity set
	Attribute of Entity
	Aggregation
	Relationship set

Example 1: Data model of a university student. A student has the attributes of name, department and matriculation number.

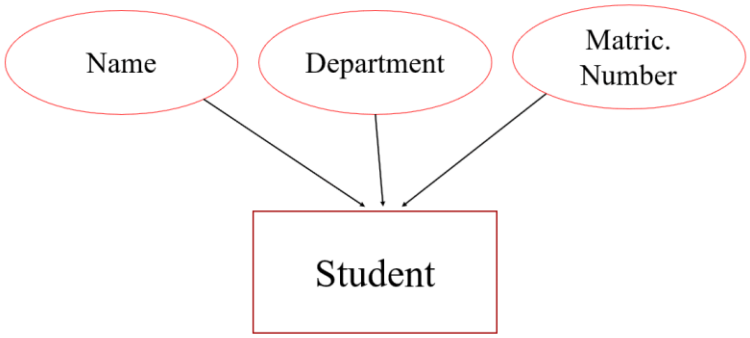


Figure 6.13 Student data model

Example 2: Semantic data model of an employee in a company. All employees have attributes of a name, kind of job and salary paid at the end of the month. All departments of the company has location, name and departmental ID. The relationship between all employees and department is that one employee must belongs to one department.

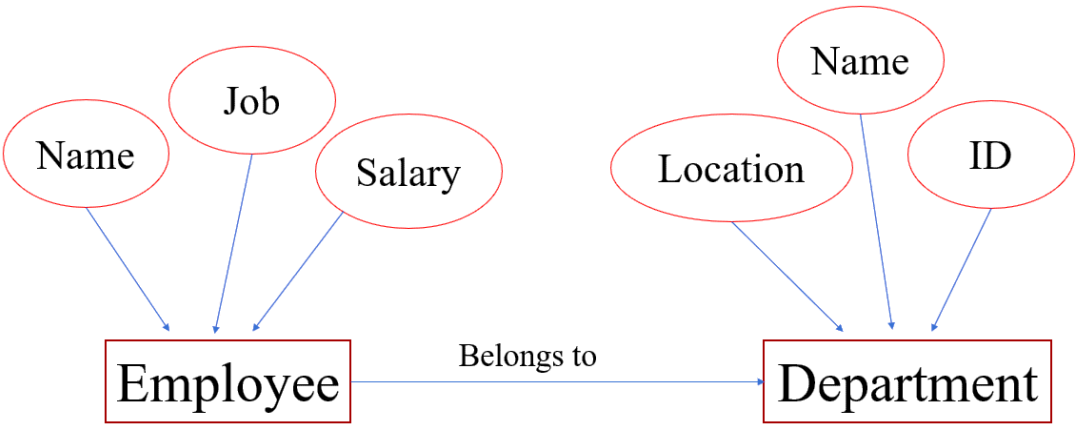


Figure 6. 14 simple semantic model of an employee

6.4.2 Abstractions for Data Modelling

The three (3) abstractions for data modelling are:

a) **Classification** is a form of abstraction in which a collection of objects is considered a higher-level object class. Essentially, it represents an is-instance-of relationship.

b) **Aggregation** is the means by which relationships between low-level types can be considered at a higher-level type. Semantic data modelling permits the aggregation of entity types (or relations) to form higher order entities.

c) **Generalization** is the means by which differences among similar objects are ignored to form a higher order type in which the similarities can be emphasized.

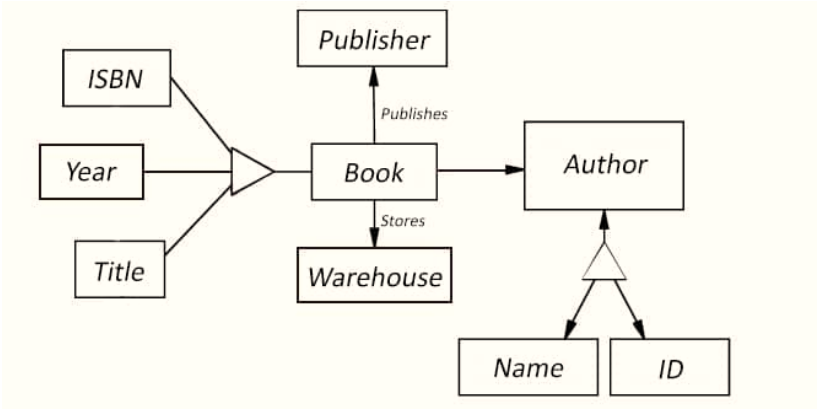


Figure 6.15. Semantic data model instance

A semantic data model can be represented graphically in an Abstraction Hierarchy diagram showing the types (as boxes) and their inter-relations (as lines). It is hierarchical in the sense that the types which reference other types are always listed above the referenced type. This simple notation principle makes the diagrams very easy to read and understand, even for non-data modellers. Figure 6.9 illustrates an example of semantic data model of a book using graphical symbols to represent data semantics as shown in Table 6.27. The graphical symbols in Figure 6.15 are meant to be generic-representative of the symbols used and illustrative of the kinds of data sets and constraints included in typical semantic data models. The solid-bordered box signifies a set of objects or values. In Figure 6.15, Author designates the writer of the book (e. g. Sakinat Folorunso) and Publisher represents the book publisher (e.g., Elsevier). Relationship-set names can be explicit or can be a composition of the names of connected object and value sets. Names in Figure 6.15 are all compositions (e.g., Author-Name names the binary relationship set between Author and name). In general, relationship sets are *n*-ary $(n \geq 2)$. Constraints on relationship sets include functional/ non-functional and mandatory/optional constraints. An arrowhead designates a functional relationship set from its tail(s) as domain space (s) to its head(s) as range space(s) (e.g., Book-Publisher is functional from Book to Publisher). Aggregation constraints, represented by a triangle, signifies sub-part/super-part constraints. By Figure 6.15, Name and ID make up an Author, and ISBN, Year and Title constitute a Book.

There are various popular, mainstream ways to model data, some of which have emerged later than others. Table 6.28 presents comparison between the approaches and highlights some of the unique qualities of the semantic data model.

Table 6.28 Comparing the Popular Data Models

Model	Example Format	Data	Metadata	Identifier	Query Syntax	Semantics (Meaning)
Object Serialization	.NET CLR Object Serialization	Object Property Values	Object Property Names	e.g. Filename	LINQ	N/A
Relational	MS SQL, Oracle, MySQL	Table Cell Values	Table Column Definitions	Primary Key (Data Column) Value	SQL	N/A

Hierarchical	XML	Tag/Attribute Values	XSD/DTD	e.g. Unique Attribute Key Value	XPath	N/A
Graph	RDF/XML, Turtle	RDF	RDFS/OWL	URI	SPARQL	Yes, using RDFS and OWL

NB **Metadata** simply means "data about data" (taken from the Greek meta- meaning "after"). The table 5 above shows some examples of how you might classify the metadata for various different models. Including semantic meaning to your data is that it can be branched across **domains of knowledge** automatically.

Peer to Peer Interaction

Discuss another example of semantic model instance

Assessments – TMA

Tutor Marked Assignment I

In your own words, what is a Semantic Web?

How does it relate to Linked data?

RDF and SPARQLE statement. Are they the same?

If Yes, Justify your response with facts.

If No, what are the differences?

Explain OWL in your own words

Compare popular data model

Explain the five-star rule of Linked data

6.5 eCRF as a FAIR Tool

6.5.1 Introduction

Data Stewardship Wizard (DSW) is a joint ELIXIR CZ and ELIXIR NL project bringing a simple but powerful solution for researchers to help them understand what is needed for good, FAIR-oriented Data Stewardship, to find ELIXIR experts to help out, and to build their own Data Management Plans. The DSW can also function as a check list for data management professionals, like the checklists used by pilots before each flight.

More information on DSW eCRF from https://youtu.be/aGpr6JFMuiE

6.5.2 FAIR

The main driver for the DSW is now to offer a convenient helpful tool for data stewards and researchers. Given a limited funding, we focus on this mission now. However, from a long-term perspective, the richness of knowledge contained in the Wizard definitively calls for being FAIR. On this page we track the progress of compliance with the FAIR principles.

6.5.3 Machine-Actionable DMPs

We are part of the initiative #activeDMPs. Here, we will post updates on concrete steps, mostly with the respect to the identified use cases. The work on this front will continue according to our available capacity and funding.

6.5.4 eCRF

An eCRF (electronic Case Report Form) is a software system used to collect data in a clinical study. Commonly, eCRFs are web-based applications containing various data forms and fields designed to receive data in clinical trials or observational studies. eCRFs are fundamental elements in clinical research since they are the instruments utilized by study coordinators and investigators to enter data from source documents, which are then cleaned and exported into the database for statistical analysis.

Welcome Page for eCRF

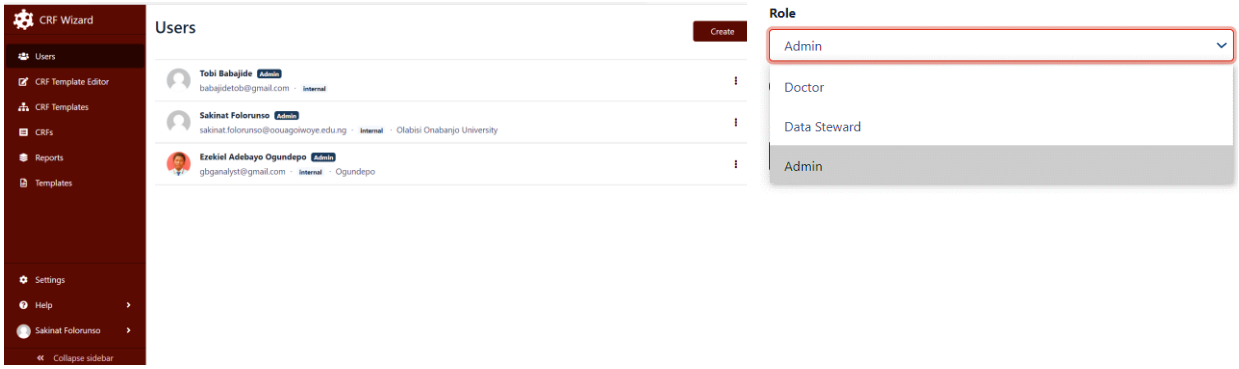
This is the welcome of the DSW eCRF. We can navigate to all other pages using all fields labelled on the left pane. Each field will be explained using the labelled alphabets

CRF Wizard logo

This the CRF Wizard name and logo indicating the wizard page to enable templates creation

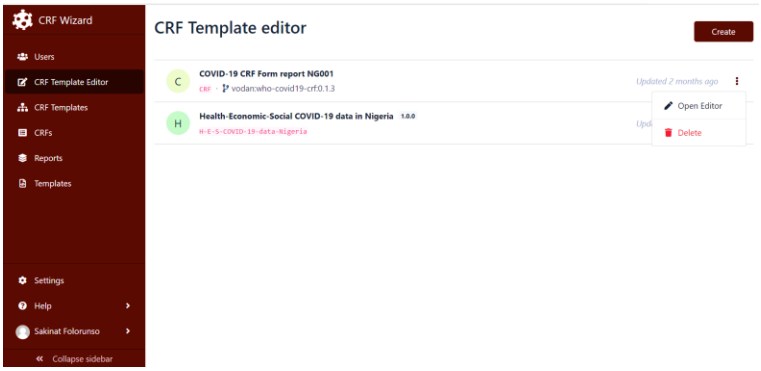
Users

Shows all DSW eCRF users. These users, depending on their role as ‘Admin’, ‘Doctor’ or ‘Data Steward’ have different rights. The Administrator ‘Admin’ can create, edit or delete users directly. Users have the permission to create and use existing templates. You can also create new users by clicking on ‘Create’ button. When editing the user, it is possible to change all the attributes from registration and also manually change the “Active” status.

