Developing GeoSPARQL Applications with Oracle Spatial and Graph

Matthew Perry
Ana Estrada
Souri Das
Jay Banerjee

First Joint International Workshop on Semantic Sensor Networks and Terra Cognita
SSN-TC 2015
October 11, 2015
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.
Agenda

1. GeoSPARQL Overview
2. Implementation in Oracle
3. Demonstration with Oracle 12c
Agenda

1. GeoSPARQL Overview
2. Implementation in Oracle
3. Demonstration with Oracle 12c
OGC GeoSPARQL

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

RDF Data

```sparql
: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 :baths "1.5"^^xsd:decimal .
```

SPARQL Query

```sparql
SELECT ?r ?ba ?br
WHERE {
  ?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>
**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**

```sparql
SELECT ?r ?ba ?br
WHERE {
  ?r :bedrooms ?br
  FILTER (?ba > 2 )
}
```

---

**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>
</tbody>
</table>
Spatial SPARQL QUERY

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)) }
```
Details of ogc:WTKLiteral

All RDFS Literals of type ogc:wktLiteral shall consist of an optional IRI identifying the spatial reference system followed by Simple Features Well Known Text (WKT) describing a geometric value [ISO 19125-1].

"<http://www.opengis.net/def/crs/OGC/1.3/CRS84> POINT(-122.4192 37.7793)"^^ogc:wktLiteral

European Petroleum Survey Group (EPSG) maintains a set of CRS identifiers.
GeoSPARQL Spatial Function Library

• Topological Relations

• Distance-based Operations
  - `ogcf:distance`, `ogcf:buffer`

• Geometry Operations
  - `ogcf:boundary`, `ogcf:convexHull`, `ogcf:envelope`, `ogcf:getSRID`,

• Geometry-Geometry Operations
  - `ogcf:difference`, `ogcf:intersection`, `ogcf:symDifference`, `ogcf:union`
Implementations
Agenda

1. GeoSPARQL Overview
2. Implementation in Oracle
3. Demonstration with Oracle 12c
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)
OGC wktLiteral Datatype

- Optional leading Spatial Reference System URI followed by OGC WKT geometry string.
  <http://xmlns.oracle.com/rdf/geo/srid/{srid}>

- WGS 84 Longitude, Latitude is the default SRS (assumed if SRS URI is absent)

```
SRS: WGS84 Longitude, Latitude
"POINT(-122.4192 37.7793)"^^ogc:wktLiteral

SRS: NAD27 Longitude, Latitude
"<http://xmlns.oracle.com/rdf/geo/srid/8260>
  POINT(-122.4181 37.7793)"^^ogc:wktLiteral
```
What Types of Spatial Data are Supported?

- **Spatial Reference Systems**
  - Built-in support for 1000’s of SRS
  - Plus you can define your own
  - 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
Spatial Function Library

Standard OGC functions

• Topological Relations

• Distance-based Operations
  – ogcf:distance, ogcf:buffer

• Geometry Operations
  – ogcf:boundary, ogcf:convexHull, ogcf:envelope, ogcf:getSRID,

• Geometry-Geometry Operations
  – ogcf:difference, ogcf:intersection, ogcf:symDifference, ogcf:union
Spatial Function Library

Oracle Extensions

• Topological Relations
  – `orageo:relate`

• Distance-based Operations
  – `orageo:distance`, `orageo:withinDistance`, `orageo:buffer`, `orageo:nearestNeighbor`

• Geometry Operations
  – `orageo:area`, `orageo:length`
  – `orageo:centroid`, `orageo:mbr`, `orageo:convexHull`

• Geometry-Geometry Operations
  – `orageo:intersection`, `orageo:union`, `orageo:difference`, `orageo:xor`
SPARQL and SPARQL in SQL Query Architecture

HTTP
- Standard SPARQL Endpoint
  Enhanced with query management control

Java
- Adapter for Apache Jena

Oracle
- SEM_MATCH
- SPARQL-to-SQL Core Logic
Data Storage Tables

**Unique key: PCSGM**

<table>
<thead>
<tr>
<th>S (subj id)</th>
<th>P (pred id)</th>
<th>C (c. obj id)</th>
<th>G (graph id)</th>
<th>M (model id)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>300</td>
<td>200</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Unique key: Vpfx-Vtyp-Vsfx-Lit-Lang**

