W3C Recommendation

SPARQL 1.1 Overview

W3C Recommendation 21 March 2013

This version:
http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/
Latest version:
http://www.w3.org/TR/sparql11-overview/
Previous version:
http://www.w3.org/TR/2012/PR-sparql11-overview-20121108/
Editor:
The W3C SPARQL Working Group, see Acknowledgements <public-rdf-dawg-comments@w3.org>

Please refer to the **errata** for this document, which may include some normative corrections.

See also **translations**.

Abstract

This document is an overview of SPARQL 1.1. It provides an introduction to a set of W3C specifications that facilitate querying and manipulating RDF graph content on the Web or in an RDF store.

Status of this Document

May Be Superseded

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

Set of Documents

This document is one of eleven SPARQL 1.1 Recommendations produced by the SPARQL Working Group:

1. SPARQL 1.1 Overview (this document)
2. SPARQL 1.1 Query Language
3. SPARQL 1.1 Update
4. SPARQL1.1 Service Description
5. SPARQL 1.1 Federated Query
6. SPARQL 1.1 Query Results JSON Format
7. SPARQL 1.1 Query Results CSV and TSV Formats
8. SPARQL Query Results XML Format (Second Edition)
9. SPARQL 1.1 Entailment Regimes
10. SPARQL 1.1 Protocol
11. SPARQL 1.1 Graph Store HTTP Protocol

No Substantive Changes

There have been no substantive changes to this document since the previous version. Minor editorial changes, if any, are detailed in the change log and visible in the color-coded diff.

Please Send Comments

Please send any comments to public-rdf-dawg-comments@w3.org (public archive). Although work on this document by the SPARQL Working Group is complete, comments may be addressed in the errata or in future revisions. Open discussion is welcome at public-sparql-dev@w3.org (public archive).

Endorsed By W3C

This document has been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and is endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited from another document. W3C's role in making the Recommendation is to drawattention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

Patents

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. W3C maintains a public list of any patent

disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

---

1 Introduction

SPARQL 1.1 is a set of specifications that provide languages and protocols to query and manipulate RDF graph content on the Web or in an RDF store. The standard comprises the following specifications:

- SPARQL 1.1 Query Language - A query language for RDF.
- SPARQL 1.1 Query Results JSON Format and SPARQL 1.1 Query Results CSV and TSV Formats - Apart from the standard SPARQL Query Results XML Format [SPARQL-XML-Result], SPARQL 1.1 now allows three alternative popular formats to exchange answers to SPARQL queries, namely JSON, CSV (comma separated values) and TSV (tab separated values) which are described in these two documents.
- SPARQL 1.1 Federated Query - A specification defining an extension of the SPARQL 1.1 Query Language for executing queries distributed over different SPARQL endpoints.
- SPARQL 1.1 Entailment Regimes - A specification defining the semantics of SPARQL queries under entailment regimes such as RDF Schema, OWL, or RIF.
- SPARQL 1.1 Update Language - An update language for RDF graphs.
- SPARQL 1.1 Protocol for RDF - A protocol defining means for conveying arbitrary SPARQL queries and update requests to a SPARQL service.
- SPARQL 1.1 Service Description - A specification defining a method for discovering and a vocabulary for describing SPARQL services.
- SPARQL 1.1 Graph Store HTTP Protocol - As opposed to the full SPARQL protocol, this specification defines minimal means for managing RDF graph content directly via common HTTP operations.
- SPARQL 1.1 Test Cases - A suite of tests, helpful for understanding corner cases in the specification and assessing whether a system is SPARQL 1.1 conformant

1.1 Example

In the following, we will illustrate the use of SPARQL's languages, protocols, and related specifications with a small example.

Some RDF graph published on the Web at the URL 'http://example.org/alice' contains personal information about Alice and her social contacts. We use Turtle [Turtle] syntax here for illustration.