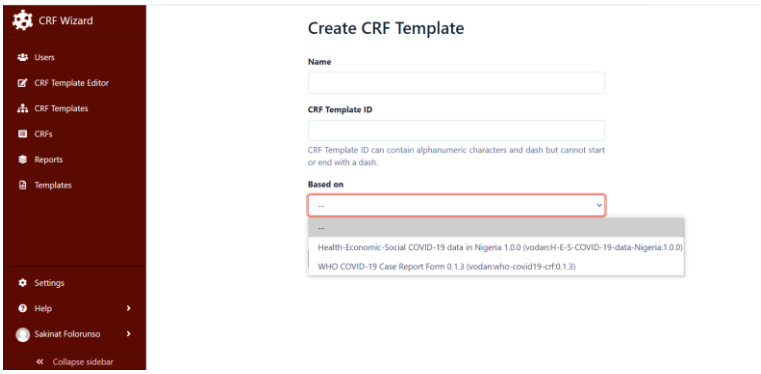


The CRF Template Editor

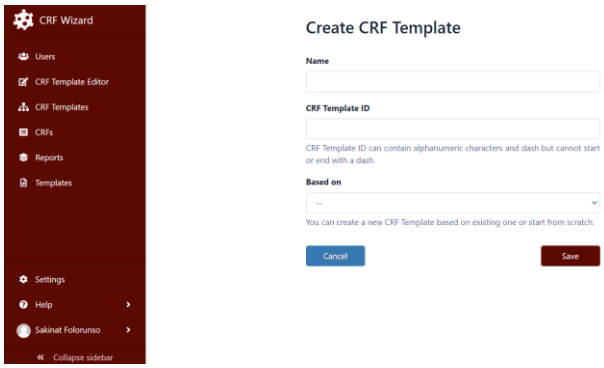
The CRF Template Editor page is used for creating and updating existing templates by clicking on 'CRF Template Editor' in the left-hand menu of the wizard. A new template can be created by clicking on the 'Create' on the top to the right. In this case, we have two (2) existing templates already where you can open to edit or delete using (⋮). It will also show the last edit date.



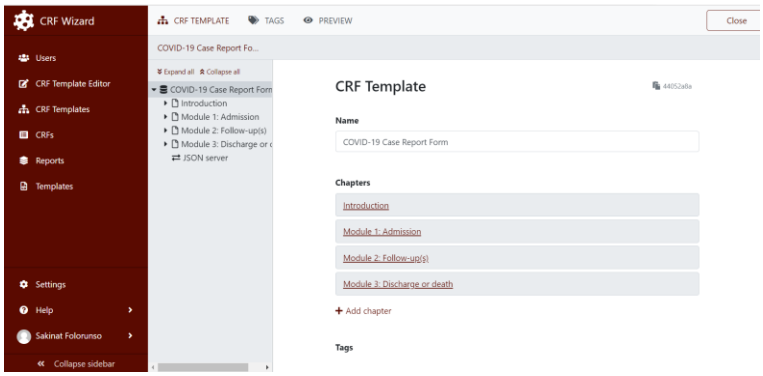
On clicking Create to create a new template, you fill the fields with appropriate **Name** (like 'Nigeria COVID-19 dataset') and **CRF Template ID** ('NG-001'). The ID can contain alphanumeric characters and dash but cannot start or end with dash. But If plan to update or reuse an existing template, you select 'based on' the existing template. In this case, we have two (2) existing templates to choose from.



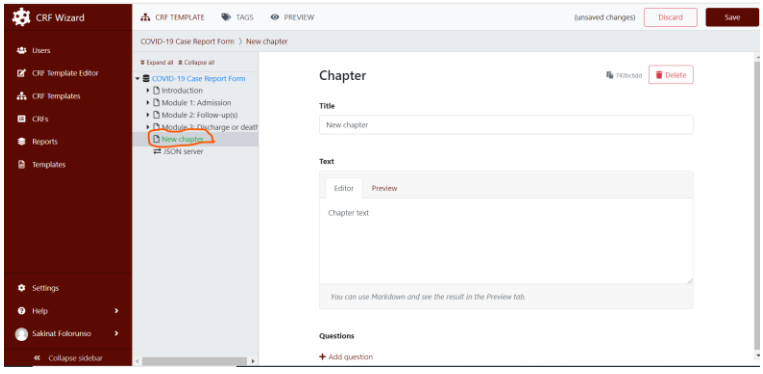
Then, click on "Save" button to save the template or cancel if you wish.



So, based on 'WHO COVID-19 Case report Form 0.1.3 (vodan:who-covid19-crf0.1.3), template we have this page.

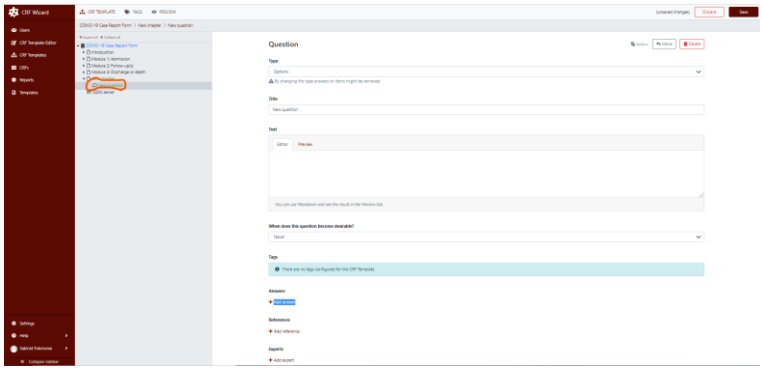


Once you have created a new template, you fill it with chapters (i.e. sections; think of them as headers of your dataset). Write a name of the template. In this case 'COVID-19 Report Form'. Press '+ Add chapter' to add a new header to your data.



Note that a ‘new chapter’ placement has been created on the left-hand side and circled red. Write a **Title** name in the box (like ‘Introduction’, ‘Discharged’ etc). Use the ‘**Text’** field to give the background information and a description of what the chapter will contain questions about. There are two tabs, ‘Editor’ and ‘Preview’, since you can use ‘Editor’ to format the text and check the result in ‘Preview’. Notice the grey area which gives an overview of your template. Use this to navigate between the different parts. Whenever you want to see the result of an addition, click on another part in the template overview

Press ‘+Add question’ to create a question in respect of the chapter



Note that a ‘New Question’ placeholder has been created on the left-hand pane and encircled red.

Add a question

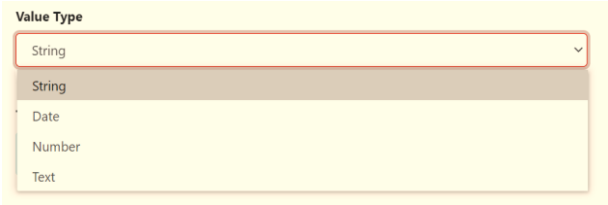
Questions can be of different types:

- Options
- List of items
- Value

Let’s create some questions of each type to demonstrate:

Value

1. Select *Value* as **Question Type**
2. Write ‘Project title’ in the **Title** field
3. In the **Text** field, write ‘Please enter the title of your project’, as instructive text
4. Use **When does this question become desirable?** to indicate in which phase of the project a question should be answered, e.g. *Never*.
5. **Value type** can be *String*, *Date*, *Number* or *Text*, select *Text*

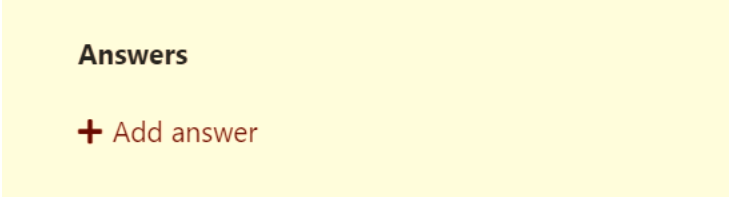


List of items

1. Click on Introduction in the grey overview area of the editor
2. Press +Add question in order to create a new question
3. Select *List of items* as **Question Type**
4. Write ‘Project members’ as **Title**
5. Write ‘Please specify the researchers participating in the project’ as **Text**
6. Select ‘Never’ as **When does this question become desirable?**
7. In the **Item Template**, click on +Add question
   7.1. Set the **Question type** to *Value*
   7.2. Write ‘Name’ as **Title**
   7.3. Back to one level up by clicking on Project members in the grey overview part of the editor
   7.4. Scroll down and click on +Add question in the **Item Template**
   7.5. Set the **Question type** to *Value*
   7.6. Write ‘Email’ as **Title**

Options

1. Click on Introduction in the grey overview area of the editor
2. Press +Add question in order to create a new question
3. Select *Options* as **Question Type**
4. Write ‘Research field’ as **Title**
5. Write ‘Please select the research field for this project’ as **Text**
6. Click on ‘+Add answer’

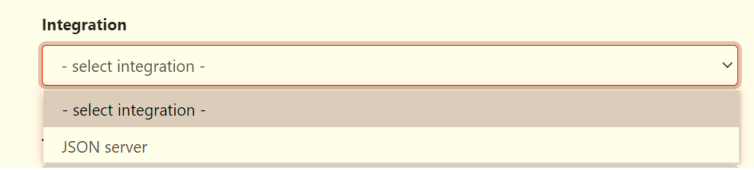


7. Write ‘Life science’ as **Label**
8. Write ‘Your project is likely going to produce a lot of data, a full data management plan will prepare you for the various challenges this will entail’ as **Advice**
9. Click on Research field in the grey overview area, scroll down to **Answers** and click on +Add answer
10. Write ‘Other’ as **Label**
11. Scroll down to **Follow-up Questions** and click on +Add follow-upp question
12. Select *Value* as **Question Type**
13. Write ‘Which other research field?’ as **Title**
14. As **Value Type**, select *Text*

Integrations

The DS Wizard support integrations of external services with API responding with JSON result containing a list of items. Such a list can be used for type hints in special type of questions - Integration question. You can create a new integration in the root of any template by clicking on **Add integration** in **Integrations** section.

