IIC3432 — Advanced topics in databases

MongoDB

1. Mongo cheat sheet

Admin part:

- `mongod` starts the database. You need to have this running to use the console.
- `mongo` starts the mongo shell. We use this to issue commands to the database.
- `show dbs` shows the existing databases.
- `use MYDB` switch to the database `MYDB` (if it doesn't exist create it).
- `db.stats()` show the statistics of the database I am currently using.
- `db` show which database I am currently using.
- `db.dropDatabase()` drop the current database.

Inserting data:

- `db.MYCOLLECTION.insert(doc)` insert the document `doc` into the collection `MYCOLLECTION` (if the key is specified and already exists it fails).
- `db.MYCOL.save(doc)` is some document has the same key as `doc` (if specified) overwrites it.

Basic queries:

- `db.MYCOL.find()` give all the docs in MYCOL
- `db.MYCOL.find{"name": "Joe"}` basic selection
- `db.MYCOL.find({.age":23, .address": {"street":...})`

The rest (you will need to use it a lot):

`https://docs.mongodb.com/manual/`

2. Wikidata: entities

We have already seen WikiData. It serves its data as JSONs as well. More info here:

https://www.mediawiki.org/wiki/Wikibase/DataModel/JSON

High level overview:

```
{
  "id": "Q298",
  "type": "item",
  "labels": {},
  "descriptions": {},
  "aliases": {}
  "claims": {},
  "sitelinks": {},
}
```

This is a typical entity in WikiData. The field `id` is the unique ID of each entity. For instance `Q298` corresponds to the entity Chile and `Q60` to New York. To view the original data on WikiData about Chile (or any entity really) go to:

https://www.wikidata.org/wiki/Q298

To see how is this served as JSON see:

https://www.wikidata.org/wiki/Special:EntityData/Q298.json

In this class we will work with the data you downloaded that contains a bunch (c.a. half a million) WikiData entities. If you forgot to download the database go to:

`https://drive.google.com/file/d/0B32bx9rxQY_CVWJrVElVcFZnbTA/view`

First, we need to import this database into mongoDB. The command to do this (from system shell) is:

```
mongoimport --db wiki --collection docs
                      --jsonArray c:\Users\domagoj\Desktop\wiki_500000.json
```

We can now start working with the database. First, note that all the entities have labels. More or less something like this:

```
"labels": {
   .
   .
   .
   "en": {
     "language": "en",
     "value": "Chile"
   }
   .
   .
   .
 }
```

Therefore, to get the label of an entity QXYZ we need to look for a document where `ID` equals QXYZ, and then obtain the value of `labels` then of `en` and finally `value`. This is of course a nested JSON.

First task: Write a query that, given an entity Q233836 finds its label in English. You might have to do some navigation. Look in the online explorer how the document is structured. Try this for Chile in the database you loaded. What happens? Well, maybe the data is not there. Go to the JSON document for Chile in WikiData, download/copy it, and insert it into the database. Careful here about how the document itself is wrapped. You only need the value inside entities.Q298. Try the query again.

Once we have the label we can also add a description. Format for descriptions is similar as in labels:

```
"descriptions": {
   .
   .
   "en": {
     "language": "en",
     "value": "country in South America"
   }
   .
   .
 }
```

Second task: Write a query that, given an entity, returns it label and description; both of those in English.

3. Wikidata: connections

Go to the WikiData web page of Chile and look for highest point. You will find a box that has this, plus the value Ojos del Salado as the value.
What's going on here? Well, WikiData saves properties of entities. In this case, the property is highest point, and it creates a connection/relation between the entity Chile and the entity Ojos del Salado. When you hover over the link for highest point you will notice that the ID of this thing is `P610`. Similarly, for Ojos del Salado we get `Q233836`. Therefore this is similar to a connection between two elements in a graph database or an edge labelled graph. The structure of this connection is seen in the document corresponding to Chile as:


```
"claims":{
     "P610":[
         {
            "id":"Q298$b184fe8e-4074-1ec6-950f-ac4303f2ccba",
            "mainsnak":{
               "snaktype":"value",
               "property":"P610",
               "datatype":"wikibase-item",
               "datavalue":{
                  "value":{
                     "entity-type":"item",
                     "numeric-id":233836
                  },
                  "type":"wikibase-entityid"
               }
            },
            "type":"statement",
            "rank":"normal"
         }
      ],
      .
      .
      .
 }
```


The important value here is `numeric-id`. If we add `Q` at the beginning of this number we will get `Q233836`; that is, the id of Ojos del Salado.

Third task: Write a query that for some fixed entity (e.g. Chile) finds its label in English, and, for each of its claims, returns the label of the property, and the ID of the entity that this property connects the base entity with. It is possible that you will not find this in your database. In this case display an appropriate message. Since none of these exist find a few of them that have to do with Chile and insert them.