UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería de la Salud

Proyecto Estándares de datos abiertos

Open data standards project

Realizado por
Jesús Aldana Martín, Hugo Ávalos de Rorthais, Pablo Molina Sánchez y Juan Carlos Ruiz Ruiz

Tutorizado por
Laura Panizo Jaime

Departamento
Lenguajes y Ciencias de la Computación

MÁLAGA, diciembre de 2022

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADUADO EN INGENIERÍA DE LA SALUD

Proyecto de estándares de datos abiertos

Open Data Standards project

Realizado por
Jesús Aldana Martín, Hugo Ávalos de Rorthais,
Pablo Molina Sánchez, Juan Carlos Ruiz Ruiz

Tutorizado por
Laura Panizo Jaime

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, DICIEMBRE DE 2022

Fecha defensa: diciembre de 2022

Índice

1 Introducción

Este es un proyecto para la asignatura de *Estándares de datos abiertos e integración de datos abiertos*. En esta memoria mostramos todo lo aprendido durante el curso para ellos hemos generado a partir de unos datos open-source algunas transformaciones XSLT y un modelo de web semántica.

1.1. Motivación

La idea del proyecto es utilizar un conjunto de datos en CSV para generar un XML utilizable y a partir de dicho XML obtendremos un conjunto de datos con el que podremos generar la ontología. Usaremos datos sobre los accidentes de bicicletas en Reino Unido, asociados al perfil del ciclista y al estado del tiempo en ese momento. Los datos los obtendremos del siguiente CSV:

Bicycle accidents in Great Britain 1979 to 2018

A partir de estos datos, obtendremos un único XML con el que podremos utilizar los datos y generar la base de datos.

Por otro lado buscamos generar una página web donde podamos mostrar todos los datos obtenidos y mostrar los grafos de la red correspondientes. Esta web será meramente para mostrar los resulatados obtenidos y facilitar la visualización de los datos. En la figura 2 vemos la que podría ser un posible diseño conceptual de la página web, este diseño no es el que tomará finalmente la misma, sin embargo nos permite hacernos una idea de como funcionará la misma. Contendrá un conjunto de botones que nos redirigirán a los diferentes resultados que hemos obtenido a lo largo del trabajo.

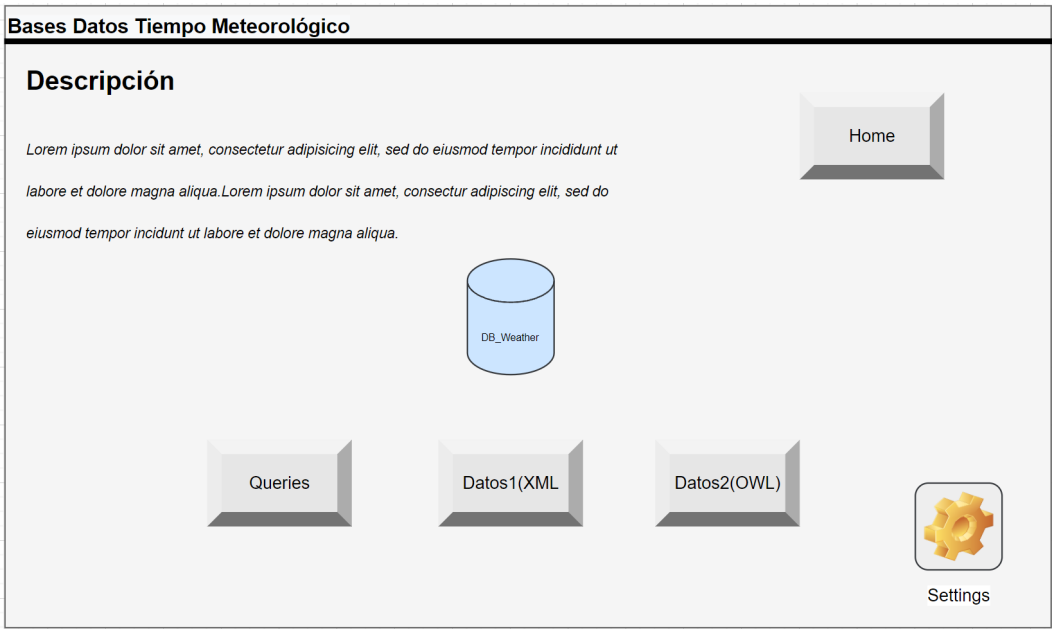

Figura 1: Diseño incial de la página web

1.2. Objetivos

El objetivo principal del proyecto es aprender a utilizar las tecnologías de manejo de bases de datos XML, transformaciones XSLT y utilizar tecnologías de generación de ontologías en OWL y RDF.