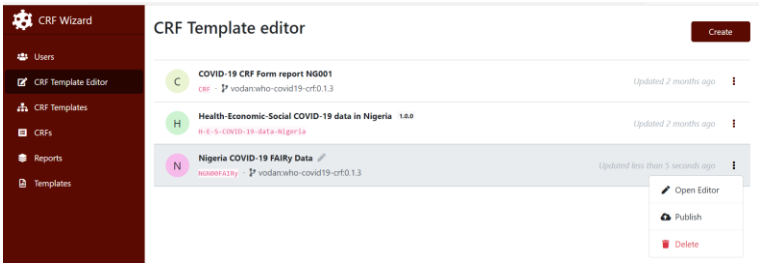
1. Select *Integration* as **Question Type**
2. Write ‘Integration’ in the **Title** field
3. In the **Text** field, write ‘Please enter the integration server’, as instructive text
4. Use **When does this question become desirable?** to indicate in which phase of the project a question should be answered, e.g. *Never*.
5. **Value type** can be *JSON server*, select *JSON server*



Save

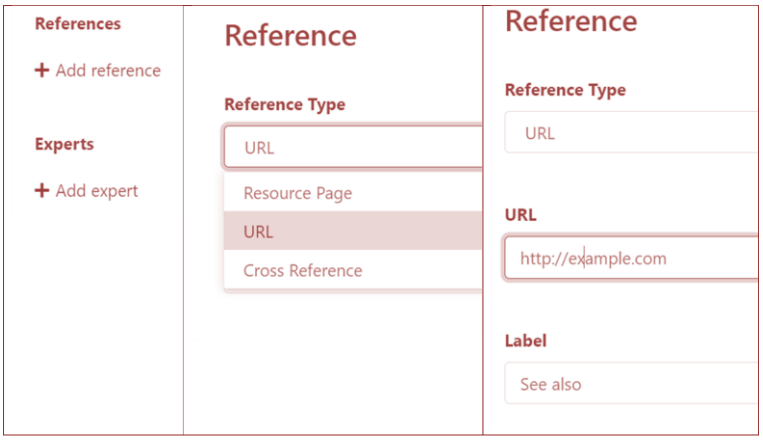
Whenever there are changes not saved in the CRF Template, the clickable options “Discard” and “Save” is visible in the top row of the wizard. Click on “Save”.

Notice that this automatically lead you to the top level of the CRF Template Editor, outside your CRF Template. If you position the mouse (⋮) on your template (‘CRF Template Editor’) you see the options of ‘Open Editor’, ‘Publish’ and ‘Delete’. Click on ‘Open Editor’ in order to continue editing the Template.



Add Reference and Expert

All question types have the possibility of adding references and experts, to be used for adding additional information and people to contact in order to get help, respectively. Let’s add one of each:



Click on the arrow at Research field in the grey overview area, and scroll down to **References**

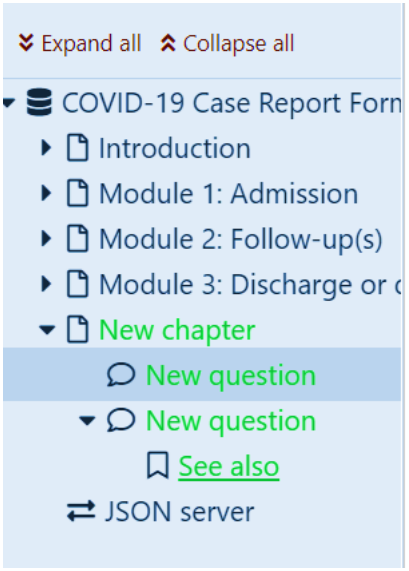
1. Click on '+Add reference' and specify the type in 'Reference Type'
2. Write "https://en.wikipedia.org/wiki/List_of_life_sciences" as **URL**
3. Write "List of Life sciences" as **Label**
4. Click on Research field again and then click on +Add reference
5. Write "My science for life guru" as **Name**
6. Write "help@scilifeguru.com" as (the fake) **Email**
7. Click on "Save" in top right corner, position the mouse on your Template ('CRF Template Editor'), and click on Open Editor

Change order of questions

It is possible to rearrange the order of questions, if they are on the same 'level' (but not between levels at the moment of writing this tutorial). Let's try:

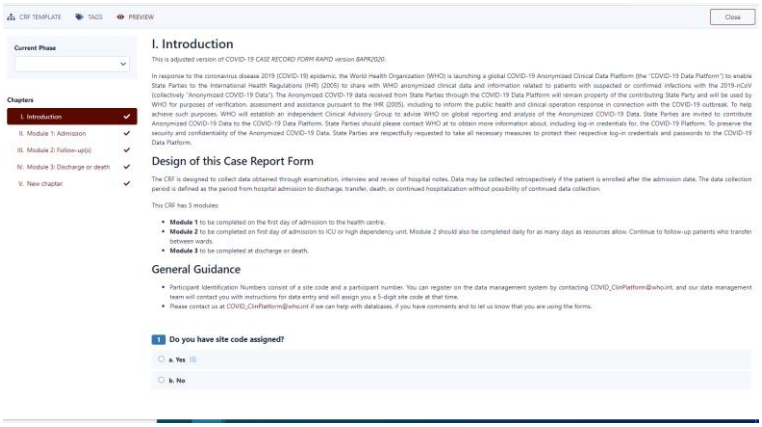
1. Click on Introduction and scroll down to **Questions**. The four questions created are all on the same level.
2. Position the mouse on the grey area next to 'Module 2: Follow-up(s), hold down the left button of the mouse and drag-and-drop above 'Introduction'

If and when the need arises to move a question to another level/group of questions, you can also rewrite the question in the new position.



Preview

In order to see what the resulting questionnaire will look like, you can click on “PREVIEW” on top row of the wizard as shown below. When you are finished just klick on ‘CRF TEMPLATE’ on the top row, to return to editing mode.



Tags

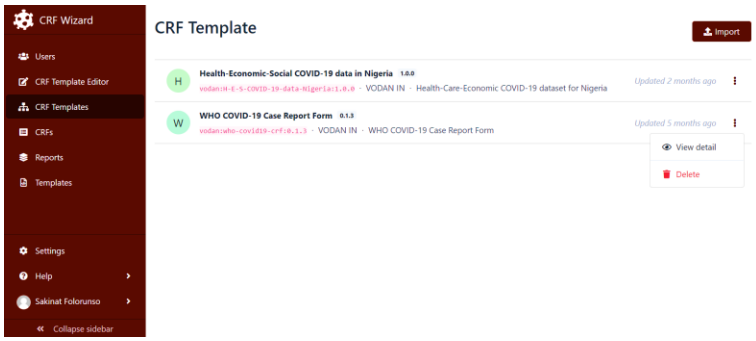
Also on the top row of the wizard, is the “TAGS” viewer function. Tags can be used to mark questions of interest to certain stakeholders, when only a subset of the questions are of interest. We have not defined any tags yet, so let’s do that:

1. Click on ‘TAGS’ in the grey overview area, and scroll down to **Tags**
2. Click on + Add tag

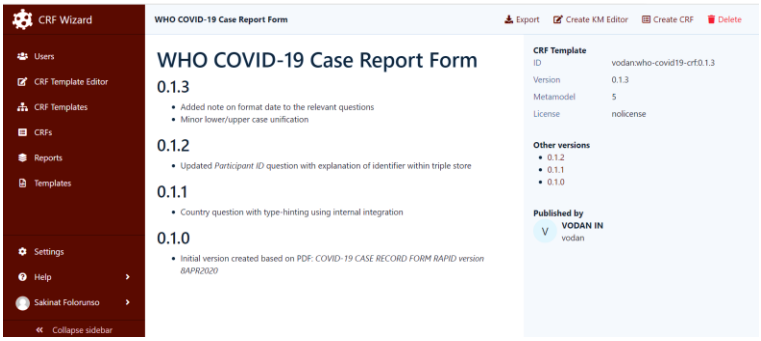
3. Write ‘Library’ as **Name**
4. Pick a color by clicking on one of the colored squares
5. Expand Introduction and click on Project title in the grey overview area
6. Scroll down to **Tags** and mark this question as of interest to the library by clicking the check box
7. Do the same for Research field
8. Click on “PREVIEW” and notice that all questions are visible. Select the Library tag, by clicking in it’s check box. Now only the two questions *Project title* and *Research field* are visible.

CRF Template

This page shows the already existing template and the last updated time. Any of the existing templates can be viewed by clicking (⋮) or deleted if necessary.

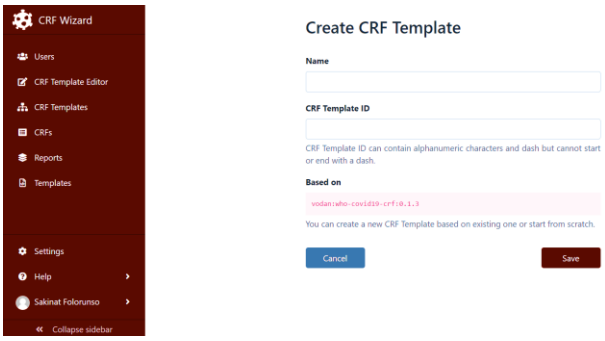


When you click on ‘view detail’ field, it takes you to the page that shows the latest and other versions of the particular template, its ID, Metamodel, license granted and the publisher. It also shows the ‘Export’, ‘Create KM Editor’, ‘Create CRF’ and ‘Delete’ buttons at the upper-right corner of the page. You can delete if you wish otherwise.

