Universidad Técnica Particular de Loja

Sistemas Informáticos y Computación

Integrantes:

- **Estevan Enrique Atiencia Prado**
- **Cristian Daniel Gaona Sánchez**

Asignatura: Sistemas Basados en Conocimiento

Docente: Ing. Janneth Alexandra Chicaiza Espinosa

Introducción

RDF existe como aquella intersección de unas cuantas tecnologías diferentes, lo que nos facilita pensar que es un formato particular de XML, en donde existe herramientas para fedds de blogs.
Ahora que es RDF, se lo conoce como un método general para descomponer conocimiento en piezas pequeñas, con algunas de las reglas acerca de la semántica o significado de esas piezas. Además, es un método tan simple que puede expresar cualquier hecho, y a la vez estructurado que aplicaciones de computadoras lo pueden usar ese conocimiento expresado para hacer cosas útiles. En particular antes que formato, porque uno puede escribir esas piezas de diferentes formas y aun así preservar la información y la estructura, tal y como podemos expresar una oración en diferentes lenguajes humanos o implementar la misma estructura de datos en múltiples lenguajes de programación.
El presente documento se estructura de la siguiente manera de acuerdo fases que se ha trabajado en el proyecto:

- La obtención respetiva de la data: en la primera fase tenemos la investigación de diversas fuentes que contenga data legible para su uso.
- Transformación de datos URL: en esta etapa lo primero que se realizo fue la limpieza de los datos, la selección de las URIs, la transformación de los datos por medio de JENA y la salida que genera el mismo.

Desarrollo

1. **Obtención de Datos**

Fuente	URL	Formato de los Datos	Actualización
Our Word in Data	https://drive.google.com/file/d/1yEC5nkKq_CQRQsyiPqG5gMxrXtaSwJXP/view?usp=sharing	**CSV**	**Diariamente**
Europe: Coronavirus (COVID-19) Subnational Cases	https://drive.google.com/file/d/1prNrikLtKDwNI4Rzai3L7x_3mBrNi5m2/view?usp=sharing	**CSV**	**Diariamente**

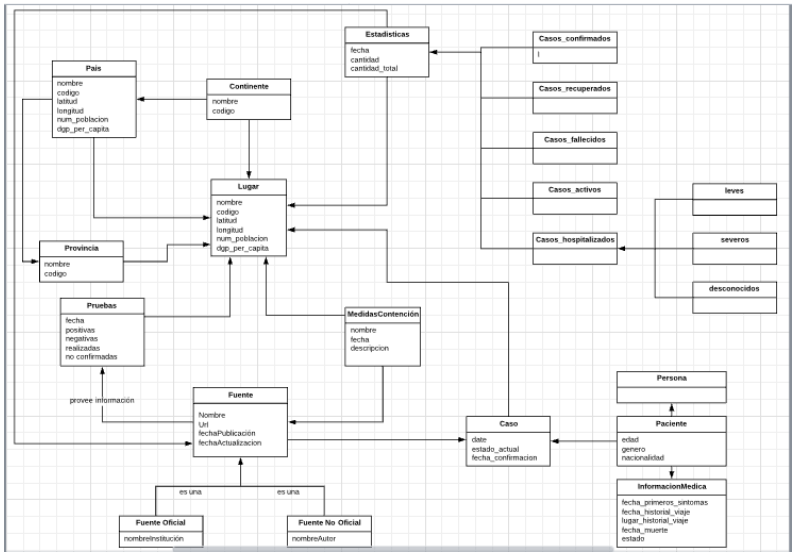
La primera Fuente **Our Word in data** nos brinda información global, por ese motivo se requirió hacer un filtro para escoger solos los datos del continente europeo, el mismo nos brinda información de geográfica del país, ademas de los casos positivos, recuperados, hospitalizados entre algunos otros datos. La segunda fuente Europe: Coronavirus (COVID-19) Subnational Cases a diferencia de la primera solo presenta datos del continente europeo, en la misma se expone datos acumulativos de los fallecidos, recuperados, personas que se encuentran en cuidado intensivos entre otros datos.

2. **Obtención de datos RDF**

2.1 Limpieza de los datos

Según el modelo ontológico defino anteriormente por parte de todo el grupo que conformaba DATA, se comenzó el proceso de limpieza de datos.