Como objetivo secundario tenemos el generar una ontología que sea ampliable y explique de forma detallada la distribución de accidentes de bicicleta en diferentes condiciones y horarios, y obtener información que relacione los datos de forma automática.

1.3. Estructura del documento

El documento constará de dos partes diferenciadas, una primera parte en la que se tratarán los conceptos de XML, XSLT y Xquery; y una segunda parte en la que usaremos OWL, RDF y SPARQL.

En la primera parte explicaremos como hacemos la transformación de nuestros datos de CSV a XML, también realizaremos varias transformaciones XSLT de XML a HTML que mostraremos en nuestra página web, por último utilizaremos Xqueries para obtener datos concretos de nuestro XML y mostrarlos, de nuevo, en nuestra página web.

En la segunda parte, nos centraremos en tecnologías usadas para el manejo de ontologías como serán RDF, en primer lugar, para generar una primera ontología. Además generaremos la ontología también en OWL y mostraremos su grafo en nuestra página web. Por último, usaremos queries de SPARQL para manejar nuestros datos y mostrarlos en la web.

1.4. Tecnologías usadas

Para el desarrollo de la primera parte utilizaremos el Visual Studio Code, para realizar las transformaciones a XML, y el ExistDB, para las consultas Xquery. Para el desarrollo de la segunda parte utilizaremos Protegé, para la creación de la ontología, y el GraphDB para las consultas en SPARQL.

Todo código utilizado en el proyecto se podrá obtener del siguiente repositorio:
https://github.com/JCruiz15/Bycicles-accidents-project

2
Datos

En esta sección vamos a poner un poco en contexto los datos con los que hemos trabajado. Los datos utilizados han sido descargados de la pagina de kagle. Todos los créditos reservados para el autor: John Harshith, hemos hecho uso de los datos mediante la licencia de creative commons CC BY-NC-SA 4.0.

2.1. Acerca de los datos

Como ya hemos comentado un poco en la introducción, este conjunto de datos recoge los accidentes de bicicleta ocurridos en Reino Unido durante 1979 y 2018. La inspiración por la cual se recogieron este tipo de datos surgió cuando se reconoció el ciclismo como deporte olímpico y el uso de la bicicleta se popularizo por toda Europa. Esto trajo consigo multitud de accidentes en los que los ciclistas se veían involucrados debido a las malas condiciones de las carreteras de la época por lo que se comenzaros a recabar datos de lo sucedido con el propósito de encontrar patrones de consecuencia en los involucrados.

2.2. Contenido de los datos

Los datos recabados se han dividido en dos CSV diferentes. En uno de ellos se muestra la información relacionada con el accidente (Accidents.csv) y en el otro con las personas involucradas (Bikers.csv). En los 39 años en los que se han ido recogiendo datos se han obtenido **827871** índices de accidentes únicos. Ambos CSV ocupan en total 101.65 MB, al tratarse de tantos datos para este proyecto hemos reducido los datos al año de 1995, sin embargo nuestros resultados son ampliables al dataset entero.

2.2.1. Accidents.csv

En este CSV encontramos diez columnas de datos diferentes:

- Índice del accidente : Identificador único del accidente. (String)
- Numero de vehículos : Numero de vehículos involucrados. (Integer)
- Numero de involucrados: Numero de heridos. (Integer)
- Fecha: Fecha en la que ocurrió el accidente. (DateTime)
- Velocidad limite: Velocidad máxima de la carretera. (Integer)
- Condiciones de la carretera: En que condiciones se encontraba la carretera en el accidente. Ej: Nevada, Húmeda... (String)
- Climatología: Como estaba el clima durante el accidente. Ej: Lluvia, Nieve... (String)
- Día: Día del accidente. (String)
- Tipo de carretera: Tipo de carretera. Ej: Sentido único, doble... (String)
- Visibilidad: Las condiciones de luz que había en el momento del accidente. Ej: Soleado, Niebla, Noche... (String)

Los datos que hemos utilizado los hemos procesado un poco utilizando python. Como se puede ver en el repositorio de Github en los archivos py, se han fusionado las columnas de tiempo y fecha en DateTime.