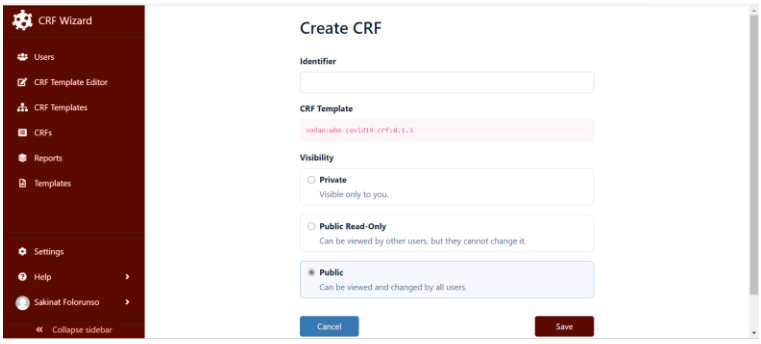


The '**Export'** button on the upper-right top menu. The template can export into a json formatted file on the computer.

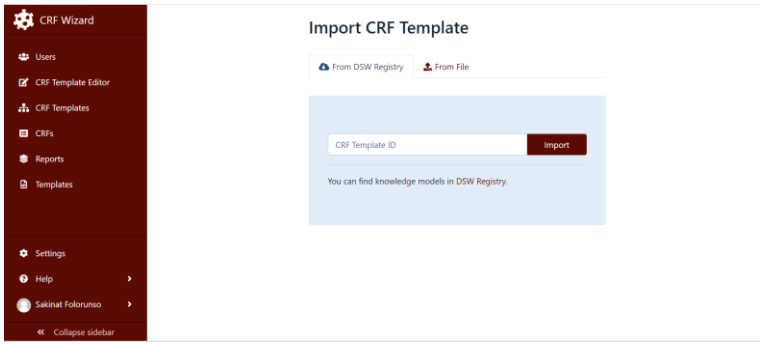
The 'Create KM Editor' creates a new template based on **current** template



The 'Create CRF' creates a new CRF template based on the current template. You can check any of the three visibility options based on personal choice. Then click'Save' button to save the new CRF template

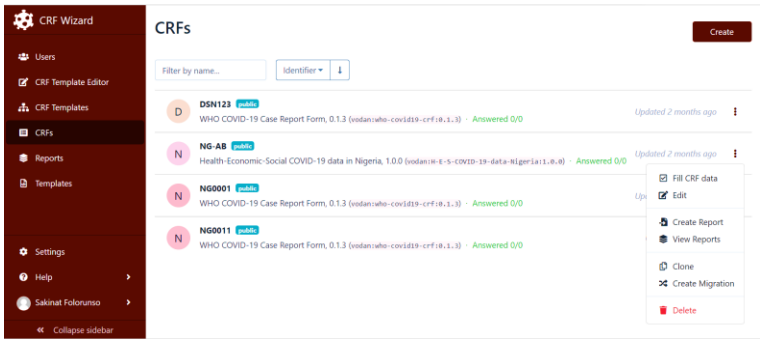


When you click on CRF Templates (d), click on the ‘Import’ button on the top-right corner of the page. You can import a CRF template from outside of the CRF Template Editor either from ‘DSW Registry’ or you upload from files on your computer. You click on the ‘Import’ button as soon as a ‘CRF Template ID’ is placed in the placeholder beside the ‘Import button.

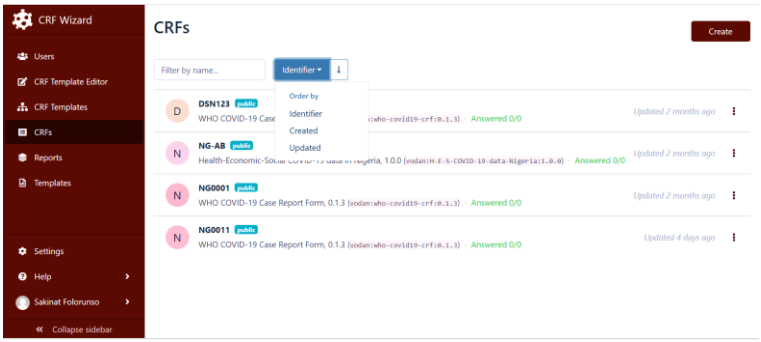


CRFs

The CRFs page contain all of the CRFs created for particular purposes. You can also create a new CRF template by clicking on ‘Create’ button on the top-right corner of the page. For each of the CRFs created by users, you can fill up the template with data by clicking on ‘Fill CRF data’ or edit the existing CRF or create report or view report or clone CRF or Create migration or delete the CRF. You can achieve all of this by clicking (⋮) on the far right corner of the particular CRF.



You can search for a particular CRF or order/arrange according to their identifier name, the date they are created or updated

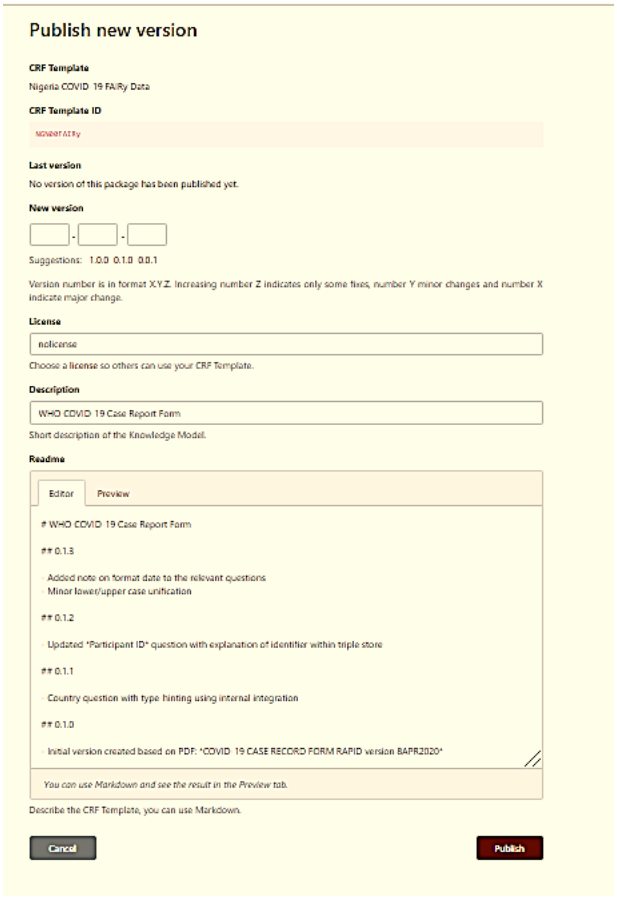


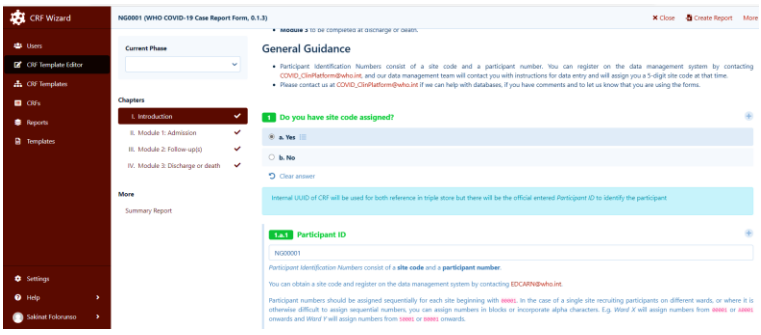
Publish

When you are happy with the content and look of your Template, it is time to make it available for people to use it (either as a start for their own KM or for users to fill it out in form of so-called Questionnaires):

a. Click on “CRF Template Editor” in the left side menu
b. Position the mouse on (⋮) on the left-hand side of ‘Nigeria COVID-19 FAIRy Data and click on Publish among the alternatives that becomes visible

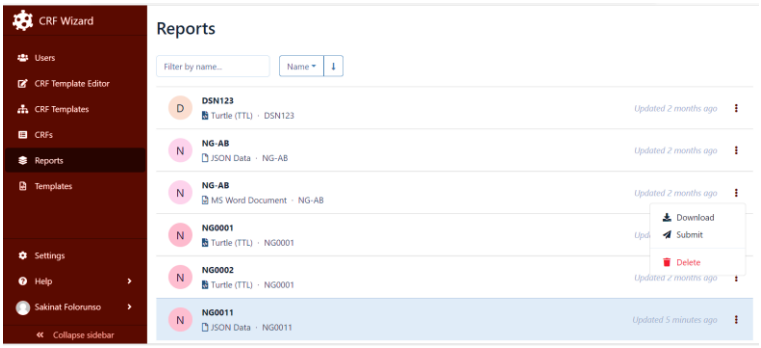
c. Add a version number in **New version** (e.g. ‘1.0.0’)
d. Write a **Description** (e.g. ‘This is the root version’)
e. Click on “Publish”



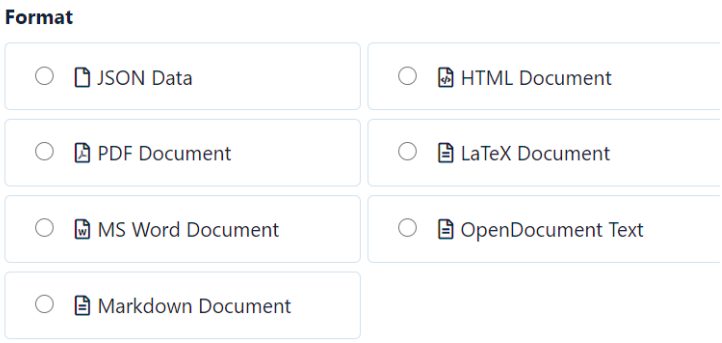


Reports

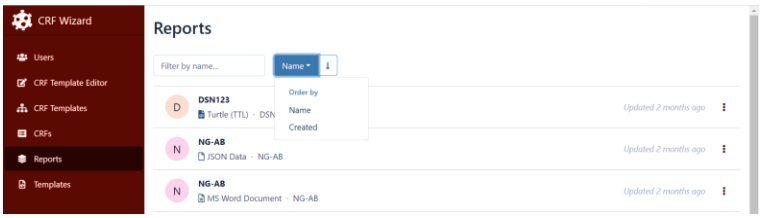
All created reports are enlisted and accessed on this page. A report could be downloaded, submitted or deleted as deemed fit.



