Glossary

Class

In RDF Schema *rdfs:Class* is defined as a special kind of *rdfs:Resource*.
Classes are terms defined in an ontology. They are used to classify resources of a dataset.
Example:
The resource ttr:RexDog is an instance of the class tto:Dog :

```
ttr:RexDog rdf:type tto:Dog .
```

Dataset

A dataset is a collection of formal descriptions describing resources.
A dataset can be loaded in a triplestore and accessed via a SPARQL endpoint to be searched.

Datatype

In RDF Schema rdfs:Datatype is defined as a special kind of *rdfs:Class*.
It is related to *Literal* resources. Every instance of literal resource must have an explicit datatype for the literal value to be properly processed by triplestores and SPARQL engines.
Example:
Harrison Ford’ birthdate is 1942-07-13. “1942-07-13” is a literal (an instance of rdfs:Literal) which has a date datatype (xsd:date as an instance of rdfs:Datatype).

```
dbpedia:Harrison_Ford dbp:predicate "1942-07-13"^^xsd:date .
```

Domain

In RDF Schema *rdfs:Domain* is defined as a special kind of *rdf:Property*.
It is used to describe the semantics of properties. A property is a term defined in an ontology. The description of a property should define its domain. The domain of a property tells which kind of resources are supposed to be used as subject in triples involving the property.
Example:
The domain of the tto:pet property is dbo:Person means that we expect resources used as subject in triples involving the *tto:pet* property to be instances of the class *dbo:Person*.
tto:pet rdfs:domain dbo:Person .

Federated query

A federated query is a way to execute queries distributed over different SPARQL endpoints. Il allows to create queries that merge data distributed across multiple datasets. Invoking a remote SPARQL endpoint is made possible by using the *SERVICE* keyword.

Example: Get Harrison Fords’ birthdate and name from dbpedia and pets from our example dataset:

```
SELECT * where {
   VALUES ?subj {dbpedia:Harrison_Ford}
   ?subj tto:pet ?pet .
   SERVICE <http://dbpedia.org/sparql> {
       ?subj dbp:name ?name .
       ?subj dbp:birthDate ?bd .
      }
}
```

HTTP

The hypertext transfer protocol is one of the pillar of the web. This protocol is used by the browsers (Firefox, MSIE, Chrome, etc.) to communicate with the web servers to retrieve the pages to be displayed. It is used in other contexts as well in particular for the communication when a query is sent to a SPARQL endpoint.

Instance

Instance is a synonym for member. If we say that Rex is a dog, then we mean that Rex is a member or an instance of the abstract class *dog*. Formally the link between an instance and its class is expressed with the *rdf:type* property.
Example:
“Rex is a dog” :

```
ttr:RexDog rdf:type tto:Dog .
```

Literal

In RDF Schema *rdfs:Literal* is defined as a special kind of *rdfs:Resource*. Instances of *Literals* have no URIs, they are *literal* values like strings, numbers, date, etc. A literal instance always has a datatype.
Example:
The resource ttr:RexDog has name “Rex” and “Rex” has a xsd:string datatype.
The resource ttr:RexDog has weight “6.2” and “6.2” has a xsd:decimal datatype.

```
ttr:RexDog  dbp:name “Rex”^^xsd:string .
ttr:RexDog  tto:weight “6.2”^^xsd:decimal .
```

Namespace

A namespace is a resource (rdfs:Resource) making available a collection of descriptions. It is identified with an URI and it is possible to associate it with a prefix which can be used as a synonym for the URI. Using prefixes improves readability of RDF and decrease the size of resources.
Example:
The namespace <http://example.org/tuto/ontology#> is where the vocabulary of our example ontology is described. Terms defined in this namespace (tto:weight) can be referred to by using the prefix (tto:) associated to this namespace.

```
@prefix tto: <http://example.org/tuto/ontology#> .
 ttr:RexDog  tto:weight “6.2”^^xsd:decimal .
```

Ontology

In the context of semantic web an ontology is a formal way to describe terms (a vocabulary) that are useful to represent a knowledge domain. The vocabulary of an ontology includes classes which are used to classify resources and properties which are used to describe semantic relationships between resources.
Example:
The class tto:Dog and the property tto:pet are defined as terms of the ontology of our example dataset. Rex is defined as a dog ans as the pet of William.

```
ttr:RexDog rdf:type tto:Dog .
ttr:William tto:pet ttr:RexDog .
```

OWL

The Web Ontology Language (OWL) is an extension of the RDF Schema ontology. Like RDF Schema it is a knowledge representation language for authoring ontologies.
“OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs, e.g., to verify the consistency of that knowledge or to make implicit knowledge explicit. “ (http://www.w3.org/2001/sw/wiki/OWL).

Predicate

The term *Predicate* is more or less a synonym of *Property*. The term *Property* is generally used in the context of RDF data whilst the term *Predicate* is more often used in the context of SPARQL queries. See also *Property*.

