Ebiquity Research Group Semantic Web Reference Card v2.0 http://ebiquity.umbc.edu/

A1. RDF/RDFS Language

RDF Node

▪ **rdfs:Resource**		*the generic class of identified concept*
rdf:type	[rdfs:Resource → rdfs:Class]	*membership*
rdfs:label	[rdfs:Resource → rdfs:Literal]	*annotation*
rdfs:comment	[rdfs:Resource → rdfs:Literal]	*annotation*
rdfs:seeAlso	[rdfs:Resource → rdfs:Resource]	*annotation*
rdfs:isDefinedBy	[rdfs:Resource → rdfs:Resource]	*annotation*
rdf:value	[rdfs:Resource → rdfs:Resource]	*complex values*
▪ **rdfs:Literal**		*the generic class of literal values*
▪ **rdf:XMLLiteral**		*the class of typed literals (c.f. XMLSchema)*

Class

▪ **rdfs:Class**		*the class of rdf classes*
rdfs:subClassOf	[rdfs:Class → rdfs:Class]	*subset relation*

Property

▪ **rdf:Property**		*the class of properties(i.e. binary relations)*
rdfs:subPropertyOf	[rdf:Property→ rdf:Property]	
rdfs:domain	[rdf:Property→ rdfs:Class]	
rdfs:range	[rdf:Property→ rdfs:Class]	

Containers

▪ **rdfs:Container**	*the generic superclass of rdf resource containers*	
rdfs:member	[rdfs:Resource → rdfs:Resrouce]	*membership*
rdf:_1, rdf_2, …	Sub-properties of rdf:member	
▪ **rdf:Alt**		*container of alternatives*
▪ **rdf:Bag**		*unordered container*
▪ **rdf:Seq**		*ordered container*
▪ **rdfs:ContainerMembershipProperty**		*all sub-properties of rdfs:member*

List

▪ **rdf:List**		*the class of RDF Lists*
rdf:first	[rdf:List → rdfs:Resource]	car
rdf:rest	[rdf:List → rdfs:List]	cdr
▪ **rdf:nil**	an instance of RDF:List representing the empty list	

Datatype

▪ **rdfs:Datatype**		*the class of datatypes*

RDF Reification

▪ **rdf:Statement**		*the class of RDF statements*
rdf:subject	[rdf:Statement → rdfs:Resource]	
rdf:predicate	[rdf:Statement → rdfs:Resource]	
rdf:object	[rdf:Statement → rdfs:Resource]	

Supported XML datatypes

xsd:decimal xsd:double xsd:float xsd:int xsd:integer xsd:long xsd:short	xsd:negativeInteger xsd:positiveInteger xsd:nonPositiveInteger xsd:nonNegativeInteger xsd:unsignedLong xsd:unsignedInt xsd:unsignedShort	xsd:anyURI xsd:base64Binary xsd:boolean xsd:byte xsd:hexBinary xsd:unsignedByte	xsd:date xsd:dateTime xsd:time xsd:gYearMonth xsd:gYear xsd:gMonthDay xsd:gDay xsd:gMonth	xsd:string xsd:normalizedString xsd:token xsd:language xsd:NMTOKEN xsd:Name xsd:NCName

A2. OWL Web Ontology Language

Classes

▪ **owl:Class**	*all OWL classes, a sub-class of rdfs:Class*	
owl:equivalentClass	[owl:Class → owl:Class]	
owl:disjointWith *	[owl:Class → owl:Class]	
owl:oneOf *	[rdfs:Class → rdf:List]	
owl:intersectionOf -	[owl:Class → rdf:List]	
owl:unionOf *	[owl:Class → rdf:List]	
owl:complementOf *	[owl:Class → owl:Class]	
▪ **owl:Restriction**		
owl:onProperty	[owl:Restriction → rdf:Property]	
owl:allValuesFrom	[owl:Restriction → rdfs:Class]	
owl:someValuesFrom	[owl:Restriction → rdfs:Class]	
owl:hasValue *	[owl:Restriction →]	*no range constraint*
owl:cardinality -	[owl:Restriction → xsd:nonNegativeInteger]	
owl:maxCardinality -	[owl:Restriction → xsd:nonNegativeInteger]	
owl:minCardinality -	[owl:Restriction → xsd:nonNegativeInteger]	
▪ **owl:DataRange ***	sets of data values, range of data-valued property	
▪ **owl:DeprecatedClass**		*version control*

