Instituto Politécnico
de Viana do Castelo

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA
Departamento de Informática

An Open Data Approach to Publish Relational Data

Miguel Bento Alves, João Ferreira Nunes

TREASURE Project

- Our project arises with the need to share and make public the data produced under the TREASURE project – a Research & Innovation Action financed by European Commission under the Horizon 2020 (grant agreement no. 634476).

- The aim of the project is to improve knowledge, skills and competences necessary to develop new sustainable pork chains based on European local pig genetic resources (local breeds).

TREASURE Data

- Our goal consists in publishing TREASURE data by means of an Open Data approach.
- The data will be available for all stakeholders in a standard format.
- Furthermore, Open Data implies the use of standards such as HTTP, RDF or SPARQL, making it easier to use on the web.

Open Data

- Open Data's principle claims that data should be **freely available**, without any kind of restrictions from copyright, patents or other mechanisms of control.
- Another key concept that it is implicit to this ideal, is the **interoperability**, which refers to the capability of several systems and organizations in working together.
  - In this specific case, it refers to the capability to combine – or inter-operate – different sets of data.

From Relational to Open Data

- Initially, the information requirements were analyzed based on a relational model approach to create a relational database.

- In order to reuse all the work produced during the initial phase, it was decided to replicate the relational model for a Semantic Web approach.

From Relational to Open Data

- All data will be kept as RDF;
- In our Semantic Web approach, all relational rules are guaranteed in the RDF data;
- Although the focus of our work was to publish the produced data from the TREASURE project on an Open Data approach, the developed system that we designed is adaptable to any relational database.

Ontologies and datasets

- We proposed a three layers model, where:
  - at the **upper-level** the most important (or most used) concepts of the relational model are modeled;
  - at the **middle-level** the meta-model of the relational database is modeled;
  - at the **lower-level** the database information is represented.

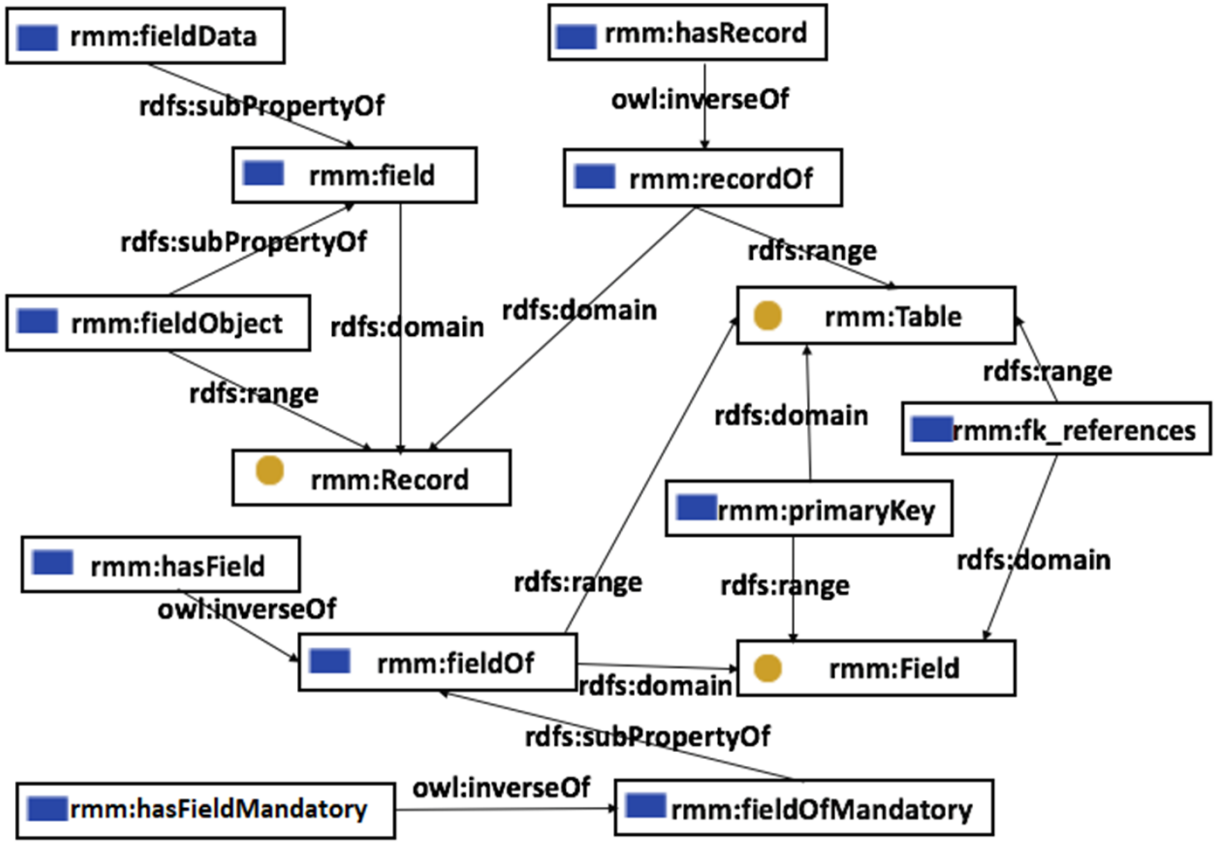
Ontologies and datasets

- With the two highest layers, we have all the knowledge on how the information in a database is organized, and from that we can extract information about what is modeled;

