Education & Economic Status in Ireland

Ontology-Driven Interactive Application

Group A:

Daniel Farrell – 18315021
Sachin Arvind Gade – 20305012
Min Wu – 20302508
Xiaokang Wang - 20303085

1. Introduction

For this project, we chose four different datasets with the goal of building a system that can retrieve knowledge across these datasets that you cannot find out from a single dataset alone. Education and economic status in Ireland is the topic for this project, and we are building this ontology-based system to help discover the correlation between level of education and economic status – both by county and individual.

2. Approach to Ontology Modelling

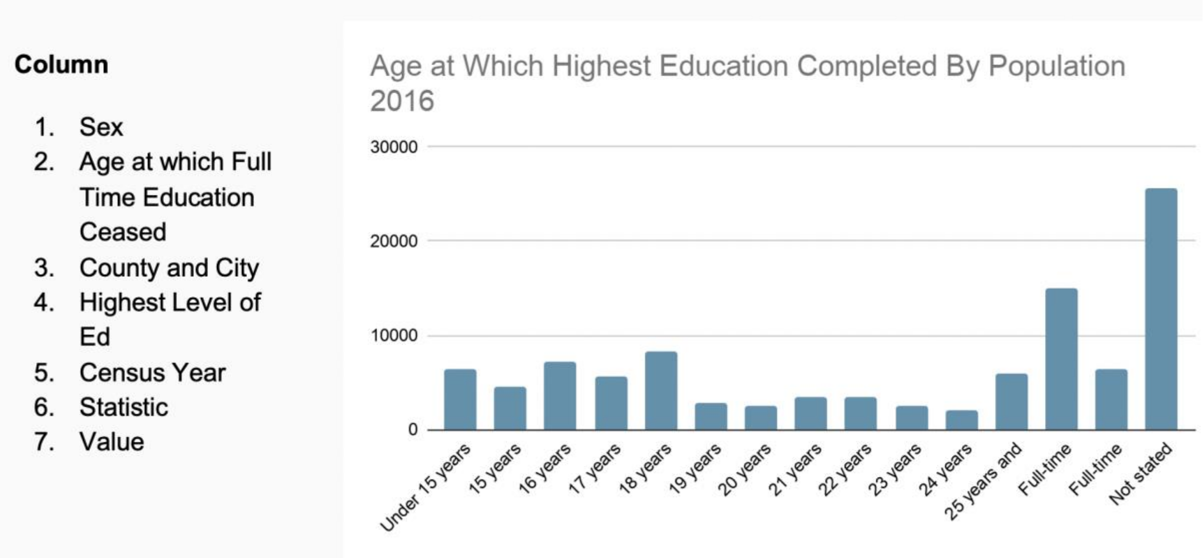
2.1 Datasets

Four different datasets were chosen for this project, they were as follows:

2.1.1 EZ073

Link: https://data.gov.ie/dataset/er-by-highest-level-of-education-completed-county-and-city-censusyear-sex-and-age-at-which-full-time

Description: This dataset contained information about the population of a county, specifying the highest level of education held. i.e. portion of population that hold a bachelor's degree as their highest level of education.



Column

1. Sex
2. Age at which Full Time Education Ceased
3. County and City
4. Highest Level of Ed
5. Census Year
6. Statistic
7. Value

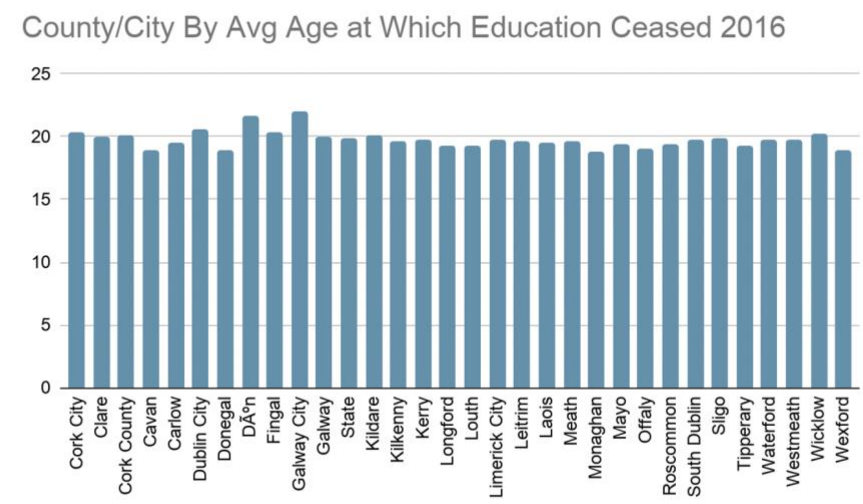
2.1.2 EA033

Link: https://data.gov.ie/dataset/ceased-of-population-aged-15-years-and-over-2011-to-2016-by-county-and-city-censusyear-and-statistic

Description: This dataset contains information about the average age at which a counties population ends their education.

