This document is based on W3C.

Uniform Resource Identifier(URI): an ASCII string that conforms to the syntax defined in [RFC 3986]:
https://tools.ietf.org/html/rfc3986#section-3

internationalized Resource Identifier(IRI): a unicode string that conforms to the syntax defined in [RFC 3987]:
http://www.ietf.org/rfc/rfc3987.txt
IRIs are extensions of URIs.
IRIs are denoted by angle brackets <> in this note.

Definition. (RDF, Resource Description Framework)
RDF is a formal language with a semantics for describing structured information[1](especially in web).

Note.
1. RDF is both a name of a formal language and the name of its conceptual data model .
2. RDF can have multiple concrete syntaxes(RDF/XML, RDFa, Turtle, etc.). A concrete syntax of RDF is called a **serialization syntax**, since it allows transformation from complex data structures(RDF graphs) into linear strings.

The RDF data model is a syntax-independent way of representing RDF statements.

RDF conceptual data model

Concept. (RDF data model)
The RDF data model is a tuple (R, L, P, S) where R, L, P, S are four sets called resources, literals, properties and statements, respectively. Thereinto,
1. R can be further divided into two groups: the set of blank nodes and the set of resources identified by IRIs.

[1] p19 of《Foundations of semantic web technologies》, Pascal Hitzler, et.al.

2. P is a subset of R
3. each element of S is a triple of the form {predicate, subject, object}, where predicate is an IRI denoting a property (an element of P), subject is a blank node or IRI of a resource, and object is a blank node, IRI denoting a resource or literal(an element of L).

Remark.

1. a **blank node** is a **resource without a global identifier**, just like a variable in algebra. Each blank node is unique. Statments involving blank nodes either say that something with the given relationships exists, without explicitly naming it[2].
2. the resource of an IRI is called the **referent** of the IRI[3].
3. A property stands for a predicate. Any $n-$ary relationship can be represented by using n predicates(just like the reification showed below).

Literals have datatypes that define the range of possible values, such as strings, numbers, and dates.

Concept. (RDF datatype)
A datatype consists of a lexical space, a value space and a lexical-to-value mapping, denoted by one or more IRIs.
A **lexical space** is a set of Unicode strings.
The **lexical-to-value mapping** of a datatype is a set of pairs whose first element is from lexical space, and the second element from the value space. Each member of the lexical space is paired with exactly one value, and is a lexical representation of that value. The mapping can be seen as a function from the lexical space to the value space.

E.g. the XML Schema datatype xsd:boolean, where each member of the value space has two lexical representations, is defined as follows:
Lexical space:
{“true”, “false”, “1”, “0”}
Value space:

---

[2] https://www.w3.org/TR/rdf11-primer/#section-blank-node

[3] For guidelines for determining the referent of an IRI, see https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#referents

{true, false}

Lexical-to-value mapping：

{ <“true”, true>, <“false”, false>, <“1”, true>, <“0”, false>}

Concept. (Literal)[4]

a literal is an element of L consisting of 2 or 3 elements:

- a Unicode string in normal form C, called the **lexical form**;
- a datatype IRI determining how the lexical form maps to a literal value
- a non-empty, well formed language tag defined by BCP47, if the datatype IRI is http://www.w3.org/1999/02/22-rdf-syntax-ns#langString. And in this case, the literal is called a **language-tagged string**.

In Turtle(a concrete RDF syntax), a langage-tagged string may look like “test”@en, “à”@fr, where @en means that the string is in English, and @fr means that the string is in French.

The **literal value associated with a literal** is:

1. If the literal is a language-tagged string, then the literal value is a pair consisting of its lexical form and its language tag, in that order.
2. If the literal's datatype IRI is in the set of recognized datatype IRIs, let d be the referent of the datatype IRI.
    - If the literal's lexical form is in the lexical space of d, then the literal value is the result of applying the lexical-to-value mapping of d to the lexical form.
    - Otherwise, the literal is **ill-typed** and no literal value can be associated with the literal. Such a case produces a semantic inconsistency but is not syntactically ill-formed. Implementations must accept ill-typed literals and produce RDF graphs from them. Implementations may produce warnings when encountering ill-typed literals.
3. If the literal's datatype IRI is not in the set of recognized datatype IRIs, then the literal value is not defined by this specification.

