Navigating the Earth with pure SPARQL

Damien Graux
Inria (France)
damien.graux@inria.fr

First, an example with shipwrecks

Sunken boats

Sunken boats

The wreck set can be modeled like so using RDF:

```
# Declaring a wreck, having
:wreckId :type      :wreck .

# 1. its Cartesian coordinates
:wreckId :abscissa "XXX"    .
:wreckId :ordinate "YYY"    .

# 2. the discovery year
:wreckId :foundIn  "year"   .

# 3. the associated C14 ratio
:wreckId :c14rate  "ratio" .
```

Sunken boats

Typical wreck record

```
:wreckId :type     :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Let’s consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (`Px`,`Py`);
c. older than 1000 years.

Sunken boats

Typical wreck record

```
:wreckId :type     :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Using SPARQL, to list all the wrecks

```
SELECT ?f WHERE {
     ?f :type :wreck .
}
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (`Px`,`Py`);
c. older than 1000 years.

Sunken boats

Typical wreck record

```
:wreckId :type      :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Using SPARQL, to list all the wrecks with a.

```
SELECT ?f WHERE {
     ?f :type :wreck .
     ?f :foundIn ?Y .
     FILTER( (2022-?Y) <= 10 )
}
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (Px,Py);
c. older than 1000 years.

Sunken boats

Typical wreck record

```
:wreckId :type     :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (Px,Py);
c. older than 1000 years.

Using SPARQL, to list all the wrecks with **a**, **b**.

```
SELECT ?f WHERE {
     ?f :type :wreck .
     ?f :foundIn ?Y .
     FILTER( (2022-?Y) <= 10 )
     ?f :abscissa ?x . ?f :ordinate ?y .
     FILTER( ( (?x-Px)*(?x-Px) +
               (?y-Py)*(?y-Py) ) <= 100*100 )
}
```

Using the formula $d^2=(\Delta x^2+\Delta y^2)$

Sunken boats

Typical wreck record

```
:wreckId :type      :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (Px,Py);
c. older than 1000 years.

Dating the wrecks requires using the ^{14}C-ratio,

$$t(r) = \left(\frac{\ln(r)}{-0.693}\right) \cdot t_{1/2}$$

in particular, the Carbon 14 has a half-life of 5700 years.

Problem: the formula involves a logarithm!

Sunken boats

Typical wreck record

```
:wreckId :type      :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (Px,Py);
c. older than 1000 years.

Dating the wrecks requires using the ^{14}C-ratio,

$$t(r) = \left(\frac{\ln(r)}{-0.693}\right) \cdot t_{1/2}$$

in particular, the Carbon 14 has a half-life of 5700 years.

<u>Problem:</u> the formula involves a logarithm!

$\Rightarrow$ Approximating it thanks to a decomposition in series

$$\forall y \in]0, +\infty[, \;\; \ln(y) = 2 \sum_{k=0}^{+\infty} \frac{1}{2k+1} \left(\frac{y-1}{y+1}\right)^{2k+1}$$

Sunken boats

Typical wreck record

```
:wreckId :type     :wreck .
:wreckId :abscissa "XXX"   .
:wreckId :ordinate "YYY"   .
:wreckId :foundIn  "year"  .
:wreckId :c14rate  "ratio" .
```

Let's consider the following conditions:

a. found in the last 10 years;
b. 100km around a specific position (Px,Py);
c. older than 1000 years.

Using SPARQL, to list all the wrecks with **a**, **b** and **c**.

```
SELECT ?f WHERE {
     ?f :type :wreck .
     ?f :foundIn ?Y .
     FILTER( (2022-?Y) <= 10 )
     ?f :abscissa ?x . ?f :ordinate ?y .
     FILTER( ( (?x-Px)*(?x-Px) +
                (?y-Py)*(?y-Py) ) <= 100*100 )
     ?f :c14rate ?rate .
     BIND(( (?rate-1)/(?rate+1) ) AS ?z )
     BIND(( ?z ) AS ?t0 )
     BIND(( (1/3)*(?z*?z*?z) ) AS ?t1 )
     BIND(( (1/5)*(?z*?z*?z*?z*?z)) AS ?t2 )
     BIND(( 2*(?t0 + ?t1 + ?t2) ) AS ?LOG )
     FILTER(5700*?LOG/(-0.693)<=1000)
}
```

Using bindings, and considering the series first three terms

Sunken boats: ...a simplified example

The previous example has been simplified for the sake of clarity,

1. The series approximation should indeed involve more terms.
2. Latitude and longitude coordinates are usually preferred.

