Matt Harbers
Databases and the Web
April 22nd, 2011

DBPedia (dbpedia.org)

What is it?

DBpedia is a community whose goal is to provide a web based open source data set of RDF triples based on Wikipedia data. The intention of this project is to use data dumps provided by Wikipedia itself to categorize as much knowledge as possible. The purpose of this knowledge is to allow users to ask sophisticated questions against Wikipedia data and to link data sets together across the web. The community has already described more than 3.5 million things which are broken down into categories such as people, places, music albums, films, video games, organizations, species and diseases.

The information provided in the DBpedia dataset is represented in a cross-domain ontology that was manually created by members of the community. The classes, (272 in all) were based on the most commonly used infoboxes in Wikipedia. The infobox in Wikipedia is the grey box situated on the right side of a Wikipedia page that contains summary data of the page you're viewing. The 272 classes are organized in a hierarchy with "owl:Thing" representing the top most level. Examples of the class structure are: "Means of Transportation" is the parent of aircraft, ship, automobile, etc. and "Event" is the parent of music festival, military conflict, convention, etc.

How is the data structured?

The DBPedia ontology is based on OWL and describes all the classes in the dataset. The data is mapped to the ontology by the DBPedia project. The group faces a somewhat complex task because the Wikipedia infobox template system as evolved in a decentralized way by many different groups for whatever purpose they needed at the time. Because of this non-standardized method of describing things, different templates may use different names for the same attribute. "birthplace and placeofbirth" are such examples. The DBPedia project focuses on mapping these equivalent property names to a more centralized system. The following information shows an example of how a piece of data is stored in a RDF triple along with the corresponding class definition in the ontology:

Ontology:

```
<owl:Class rdf:about="http://dbpedia.org/ontology/Person">
        <rdfs:label xml:lang="en">person</rdfs:label>
        <rdfs:label xml:lang="de">Person</rdfs:label>
        <rdfs:label xml:lang="pt">pessoa</rdfs:label>
        <rdfs:label xml:lang="fr">personne</rdfs:label>
        <rdfs:subClassOf  rdf:resource = "http://www.w3.org/2002/07/owl#Thing"></rdfs:subClassOf>
        <owl:equivalentClass rdf:resource="http://xmlns.com/foaf/0.1/Person"></owl:equivalentClass>
  </owl:Class>
```

As you can see, this class definition also adds support for multiple languages in addition to defining a subclass of "Thing". This means that "Person" is one level of inheritance below the root. It also provides a mapping to FOAF (Friend of a Friend) definition of "Person". This provides a machine-readable ontology for a "person", including their activities and relations to other people and objects.

Instance:

```
<http://dbpedia.org/resource/Aristotle> <http://dbpedia.org/ontology/deathPlace> <http://dbpedia.org/resource/Chalcis> .
<http://dbpedia.org/resource/Aristotle> <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Stageira> .
<http://dbpedia.org/resource/Aristotle> <http://purl.org/dc/elements/1.1/description> "Greek philosopher"@en .
<http://dbpedia.org/resource/Aristotle> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1/Person> .
<http://dbpedia.org/resource/Aristotle> <http://xmlns.com/foaf/0.1/name> "Aristotle"@en .
```

The above screenshot represents instance data that corresponds to the ontology for "Person". In this case, we have several pieces of information about Aristotle. It follows the RDF standard of subject, predicate and object, where we use the property "deathPlace", "birthplace", etc to describe Aristotle. Each of these properties corresponds to an object which may either be a literal value such as, "Greek philosopher", or reference another resource, or "subject". If you reference the resource, "http://dbpedia.org/resource/Chalcis", you'll find information about the Greek town "Chalcis".

In addition to a description, you'll also find properties that reference other DPpedia objects with their own corresponding resource pages. In this way, a net of information is built with the edges of the net creating relationships between objects. A web based flash application called "RelFinder" provides a

visual representation of these relationships and shows how “things” are connected and by which relationships.

Where does the data come from?

The data used in DBpedia comes from Wikipedia as was mentioned previously. Wikipedia has become one of the central sources of human knowledge recently and remains so because the community keeping it up to date is so vast. While much of the data in Wikipedia is largely unstructured, the infobox templates, categorization information, images, geo information, and external url links are all structured and can be represented in RDF. Below, you’ll see an example of a Wikipedia infobox containing structured data about a city. With information structured in this way, interesting and meaningful queries can be created. Such queries might include searching for towns in Austria with a specific population. With the latitude and longitude numbers, you can do radius searches for events nearby. Because the data is structured in this way, you can be sure that your search results are both meaningful and correct.

