Group J Knowledge And Data Engineering Deliverable 3

Liam Bartsch, Oisin Farrell, Yannick Gloster,
Sanil Gupta and Finn Jaksland

November 2022

Contents

1 Approach To Ontology Modelling

1.1 Description of Competency Questions that ontology answers

Our ontology answers the competency questions by providing key insights into trends and patterns from the datasets. Identifying regions/areas where changes in crime amount, house price and accommodation availability is one aspect in which our ontology answers the competency questions.

1.2 Description of datasets selected for application

The three datasets we have selected for our application encompass crime, housing price and accommodation availability. All datasets are licensed under Creative Commons Attribution 4.0.

- Crimes at Garda Stations Level 2010-2016 (Published by the All-Island Research Observatory)
- HSA06 - Average Price of Houses (Published by Department of Housing, Local Government, and Heritage)
- Accommodation (Published by Fáilte Ireland)
- Activities (Published by Fáilte Ireland)
- Attractions (Published by Fáilte Ireland)
- Enrolments of Full-Time Students (Published by Department of Education and Skills)
- All Persons- Live Register (Published by All-Island Research Observatory)

1.3 Assumptions made

The assumptions we made were that there was some influence between the prices of accommodations and availability of accommodations with the amount of crime. We assumed this influence would be present in both long-term accommodations like houses and short-term accommodations like hotels.

1.4 References to sources used/reused e.g. SIOC, FOAF for people

We used GeoSPARQL in our ontology.

1.5 Discussion of your data mapping process

In order to answer the competency questions we need to upload our data to a database so that it can be queried. Before the data is uploaded to the database it must be transformed into rdf. To accomplish this we need to create a mapping of the data and use it to uplift the data. To create the mapping the first step was to look at the data sets. It is important to inspect the data you have and from that understand what you will need. Not all of the information from the data was relevant to this project. For example, in the data for the Garda stations each station has a division attribute but this was not necessary for answering our competency questions so it was omitted from our mapping. Once we had an idea of what we wanted we could begin creating the mappings. To create a mapping you first need to create a mapping file which tells the computer how to take your data and uplift it so that it can be queried in the database. To create the mapping file we had to create predicate object maps for each attribute for the data which correspond to each column of the data set. The result was a list of rdf triples that would be used to uplift the data. When the mapping was done we created a properties file that outlined all the files that would be involved in the uplift. The properties files indicated the path to the data, input mapping file and output file. Using this and the R2RML Java Archive (JAR) file provided we uplifted the data. This took the data and transformed it into rdf according to the mapping that we created. This was done for all three of our data sets. Once we had the rdf files we uploaded them to GraphDB, a graph database, to be queried.

1.6 Explanation of use of inverse, symmetric and transitive properties

- **Inverse:** For our project we set out to determine if house prices are affected by crime. In our ontology we created a class for houses and a class for crime. We assume that the price of a house is affected by crime so we created an object property 'priceIsAffctedBy' which shows the relationship between houses and crime. This gave rise to the inverse property 'affectsPriceOf' which displays the relationship between crime and house as we assume that crime does affect the price of houses.

- **Symmetric:** A common attribute between all the data sets is location. For our project we defined an object property 'adjacentLoation' which is a symmetric property. By definition adjacent meaning next to or adjoining. If an object is joined to another object the relationship of their location goes both ways. If A is joined to B then B is joined to A. In our ontology an example of this is the Garda station that is in Lucan and the Garda station that is located in Ballyfermot. Lucan is adjacent to Ballyfermot in term of locatiion meaning Ballyfermot is adjacent to Lucan.

- **Transitive:** When it comes to location there can be different levels. For example, you could have continent, country, region, state or precise lo-

cation. In our data we have precise location in terms of longitude and latitude and also area and county. This gives rise to the transitive object property 'locatedIn'. For example, Donnybrook is located in the Dublin region which is located in the Leinster region. Thus Donnybrook is located in the Leinster region.

2 Overview of Design

2.1 Description of Application Query Interface

We used Next.js, a React JavaScript framework, to create the Application Query Interface. Next.js allows for the ability to do full stack development without directly interacting with Node or some other type of backend framework. I created an API endpoint which takes a SPARQL query as a parameter and then makes a request to GraphDB. GraphDB returns an object type that isn't standard when dealing with JavaScript applications. I parsed through the object to create a new object that I was more comfortable working with. At the same time, I also parsed floats, integers, and years into their respective JavaScript object as the XML Schema returned them as a String.