   For P1(lat1, lon1) and P2(lat2, lon2), the Haversine formula should be used:

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 . \cos\varphi_2 . \sin^2\left(\frac{\Delta\lambda}{2}\right)$$
$$c = 2\ .\ \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right)$$
$$d = R\ .\ c$$

φ latitude in rad: $\frac{lat.\pi}{180}$
λ longitude in rad: $\frac{lon.\pi}{180}$
R the Earth radius: 6 371 km

⇒ The query designer would have to write multiple decompositions in series!

Focus of this study

On a planet

The previous example has been simplified for the sake of clarity,

1. The series approximation should indeed involve more terms.
2. Latitude and longitude coordinates are usually preferred.

   For P1(lat1, lon1) and P2(lat2, lon2), the Haversine formula should be used:

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 . \cos\varphi_2 . \sin^2\left(\frac{\Delta\lambda}{2}\right)$$
$$c = 2 \,.\, \operatorname{atan2}\left(\sqrt{a}, \sqrt{1-a}\right)$$
$$d = R \,.\, c$$

φ latitude in rad: $\frac{lat.\pi}{180}$

λ longitude in rad: $\frac{lon.\pi}{180}$

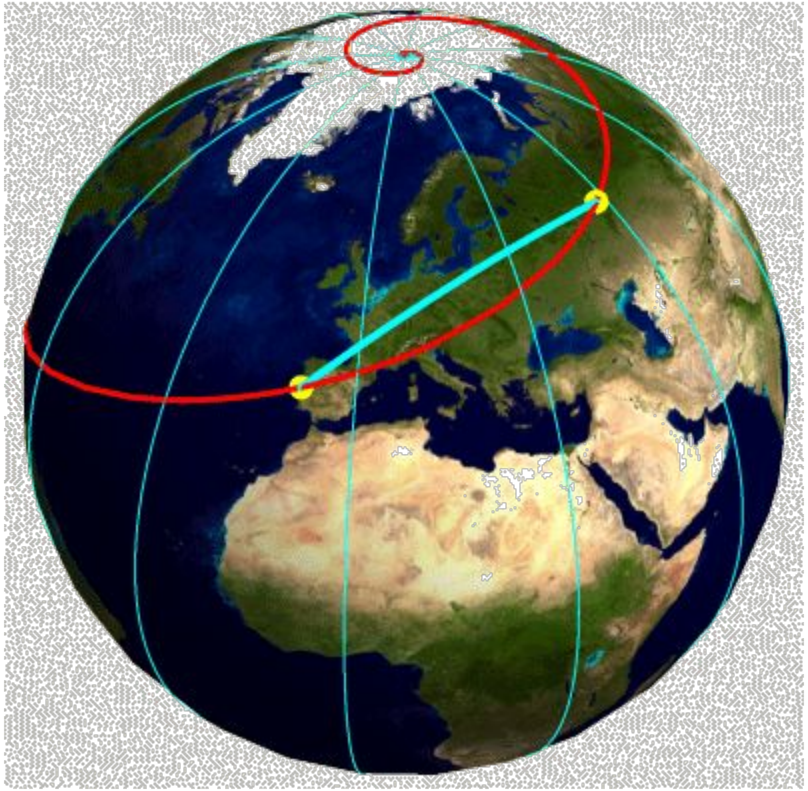
R the Earth radius: 6 371 km

$\Rightarrow$ The query designer would have to write multiple decompositions in series!