A report could be generated in different file formats. It could be Turtle (TTL) for CRF form or JSON etc for Questionnaire report as shown in the diagram below.

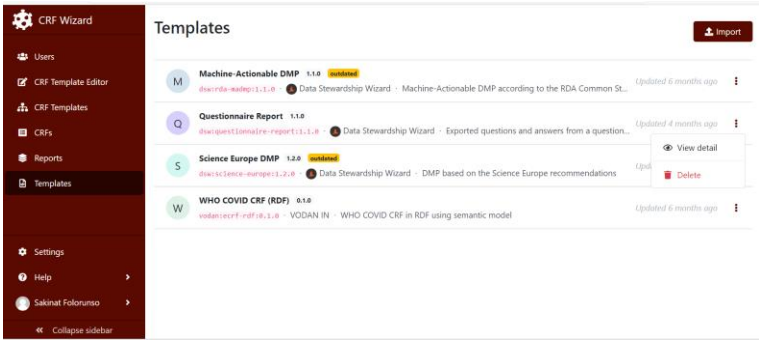


A report could be ordered/arranged in ascending or descending order by name of the report or the date created.



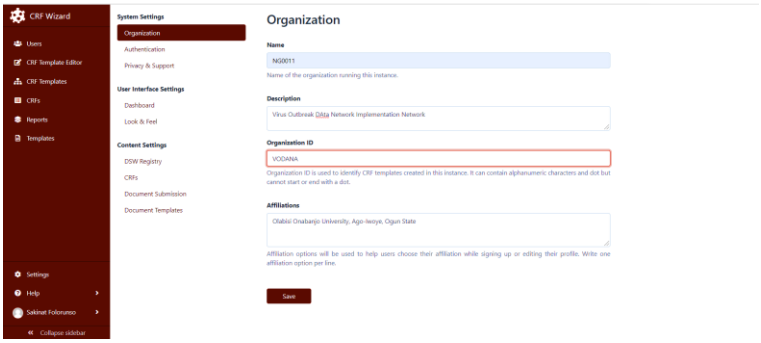
Templates

This page contains the entire templates available to the user. They were all imported either from DSW Registry (ds-wizard.org) or from the files on the computer or by default.

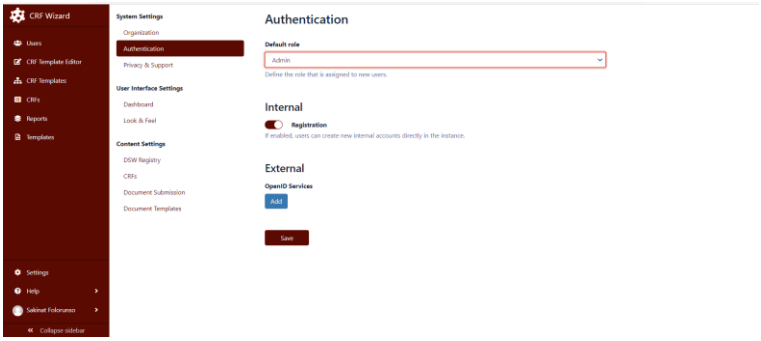


Settings

The administrator can set the organization name and ID. Organization name is the visible name of the organization that uses DSW instance and Organization ID is the unique identifier of the organization, it is then used in identifier of created Knowledge Models. Click on the 'Save' to save the page.



Users authentication is done the role of the user usually the admin. Click 'Save' to save the page.



The DSW Registry should be ‘Enabled’ and ‘Token’nised to be able to import templates from DSW. You can get valid token when you sign up for it at DSW Registry (ds-wizard.org) and click ‘Save’

Create a simple eCRF metadata template

6.6 CEDAR as a FAIR tool

The CEDAR Workbench is an essential component of open science, ensuring FAIR data and enhancing scientific reproducibility. The CEDAR Workbench makes it easy to collect and use metadata. CEDAR tools help you create forms to collect metadata, make those available to users, and download the information that users have provided.

Link to the CEDAR Workbench for students https://youtu.be/A2dWI85wrK4

6.6.1 What is CEDAR?

The Center for Expanded Data Annotation and Retrieval (CEDAR) was established in 2014 to create a computational ecosystem for development, evaluation, use, and refinement of biomedical

metadata. The approach is centered on the use of *metadata templates*, which define the data elements needed to describe particular types of biomedical experiments. The templates include controlled terms and synonyms for specific data elements. CEDAR uses a library of such templates to help scientists submit annotated datasets to appropriate online data repositories.

CEDAR is an end-to-end process that enables

- community-based organizations to collaborate to create metadata templates,
- investigators or curators to use the templates to define the metadata for individual experiments,
- scientists to search the metadata to access and analyze the corresponding online datasets.

6.6.2 Capabilities of CEDAR

CEDAR can

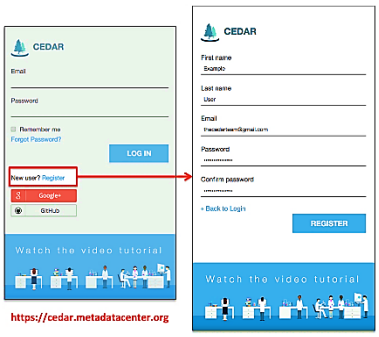
- create user-friendly, shareable forms for collecting metadata, with features like
- share your forms and metadata
- link your questions (fields) and possible answers (values) to controlled terms
- view metadata responses meeting your search criteria, in several forms
- use the Workbench Desktop interface to manage your content
- enable intelligent metadata suggestions in your
- remotely access CEDAR content and capabilities using the CEDAR REST API

With these capabilities, you can capture simple or rich metadata for your project, build a repository of project metadata, or design and prototype a new metadata specification to meet your particular needs.

6.6.3 Creating a CEDAR Account

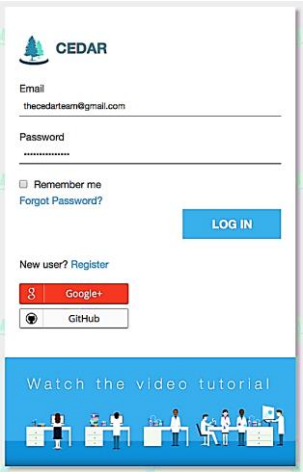
The first step to create metadata templates and metadata with the CEDAR workbench is to create a CEDAR workbench user account. To do this, visit https://cedar.metadatacenter.org and click on the "Register" link below the Password line.

In the next window, enter your first name, last name, a functional email address, and a password as shown in the screenshot below. You should make the first name and last name what you want to appear on your user folder. After clicking on the REGISTER button, check your email (at the address you provided) for a link to validate your user account. Click on the link in your email to complete the account creation process. After validation you can easily log in to the CEDAR workbench as indicated in the next section.



6.6.4 Logging in with CEDAR

To log in, go to https://cedar.metadatacenter.org, and enter the account name and your password as shown in the screenshot below. If you have forgotten your password, you can use the "Forgot Password?" link just below the Password line. Once you have logged in, you will see the CEDAR Workspace.



6.6.5 Visibility of your CEDAR Account

All of the CEDAR account names are visible to other CEDAR users, by navigating to the **/All/Users folder.** (Only 100 users are loaded at a time; scroll to the bottom of the page to load more names.)

In fact, because sometimes you may want to share or describe the location of your directory, we encourage you to make sure your user name is unique, and if you have two accounts, to give them different user names. If you want to change an existing user name on an account, please contact us at user-support@metadatacenter.org.

If the shared resources include any folders, everything within those folders (recursively) will be shared with the same permissions.

If you share content that you have created in your workspace, anyone who can see your shared resources will also see name of the folder hierarchy within which they live. (They will not be able to navigate into unshared parts of that hierarchy.)

Finally, when you create or update assets, the unique identifier that CEDAR creates for your account is saved in the metadata of the asset. Eventually we may publish a service that provides your account name—and any other profile information you have explicitly shared—to someone who visits the unique identifier. To avoid this, keeping all your work private will prevent anyone from discovering your unique identifier.

6.6.6 CEDAR's Metadata Template

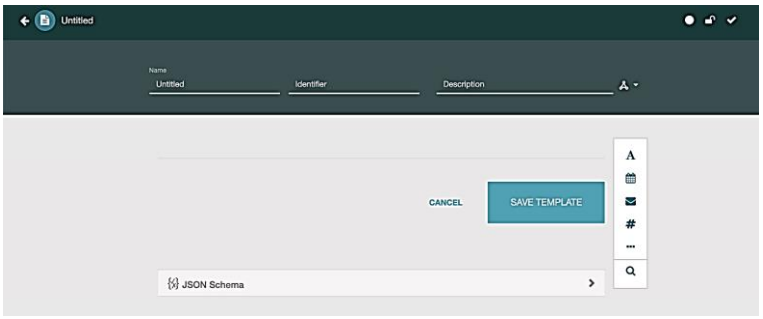
In CEDAR, forms are created as what we call Metadata Templates, or just 'templates'. The templates contain individual questions—what we call Metadata Fields, or 'fields'—and collections of those fields called Metadata Elements ('elements').

Creating your CEDAR Workspace

In the first steps to create a CEDAR Metadata Template resource, you will provide a human-readable label, a unique identifier, and a description of what the Template resource represents (e.g., "COVID-19 patients").

To create a new template, first click the "New" button on the Desktop's navigation sidebar (upper left of the Workspace view) and select the "Template" option in the dropdown menu. This step opens the Template Designer as shown below.

Enter the human-readable Name, Identifier and Description of the Template resource using the three text input fields ('Untitled', 'Identifier', 'Description') underlined in the image below. The Name is used as the name of the artifact in the Desktop, and can be changed from the Desktop view.



You can save and close your Template at any time and return to it later. You can open it for viewing or editing (or any other Template for which you have those access privileges) by double-clicking on its icon in the Desktop.

TiP	By default, CEDAR searches look at the artifact Name, Identifier, Description, and Version fields for matches, so consider what terms will be most important to include for searches to find your template.

CEDAR Workspace

When you log in to CEDAR, you will be in your own workspace as shown in Figure 1. Though a new workspace will not include the white space in the diagram.