- This allows us to support reasoning on the data model;

A. Relational Model Ontology

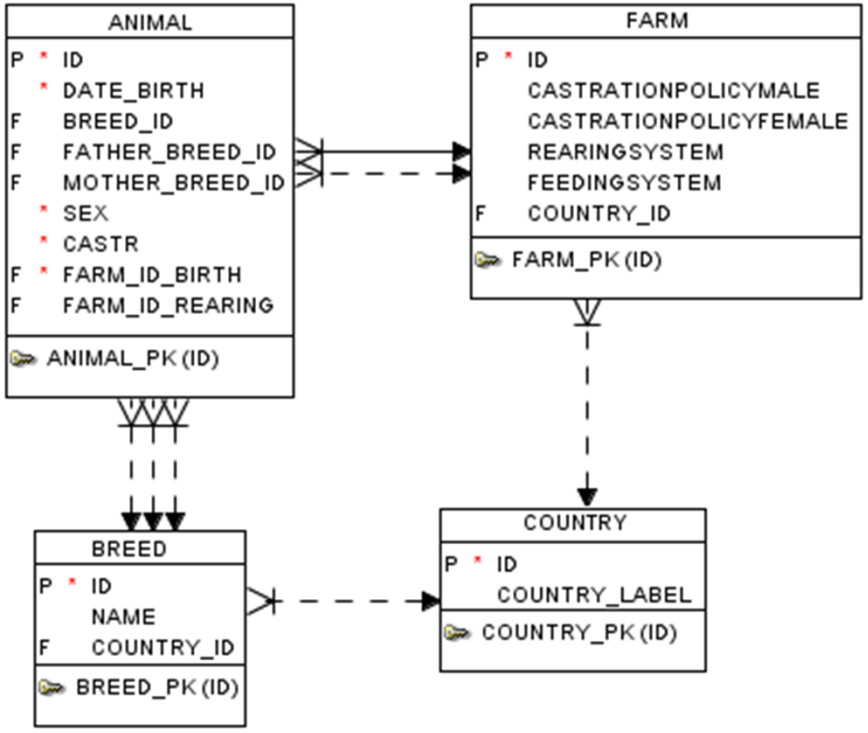
- In the **upper-level**, we have created an ontology to represent the concepts of relational databases.



B. Database meta-model

- At the **middle-level** it is represented the meta-model of the database, namely which tables were created, which fields have each table, what are the primary keys of the tables and the foreign keys.

B. Database meta-model




```
exa_mm:Country rdf:type rmm:Table ;
    rmm:primaryKey <http://www.example.com/exa_mm.ttl/Country#id>.
<http://www.example.com/exa_mm.ttl/Country#id>rmm:fieldOf exa_mm:Country ;
    rdfs:subPropertyOf rmm:fieldData .
<http://www.example.com/exa_mm.ttl/Country#country_label>
    rmm:fieldOf exa_mm:Country ;
    rdfs:subPropertyOf rmm:fieldData .

exa_mm:Breed rdf:type rmm:Table ;
    rmm:primaryKey <http://www.example.com/exa_mm.ttl/Breed#id>.
<http://www.example.com/exa_mm.ttl/Breed#id>rmm:fieldOf exa_mm:Breed ;
    rdfs:subPropertyOf rmm:fieldData .
<http://www.example.com/exa_mm.ttl/Breed#country_id>
    rmm:fieldOf exa_mm:Breed ;
    rdfs:subPropertyOf rmm:fieldObject ;
    rmm:fk_references exa_mm:Country .

exa_mm:Farm rdf:type rmm:Table ;
    rmm:primaryKey <http://www.example.com/exa_mm.ttl/Farm#id>.
<http://www.example.com/exa_mm.ttl/Farm#id>rmm:fieldOf exa_mm:Farm ;
    rdfs:subPropertyOf rmm:fieldData .
<http://www.example.com/exa_mm.ttl/Farm#castrationPolicyMale>
    rmm:fieldOf exa_mm:Farm ;
    rdfs:subPropertyOf rmm:fieldData .
<http://www.example.com/exa_mm.ttl/Farm#country_id>
    rmm:fieldOf exa_mm:Farm ;
    rdfs:subPropertyOf rmm:fieldObject ;
    rmm:fk_references exa_mm:Country .
```

B. Database meta-model

- Can be fed by the database catalogue (data dictionary)

```
Select utc.table_name, utc.column_name, utc.data_type,
sql.constraint_type, sql.foreignTable, sql.foreignColumn
from user_tab_columns utc
   left join
   (select constraint_type, uc.table_name,
        ucc.column_name, uccr.table_name foreignTable,
        uccr.column_name foreignColumn
    from user_constraints uc
        inner join user_cons_columns ucc
        on (uc.constraint_name = ucc.constraint_name)
        left join user_cons_columns uccr
        on (uc.r_constraint_name = uccr.constraint_name)) sql
    on (utc.table_name = sql.table_name
        and utc.column_name = sql.column_name)
order by table_name, column_id;
```