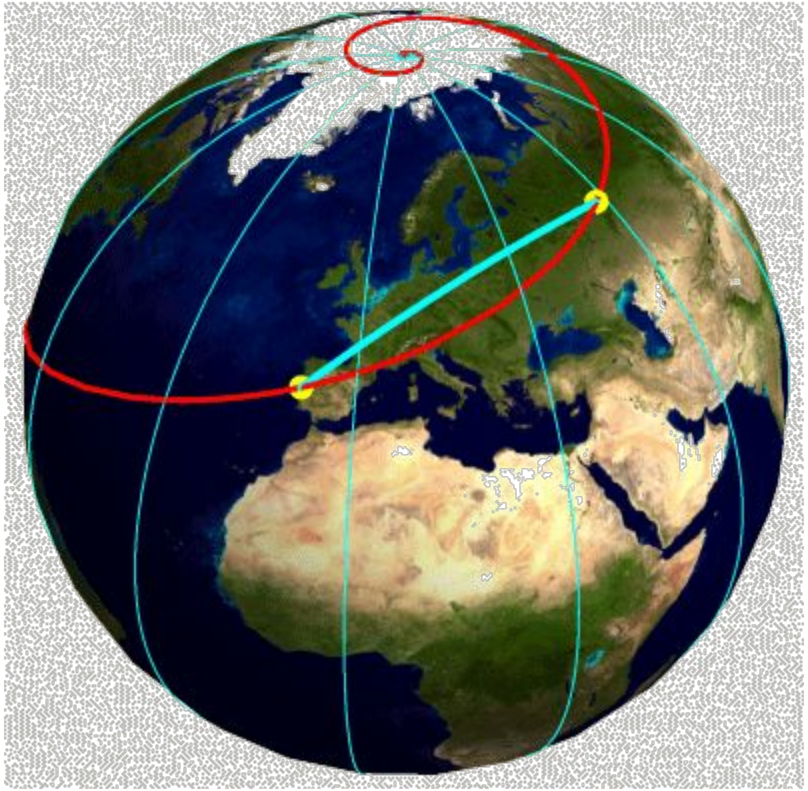
Earth surface geometry

Why are plane routes not straight lines on a map?

The Earth isn't flat!



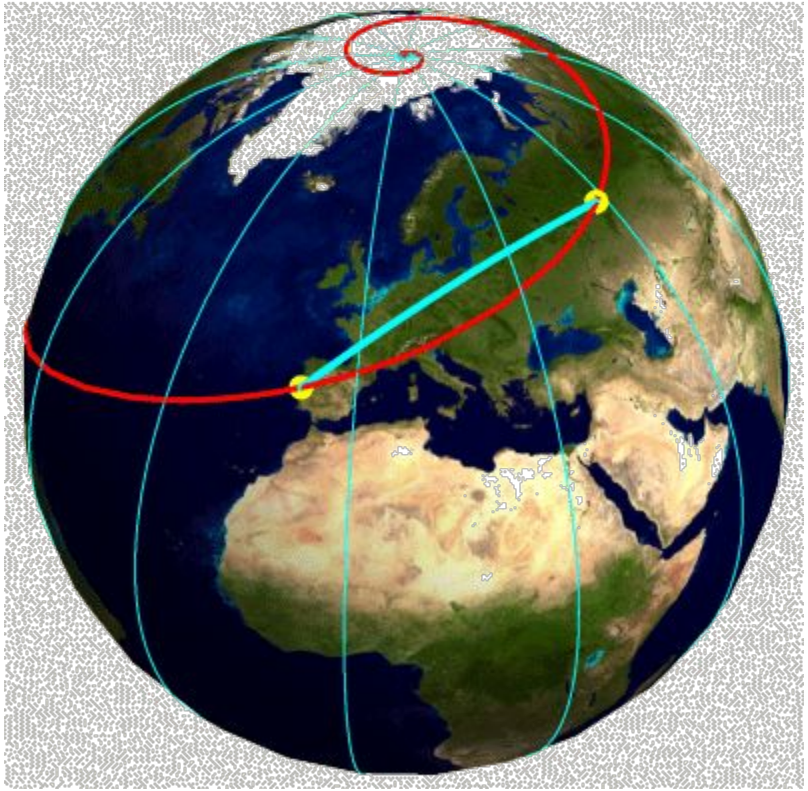
The Earth isn't flat!



