Semantic Days 2011 Tutorial
Semantic Web Technologies

Lecture 5: Presenting Relational Databases as RDF

Martin Giese

7th June 2011

Department of Informatics

University of Oslo

Outline

Outline

RDBMS to RDF

- Most of the worlds business data is stored in relational databases

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication. . .

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication…
    - Probably won't switch whole enterprise to RDF store

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
  - Read all records, export RDF
  - Bad idea: data replication...
  - Probably won't switch whole enterprise to RDF store
  - Need to convert to RDF regularly

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
  - Read all records, export RDF
  - Bad idea: data replication...
  - Probably won't switch whole enterprise to RDF store
  - Need to convert to RDF regularly
- Often a better idea: RDF view

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication...
    - Probably won't switch whole enterprise to RDF store
    - Need to convert to RDF regularly
- Often a better idea: RDF view
    - SPARQL endpoint translates incoming queries to SQL

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication...
    - Probably won't switch whole enterprise to RDF store
    - Need to convert to RDF regularly
- Often a better idea: RDF view
    - SPARQL endpoint translates incoming queries to SQL
    - Translates result to SPARQL SELECT result or RDF

RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication. . .
    - Probably won't switch whole enterprise to RDF store
    - Need to convert to RDF regularly
- Often a better idea: RDF view
    - SPARQL endpoint translates incoming queries to SQL
    - Translates result to SPARQL SELECT result or RDF
    - Data remains where it is, no duplication

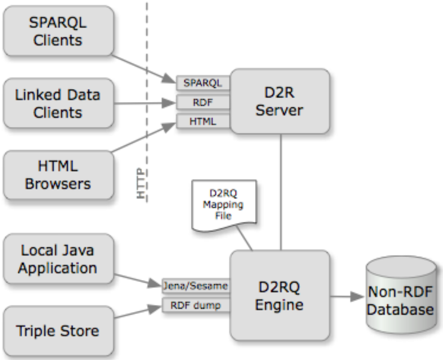
RDBMS to RDF

- Most of the worlds business data is stored in relational databases
- Need a way to make data in RDBMS available as RDF
- First idea: RDF export
    - Read all records, export RDF
    - Bad idea: data replication. . .
    - Probably won't switch whole enterprise to RDF store
    - Need to convert to RDF regularly
- Often a better idea: RDF view
    - SPARQL endpoint translates incoming queries to SQL
    - Translates result to SPARQL SELECT result or RDF
    - Data remains where it is, no duplication
    - Drawback: need to keep "old-fashioned" DB backend

Outline

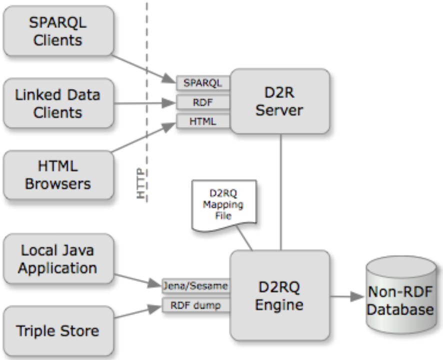
D2R/D2RQ

- Allows to treat relational databases as RDF



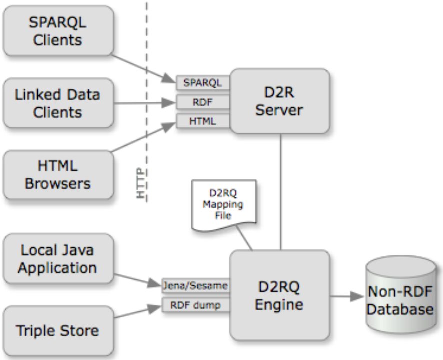
D2R/D2RQ

- Allows to treat relational databases as RDF
- Developed by FU Berlin



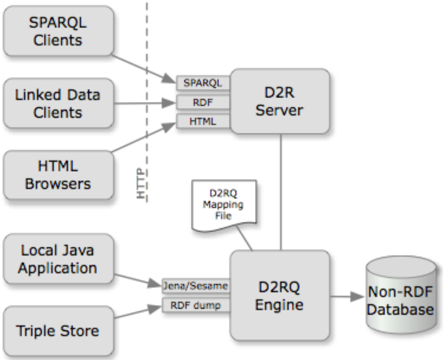
D2R/D2RQ

- Allows to treat relational databases as RDF
- Developed by FU Berlin
- Mapping describes relation between DB and RDF



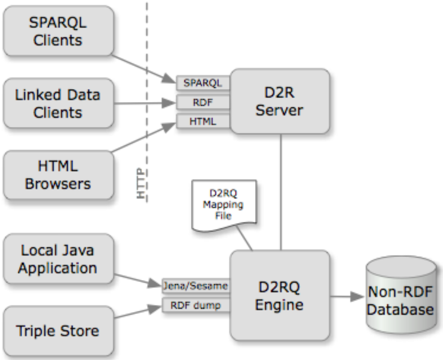
D2R/D2RQ

- Allows to treat relational databases as RDF
- Developed by FU Berlin
- Mapping describes relation between DB and RDF
- Can create SPARQL endpoint without transforming the whole database: *Virtual* RDF graph.



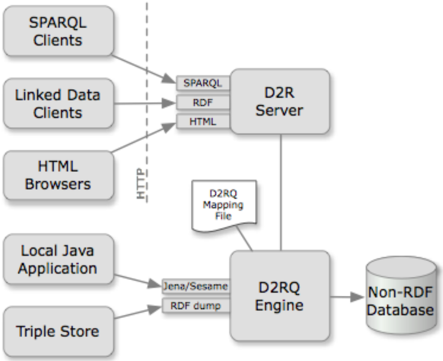
D2R/D2RQ

- Allows to treat relational databases as RDF
- Developed by FU Berlin
- Mapping describes relation between DB and RDF
- Can create SPARQL endpoint without transforming the whole database: *Virtual* RDF graph.
- Also on-demand RDF/HTML pages following LOD protocol



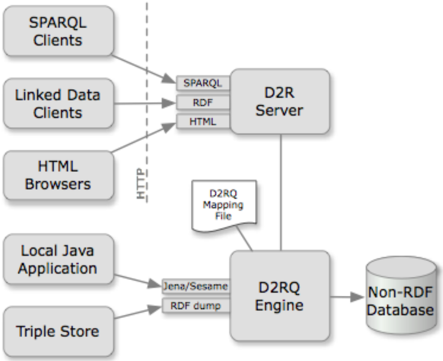
D2RQ Engine

- Reads a “Mapping File”



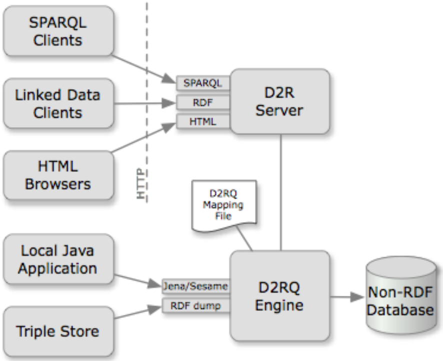
D2RQ Engine

- Reads a “Mapping File”
  - Table $\rightarrow$ Class



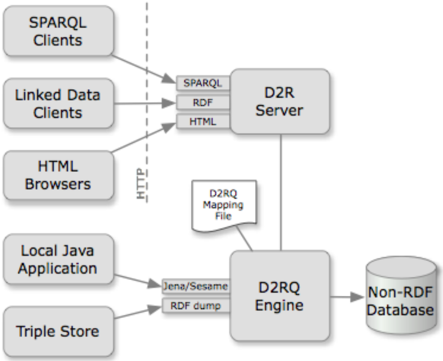
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource



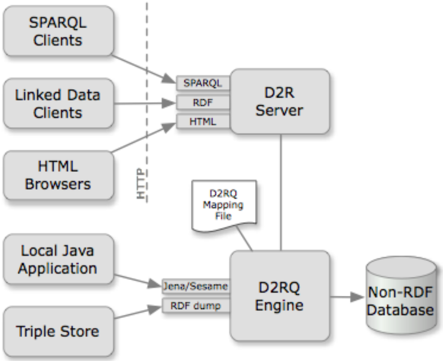
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource
  - Column → Property



