Hands-on with D2RQ

Author: Mingjie Ye

0. Environment

- System: MacOS
- MySQL 8.0.15
- D2RQ 0.8.1
- Java 1.8

1. Introduction

The topic I chose in the Seminar 1 is about RDB2RDF and query a non-RDF database using SPARQL. My hands-on work is based on D2RQ which is a platform for accessing relational database as RDF ways. In this work, I create a relational database in MySQL and use SQL to SELECT some data. Then we run D2R server from the command line to execute the D2RQ mapping. We could access querying in HTML browsers or command line. The results are clearly showed in the page.

2. Installation

D2RQ is available in its website http://d2rq.org/#compatibility

D2RQ supports many relational databases:

- Oracle
- MySQL
- PostgreSQL
- SQL Server
- HSQLDB
- Interbase/Firebird

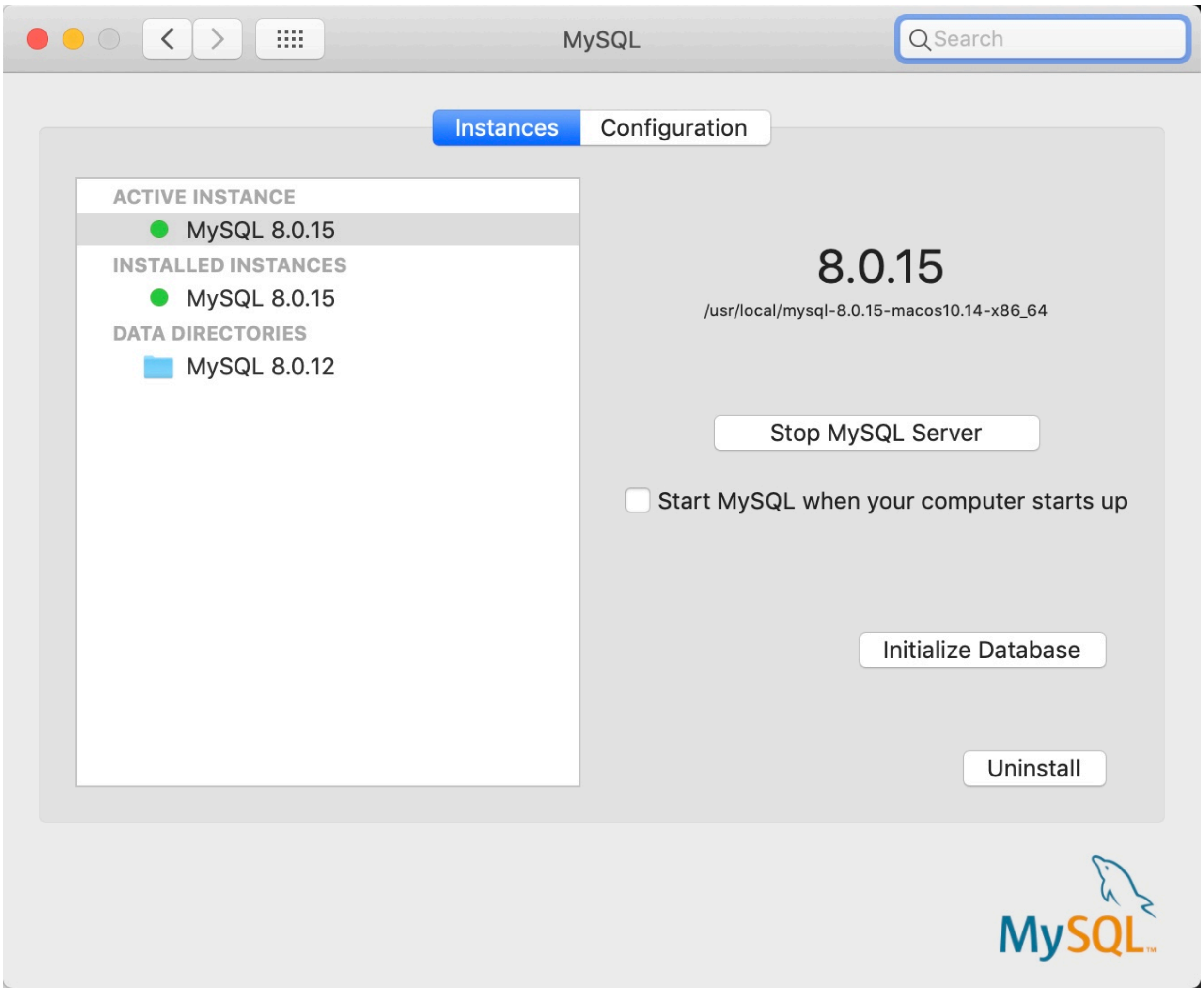
Here, I chose MySQL as the relational database. The installation of MySQL is easy with the document of MySQL tutorial https://dev.mysql.com/downloads/installer/.