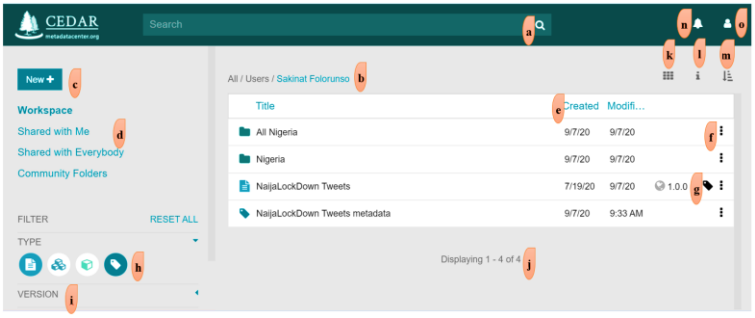


Figure 1: CEDAR workspace

Explanation of the content of Figure 1 using the letters in the orange boxes.

(a) The search bar lets you find other CEDAR resources.

(b) The location string tells you the local folder for this display.

(c) To start creating your own content like templates, elements, fields and folders, you will click on the "New+" button and select.

(d) These 3 options on the left of the window control which kind of resources you will see. The "Workspace" view shows your home directory. The "Shared with Me" view shows content that has been explicitly shared with you, or a team you are on. The "Shared with Everybody" view shows content that has been shared with everyone on CEDAR.

(e) The large white Resources box lists the resources you can view. Controls '(a)', '(b)', '(d)', '(h)', and '(i)' affect what content you see in this box. You can sort the items in this box by clicking on the '(m)' icon.

(f) Each resource in the Resources box has its own menu dropdown, selected by (⋮) at the right of the resource. With this menu you can rename, copy, move, share, and perform many other functions with the item.

(g) Click on this metadata tag to enter the Metadata Creator and start filling out metadata following that particular template.

(h) These round icons on the left side of the window filter what kind of content you can see. If the icon is green (with white figures), it is highlighted and you can see the corresponding content. If the icon is white (with green figures), that content type is disabled. (Folders are always visible.)

(i) The Version selector dropdown menu lets you see either the current published version of a template or all versions.

(j) The number of items listed in the Resources box is shown here. If the number of items is larger, only a subset is shown. Scroll the display up to see more items.

(k) In this location is an icon to let you switch your Resources box to cards or list format. The cards format is shown in the left-hand smaller image. The left-hand image in Figure 2 shows the cards-based view of the Resources box while the right-hand image shows the informational panel with metadata displayed for the highlighted resource.

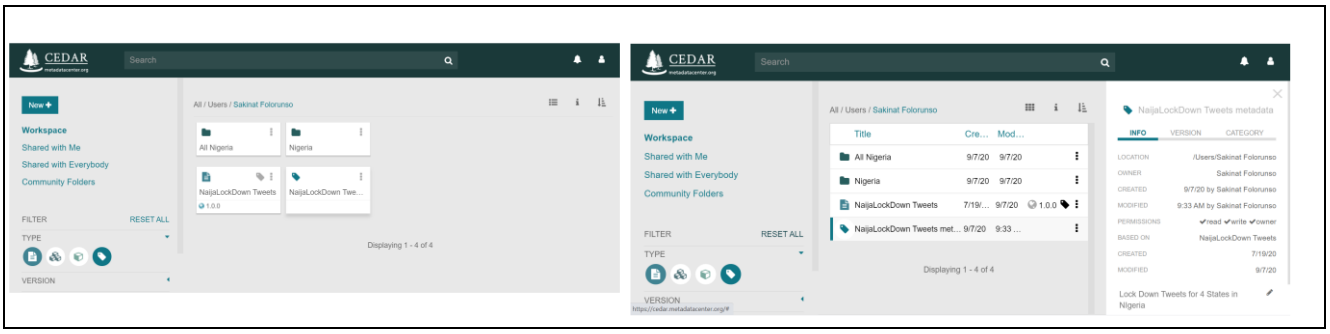


Figure 2: Resource box format

(l) The 'i' icon will bring up an information panel to the right side of your display, providing additional information (metadata) about the selected resource. If no resource is selected, the 'i' icon presents metadata about the folder (B) shown in the Resources box. The right-hand smaller image below shows the information panel for one of these artifacts.

(m) The sort icon lets you choose the sort criteria for the Resource box contents.

(n) The bell icon indicates whether you have any messages from CEDAR. A white icon indicates no messages.

(o) The profile icon lets you see information about your CEDAR profile, including your API key, and offers features like 'Support' and 'Logout'.

Creating a template

The CEDAR Metadata Template ('template') serves three goals:

- defining the template questions (including their possible answers and other features);
- documenting the order in which the questions and elements will appear; and
- describing the template artifact: its name, its provenance including creation and update times, and its other characteristics.

Basic Description of CEDAR's Metadata Template

The template is in JSON Schema format, and conforms to a higher-level JSON Schema specification. The higher-level JSON Schema can be used to validate the syntax of any CEDAR template at any time; errors in this validation are displayed in the Template Designer user interface, as described in Filling Out (Creating) Metadata: Saving and Validating.

You can specify the content and order of questions in the template using the Template Designer. You make up a template from Template Fields and Template Elements. The Template Elements are made up of other Template Elements and Template Fields. The Building Basic Templates chapter describes the template specification process using the Template Designer.

Template Organization

CEDAR organizes templates in two modes: Internal and External

Internal Organization

All information for the Metadata Template is stored internally in the JSON Schema format. CEDAR lets you view the template in this format at any time during template creation. See Viewing Resource as Raw JSON to learn how to view the template in its raw form.

The three Metadata Template functions (define questions, order the questions, and describe the template) are roughly organized into three sections in the JSON Schema template artifact.

The details of the template artifact's format is described in more detail in the CEDAR Template Model.

JavaScript Object Notation (JSON) is a generic data format with minimal number of value types like strings, numbers, Booleans, lists, objects and null. Although the notation is a subset of JavaScript, these types are represented in all common programming langiages making JSON a good candidate to transmit data across language gaps. A simple example of JSON syntax is thus:

```
{
  "firstName": "Sakinat",
  "lastName":  "Folorunso",
   "#oflogin": 7,
   "isIstructor": true,
   "worksWith": ["Olabisi Onabanjo University"]
   "music": [
     {
       "Singer"; "Amirah Ajao"
       "type": "Gospel"
      }
   ]
}
```

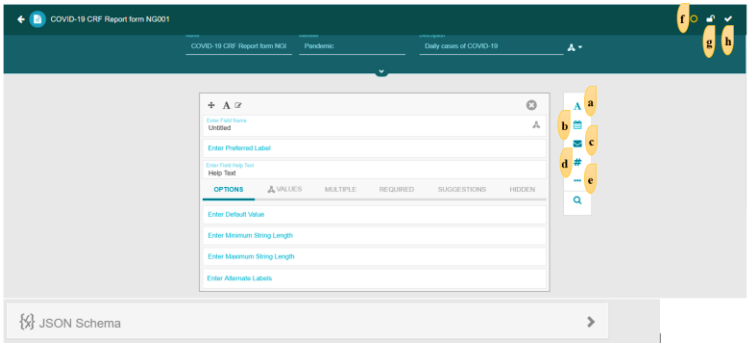
Logical Organization

As a CEDAR user you will compose Metadata Templates from Template Elements and Template Fields. These three CEDAR resources, which we call templating resources, together establish the questions for the metadata user to fill out. The actual process is described in the Building Basic Templates.

As you reuse elements that you or others have created, CEDAR copies those elements into the main template, in whatever order you place them. You can reorder the elements and top-level fields in the template, but can not reorder fields within the template's elements.

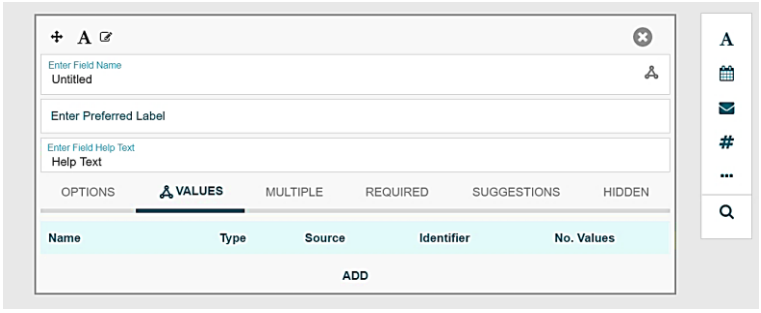
Metadata *about* the template are entered automatically by the system, and manually by you as the template creator.

From this point onward, the diagram below will be used as reference to all explanations



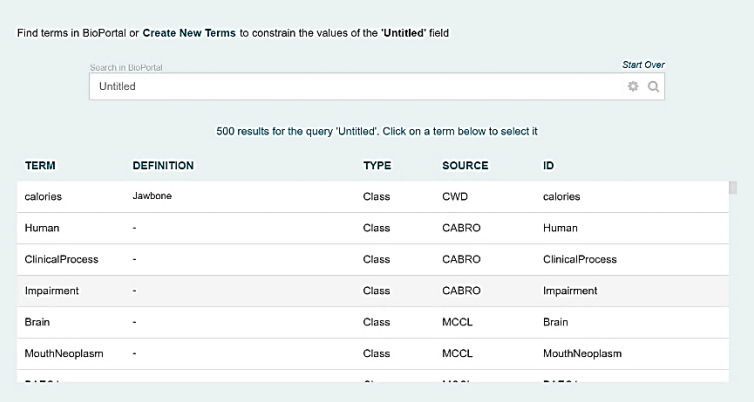
a. **Text:** on clicking the '**A**', the window to enter values to create a text field. You give a name to the field and corresponding label

Values: The Values tab shows a view of the current value set(s) that the metadata author can use to fill out a text field. If no selections are listed, the metadata author is prompted to enter free text. This ability to choose controlled values for the field is fundamental to CEDAR's semantic capabilities.



Clicking on the "ADD" button brings up a modal window from which you can select the term, branch, or ontology from which legal values may be chosen by the metadata author. After you've made the selection, you will be returned to the view of the field which includes a line describing your selection. You can repeat this process by (clicking the Add button) as often as you want to

select additional terms, branches, or ontologies. You can remove any of the selected terms, branches, or ontologies by clicking on the X to the right of the row you want to delete.



Multiple

When Multiple is set to Yes, the Template Designer displays a control to define the minimum and maximum number of entries allowed for the field. If Multiple is set to Yes, then the Metadata Editor provides an interface to fill out the field multiple times. The Metadata Editor forces the specified minimum number of fields to be present (though the user is not forced to fill out the fields with values). The Metadata Editor prevents creating more than the maximum number of entries by making the field Copy icon unavailable.