Figura 1. Modelo Ontológico



Teniendo en cuenta el modelo ontológico se comenzó con la limpieza de datos, en nuestro caso utilizamos Excel (Hoja de cálculo), en la cual se procedió a la eliminación de columnas o filas obsoletas o que retrasarían el proceso de generación de RDF realiza posteriormente. Además, cabe destacar que usamos la fuente uno y dos para completar la información del mismo, por motivo de que algunos campos se encontraban vacíos. En algunos casos se procedió a buscar otras fuentes de información para llenar las filas que se encontraban vacías.

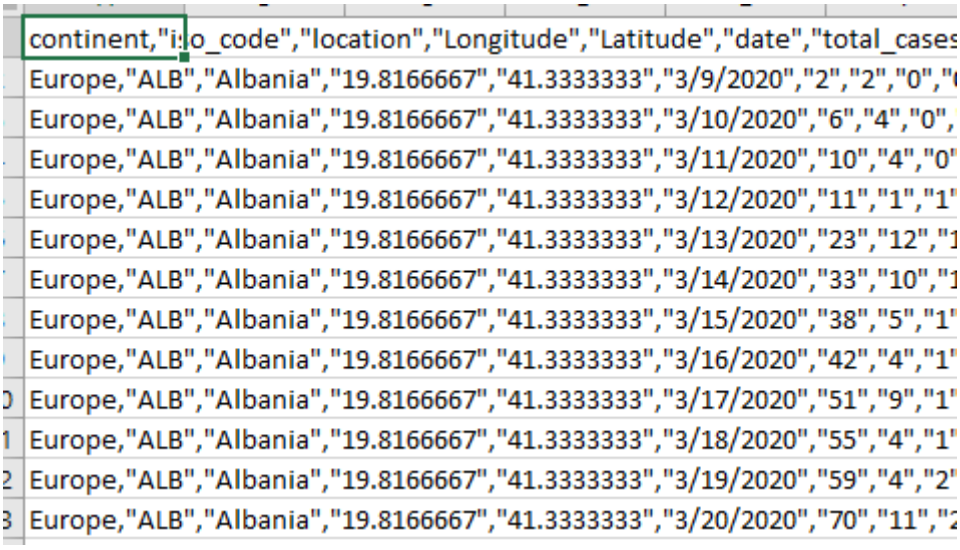
Figura 2. Hoja de Excel

iso_co	locatio	Longitud	Latitude	date	total_cas	new_cas	total_dea	new_deat
ALB	Albania	19.8166667	41.3333333	09/03/2020	2	2	0	0
ALB	Albania	19.8166667	41.3333333	10/03/2020	6	4	0	0
ALB	Albania	19.8166667	41.3333333	11/03/2020	10	4	0	0
ALB	Albania	19.8166667	41.3333333	12/03/2020	11	1	1	1
ALB	Albania	19.8166667	41.3333333	13/03/2020	23	12	1	0
ALB	Albania	19.8166667	41.3333333	14/03/2020	33	10	1	0
ALB	Albania	19.8166667	41.3333333	15/03/2020	38	5	1	0
ALB	Albania	19.8166667	41.3333333	16/03/2020	42	4	1	0
ALB	Albania	19.8166667	41.3333333	17/03/2020	51	9	1	0
ALB	Albania	19.8166667	41.3333333	18/03/2020	55	4	1	0
ALB	Albania	19.8166667	41.3333333	19/03/2020	59	4	2	1
ALB	Albania	19.8166667	41.3333333	20/03/2020	70	11	2	0
ALB	Albania	19.8166667	41.3333333	21/03/2020	70	0	2	0
ALB	Albania	19.8166667	41.3333333	22/03/2020	76	6	2	0

En la figura 2 se expone una parte de la limpieza de los datos que aún se encuentra en un formato xlsx, que correspondería a nuestro dataset, que una vez terminado todo el proceso de limpieza y verificación de los datos se lo transforma a un formato csv, el cual puede ser proceso por un entorno

de desarrollo que en este caso es NetBeans, los datos que se muestran en la siguiente figura 3 es la transformación de xlsx a csv.

Figura 3. Archivo de Excel en formato csv



	continent,"iso_code","location","Longitude","Latitude","date","total_cases
	Europe,"ALB","Albania","19.8166667","41.3333333","3/9/2020","2","2","0",
	Europe,"ALB","Albania","19.8166667","41.3333333","3/10/2020","6","4","0",
	Europe,"ALB","Albania","19.8166667","41.3333333","3/11/2020","10","4","0"
	Europe,"ALB","Albania","19.8166667","41.3333333","3/12/2020","11","1","1"
	Europe,"ALB","Albania","19.8166667","41.3333333","3/13/2020","23","12","1
	Europe,"ALB","Albania","19.8166667","41.3333333","3/14/2020","33","10","1
	Europe,"ALB","Albania","19.8166667","41.3333333","3/15/2020","38","5","1"
	Europe,"ALB","Albania","19.8166667","41.3333333","3/16/2020","42","4","1"
0	Europe,"ALB","Albania","19.8166667","41.3333333","3/17/2020","51","9","1"
1	Europe,"ALB","Albania","19.8166667","41.3333333","3/18/2020","55","4","1"
2	Europe,"ALB","Albania","19.8166667","41.3333333","3/19/2020","59","4","2"
3	Europe,"ALB","Albania","19.8166667","41.3333333","3/20/2020","70","11","2

Una vez terminado este proceso, comenzamos a la selección de patrones de nuestras URI, que utilizaremos en las siguientes etapas.

2.2 Selección de la URI

Para la selección de nuestras URI utilizadas en nuestro proyecto, se optó por utilizar URI que sean significativas. Por ende, se expone las siguientes URI:

- URI Base: nuestra URI debería ser significativa por este motivo se escogió la siguiente URI:

http://utpl.edu.ec/sbc/dataCOVID/

- URI de TBox: se anexa el nombre del concepto o de la propiedad a la estructura base del URI para incluir conceptos y propiedades disponibles en nuestra ontología.

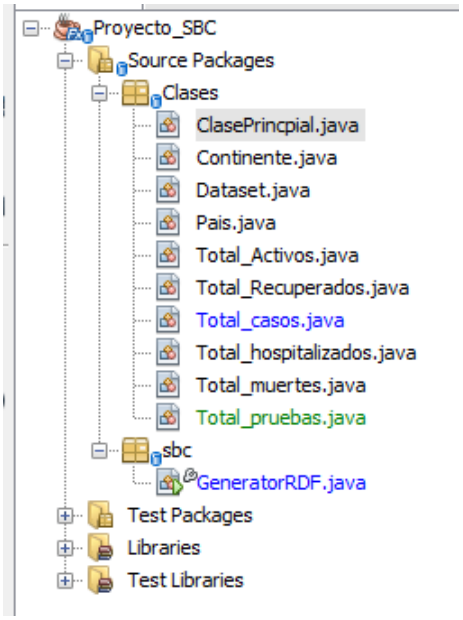
http://utpl.edu.ec/sbc/data/Ontology/{#term_claseorproperty}

2.3 Transformación de los datos con JENA

Una vez que se realizó la limpieza de los datos, la definición de las URI y de la licencia, se comenzó con la transformación de los datos, para este procedimiento de ocupo el entorno de desarrollo NetBeans, el mismo que no permite incluir la librería JENA que se encarga crear los RDF, por medio de la codificación del mismo.

Como primer punto tenemos la lógica que se aplicó, en la que se decidió utilizar dos paquetes, en uno de ellos se guardan las clases general con los atributos necesarios, y en el otro paquete tenemos la clase principal en donde llamamos se realizara todo el procedimiento para la transformación de los datos.

Figura 4. Estructura del proyecto



Antes de iniciar con todo el proceso de transformación tenemos que iniciar dos variables de suma importancia que **DataFilePath** que es el encargado de llamar a nuestro CSV (Dataset) en donde se encuentra todos nuestros datos para ser procesados, la siguiente variable es **GenFilePath** esta vriable se encarga de guarda nuestro RDF en la dirección de le proporcionemos.

Figura 5. Definicion de variables

```
static String DataFilePath = "/Users/Estevan/Documents/NetbeansProjects/Proyecto_SBC/Europa-Date8.csv";
static String GenFilePath = "/Users/Estevan/Documents/NetbeansProjects/Proyecto_SBC/DatosGenerados.rdf";
```

El siguiente paso es leer nuestro CSV como se lo indican en la Figura 6, el mismo que se lo recorre en un while, que por cada iteración los datos serán guardados en un arreglo. Este arreglo pertenece a las clases antes mencionadas que se encuentran en el paquete clase.

Cabe mencionar que se omitieron algunas capturas de cómo se realiza la importación del paquete clases al paquete sbc.

Figura 6. Función para leer el CSV

```
BufferedReader bufferLectura = null;
try {
    bufferLectura = new BufferedReader(new Fi
    String titulo = bufferLectura.readLine();
    String linea = bufferLectura.readLine();
    String[] campos;
    Date fecha= new Date();
    int i=1;
    while (linea != null) {
        campos = linea.split(";");
        linea = bufferLectura.readLine();
        Continente cont= new Continente(campo
        Pais pais = new Pais(campos[1],campos
        System.out.println(String.valueOf(cam
        Dataset ds= new Dataset(campos[22],ca
        Total_Activos TA= new Total_Activos("
        Total_muertes tm= new Total_muertes("
        Total_hospitalizados th= new Total_ho
        Total_Recuperados tr = new Total_Recu
        Total_pruebas tp=new Total_pruebas("P
        System.out.println(campos[5]);
        Total_casos TC= new Total_casos("Conf
        LAC.add(TA);
        LTC.add(TC);
        i++;
    }
```

Para la creación diferentes propiedades que utilizamos en el trascurso de este proyecto, las misma fueron definidas anteriormente en el modelo ontológico. Figura 7.

Figura 7. Definición de los prefijos

```
// create an empty Model
Model model = ModelFactory.createDefaultModel();
File f = new File(GenFilePath); //File to save the results of
FileOutputStream os = new FileOutputStream(f);

//Set prefix for the URI base (data)
String dataPrefix = "http://utpl.edu.ec/sbc/dataCOVID/";
model.setNsPrefix("data", dataPrefix);

String newOnto = "http://utpl.edu.ec/sbc/data/Ontology/";
model.setNsPrefix("newOnto", newOnto);
Model newOntoM = ModelFactory.createDefaultModel();
//Vocab and models present in JENA
//SCHEMA
String schema = "http://schema.org/";
model.setNsPrefix("schema", schema);
Model schemaModel = ModelFactory.createDefaultModel();
//Dbpedia Ontology- DBO
String dbo = "http://dbpedia.org/ontology/";
model.setNsPrefix("dbo", dbo);
Model dboModel = ModelFactory.createDefaultModel();
//Dbpedia Resource - DBR
String dbr = "http://dbpedia.org/resource/";
model.setNsPrefix("dbr", dbr);
Model dbrModel = ModelFactory.createDefaultModel();
//Geonames - gn
String gn = "http://www.geonames.org/ontology#";
model.setNsPrefix("gn", gn);
Model gnModel = ModelFactory.createDefaultModel();
```

Prosiguiendo con el tema de la transformación de los datos, tenemos los métodos utilizados para la creación del RDF, los mismo fueron proporcionados por el docente. Cabe resaltar que los métodos se encuentran dentro de un for, el mismo que recorre todos los datos del Dataset.

Figura 8. Métodos de transformación RDF

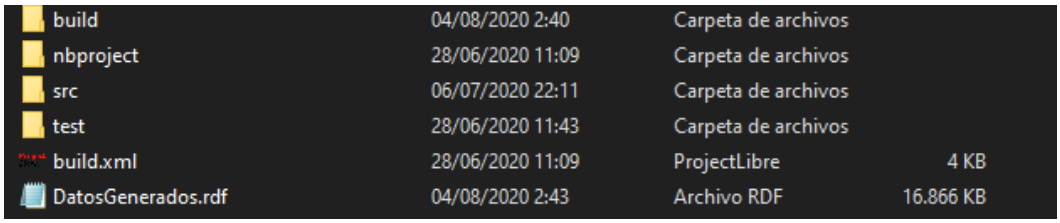
```
for (Total_casos total_casos : LTC) {

   Resource con = model.createResource(dataPrefix + total_casos.getPais().getCon().getNombre())
            .addProperty(RDF.type, dboModel.getProperty(dbo, "Continent"))
            .addProperty(dboModel.getProperty(dbo,"name"),total_casos.getPais().getCon().getNombre() );

   String aux=total_casos.getPais().getNombre().replaceAll(" ", "_");

    Resource rC = model.createResource(dataPrefix + aux)
            .addProperty(RDF.type, dboModel.getProperty(dbo, "Country"))
            .addProperty(dboModel.getProperty(dbo,"name"),total_casos.getPais().getNombre() )
            .addProperty(gnModel.getProperty(gn, "countryCode"), total_casos.getPais().getIso_code())
            .addProperty(schemaModel.getProperty(schema, "latitude"), total_casos.getPais().getLatitud())
            .addProperty(schemaModel.getProperty(schema, "longitude"), total_casos.getPais().getLongitud())
            .addProperty(dboModel.getProperty(dbo, "populationTotal"), total_casos.getPais().getPoblacion())
            .addProperty(dboModel.getProperty(dbo, "grossDomesticProductNominalPerCapita"),total_casos.getPais().

            con.addProperty(dboModel.getProperty(dbo, "country"),rC);
```

Al momento de completar todos los métodos correspondientes a la transformación de datos, comenzamos la ejecución de nuestro proyecto, el mismo nos imprime los siguiente en la consola Figura 9 y en la Figura 10 tenemos el archivo en el directorio anteriormente definido.

Figura 9. Parte de la impresión por consola

```
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://www.w3.org/ns/prov#wasDerivedFrom http://utp
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://www.geonames.org/ontology#locatedIn http://t
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://utpl.edu.ec/sbc/data/Ontology/totalQuantity
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://utpl.edu.ec/sbc/data/Ontology/quantity  "0"
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://schema.org/observationDate  "21/01/2020" .
http://utpl.edu.ec/sbc/dataCOVID/Confr6636 http://www.w3.org/1999/02/22-rdf-syntax-ns#type ht
http://utpl.edu.ec/sbc/dataCOVID/Hosp561 http://www.w3.org/ns/prov#wasDerivedFrom http://utpl
http://utpl.edu.ec/sbc/dataCOVID/Hosp561 http://www.geonames.org/ontology#locatedIn http://utp
http://utpl.edu.ec/sbc/dataCOVID/Hosp561 http://utpl.edu.ec/sbc/data/Ontology/totalQuantity
http://utpl.edu.ec/sbc/dataCOVID/Hosp561 http://schema.org/observationDate  "27/01/2020" .
http://utpl.edu.ec/sbc/dataCOVID/Hosp561 http://www.w3.org/1999/02/22-rdf-syntax-ns#type http
```

Figura 10. Archivo generado en la raíz del proyecto



2.4 Resultados de la transformación

Termina la transformación de los tenemos los siguientes resultados, que están representados en la tabla:

Clase	Número de instancias
Continent	1
Country	51
Dataset	2
Cases	6877
Confirmed_cases	6877
Muertos_cases	6877
Pruebas	6877
Recovered	6877
Active	6877

2.5 Repositorio de almacenamiento

Una vez realizada la transformación de datos a RDF, se procesó a la siguiente etapa que es la búsqueda de un repositorio RDF, que nos permite el almacenamiento del mismo, para ello se tomó en cuenta dos que son GraphDB y Virtuoso.

Posterior a una investigación en busca de información referente a estos dos repositorios RDF, se tomó la decisión de utilizar Virtuoso pues tiene la capacidad de virtualización de los datos lo que nos permite la construcción e implementación de grafos de conocimiento sobre datos existentes por API como HTTP, ODBC, entre otro. Además, nos permite importar y almacenar datos RDF, de manera directa para su uso. Como conclusión final tenemos que virtuoso facilita la administración de base de datos, gafos, tablas, administración de dominios y directorios entre otros.

3. Diseño de Aplicación

3.1 Trabajos relacionados

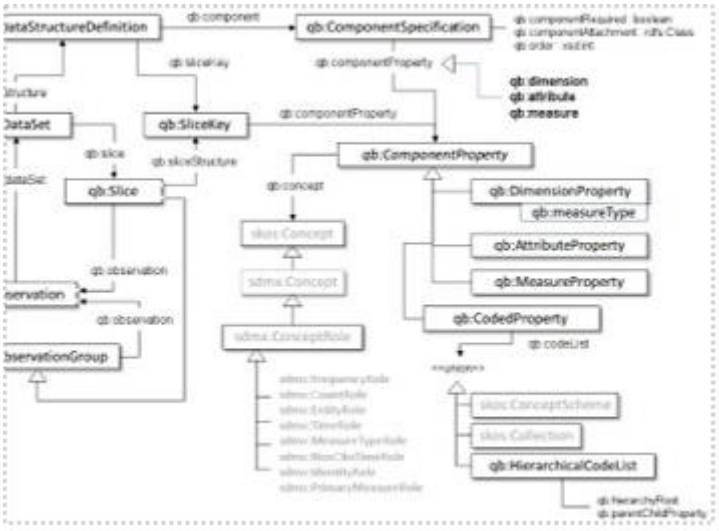
- **COVID 19 MAPS – PAISES EUROPEOS.-** es una aplicación de mapas utiliza datos de Worldmeters y datos de población recuperado por Countryinfo. Permite ver el valor la ratio, calcula cuantas personas contagiadas hay. Si su valor es superior a 1, implica que la enfermedad se sigue propagando.

Figura 11. Imagen de referencia

- **COVID-19 SEMANTIC WEB - DPC RDF DATA CUBE VOCABULARY PROJECT.** Proyecto para modelar conjunto de datos de monitoreo de salud COVID 19 en Italia, datos publicados por el Diario de Protección Civil, disponibles en el repositorio oficial https://github.com/pcm-dpc/COVID-19, presenta un modelo ontológico en donde distribuye la respectiva

información utilizando el modelo de vocabulario RDF DATA CUBE (https://www.w3.org/TR/vocab-data-cube/)

Figura 12. Imagen de referencia



- **CORONAVIRUS - (COVID-19) FULL STACK APPLICATION.-** La idea detrás de esta aplicación es mostrar las estadísticas de Coronavirus COVID-19 en todo el mundo y los datos se están recopilando del Centro de Ciencias e Ingeniería de Sistemas de la Universidad Johns Hopkins JHU CSSE y actualiza los casos constantemente en este sitio web en todo el mundo.

Figura 13. Imagen de Referencia

3.2 Herramientas

Para construir nuestra aplicación se necesitaron de algunas herramientas tecnológica que nos ayudan a terminar la última etapa de nuestro proyecto, las mismo se describen a continuación:

- **Virtuoso:** Virtuoso Universal Server es un híbrido de middleware y motor de base de datos que combina la funcionalidad de un sistema tradicional de gestión de bases de datos

relacionales (RDBMS), base de datos relacional de objetos (ORDBMS), base de datos virtual, RDF , XML , texto libre , servidor de aplicaciones web y servidor de archivos funcionalidad en un solo sistema.

- **Spring Boot:** Spring es un framework alternativo al stack de tecnologías estándar en aplicaciones JavaEE.
- **Dependencia Jena de Spring Boot:** Apache Maven es una herramienta para ayudar a los proyectos Java a administrar sus dependencias en el código de la biblioteca, como Jena. Al declarar una dependencia en el núcleo de Jena en el pom.xml archivo de su proyecto, obtendrá también el conjunto consistente de archivos de biblioteca de los que Jena depende.
- **Angular:** Angular es un framework OpenSource desarrollado por Google para facilitar la creación y programación de aplicaciones web de una sola página, las webs SPA (Single Page Application).

3.3 Consulta realizadas en Sparql

En un inicio se realizaron algunas preguntas sobre los datos que se estaban recolectando, en este apartado se intenta dar contestación a las mismas, por medio de consulta realizadas en Sparql, con el archivo RDF cargado en Virtuoso

Consulta 1: ¿QUÉ PAÍSES EUROPEOS PRESENTAN MAYOR NÚMERO CASOS DE COVID-19?

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX schema: <http://schema.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX gn: <http://www.geonames.org/ontology#>
prefix newOnto:<http://utpl.edu.ec/sbc/data/Ontology/>
prefix data:<http://utpl.edu.ec/sbc/dataCOVID/>
prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix dbr:<http://dbpedia.org/resource/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

# países con mayor casos de covid-19

select  * where {
?URI rdf:type newOnto:Confirmed_Cases ;
     newOnto:totalQuantity ?CasesByDay;
     gn:locatedIn ?country;
     schema:observationDate ?date.
    ?country dbo:name ?name .
}ORDER BY DESC (xsd:integer(?CasesByDay))
```

Resultado:

	name	date	cases
1	"Russia"	"15/06/2020"	"528964"
2	"United Kingdom"	"15/06/2020"	"295889"
3	"Spain"	"15/06/2020"	"244109"
4	"Italy"	"15/06/2020"	"236989"
5	"Germany"	"15/06/2020"	"186461"
6	"France"	"15/06/2020"	"157220"
7	"Belgium"	"15/06/2020"	"60029"
8	"Belarus"	"15/06/2020"	"53973"

Consulta 2: ¿QUÉ PAÍSES EUROPEOS PRESENTAN EL MAYOR ÍNDICE DE LETALIDAD?

```
prefix newOnto:<http://utpl.edu.ec/sbc/data/Ontology/>

prefix data:<http://utpl.edu.ec/sbc/dataCOVID/>

prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

prefix dbr:<http://dbpedia.org/resource/>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>


#Total de Muertes por dia / países con mayor indice de muertes

select  * where {

?var rdf:type newOnto:Muerte_Cases ;

     newOnto:totalQuantity ?cantidad;

     gn:locatedIn ?pais;

     schema:observationDate ?fecha.

    ?pais dbo:name ?nombre .

}ORDER BY DESC (xsd:integer(?cantidad))
```

Resultado

nombre	cantidad
"United Kingdom"	"41736"
"United Kingdom"	"41698"
"United Kingdom"	"41662"
"United Kingdom"	"41481"
"United Kingdom"	"41279"

Consulta 3: ¿PAÍSES CON MAYOR ÍNDICE DE CASOS EN UNA FECHA DETERMINADA?

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

PREFIX schema: <http://schema.org/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX gn: <http://www.geonames.org/ontology#>

prefix newOnto:<http://utpl.edu.ec/sbc/data/Ontology/>

prefix data:<http://utpl.edu.ec/sbc/dataCOVID/>

prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

prefix dbr:<http://dbpedia.org/resource/>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>


# países con mayor casos de covid-19 en una fecha determinada

select  *  where {

?URI rdf:type newOnto:Confirmed_Cases ;

     newOnto:totalQuantity ?NumCases;

     gn:locatedIn ?country;

     schema:observationDate ?date.
```

```
    ?country dbo:name ?name .
    #Filter regex(?name, "^Germany")
    FILTER(?date >= "16/06/2020"^^xsd:string &&
         ?date <= "16/06/2020"^^xsd:string)
}ORDER BY DESC (xsd:integer(?NumCases))
```

Resultado

	name	date	cases
1	"Russia"	"15/06/2020"	"528964"
2	"United Kingdom"	"15/06/2020"	"295889"
3	"Spain"	"15/06/2020"	"244109"
4	"Italy"	"15/06/2020"	"236989"
5	"Germany"	"15/06/2020"	"186461"
6	"France"	"15/06/2020"	"157220"
7	"Belgium"	"15/06/2020"	"60029"

Consulta 4: ¿QUÉ PAÍSES MAS PRUEBAS COVID-19 HAN REALIZADO?

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX schema: <http://schema.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX gn: <http://www.geonames.org/ontology#>
prefix newOnto:<http://utpl.edu.ec/sbc/data/Ontology/>
prefix data:<http://utpl.edu.ec/sbc/dataCOVID/>
prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix dbr:<http://dbpedia.org/resource/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
# países que han superado los 1000 test diarios
select  ?name  (COUNT(?name) as ?n) where {
?URI rdf:type newOnto:Pruebas ;
     newOnto:quantity ?test;
```