Great-circle (in blue):

the intersection of the sphere and a plane that passes through the center point of the sphere.

The Earth isn't flat!



Great-circle (in blue):

the intersection of the sphere and a plane that passes through the center point of the sphere.

Rhumb line (in red):

an arc crossing all meridians of longitude at the same angle, *i.e.* constant bearing.

How to set a route using SPARQL 1.1?

Current literature approach

- Providing built-in functions, *e.g.*:
  - Virtuoso `bif: <http://www.openlinksw.com/schemas/bif#>`
  - GraphDB `f: <http://www.ontotext.com/sparql/functions/>`
  - Jena `math: <http://www.w3.org/2005/xpath-functions/math#>`
- Working group for next SPARQL version to add math functions in the standard

Current literature approach

- Providing built-in functions, *e.g.*:
  - Virtuoso bif: <http://www.openlinksw.com/schemas/bif#>
  - GraphDB f: <http://www.ontotext.com/sparql/functions/>
  - Jena math: <http://www.w3.org/2005/xpath-functions/math#>
- Working group for next SPARQL version to add math functions in the standard

Problems

- lack of interoperability between engines → with built-in functions
- complex for query writers
- impossibility to have complex math formulae

→ with manual writing

GeoSPARQL

```
PREFIX geosparql:  <http://www.opengis.net/ont/geosparql#>
PREFIX geof:     <http://www.opengis.net/def/function/geosparql/>
PREFIX uom:      <http://www.opengis.net/def/uom/OGC/1.0/>

SELECT ?label ?lat ?long ?coordinates WHERE {
   ?x rdfs:label ?label ;
      geosparql:hasGeometry [ geosparql:asWKT ?coordinates];
      geo:lat ?lat; geo:long ?long .

   BIND ("Point(0.1413499 45.1423348)"^^geosparql:wktLiteral
    AS ?Currentposition)
   BIND (geof:distance(?coordinates, ?Currentposition, uom:metre)
    AS ?distance)
}
ORDER BY ?distance LIMIT 1
```

How to set a route using <u>**exclusively**</u> SPARQL 1.1?

W3C Recommendation

W3C®

SPARQL Query Language for RDF

W3C Recommendation 15 January 2008

New Version Available: SPARQL 1.1 (Document Status Update, 26 March 2013)

The SPARQL Working Group has produced a W3C Recommendation for a new version of SPARQL which adds features to this 2008 version. Please see SPARQL 1.1 Overview for an introduction to SPARQL 1.1 and a guide to the SPARQL 1.1 document set.

This version:
http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/
Latest version:
http://www.w3.org/TR/rdf-sparql-query/
Previous version:
http://www.w3.org/TR/2007/PR-rdf-sparql-query-20071112/
Editors:
Eric Prud'hommeaux, W3C <eric@w3.org>
Andy Seaborne, Hewlett-Packard Laboratories, Bristol <andy.seaborne@hp.com>

W3C Recommendation

W3C

SPARQL Query Language for RDF

W3C Recommendation 15 January 2008

New Version Available: SPARQL 1.1 (Document Sta

The SPARQL Working Group has produced a W3C Recom
SPARQL which adds features to this 2008 version. Please
introduction to SPARQL 1.1 and a guide to the SPARQL 1.