Two literals are **term-equal** (the same RDF literal) if and only if all of their components listed above are the same.

---

[4] here is just a brief and partial definition. see https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#section-Graph-Literal

Concept. (RDF graph)
The set S can be viewed as a directed labeled graph $G(V, E)$, called a **RDF graph:** a vertex(elements of V) can be an IRI/literal/blank node; a triple {p, s, o} is an arc from s to o, labeled by p.
This can be read either 『o is the value of p for s』
or 『s has a property p with a value o』
or even 『the p of s is o』 .

For a IRI or a blank node s, if there is a statement {p, s, o} in S, then we also say s **has a property** p.

Concept. (RDF term & term equality)
IRIs, literals and blank nodes are collectively known as **RDF terms**. Two RDF terms are said term-equal iff
(1) For two IRIs, they are equivalent under simple string comparison(no normalization of IRIs should be performed).
(2) For two literals, they have the same lexical form(character by character), the same datatype IRI, and the same language tag(if any). In this case, these two literals are also called **literal-equal**.
(3) RDF makes no reference to any internal structure of blank nodes. Given two blank nodes, it is possible to determine whether or not they are the same. For two blank nodes in concrete RDF syntax like Turtle, they are term-equal iff they have the same identifiers.
Otherwise, they are not term-equal.

sometimes it's desirable to work with multiple RDF graphs while keeping their contents separate. RDF datasets support this requirement.

Concept. (RDF dataset)
A RDF dataset is a collection of RDF graphs subsuming 0 or more graphs associated with an IRI or blank node, called the **named graphs**, and one graph without called the **default graph**.
see
https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#section-dataset

RDF built-in vocabulary

An **RDF vocabulary** is a collection of IRIs[5] with a clearly defined meaning, intended for use in RDF graphs.

The IRIs in an RDF vocabulary often begin with a common substring known as a **namespace IRI**. Some namespace IRIs are associated by convention with a short name known as a namespace prefix.

Some example namespace prefixes and IRIs(from W3C)

Namespace prefix	Namespace IRI	RDF vocabulary
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	The RDF built-in vocabulary
rdfs	http://www.w3.org/2000/01/rdf-schema#	The RDF Schema vocabulary
xsd	http://www.w3.org/2001/XMLSchema#	The RDF-compatible XSD types

Suppose the namespace prefix for a namespace IRI <r/> is p. Then we can use p:name to express <r/name>.

The **RDF built-in vocabulary**, or basic RDF vocabulary, consists of the following(actually some IRIs can be better illustrated using RDF Schema vocabulary[6]):

Definition. (type property)

There is an element of *Properties* known as rdf:type.

Members of *Statements* of the form {rdf:type, sub, obj} must satisfy that sub and obj are members of *Resources.* The vocabulary RDF schema places additional restrictions on the use of type.

Definition. (reification)

There is an element of *Resources*, not contained in *Properties*, known as rdf:Statement.

---

[5] https://www.w3.org/TR/rdf11-concepts/#vocabularies

[6] The ontology of RDF built-in vocabulary described via RDF Schema: https://www.w3.org/1999/02/22-rdf-syntax-ns#

There are 3 elements in *Properties* known as rdf:predicate, rdf:subject and rdf:object.

Reification of a statement s={pred, sub, obj} is a resource $r \in R$, called the **reified statement**, representing s such that S has the following 4 statements:

s_1: {rdf:predicate, r, pred}

s_2: {rdf:subject, r, subj}

s_3: {rdf:object, r, obj}

s_4: {rdf:type, r, [rdf:Statement]}

(these 4 statements should always occur together)

With reification we can label the properties of a statement; like the author, the date of creation, etc.

it is frequently necessary to represent a collection of resources or literals; for example to state that a property has an ordered sequence of values.