Required

If this option is set to Yes, the Metadata Editor indicates to the metadata author that the field is required, and issues a warning when the metadata is saved while this field has not been filled out with metadata. The metadata author can choose to ignore the warning and save the metadata even though the required field is not completed.

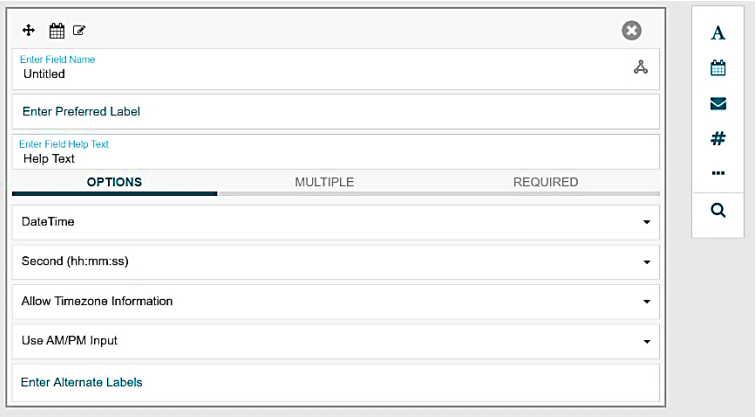
Suggestions

Set the Suggestions tab to Yes to enable intelligent authoring suggestions for this field from CEDAR. You can read a detailed explanation of the suggestions system at Understanding the Suggestion System.

Hidden

Set the Hidden tab to Yes to enable show any hidden intelligent authoring suggestions for this field from CEDAR. You can read a detailed explanation of the suggestions system at Understanding the Suggestion System.

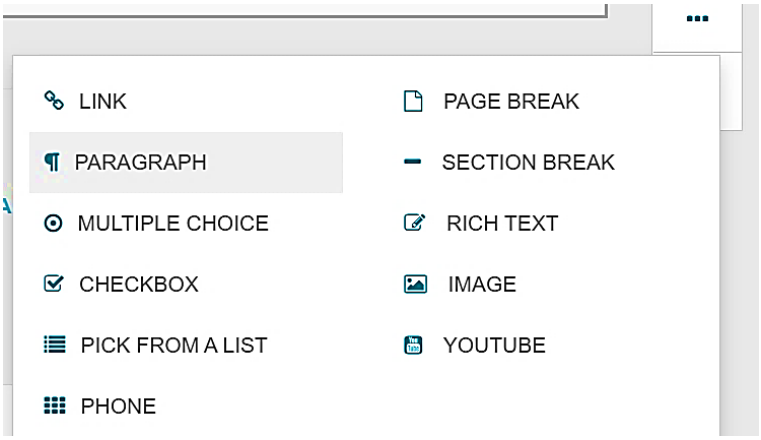
b. The date/time field creator. This field allows you to create a date or time field. You name the field and give specification according to ‘OPTIONS’ ‘MULTIPLE’ and ‘REQUIRED’ field



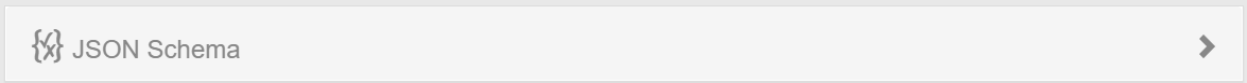
c. Email creator field. This icon can be used to create an email address field

d. This field labelled ‘d’ can be used to create a numeric field. It could either be a floating or integer number. The length of the value can be specified and also the minimum and maximum values.

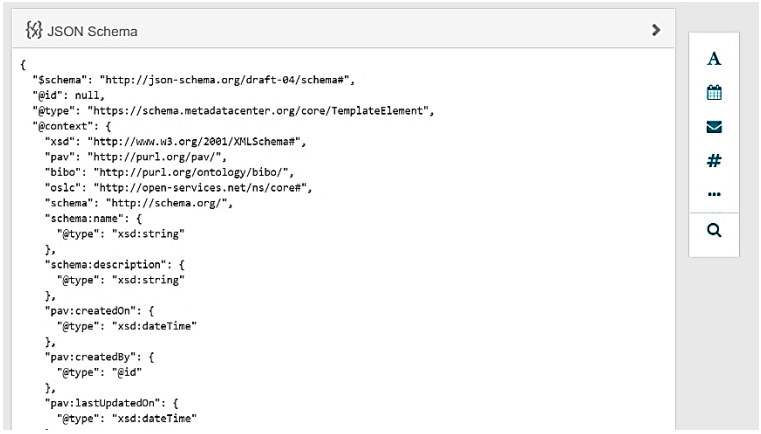
e. Many other common fields but not listed can be accessed on clicking the (⋯) button. The list is shown in the diagram below



Then, you can generate a JSON schema of your template by clicking on the (>) button.



A sample JSON schema is shown below.



Saving and Closing

f. **Saving:** To save a templating resource—Template, Element, or Field—click on the Save button at the bottom right of the templating resource. This will be labelled Save Template, Save Element, or Save Field depending on the type of templating resource you are editing. At the top right of the templating resource, there are 3 icons that indicate the resource status. The left-most icon (the circle) is filled when the instance has no unsaved changes, but is a hollow yellow circle when there are unsaved changes.

g. **Lock:** If the lock (the middle icon) is yellow and locked, you will not be able to save your changes, even to a separate file. To create an editable version of the resource, you will have to exit the resource, make a copy of it from the Desktop, and edit the copy you made.

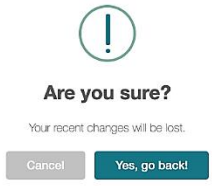
h. Validation

CEDAR performs syntactical validation of a templating resource when it is opened and saved. This checks that the resource conforms to the Template Model schema, which is expressed as JSON Schema. The validation should always succeed and yields a white checkmark in the third icon. If the syntactical validation fails, the checkmark turns yellow, indicating there is a problem in the CEDAR system. Often you still will be able to work with the resource, but it is best to contact the CEDAR team to alert them to the problem. On a successful verification, CEDAR issues a green "[Artifact] saved successfully" notification, where '[Artifact]' will be replaced by 'Template', 'Element', or 'Field'.

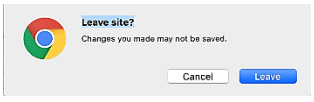
Closing

Saving before closing

While a templating resource is being edited in the Template Designer, it can be saved at any time.



If you have made changes and try to leave the resource without saving it by using the CEDAR back button, CEDAR will issue a warning. If you click Continue, any changes you made will be lost and CEDAR will return to the Desktop view. Choosing the Go Back button will return you to the templating resource.



If you have made changes and try to leave the resource without saving it by using the browser back button or closing the browser window, the browser will issue an error message. In Chrome, click on Leave Site (the default) to close the resource, and on Cancel to remain on the page.

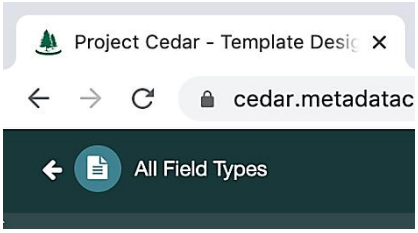
Because there might be unusual conditions that cause you to lose your session,

we strongly encourage you to save your work often.

We are unable to restore work that has been lost when the session is lost.

If you have remained on the page, you can save the resource and then return to the Desktop view or close the window.

Returning to Desktop



When you close the template to return to the CEDAR Desktop view, there are two ‘return left’ buttons—the one in the browser, and the one at the upper left of the CEDAR Template Designer. The CEDAR button remembers the last folder you had open in the Desktop view, ignoring any searches you may have performed since then.

If you launched the Template Designer from a CEDAR search results page, and want to return to that page, use the browser back button, and the same search should be performed. (If you mistakenly chose the CEDAR back button, you can still use the browser button twice to get back to the previous search results.)

Activity Area Learning Activity	Create a simple CEDAR metadata template

Glossary of Terms

a. **artifact**: any of template, element, field, instance. All CEDAR artifacts are specified using CEDAR's Template Model.
b. **CEDAR Workbench**: suite of web-based tools and RESTful APIs that allow users to construct templates, populate templates with metadata, and to share and manage these resources.
c. **copy**: make a copy of an artifact to a given folder.
d. **details panel**: right-hand panel that shows the details of a particular template or folder.
e. **draft**: early version of an artifact.
f. **element**: reusable artifact that may contain one or more fields and/or one or more nested elements. *e.g.: address element with three fields: street, city, and ZIP code*
g. **element designer**: component of the CEDAR Workbench that allows users to construct elements.
h. **field**: reusable artifact used to capture an atomic piece of metadata. *e.g.: title, description, start date*
i. **field designer**: component of the CEDAR Workbench that allows users to construct fields.
j. **finder**: dialog which allows you to select a library component to add to your template or element.
k. **folder**: storage space where CEDAR artifacts and other folders can be placed together.
l. **global, global open**: globally accessible without logging in, aka public, FAIR.
m. **group**: logical collection of CEDAR users.
n. **instance**: set of metadata values resulting from a populated template.
o. *e.g.: title = "Study about CRC", start date = "03/07/2014", ZIP code = "94305", etc.*
p. **library component**: a reusable field or element.
q. **metadata editor**: component of the CEDAR Workbench that allows users to create metadata by populating templates with values.
r. **metadata repository**: centralized place where all CEDAR artifacts are stored. CEDAR's metadata repository stores templates, elements, fields, and instances.
s. **move**: move an artifact or a folder to a given folder.
t. **notification**: information shown by the CEDAR Workbench to a particular user or group of users to inform them when a particular event related to their CEDAR resources occurs.
u. **open**: open an artifact for editing.
v. **populate**: fill out a template.
w. **publish**: specify that the status of an artifact is final. Published artifacts cannot be modified or deleted.
x. **resource**: any of template, element, field, instance, folder, user, group (CEDAR RESTful resources).
y. **resource manager**: primary front-end component of the CEDAR Workbench that allows users to search, browse, and manage CEDAR resources.

z. **share**: share an artifact or folder with a user or group of users.

aa. **side navigation panel**: left-hand panel that shows create and search options.

bb. **submission**: metadata content that is sent out from CEDAR.

cc. **submit**: send metadata to a repository along with data files.

dd. **template**: group of fields and/or elements that is used to capture metadata for a specific purpose.

ee. *e.g.: study template with three fields: title, description, start date, and one element: address*

ff. **template designer**: component of the CEDAR Workbench that allows users to construct templates, aka form builder.

gg. **toolbar**: template designer's floating toolbar on the right-hand side, used to select a library component to add to the template.

hh. **user**: a person who has created an account on the CEDAR Workbench and who uses any of its tools.

ii. **version**: way to categorize the different states of an artifact.

