RDF and RDB 1

Some slides adapted from a presentation by Ivan Herman at the Semantic Technology & Business Conference, 2012.
Mapping Relational data to RDF

Suppose we have data in a relational database that we want to export as RDF

1. Choose an RDF vocabulary to represent the data
2. Define a mapping from the relational tables to RDF

Then either:

a) Materialize the RDF triples from the database using the mappings
b) Use a server to dynamically access the relational data given a SPARQL query
c) Use a DBMS that directly supports RDF (e.g., Oracle 11g, DB2)
Relational database vendors realize the importance of the Semantic Web market.

Many systems have a “hybrid” view:
- Traditional, relational storage, usually coupled with SQL
- RDF storage, usually coupled with SPARQL
- Examples include Oracle 11g, IBM’s DB2 and OpenLink Virtuoso

The model involves exporting relational data to RDF.
Export does not necessarily mean physical conversion
- for very large databases a “duplication” would not be an option
- systems may provide SPARQL⇔SQL “bridges” to make queries on the fly

Result of export is a “logical” view of the relational content
Simple export: Direct Mapping

- Provide a canonical RDF “view” of relational tables
- Only needs the information in the RDB Schema
Direct mapping approach

Each column name provides a predicate

Each row is a subject

Foreign keys refer to subjects in another table

Cell values are literal objects
Direct mapping approach

- RDF graph generated from relational database with its schema
- Can automatically generate an SQL query to answer a SPARQL query that directly uses the relational DB
Advantages of Direct mapping

- Simple, does not require any other concepts
- Know schema ⇒ know RDF graph structure
- Know RDF graph structure ⇒ good idea of schema (!)

Disadvantages:

- Resulting may not be what application wants
- Except for foreign keys, all cell values become literals, i.e. strings, not things
- Don’t want to force the database to be re-designed to expose more cell values as objects
Extended mapping approach

- **RDB Schema**
- **Direct Mapping**
  - "Direct Graph"
- **Graph Processing** (Rules, SPARQL, …)
- **Final, Application Graph**
- **Tables**
Beyond Direct Mapping: R2RML

- R2RML: RDB to RDF Mapping Language
  - W3C recommendation 9/2012 [link](#)

- Separate vocabulary to control the details of the mapping, e.g.:
  - finer control over choice of the subject
  - creation of URI references from cells
  - predicates may be chosen from a vocabulary
  - datatypes may be assigned
  - etc.

- Produce final RDF graph in one step
Beyond Direct Mapping: R2RML

- RDB Schema
- R2RML Instance
- Tables

Final, Application Graph
Fundamentals are similar:
- Each row => set of triples with common subject

Direct mapping is a “default” R2RML mapping

Which approach?
- depends on local tools, personal experiences and background,…
- You can begin with a “default” R2RML, and gradually refine it
D2RQ was a practical system first developed in 2004 that is widely used.

W3C formed a RDB2RDF working group in 2009 to develop a standard.

R2RML: RDB to RDF Mapping Language is a W3C recommendation since 2013-09-27.

Several implementations are available.