The dataset is gathered in an automated process using DBpedia’s extractor that pulls all properties from all infoboxes and templates for all Wikipedia articles. This also includes languages besides English. While about half of the data in DBpedia is in English, there is a great deal of information in other languages, as well. Very little clean-up is done to the actual data and there is no consistent ontology for the dataset. Because of this, there are about 8000 different property types. An example of the limited data clean-up performed is the conversion of dates from a long format (June 2009), to an XML schema format (2009-06). The Infobox Ontology is generated using hand generated mappings. This means that there is less data (259 classes), but the mappings fix weaknesses in the Wikipedia infobox system; such as the problem of using different infoboxes for the exact same classes.

```
{{Infobox Town AT |
  name = Innsbruck |
  image_coa =  InnsbruckWappen.png |
  image_map = Karte-tirol-I.png |
  state = [[Tyrol]] |
  regbzk = [[Statutory city]] |
  population = 117,342 |
  population_as_of = 2006 |
  pop_dens = 1,119 |
  area = 104.91 |
  elevation = 574 |
  lat_deg = 47 |
  lat_min = 16 |
  lat_hem = N |
  lon_deg = 11 |
  lon_min = 23 |
  lon_hem = E |
  postal_code = 6010-6080 |
  area_code = 0512 |
  licence = I |
  mayor = Hilde Zach |
  website = [http://innsbruck.at] |
}}
```

How to Access the Data:

At the most basic level, you can browse through one of the datasets looking at triplets independent of one another. There are over 20 downloadable datasets each containing hundreds of megabytes of text, so this approach is very impractical. Additionally, browsing a dataset prevents you from taking advantage of the net of information as you're unable to connect different resources together to produce meaningful data.

To that end, multiple groups have produced interfaces with which users may interact with the data in a more meaningful manner. Each of these interfaces uses the same underlying SPARQL queries to retrieve data. By importing various namespaces, you can specify what kind of data is available for your query. The queries return data in xml, but the SPARQL Explorers convert the data into html for easier viewing. The SPARQL queries behave much like standard SQL queries, except you define the format of your desired result with subject-predicate-objects.

A difference between using SPARQL with DBpedia rather than SQL is that you can specify the "type" of "thing" that is returned by specifying a property that is only present in that type of thing. This is best described with the example of querying for educational institutions in Atlanta. The basic query of doing a radius search around "Atlanta" returns hundreds of results of "things" that are located in

Atlanta, such as: American Cancer Society, Anna Seward Pruitt, Court of Appeals, etc. The results span many different categories. By simply including a property called "established", we filter our results to entries that have that property defined; in this case, educational institutions.

This method of querying is very powerful, but it also requires that the user running the queries be very familiar with the dataset and fully aware of available properties. Mentioned earlier, there are over 8000 properties defined, so this isn't really a practical approach to querying.

Other groups have implemented "search engines" that rely on different methods of searching to provide a more user friendly interface. One such engine is the OpenLink Virtuoso browser that allows you to run simple text searches on DBpedia data. This interface is better, but another group "neofonie" has created a faceted search that allows users to perform both text searches as well as filter by category and individual property. At each stage, you can provide data ranges for each filterable property. This allows users to easily run a query that produces interesting results. An example query of, "cities at mile high elevation with specific population over 1 million people" is listed below. This query is a great example of what a semantic search engine gains over standard text based search engines.

Examples:

http://www.visualdataweb.org/relfinder/relfinder.php

- Porsche, Volkswagen, Allan McNish, Audi
- Physics, Albert Einstein, Literature (then + Barack Obama)
- George Clooney, O Brother, Where Art Though + John Turturro (start clicking on classes)

http://dbpedia.neofonie.de/browse

- http://dbpedia.neofonie.de/browse/rdf-type:City/elevation~:1600~100000/populationTotal~:1000000~10000000/?fc=12

http://dbpedia.org/snorql/

All "Things" about Atlanta:

```
SELECT * WHERE {
   <http://dbpedia.org/resource/Atlanta> ?p ?o .
   FILTER (LANG(?o)='en') .
}
```

People who were born in Germany before the year 1800, but died in Paris:

```
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT * WHERE {
   ?person dbo:birthDate ?birth .
   ?person dbo:deathPlace :Paris.
```

```
    ?person dbo:birthPlace :Germany .
    ?person foaf:name ?name .
    ?person rdfs:comment ?description .
    FILTER (LANG(?description) = 'en') .
    FILTER (?birth < "1800-01-01"^^xsd:date) .

}
ORDER BY ?name
```

Schools within 10km of Atlanta:

```
SELECT DISTINCT ?Link ?SchoolName ?EstablishedDate ?lat ?long WHERE
 {
  <http://dbpedia.org/resource/Atlanta> geo:geometry ?sourcegeo .
  ?resource geo:geometry ?matchgeo .
  ?resource geo:lat ?lat .
  ?resource geo:long ?long .
  FILTER ( bif:st_intersects ( ?matchgeo, ?sourcegeo, 5 ) ) .
  ?Link ?somelink ?resource .
  ?Link <http://dbpedia.org/property/established> ?EstablishedDate .
  ?Link rdfs:label ?SchoolName .
  FILTER ( lang ( ?SchoolName ) = "en" )
 }
order by ?SchoolName
```