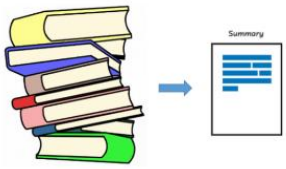
jj. **workspace**: folder that acts as the default storage location for all the CEDAR reources created by a user.

References

1. “Library Linked Data Incubator Group Wiki,” main page, W3C website, last modified February 22, 2012, www.w3.org/2005/Incubator/lld/wiki/Main_Page.
2. Tim Berners-Lee, “Linked Data,” W3C website, July 27, 2006, last modified June 18, 2009, www.w3.org/DesignIssues/LinkedData.html.
3. Karen Coyle and Thomas Baker, “Guidelines for Dublin Core Application Profiles,” W3C website, May 18, 2009, http://dublincore.org/documents/profile-guidelines.
4. Lampa, Samuel. 2018. "Semantic Web Data Science? - Practical large scale semantic data handling with RDFIO and RDF-HDT." SlideShare, April 10. Accessed 2017-05-27.
5. Idehen, Kingsley Uyi. 2017. "Semantic Web Layer Cake Tweak, Explained." OpenLink Software, via Medium, July 14. Accessed 2018-05-25.
6. Richard Cyganiak and Anja Jentzsch, “The Linking Open Data Cloud Diagram,” last modified September 19, 2011, http://richard.cyganiak.de/2007/10/lod.
7. Anja Jentzsch,”DBpedia - Querying Wikipedia like a Database,” Anja Jentzsch homepage on Freie Universität Berlin website, last modified February 10, 2012, www.wiwiss.fu-berlin.de/en/institute/pwo/bizer/team/JentzschAnja.html.
8. http://linkeddata.org/faq
9. Chapter 2: Semantic Web and Linked Data | Coyle | Library Technology Reports (ala.org)
10. Categories used in the LOD cloud diagram. Source: Anja Jentzsch, “LOD Cloud Diagram,” Wikimedia Commons, September 2011, **http://en.wikipedia.org/wiki/File:LOD_Cloud_Diagram_as_of_September_2011.png**.
11. https://www.w3.org/RDF/Metalog/docs/sw-easy
12. https://www.w3.org/standards/semanticweb/
13. (Berners-Lee et al.),
14. *"XML and Semantic Web W3C Standards Timeline" (PDF). 2012-02-04.*
15. ^ *"World Wide Web Consortium (W3C), "RDF/XML Syntax Specification (Revised)", 10 Feb. 2004"*.

16. ^ *"World Wide Web Consortium (W3C), "OWL Web Ontology Language Overview", W3C Recommendation, 10 Feb. 2004"*
17. Berners-Lee, Tim; James Hendler; Ora Lassila (May 17, 2001). "The Semantic Web". *Scientific American*. Retrieved July 2, 2019.
18. What is an Ontology? The simplest definition you'll find… or your money back* (kdnuggets.com)
19. Components of an Ontology | Ontogenesis (knowledgeblog.org)
20. Pierre de Keyser (2012). Taxonomies and ontologies. Chapter 7: In Chandos Information Professional Series. Chandos Publishing. Pp: 121-142. ISBN 9781843342922. https://doi.org/10.1016/B978-1-84334-292-2.50007-6. (http://www.sciencedirect.com/science/article/pii/B9781843342922500076)
21. https://www.ontotext.com/services/semantic-data-modeling/
22. http://www.linkeddatatools.com/index.php
23. Ivan HermanW3C: http://w3.org/People/Ivan/CorePresentations/SW_QA/
24. What is an eCRF? - Sofpromed
25. About | Data Stewardship Wizard (ds-wizard.org)
26. DSW Team (2020) Data Stewardship Wizard, Release 2.9.0. Pp 1-98
27. https://metadatacenter.org/
28. Musen MA, Bean CA, Cheung K-H, Dumontier M, Durante KA, Gevaert O, Gonzalez-Beltran A, Khatri P, Kleinstein SH, O'Connor MJ et al.. 2015. The Center for Expanded Data Annotation and Retrieval. Journal of the American Medical Informatics Association, JAMIA. 22 (6) 1148-1152
29. https://metadatacenter.github.io/cedar-manual/#toc
30. https://metadatacenter.github.io/cedar-manual/glossary
31. Embley D.W. (2009) Semantic Data Model. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_105
32. Devopedia. 2020. "Semantic Web." Version 6, May 27. Accessed 2020-11-24. https://devopedia.org/semantic-web
33. https://www.w3.org/TR/rdf-primer/
34. https://www.w3.org/TR/rdf-schema/
35. https://www.w3.org/TR/rdf-sparql-query/
36. https://en.wikipedia.org/wiki/SPARQL
37.

Summary of Study Unit 6



In this study unit, you have learnt that:

1. Semantic Web is a Web 3.0 web technology. It is a way of linking data between systems or entities that allows for rich, self-describing interrelations of data available across the globe on the web.
2. RDF defines the basic unit of the Semantic Web as a three-part structure, commonly referred to as a triple
3. SKOS is a standard for encoding thesauri and controlled lists and one of the first structures built on top of RDF.
4. OWL is a standard that extends RDF and is used to define specific Semantic Web metadata vocabularies called ontologies.
5. SPARQL is designed specifically to query the underlying triples of the Semantic Web using an SQL-like query format.
6. Microformat is a kind of format within a format. Microformats are designed for automated processing of data within webpages.
7. Linked Data is a set of best practices for publishing and connecting structured data on the Web. It refers to the collection interrelated datasets on the web (web of data).
8. Open data Web is visualized in the linked data cloud. The *linked open data* (LOD) referring to data that is available for unfettered use on the Web.
9. DBpedia is an extraction of data from the information boxes of *Wikipedia*
10. Semantic captures the “meaning” of your data with all its inherent relationships in a single enterprise Knowledge Graph for your entire organization
11. Semantic modelling shows the relationships that exist among specific values of data.
12. The three levels of abstraction in Semantic modelling are Classification, Generalisation and Aggregation
13. Semantic modelling can be presented in graphic format

14. An ontology is a formal naming and definition of the types, properties, and interrelationships of the entities that really or fundamentally exist for a particular domain of discourse. It is thus a practical application of philosophical ontology, with a taxonomy.
15. The DSW can function as a check list for data management professionals.
16. An eCRF is a software system used to collect data in a clinical study.
17. eCRFs are web-based applications containing various data forms and fields designed to receive data in clinical trials or observational studies.
18. The CEDAR Workbench is an essential component of open science, ensuring FAIR data and enhancing scientific reproducibility.
19. The CEDAR Workbench makes it easy to collect and use metadata.
20. CEDAR tools help you create forms to collect metadata, make those available to users, and download the information that users have provided.

Self-Review Questions for Study Unit 6

Now that you have completed this study unit, you can assess how well you have achieved its Learning Outcomes by answering these questions.

1. What are the capabilities of CEDAR?
2. Create a simple template in CEDAR
3. Create a new eCRF template
4. Name the three constituents of Ontology? Briefly describe them
5. Mention three uses of semantic modelling
6. What is DBpedia?
7. List the steps to building linked data
8. How is SPARQLE related to Semantic Web?
9. What is RDF? Describe its structure
10. Briefly describe the three level of abstraction in Semantic modelling

Self-Review Answers (SRA) to Self-Review Questions of Study Unit 6

1. CEDAR can
   - create user-friendly, shareable forms for collecting metadata, with features like
   - share your forms and metadata
   - link your questions (fields) and possible answers (values) to controlled terms
   - view metadata responses meeting your search criteria, in several forms
   - use the Workbench Desktop interface to manage your content
   - enable intelligent metadata suggestions in your
   - remotely access CEDAR content and capabilities using the CEDAR REST API
2. Creating a CEDAR Metadata Template resource, you will provide a human-readable label, a unique identifier, and a description of what the Template resource represents (e.g., "COVID-19 patients"). Then, click the "New" button on the Desktop's navigation sidebar (upper left of the Workspace view) and select the "Template" option in the dropdown menu. This step opens the Template Designer as shown below. Enter the human-readable Name, Identifier and Description of the Template resource using the three text input fields ('Untitled', 'Identifier', 'Description') underlined in the image below.
3. You can create a new CRF template by clicking on 'Create' button on the top-right corner of the CRF page. For each of the CRFs created by users, you can fill up the template with data by clicking on 'Fill CRF data' or edit the existing CRF or create report or view report or clone CRF or Create migration or delete the CRF. You can achieve all of this by clicking (⋮) on the far-right corner of the particular CRF.
4. Ontology can describe concepts (Classes), relationships between entities (Relations) and categories of things (Individuals). A Concept represents a group of different Individuals, that share common characteristics, which may be more or less specific. Individuals also known as instances or particulars are the base unit of an ontology. They may model concrete objects such people, machines or proteins. Relations describe the way in which individuals relate to each other.
5. Three uses of Semantic modelling are to

a. capture the “meaning” of your data with all its inherent relationships in a single enterprise Knowledge Graph for your entire organization;
b. allow your data model to evolve at the pace of your business demands so you can include additional business requirements, data sources and other models;
c. make your data more accessible to data scientists and business analysts by granting a unified access to knowledge from multiple sources;
d. provide the ability to query the data and ask questions that you haven’t anticipated while modeling your data;
e. translate your data into usable information consumable for decision-making purposes.

6. DBpedia is an extraction of data from the information boxes of Wikipedia.
7. Define your model; Select (or define) your metadata terms; Select or define any controlled vocabularies you will use and Create links from your data to related data on the Web
8. SPARQL is designed specifically to query the underlying triples of the Semantic Web using an SQL-like query format.
9. RDF is a formal language that defines the basic structure of the linked data that makes up the Semantic. Its structure is Subject, Predicate and Object
10. The three abstractions for data modelling:

    a. Classification is a form of abstraction in which a collection of objects is considered a higher-level object class. Essentially, it represents an is-instance-of relationship.

    b. Aggregation is the means by which relationships between low-level types can be considered at a higher-level type. Semantic data modelling permits the aggregation of entity types (or relations) to form higher order entities.

    c. Generalization is the means