This version:
http://www.w3.org/TR/2008/REC-rdf-sparql-query-2008
Latest version:
http://www.w3.org/TR/rdf-sparql-query/
Previous version:
http://www.w3.org/TR/2007/PR-rdf-sparql-query-20071
Editors:
Eric Prud'hommeaux, W3C <eric@w3.org>
Andy Seaborne, Hewlett-Packard Laboratories, Bristol

http://www.w3.org/TR/sparql11-query/

W3C Recommendation

W3C

SPARQL 1.1 Query Language

W3C Recommendation 21 March 2013

This version:
http://www.w3.org/TR/2013/REC-sparql11-query-20130321/
Latest version:
http://www.w3.org/TR/sparql11-query/
Previous version:
http://www.w3.org/TR/2012/PR-sparql11-query-20121108/
Editors:
Steve Harris, Garlik, a part of Experian
Andy Seaborne, The Apache Software Foundation
Previous Editor:
Eric Prud'hommeaux, W3C

Please refer to the errata for this document, which may include some normative corrections.

See also translations.

http://www.w3.org/TR/sparql11-query/

SPARQL 1.1 adds the variable assignments:
- **VALUES ?x { … }**
- **BIND (… AS ?y)**

W3C Recommendation

W3C

SPARQL Query Language for RDF

W3C Recommendation 15 January 2008

New Version Available: SPARQL 1.1 (Document Sta…

The SPARQL Working Group has produced a W3C Recom… SPARQL which adds features to this 2008 version. Please … introduction to SPARQL 1.1 and a guide to the SPARQL 1…

This version:
http://www.w3.org/TR/2008/REC-rdf-sparql-query-20…
Latest version:
http://www.w3.org/TR/rdf-sparql-query/
Previous version:
http://www.w3.org/TR/2007/PR-rdf-sparql-query-20071…
Editors:
Eric Prud'hommeaux, W3C <eric@w3.org>
Andy Seaborne, Hewlett-Packard Laboratories, Bristol

W3C Recommendation

W3C

SPARQL 1.1 Query Language

W3C Recommendation 21 March 2013

This version:
http://www.w3.org/TR/2013/REC-sparql11-query-20130321/
Latest version:
http://www.w3.org/TR/sparql11-query/
Previous version:
http://www.w3.org/TR/2012/PR-sparql11-query-20121108/
Editors:
Steve Harris, Garlik, a part of Experian
Andy Seaborne, The Apache Software Foundation
Previous Editor:
Eric Prud'hommeaux, W3C

Please refer to the errata for this document, which may include some normative corrections.

See also translations.

Using `BIND` to carry Earth constants

```
# Useful variables.
BIND ( xsd:double("3.14159265359") AS ?PI  ) # π with 11 digits.
BIND ( xsd:double("6.28318530718") AS ?2PI ) # 2π with 11 digits.
BIND ( xsd:double("6371") AS ?E_radius )     # Earth's radius, in km.
```

Using `BIND` to convert degrees to radians

```
# Useful variables.
BIND ( xsd:double("3.14159265359") AS ?PI  ) # π with 11 digits.
BIND ( xsd:double("6.28318530718") AS ?2PI ) # 2π with 11 digits.
BIND ( xsd:double("6371") AS ?E_radius )     # Earth's radius, in km.


# Degrees to radians.
BIND ( (xsd:double(?lat) * ?PI/180) AS ?lar )
```

Using `BIND` to compute *e.g.* intermediate results

```
# Useful variables.
BIND ( xsd:double("3.14159265359") AS ?PI  ) # π with 11 digits.
BIND ( xsd:double("6.28318530718") AS ?2PI ) # 2π with 11 digits.
BIND ( xsd:double("6371") AS ?E_radius )     # Earth's radius, in km.


# Degrees to radians.
BIND ( (xsd:double(?lat) * ?PI/180) AS ?lar )

# Having two pairs of coordinates, below are the deltas in radians.
BIND ( ((xsd:double(?lat2)-xsd:double(?lat1)) * ?PI/180) AS ?dellar )
BIND ( ((xsd:double(?lon2)-xsd:double(?lon1)) * ?PI/180) AS ?dellor )
```

General Strategy

1. Set up usual constants;
2. Use bindings to express the mathematical expressions;
3. Approximate with series when trigonometric or exponential functions are required;
4. Wrap binding sets into standalone blocs;
5. Make them available.

General Strategy

1. Set up usual constants;
2. Use bindings to express the mathematical expressions;
3. Approximate with series when trigonometric or exponential functions are required;
4. Wrap binding sets into standalone blocs;
5. Make them available.