In order to access database, we need JDBC driver which is a software component enabling a Java application to interact with a database. We could download MySQL JDBC driver named Connector/J from the official website https://dev.mysql.com/downloads/connector/j/. Then place the **mysql-connector-java-8.0.15.jar** file into '**/d2r-server-0.7/lib/**'

3. Running

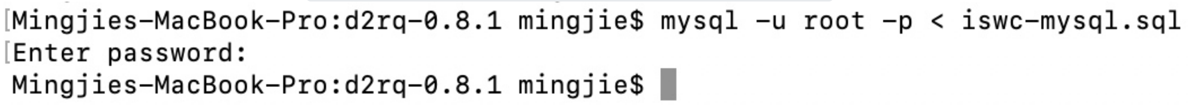
3.1 Run MySQL

First, we open mysql in Preference.



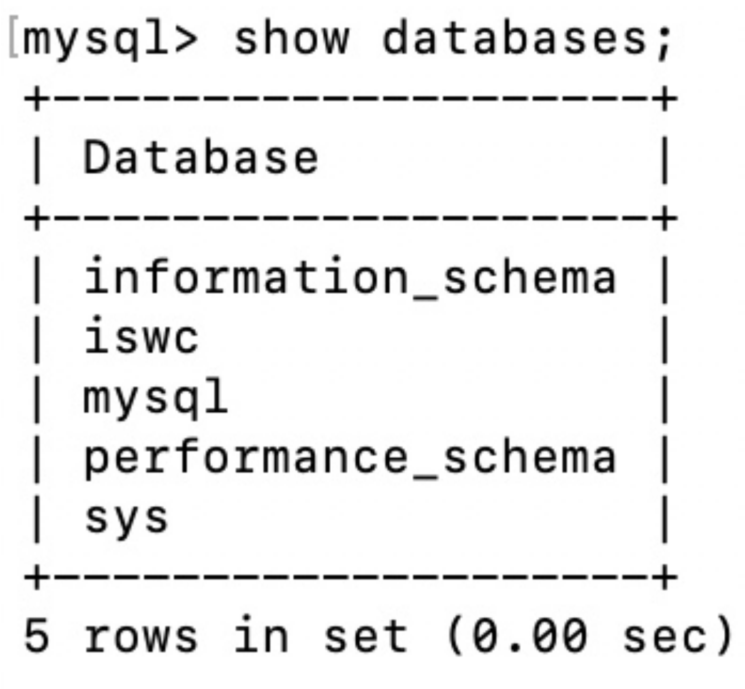
Here, I use root as my username and 12345678 as my passwor and use the database from the ISWC 2002 conference available here as an exammple. I add some lines in this sql file in order to create a database "iswc" in mysql first and use iswc. So the sql file here is slightly different.

Then in command line: `mysql -u root -p < iswc-mysql.sql`

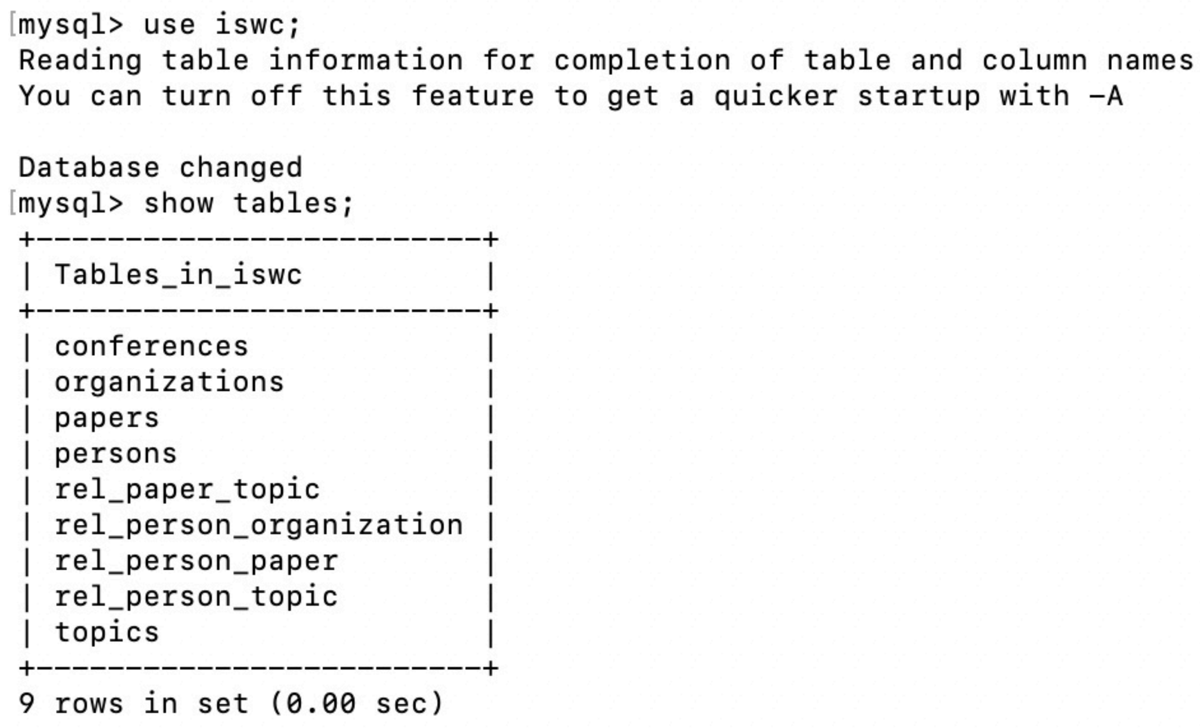


```
[Mingjies-MacBook-Pro:d2rq-0.8.1 mingjie$ mysql -u root -p < iswc-mysql.sql
[Enter password:
 Mingjies-MacBook-Pro:d2rq-0.8.1 mingjie$ 
```