	TABLE_NAME	COLUMN_NAME	DATA_TYPE	CONSTRAINT_TYPE	FOREIGNTABLE	FOREIGNCOLUMN
1	ANIMAL	ID	NVARCHAR2	P	(null)	(null)
2	ANIMAL	DATE_BIRTH	DATE	C	(null)	(null)
3	ANIMAL	BREED_ID	NVARCHAR2	R	BREED	ID
4	ANIMAL	FATHER_BREED_ID	NVARCHAR2	R	BREED	ID
5	ANIMAL	MOTHER_BREED_ID	NVARCHAR2	R	BREED	ID
6	ANIMAL	SEX	NUMBER	C	(null)	(null)
7	ANIMAL	CASTR	NUMBER	C	(null)	(null)
8	ANIMAL	FARM_ID_BIRTH	NVARCHAR2	R	FARM	ID
9	ANIMAL	FARM_ID_BIRTH	NVARCHAR2	C	(null)	(null)
10	ANIMAL	FARM_ID_REARING	NVARCHAR2	R	FARM	ID
11	BREED	ID	NVARCHAR2	P	(null)	(null)
12	BREED	NAME	NVARCHAR2	C	(null)	(null)
13	BREED	COUNTRY_ID	CHAR	R	COUNTRY	ID
14	COUNTRY	ID	CHAR	P	(null)	(null)
15	COUNTRY	COUNTRY_LABEL	NVARCHAR2	(null)	(null)	(null)
16	FARM	ID	NVARCHAR2	P	(null)	(null)
17	FARM	CASTRATIONPOLICYMALE	NUMBER	(null)	(null)	(null)
18	FARM	CASTRATIONPOLICYFEMALE	NUMBER	(null)	(null)	(null)
19	FARM	REARINGSYSTEM	NUMBER	(null)	(null)	(null)
20	FARM	FEEDINGSYSTEM	NUMBER	(null)	(null)	(null)
21	FARM	COUNTRY_ID	CHAR	R	COUNTRY	ID

C. Dataset

- In the most specific layer (the **lower-level**), the data itself is represented.

```
country:pt rmm:recordOf exa_mm:Country ;
      country:id "PT" ;
      country:country_label "Portugal" .

exa_mm:Country rmm:hasRecord [
      country:id "FR" ;
      country:country_label "French"
].

breed:b2703 rmm:recordOf exa_mm:Breed ;
      breed:id "b2703" ;
      breed:name "Bisaro" ;
      breed:country_id country:pt .
```

Developed System

- We've developed two different tools to manipulate data in the open data repository.
- The first one allows to select one local database and transfer the data to the Open Data repository.
- The second one is a SPARQL endpoint, that can be used either in a program or with a Web interface, that allows the execution of SPARQL commands in the central repository.
- Both tools guarantees the integrity of the data considering the relational constraints implemented.

Developed System

- We've developed our own SPARQL endpoint instead of using Fuseki (the SPARQL server of Jena package), because we implemented several relational constraints over our central repository and we wanted to control the integrity of the data against the relational constraints.

Developed System

Sparql endpoint

SPARQL Command

```
prefix exa_mm: <http://www.example.com/exa_mm.ttl/>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix rmm: <http://www.estg.ipvc.pt/mbentoalves/rmm.ttl#>
prefix country: <http://www.example.com/exa_mm.ttl/Country#>

Select ?id ?name {
        ?s rmm:recordOf exa_mm:Country ;
                country:id ?id ;
                country:country_label ?name .
}
```

Get Results

Output

```
----------------------
| id    | name       |
======================
| "PT"  | "Portugal" |
| "FR"  | "French"   |
----------------------
```

Developed System

- Our system was developed using the Jena Framework, a free and open source Java framework for building Semantic Web applications.
- Jena provides a programmatic environment for RDF, RDFS, OWL, a query engine for SPARQL and it includes a rule-based inference engine.
- Jena is widely accepted for Semantic Web applications because it offers an "all-in-one" Java solution.

Rules to implement relational constraints

- All relational constraints are guaranteed by semantic rules:
  - RULE 1: avoid repeated fields in a table;
  - RULE 2: avoid violation of primary key constraint;
  - RULE 3: avoid violation of foreign key constraint;

Conclusions

- We've developed a system, in a semantic web approach, to publish relational data as Open Data;
- A three layer model was proposed to support the knowledge about relational data;
  - A relational model ontology it was also developed;
- Semantic rules were developed to support the relational data constraints;

Conclusions

- We've developed two tools to manipulate data in the open data repository.
  - The first one allows to select one local database and transfer the data to the Open Data repository.
  - The second one is a SPARQL endpoint, that can be used either in a program or with a Web interface, that allows the execution of SPARQL commands in the central repository.

Instituto Politécnico
de Viana do Castelo

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA
Departamento de Informática

An Open Data Approach to Publish Relational Data

Miguel Bento Alves, João Ferreira Nunes

THANK YOU