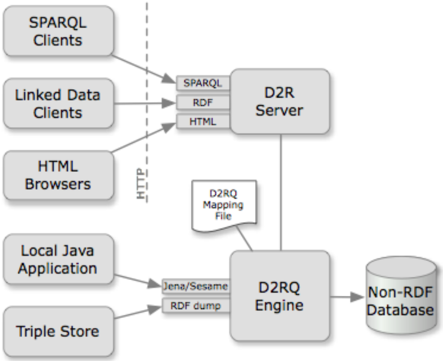
D2RQ Engine

- Reads a “Mapping File”
  - Table → Class
  - Row → Resource
  - Column → Property
  - RDF-encoded



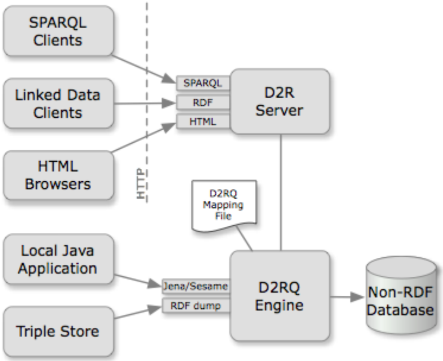
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource
  - Column → Property
  - RDF-encoded
- Translates SPARQL to SQL



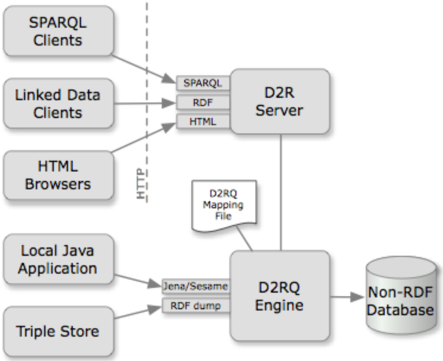
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource
  - Column → Property
  - RDF-encoded
- Translates SPARQL to SQL
- Can also act as Jena Graph



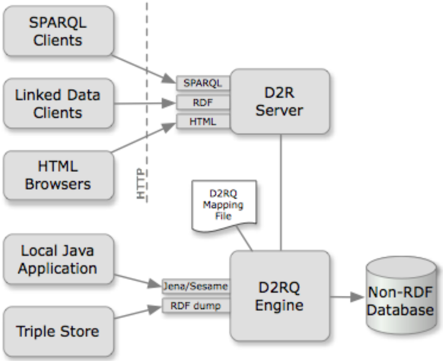
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource
  - Column → Property
  - RDF-encoded
- Translates SPARQL to SQL
- Can also act as Jena Graph
- Or the Sesame equivalent



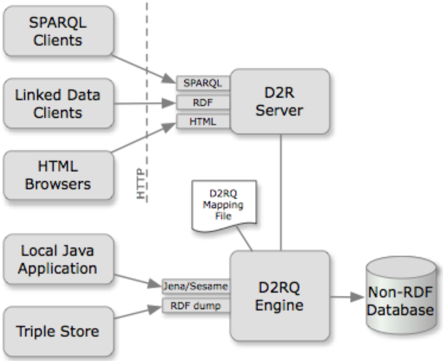
D2RQ Engine

- Reads a "Mapping File"
  - Table → Class
  - Row → Resource
  - Column → Property
  - RDF-encoded
- Translates SPARQL to SQL
- Can also act as Jena Graph
- Or the Sesame equivalent
- Can also export whole DB



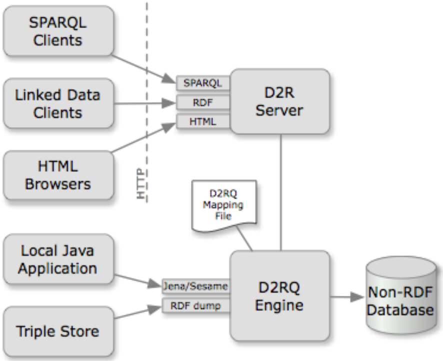
D2R Server

- Provides WWW-frontend



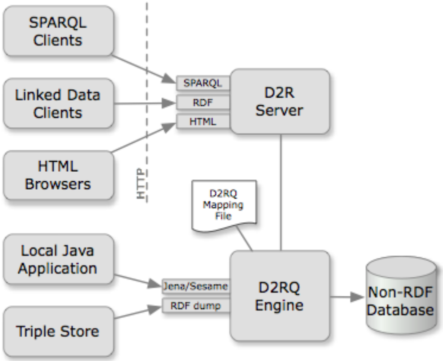
D2R Server

- Provides WWW-frontend
- SPARQL Endpoint



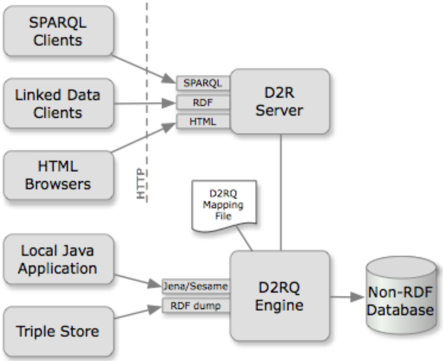
D2R Server

- Provides WWW-frontend
- SPARQL Endpoint
- Serves RDF as linked open data



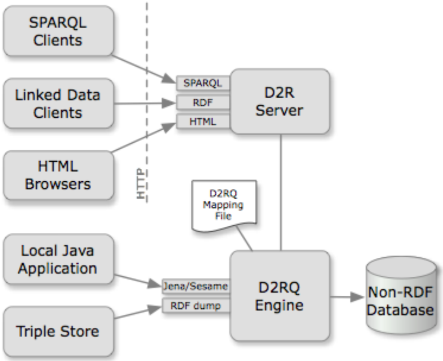
D2R Server

- Provides WWW-frontend
- SPARQL Endpoint
- Serves RDF as linked open data
- Pages of data for HTTP browsers



D2R Server

- Provides WWW-frontend
- SPARQL Endpoint
- Serves RDF as linked open data
- Pages of data for HTTP browsers
- All requests translated to SPARQL



Example: World Database

- An example database from MySQL distribution

Example: World Database

- An example database from MySQL distribution
- Table City:

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:
  - Code (key): the code for a country

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:
  - Code (key): the code for a country
  - Name: the Country's name

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:
  - Code (key): the code for a country
  - Name: the Country's name
  - Continent: the Continent the country lies in

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:
  - Code (key): the code for a country
  - Name: the Country's name
  - Continent: the Continent the country lies in
  - Capital: the City ID of the country's capital

Example: World Database

- An example database from MySQL distribution
- Table City:
  - ID (key): a unique number
  - Name: the city's name
  - CountryCode: Code for the country the city lies in
  - ...
- Table Country:
  - Code (key): the code for a country
  - Name: the Country's name
  - Continent: the Continent the country lies in
  - Capital: the City ID of the country's capital
  - ...

Example: World Database (cont.)

- Table City:

ID	**Name**	**CountryCode**	...
	...		
2806	Kingston	NFK	...
2807	Oslo	NOR	...
2808	Bergen	NOR	...
	...		

Example: World Database (cont.)

- Table City:

ID	**Name**	**CountryCode**	...
...			
2806	Kingston	NFK	...
2807	Oslo	NOR	...
2808	Bergen	NOR	...
...			

- Table Country:

Code	**Name**	**Continent**	**Capital**	...
...				
NLD	Netherlands	Europe	5	...
NOR	Norway	Europe	2807	...
NPL	Nepal	Asia	2729	...
...				

Outline

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class
    - contains placeholder with DB table and column

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a "class mapping"
- link to a resource describing the DB connection
- give the pattern for resources of this class
    - contains placeholder with DB table and column
- give the RDFS class for those resources

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class
  - contains placeholder with DB table and column
- give the RDFS class for those resources
- give the label for that class.

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class
  - contains placeholder with DB table and column
- give the RDFS class for those resources
- give the label for that class.
- Generates:

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class
    - contains placeholder with DB table and column
- give the RDFS class for those resources
- give the label for that class.
- Generates:
    - `<http://.../City/2806> a vocab:City.`

Where Classes Come From

- From a mapping file for the World database:

```
map:City a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "City/@@City.ID@@" ;
    d2rq:class vocab:City ;
    d2rq:classDefinitionLabel "City" .
```

- identify a "class mapping"
- link to a resource describing the DB connection
- give the pattern for resources of this class
  - contains placeholder with DB table and column
- give the RDFS class for those resources
- give the label for that class.
- Generates:
  - `<http://.../City/2806> a vocab:City.`
  - `<http://.../City/2807> a vocab:City.`

Where Classes Come From

- From a mapping file for the World database:

  ```
  map:City a d2rq:ClassMap ;
      d2rq:dataStorage map:database ;
      d2rq:uriPattern "City/@@City.ID@@" ;
      d2rq:class vocab:City ;
      d2rq:classDefinitionLabel "City" .
  ```

- identify a “class mapping”
- link to a resource describing the DB connection
- give the pattern for resources of this class
  - contains placeholder with DB table and column
- give the RDFS class for those resources
- give the label for that class.
- Generates:
  - `<http://.../City/2806> a vocab:City.`
  - `<http://.../City/2807> a vocab:City.`
  - `<http://.../City/2808> a vocab:City.`

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!
- For continents, add mapping:

```
map:Continent a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Continent/@@Country.Continent|urlify@@";
    d2rq:class vocab:Continent ;
    d2rq:classDefinitionLabel "Continent" .
```

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!
- For continents, add mapping:

```
map:Continent a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Continent/@@Country.Continent|urlify@@";
    d2rq:class vocab:Continent ;
    d2rq:classDefinitionLabel "Continent" .
```

- For everything in the `Continent` column of `Country`...

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!
- For continents, add mapping:

```
map:Continent a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Continent/@@Country.Continent|urlify@@";
    d2rq:class vocab:Continent ;
    d2rq:classDefinitionLabel "Continent" .
```

- For everything in the `Continent` column of `Country`...
- ...generate a resource with URI `.../Continent/...`

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!
- For continents, add mapping:

```
map:Continent a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Continent/@@Country.Continent|urlify@@";
    d2rq:class vocab:Continent ;
    d2rq:classDefinitionLabel "Continent" .
```

- For everything in the `Continent` column of `Country`…
- …generate a resource with URI `.../Continent/...`
- …removing spaces from “North America”, etc.

Resources for Countries and Continents

- The same for countries:

```
map:Country a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Country/@@Country.Code@@" ;
    d2rq:class vocab:Country ;
```

- Can have more classes than tables!
- For continents, add mapping:

```
map:Continent a d2rq:ClassMap ;
    d2rq:dataStorage map:database ;
    d2rq:uriPattern "Continent/@@Country.Continent|urlify@@";
    d2rq:class vocab:Continent ;
    d2rq:classDefinitionLabel "Continent" .
```

- For everything in the `Continent` column of `Country`...
- ...generate a resource with URI `.../Continent/...`
- ...removing spaces from “North America”, etc.
- E.g. `http://.../resource/Continent/North_America`

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate
- the object is a *literal* taken from this column

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate
- the object is a *literal* taken from this column
    - `<http://.../City/2806> vocab:name "Kingston".`

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate
- the object is a *literal* taken from this column
    - `<http://.../City/2806> vocab:name "Kingston".`
    - `<http://.../City/2807> vocab:name "Oslo".`

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate
- the object is a *literal* taken from this column
  - `<http://.../City/2806> vocab:name "Kingston".`
  - `<http://.../City/2807> vocab:name "Oslo".`
  - `<http://.../City/2808> vocab:name "Bergen".`

Where Properties Go To

- A mapping for city names:

```
map:City_Name a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:name ;
    d2rq:propertyDefinitionLabel "name" ;
    d2rq:column "City.Name" .
```

- Identify a “property bridge”
- that adds properties to the resources described in `map:City`
- give the predicate
- give a label to the predicate
- the object is a *literal* taken from this column
  - `<http://.../City/2806> vocab:name "Kingston".`
  - `<http://.../City/2807> vocab:name "Oslo".`
  - `<http://.../City/2808> vocab:name "Bergen".`
- Also possible to define literals with patterns containing columns

Linking Cities and Countries

- Want URIs as objects, not literal country codes.

Linking Cities and Countries

- Want URIs as objects, not literal country codes.
- Use the following property bridge:

```
map:City_CountryCode a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:inCountry ;
    d2rq:refersToClassMap map:Country ;
    d2rq:join "City.CountryCode=>Country.Code" .
```

Linking Cities and Countries

- Want URIs as objects, not literal country codes.
- Use the following property bridge:

```
map:City_CountryCode a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:inCountry ;
    d2rq:refersToClassMap map:Country ;
    d2rq:join "City.CountryCode=>Country.Code" .
```

- Foreign key: link to resource from another class map

Linking Cities and Countries

- Want URIs as objects, not literal country codes.
- Use the following property bridge:

```
map:City_CountryCode a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:inCountry ;
    d2rq:refersToClassMap map:Country ;
    d2rq:join "City.CountryCode=>Country.Code" .
```

- Foreign key: link to resource from another class map
- Say how columns for `map:City` correspond to those for `map:Country`

Linking Cities and Countries

- Want URIs as objects, not literal country codes.
- Use the following property bridge:

```
map:City_CountryCode a d2rq:PropertyBridge ;
    d2rq:belongsToClassMap map:City ;
    d2rq:property vocab:inCountry ;
    d2rq:refersToClassMap map:Country ;
    d2rq:join "City.CountryCode=>Country.Code" .
```

- Foreign key: link to resource from another class map
- Say how columns for `map:City` correspond to those for `map:Country`
- From countries to capitals:

```
map:Country_Capital a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property vocab:capital;
    d2rq:refersToClassMap map:City;
    d2rq:join "Country.Capital=>City.ID";
```

Resulting Graph

After adding similar mappings for country names and `inContinent`:

```
<http://.../City/2807> a vocab:City ;
         vocab:name "Oslo" ;
         vocab:inCountry <http://.../Country/NOR> .

<http://.../Country/NOR> a vocab:Country ;
         vocab:name "Norway" ;
         vocab:capital <http://.../City/2807> ;
         vocab:inContinent <http://.../Continent/Europe> .
```

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes
- No problem to link to “external” URIs.

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes
- No problem to link to “external” URIs.
- Careful: Generating links like this often fails for some cases:

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes
- No problem to link to “external” URIs.
- Careful: Generating links like this often fails for some cases:
  - World DB country name: `Sao Tome and Principe`

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes
- No problem to link to “external” URIs.
- Careful: Generating links like this often fails for some cases:
    - World DB country name: `Sao Tome and Principe`
    - DBpedia URI: `http://.../São_Tomé_and_Príncipe`

Linking to DBpedia

- Add property bridge:

```
map:Country_DBpedia a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Country;
    d2rq:property owl:sameAs;
    d2rq:uriPattern
"http://dbpedia.org/resource/@@Country.Name|urlify@@" .
```

- No problem to use “external” properties or classes
- No problem to link to “external” URIs.
- Careful: Generating links like this often fails for some cases:
  - World DB country name: `Sao Tome and Principe`
  - DBpedia URI: `http://.../São_Tomé_and_Príncipe`
- Better in general to have a DB table with corresponding URIs

Adding Reasoning

- Given an ontology saying that
  - Every Country is a Place
  - Every Contintent is a Place
- A SPARQL Query that asks for all places with "ica" in the name should find
  - South America, Africa,...
  - Costa Rica,...
- Not possible to attach reasoning to D2R (yet)
- Need to do some programming to connect reasoner and D2R
- See exercises to play with this!...

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009
- No full-fledged alternatives yet

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009
- No full-fledged alternatives yet
- W3C working draft on R2ML:

  `http://www.w3.org/TR/r2rml/`

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009
- No full-fledged alternatives yet
- W3C working draft on R2ML:

  `http://www.w3.org/TR/r2rml/`

- Very similar in most respects

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009
- No full-fledged alternatives yet
- W3C working draft on R2ML:

  `http://www.w3.org/TR/r2rml/`

- Very similar in most respects
- Only partial, experimental implementations so far

The Future of D2R/D2RQ

- Last version of D2R, v0.7, is from August 2009
- No full-fledged alternatives yet
- W3C working draft on R2ML:

  `http://www.w3.org/TR/r2rml/`

- Very similar in most respects
- Only partial, experimental implementations so far
- Full implementations will come, and replace D2R

Exercise: D2R Server

- Install D2R server.
- Generate mapping.
- Start server.
- Change mapping.
- Run Java program querying D2R data after reasoning.

Go to `http://sws.ifi.uio.no/semdays2011/` for the full exercise text.