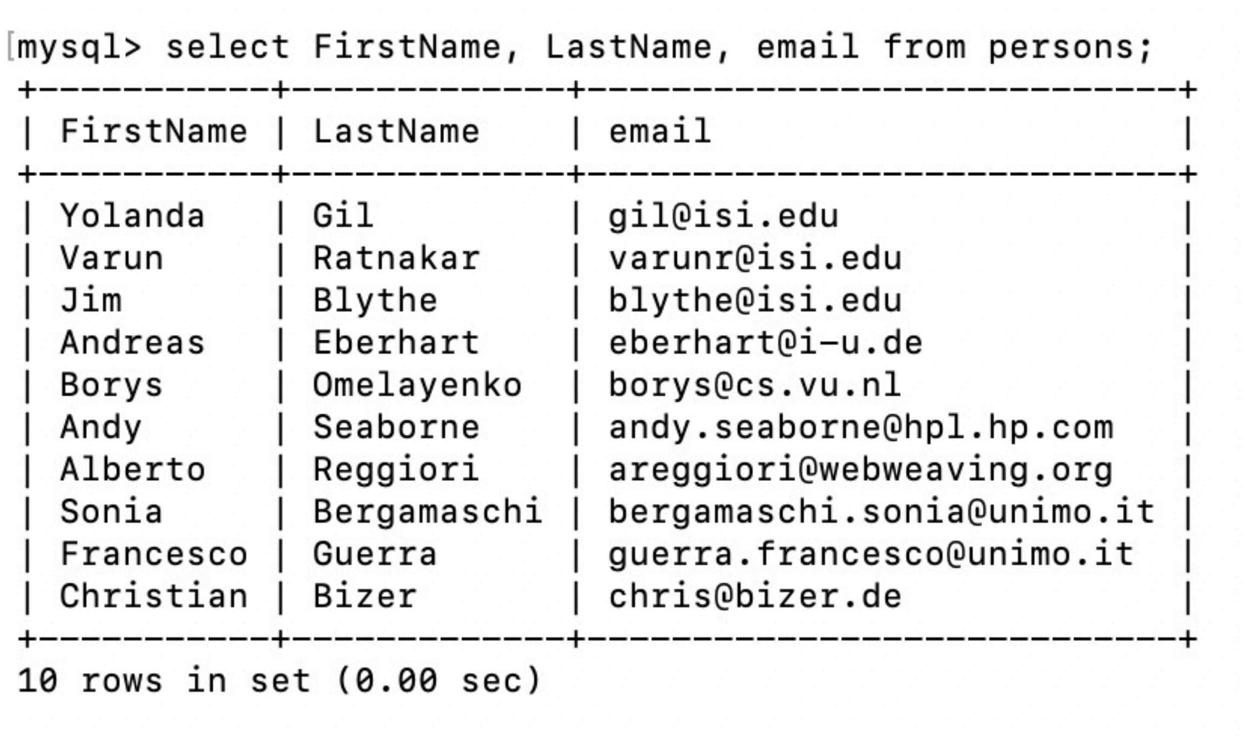
See our databases, tables and execute a simple SELECT in MySQL.



```
[mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| iswc               |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```



```
[mysql> use iswc;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
[mysql> show tables;
+-------------------------+
| Tables_in_iswc          |
+-------------------------+
| conferences             |
| organizations           |
| papers                  |
| persons                 |
| rel_paper_topic         |
| rel_person_organization |
| rel_person_paper        |
| rel_person_topic        |
| topics                  |
+-------------------------+
9 rows in set (0.00 sec)
```



```
mysql> select FirstName, LastName, email from persons;
+-----------+-------------+----------------------------+
| FirstName | LastName    | email                      |
+-----------+-------------+----------------------------+
| Yolanda   | Gil         | gil@isi.edu                |
| Varun     | Ratnakar    | varunr@isi.edu             |
| Jim       | Blythe      | blythe@isi.edu             |
| Andreas   | Eberhart    | eberhart@i-u.de            |
| Borys     | Omelayenko  | borys@cs.vu.nl             |
| Andy      | Seaborne    | andy.seaborne@hpl.hp.com   |
| Alberto   | Reggiori    | areggiori@webweaving.org   |
| Sonia     | Bergamaschi | bergamaschi.sonia@unimo.it |
| Francesco | Guerra      | guerra.francesco@unimo.it  |
| Christian | Bizer       | chris@bizer.de             |
+-----------+-------------+----------------------------+
10 rows in set (0.00 sec)
```

The result is showed in command line.

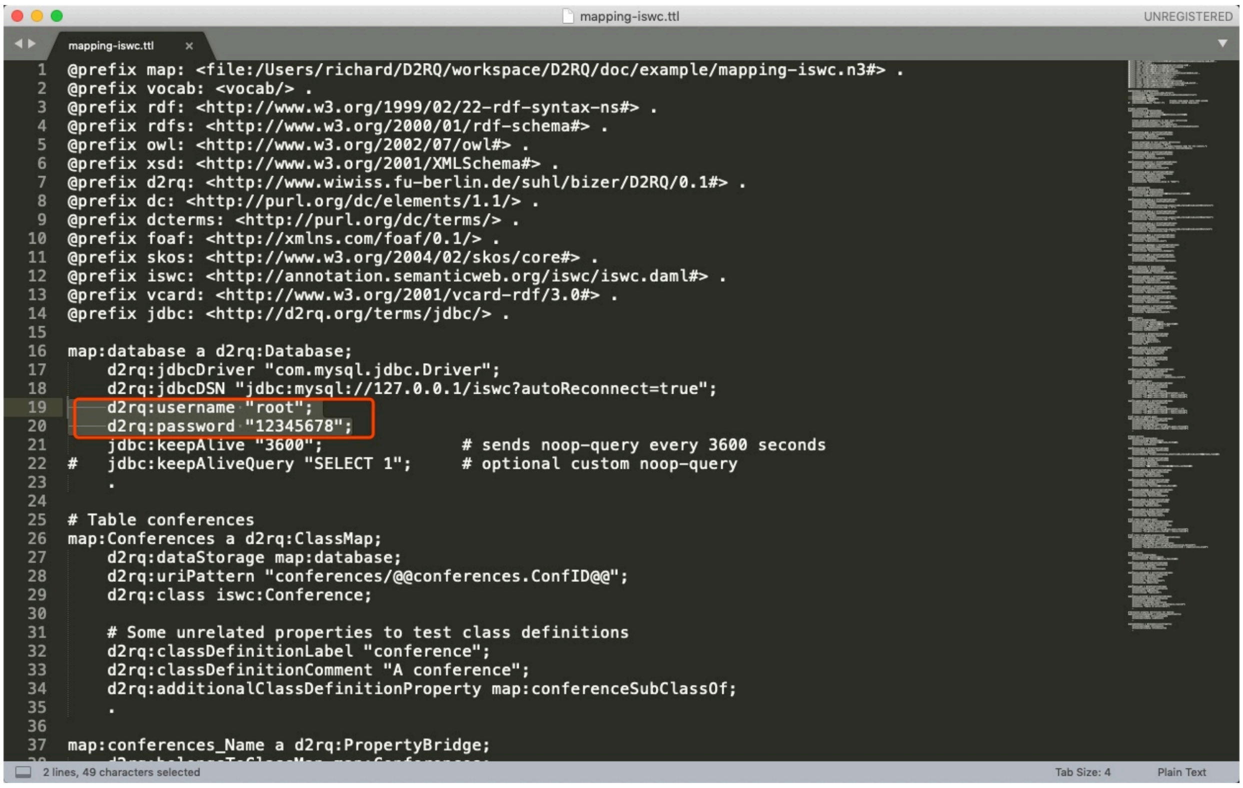
3.2 D2RQ

Now we are going to mapping this relational database into RDF way and query by SPARQL language.

First, we have to generate a mapping file for our database schema using the generate-mapping tool. It's like

```
generate-mapping -o mapping.ttl -d driver.class.name
    -u db-user -p db-password jdbc:url:...
```

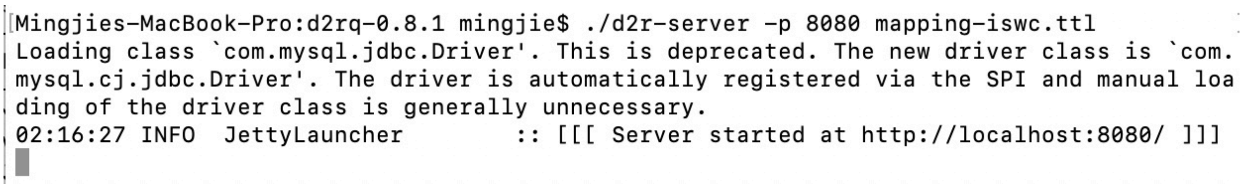
Here, we use this mapping file. Pay attention to the username and password in the file. You need change them to yours before mapping.



```
@prefix map: <file:/Users/richard/D2RQ/workspace/D2RQ/doc/example/mapping-iswc.n3#> .
@prefix vocab: <vocab/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix iswc: <http://annotation.semanticweb.org/iswc/iswc.daml#> .
@prefix vcard: <http://www.w3.org/2001/vcard-rdf/3.0#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .

map:database a d2rq:Database;
    d2rq:jdbcDriver "com.mysql.jdbc.Driver";
    d2rq:jdbcDSN "jdbc:mysql://127.0.0.1/iswc?autoReconnect=true";
    d2rq:username "root";
    d2rq:password "12345678";
    jdbc:keepAlive "3600";          # sends noop-query every 3600 seconds
#   jdbc:keepAliveQuery "SELECT 1";     # optional custom noop-query
    .

# Table conferences
map:Conferences a d2rq:ClassMap;
    d2rq:dataStorage map:database;
    d2rq:uriPattern "conferences/@@conferences.ConfID@@";
    d2rq:class iswc:Conference;

    # Some unrelated properties to test class definitions
    d2rq:classDefinitionLabel "conference";
    d2rq:classDefinitionComment "A conference";
    d2rq:additionalClassDefinitionProperty map:conferenceSubClassOf;
    .

map:conferences_Name a d2rq:PropertyBridge;
```

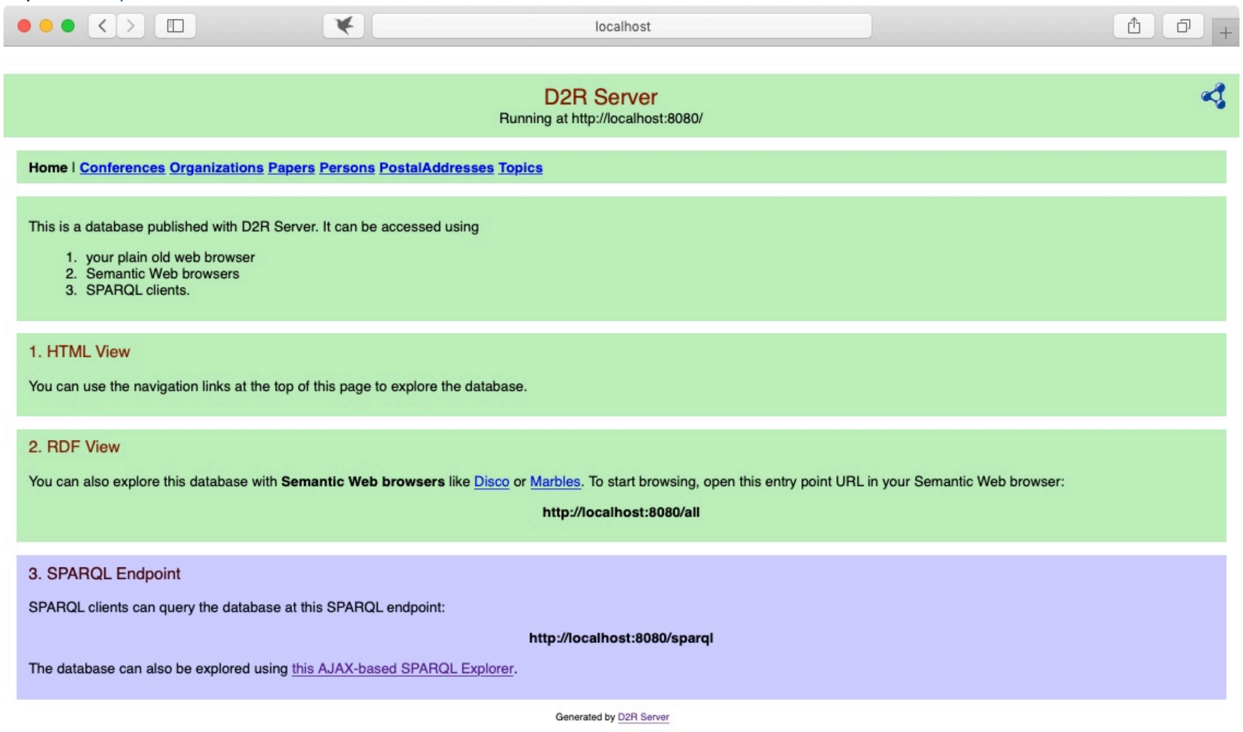
Then, run it by using