2.2.2. Bikers.csv

En el csv de los ciclistas encontramos cuatro columnas:

- Índice del accidente : Identificador único del accidente. (String)
- Genero: Genero de la persona que conducía la bicicleta. (String)
- Gravedad: Gravedad del accidente. Ej: Severo o leve (String)
- Edad: intervalo de edad de las víctimas. (String)

3
XML

3.1. Creación del XML

Lo primero que hemos tenido que hacer ha sido generar un XML a partir de nuestros datos en CSV originales. Nuestro CSV contenía información acerca de todos los accidentes de bicicletas producidos en Reino Unido desde 1979 hasta 2018. Este dataset al ser excesivamente grande para nuestro cometido en el proyecto hemos decidido utilizar 107 elementos del CSV, comprendiendo aproximadamente los accidentes producidos durante 1995.

En primer lugar para generar el XML hemos creado un XML schema que nos ayudase a verificar la correcta implementación de los datos. Este schema nos ha permitido decidir como vamos a disponer los datos en el XML final. Este schema podemos verlo en nuestro repositorio de github

Una vez creado el schema, utilizamos la web convert CSV to XML para obtener el XML resultante ya que el volumen de datos que habría que traducir a XML sería excesivamente grande. Usando la siguiente plantilla obtenida a partir del schema conseguiremos el XML:

```
<?xml version="1.0"?>{br}<accidents>{br}

<Accident {h1}="{f1}">
  <N_vehicules>{f2}</N_vehicules>
  <N_casualties>{f3}</N_casualties>
  <{h4}>{f4}</{h4}>
  <{h5}>{f5}</{h5}>
  <Week_day>{f9}</Week_day>
  <Conditions>
    <{h6}>{f6}</{h6}>
    <{h7}>{f7}</{h7}>
    <{h10}>{f10}</{h10}>
    <{h8}>{f8}</{h8}>
```

```
    <{h11}>{f11}</{h11}>
  </Conditions>
  <Biker>
    <{h12}>{f12}</{h12}>
    <{h14}>{f14}</{h14}>
    <{h13}>{f13}</{h13}>
  </Biker>
</Accident>
</accidents>
```

Listing 1: Creacion del XML

Así con la plantilla obtenemos el XML final que utilizaremos para realizar las Xqueries y las transformaciones. La forma del XML obtenido sería el siguiente.

```
<accidents>
  <Accident Accident_Index="19954100B0030">
    <N_vehicules>2</N_vehicules>
    <N_casualties>1</N_casualties>
    <Date>1995-01-25</Date>
    <Time>22:45</Time>
    <Week_day>Wednesday</Week_day>
    <Conditions>
      <Speed_limit>30.0</Speed_limit>
      <Road_conditions>Wet</Road_conditions>
      <Road_type>Single carriageway</Road_type>
      <Weather_conditions>Clear and windy</Weather_conditions>
      <Light_conditions>Darkness lights lit</Light_conditions>
    </Conditions>
    <Biker>
      <biker_gender>Male</biker_gender>
      <biker_age_group>26 to 35</biker_age_group>
      <biker_severity_injury>Slight</biker_severity_injury>
    </Biker>
  </Accident>
  ...
</accidents>
```

Listing 2: Estructura del XML (primera instancia)

3.2. Consultas XQuery

Lo siguiente que haremos será implementar tres consultas XQuery. Una consulta XQuery nos permite obtener información concreta a partir de nuestro XML. Hemos implementado una consulta para conocer cuantas mujeres han sufrido accidentes en bicicleta, otra que nos muestra información sobre cuantos accidentes se han producido en cada día de la semana y por último hemos creado una tercera que nos muestra las condiciones en las que puede encontrarse la carretera.

3.2.1. Número de accidentes de mujeres

En la primera Xquery que hemos hecho, hemos calculado el número de mujeres que han tenido un accidente en bicicleta. Este ha sido el código utilizado:

```
xquery version "3.1";
count(doc('/db/apps/doc/xml_bycicle_accidents.xml')/accidents/Accident/
    Biker[biker_gender="Female"])
```

Listing 3: Consulta número de accidentes de mujeres

El resultado obtenido ha sido el siguiente:

```
18
```

Listing 4: Resultado de la consulta 1

Como podemos ver, es un número bastante bajo respecto al total (107), suponiendo el número de accidentes en bicicleta protagonizado por mujeres solo un 16,8 por ciento del total.