<table>
<thead>
<tr>
<th>val. id</th>
<th>vname_prefix</th>
<th>vname_suffix</th>
<th>value_type</th>
<th>Literal type</th>
<th>Lang type</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td><a href="http://xyz.com/">http://xyz.com/</a></td>
<td>geom1</td>
<td>UR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>POINT(...)</td>
<td>asWKT</td>
<td>TL</td>
<td></td>
<td>ogc:wktLiteral</td>
</tr>
<tr>
<td>300</td>
<td><a href="http://opengis">http://opengis</a>..</td>
<td></td>
<td>UR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Spatial Index on ogc:wktLiteral Datatype

• Prepare for spatial querying by creating a spatial index for the ogc:wktLiteral datatype

```sql
SQL> exec sem_apis.add_datatype_index(
    'http://www.opengis.net/ont/geosparql#wktLiteral',
    options=>'TOLERANCE=0.1 SRID=8307
             DIMENSIONS=((LONGITUDE,-180,180) (LATITUDE,-90,90))
);
```
Helper function to generate SDO_GEOMETRY objects

FUNCTION getV$GeometryVal (  
    value_type IN VARCHAR2,  
    vname_prefix IN VARCHAR2,  
    vname_suffix IN VARCHAR2,  
    literal_type IN VARCHAR2,  
    language_type IN VARCHAR2,  
    srid IN NUMBER  
) RETURN MDSYS.SDO_GEOMETRY

<table>
<thead>
<tr>
<th>val. id</th>
<th>vname_prefix</th>
<th>vname_suffix</th>
<th>value_type</th>
<th>Literal type</th>
<th>Lang type</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td><a href="http://xyz.com/">http://xyz.com/</a></td>
<td>geom1</td>
<td>UR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>POINT(...)</td>
<td>asWKT</td>
<td>TL</td>
<td></td>
<td>ogc:wktLiteral</td>
</tr>
<tr>
<td>300</td>
<td><a href="http://opengis">http://opengis</a>..</td>
<td></td>
<td>UR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Spatial Indexing

ogcf:sfOverlaps(?geom, "POLYGON((...))")

1) Primary Filter

2) Secondary Filter

Function-based R-Tree Index (BBox)

getV$GeometryVal(value_type, vname_prefix, ...)

<table>
<thead>
<tr>
<th>val. id</th>
<th>vname_prefix</th>
<th>vname_suffix</th>
<th>value_type</th>
<th>Literal type</th>
<th>Lang type</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td><a href="http://xyz.com/">http://xyz.com/</a></td>
<td>geom1</td>
<td>UR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>POINT(...)</td>
<td>asWKT</td>
<td>TL</td>
<td></td>
<td>ogc:wktLiteral</td>
</tr>
<tr>
<td>300</td>
<td><a href="http://opengis">http://opengis</a>...</td>
<td></td>
<td>UR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Spatial Query Execution

SELECT ?s ?l ?wkt
WHERE {
  ?s rdf:type lgd:Monument .
  ?s rdfs:label ?l .
  ?s geovocab:geometry ?geom .
FILTER(ogcf:sfWithin(?wkt, "POLYGON((-71.44 42.50, -71.42 42.40, -71.08 42.39, -71.03 42.56, -71.44 42.50))")^^ogc:wktLiteral))"
Agenda

1. GeoSPARQL Overview
2. Implementation in Oracle
3. Demonstration with Oracle 12c
Demonstration with Oracle 12c

Oracle Big Data Lite VM
+ Spatial & Graph Support
+ Map Viewer

HistoricThing.node
4.1 Million Triples
268K Points (WGS 84)

1. Create semantic model
2. Bulk load into model
3. Create spatial index
4. Execute GeoSPARQL queries
5. Create Map Viewer layer
Integrated Cloud
Applications & Platform Services