Teoría de Grafos para Big Data

# SPARQL y Web Semántica

En caso de dudas:

`http://www.w3.org/TR/sparql11-query/#WritingSimpleQueries`

## 1. DBPedia

Lo primero que vamos a hacer es ingresar al SPARQL endpoint de DBpedia. Este SPARQL Endpoint tiene información de Wikipedia, y nos va a permitir hacer consultas SPARQL.

`http://dbpedia.org/sparql`

El objetivo de esta actividad es dominar la sintaxis de SPARQL, descubriendo cuál es el chileno más viejo en Wikipedia. ¿Cómo se podría buscar esta información de forma manual? Con SPARQL podemos computarlo en menos de 1 segundo.

1. Lo primero es localizar el URI correspondiente a "Chile" en esta base de datos. Para esto, prueba escribiendo:

   ```
   PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
   SELECT DISTINCT ?s WHERE { ?s rdfs:label "Chile"@en }
   ```

   Estamos buscando todos los triples que se ven de la forma `(?s, rdfs:label, "Chile")`. El `@en` es solo para señalar el idioma; puedes aprender sobre esto en el tutorial de más arriba.

   De todas las URIs que aparecen, solo una representa al recurso "Chile". En caso de duda, es posible hacer click sobre el URI de cada una de las propiedades para acceder a su descripción.

2. Una vez localizado el URI de Chile, lo siguiente es ver como se escribe la propiedad "lugar de nacimiento", o "birth place" en esta base de datos. Hay más de una propiedad que significa lugar de nacimiento; las buscaremos todas todas.

   Podemos buscar todas las propiedades `?p` que están en un triple de la forma `(?s ?p <Chile>)`, reemplazando a Chile por el URI que descubrimos antes.

   Ejecutamos la consulta

```
SELECT DISTINCT ?p WHERE {
  ?s ?p <http://dbpedia.org/resource/Chile>
}
```

Y buscamos cuáles son las propiedades que podrían corresponder al lugar de nacimiento de una persona.

3. Con los URIs de birth place ahora podemos hacer una consulta que tenga a todos los recursos que nacieron en Chile. Reemplaza las URIs que ya hemos encontrado en la siguiente consulta:

```
SELECT * WHERE {
  ?s ?p <URI Chile> .
  FILTER(
    ?p = <uri-birthplace-1> ||
    ?p = <uri-birthplace-2> ||
    ...
    ?p = <uri-birthplace-n>
  )
}
```

4. Encuentra ahora la propiedad "fecha de nacimiento" usando la técnica anterior sobre algún sujeto chileno.

```
SELECT DISTINCT ?p WHERE {
  <http://dbpedia.org/resource/Iván_Zamorano> ?p ?o .
}
```

(Cuidado con la tilde en Iván)

5. Finalmente, encuentra la propiedad "fecha de nacimiento" y ordena a estos sujetos por ella. Para ordenar, puedes usar una consulta similar a esta:

```
SELECT DISTINCT ?s ?fecha WHERE {
  ?s ?p <URI Chile> .
  ?s <uri-fecha-de-nacimiento> ?fecha
  FILTER(
    ?p = <uri-birthplace-1> ||
    ?p = <uri-birthplace-2> ||
    ...
    ?p = <uri-birthplace-n>
  )
} ORDER BY (?fecha)
```

¿Quienes son los más antiguos? ¿Hay errores evidentes en los datos?

Es importante revisar los datos aunque sea superficialmente para detectar posibles inconsistencias. Por ejemplo, alguno de los años de nacimiento podría estar escrito con sólo dos dígitos, lo que haría que alguien nacido el año 98 fuera más joven que alguien nacido en 1910.

# 2. Wikidata

Para trabajar en Wikidata tenemos dos opciones. La primera (mucho más simple) es localizar el endpoint de Wikidata:

`https://query.wikidata.org/`

La manera enredada (pero útil si queremos hacer muchas consultas) es instalar un DBMS de SPARQL/RDF y los datos directamente.
Instrucciones generales para bajar virtuoso (windows/linux y MacOS):

`http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSDownload`
`http://carsten.io/virtuoso-os-on-mac-os/`

Y Wikidata en RDF:

`https://www.wikidata.org/wiki/Wikidata:Database_download#RDF_dumps`

## 2.1. ¿Cómo representa Wikidata información en RDF?

1. Wikidata representa información de entidades, propiedades, y las relaciones entre entidades. Tomemos por ejemplo la entidad Q298 y la propiedad P1082 (*population*). Abre en un navegador la URI:

   `https://www.wikidata.org/wiki/Q298`

   y busca todas las entidades que están conectadas a la entidad Q298 mediante la relación *population*.

2. Usa el endpoint de Wikidata para investigar acerca de cómo se representa esta información. Compara las siguientes dos consultas:

```
SELECT * WHERE {
  wd:Q298 wdt:P1082 ?p
}

SELECT * WHERE {
  wd:Q298 p:P1082 ?p
}
```

¿Qué diferencia hay en las respuestas de ambas consultas? ¿Cuál es la diferencia entre `wdt:P1082` y `p:P1082`? ¿Cuándo es útil cada una?