3.2.2. Número de accidentes en cada día de la semana

La segunda Xquery que hemos hecho consiste en calcular el número de accidentes que han ocurrido por cada día de la semana. Mostrándolo finalmente como un XML mostrando el día de la semana y el número de accidentes que se han producido. Hemos hecho uso de la

clausula let para asignar a una una variable los valores distintos para hacer uso de ella en la consulta de abajo. El resultado se devuelve en formato html. Esta query la hemos hecho con el siguiente código:

```
xquery version "3.1";

let $days := distinct-values(
    for $x in doc("xml_bycicle_accidents.xml")/accidents/Accident
    return data($x/Week_day)
    )

for $day in $days
return <results>
        <day>{$day}</day>
        <n_accidents>{
            count(
                for $x in doc("xml_bycicle_accidents.xml")/accidents/
   Accident/Week_day
                where $x = $day
                return $x
            )
        }</n_accidents>
    </results>
```

Listing 5: Consulta del número de accidentes en cada día de la semana

Así la salida que hemos obtenido de la query es la siguiente:

```
<results>
    <day>Wednesday</day>
    <n_accidents>19</n_accidents>
</results>

<results>
    <day>Thursday</day>
    <n_accidents>16</n_accidents>
</results>
```

```
        ...
<results>
    <day>Saturday</day>
    <n_accidents>12</n_accidents>
</results>
```

Listing 6: Resultado consulta 2

Así los días con mayor número de accidentes en nuestro dataset son los lunes y los miércoles.

3.2.3. Tipo de accidentes según cada condición de la carretera

En la tercera Xquery que hemos hecho, hemos pedido obtener las diferentes condiciones de la carretera en los accidentes producidos con en este código:

```
xquery version "3.1";
distinct-values(
for $cond in doc('/db/apps/doc/xml_bycicle_accidents.xml')/accidents/
    Accident/Conditions/Road_conditions
return data($cond)
)
```

Listing 7: Consulta del tipo de accidentes según cada condición de la carretera

Hemos obtenido este resultado:

```
"Wet"
"Dry"
"Snow"
"Frost"
```

Listing 8: Resultado consulta 3

Como podemos ver, los accidentes en bicicleta ocurren en cualquier condición de la carretera, por lo que podemos llegar a la conclusión de que suelen estar más influenciados por el comportamiento del ciclista o el conductor.

3.3. Transformaciones XSLT

También hemos creado dos transformaciones XSLT, una que transforma el XML en un nuevo XML con los campos traducidos al español y una segunda transformación que transforma el XML a un HTML que muestra los datos en una tabla y es incrustable en una página web.

3.3.1. De XML a XML

Con esta transformación buscamos traducir los atributos y etiquetas, que originalmente, estaban en inglés. También hemos hecho unos pequeños cambios en la estructura, la fecha ahora está unida en una sola etiqueta cuando antes estaba dividida en tres. A su vez hemos creado la etiqueta 'Detalles' para agrupar 'Fecha', 'Vehículos involucrados' y 'Damnificados' dentro de ella.

Haciendo click en el siguiente enlace podemos ver el archivo XSLT con el código utilizado para la transformación: XML a XML

En el siguiente listado se muestra únicamente el primer accidente, el resto de ellos se pueden ver en repositorio de Github. Resultado de la transformación:

```
<?xml version="1.0" encoding="UTF-8"?>
<Accidentes>
   <Accidente ID="19954100C0101">
      <Detalles>
         <Fecha>Monday - 1995-01-16 - 12:30</Fecha>
         <Vehículos_involucrados>2</Vehículos_involucrados>
         <Damnificados>1</Damnificados>
      </Detalles>
      <Condiciones>
         <Tipo_Via>Roundabout</Tipo_Via>
         <Limite_Velocidad>30.0</Limite_Velocidad>
         <Via_Estado/>
         <Clima>Clear</Clima>
         <Luminosidad>Daylight</Luminosidad>
      </Condiciones>
      <Ciclista>
         <Género>Male</Género>
```

```
            <Edad>26 to 35</Edad>
            <Grado_Lesion>Slight</Grado_Lesion>
        </Ciclista>
    </Accidente>
      ...
<Accidentes>
```

Listing 9: Transformación de XML a XML

3.3.2. De XML a HTML

Con esta segunda transformación xslt conseguimos de salida un archivo HTML que muestre una tabla con los datos XML de inicio. Para ello hemos creado la estructura de la tabla en HTML con las etiquetas <table>, <thead>, <tr>, <th>; escribiendo los nombres de las columnas y aplicando un 'apply-templates' para las filas. En la plantilla creada, seleccionamos la etiqueta XML con su respectiva columna de la tabla. La tabla se encuentra ordenada por la fecha de los accidentes.