Prefix

A prefix is a synonym for a namespace *URI*. It can be used both in RDF resource descriptions and in SPARQL queries. See also *Namespace*.
Example:
If tto: is declared as a prefix for the namespace <http://example.org/tuto/ontology#> with:

```
@prefix tto: <http://example.org/tuto/ontology#> .
```

Then writing tto.Dog is equivalent to write <http://example.org/tuto/ontology#Dog>

Property

In RDF schema *rdf:Property* is described as a special kind of *rdfs:Resource*. A property (or predicate) is used to state that some relationship between a subject and an object holds.
Example:
The property tto:pet can be used to describe the relationship between the resource ttr:William and the resource ttr:RexDog :

```
ttr:William tto:pet ttr:RexDog .
```

Range

In RDF Schema *rdfs:Range* is defined as a special kind of *rdf:Property*.
It is used to describe the semantics of properties. A property is a term defined in an ontology. The description of a property should define its range. The range of a property tells which kind of resources are supposed to be used as an object in triples involving the property.
Example:
The range of the tto:pet property is tto:Animal means that we expect resources used as an object in triples involving the *tto:pet* property to be instances of the class *tto:Animal*.

```
tto:pet rdfs:range tto:Animal .
```

RDF

The Resource Description Framework (RDF) is extensible knowledge representation language.
"RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a "triple"). Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications.
This linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes. This graph view is the easiest possible mental model for RDF and is often used in easy-to-understand visual explanations.
" (http://www.w3.org/RDF/).
See also W3C recommendation: http://www.w3.org/TR/rdf11-concepts/

RDF Schema (also RDFS)

"RDF Schema (Resource Description Framework Schema, variously abbreviated as RDFS, RDF(S), RDF-S, or RDF/S) is a set of classes with certain properties using the RDF extensible knowledge representation data model, providing basic elements for the description of ontologies, otherwise called RDF vocabularies, intended to structure RDF resources." (http://en.wikipedia.org/wiki/RDF_Schema).
See also W3C recommendation: http://www.w3.org/TR/rdf-schema/

rdf:type

rdf:type is a property defined in the rdf namespace. It is used in the description of a resource to tell that the resource belongs to some category or more formally that the resource is an instance of some class.
Example:
Harrison Ford (the resource) is a person (fits into the category or class *person*) :

```
dbpedia:Harrison_Ford rdf:type dbo:Person .
```

rdfs:range
See Range

rdfs:subClassOf
rdfs:subClassOf is a property defined in the rdfs namespace. It is used to organize classes defined in an ontology. If class1 is a subclass of class2, it means that class1 is more specific than class2. In other words all the resources belonging to class1 also belong to class2.
Example:
Assume that Rex is a dog :
`tto:Dog rdfs:subClassOf tto:Animal .`
Then if dog is defined as a subclass of animal :
`ttr:RexDog rdf:type tto:Dog .`
Then we expect Rex to be an animal as well.

Resource
In RDF a resource can represent a thing, an event, a concept, a term, etc. Anything identified with an URI is a resource and in RDF Schema the resource class is defined as the most general class. But literal values (numbers, string, date,...) are not resources.
Example:
<http://example.org/tuto/resource#RexDog> (equiv. to ttr:RexDog) is a resource.
<http://example.org/tuto/ontology#pet> (equiv. to tto:pet) is a resource
`“Rex”^^xsd:string or “6.5”^^xsd:decimal` are NOT resources

SNORQL
SNORQL is an utility for exploring RDF SPARQL endpoints. It was originally created by Richard Cyganaik (http://richard.cyganiak.de/). Many SPARQL endpoint providers make a SNORQL explorer available so that their datasets can be explored and easily queried.
Example:
The SNORQL explorer of dbpedia: http://dbpedia.org/snorql/
The SNORQL extension of neXtProt: http://snorql.nextprot.org/

SPARQL
SPARQL stands for **S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage. It is a semantic query language which makes it possible to retrieve and manipulate data stored in Resource Description Framework (RDF) format.
See also http://www.w3.org/TR/sparql11-overview/

SPARQL endpoint

A SPARQL endpoint is a service available on the web that accepts SPARQL queries as input and returns results as output.
Example:
http://dbpedia.org/sparql is the SPARQL endpoint of dbpedia

Statement

Statement is a synonym for triple. Or more precisely: in RDF a statement about the real world is encoded in a triple.
See *Triple*.

Subject

The subject is the first element of a triple. The triple encodes a statement describing the subject.

Thing

In RDF a thing is more or less a synonym for resource. The term is very generic because resources can refer to both concrete and abstract objects.
More formally, the Web Ontology Language (OWL), an extension of the RDF Schema, defines the class *owl:Thing* as the class of all the individuals (or instances): all the resources are members of the class *owl:Thing*.

Triple

The triple is the basic unit of information in RDF. It has three components: a subject, a predicate (or property) and an object. The assertion of an RDF triple says that some relationship, indicated by the predicate, holds between the resources denoted by the subject and object.

Triplestore

A triplestore is a database system for the storage of RDF data and the retrieval through SPARQL semantic queries.
Example:
OpenLink Virtuoso: http://virtuoso.openlinksw.com/

ttl

RDF data can be stored in the *turtle* format. By convention *ttl* is the name extension of files containing RDF data in turtle format.
Example:

```
my-rdf-data-file.ttl
```

Turtle

Turtle is one of the serialization format for RDF data. Its syntax is close to the syntax of SPARQL. It is more concise and human readable than the native XML format.
See also http://www.w3.org/TR/turtle/

URI

URI stands for Uniform Resource Identifier. An URI is a string of characters used to identify a resource over the the World Wide Web.
Example:
The URI for Harrison Ford in dbpedia: http://dbpedia.org/resource/Harrison_Ford