Column

1. County and City
2. Census Year
3. Statistic
4. Value



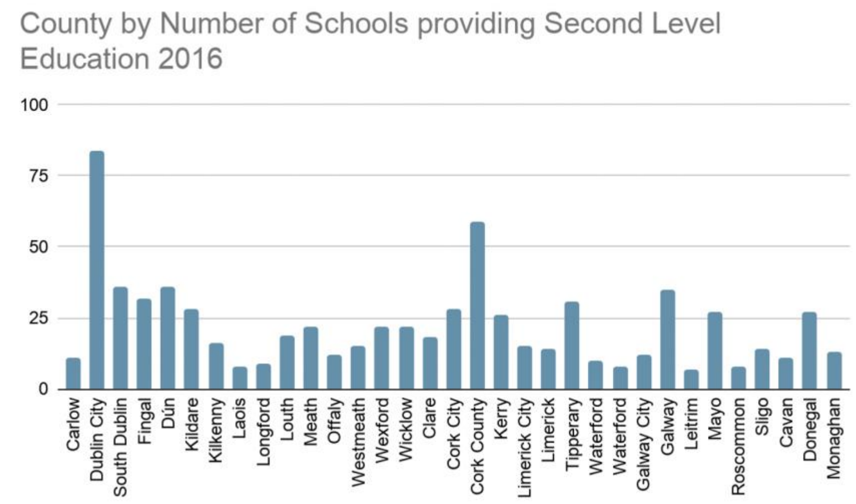
2.1.3 EDA69

Link: https://data.gov.ie/dataset/second-level-schools-and-pupils-by-year-county-type-of-school-and-statistic

Description: This dataset contains information about the types of schools in each county of Ireland, and the number of students in each of those school types.

Column

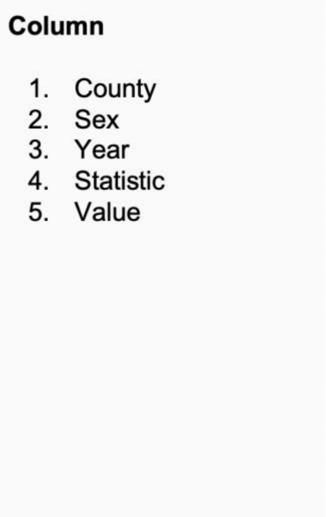
1. County
2. Type of School
3. Year
4. Statistic
5. Value



2.1.4 NEA08

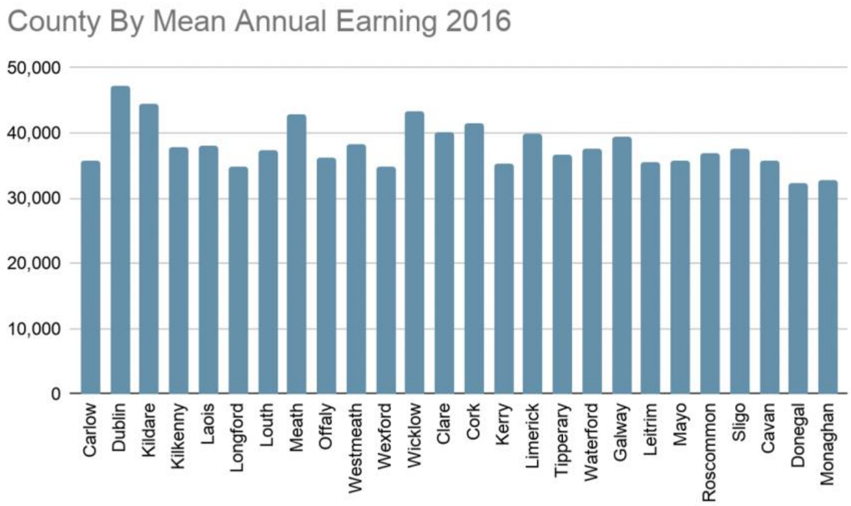
Link: https://data.gov.ie/dataset/nea08-mean-and-median-annual-earnings-by-county-sex-year-and-statistic

Description: This dataset contains information about the mean and median earnings in each county in Ireland.



Column

1. County
2. Sex
3. Year
4. Statistic
5. Value



2.2 Competency Questions

After selecting our datasets, we came up with 10 competency questions we would like our system to be able to answer. Most of these questions require data from two datasets.