Haciendo click en el siguiente enlace podemos ver el archivo XSLT utilizado para la transformación: XML a HTML

A continuación se puede observar la tabla que conseguimos de salida utilizando esta transformación. Para un mejor diseño hemos utilizado un framework de front-end basado en CSS (Bootstrap).

Accidentes de bicicletas en Reino Unido

#	Index	N_vehicules	N_casualties	Date	Time	Week_day	Speed limit	Road Conditions	Road Type	Weather Conditions	Light Conditions	Biker gender	Biker age group	Biker severity injury
1	19954100C0101	2	1	1995-01-16	12:30	Monday	30.0	Dry	Roundabout	Clear	Daylight	Male	26 to 35	Slight
2	19954100C0035	2	2	1995-01-19	08:15	Thursday	30.0	Frost	Single carriageway	Clear	Daylight	Female	56 to 65	Serious
3	19954100C0048	2	2	1995-01-21	11:38	Saturday	30.0	Wet	Roundabout	Rain and windy	Daylight	Male	16 to 20	Slight
4	19954100C0057	2	1	1995-01-22	09:20	Sunday	30.0	Wet	Single carriageway	Unknown	Daylight	Male	16 to 20	Slight
5	19954100B0043	2	1	1995-01-24	06:40	Tuesday	50.0	Dry	Single carriageway	Clear	Daylight	Male	26 to 35	Slight
6	19954100B0030	2	1	1995-01-25	22:45	Wednesday	30.0	Wet	Single carriageway	Clear and windy	Darkness lights lit	Male	26 to 35	Slight
7	19954100B0031	2	1	1995-01-26	17:30	Thursday	60.0	Wet	Single carriageway	Clear and windy	Darkness lights lit	Male	26 to 35	Slight
8	19954100C0067	2	2	1995-01-28	20:25	Saturday	30.0	Wet	Single carriageway	Rain	Darkness lights lit	Male	26 to 35	Slight
9	19954100B0058	2	1	1995-02-03	12:30	Thursday	30.0	Dry	Roundabout	Clear	Daylight	Female	21 to 25	Slight
10	19954100C0281	2	1	1995-02-05	08:30	Tuesday	60.0	Dry	Single carriageway	Clear	Daylight	Male	16 to 20	Slight

Figura 2: Transformación de XML a HTML

4
Ontología

4.1. Grafo RDF

La información en contenida en nuestra web, anteriormente explicada en la sección *Datos*, vamos a representarla mediante un grafo etiquetado y dirigido en formato RDF. Este grafo RDF lo hemos realizado directamente con el creador y editor de ontologías *Protégé*. Nuestro proyecto consta de 4 clases principales: **Accident**, **Biker**,**Severity_Injuries** y por último **Conditions** que tiene dos subclases asociadas como *Roads*, con tres subclases Road_conditions, Road_type y Speed_limit, y *Weather*, con dos subclases que son Light y Weather_conditions. Estas últimas subclases junto a *Severity_injuries*, tienen unas instancias fijas declaradas con las que el resto de clases se relacionarán, funcionando como enumerados.

4.1.1. Object Properties

Las propiedades de objetos son importantes para establecer Relaciónes entre clases. Estas propiedades de objetos de nuestro grafo RDF son las siguientes:

- **accident_related**: Relación entre Biker y Accident. Muestra el ciclista causante del accidente.
- **accident_severity**: Relación funcional entre Accident y Severity_Injuries. Guarda la gravedad del accidente.
- **biker_affected**: Relación funcional entre Accident y Biker. Muestra el ciclista afectado por el accidente.
- **biker_injury**: Relación funcional entre Biker y Severity_Injuries la cual muestra la lesión producida en el ciclista provocada por el accidente.