The queries were stored as list of objects containing the query, name of the query, as well as a few parameters that helped the UI know which graphs to plot the data on.

In the actual UI, the user can select one of our queries from a drop down menu. The user can see the SPARQL text of the query and can choose to edit the query if they so choose. This was to prove that the UI was actually using SPARQL. Once the query is executed, the user is automatically presented with the best visualization for the query. For queries that return location based information, this information is plotted onto a geographical map. This map was built using Leaflet.js. The user can click on the way markers on the map and find more information about each point. For other types of data, the data is plotted on either line charts, bar charts, or scatter plots. This plots were built using Rechart. I created custom tooltips so that users can see and understand all the data when they hover over each datapoint.

Finally, if the user wants to look at the raw data, the user can choose to look at a table containing the results. This functionality mimics the GraphDB SPARQL interface where a user can run a custom SPARQL query and look at the results in a table. This experience also exists at the `/custom-query` page.

To run the development server to interact with the UI, you must navigate to `/user-interface` in our project files. The `.env.example` file should be renamed to `.env`. In the file, you need to set the name of the GraphDB repository that is set up with the database. Following this, in the same folder, run `npmi` to install the required packages. To launch the development server, you use the command `npmrundev`. This spins up the UI and it can be accessed at `localhost:300/`. To deploy it properly for commercial use, you would be able to follow the Next.js deployment guide.

2.2 Description of Queries

Our queries aim to find a link between the datasets and answer some basic questions regarding the country of Ireland.

We go over more details of each query in Creating Queries:.

The queries cover all the datasets used and help get a better picture of various factors based on location, while they are limited to Ireland, they do help provide a baseline that can be used for other countries. This baseline can reveal interesting trends that can be used for better law enforcement, planning and living.

3 Discussion of challenges faced while ontology modelling or creating queries and mappings

3.1 Mapping:

One of the first challenges we faced was when creating the mappings was transforming the data set into a usable data set. Initially each station had roughly 160 attributes. Each station had attributes for id, division, x and y coordinates and attributes for the types of crime for a given year. The crime attributes were originally divided as follows: crime type 1 2003, crime type 1, 2004, ... crime type 1 2016, crime type 2 2003, crime type 2 2004 and so on. We thought that this was inefficient as you could just have a year attribute and this would reduce the number of columns to 12.

The second major obstacle was removing bad data from the data sets. One of our data sets is a data set for accommodation. When we downloaded it everything looked fine and even after the mapping had been uplifted the rdf output looked good to go. However, when we uploaded it to the database we got an error saying that we had an invalid IRI value. We were completely stumped and couldn't seem to figure out what was causing this error. What made matters more confusing was that two of the three mappings worked just fine and we followed the same methodology for creating them so in theory they should all work. After a lot of debugging we decided that we needed to post on the clinic for help. Albert Navarro Gallinad responded to our message in the clinic and figured that it was due to some special characters such as the Irish fada that were causing the issue. We then had to go through the data and remove any special characters.

The final issue we faced when creating the mapping was again due to invalid characters in the data set. After we had removed special characters we noticed that we were still getting invalid IRI error. We knew that this was something specific as we were able to manually upload some of the triples so we knew that it was just a few that were the issue. AWe went through a process of uploading half at a time, then removing the half and so on until we found the specific

entries that were causing the issue. It turns out that eight entries in the data had a new line character attached to the end and once this was removed all of the mappings worked fine.

3.2 Ontology Modelling:

Modelling an ontology about an area that you are not an expert in is always going to present challenges. You need to make sure that you don't miss something or add something where it shouldn't be. This is why understanding the area you are modelling is so important. This was the case here. The biggest difficulty that we faced as a team was coming up with meaningful classes and relationships for the ontology.

Another factor that made coming up with meaningful classes and actors was the slightly limited data sets we chose. Initially we only had two data sets which left us in a slightly narrow area. In an effort to increase variety and flesh out the project we added a third data set. Using this we were able to get more classes but looking back perhaps we could have added more data sets.