```
      gn:locatedIn ?country;
      schema:observationDate ?date.
    ?country dbo:name ?name .
    #Filter regex(?name, "^Germany")
    FILTER(?test >= "1000"^^xsd:integer)
}
GROUP BY ?name
ORDER BY ASC (xsd:integer(?test))
```

Resultado

	name	n
1	"France"	"20"^^xsd:integer
2	"Belarus"	"36"^^xsd:integer
3	"Bulgaria"	"43"^^xsd:integer
4	"Ukraine"	"48"^^xsd:integer
5	"United Kingdom"	"50"^^xsd:integer
6	"Norway"	"58"^^xsd:integer
7	"Greece"	"63"^^xsd:integer
8	"Luxembourg"	"64"^^xsd:integer

3.4 Diseño de la aplicación

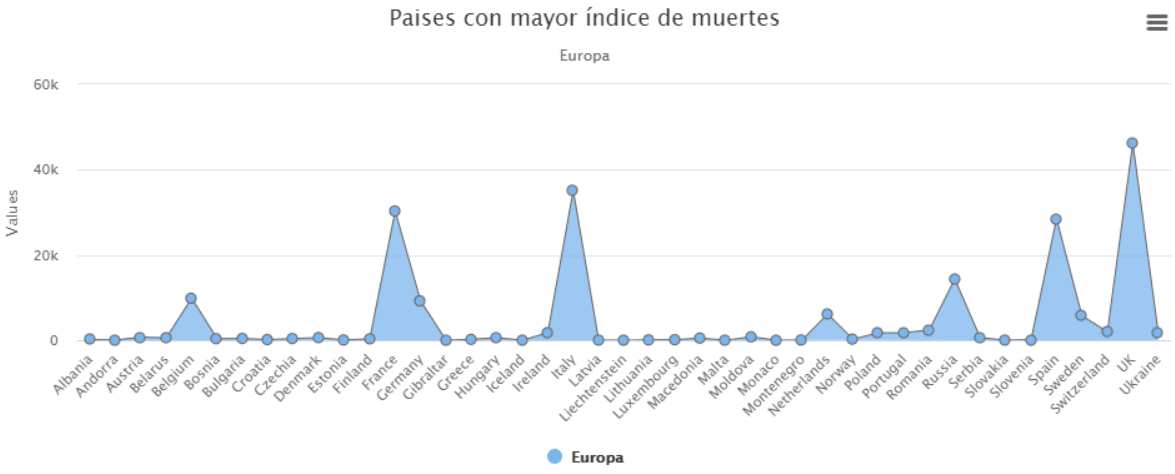
El presente diseño se lo ha realizado con de explicar cómo es la arquitectura de nuestra aplicación

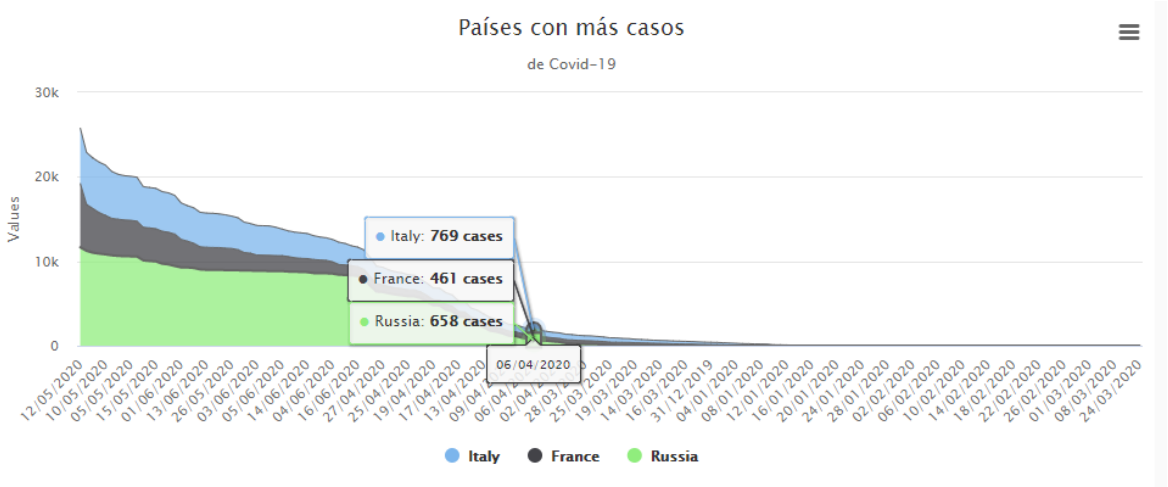
Lo que se quiere señalar con la arquitectura es que se encuentra almacenado un archivo RDF en virtuoso el mismo que será consumido por Spring Boot, mediante una conexión que es permitida por jdbc:virtuoso. Una vez consumida la data se procede a la creación de API, que sus resultados serán consumidos por cualquier aplicación, en este caso para probar utilizamos Angular.

La API le definimos algunas direcciones dependiendo el resultado que desee, para un ejemplo utilizamos a Francia en donde consumimos los caso de ese y se tendrá su resultado es el siguiente:


```
[
  {
    pruebas: "33728",
    fecha: "24/05/2020",
    pais: "France"
  },
  {
    pruebas: "33728",
    fecha: "25/05/2020",
    pais: "France"
  },
  {
    pruebas: "33728",
    fecha: "26/05/2020",
    pais: "France"
  },
  {
    pruebas: "33728",
    fecha: "27/05/2020",
    pais: "France"
  },
  {
    pruebas: "33728",
    fecha: "28/05/2020",
    pais: "France"
  },
```


Una vez que la API están generadas se las debe consumir desde el Front-end que como se dijo anteriormente esta realizado en Angular. Como se observa en la siguiente imagen se presenta una grafica.





Num.	País	Casos	Muertes	Recuperados
0	Albania	5620	172	3031
1	Andorra	937	52	821
2	Austria	21385	718	19063
3	Belarus	68166	571	62943
4	Belgium	70314	9850	17598

Items per page: 5 1 – 5 of 43

4. Conclusiones

El proyecto realizado anteriormente sobre la generación de datos RDF nos permite evidenciar que se debe seguir una serie de etapas para la construcción y consumo de los datos que son almacenados en repositorios según los requerimientos de velocidad que requiera el consumidor.

Una vez alcanzado el objetivo del proyecto, se puede argumentar que cada etapa desarrollada se debe realizarse de manera efectiva y ágil para conseguir que al momento de su publicación los datos sean legibles para las distintas aplicaciones informáticas.