- **conditions_light** : Relación funcional entre Accident y Conditions. Guarda las condiciones de visibilidad en el lugar y momento del accidente.

- **conditions_roadCondition** : Relación funcional entre Accident y Road_conditions. Guarda las condiciones del terreno en el lugar y momento del accidente.

- **conditions_roadType** : Relación funcional entre Accident y Road_type. Guarda el tipo de carretera en el lugar y momento del accidente.

- **conditions_speedLimit** : Relación funcional entre Accident y Speed_limit. Muestra el límite de velocidad de la carretera donde se ha producido el accidente.

- **conditions_weather** : Relación funcional entre Accident y Weather_conditions. Guarda las condiciones meteorológicas en el lugar y momento del accidente.

4.1.2. Data Properties

También, hemos definido propiedades de datos para añadir mayor cantidad de información a las clases que componen nuestro grafo. Estas son:

- **age_group**: Perteneciente a la clase Biker, tiene los posibles valores de ”6 to 10”, ”11 to 15”, ”16 to 20”, ”21 to 25”, ”26 to 35”, ”36 to 45”, ”46 to 55”, ”56 to 65”, ”66 to 75”.

- **date**: Perteneciente a la clase Accident de tipo cadena.

- **casualties_number** : Perteneciente a la clase Accident de tipo cadena

- **gender** : Perteneciente a la clase Biker de tipo cadena y con los posibles valores de ”Female”, ”Male”, .ºther”.

- **time**: Perteneciente a la clase Roads de tipo cadena

- **vehicles_number**: Perteneciente a la clase Accident de tipo entero positivo

- **week_day**: Perteneciente a la clase Accident de tipo cadena y con los posibles valores ”Friday”, ”Monday”, ”Saturday”, ”Sunday”, ”Thursday”, ”Tuesday”, ”Wednesday”

4.1.3. WebVOWL

Para entender mejor el modelo, vamos a mostrar el grafo cre ado. Para ello, nos hacemos uso de la herramienta WebVOWL, portal para mostrar ontologías. Podemos visualizar el grafo en la figura 3.

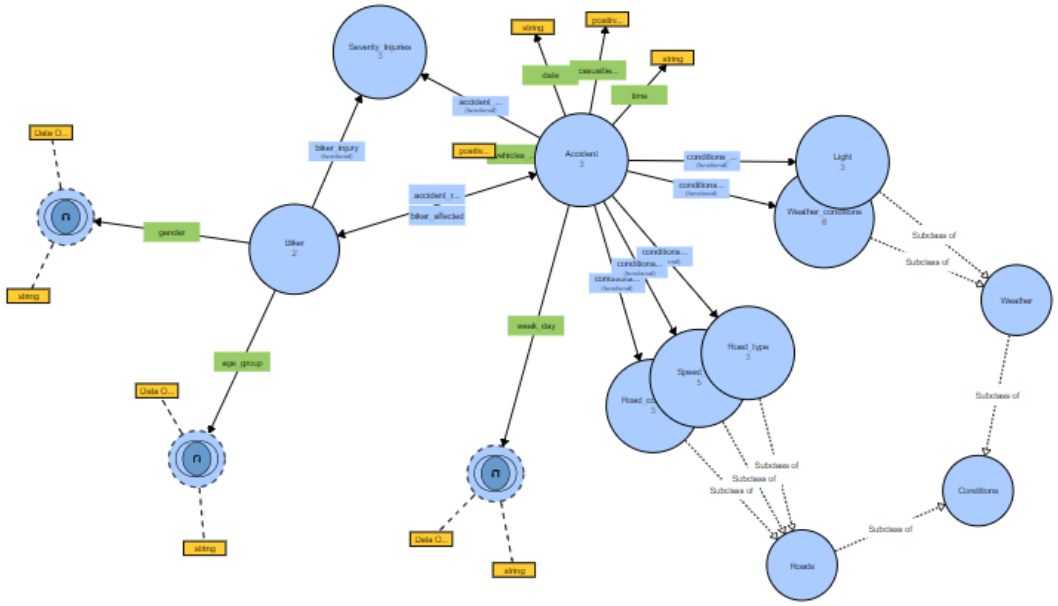

Figura 3: Modelo OWL (Ver el grafo en pantalla completa)