Definition. (Container)

There are three elements of *Resources*, not contained in *Properties*, known as rdf:Seq (ordered lists), rdf:Bag (unordered lists), and rdf:Alt (alternatives for single value of a property).

There is a subset of *Properties* corresponding to the ordinals (1, 2, 3, ...) called *Ord*. We refer to elements of *Ord* as RDF:_1, RDF:_2, RDF:_3, …

RDF Schema

RDF Schema is an RDF vocabulary(viewed as an extension of built-in RDF vocabulary), used to describe properties of other RDF resources(including properties) which define application-specific RDF vocabularies.

“The RDF Schema class and property system is similar to the type systems of object-oriented programming languages such as Java. RDF Schema differs from many such systems in that instead of defining a class in terms of the properties its instances may have, RDF Schema describes properties in terms of the classes of resource to which they apply.”

Class System
A class is a resource, identified by an IRI and associated with a set called the **class extension** of the class, which is a set of resources. Each member of the class extension is called an instance of the class. Two different classes may have the same set of instances, yet to have different properties.

If a class C is a ***subclass*** of a class C', then all instances of C will also be instances of C'.

Datatype System[7]
In RDF Schema, a datatype is a class of which the instance is a member of the value space of the datatype.

rdfs:Datatype is the class of all datatypes. All instances
of rdfs:Datatype correspond to the concept "datatype" in RDF data model.

Core classes in RDF Schema:
- rdfs:Resource, the class of all things described by RDF(**so all other classess are subclasses of this class**).
- rdfs:Class, the class of all classes
- rdfs:Literal, the class of all literal **values**
- rdfs:Datatype,the class of datatypes. Each instance of rdfs:Datatype is a subclass of rdfs:Literal.
- rdf:Property, the class of all properties
- rdf:Statement, the class of all reified statements
- rdf:type: an instance of rdf:Property that is used to state that a resource is an instance of a class.

Properties System
The **domain** of a property is the set of resources to which the property can be applied (as subjects).
The **range** of a property is the set of resources or literals to which the property can be applied (as objects).

A property P is a **subproperty** of property P' iff
$\{\{P', s, o\} \mid \{P, s, o\} \in S\} \subset S.$
And P' is called a **superproperty** of P.

Core properties

[7] https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#section-Graph-Literal

- rdfs:subClassOf, which relates a class to one of its superclasses. All instances of a class are instances of its superclass. Note that a class may be a subclass of more than one class. As an example, the class femaleProfessor may be a subclass of both female and professor.
- rdfs:subPropertyOf, which relates a property to one of its superproperties.

Core Properties for Restricting Properties
- rdfs:domain, which specifies the domain of a property P and states that any resource that has a given property is an instance of the domain classes.
- rdfs:range, which specifies the range of a property P and states that the values of a property are instances of the range classes.

some collections in RDF Schema can be seen:
https://www.w3.org/TR/2014/REC-rdf-schema-20140225/#ch_collectionvocab

Utility Properties
- rdfs:seeAlso relates a resource to another resource that explains it.
- rdfs:isDefinedBy is a subproperty of rdfs:seeAlso and relates a resource to the place where its definition, typically an RDF schema, is found.
- rdfs:comment. Comments, typically longer text, can be associated with a resource.
- rdfs:label. A human-friendly label (name) is associated with a resource. Among other purposes, it may serve as the name of a node in a graphic representation of the RDF document.

the complete RDF Schema can be viewed in
https://www.w3.org/TR/2014/REC-rdf-schema-20140225/#ch_summary

References

1. The semantic web. Tim Berners-Lee, James Hendler and Ora Lassila
2. https://www.w3.org/TR/rdf11-concepts/
3. https://www.w3.org/TR/2014/REC-rdf-schema-20140225/
4. https://tools.ietf.org/pdf/rfc3986.pdf
5. https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/#bib-RDF11-SCHEMA
6. https://www.w3.org/TR/1999/REC-rdf-syntax-19990222