Relying and extending **MINDS [1]**

--
[1] Graux, Damien, et al. "MINDS: a translator to embed mathematical expressions inside SPARQL queries." SEMANTiCS. Springer, Cham, 2020.

Extending MINDS to ease the query development

MINDS helps to write mathematical expressions in *pure* SPARQL:

- Translates mathematical expressions into a list of bindings;
- Obtained queries can be executed by any evaluator;
- Easing the query design with a python interface.

https://github.com/SmartDataAnalytics/minds

Extending MINDS to ease the query development

For instance, using an exponential function in a complex formula $x^2+e^{(y+3z)}$

```
#math2sparql > ?X**2 + exp (?Y + 3 * ?Z)
```

Extending MINDS to ease the query development

For instance, using an exponential function in a complex formula $x^2+e^{(y+3z)}$

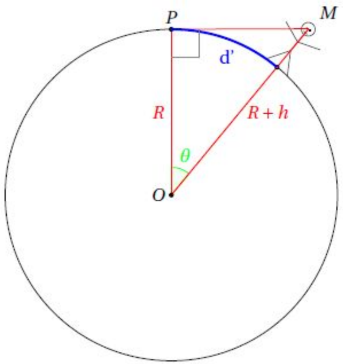
```
#math2sparql > ?X**2 + exp (?Y + 3 * ?Z)
BIND ((0+(1)/1.0                                        # 1
  +(1*(xsd:double(?Y)+3*xsd:double(?Z)))/1.0           # y + 3z
  +(1*(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z)))/2.0           # (y+3z)^2/2!
  +(1*(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z)))/6.0           # (y+3z)^3/3!
  +(1*(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z))
     *(xsd:double(?Y)+3*xsd:double(?Z)))/24.0          # (y+3z)^4/4!
)AS ?sub1)
BIND ( ( FLOOR((
    (1*xsd:double(?X)*xsd:double(?X)) +?sub1           # x^2 + sub_1
  )*100)/100 ) AS ?result )
```

Considering the 5 first terms of:

$$\exp x = \sum_{k=0}^{+\infty} \frac{x^k}{k!}$$

Example: distance from the horizon to a crow's nest

Example: distance from the horizon to a crow's nest

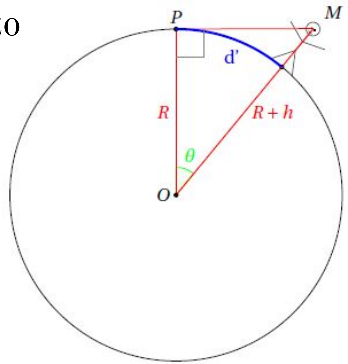


Example: distance from the horizon to a crow’s nest

At a height h above the ground, the distance to the horizon d, is given by:

$$\mathbf{d = \sqrt{(2 * R * h/b)}}$$

with b = 0.8279 a factor for atmospheric refraction.

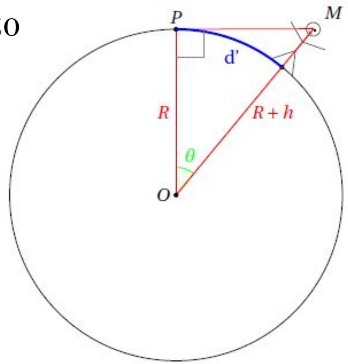


Example: distance from the horizon to a crow's nest

At a height h above the ground, the distance to the horizon d, is given by:

$$\mathbf{d = \sqrt{(2 * R * h/b)}}$$

with b = 0.8279 a factor for atmospheric refraction.




```
BIND ( "0.8279" AS ?b )
BIND ( (2*xsd:double(?E_radius)*xsd:double(?h)/xsd:double(?b)) AS ?int )
BIND ((0+(1*(((?int)-1)/((?int)+1)))/1.0
+(1*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1)))/3.0
+(1*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*
(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1)))/5.0
+(1*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*
(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1))*
(((?int)-1)/((?int)+1))*(((?int)-1)/((?int)+1)))/7.0
)AS ?sub1)
BIND ((0+(1)/1.0+(1*?sub1)/1.0+(1*?sub1*?sub1)/2.0 + (1*?sub1*?sub1*?sub1)/6.0 )AS ?sub2)
BIND ( ( FLOOR((?sub2)*10000)/10000 ) AS ?distance )
```