Q1. Out of the county with the highest number of bachelor degrees, what is the mean earning of its population ? (Merges dataset EZ073 & NEA08)

Q2. What county has the lowest number of third level degrees and what age on average does their education cease ? (Merges dataset EZ073 & EA033)

Q3. What county has the lowest earnings and what age on average does their education cease? (Merges dataset NEA08 & EA033)

Q4. For the county with the lowest number of secondary schools, how much does the population earn on average ? (Merges dataset EDA69 & NEA08)

Q5. For the county with the lowest number of bachelor’s degrees, how many schools there are providing secondary level education? (Merges dataset EZ073 & EDA69)

Q6. Does the county with the lowest average age for ceasing education earn the least? (Merges dataset EA033 & NEA08)

Q7. Does the county with the highest average age for ceasing education earn the most? (Merges dataset EA033 & NEA08)

Q8. For the county with the most postgraduate degree holders, what was the mean earnings in 2018? (Merges dataset EZ073 & NEA08)

Q9. For the county with highest percentage of PhD holders, what is the mean earning, average age for ending education, and the number of secondary school
(Merges all datasets)

Q10. In 2016, the county with the lowest mean earning , what age on average do they finish their education ? (Merges dataset EA033 & NEA08)

2.3 Sources Used

When modelling our ontology in protégé, we used this guide by the University of Manchester: http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_1.pdf

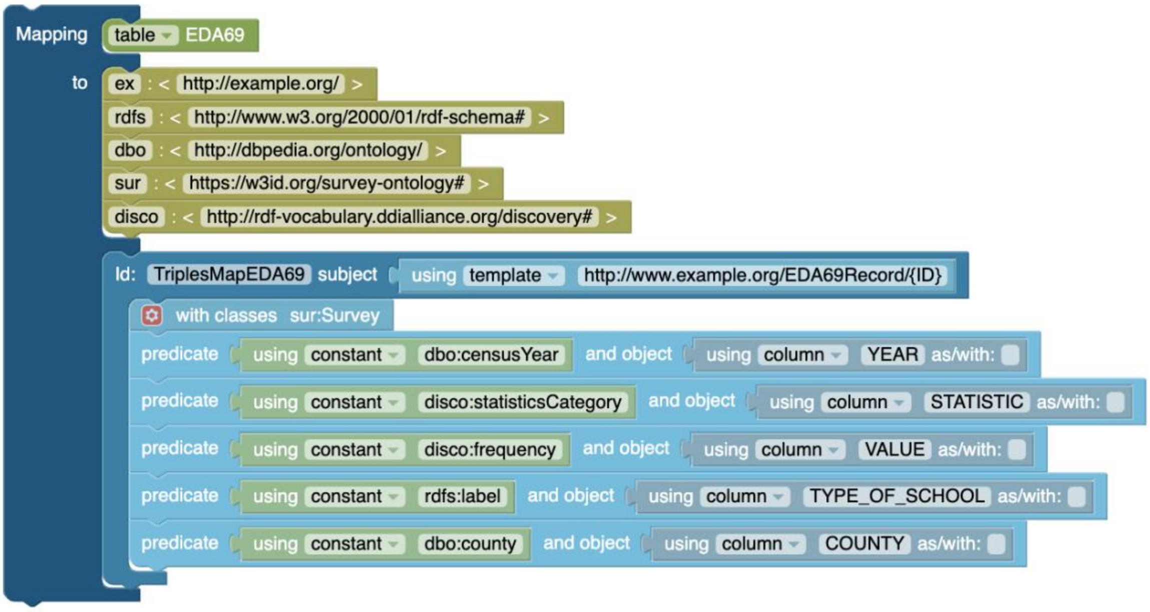
When using Jena, we used the official documentation to better understand the functionalities: https://jena.apache.org/documentation/

In order to find prefixes when uplifting our data we used https://prefix.cc/ and also to find particular vocabulary we used https://lov.linkeddata.es/dataset/lov/

Prefixes used:
http://rdf-vocabulary.ddialliance.org/discovery
http://dbpedia.org/ontology/
https://w3id.org/survey-ontology
http://www.w3.org/2000/01/rdf-schema

2.4 Data Mapping

JUMA was used for the majority of the mapping and uplifting stages of this project. When searching for relevant vocabulary we used https://lov.linkeddata.es/dataset/lov/ and if we needed to find the prefix we used https://prefix.cc/. We tried to find the most relevant vocabulary for each column of our datasets, for example dbo:censusYear for the year each census was recorded. Also, when JUMA was down, we had to write the mappings manually and used the R2RML engine to uplift our data from the original CSV to RDF. We also reused the vocabulary as much as we could for all of the datasets so that they are standardized. Once we had the outputted turtle file from either JUMA or R2RML, it was loaded into our triplestore on GraphDB.



```
Mapping  table EDA69
to  ex : < http://example.org/ >
    rdfs : < http://www.w3.org/2000/01/rdf-schema# >
    dbo : < http://dbpedia.org/ontology/ >
    sur : < https://w3id.org/survey-ontology# >
    disco : < http://rdf-vocabulary.ddialliance.org/discovery# >
Id: TriplesMapEDA69 subject using template http://www.example.org/EDA69Record/{ID}
  with classes sur:Survey
  predicate using constant dbo:censusYear and object using column YEAR as/with:
  predicate using constant disco:statisticsCategory and object using column STATISTIC as/with:
  predicate using constant disco:frequency and object using column VALUE as/with:
  predicate using constant rdfs:label and object using column TYPE_OF_SCHOOL as/with:
  predicate using constant dbo:county and object using column COUNTY as/with:
```

Fig 2.5: JUMA Example

2.5 Assumptions Made

Before modelling the ontology, the following assumptions were made:

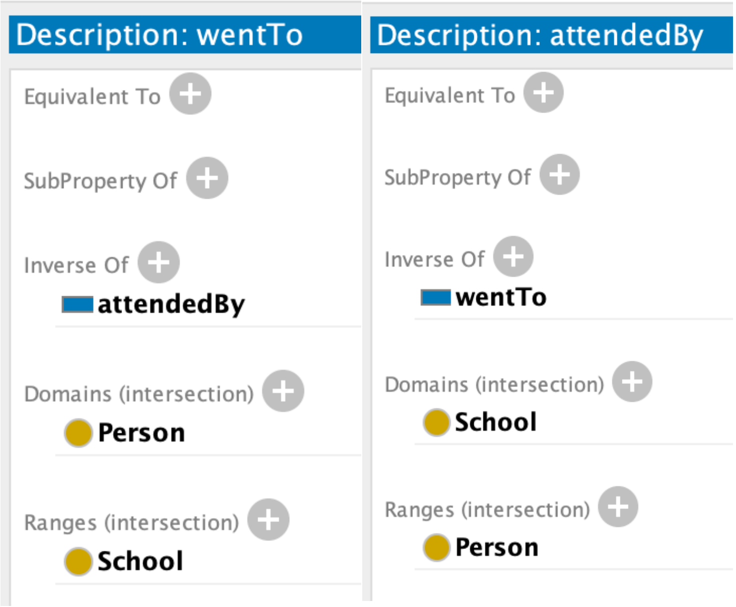
- A person can have more than one degree
- Education level can be of many different types
- Schools can be of many different types
- Counties can contain more than one school
- Schools can have different numbers of pupils

2.6 Properties

When modelling our ontology, we used a few different types of object properties.

Inverse Property:

We created inverses for all of our object properties.
An example would be
Person wentTo School
&
School attendedBy Person:



Description: wentTo

Equivalent To

SubProperty Of

Inverse Of
attendedBy

Domains (intersection)
Person

Ranges (intersection)
School

Description: attendedBy

Equivalent To

SubProperty Of

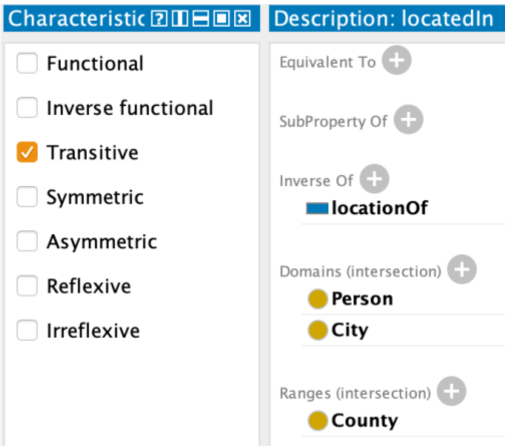
Inverse Of
wentTo

Domains (intersection)
School

Ranges (intersection)
Person

Transitive Property:

One transitive property we had was: Person locatedIn City locatedIn County
From this we can deduce that Person is also locatedIn County.



Characteristic

- ☐ Functional
- ☐ Inverse functional
- ☑ Transitive
- ☐ Symmetric
- ☐ Asymmetric
- ☐ Reflexive
- ☐ Irreflexive

Description: locatedIn

Equivalent To

SubProperty Of

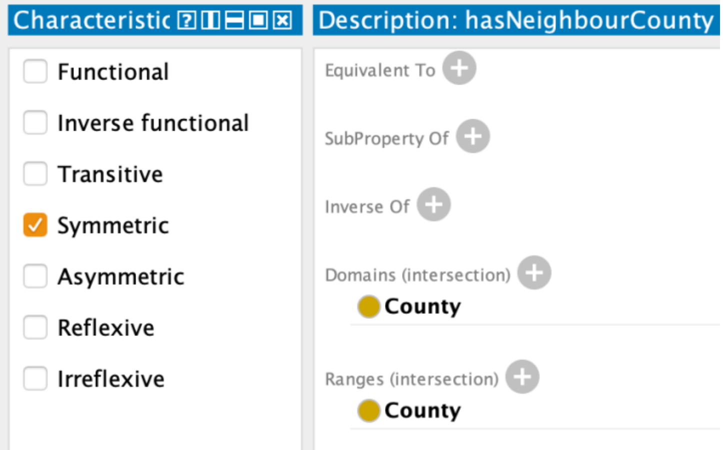
Inverse Of
locationOf

Domains (intersection)
● Person
● City

Ranges (intersection)
● County

Symmetric Property:

One symmetric property we had was: County hasNeighbourCounty County
If county1 is a neighbour of county2, then county2 is a neighbour of county1.



Characteristic

- ☐ Functional
- ☐ Inverse functional
- ☐ Transitive
- ☑ Symmetric
- ☐ Asymmetric
- ☐ Reflexive
- ☐ Irreflexive

Description: hasNeighbourCounty

Equivalent To

SubProperty Of

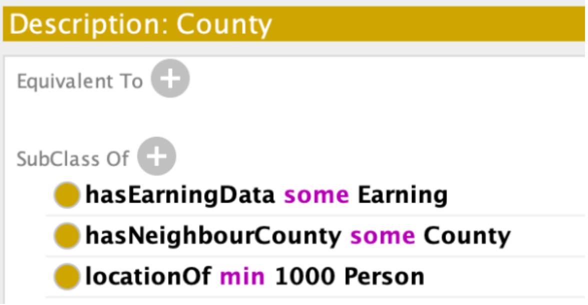
Inverse Of

Domains (intersection)
● County

Ranges (intersection)
● County

Cardinalities:

We also added many cardinalities to our ontology, an example would be a County has to have some earning data, has to have a neighbouring county, and must have a minimum of 1000 people living there.



Description: County

Equivalent To

SubClass Of
● hasEarningData some Earning
● hasNeighbourCounty some County
● locationOf min 1000 Person

3. Overview of Design

3.1 Description of Application Query Interface

For the front-end of the application, we used React. The main reason for this is that it makes the process a lot more manageable and faster to build. We also used Sass, MaterialUI and bootstrap for creating a better-looking interface. In this UI, there are tabs for every question. On each tab, you will find the following things:

1. Question
2. Datasets used for the question
3. Execute Query button to execute SPARQL query
4. Reset Result button to reset the result gathered from the executed query
5. Textarea containing SPARQL query for that question
6. Result Table

For the backend of the application, we used the FUSEKI JENA server. We uploaded our uplifted data on the server. To retrieve data from the server we used AJAX call from the frontend and called the SPARQL service of Fuseki Jena. We passed the query to the service in ajax call and in response, we got the result.

3.2 Description of Queries & Mappings

Queries and mappings have been presented and described in our Widoco document*

4. Challenges Faced

Whilst carrying out this project, we encountered a few challenges along the way that we had to overcome. Some examples of these challenges were:

- Deciding on the right vocabulary to use in the mapping stage
- Finding these vocabulary and their prefixes proved to be quite tough
- Deciding which parts of our dataset should be classes or subclasses
- Creating properties to link our four datasets together
- Creating the right SPARQL queries to answer our competency questions was quite difficult and was an iterative process
- Introducing a transitive and symmetric property to our ontology

5. Conclusion

On completion of the project, we all now have a much better understanding of the process that goes into creating knowledge graphs and machine readable data. We gained a lot of knowledge from this project, and if we were to do a similar project again, the entire process would seem a lot more approachable and the challenges we highlighted in the previous section would not seem like major hurdles.

We have also listed below some strengths and weaknesses of our ontology model, SPARQL queries, and user interface.

Strengths:

- All classes in the ontology model are intricately linked together using different data properties such as symmetric, transitive, and inverse properties
- Many individuals and data types have been added to our ontology also
- We wrote complex queries that could use data from all four datasets
- The UI perfectly executes and displays all 10 of our SPARQL queries without issue

Weaknesses:

- For some entities in our datasets, we could not find very suitable vocabulary. For example “Type of school”
- The UI does not support the creation of custom queries
- Some queries may be overly complex