

ChemPedia RDF #1: the SPARQL end point

Egon Willighagen 

Published November 19, 2009

Citation

Willighagen, E. (2009, November 19). ChemPedia RDF #1: the SPARQL end point. *Chem-bla-ics*. <https://doi.org/10.59350/qfhff-gen31>

Keywords

Rdf, Sparql, Chempedia

Copyright

Copyright © Egon Willighagen 2009. Distributed under the terms of the [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

chem-bla-ics

Well, you might spot a pattern here; yes, another chemical [SPARQL end point](#) (actually, it shares the end point with the [Solubility data](#)). This time around [Rich's ChemPedia](#). Taking advantage of the [CCO-licensed downloads](#), I have created a small [Groovy](#) script (using this [JSON library](#)) to convert the ChemPedia [JSON](#) into [Notation3](#):

```
import net.sf.json.groovy.JsonSlurper;

input = new File("substances.json")
json = new JsonSlurper().parse(input);

println "@prefix dc: <http://purl.org/dc/elements/1.1/>";
println "@prefix cp: <http://rdf.openmolecules.net/chempedia/onto#>";
json.each { it ->
    println "<" + it.uri + "> dc:identifier \"" + it.gsid + "\"";
    println " <http://www.w3.org/2002/07/owl#sameAs> <http://
rdf.openmolecules.net/?" + it.inchi + ">";
    println " <http://www.iupac.org/inchi> \"" + it.inchi + "\".";
    if (it.namings.size() > 0) {
        for (int i = 0; i<it.namings.size(); i++) {
            naming = it.namings.get(i);
            namingURI = it.uri + "/naming" + i;
            println "<" + it.uri + "> cp:hasNaming " +
                "<" + namingURI + ">.";
            println "<" + namingURI + "> a cp:Naming;";
            println " cp:hasName \"" + naming.name + "\"";
            println " cp:hasStatus \"" + naming.status + "\"";
            println " cp:hasScore \"" + naming.score + "\".";
        }
    }
}
```

After uploading it into [Virtuoso](#) (now using `DB.DBA.TTLP` instead of `DB.DBA.RDF_LOAD_RDFXML_MT`), we can now have our regular SPARQL fun with the data from ChemPedia. For example, list the 10 names with the most votes:

```
prefix dc: <http://purl.org/dc/elements/1.1/>
prefix cp: <http://rdf.openmolecules.net/chempedia/onto#>

select distinct ?name ?score where {
    ?s a cp:Naming ;
        cp:hasName ?name ;
        cp:hasScore ?score .
} ORDER BY DESC(?score) LIMIT 10
```