Candidates 140 and 162

About our project

Our project is based on the open data from the national tourist roads in Norway (https://www.nasjonaleturistveger.no/). The data set provided by this website is an xml file with the names of the 18 tourist roads in Norway, the latitude and longitude of the start and end of each road as well as possible activities along the roads. In addition, we used an rdf dump data set from GeoNames, which was a text file that we were able to lift and include in our triples. We also extracted data from a reverse geocoder and were able to get addresses from a latitude and longitude.

The national tourist roads website gives information on those roads, indicates in which area of Norway they are, and even suggests activities to do along the road. But those roads are not geographical data, most of them have their own names that are not exactly the same than the name of the city or region in which they are, which means that anyone trying to find one of those places will have to look on their own and might struggle finding the right information. By collecting the semantic data from the website, we were able to lift the latitude and longitude of the start and end of each road, as well as the activities mentioned. We made triples using RDF vocabulary and RDF classes. We used then a reverse geocoder (geopy) that would find some address information from the latitude and longitude. We integrated these addresses into our triples where possible, in this way roads and activities had now a geographical location.

We also extracted a dump dataset from GeoNames and were able to integrate it in our code such that we could also look for any geographical place registered in GeoNames that would be in proximity to a tourist road. This means that, by using a SPARQL query program, we can search for a specific tourist road start or end based on address information. But we can also look for the address of a specific activity mentioned by the website for example. The places and activities given from the website have now a geographical location associated with them. Moreover, the Norway GeoNames dataset allows us to look for even for locations or any geographical place or service (like an airport for example) that would be in the vicinity of the end of a tourist road for example. The query results also in GeoNames direct links to the GeoNames open database. So, by using semantic data in this context, we were able to combine data sets that were not working together before and add new information to the original data set of the tourist roads giving them now a geographical address as well as links to related places in GeoNames. The names of the tourist roads and activities were not locations, but they now have an address and geographical location.

Technologies, tools and vocabularies.

For this project we employed rdflib, geopy and sparqlwrapper mainly. These tools allowed us to lift the datasets and some python scripts were used for converting data formats and producing an RDF graph. We also used two geographical API's "Nominatim" and "GeoNames" both in the geopy package. For vocabularies we made some of it ourselves using http://example.org/Tourism/ as a namespace, however, we also used some "geonames", "wgs84_pos" and of course "rdflib" vocabulary. Lastly we used "sparqlwrapper" and "blazegraph" in order to query our datasets.

Difficulties and furtherment

We had however some issues with GeoNames, as it was not easy finding and using the geonames data set without having to lift the whole GeoNames dataset, this is in the end how we did it though. The open data has also some issues when it comes to retrieving geographical information from a latitude and longitude because the reverse geocoder offered by GeoNames requires a username, which can create issues when someone else would be using the code. So, we also used a different geocoder from geopy. Our idea was to retrieve the GeoNames ID from the postcode associated by the reverse geocoder for each place registered in our dataset, in order to be able to create a GeoNames link for each place. We considered only using a geocoder or DBpedia, but GeoNames was the best data set with exact locations to use when we only had latitude and longitude. This why we decided and managed to combine the geopy reverse geocoder and the transformed data set from GeoNames. By also adding the Norwegian transformed dataset from GeoNames, we were able to link those places and addresses to the geographical surroundings of those places in GeoNames, and in this way combining different data sets.

If we were to further develop this project we would love to add some geospatial querying such as "GeoSparql". This would allow us to use coordinates in a much less verbose way than our current filterings, and instead create things such as bounding boxes, circles with a given radius and rectangles. "GeoSparql" even has functionality which is "spatial:Nearby" in which you can simply give coordinates and specify the distance. This could then develop into a fully usable interface to search for things such as hotels, airports etc for anyone planning trips with the tourist roads or attractions.

Contributions:
As we were only two in the group, we worked pretty closely together and developed the project together, there was no strict individual contribution to the project.