```
./d2r-server -p 8080 mapping-iswc.ttl
```



```
Mingjies-MacBook-Pro:d2rq-0.8.1 mingjie$ ./d2r-server -p 8080 mapping-iswc.ttl
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.
mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loa
ding of the driver class is generally unnecessary.
02:16:27 INFO  JettyLauncher        :: [[[ Server started at http://localhost:8080/ ]]]
```

Open http://localhost:8080/ in browsers.



D2R Server

Running at http://localhost:8080/

Home | Conferences Organizations Papers Persons PostalAddresses Topics

This is a database published with D2R Server. It can be accessed using

1. your plain old web browser
2. Semantic Web browsers
3. SPARQL clients.

1. HTML View

You can use the navigation links at the top of this page to explore the database.

2. RDF View

You can also explore this database with **Semantic Web browsers** like Disco or Marbles. To start browsing, open this entry point URL in your Semantic Web browser:

http://localhost:8080/all

3. SPARQL Endpoint

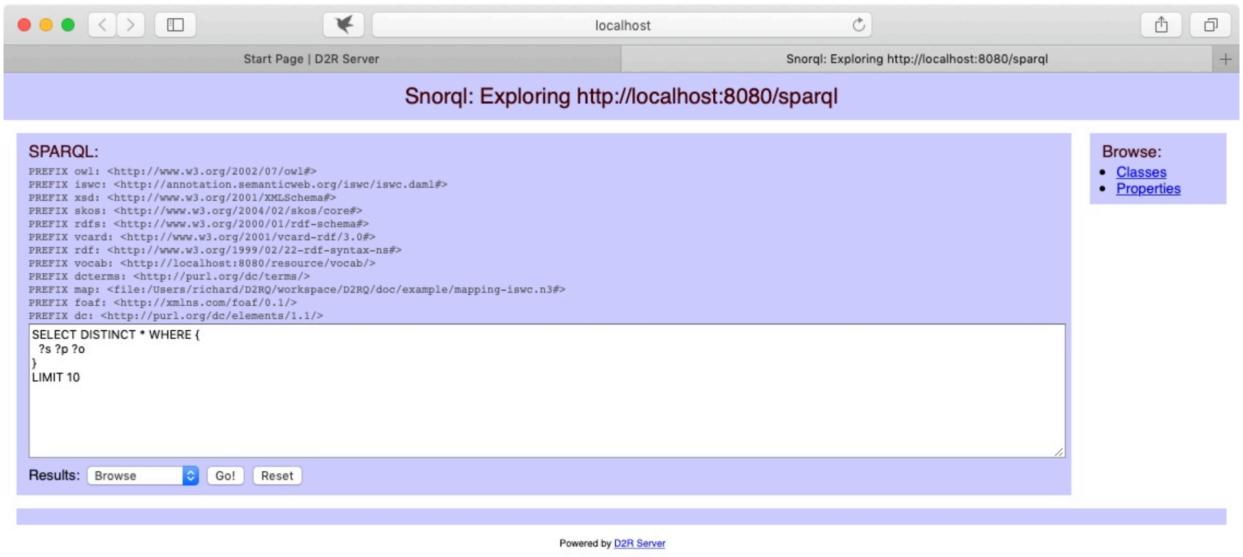
SPARQL clients can query the database at this SPARQL endpoint:

http://localhost:8080/sparql

The database can also be explored using this AJAX-based SPARQL Explorer.

Generated by D2R Server

Turn to http://localhost:8080/snorql/ to use SPARQL.



Snorql: Exploring http://localhost:8080/sparql

SPARQL:

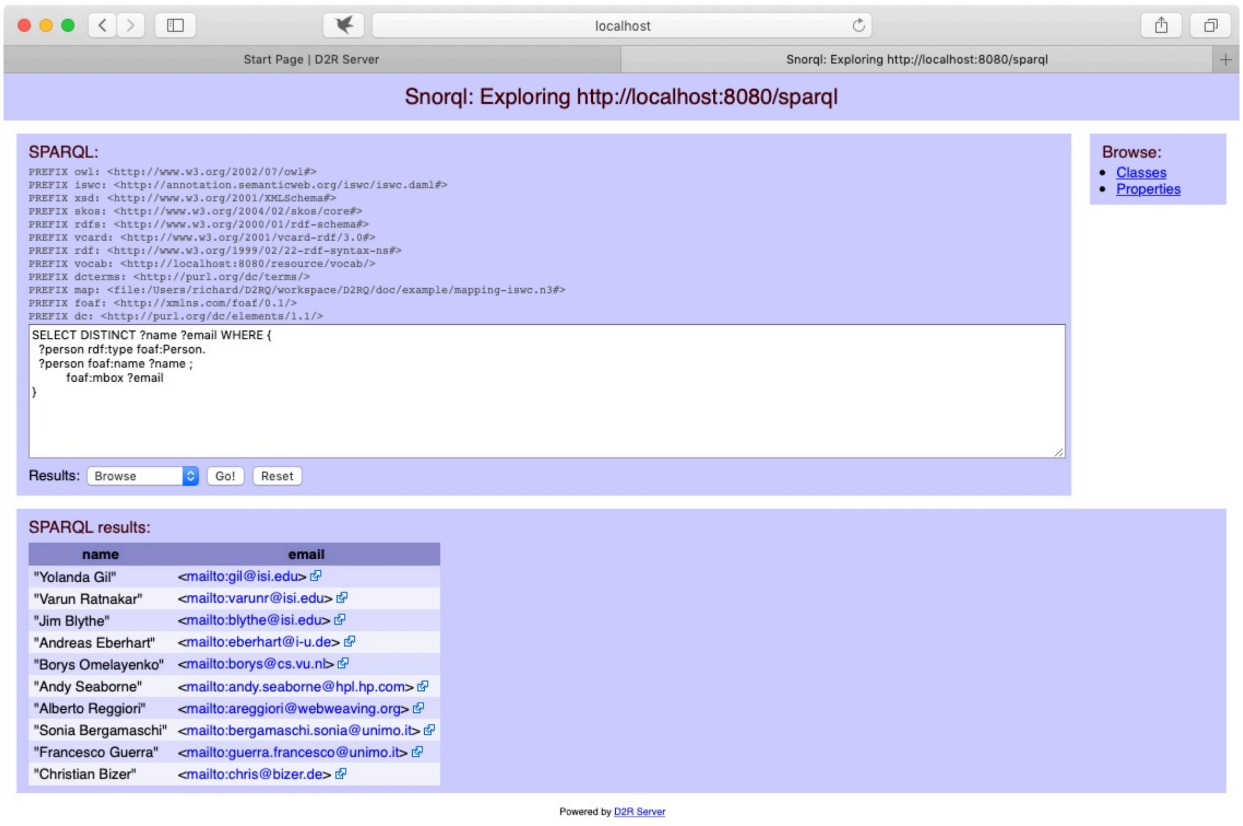
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX iswc: <http://annotation.semanticweb.org/iswc/iswc.daml#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX vcard: <http://www.w3.org/2001/vcard-rdf/3.0#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX vocab: <http://localhost:8080/resource/vocab/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX map: <file:/Users/richard/D2RQ/workspace/D2RQ/doc/example/mapping-iswc.n3#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT DISTINCT * WHERE {
  ?s ?p ?o
}
LIMIT 10
```

Browse:

- Classes
- Properties

Powered by D2R Server

In the 3.1, I use `select FirstName, LastName, email from persons;` in MySQL and get some results. Here, I use SPARQL to achieve the same query and get the same results.



SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX iswc: <http://annotation.semanticweb.org/iswc/iswc.daml#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX vcard: <http://www.w3.org/2001/vcard-rdf/3.0#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX vocab: <http://localhost:8080/resource/vocab/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX map: <file:/Users/richard/D2RQ/workspace/D2RQ/doc/example/mapping-iswc.n3#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT DISTINCT ?name ?email WHERE {
  ?person rdf:type foaf:Person.
  ?person foaf:name ?name ;
        foaf:mbox ?email
}
```

SPARQL results:

name	email
"Yolanda Gil"	<mailto:gil@isi.edu>
"Varun Ratnakar"	<mailto:varunr@isi.edu>
"Jim Blythe"	<mailto:blythe@isi.edu>
"Andreas Eberhart"	<mailto:eberhart@i-u.de>
"Borys Omelayenko"	<mailto:borys@cs.vu.nl>
"Andy Seaborne"	<mailto:andy.seaborne@hpl.hp.com>
"Alberto Reggiori"	<mailto:areggiori@webweaving.org>
"Sonia Bergamaschi"	<mailto:bergamaschi.sonia@unimo.it>
"Francesco Guerra"	<mailto:guerra.francesco@unimo.it>
"Christian Bizer"	<mailto:chris@bizer.de>

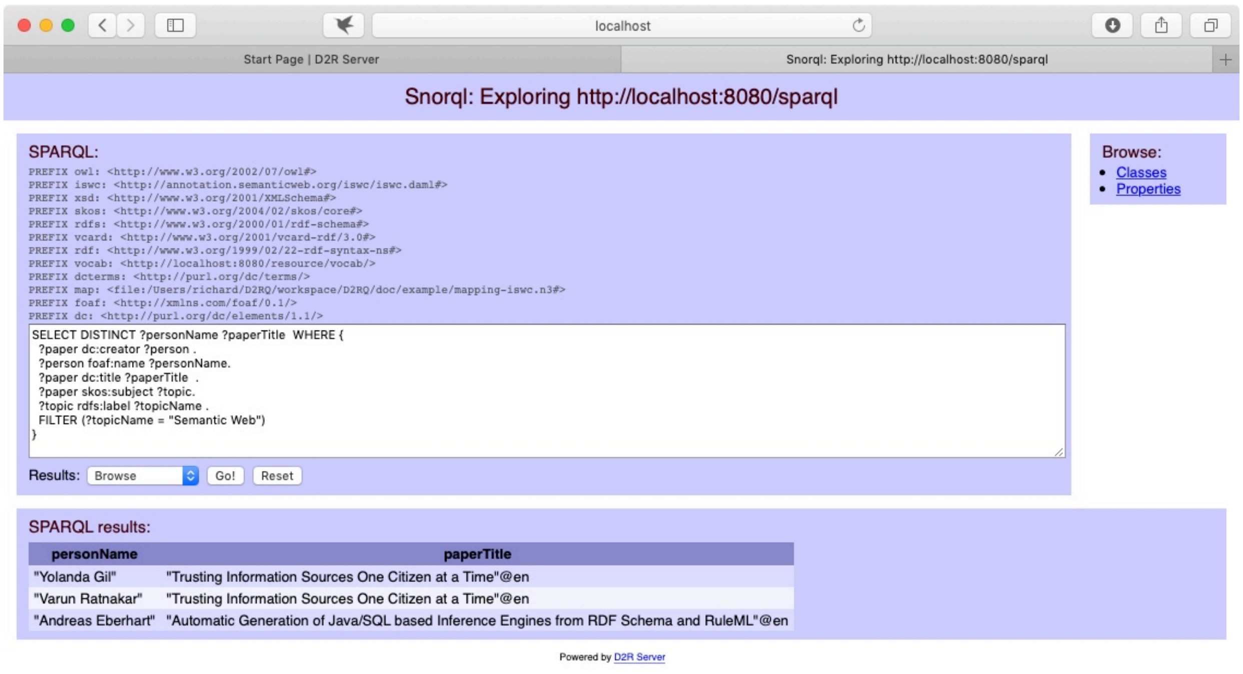
The results could also be saved as JSON, XML or XML+XSLT. For example, the JSON form of the query in this case is:

```
{
  "head": {
    "vars": [ "name" , "email" ]
  } ,
  "results": {
    "bindings": [
      {
        "name": { "type": "literal" , "value": "Yolanda Gil" } ,
        "email": { "type": "uri" , "value": "mailto:gil@isi.edu" }
      } ,
      {
        "name": { "type": "literal" , "value": "Varun Ratnakar" } ,
        "email": { "type": "uri" , "value": "mailto:varunr@isi.edu" }
      } ,
      {
        "name": { "type": "literal" , "value": "Jim Blythe" } ,
        "email": { "type": "uri" , "value": "mailto:blythe@isi.edu" }
      } ,
      {
        "name": { "type": "literal" , "value": "Andreas Eberhart" } ,
        "email": { "type": "uri" , "value": "mailto:eberhart@i-u.de" }
      } ,
      {
        "name": { "type": "literal" , "value": "Borys Omelayenko" } ,
          "email": { "type": "uri" , "value": "mailto:borys@cs.vu.nl" }
        } ,
        {
          "name": { "type": "literal" , "value": "Andy Seaborne" } ,
          "email": { "type": "uri" , "value":
  "mailto:andy.seaborne@hpl.hp.com" }
        } ,
        {
          "name": { "type": "literal" , "value": "Alberto Reggiori" } ,
          "email": { "type": "uri" , "value":
  "mailto:areggiori@webweaving.org" }
        } ,
        {
          "name": { "type": "literal" , "value": "Sonia Bergamaschi" } ,
          "email": { "type": "uri" , "value":
  "mailto:bergamaschi.sonia@unimo.it" }
        } ,
        {
          "name": { "type": "literal" , "value": "Francesco Guerra" } ,
          "email": { "type": "uri" , "value":
  "mailto:guerra.francesco@unimo.it" }
        } ,
        {
          "name": { "type": "literal" , "value": "Christian Bizer" } ,
          "email": { "type": "uri" , "value": "mailto:chris@bizer.de" }
        }
      ]
    }
  }
```

Another example of query using SPARQL is "Select people who wrote papers on the topic 'Semantic Web'":

```
SELECT DISTINCT ?personName ?paperTitle  WHERE {
  ?paper dc:creator ?person .
  ?person foaf:name ?personName.
  ?paper dc:title ?paperTitle  .
  ?paper skos:subject ?topic.
  ?topic rdfs:label ?topicName .
  FILTER (?topicName = "Semantic Web")
}
```

The result is showed as the figure below.



Snorql: Exploring http://localhost:8080/sparql

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX iswc: <http://annotation.semanticweb.org/iswc/iswc.daml#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX vcard: <http://www.w3.org/2001/vcard-rdf/3.0#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX vocab: <http://localhost:8080/resource/vocab/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX map: <file:/Users/richard/D2RQ/workspace/D2RQ/doc/example/mapping-iswc.n3#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT DISTINCT ?personName ?paperTitle  WHERE {
 ?paper dc:creator ?person .
 ?person foaf:name ?personName.
 ?paper dc:title ?paperTitle  .
 ?paper skos:subject ?topic.
 ?topic rdfs:label ?topicName .
 FILTER (?topicName = "Semantic Web")
}
```

SPARQL results:

personName	paperTitle
"Yolanda Gil"	"Trusting Information Sources One Citizen at a Time"@en
"Varun Ratnakar"	"Trusting Information Sources One Citizen at a Time"@en
"Andreas Eberhart"	"Automatic Generation of Java/SQL based Inference Engines from RDF Schema and RuleML"@en

Besides d2r-server, we could also use d2r-query to run a SPARQL query from the command line:

```
d2r-query mapping.ttl "SELECT * { ?s ?p ?o } LIMIT 10"
```

or

```
d2r-query mapping.ttl @query.sparql
```

4. Reference

- D2RQ platform http://d2rq.org/#compatibility
- D2R Server tutorial http://vadimeisenberg.github.io/d2rq/tutorial.html
- MySQL Connector/J https://dev.mysql.com/downloads/connector/j/