Using Linked Data Concepts to Blend and Analyze Geospatial and Statistical Data

Creating a Semantic Data Platform

Hans Viehmann
Product Manager EMEA
ORACLE Corporation
October 17, 2018

@SpatialHannes
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
What is Linked Data

• Concept of publishing and interlinking structured data on the web
  – Moving from documents to useable data

• Based on W3C standards
  – Resource Description Framework (RDF), OWL, SPARQL ...

• Originally developed by Tim Berners-Lee

• Design principles
  – Use Uniform Resource Identifiers (URIs) to uniquely identify things (data entities)
  – Use HTTP URLs, corresponding to these URIs, so that information can be retrieved
  – Provide metadata using open standards such as RDF
  – Include links to related URIs, so that people can discover more things
Why Linked Data for Statistics Agencies?

• Data dissemination is key to the work of NSIs
  – Production of accurate data is not everything

• Need to broaden reach
  – Simplifying access

• Various models to represent data in the past
  – eg. SDMX (Statistical Data and Metadata Exchange)

• Linked Data allow to
  – Associate data with metadata (or meaning) – major step towards AI
  – Use a common vocabulary
  – Refer to data owner/authoritative datasource or other 3rd party datasets
Two example projects

Based on Oracle Spatial and Graph as Linked Data Platform

• CensLOD project, ISTAT, Italy
  – Publishing 2015 census data as linked (open) data
  – Infrastructure development to publish Linked Open Data
  – Project details kindly provided by Monica Scannapieco

• E-stat Portal, National Statistics Center of Japan (NSTAC)
  – Expanding e-stat portal (www.e-stat.go.jp) to publish statistics as linked open data
  – Using RDF Data Cube vocabulary
  – Geometries based on Statistical standard area, small area, and grid square code
  – Material kindly provided by Shoki Nishimura (NSTAC)
General project flow

• Domain analysis and ontology definition
  – Using Protégé or Topbraid Composer as ontology editor

• Creation of subject-predicate-object triples
  – Mapping of source data
  – Inferencing

• Publishing
  – Setup of SPARQL endpoint
  – Development of UI
Ontology definition

ISTAT CensLOD

- Territorial data ontology, census data ontology
- Modeled in OWL using Protégé
- Based on existing meta-ontologies
  - SKOS and XKOS: skos:Concept, ...
  - ADMS: adms:AssetRepository, ...
  - Data Cube Vocabulary: qb:DataSet, qb:Observation, ...
  - PROV: prov:wasGeneratedBy, ...
  - GeoNames: gn:name, gn:countryCode, gn:parentCountry, ...
- Territorial data resulting in 95 entities and 200 rules, eg. using EquivalentTo to link entities to respective Geonames entity
Generating triples

- Describing mapping rules using R2RML standard
  - eg. associating column names with entities

- Choosing rulebase and possible optimizations
  - Using inferencing engine to materialize additional triples for performance

- Creating RDF Views on relational data (optional)
  - No duplication of data and storage

Image courtesy of: NSTAC, Japan
Creating a SPARQL endpoint
  – For machine-to-machine access
  – For advanced users

Linked Data interface
  – Faceted search/graph browser
  – For basic users

GUI to download datasets
  – For basic users
  – For advanced users
Linked Geodata and Semantic GIS

• GIS applications with semantically complex thematic aspects
  – Logical reasoning to classify features, eg. land cover type, suitable farm land, etc., combined with spatial queries
  – Linking to available datasources (geonames.org, dbpedia, ...)

• Requirements
  – Consistent modeling of geospatial data, both simple and complex
  – Geometric functions and topological queries based on Spatial indexing

• Conceptual solution provided by OGC standards
  – Simple Features as WKT literals, eg. "Point(-83.4 34.3)"^^ogc:wktLiteral
  – Queries in GeoSPARQL
Integrating Geometry

Administrative Divisions

Japan

- prefectural divisions (e.g., Tokyo)
- municipal divisions (e.g., Shinjuku-ward)

- standard area codes

Statistics

- world grid square system
- statistics-specific codes

Grid Squares (added in 2018)

- Divide Japan into grids in three levels (resolutions)

Small Areas (added in 2018)

- Subdivided portions of a municipal divisions

Image courtesy of: NSTAC, Japan
Integrating Geometry

• Storing boundary data with corresponding geometry
  – Either locally or remote

• Retrieving statistical data associated with target area
  – GeoSPAQRL FILTER statement including geometry
  – Can be federated query across SPARQL endpoints

Image courtesy of: NSTAC, Japan
Example – Federated Queries
blogs.oracle.com/oraclespatial

• Fundamental concept of Semantic Technologies
  – Implemented through SERVICE clause in SPARQL

• Federated SPARQL query inside Oracle Database is possible
  – In this case SPARQL in SQL
  – Also works with other SPARQL clients
  – Take predicate pushdown into account
## Single platform for geospatial and linked data

### Transformation and Modeling Tools
- Relational2RDF
- Plug-in for Protégé
- Topbraid Composer integration
- Support for Apache Jena
- Natural Language Processing Extraction (partners)

### Load, Query and Inferencing
- RDF/OWL Data Management
- SQL & SPARQL Query
- OWL Inferencing
- Semantic Rules
- Semantic Indexing
- Scalability & Security
- SQL Developer integration

### Solution Development and Analytic Tools
- Java, HTTP access
- JSON, XML output
- Graph visualization (Cytoscape)
- Oracle Advanced Analytics (R, Mining)
- Oracle Business Intelligence (OBIEE)
- Map (GIS) Visualization
Geospatial Linked Data Platform for NSIs

Graph Analysis and Reporting

Linked Data

Graph Data Management
Query Engine
Ontologies
Inferencing Engine

GeoSPARQL, SQL, Java, ...

RDF Graph Layer

Enterprise Data Stores

Statistics Data
Geometry Data

Public LOD endpoints (DBPedia, GeoNames, ...)

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.
Linked Data Platform at NSTAC

- Moved from database consolidation platform to Oracle Cloud
- Increasing data volume from 500M RDF triples to > 2bn triples by end-2018
- Planned work:
  - Expand number of datasets
  - Generate RDF triple from legacy data semi-automatically
Resources

• Oracle Spatial and Graph OTN product page here
  – White papers, software downloads, documentation and videos
  – Performance White Paper on 1 Trillion Triple Benchmark

• Oracle Big Data Lite Virtual Machine - a free sandbox to get started:

• Hands On Lab with RDF Graph data included
  – Content also available on GITHUB under http://github.com/oracle/BigDataLite/

• Blog – examples, tips & tricks: blogs.oracle.com/oraclespatial

• @OracleBigData, @SpatialHannes, @JeanIhm

Oracle Spatial and Graph Group
Integrated Cloud
Applications & Platform Services
Appendix

OGC GeoSPARQL Support in Oracle Spatial and Graph 12c
OGC GeoSPARQL

• GeoSPARQL – A Geographic Query Language for RDF Data
  – OGC Standard (document 11-052r4)
  – Published in June 2012
  – Submitting Organizations
Why GeoSPARQL? – Linked Geo Data

• Many Linked Open Data (LOD) datasets have geospatial components
• Barriers to integration
  – Vendor-specific geometry support
  – Different vocabularies
    • W3C Basic Geo, GML XMLLiteral, Vendor-specific
  – Different spatial reference systems
    • WGS84 Lat-Long, British National Grid
Why GeoSPARQL? – Semantic GIS

• GIS applications with semantically complex thematic aspects
  – Logical reasoning to classify features
    • Land cover type, suitable farm land, etc.
  – Complex Geometries
    • Polygons and Multi-Polygons with 1000’s of points
  – Complex Spatial Operations
    • Union, Intersection, Buffers, etc.

Find parcels with an area of at least 3 sq. miles that touch a local feeder road and are inside an area of suitable farm land.
From SPARQL to GeoSPARQL

RDF Data

:res1 rdf:type :House .
:res1 :baths "2.5"^^xsd:decimal .
:res1 :bedrooms "3"^^xsd:decimal .

:res2 rdf:type :Condo .
:res2 :baths "2"^^xsd:decimal .
:res2 :bedrooms "2"^^xsd:decimal .

:res3 rdf:type :House
:res3 :baths "1.5"^^xsd:decimal .

SPARQL Query

SELECT ?r ?ba ?br
?r :bedrooms ?br }

Result Bindings

<table>
<thead>
<tr>
<th>?r</th>
<th>?ba</th>
<th>?br</th>
</tr>
</thead>
<tbody>
<tr>
<td>:res1</td>
<td>&quot;2.5&quot;</td>
<td>&quot;3&quot;</td>
</tr>
<tr>
<td>:res3</td>
<td>&quot;1.5&quot;</td>
<td>&quot;3&quot;</td>
</tr>
</tbody>
</table>
**From SPARQL to GeoSPARQL**

**RDF Data**

```r
:res1 rdf:type :House .
:res1 :baths "2.5"^^xsd:decimal .
:res1 :bedrooms "3"^^xsd:decimal .

:res2 rdf:type :Condo .
:res2 :baths "2"^^xsd:decimal .
:res2 :bedrooms "2"^^xsd:decimal .

:res3 rdf:type :House
:res3 :baths "1.5"^^xsd:decimal .
```

**SPARQL Query**

```sql
SELECT ?r ?ba ?br
  ?r :bedrooms ?br
  FILTER (?ba > 2) }
```

**Result Bindings**

```
?r | ?ba | ?br
:res1 | "2.5" | "3"
```
From SPARQL to GeoSPARQL

**Spatial RDF Data**

```
:res1 rdf:type :House .
:res1 :baths "2.5"^^xsd:decimal .
:res1 :bedrooms "3"^^xsd:decimal .
:res1 ogc:hasGeometry :geom1 .
:geom1 ogc:asWKT "POINT(-122.25 37.46)"^^ogc:wktLiteral .
:res3 :baths "1.5"^^xsd:decimal .
:res3 ogc:hasGeometry :geom3 .
:geom3 ogc:asWKT "POINT(-122.24 37.47)"^^ogc:wktLiteral .
```

**GeoSPARQL Query**

```
SELECT ?r ?ba ?br
  ?r ogc:hasGeometry ?g . ?g ogc:asWKT ?wkt
  FILTER(ogcf:sfWithin(?wkt, "POLYGON(...)"^^ogc:wktLiteral))
}
```

This is what GeoSPARQL standardizes

Vocabulary & Datatypes

Extension Functions

Find houses within a search polygon
RDB2RDF for viewing Spatial Data as RDF

<table>
<thead>
<tr>
<th>Relational Data</th>
<th>HOUSE table</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>baths</td>
</tr>
<tr>
<td>int</td>
<td>number</td>
</tr>
<tr>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
</tr>
</tbody>
</table>

RDB2RDF: Direct Mapping

```sql
sem_apis.CREATE_RDFVIEW_MODEL ('House_Model', sys.odcivarchar2list('HOUSE'), 'http://dm/');
```

RDF View (of Relational Data)

```xml
<http://dm/RDFUSER.HOUSE/ID=1>
  rdf:type
    <http://dm/RDFUSER.HOUSE>;
  :baths
    "2.5"^^xsd:decimal;
  :bedrooms "3"^^xsd:decimal;
  :geom
    "POINT(...)"^^ogc:wktLiteral.
</http://dm/RDFUSER.HOUSE/ID=1>

<http://dm/RDFUSER.HOUSE/ID=3>
  rdf:type
    <http://dm/RDFUSER.HOUSE>;
  :baths
    "1.5"^^xsd:decimal;
  :bedrooms "3"^^xsd:decimal;
  :geom
    "POINT(...)"^^ogc:wktLiteral.
</http://dm/RDFUSER.HOUSE/ID=3>
```
### RDB2RDF for viewing Spatial Data as RDF

#### Relational Data

<table>
<thead>
<tr>
<th>id</th>
<th>baths</th>
<th>bedrooms</th>
<th>geom</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>number</td>
<td>number</td>
<td>SDO_GEOMETRY</td>
</tr>
<tr>
<td>1</td>
<td>2.5</td>
<td>3</td>
<td>POINT(-122.25 37.46)</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
<td>3</td>
<td>POINT(-122.24 37.47)</td>
</tr>
</tbody>
</table>

#### House table

```
<table>
<thead>
<tr>
<th>id</th>
<th>baths</th>
<th>bedrooms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.5</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>1.5</td>
<td>3</td>
</tr>
</tbody>
</table>
```

#### RDB2RDF: Direct Mapping

```python
sem_apis.CREATE_RDFVIEW_MODEL ('House_Model',
   sys.odcivarchar2list('HOUSE'),
   'http://dm/');
```

#### Querying RDF View

```sql
PREFIX : <http://dm/RDFUSER.HOUSE#>.
SELECT ?r ?ba ?br
WHERE {
  ?r rdf:type <http://dm/RDFUSER.HOUSE>;
  :baths ?ba;
  :bedrooms ?br;
  :geom ?wkt.
FILTER (ogcf:sfWithin(?wkt, "POLYGON(...)"^^ogc:wktLiteral))
}
```
GeoSPARQL Support in Oracle

• Oracle Spatial and Graph supports the following conformance classes for GeoSPARQL
  – Core
  – Topology Vocabulary Extension (Simple Features)
  – Geometry Extension (WKT, 1.2.0)
  – Geometry Topology Extension (Simple Features, WKT, 1.2.0)
  – RDFS Entailment Extension (Simple Features, WKT, 1.2.0)
Builds on the power of Oracle Spatial

• Efficient Spatial Indexing
• Spatial Reference Systems
  – Built-in support for 1000’s of SRS
  – Coordinate system transformations applied transparently during indexing and query
• Geometry Types
  – Support OGC Simple Features geometry types
    • Point, Line, Polygon
    • Multi-Point, Multi-Line, Multi-Polygon
    • Geometry Collection
  – Up to 500,000 vertices per Geometry
GeoSPARQL – New 12.2 Features

• New utility functions
• Support for EPSG SRID URIs
• Revised Geometry Storage Scheme – big performance gain
• SDO_JOIN
• Spatial Aggregates
• 3D Support
Integrated Cloud
Applications & Platform Services