Graph: http://example.org/alice

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://example.org/alice#me> a foaf:Person .
<http://example.org/alice#me> foaf:name "Alice" .
<http://example.org/alice#me> foaf:mbox <mailto:alice@example.org> .
<http://example.org/alice#me> foaf:knows <http://example.org/bob#me> .
<http://example.org/bob#me> foaf:knows <http://example.org/alice#me> .
<http://example.org/bob#me> foaf:name "Bob" .
<http://example.org/alice#me> foaf:knows <http://example.org/charlie#me> .
<http://example.org/charlie#me> foaf:knows <http://example.org/alice#me> .
<http://example.org/charlie#me> foaf:name "Charlie" .
<http://example.org/alice#me> foaf:knows <http://example.org/snoopy> .
<http://example.org/snoopy> foaf:name "Snoopy"@en .
```

With SPARQL 1.1 one can query such graphs, load them into RDF stores and manipulate them in various ways.

2 SPARQL 1.1 Query Language

Assuming the graph data from above is loaded into a SPARQL service (i.e., an HTTP service endpoint that can process SPARQL queries), the SPARQL 1.1 Query Language can be used to formulate queries ranging from simple graph pattern matching to complex queries. For instance, one can ask using a SPARQL SELECT query for names of persons and the number of their friends:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name (COUNT(?friend) AS ?count)
WHERE {
    ?person foaf:name ?name .
    ?person foaf:knows ?friend .
} GROUP BY ?person ?name
```

Just like in the earlier SPARQL1.0 specification [SPARQL-Query] from 2008, complex queries may include union, optional query parts, and filters; new features like value aggregation, path expressions, nested queries, etc. have been added in SPARQL 1.1. Apart from SELECT queries - which return variable bindings - SPARQL supports ASK queries - i.e. boolean "yes/no" queries - and CONSTRUCT queries - by which

new RDF graphs can be constructed from a query result; all the new query language features of SPARQL 1.1 are likewise usable in ASK and CONSTRUCT queries.

Compared to SPARQL1.0, SPARQL 1.1 adds a number of new features to the query language, including subqueries, value assignment, path expressions, or aggregates - such as COUNT, as used in the above example query - etc.

The SPARQL 1.1 Query Language document defines the syntax and semantics of SPARQL 1.1 queries and provides various examples for their usage.

3 Different query results formats supported by SPARQL 1.1 (XML, JSON, CSV, TSV)

Results of SELECT queries in SPARQL comprise bags of mappings from variables to RDF terms, often conveniently represented in tabular form. For instance, the query from Section 2 has the following results:

?name	?count
"Alice"	3
"Bob"	1
"Charlie"	1

In order to exchange these results in machine-readable form, SPARQL supports four common exchange formats, namely the Extensible Markup Language (XML), the JavaScript Object Notation (JSON), Comma Separated Values (CSV), and Tab Separated Values (TSV). These results formats are described in three different documents:

- the SPARQL Query Results XML Format (*please, particularly note that the SPARQL 1.1 WG has made some minor Errata to this specification*),
- the SPARQL 1.1 Query Results JSON Format, and
- the SPARQL 1.1 Query Results CSV and TSV Formats

These documents specify details of how particular solutions and RDF terms occurring in solutions are encoded in the respective target formats.

The results of our example query, in these three formats look as follows.

XML:

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
 <head>
   <variable name="name"/>
   <variable name="count"/>
 </head>
 <results>
   <result>
     <binding name="name">
       <literal>Alice</literal>
     </binding>
     <binding name="count">
       <literal datatype="http://www.w3.org/2001/XMLSchema#integer">3</literal>
     </binding>
   </result>
   <result>
     <binding name="name">
       <literal>Bob</literal>
     </binding>
     <binding name="count">
       <literal datatype="http://www.w3.org/2001/XMLSchema#integer">1</literal>
     </binding>
   </result>
   <result>
     <binding name="name">
       <literal>Charlie</literal>
     </binding>
     <binding name="count">
       <literal datatype="http://www.w3.org/2001/XMLSchema#integer">1</literal>
     </binding>
   </result>
 </results>
</sparql>
```

JSON:

```
{
  "head": {
    "vars": [ "name" , "count" ]
  } ,
  "results": {
    "bindings": [
      {
        "name": { "type": "literal" , "value": "Alice" } ,
        "count": { "datatype": "http://www.w3.org/2001/XMLSchema#integer" , "type": "typed-literal" , "value": "3" }
      } ,
      {
        "name": { "type": "literal" , "value": "Bob" } ,
        "count": { "datatype": "http://www.w3.org/2001/XMLSchema#integer" , "type": "typed-literal" , "value": "1" }
      } ,
      {
        "name": { "type": "literal" , "value": "Charlie" } ,
        "count": { "datatype": "http://www.w3.org/2001/XMLSchema#integer" , "type": "typed-literal" , "value": "1" }
      }
    ]
  }
}
```

CSV:

```
name,count
Alice,3
Bob,1
Charlie,1
```

TSV:

```
?name<TAB>?count
"Alice"<TAB>3
"Bob"<TAB>1
"Charlie"<TAB>1
```

(***Note:*** *tab characters are visually marked with* '`<TAB>`' *here for illustration only.*)

4 SPARQL 1.1 Federated Query

The SPARQL 1.1 Federated Query document describes an extension of the basic SPARQL 1.1 Query Language to explicitly delegate certain subqueries to different SPARQL endpoints.

For instance, in our example, one may want to know whether there is anyone among Alice's friends with the same name as the resource identified by the IRI <http://dbpedia.org/resource/Snoopy> at DBpedia. This can be done by combining a query for the names of friends with a remote call to the SPARQL endpoint at `http://dbpedia.org/sparql` finding out the name of <http://dbpedia.org/resource/Snoopy> using the `SERVICE` keyword as follows:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
    <http://example.org/alice#me> foaf:knows [ foaf:name ?name ] .
    SERVICE <http://dbpedia.org/sparql> { <http://dbpedia.org/resource/Snoopy> foaf:name ?name }
}
```

with the following result:

?name
"Snoopy"@en

Here, the first part of the pattern in the `WHERE` part is still matched against the local SPARQL service, whereas the evaluation of the pattern following the `SERVICE` keyword is delegated to the respective remote SPARQL service.

5 SPARQL 1.1 Entailment Regimes

SPARQL could be used together with ontological information in the form of, for example, RDF Schema or OWL axioms. For instance, let us assume that - apart from the data about Alice - some ontological information in the form of RDF Schema [RDF-Schema] and OWL [OWL2-Overview] constructs defining the FOAF vocabulary is loaded into our example SPARQL service.

The FOAF ontology: (only an excerpt given)

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
...
foaf:name rdfs:subPropertyOf rdfs:label .
...
```

The following query asks for labels of persons:

```
SELECT ?label
WHERE { ?person rdfs:label ?label }
```

A SPARQL engine that does not consider any special entailment regimes (on top of standard simple entailment) would not return any results for this query, whereas an RDF Schema aware query engine will return

?label
"Alice"
"Bob"
"Charlie"
"Snoopy"@en

since `foaf:name` is a sub-property of `rdfs:label`.

The SPARQL 1.1 Entailment Regimes specification defines which answers should be given under which entailment regime, specifying entailment regimes for RDF, RDF Schema, D-Entailment [RDF-MT], OWL [OWL2-Overview], and RIF [RIF-Overview].

6 SPARQL 1.1 Update Language

The SPARQL 1.1 Update specification defines the syntax and semantics of SPARQL 1.1 update requests and provides various examples for their usage. Update operations can consist of several sequential requests and are performed on a collection of graphs in a Graph Store. Operations are provided to update, create and remove RDF graphs in a Graph Store.

For instance, the following request inserts a new friend of Alice named Dorothy into the default graph of our example SPARQL service and thereafter deletes all names of Alice's friends with an English language tag.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/> .

INSERT DATA { <http://www.example.org/alice#me> foaf:knows [ foaf:name "Dorothy" ]. } ;
DELETE { ?person foaf:name ?mbox }
WHERE { <http://www.example.org/alice#me> foaf:knows ?person .
        ?person foaf:name ?name FILTER ( lang(?name) = "EN" ) .}
```

As the second operation shows, insertions and deletions can be dependent on the results of queries to the Graph Store; the respective syntax used in the WHERE part is derived from the SPARQL 1.1 Query Language.

7 SPARQL 1.1 Protocol for RDF

The SPARQL 1.1 Protocol for RDF defines how to transfer SPARQL 1.1 queries and update requests to a SPARQL service via HTTP. It also defines how to map requests to HTTP GET and POST operations and what respective HTTP responses to such requests should look like.

For instance, the query from Section 3 above issued against a SPARQL query service hosted at http://www.example.org/sparql/ could according to this specification be wrapped into an HTTP GET request (where the query string is URI-encoded):

```
GET /sparql/?query=PREFIX%20foaf%3A%20%3Chttp%3A%2F%2Fxmlns.com%2Ffoaf%2F0.1%2F%3E%0ASELECT%20%3Fname%20%28COUNT%28%3Ffriend%29%20AS%20%3Fco
Host: www.example.org
User-agent: my-sparql-client/0.1
```

Details about response encoding and different operations for query and update requests, as well as supported HTTP methods, are described in the Protocol specification.

8 SPARQL 1.1 Service Description

The SPARQL 1.1 Service Description document describes a method for discovering and an RDF vocabulary for describing SPARQL services made available via the SPARQL 1.1 Protocol for RDF.