La última consulta devuelve una lista de entidades. Podemos consultar que devuelva todo lo que está conectado a una de esas entidades:

```
SELECT * WHERE {
  <uri-entidad> ?p ?o
}
```

¿Qué propiedades son las que nos interesan de esta entidad?

Bajo "population" en la página de resumen de Chile aparecen distintos niveles de población para el país, definidos como *point in time*.

Nótese la diferencia entre los prefijos `p`[1] y `wdt`[2]. Si ejecutamos

```
SELECT * WHERE {
  wd:Q298 wdt:P1082 ?p
}
```

obtenemos en `?p` la población más reciente de Chile. Sin embargo, para obtener más detalle nos interesa consultar con p:P1082. Ejecutando la siguiente consulta:

```
SELECT * WHERE {
  wd:Q298 p:P1082 ?p
}
```

Vemos una serie de objetos que empiezan con `wd:statement`. Podemos consultar uno de estos arbitrariamente para estudiarlo:

```
SELECT * WHERE {
  wds:Q298-07596AD0-5B2F-49BB-8B14-2694C42B109D ?p ?o
}
```

Con esto, encontramos que se relaciona con una fecha a través de `pq:P585` y con una cifra a través de `ps:P1082`.

3. Explique cómo se ha mapeado la base de datos de documentos de Wikidata al modelo RDF

[1] `http://www.wikidata.org/prop/`
[2] `http://www.wikidata.org/prop/direct/`

## 2.2. Salida al mar y PIB per cápita

1. Tomemos la lista de los países:

```
SELECT ?country WHERE {
  ?s wdt:P31 wd:Q6256 .
  ?s rdfs:label ?country .
  FILTER (lang(?country) = 'es')
}
```

2. Agreguemos el PIB per cápita:

```
SELECT ?country ?gdp WHERE {
  ?s wdt:P31 wd:Q6256 .
  ?s rdfs:label ?country .
  ?s wdt:P2132 ?gdp
  FILTER (lang(?country) = 'es')
} ORDER BY desc(?gdp)
```

3. Agreguemos si tienen o no salida al mar, y ordenemos por PIB per cápita:

```
SELECT ?country ?gdp (bound(?landlocked) as ?sin_mar) WHERE {
  ?s wdt:P31 wd:Q6256 .
  ?s rdfs:label ?country .
  ?s wdt:P2132 ?gdp .
  OPTIONAL {
    ?s wdt:P31 ?landlocked
    FILTER (?landlocked = wd:Q123480)
  }
  FILTER (lang(?country) = 'es')
} ORDER BY DESC(?gdp)
```

   ¿Qué se observa?

4. Agrupemos por salida al mar y calculemos el PIB per cápita promedio

```
SELECT (AVG(?gdp) AS ?pib_promedio) ?sin_mar WHERE {
  ?s wdt:P31 wd:Q6256 .
  ?s rdfs:label ?country .
  ?s wdt:P2132 ?gdp .
  OPTIONAL {
    ?s wdt:P31 ?landlocked
    FILTER (?landlocked = wd:Q123480)
  }
  FILTER (lang(?country) = 'es')
} GROUP BY (bound(?landlocked) as ?sin_mar)
```

## 2.3. Más consultas

Ejecute las siguientes consultas usando el endpoint de Wikipedia. Como ayuda, Wikidata tiene una lista de ejemplos de consultas en SPARQL y la lista completa de prefijos:

```
https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/queries
https://en.wikibooks.org/wiki/SPARQL/Prefixes
```

1. Todos los países en Wikidata, junto a sus etiquetas en español. Hint:

   ```
   lang(?x) = 'es'
   ```

2. El país que comparte fronteras con más países (hint: `GROUP BY` y `COUNT`)

3. El país con más población el año 2000 (hint: `YEAR(?fecha)`)

   Nótese que el país con más población parece ser India. Esto se debe a que no hay datos para la población de China (Q148) el 2000; solamente hay puntos para el 2010, 2012 y 2015.

4. Los 10 países con mayor crecimiento porcentual de población entre 2000 y 2010

5. Los países cuyo presidente (o cabeza de estado) es mujer.

   Nótese que existen dos propiedades: P6 (*head of government*) y P35 (*head of state*). En muchos casos, como en Chile, estos dos son la misma persona. Pero en otros (por ejemplo Canadá), estos son personas distintas. Por ahora veamos solamente cabezas de estado, así que nos quedaremos con P35.

6. Las 20 cumbres más altas, con información de su cordillera (si existe). Ojo: las alturas pueden estar expresadas en unidades distintas. En general, un objeto de tipo `heightStatement` se relaciona mediante `psv:P2044` con una cantidad, y mediante `psn:P2044` con una cantidad normalizada.

7. Todos los escritores o escritoras chilenas vivos. Hint: `!bound()`

8. Las escritoras chilenas. Si han ganado un premio, inclúyalo.

9. Los chilenos que trabajan en ciencia de la computación