3.3 Creating Queries:

Another challenge we faced was creating the SPARQL Queries since it was difficult to convert our questions into the required queries.

In the end we had our 10 queries as follows:

1. **Where can I go camping where there is low theft and robbery rates?**

   When going camping one question can be, how safe is it, and this query helped us find areas with low crime.

2. **What is the minimum and maximum of housing prices in Ireland?**

   This query helps give an indication of a price range for houses in Ireland.

3. **What trends are there in housing value with relation to the amount of crime in Ireland?**

   This query aims to find a link between the value of houses and the amount of crime in Ireland.

4. **Does more attractions mean higher house prices? Here are the house prices and number of attractions.**

   This query links attractions quantity with house price and could provide an interesting pattern.

5. **How many crimes reported per station (Sum of entire dataset)**

   This query shows the amount of crimes reported for each station.

6. **How many total people are there in total and on welfare each year?**

   This query looks into the total population as well as those who are on welfare.

7. **Does the least expensive region have more B&B's than the most expensive region?**

   This query looks at the quantity of Bed and Breakfast's and compares it from the least to the most expensive region.

8. **Do people go camping where there is high crime?**

   This query looks at whether or not there are lot of people that go camping in high crime areas.

9. **In low crime areas, are new houses sold for more than old houses?**

   This query checks if new houses are more expensive in low crime area, possibly showing a trend.

10. **Look at trends in third level student enrolment, crime, and housing prices, how do they relate? Does a more educated population mean higher prices?**

    This query looks at whether or not more education leads to higher prices and the relation between student enrolment, crime and housing prices.

4 Report how you organised your project

- **WhatsApp:** Once each group member had been assigned to our group, the first thing we did was create a WhatsApp group chat which we decided would be our main source of communication throughout the project. We use this group chat regularly in order to organise our weekly in person meetings, and to raise problems that any of the group members have encountered since the last group meeting. Any minor project issues that are arisen here are usually dealt with swiftly via the messaging platform however if it is a major issue, we sometimes organise an impromptu meeting to discuss it.

- **Google Meet:** Although we do our best to keep all our meetings in-person, it has sometimes occurred that one or more of our members are unable to attend in person due to illnesses or other reasons. In this case, we continue ahead with the meeting in person but allow members to join us remotely if necessary. We use Google Meet in this scenario which has been a perfect way for members to attend our meetings from home.

- **Google Docs:** Our group uses Google Doc in order to organise and monitor all our collaborative work. This includes each of our data sets, all our presentation work, and any documentation what has been required

for our group project. Google Docs has worked very well for us as not only is it an easy and efficient way to organise our work, but it also allows each member to modify the documents simultaneously, and track every revision if needed. Due to the fact that each of us already had a Google Docs account, there was no need for any of us to sign up to anything and therefore it was the perfect platform for us.

- **GitHub:** We use GitHub in order to store all our code related work, including our OWL ontology models, R2RML mappings, SPARQL queries, and code for Application Query Interface, into a group repository. Similarly to Google Docs, GitHub was the obvious choice of code storage for us due to the fact that we all had previous experience with it, and there was no need to sign up for any new platforms.

- **Overleaf:** In order to keep the structuring of our technical report as professional as possible, we made the decision to use Overleaf in order to create the document. A few of our members had very little experience with Overleaf however we found that it was very easy to get used to and was a great way to write and edit documents in a collaborative manner whilst keeping the formatting as structured as possible.

5 Conclusions: Self reflection of group on Strengths / weakness of ontology model, queries & interface

As a group, we started off strong picking our datasets and determining questions. After the presentation, we realized that we needed to adjust our questions and datasets and we adapted very well.

As a group, we struggled to come up with interesting queries and could have done a better job taking advantage of our datasets.

As a group, we didn't work on the project early enough. This lead to a big push to finish the assignment in last couple of weeks. Members of the group as well could have participated in more aspects of the development.

The interface was built to be dynamic and flexible and worked fairly well. I believe that it could be improved to be more dynamic to allow for better custom queries. The UI is very straightforward and easy to use.

Our weaknesses were that we didn't foresee the problems with our mappings until they occurred. This could have been mitigated with more planning.