According to this specification, a service endpoint, when accessed via an HTTP GET operation without further (query or update request) parameters should return an RDF description of the service provided. For instance, the following HTTP request:

```
GET /sparql/ HTTP/1.1
Host: www.example.org
```

issued against the SPARQL endpoint hosted at http://www.example.org/sparql/ should return an RDF description, using the Service Description vocabulary. Such a description provides, for instance, information about the default dataset of the respective endpoint, or about SPARQL query language features and entailment regimes that are supported.

9 SPARQL 1.1 Graph Store HTTP Protocol

For many applications and services that deal with RDF data, the full SPARQL 1.1 Update language might not be required. To this end, the SPARQL 1.1 Graph Store HTTP Protocol provides means to perform certain operations to manage collections of graphs directly via HTTP operations.

For instance, the first part of the update request in Section 4 above is a simple insertion of triples into an RDF graph. On a service supporting this protocol, such insertion can - instead of via a SPARQL 1.1 update request - directly be performed via an HTTP POST operation taking the RDF triples to be inserted as payload:

```
POST /rdf-graphs/service?graph=http%3A%2F%2Fwww.example.org%2Falice HTTP/1.1
Host: example.org
Content-Type: text/turtle
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<http://www.example.org/alice#me> foaf:knows [ foaf:name "Dorothy" ] .
```

Other direct HTTP operations for modifying (e.g. to use HTTP PUT to replace an entire graph, or HTTP DELETE to drop an RDF graph) or retrieving (via HTTP GET) RDF graphs are described in the SPARQL 1.1 Graph Store HTTP Protocol specification, which can be viewed as a lightweight alternative to the SPARQL 1.1 protocol in combination with the full SPARQL 1.1 Query and SPARQL 1.1 Update languages.

10 Acknowledgements

The members of the W3C SPARQL Working group who actively contributed to the SPARQL 1.1 specifications are:

- Carlos Buil Aranda, Universidad Politécnica de Madrid
- Olivier Corby, Institut National de Recherche en Informatique et en Automatique (INRIA)
- Souripriya Das, Oracle Corporation
- Lee Feigenbaum, Cambridge Semantics
- Paul Gearon, Revelytix Inc
- Birte Glimm, Universität Ulm
- Steve Harris, Garlik Ltd
- Sandro Hawke, W3C
- Ivan Herman, W3C
- Nicholas Humfrey, BBC
- Nico Michaelis, Dreamlab Technologies AG
- Chimezie Ogbuji, Invited Expert
- Matthew Perry, Oracle Corporation
- Alexandre Passant, DERI, National University of Ireland, Galway
- Axel Polleres, Siemens AG
- Eric Prud'hommeaux, W3C
- Andy Seaborne, The Apache Software Foundation
- Gregory Todd Williams, Rensselaer Polytechnic Institute

11 References

SPARQL-XML-Result
SPARQL Query Results XML Format (Second Edition), D. Beckett, J. Broekstra, Editors, W3C Recommendation, 21 March 2013, http://www.w3.org/TR/2013/REC-rdf-sparql-XMLres-20130321. Latest version available at http://www.w3.org/TR/rdf-sparql-XMLres. (See http://www.w3.org/TR/rdf-sparql-XMLres/.)

RDF-Schema
RDF Vocabulary Description Language 1.0: RDF Schema , ed. Dan Brickley and R.V. Guha, W3C Recommendation 10 February 2004 (See http://www.w3.org/TR/rdf-schema/.)

RDF-MT
RDF Semantics , ed. Pat Hayes, W3C Recommendation 10 February 2004 (See http://www.w3.org/TR/rdf-mt/.)

OWL2-Overview
OWL 2 Web Ontology Language Document Overview, W3C OWL Working Group, W3C Recommendation 27 October 2009 (See http://www.w3.org/TR/owl2-overview/.)

RIF-Overview
RIF Overview, ed. Michael Kifer and Harold Boley, W3C Working Group Note 22 June 2010 (See http://www.w3.org/TR/rif-overview/.)

Turtle
Turtle - Terse RDF Triple Language, ed Eric Prud'hommeaux and Gavin Carothers, Working Draft 09 August 2011. (See http://www.w3.org/TR/turtle/.)

SPARQL10-Query
SPARQL Query Language for RDF, ed. Eric Prud'hommeaux and Andy Seaborne, W3C Recommendation 15 January 2008 (See http://www.w3.org/TR/rdf-sparql-query/.)

Change Log

Changes since Proposed Recommendation

- Removed reference to obsolete document

Changes since Last Call

- Remove use of term "REST"; updated references.