4.2. Consultas SPARQL

Las consultas se han generado utilizando **GraphDB**. A continuación hemos indicado el prefijo con el que llamamos a nuestra ontología:

```
PREFIX bikers_accident: <http://www.semanticweb.org/
    estandares_integreacion_datos/ontologies/2022/10/bikers_accident#>
```

Listing 10: Prefijo de nuestra ontología

4.2.1. Accidentes de bicicleta ocurridos los lunes

La siguiente consulta nos devuelve todos los accidentes que se han dado en un día concreto de la semana. Esta información podría resultar útil para descubrir cual es el día de la semana en el que mas accidentes ocurren.

```
SELECT ?accident ?day
WHERE {
    ?accident rdf:type bikers_accident:Accident .
    ?accident bikers_accident:week_day ?day .
    FILTER(?day = "Monday") .
}
```

Listing 11: Accidentes de bicicleta ocurridos los lunes

4.2.2. Datos de los accidentes ocurridos

Esta segunda consulta muestra toda la información que define un accidente, para ello en la clausula where indicamos de la relación bikers_accident todos esos atributos a mostrar.

```
SELECT ?accident ?light ?rtype ?speed ?weather ?rconditions
WHERE {
    ?accident rdf:type bikers_accident:Accident .
    ?accident bikers_accident:conditions_light ?light .
    ?accident bikers_accident:conditions_weather ?weather .
    ?accident bikers_accident:conditions_roadType ?rtype .
    ?accident bikers_accident:conditions_speedLimit ?speed .
    ?accident bikers_accident:conditions_roadCondition ?rconditions .
}
```

Listing 12: Datos de los accidentes ocurridos

4.2.3. Numero de accidentes ocurridos filtrados por género

Para esta consulta hemos hecho uso de la función **count** para obtener el numero de personas de cada sexo de nuestros datos.

```
SELECT ?gender (COUNT(?gender) as ?amount)
WHERE {
    ?biker rdf:type bikers_accident:Biker .
    ?biker bikers_accident:gender ?gender .
}
GROUP BY ?gender
```

Listing 13: Numero de accidentes ocurridos filtrados por género

4.2.4. Accidentes de mayor gravedad

Esta ultima consulta pretende mostrar aquellos accidentes de mayor gravedad, así como el numero de vehículos involucrados y la climatología del accidente, estos datos son agrupados por la severidad del accidente. Finalmente le aplicamos de nuevo un **count** para conseguir el numero de accidentes con estas características.

```
SELECT ?injury ?weather ?n_vehicules (COUNT(?n_vehicules) as ?
   number_of_accidents)
WHERE {
    ?accident rdf:type bikers_accident:Accident .
    ?accident bikers_accident:accident_severity ?injury .
    ?accident bikers_accident:vehicles_number ?n_vehicules .
    ?accident bikers_accident:conditions_weather ?weather
}
GROUP BY ?injury ?n_vehicules ?weather
```

Listing 14: Accidentes de mayor gravedad

5

Conclusiones y Líneas Futuras

5.1. Conclusiones

Finalmente podemos concluir que en este proyecto se han explorado todos los conceptos estudiados en la asignatura y hemos podido generar una ontología a partir de datos de accidentes de ciclistas que podría ser útil. Esta ontología nos ofrece una visión forma más óptima de visualizar y estudiar este tipo de datos, además que nos permite hacer consultas que podrían darnos gran información para prevenir este tipo de accidentes. Si vemos las consultas SPARQL podemos utilizar la consulta 4.2.3 para realizar estudios estadísticos sobre que personas sufren más accidentes o la consulta 4.2.4 para buscar que condiciones son las más peligrosas. Además gracias a las transformaciones XSLT podremos mostrar todos estos resultados en la página web. Asimismo, podemos decir que los resultados de este proyecto son satisfactorios y podrían ser utilizados para realizar estudios útiles en este campo y para mejorar la protección de los ciclistas.

5.2. Líneas Futuras

Como futuras lineas de investigación pensamos que seria interesante ampliar nuestro proyecto recabando mas datos y actualizándolos. Además de crear muchos mas individuos en nuestra ontología y no solo los que hemos creado nosotros como ejemplo. Tras estas mejoras en nuestro proyecto el objetivo seria recabar datos de accidentes no solo de bicicleta y recoger información de todo tipo de accidentes(ej. aviones) Todos estos datos integrados en la web semántica ayudarían a tener este tipo de información bien organizada ofreciendo búsquedas por significado .

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA