Universita degli Studi dell' Aquila

Open Data and Web Services (ODWS)

Final Project

by

Jai Jobe, Lamin Ann & Michael Dubem Igbomezie

jai.jobe@student.univaq.it

lamin.ann@student.univaq.it

michaeldubem.igbomezie@student.univaq.it

Lecturers: Marco Autili & Claudio Pompilio

marco.autili@univaq.it , claudio.pompilio@univaq.it

Table of Contents

1. Introduction

Datafication, Digitisation and System Integration has become a crucial part of today's world and connected systems, especially when we see the great potentials of integrated web services, third party libraries, model engineering and Open Data. These areas have collectively reduced the time it takes to set up functioning complex systems. It has reduced the error that comes from spreading efforts too thin in trying to "reinvent the wheel", and has further facilitated knowledge sharing and expertise integration.

Open Data refers to the information collected, produced or paid for by the public bodies and made freely available for re-use for any purpose. Its benefits are diverse and range from improved efficiency of public administrations, economic growth in the private sector to wider social welfare.

Thanks to Web services, clients or other web-based programs can access these data or information using standardized messaging protocols. Web services are built using different protocols in order to integrate with various applications. However, we use Representational State Transfer (REST) which is a set of guidelines that offers flexible implementation. While not all web services use the REST protocol, applications built with RESTAPIs are more lightweight, manageable and scalable.

The aim of this project is to implement a web service which harnesses Open Data to make some information available to users. This involved the gathering of licensed data in a certain knowledge domain, from open data platforms and communities, cleaning, enriching and linking of these datasets, and converting them into RDF graph so as to query these graphs and make their content available on a certain endpoint.

2. Objective

2.1 Primary Objective

The primary objective of this project is to successfully implement a REST service using the specified software which includes the OpenRefine, Eclipse IDE, Apache Tomcat, Apache CXF and Apache Maven, which exposes an RDF graph (extracted from the cleaned and converted data loaded in OpenRefine) using SPARQL queries.

2.2 Secondary Objective

The secondary objective of this project is focused on the type of information to be made available through the REST service. The area of interest of this projects was the economic impact of the COVID-19 pandemic on 8 major countries in the world, the health impact on their population, their readiness to such an event prior to the outbreak and their reaction during the outbreak.

3. Methodology

3.1 Knowledge Domain Selection

Covid-19 is the world's greatest pandemic in the recent years. It has led to a dramatic loss of human life worldwide and presents an unprecedented challenge to public health, food systems and the world of work. The economic and social disruption caused by the pandemic is devastating: tens of millions of people are at risk of falling into extreme poverty, while the number of undernourished people, currently estimated at nearly 690 million, could increase by up to 132 million by the end of the year. These factors triggered our interest in picking different datasets about COVID-19.

3.2 Data Gathering and Project Creation on Open Refine

Kaggle is an online community platform that allow users to find and publish datasets. Below are the selected Datasets downloaded from Kaggle:

- COVID 19 Impact on Global Economy

- Economic Data for - 9 Countries (1980-2020)

Economic Data - 9 Countries (1980-2020)

Movement of Stock Market with Macroeconomic changes

Data | Code (0) | Discussion (0) | Metadata

About Dataset

The dataset contains data for 8 countries and one special administrative region (China, France, Germany, Hong Kong, India, Japan, Spain, United Kingdom and United States of America) from 1980 through 2020. It include major macroeconomic factors like inflation, unemployment, GDP, exchange rate (base USD) and per capita income. Apart from that it has the stock prices of the respective country's major stock index which can help in analysing the data set to identify the impact of major macroeconomic variables on the movement of stock index prices.

Investing | Economics

Usability 9.12

License CC BY-NC-SA 4.0

Expected update frequency Annually

Economic Data - 9 Countries (1980-2020).csv (34 KiB)

Detail | Compact | Column | 10 of 14 columns

About this file

From 1980 through 2020, data is available for eight nations and one special administrative region (China, France, Germany, Hong Kong, India, Japan, Spain, the United Kingdom, and the United States of America). Major macroeconomic variables such as inflation, unemployment, GDP, exchange rate (base USD), and per capita income are included.

Data Explorer 34 KiB

Economic Data - 9 Countrie...

- Countries by Government Budget per Capita

List_of_countries_by_government_budget_per_capita

List_of_countries_by_government_budget_per_capita.csv

Data | Code (0) | Discussion (0) | Metadata

About Dataset

This dataset is extracted from https://en.wikipedia.org/wiki/List_of_countries_by_government_budget_per_capita. Context: There s a story behind every dataset and heres your opportunity to share yours.Content: What s inside is more than just rows and columns. Make it easy for others to get started by describing how you acquired the data and what time period it represents, too. Acknowledgements:We wouldn t be here without the help of others. If you owe any attributions or thanks, include them here along with any citations of past research.Inspiration: Your data will be in front of the world s largest data science community. What questions do you want to see answered?

Usability 5.88

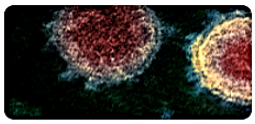
License CC0: Public Domain

Expected update frequency Not specified

- COVID 19 Case Reported in 2020

COVID-19 Dataset

Number of Confirmed, Death and Recovered cases every day across the globe



Data | Code (581) | Discussion (41) | Metadata

About Dataset

MADE WITH PYTHON | USES GIT

Context

- A new coronavirus designated 2019-nCoV was first identified in Wuhan, the capital of China's Hubei province
- People developed pneumonia without a clear cause and for which existing vaccines or treatments were not effective.
- The virus has shown evidence of human-to-human transmission
- Transmission rate (rate of infection) appeared to escalate in mid-January 2020
- As of 30 January 2020, approximately 8,243 cases have been confirmed

Content

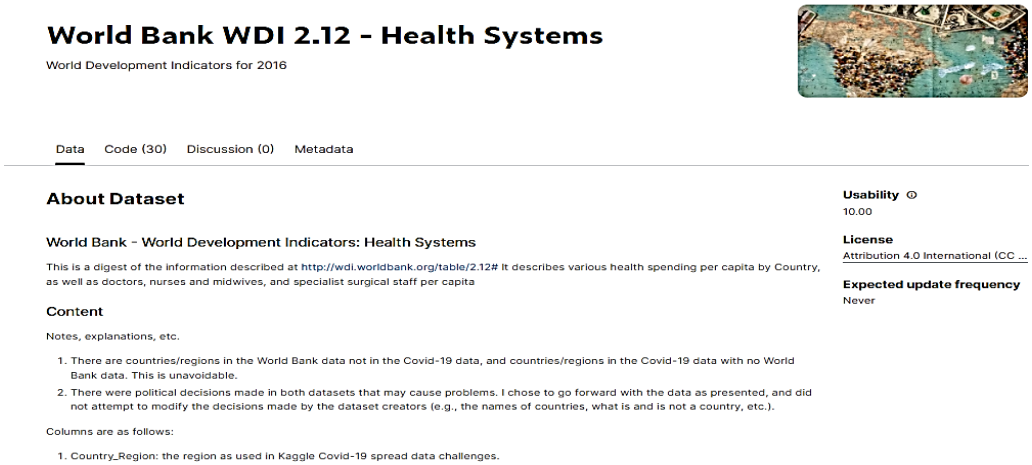
- full_grouped.csv - Day to day country wise no. of cases (Has County/State/Province level data)
- covid19clean_complete.csv - Day to day country wise no. of cases (Doesn't have County/State/Province level data)

Usability 10.00

License Other (specified in description)

Expected update frequency Daily

- Public Health Safety measure and severity



World Bank WDI 2.12 - Health Systems

World Development Indicators for 2016

Data Code (30) Discussion (0) Metadata

About Dataset

World Bank - World Development Indicators: Health Systems

This is a digest of the information described at http://wdi.worldbank.org/table/2.12# It describes various health spending per capita by Country, as well as doctors, nurses and midwives, and specialist surgical staff per capita

Content

Notes, explanations, etc.

1. There are countries/regions in the World Bank data not in the Covid-19 data, and countries/regions in the Covid-19 data with no World Bank data. This is unavoidable.
2. There were political decisions made in both datasets that may cause problems. I chose to go forward with the data as presented, and did not attempt to modify the decisions made by the dataset creators (e.g., the names of countries, what is and is not a country, etc.).

Columns are as follows:

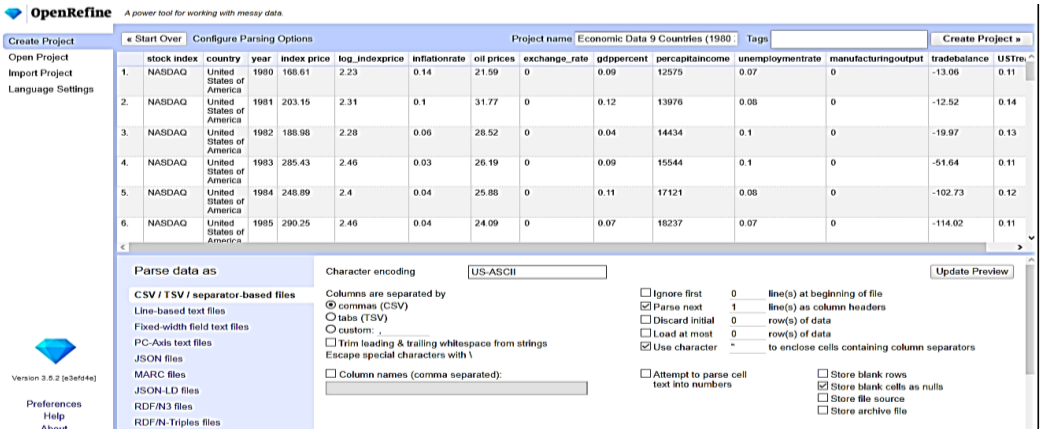
1. Country_Region: the region as used in Kaggle Covid-19 spread data challenges.

Usability 10.00

License Attribution 4.0 International (CC ...

Expected update frequency Never

To create a project on OpenRefine; we imported the dataset and select 'Create Project' at the top right corner.

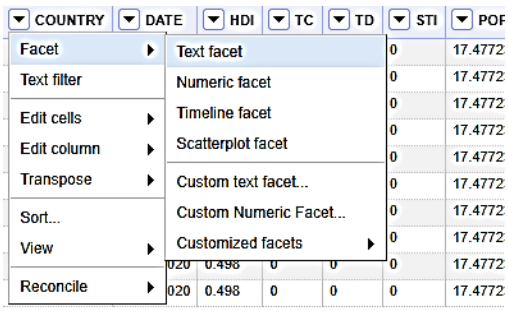


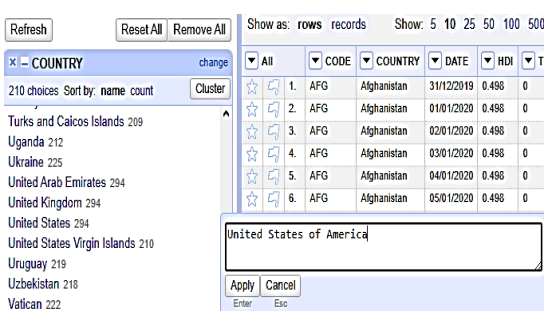
Project created with dataset that contains 369 number of rows.

3.3 Data Cleaning using Open Refine

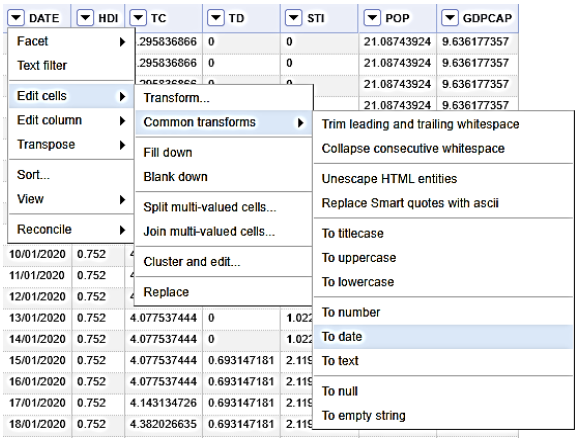
Almost every dataset we encountered was messy. There were inconsistencies in the way the data was entered, this made the data difficult to analyse and that made it compulsory for us to clean it.

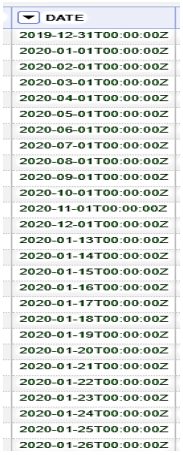
Text facet was use on this dataset to correct country column 'United States of America' and multiple representation of the same category (misspelling). It also confirmed that all dates were having the same format.





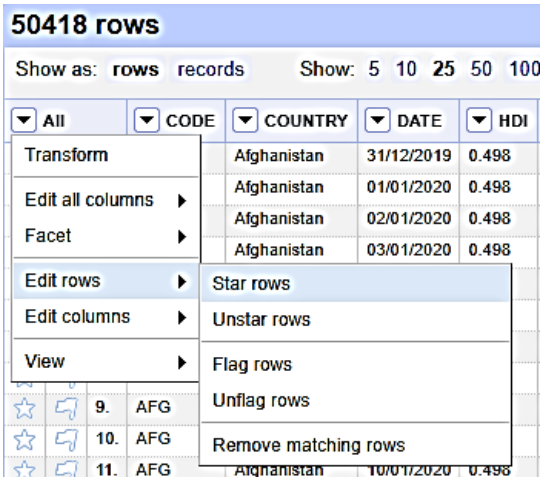
We used **Cell transformation** on the date column. The aim was to convert the date column to Datetime format.

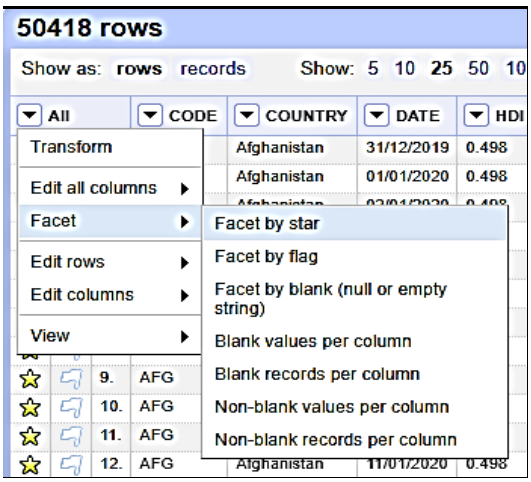


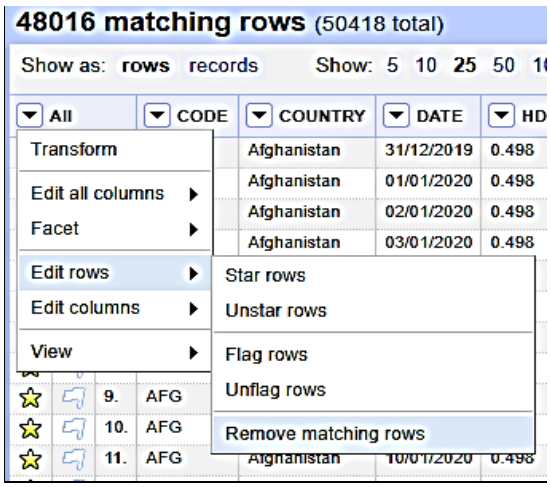


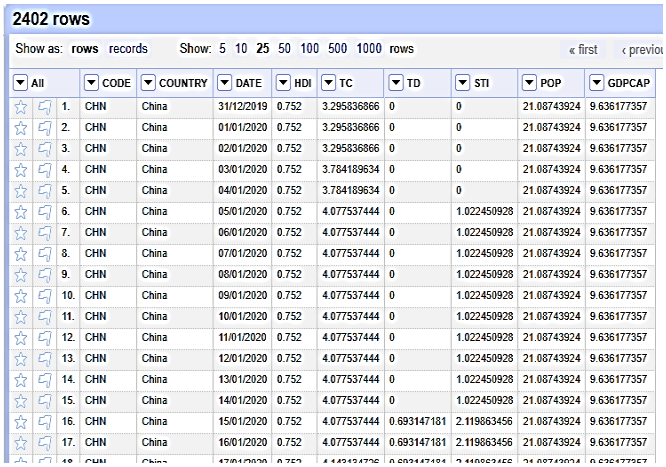
Redundant data: this means there are multiple copies of the same information spread over the dataset.

We **starred** all redundant rows on this particular dataset, applied '**Facet by star**' and then **removed matching rows** from this particular dataset.









2402 rows

Show as: rows records Show: 5 10 25 50 100 500 1000 rows « first ‹ previou

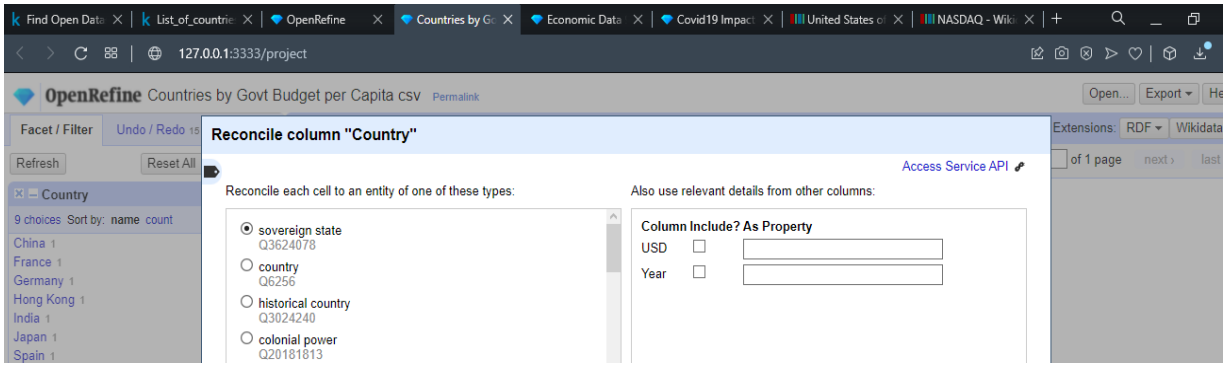
All		CODE	COUNTRY	DATE	HDI	TC	TD	STI	POP	GDPCAP
☆	1.	CHN	China	31/12/2019	0.752	3.295836866	0	0	21.08743924	9.636177357
☆	2.	CHN	China	01/01/2020	0.752	3.295836866	0	0	21.08743924	9.636177357
☆	3.	CHN	China	02/01/2020	0.752	3.295836866	0	0	21.08743924	9.636177357
☆	4.	CHN	China	03/01/2020	0.752	3.784189634	0	0	21.08743924	9.636177357
☆	5.	CHN	China	04/01/2020	0.752	3.784189634	0	0	21.08743924	9.636177357
☆	6.	CHN	China	05/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	7.	CHN	China	06/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	8.	CHN	China	07/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	9.	CHN	China	08/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	10.	CHN	China	09/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	11.	CHN	China	10/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	12.	CHN	China	11/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	13.	CHN	China	12/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	14.	CHN	China	13/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	15.	CHN	China	14/01/2020	0.752	4.077537444	0	1.022450928	21.08743924	9.636177357
☆	16.	CHN	China	15/01/2020	0.752	4.077537444	0.693147181	2.119863456	21.08743924	9.636177357
☆	17.	CHN	China	16/01/2020	0.752	4.077537444	0.693147181	2.119863456	21.08743924	9.636177357

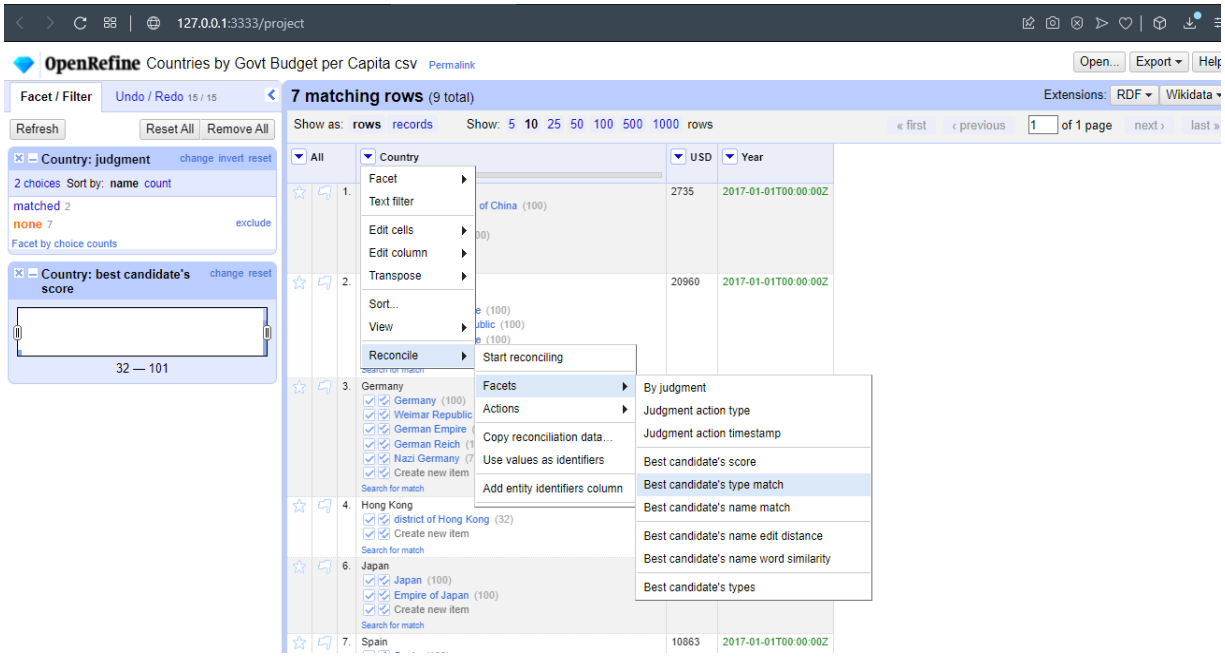
This is the data with no redundancy. A total of 48016 matching rows were removed reducing the dataset to 2402 number of rows.

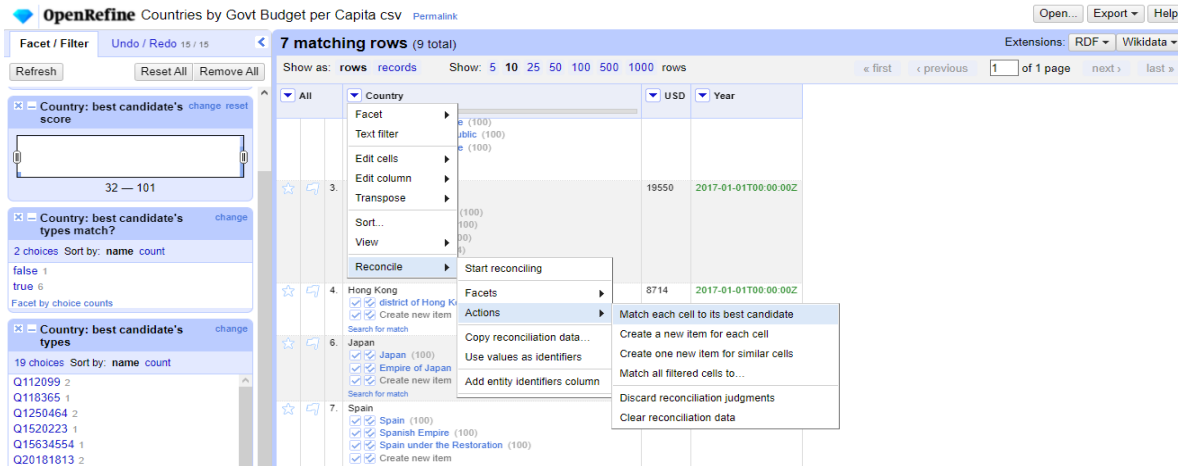
Similar actions were carried out on the remaining datasets.

3.4 Data Enriching and Linking

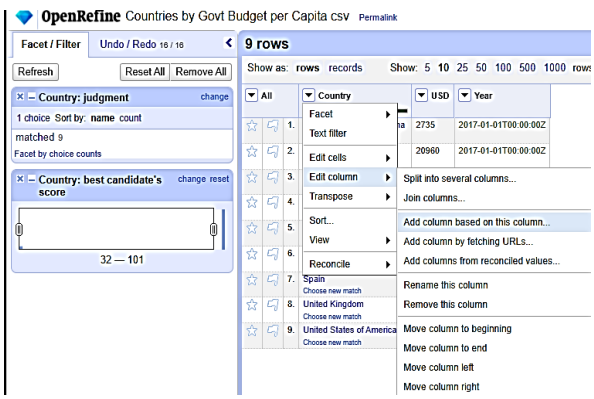
Reconciliation is the process of matching a dataset with external datasets. We Reconciled the country column by adding the Wikidata Standard, and selected the Wikidata Standard service, followed by the below steps:

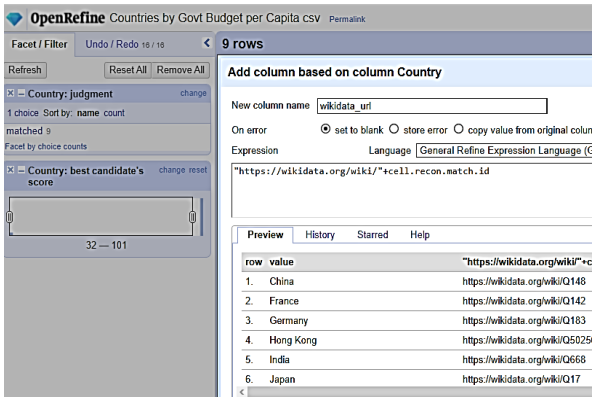


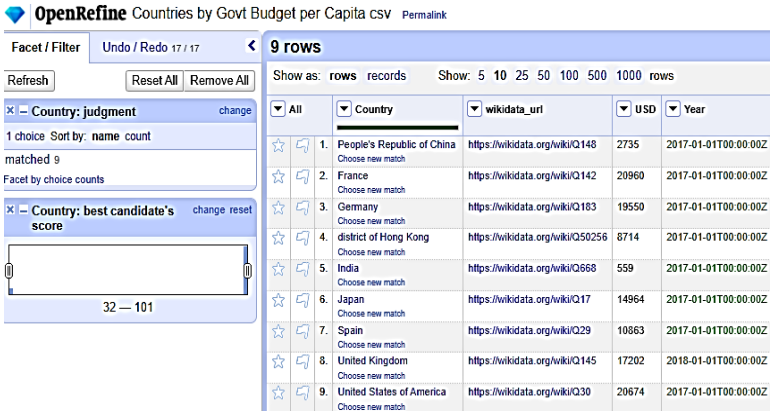




We then **enriched** the dataset with the Wikidata URLs



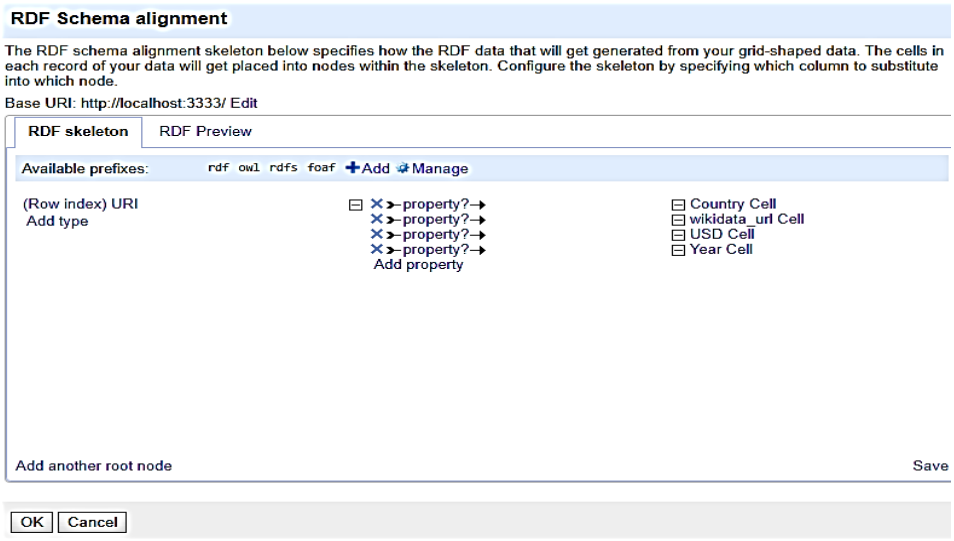




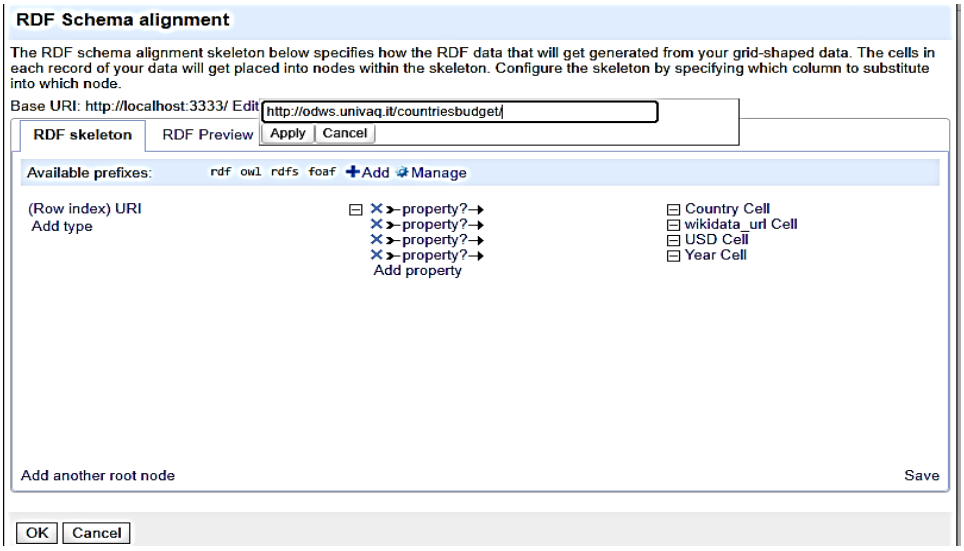
This is the resulting dataset after enriching and linking.

3.5 RDF Mapping and Exportation

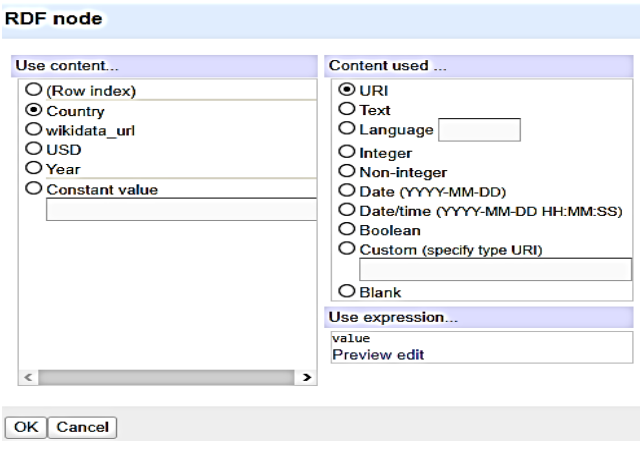
RDF schema alignment



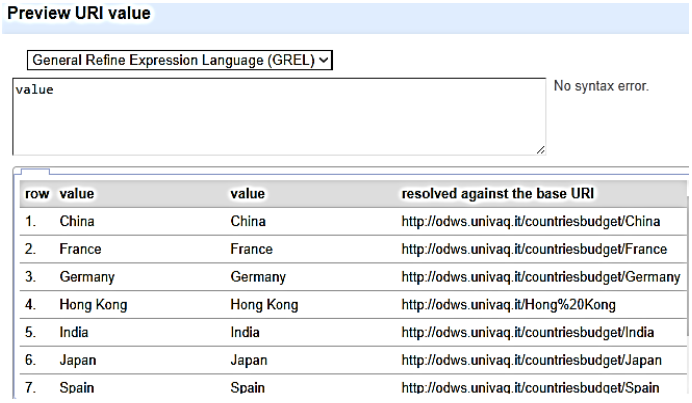
We edited the Base URI



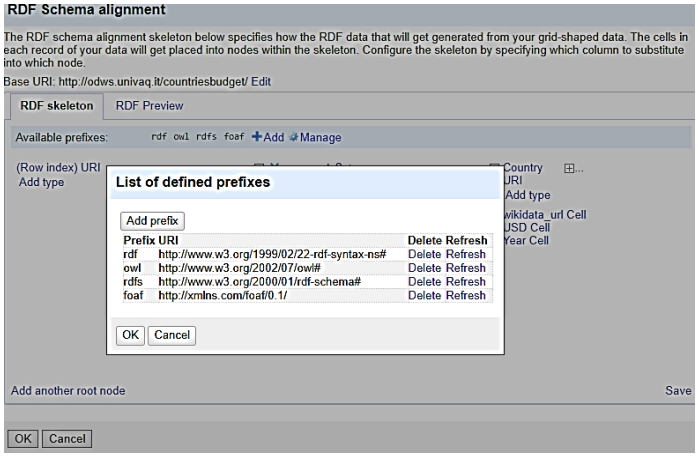
We then set up the identity in URIs



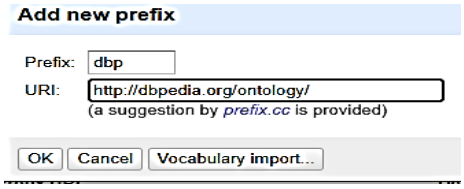
Linked Data Principles

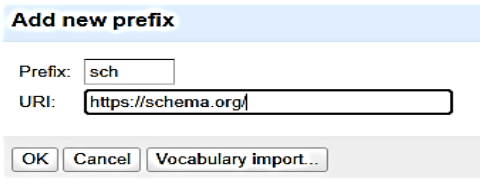


Default prefixes

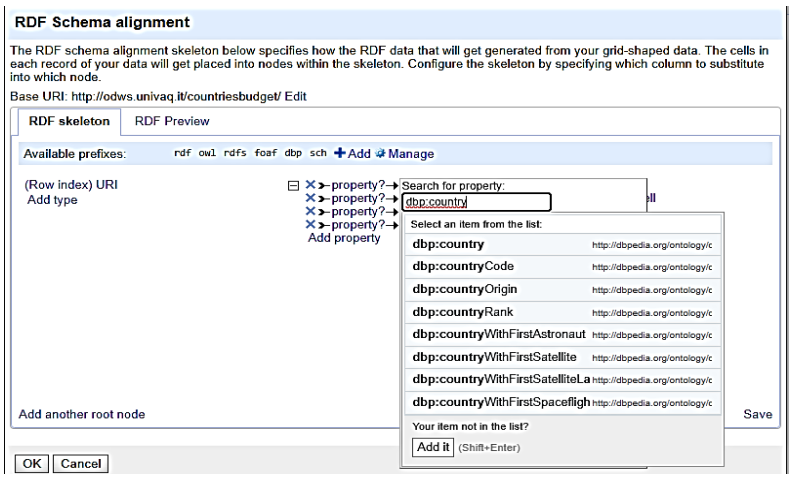


We added prefixes for dbpedia.org and schema.org



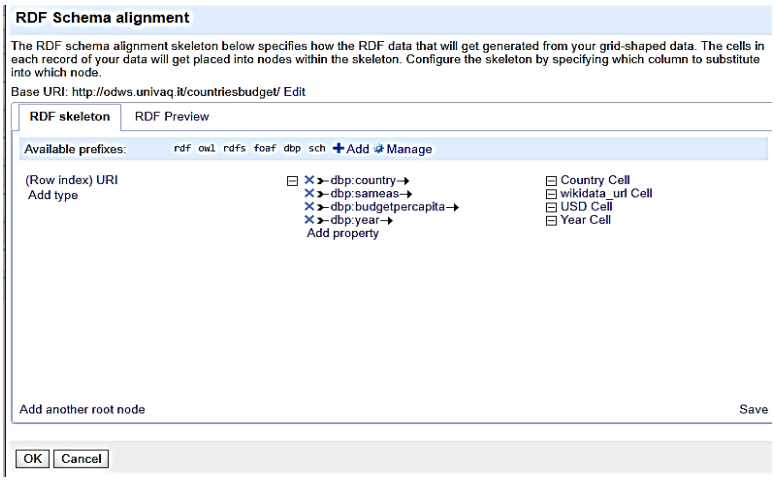


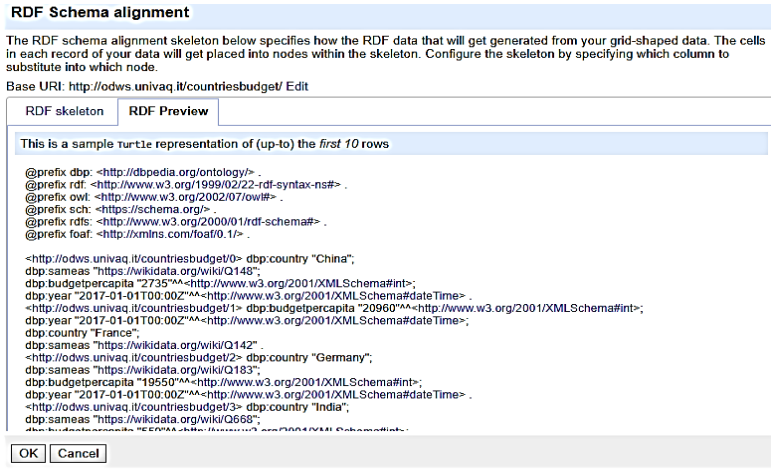
Then we mapped the country cell



Similarly, wikidata_url cell, USD cell and Year cell were all mapped in the same way.

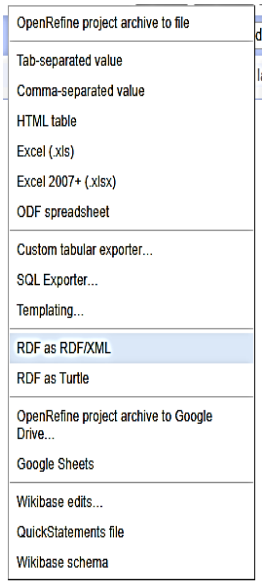
Below is the resulting mapping and the RDF preview

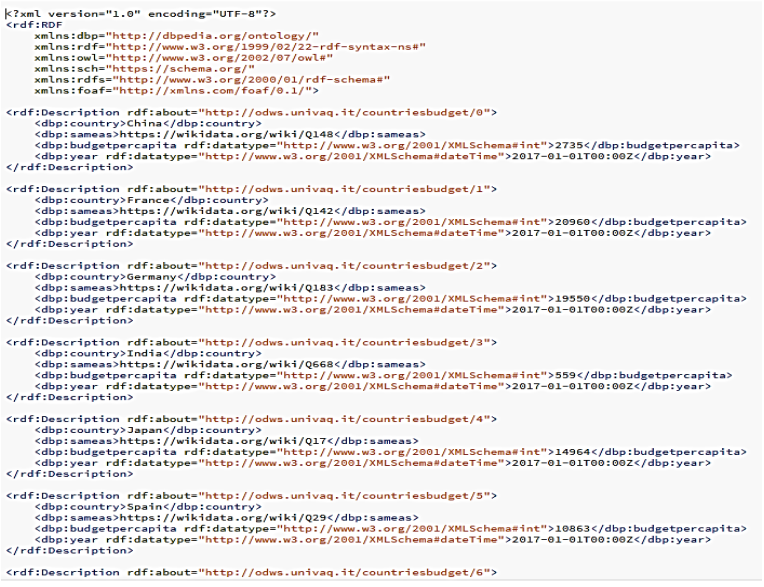




Exportation

We exported the data as RDF/XML





```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
    xmlns:dbp="http://dbpedia.org/ontology/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:sch="https://schema.org/"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/">

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/0">
    <dbp:country>China</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q148</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">2735</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/1">
    <dbp:country>France</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q142</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">20960</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/2">
    <dbp:country>Germany</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q183</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">19550</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/3">
    <dbp:country>India</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q668</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">559</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/4">
    <dbp:country>Japan</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q17</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">14964</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/5">
    <dbp:country>Spain</dbp:country>
    <dbp:sameas>https://wikidata.org/wiki/Q29</dbp:sameas>
    <dbp:budgetpercapita rdf:datatype="http://www.w3.org/2001/XMLSchema#int">10863</dbp:budgetpercapita>
    <dbp:year rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2017-01-01T00:00Z</dbp:year>
</rdf:Description>

<rdf:Description rdf:about="http://odws.univaq.it/countriesbudget/6">
```

3.6 SPARQL Queries

SPARQL is fundamentally a language for formulating queries, through which information can be retrieved from datasets. However, since it is targeted at datasets published on the World Wide Web, it also provide a protocol for specifying SPARQL commands.

The following query types were used in the Apache Jena to retrieve the important information we needed to know from the linked datasets:
SELECT: the query returns a table in which columns represent variables and rows represent variable bindings matching the search pattern.
FROM clause: defines the graph to query. By default, if a query omits FROM clauses, the scope of the query is limited to the default graph (DEFAULTSET).

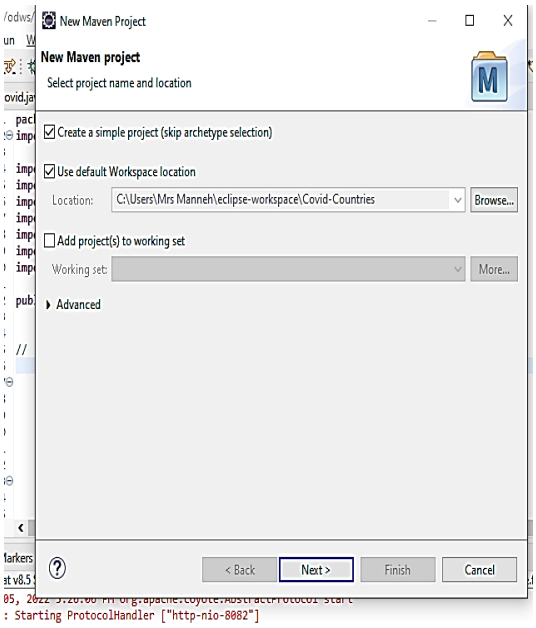
In the result content, we showed the query of individual graphs and a combined graph.

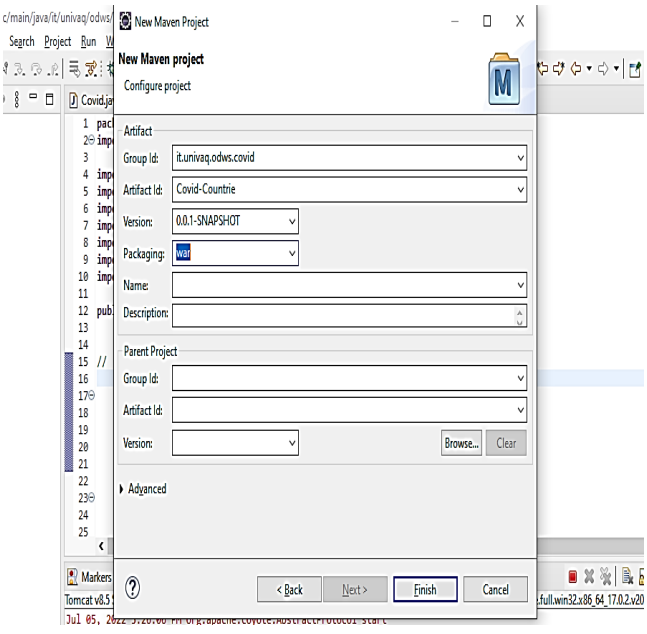
3.7 REST Web Service: Is an architectural style based on transferring representations of resources from a server to a client. We use rest and maven for the implementation of the project. The maven help us to load and install some of the dependencies that will be needed throughout the project. The rest help us to render our project as a client service in other to extract data from the RDF files.

The plugins that were installed includes:

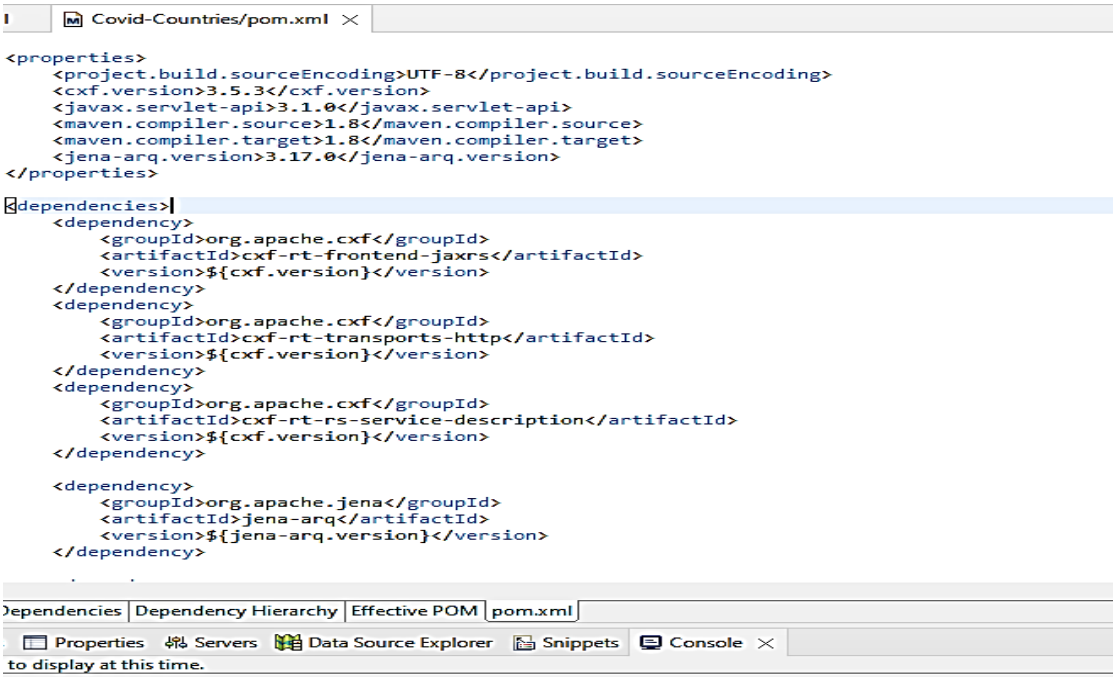
- Eclipse IDE
- Apache Tomcat
- Apache CXF
- Maven

We created a new Maven Project

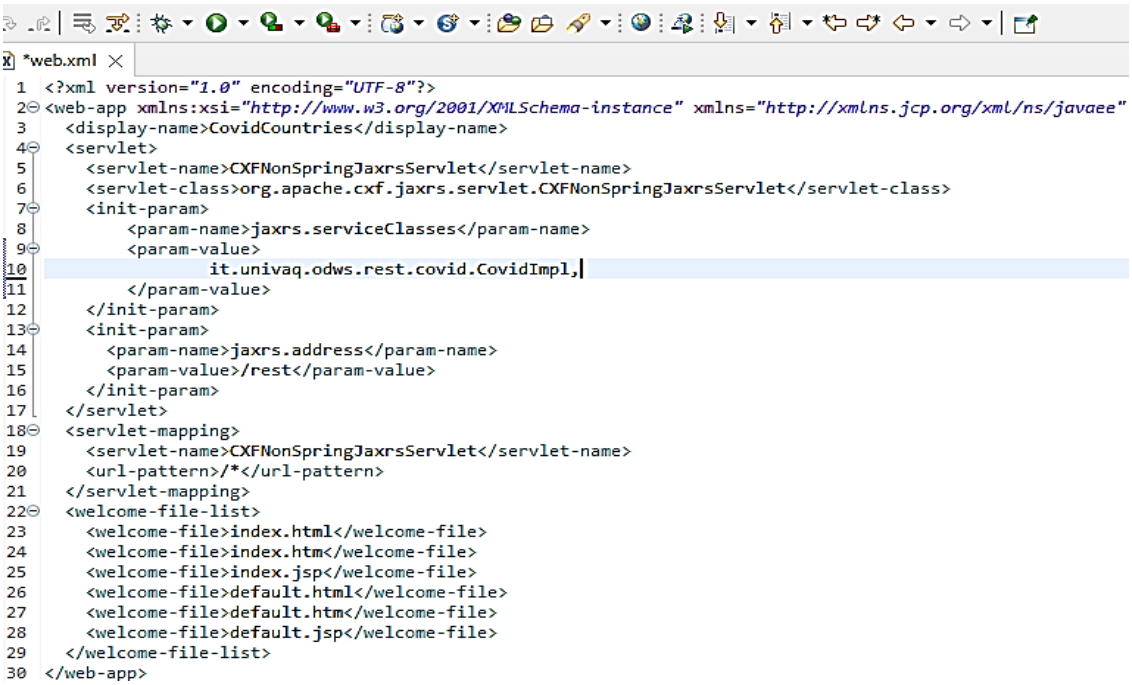




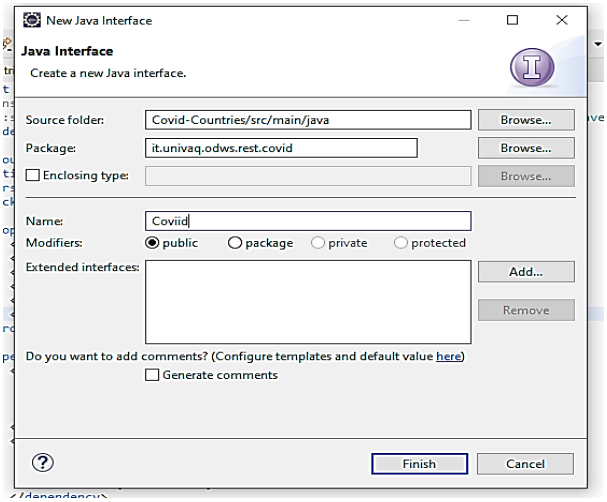
We edited the POM (Project Object Model) file



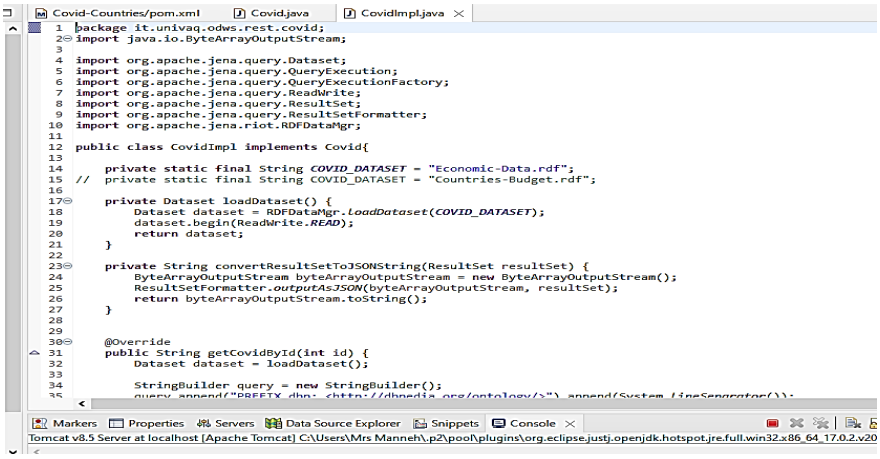
We edited the web XML file



We created a COVID interface



We created the implementation class, (CovidImpl), so as to implement the Covid interface.



We deployed our service

We then queried this dataset by index 22

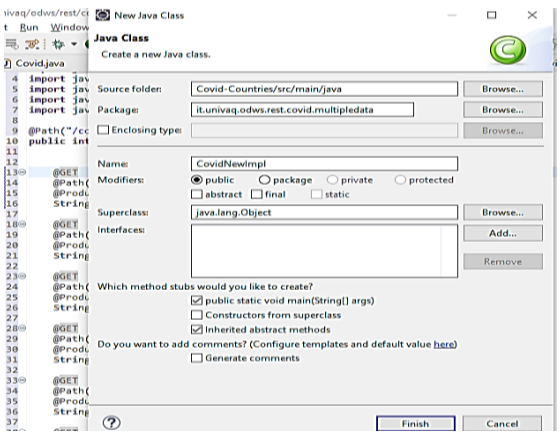
```
    }


    @Override
    public String getCovidById(int id) {
        Dataset dataset = loadDataset();

        StringBuilder query = new StringBuilder();
        query.append("PREFIX dbp: <http://dbpedia.org/ontology/>").append(System.lineSeparator());
        query.append("PREFIX sch: <https://schema.org/>").append(System.lineSeparator());
        query.append("SELECT ?country  ?stockindex ?sameasstockindex ?gdppercent ?year ?indexprice ?inflationrate ?imdb_url ?wikidata_url")
        query.append("WHERE {").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+">  dbp:country ?country.").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+">  dbp:stockindex ?stockindex.").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+">  dbp:sameasstockindex ?sameasstockindex.").append(System.lineS
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+"> dbp:gdppercent ?gdppercent.").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+"> dbp:year ?year.").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+"> dbp:indexprice ?indexprice.").append(System.lineSeparator());
        query.append("  <http://odws.univaq.it/countrieseconomicdata/"+id+"> dbp:inflationrate ?inflationrate").append(System.lineSeparator
        query.append("}").append(System.lineSeparator());

        QueryExecution queryExecution = QueryExecutionFactory.create(query.toString(), dataset);
        ResultSet resultSet = queryExecution.execSelect();
        return convertResultSetToJSONString(resultSet);
    }

```

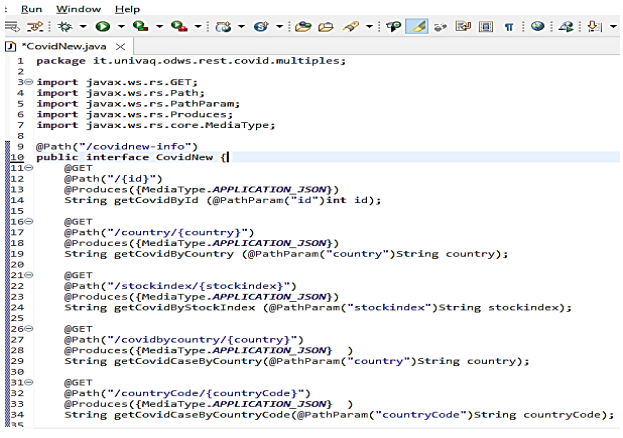
We also queried by country Name

```
ovid-Countries/pom.xml   Covid.java   CovidImpl.java

    @Override
    public String getCovidByCountry(String country) {
        // TODO Auto-generated method stub
        Dataset dataset = loadDataset();

        StringBuilder query = new StringBuilder();
        query.append("PREFIX dbp: <http://dbpedia.org/ontology/>").append(System.lineSeparator());
        query.append("PREFIX sch: <https://schema.org/>").append(System.lineSeparator());
        query.append("SELECT ?country ?year ?gdppercent ?indexprice ?stockindex ?sameasstockindex ?inflationrate ").append(System.lineSepa
        query.append("WHERE {").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:country ?country.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:stockindex ?stockindex.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:sameasstockindex ?sameasstockindex.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:year ?year.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:gdppercent ?gdppercent.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:indexprice ?indexprice.").append(System.lineSeparator());
        query.append("  ?countrieseconomicdata dbp:inflationrate ?inflationrate.").append(System.lineSeparator());
        query.append("  FILTER((LCASE(?country) = \"" + country.toLowerCase() + "\"))").append(System.lineSeparator());
        query.append("}").append(System.lineSeparator());

        QueryExecution queryExecution = QueryExecutionFactory.create(query.toString(), dataset);
        ResultSet resultSet = queryExecution.execSelect();
        return convertResultSetToJSONString(resultSet);
    }
```

We filtered NASDAQ from the dataset

We then created another Package for implementing multiple RDF files.





CovidNew.java | CovidNewImpl.java

```java
package it.univaq.odws.rest.covid.multiples;

import java.io.ByteArrayOutputStream;

import org.apache.jena.query.Dataset;
import org.apache.jena.query.QueryExecution;
import org.apache.jena.query.QueryExecutionFactory;
import org.apache.jena.query.ReadWrite;
import org.apache.jena.query.ResultSet;
import org.apache.jena.query.ResultSetFormatter;
import org.apache.jena.riot.RDFDataMgr;

public class CovidNewImpl implements CovidNew{

    private static final String COVID_DATASET = "Economic-Data.rdf";
    private static final String COVID_DATASET_NEW = "Covid19-Impact.rdf";

    private Dataset loadDataset() {
        Dataset dataset = RDFDataMgr.loadDataset(COVID_DATASET);
        RDFDataMgr.read(dataset, COVID_DATASET_NEW);
        dataset.begin(ReadWrite.READ);
        return dataset;
    }

    private String convertResultSetToJSONString(ResultSet resultSet) {
        ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
        ResultSetFormatter.outputAsJSON(byteArrayOutputStream, resultSet);
        return byteArrayOutputStream.toString();
    }


    @Override
    public String getCovidById(int id) {
        Dataset dataset = loadDataset();
```

We filtered by stock index NASDAQ in this data set.

```java
// Filtering by Stockindex NASDAQ in the Data
@Override
public String getCovidCaseByCountry(String country) {

    Dataset dataset = loadDataset();

    StringBuilder query = new StringBuilder();
    query.append("PREFIX dbp: <http://dbpedia.org/ontology/>").append(System.lineSeparator());
    query.append("PREFIX sch: <https://schema.org/>").append(System.lineSeparator());
    query.append("SELECT ?country ?year ?gdppercent ?indexprice ?stockindex ?sameasstockindex ?inflationrate ").append(System.lineSeparator());
    query.append("WHERE {").append(System.lineSeparator());
    query.append("  ?countrieseconomicdata dbp:country ?country.").append(System.lineSeparator());
    query.append("  ?countrieseconomicdata dbp:year ?year.").append(System.lineSeparator());
    query.append("  ?countrieseconomicdata dbp:gdppercent ?gdppercent.").append(System.lineSeparator());
    query.append("  ?countrieseconomicdata dbp:indexprice ?indexprice.").append(System.lineSeparator());
    query.append("  ?countrieseconomicdata dbp:stockindex ?stockindex.").append(System.lineSeparator());
    query.append("  FILTER(CONTAINS(?stockindex, \"NASDAQ\") && ?country = \"" + country + "\")").append(System.lineSeparator());
    query.append("}").append(System.lineSeparator());

    QueryExecution queryExecution = QueryExecutionFactory.create(query.toString(), dataset);
    ResultSet resultSet = queryExecution.execSelect();
    return convertResultSetToJSONString(resultSet);
}
```

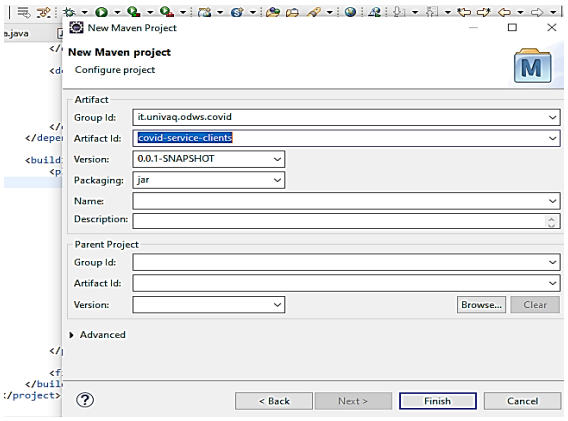
We implemented the code to append country code in the economic data set.

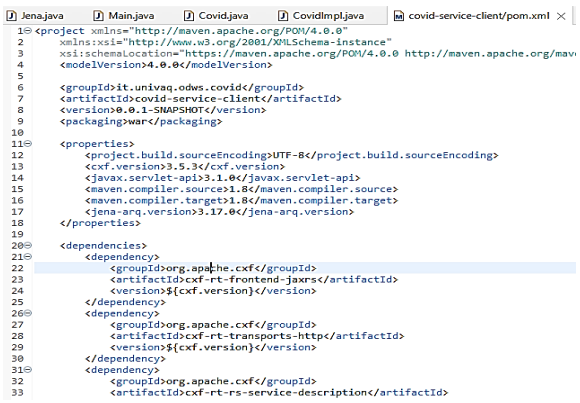
CovidNew.java | CovidNewImpl.java

```java
        QueryExecution queryExecution = QueryExecutionFactory.create(query.toString(), dataset);
        ResultSet resultSet = queryExecution.execSelect();
        return convertResultSetToJSONString(resultSet);
    }

    @Override
    public String getCovidCaseByCountryCode(String countryCode) {
        Dataset dataset = loadDataset();

        StringBuilder query = new StringBuilder();
        query.append("PREFIX dbp: <http://dbpedia.org/ontology/>").append(System.lineSeparator());
        query.append("PREFIX sch: <https://schema.org/>").append(System.lineSeparator());
        query.append("SELECT ?country ?countryCode ?year ?gdppercent ?indexprice ?stockindex ?sameasstockindex ?inflationrate ").append
        query.append("WHERE {").append(System.lineSeparator());
        query.append("  ?covid dbp:country ?country.").append(System.lineSeparator());
        query.append("  ?covidnew dbp:countryCode ?countryCode.").append(System.lineSeparator());
        query.append("  ?covid dbp:stockindex ?stockindex.").append(System.lineSeparator());
        query.append("  ?covid dbp:sameasstockindex ?sameasstockindex.").append(System.lineSeparator());
        query.append("  ?covid dbp:year ?year.").append(System.lineSeparator());
        query.append("  ?covid dbp:gdppercent ?gdppercent.").append(System.lineSeparator());
        query.append("  ?covid dbp:indexprice ?indexprice.").append(System.lineSeparator());
        query.append("  ?covid dbp:inflationrate ?inflationrate.").append(System.lineSeparator());
        query.append("  FILTER((LCASE(?countryCode) = \"" + countryCode.toLowerCase() + "\"))").append(System.lineSeparator());
        query.append("}").append(System.lineSeparator());

        QueryExecution queryExecution = QueryExecutionFactory.create(query.toString(), dataset);
        ResultSet resultSet = queryExecution.execSelect();
        return convertResultSetToJSONString(resultSet);
    }
```

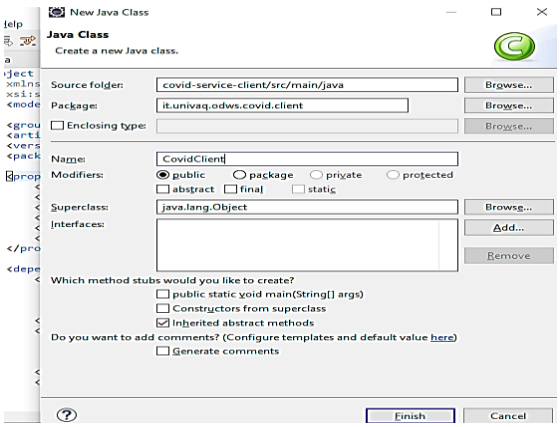
3.8 Java Client

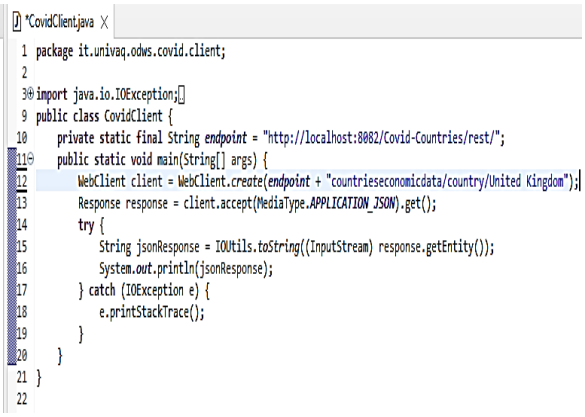
We created a java client project, and edited the java client POM file.





We then created the java client class and implemented it.





4. Results

We got the following result from each individual dataset:

Countries by Government Budget Per Capita at index 1


```
{ "head": {
    "vars": [ "country" , "sameas" , "budgetpercapita" , "year" , "imdb_url" , "wikidata_url" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "France" } ,
        "sameas": { "type": "literal" , "value": "https://wikidata.org/wiki/Q142" } ,
        "budgetpercapita": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "20960" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2017-01-01T00:00Z" }
      }
    ]
  }
}
```


Covid 19 Impact on Global Economy at index 2350


```
{ "head": {
    "vars": [ "countryCode" , "country" , "sameascountry" , "date" , "humanDevelopmentIndex" , "tradediversion" , "gdpPerCapita" , "imdb_url" , "wikidata_url" ]
  } ,
  "results": {
    "bindings": [
      {
        "countryCode": { "type": "literal" , "value": "USA" } ,
        "country": { "type": "literal" , "value": "United States of America" } ,
        "date": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2020-10-19T00:00Z" } ,
        "humanDevelopmentIndex": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.924" } ,
        "tradediversion": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "12.29989991" } ,
        "gdpPerCapita": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "10.90090556" }
      }
    ]
  }
}
```


Covid 19 case Reported in 2020 at index 0


```
{ "head": {
    "vars": [ "country" , "sameascountry" , "continent" , "population" , "totalcases" , "totaldeaths" , "criticalcases" , "casespermillionpeople" , "deathspermillionpeople" , "totaltests"
, "testspermillionpeople" , "whoregion" , "imdb_url" , "wikidata_url" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "United State of America" } ,
        "continent": { "type": "literal" , "value": "North America" } ,
        "population": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "331198130" } ,
        "totalcases": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "5032179" } ,
        "totaldeaths": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "162804" } ,
        "criticalcases": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "18296" } ,
        "casespermillionpeople": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "15194" } ,
        "deathspermillionpeople": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#int" , "value": "492" } ,
        "whoregion": { "type": "literal" , "value": "Americas" }
      }
    ]
  }
}
```


Economics Data for-9 Countries by country Name (United States of America)


```
{ "head": {
    "vars": [ "country" , "year" , "gdppercent" , "indexprice" , "stockindex" , "sameasstockindex" , "inflationrate" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2020-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "-0.03" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "10201.51" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q575782" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.01" }
      } ,
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "7940.36" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q575782" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" }
      } ,
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2018-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.03" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "7425.96" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q575782" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" }
      }
    ]
  }
}
```

Covid 19 Impact on Global Economy and Economics Data for-9 Countries

Query result from combining two RDF files: we added country code to the Economics Data.


```
{ "head": {
    "vars": [ "country" , "countryCode" , "year" , "gdppercent" , "indexprice" , "stockindex" , "sameasstockindex" , "inflationrate" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "Japan" } ,
        "countryCode": { "type": "literal" , "value": "CHN" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "21697.23" } ,
        "stockindex": { "type": "literal" , "value": "Nikkei 225" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q507338" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" }
      } ,
      {
        "country": { "type": "literal" , "value": "Japan" } ,
        "countryCode": { "type": "literal" , "value": "CHN" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "21697.23" } ,
        "stockindex": { "type": "literal" , "value": "Nikkei 225" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q507338" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" }
      } ,
      {
        "country": { "type": "literal" , "value": "Japan" } ,
        "countryCode": { "type": "literal" , "value": "CHN" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "21697.23" } ,
        "stockindex": { "type": "literal" , "value": "Nikkei 225" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q507338" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" }
      } ,
      {
        "country": { "type": "literal" , "value": "Japan" } ,
        "countryCode": { "type": "literal" , "value": "CHN" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "21697.23" } ,
```


Result from filtering two RDF files

Microsoft Teams | SPARQL By Example | RPubs - ISLR: Exerci... | YouTube | 2.1 Random Variabl... | Bayes' Theorem Pro... | Probability, S


```
{ "head": {
    "vars": [ "country" , "year" , "gdppercent" , "indexprice" , "stockindex" , "sameasstockindex" , "inflationrate" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2020-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "-0.03" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "10201.51" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" }
      } ,
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "7940.36" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" }
      } ,
      {
        "country": { "type": "literal" , "value": "United States of America" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2018-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.03" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "7425.96" } ,
        "stockindex": { "type": "literal" , "value": "NASDAQ" }
      }
    ]
  }
}
```


This is the result we got from the java client

CovidClient.java

```
1  package it.univaq.odws.covid.client;
```

Markers | Properties | Servers | Data Source Explorer | Snippets | Console

<terminated> CovidClient [Java Application] C:\Users\Mrs Manneh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\


```
{ "head": {
    "vars": [ "country" , "year" , "gdppercent" , "indexprice" , "stockindex" , "sameasstockindex" , "inflationrate" ]
  } ,
  "results": {
    "bindings": [
      {
        "country": { "type": "literal" , "value": "United Kingdom" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2018-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.01" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "6728.13" } ,
        "stockindex": { "type": "literal" , "value": "FTSE 100" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q466496" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" }
      } ,
      {
        "country": { "type": "literal" , "value": "United Kingdom" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2020-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "-0.1" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "6460.52" } ,
        "stockindex": { "type": "literal" , "value": "FTSE 100" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q466496" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.01" }
      } ,
      {
        "country": { "type": "literal" , "value": "United Kingdom" } ,
        "year": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#dateTime" , "value": "2019-01-01T00:00Z" } ,
        "gdppercent": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.01" } ,
        "indexprice": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "7542.44" } ,
        "stockindex": { "type": "literal" , "value": "FTSE 100" } ,
        "sameasstockindex": { "type": "literal" , "value": "https://www.wikidata.org/wiki/Q466496" } ,
        "inflationrate": { "type": "literal" , "datatype": "http://www.w3.org/2001/XMLSchema#double" , "value": "0.02" }
      }
    ]
  ,
```

5. Conclusion

We successfully implemented the REST web service in maven as well as the java client and we were able to retrieved some relevant information from the dataset.

6. Appendix

Dataset Descriptions, and their licenses.

- Covid19 Impact on Global Economy**:** This dataset has GDP and covid 19 cases information from many countries. It presents estimated economy-wide and sector-specific impacts of the COVID-19 outbreak. It updates the estimates released on March 28, 2020. Some of its indicators includes, Human Development Index (HDI), GDP per Capita (GDPCAP) and Trade Diversion (TD).
    - License**:** https://data.mendeley.com/datasets/b2wvnbnpj9/1

- List_of_countries_by_government_budget_per_capita**:** This list shows the government budget of each country divided by its total population, not adjusted to purchasing power parity. The Chinese, Brazilian, Indian, and United States government budgets used are the figures reported by the International Monetary Fund.
    - License: https://creativecommons.org/publicdomain/zero/1.0/

- Economic Data – 9 Countries: The dataset contains data for 8 countries and one special administrative region (China, France, Germany, Hong Kong, India, Japan, Spain, United Kingdom and United States of America) from 1980 through 2020. It include major macroeconomic factors like inflation, unemployment, GDP, exchange rate (base USD) and per capita income. Apart from that it has the stock prices of the respective country's major stock index which can help in analysing the data set to identify the impact of major macroeconomic variables on the movement of stock index prices.
    - License: https://creativecommons.org/licenses/by-nc-sa/4.0/

- COVID-19 Dataset: A new coronavirus designated 2019-covid was first identified in Wuhan, the capital of China's Hubei province. People developed pneumonia without a clear cause and for which existing vaccines or treatments were not effective. The virus has shown evidence of human-to-human transmission. Transmission rate (rate of infection) appeared to escalate in mid-January 2020. As of 30 January 2020, approximately 8,243 cases. This dataset contains the number of Confirmed, Death and Recovered cases.
    - License: https://github.com/CSSEGISandData/COVID-19

- COVID-19 Public Health Social Measures: A global database of public health and social measures 2021. Public health and social measures

(PHSMs) are measures or actions by individuals, institutions, communities, local and national governments and international bodies to slow or stop the spread of an infectious disease, such as COVID-19. Since the start of the COVID-19 pandemic, a number of organizations have begun tracking implementation of PHSMs around the world, using different data collection methods, database designs and classification schemes.

- License: https://www.who.int/about/policies/publishing/copyright
- License: https://creativecommons.org/licenses/by-nc-sa/3.0/igo/

7. Reference

- https://www.kaggle.com/datasets/shrutisaxena/covid-19-impact-on-global-economy
- https://www.kaggle.com/datasets/mathurinache/list-of-countries-by-government-budget-per-capita
- https://www.kaggle.com/datasets/pratik453609/economic-data-9-countries-19802020
- https://www.kaggle.com/datasets/imdevskp/corona-virus-report
- https://www.kaggle.com/datasets/hijest/covid19-public-health-social-measures

8. Task

S/N	Task	Michael	Lamin	Jai
1	Dataset Gathering	100%	-	-
2	Data cleaning on OpenRefine,	100%	-	-
3	Dataset enrichment, linkage, RDF Extraction	100%	-	-
4	SPARQL Queries	10%	10%	80%
5	Rest Web Service	-	100%	-
6	Java Client	-	100%	-
7	Report	10%	-	90%