Properties

▪ **owl:DatatypeProperty**		*range is instance of rdfs:Datatype*
▪ **owl:ObjectProperty**		*range is instance of owl:Class*
owl:inverseOf	[owl:ObjectProperty → owl:ObjectProperty]	
▪ **owl:OntologyProperty**		*domain/range are owl:Ontology*
▪ **owl:AnnotationProperty**		*range is rdfs:Literal*
▪ **owl:FunctionalProperty**		*(s,p,o1) , (s, p,o2) =>sameAs(o1, o2)*
▪ **owl:InverseFunctionalProperty**		*(s1,p,o) , (s2, p,o) => sameAs(s1, s2)*
▪ **owl:SymmetricProperty**		*(s,p,o) => (o,p,s)*
▪ **owl:TransitiveProperty**		*(a,p,b) (b,p,c) => (a,p,c)*
▪ **owl:DeprecatedProperty**		*version control*
owl:equivalentProperty	[rdf:Property → rdf:Property]	

Special classes

▪ **owl:Thing**		*all OWL individuals*
owl:differentFrom	[owl:Thing → owl:Thing]	
owl:sameAs	[owl:Thing → owl:Thing]	
▪ **owl:Nothing**		*the complement of owl:Thing*
▪ **owl:AllDifferent**		*OWL built-in*
owl:distinctMembers	[owl:AllDifferent → rdf:List]	*OWL built-in*

Ontology

▪ **owl:Ontology**		*ontology description*
owl:backwardCompatibleWith	[owl:Ontology → owl:Ontology]	
owl:imports	[owl:Ontology → owl:Ontology]	
owl:incompatibleWith	[owl:Ontology → owl:Ontology]	
owl:priorVersion	[owl:Ontology → owl:Ontology]	
owl:versionInfo	[→]	*no domain or range constraint*

notations:
** means only not supported by OWL Lite.*
- means supported in OWL Lite with restrictions

A3. RDF/XML Syntax Language

Reserved Terms

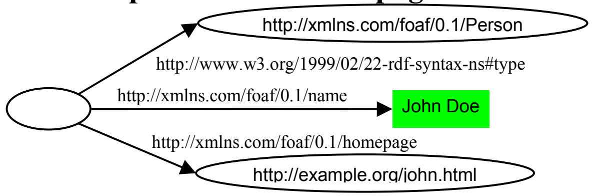
▪ **rdf:RDF**	*special XML element containing a serialized RDF graph*
▪ **rdf:Description**	*node element*
▪ **rdf:resource**	*leaf node element in XML parse tree*
▪ **rdf:ID**	*ID of node, local name, augmented by xml:base (global)*

note: the rdf:ID in property element will add a reified RDF statement for the triple

▪ **rdf:about**	*ID of node, URIref, like hyperlink, (global)*
▪ **rdf:nodeID**	*ID of blank node, local name (local)*
▪ **rdf:datatype**	*shows the object node of a predicate is a typed literal*
▪ **rdf:parseType="Literal"**	*what follows should be parsed as literal*
▪ **rdf:parseType="resource"**	*omits a blank node (predicate→predicate)*
▪ **rdf:parseType="Collection"**	*lets property element contain multiple nodes*
▪ **rdf:li**	*container membership, similar to rdf:_1, rdf:_2…*
▪ **xml:base**	*applies to rdf:about, rdf:resource, rdf:ID and rdf:datatype*
▪ **xml:lang**	*identification of content language*

source: http://www.w3.org/TR/rdf-syntax-grammar/

A4. Examples - John's homepage


http://xmlns.com/foaf/0.1/Person
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://xmlns.com/foaf/0.1/name
John Doe
http://xmlns.com/foaf/0.1/homepage
http://example.org/john.html

(RDF/XML version)

```
<?xml version="1.0" ?>
<rdf:RDF   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:foaf = "http://xmlns.com/foaf/0.1/" >
<foaf:Person>
  <foaf:name>John Doe</foaf:name>
  <foaf:homepage>
     <rdf:Description rdf:about="http://example.org/john.html"  />
  </foaf:homepage>
</foaf:Person>
</rdf:RDF>
```

(N3 version)

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

 [ ]   a       foaf:Person ;
       foaf:homepage    <http://example.org/john.html> ;
       foaf:name      "John Doe" .
```

(NTriples version)

Line1:	_:x <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
Line2:	_:x <http://xmlns.com/foaf/0.1/homepage> <http://example.org/john.html> .
Line3:	_:x <http://xmlns.com/foaf/0.1/name> "John Doe".

http://ebiquity.umbc.edu/ Semantic Web Reference Card v2.0 Ebiquity Research Group

B1. SPARQL (based on W3C WD 2005-July-21)

Concepts

IRI *(see RFC3987)*	"Internationalized Resource Identifiers", generalization of URI. ▪ <http://foo.com/john> -- IRI delimited by "<>" ▪ foaf:name -- IRI represented by prefix+localName
blank node	drawn from RDF graph. Indicated by "_:", e.g. _:b12
RDF Literal	string with optional tags, e.g. ▪ "chat" ▪ "chat"@fr - string with **language tag** "fr" ▪ "abc"^^myNS:myType - literal with customized **type tag** ▪ 1 - the same as "1"^^xsd:integer ▪ 1.0e6 - the same as "1.0e6"^^xsd:double ▪ true -- the same as "true"^^xsd:boolean
RDF Term (RDF-T)	the union of all IRIs, blank nodes, and RDF Literals
Query Variables (V)	symbols disjoint from RDF Term. Lead by "?", e.g. ?name *Note: Variables in SPARQL query have global scope.*
triple pattern	member of (RDF-T union V) x (I union V) x (RDF-T union V) each triple pattern is terminated by "."
graph pattern	a set of triple patterns
RDF dataset	one default graph merging graphs referred in FROM clauses; a set of (IRI, graph) pairs referred in FROM NAME clauses
solution modifiers	projection, distinct, order, limit, offset
result forms	including: SELECT, CONSTRUCT, DESCRIBE, ASK
SPARQL Query	(graph pattern, RDF dataset, solution modifiers, result form)

Query 1 (search top 5-10 named persons younger than 30)

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
PREFIX  ns:  <http://example.org/ns#>
SELECT  ?name ?mbox ?age
WHERE   {  ?x    foaf:name  ?name.
           ?x    ns:age ?age . FILTER (?age < 30).
           OPTIONAL { ?x    foaf:mbox   ?mbox .}
}
ORDER BY  ?age  DESC(?name)
OFFSET 5
LIMIT  10
```

Query 2 (construct vCard RDF graph from FOAF data)

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
PREFIX vcard:   <http://www.w3.org/2001/vcard-rdf/3.0#>

CONSTRUCT {       ?x      vcard:N     _:v .
                 _:v     vcard:givenName     ?gname .
                 _:v     vcard:familyName    ?fname }
WHERE
{
   { ?x   foaf:firstname ?gname } UNION  { ?x   foaf:givenname   ?gname } .
   { ?x   foaf:surname   ?fname } UNION  { ?x   foaf:family_name ?fname } .
 }
```

Query3 (query three RDF datasets with graph pattern)

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?who ?g ?mbox
FROM <http://example.org/dft.ttl>
FROM NAMED <http://example.org/alice>
FROM NAMED <http://example.org/bob>
WHERE
{
  ?g dc:publisher ?who .
  GRAPH ?g { ?x    foaf:mbox   ?mbox }
}
```

Keywords

a	short form of "rdf:type"
true, false	boolean constants
ASK	tests whether or not a query pattern has a solution.
ASC/ DESC	sorts results. The default order is ASC.
BASE	the base URL for resolving relative IRIs in SPARQL query
CONSTRUCT	returns one RDF graph specified by the graph template
DESCRIBE	returns one RDF graph containing describing resources. *note: this concept is vague and depends on specific query processors.*
DISTINCT	ensures solutions in the sequence are unique.
FILTER	eliminates solutions result in false. An operand is false when ▪ it is unbound. ▪ it is an xs:boolean with a FALSE value. ▪ it is a 0-length untyped RDF literal or xs:string. ▪ it is any numeric type with a value of 0. ▪ it is an xs:double or xs:float with a value of NaN
FROM	selects an RDF graph to be merged to the default RDF graph
FROM NAMED	selects a named RDF graph, for GRAPH constraint
GRAPH	indicates pattern on a named RDF graph
LIMIT	restricts the number of solutions processed for query results
OFFSET	generates solutions after the specified number of solutions
OPTIONAL	generates additional bindings when patterns can be matched
ORDER BY	puts the solutions in order
PREFIX	facilitates a QName-like syntax for shorter forms of IRIs. *note: prefixes may be used anywhere after they are declared.*
SELECT	returns results in projected form, i.e. variable bindings *note: SELECT * is an abbreviation that selects all named variables.*
UNION	combines alternative grouped graph patterns. *note: grouped graph patterns are matched conjunctively by default.*
WHERE	indicates where-clause, which specifies graph pattern.

note: keywords are shown in uppercase but are matched in a case-insensitive manner.

Other syntactic forms (for abbreviation purpose)

1. Predicate-Object Lists: use ";" for common subject.

?x foaf:name ?name ; foaf:mbox ?mbox .	?x foaf:name ?name . ?x foaf:mbox ?mbox .

note: the two set of triple patterns have the same meaning

2. Object Lists: use "," to separate objects sharing same subject, predicate

?x foaf:name "finin" , "finin"@en .	?x foaf:name "finin" . ?x foaf:name "finin"@en .

3. blank nodes: use "[" and "]" to bound scope of a blank node as common subject.

[:p "v"] .	[] :p "v" .	_:b57 :p "v" .

another example – triples sharing the same blank node as subject.

[foaf:name ?name ; foaf:mbox <alice@example.org>]	_:b18 foaf:name ?name . _:b18 foaf:mbox <alice@example.org> .

4. RDF Collections: use "(" and ")" to bound a list.

e.g. (1 ?x 3) refers to a three-element collection represented by RDF:List.

Operators

||, &&	binary, combine logic expressions
=, !=	binary, compare RDF Terms
>, <, <=, >=	binary, compare numeric or dataTime typed-literal
+, -, * , /	binary, math operator
!, -, +,	unary, for logic or numeric expression
STR	unary, cast to string form
DATATYPE	unary, obtain typed-literal's datatype
LANG	unary, obtain literal's language
BOUND	unary, test if variable is bound to a value
isURI, isBLANK, isLITERAL	unary, test if variable is bound to a URI, a blank node, or a literal respectively
REGEX	(string, pattern [, flags]) regular expression match

B2. Facts for practice

Popular Ontologies and Terms

cc	http://web.resource.org/cc/ *(Creative Commons)* C Agent C License P license
daml	http://www.daml.org/2001/03/daml+oil#
dc	http://purl.org/dc/elements/1.1/ *(Dublin Core Element Set 1.1)* P date P creator P language P subject P description P title P source
dcterms	http://purl.org/dc/terms/ *(Dublin Core Terms)* P created P issued
foaf	http://xmlns.com/foaf/0.1/ *(Friend Of A Friend)* C Agent C Person C Document C Image P mbox_sha1sum P name P homepage P mbox P knows P nick P depiction P firstName P surname P interest
geo	http://www.w3.org/2003/01/geo/wgs84_pos# C Point P lat P long
iw	http://inferenceweb.stanford.edu/iw.owl# *(Inference Web)*
owl	http://www.w3.org/2002/07/owl#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
rss	http://purl.org/rss/1.0/ *(RDF Site Summary 1.0)* C channel C image C item P items P description P link P title
ruml	http://www.w3.org/2003/11/ruleml#
service	http://www.daml.org/services/owl-s/1.1/Service.owl# C ServiceModel C ServiceProfile C ServiceGrounding P presents P describedBy P supports
swrl	http://www.w3.org/2003/11/swrl#
wn	http://xmlns.com/wordnet/1.6/ *(WordNet 1.6)*

Popular Tools

- RDF/OWL editors: Swoop, Protégé
- Semantic Web Search Engines: Swoogle, Semantic Web Search, SchemaWeb
- RDF APIs: Jena's ARP, Redland, Sesame's Rio, RDF-Lib
- RDF/OWL Reasoners: Jena, Pellet, FaCT++, Racer, JTP
- RDF/OWL database: Kowari, Sesame, 3store, Jena, IBM Minerva, Oracle 10g
- Integrated toolkit: IBM IODT (eclipse plug-in), Jena, Sesame

Tips

- MIME type for RDF/XML is "application/rdf+xml".
- File extensions: RDF/XML→ .rdf, N3→ .n3, NTriples→ .nt, OWL→ .owl
- Some XML based tools only support ANSI encoding but not UTF-8 encoding

An example ontology written in RDF/XML

```
<?xml version="1.0"   encoding="utf-8"?>
<rdf:RDF   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
           xmlns:owl="http://www.w3.org/2002/07/owl#"
           xml:base = "http://example.org/ex.owl">
<owl:Ontology rdf:about="">
   <rdfs:label xml:lang="en-US">Example Ontology</rdfs:label>
   <owl:imports rdf:resource="http://xmlns.com/foaf/0.1/" />
</owl:Ontology>
<owl:Class rdf:ID="RDFDocument">
  <rdfs:subClassOf  rdf:resource="http://xmlns.com/foaf/0.1/Document"/>
  <rdfs:comment xml:lang="en-US"> All RDF documents. </rdfs:comment>
</owl:Class>
</rdf:RDF>
```

Authored by Li Ding and Tim Finin

Version 2.0 (2005-11-04)

Partial support by DARPA contract F30602-00-0591 and
NSF awards NSF-ITR-IIS-0326460 and NSF-ITR-IDM-0219649.

Please send comments to feedback@ebiquity.umbc.edu
Copies available at http://ebiquity.umbc.edu/v2.1/resource/html/id/94/