Sharing the bindings

```
##############  BEGINNING OF GENERATED SPARQL BINDINGS  ##############

BIND ((0+1*(1)/1.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/2.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/24.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/720.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/40320.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/3628800.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/479001600.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/87178291200.0
)AS ?Xsub1)

BIND ( ( FLOOR(( (1*(?dellor*?Xsub1)*(?dellor*?Xsub1)) )*100000)/100000 ) AS ?X2 )

BIND ((0+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/1.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/3.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/5.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/7.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/9.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/11.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+
(1*(?dellar)*(?dellar)) )+1)))/13.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+
(1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/15.0
)AS ?sub1)

BIND ((0+(1)/1.0
+(1*?sub1)/1.0
+(1*?sub1*?sub1)/2.0
+(1*?sub1*?sub1*?sub1)/6.0
+(1*?sub1*?sub1*?sub1*?sub1)/24.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1)/120.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/720.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/5040.0
)AS ?sub2)

BIND ( ( FLOOR((?E_radius*?sub2)*100000)/100000 ) AS ?distance )


### END OF BINDINGS
```

```
##############  BEGINNING OF GENERATED SPARQL BINDINGS  ##############

BIND ((0+1*(1)/1.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/2.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/24.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/720.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/40320.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/3628800.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/479001600.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/87178291200.0
)AS ?Xsub1)

BIND ( ( FLOOR(( (1*(?dellor*?Xsub1)*(?dellor*?Xsub1)) )*100000)/100000 ) AS ?X2 )

BIND ((0+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/1.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/3.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/5.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/7.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/9.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/11.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/13.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/15.0
)AS ?sub1)

BIND ((0+(1)/1.0
+(1*?sub1)/1.0
+(1*?sub1*?sub1)/2.0
+(1*?sub1*?sub1*?sub1)/6.0
+(1*?sub1*?sub1*?sub1*?sub1)/24.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1)/120.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/720.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/5040.0
)AS ?sub2)

BIND ( ( FLOOR((?E_radius*?sub2)*100000)/100000 ) AS ?distance )


### END OF BINDINGS
```

These correspond to the **equirectangular** projection!

```
##############  BEGINNING OF GENERATED SPARQL BINDINGS  ##############

BIND ((0+1*(1)/1.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/2.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/24.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/720.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/40320.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/3628800.0
+1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/479001600.0
+-1*(1* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )*
(((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) )* (((?lar1+?lar2)/2)-?2PI*FLOOR(((?lar1+?lar2)/2)/?2PI) ))/87178291200.0
)AS ?Xsub1)

BIND ( ( FLOOR(( (1*(?dellor*?Xsub1)*(?dellor*?Xsub1)) )*100000)/100000 ) AS ?X2 )

BIND ((0+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/1.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/3.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/5.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/7.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/9.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/11.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+
(1*(?dellar)*(?dellar)) )+1)))/13.0
+(1*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+
(1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*
(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar))
)-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+
(1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1))*(((?X2+ (1*(?dellar)*(?dellar)) )-1)/((?X2+ (1*(?dellar)*(?dellar)) )+1)))/15.0
)AS ?sub1)

BIND ((0+(1)/1.0
+(1*?sub1)/1.0
+(1*?sub1*?sub1)/2.0
+(1*?sub1*?sub1*?sub1)/6.0
+(1*?sub1*?sub1*?sub1*?sub1)/24.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1)/120.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/720.0
+(1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1*?sub1)/5040.0
)AS ?sub2)

BIND ( ( FLOOR((?E_radius*?sub2)*100000)/100000 ) AS ?distance )


### END OF BINDINGS
```

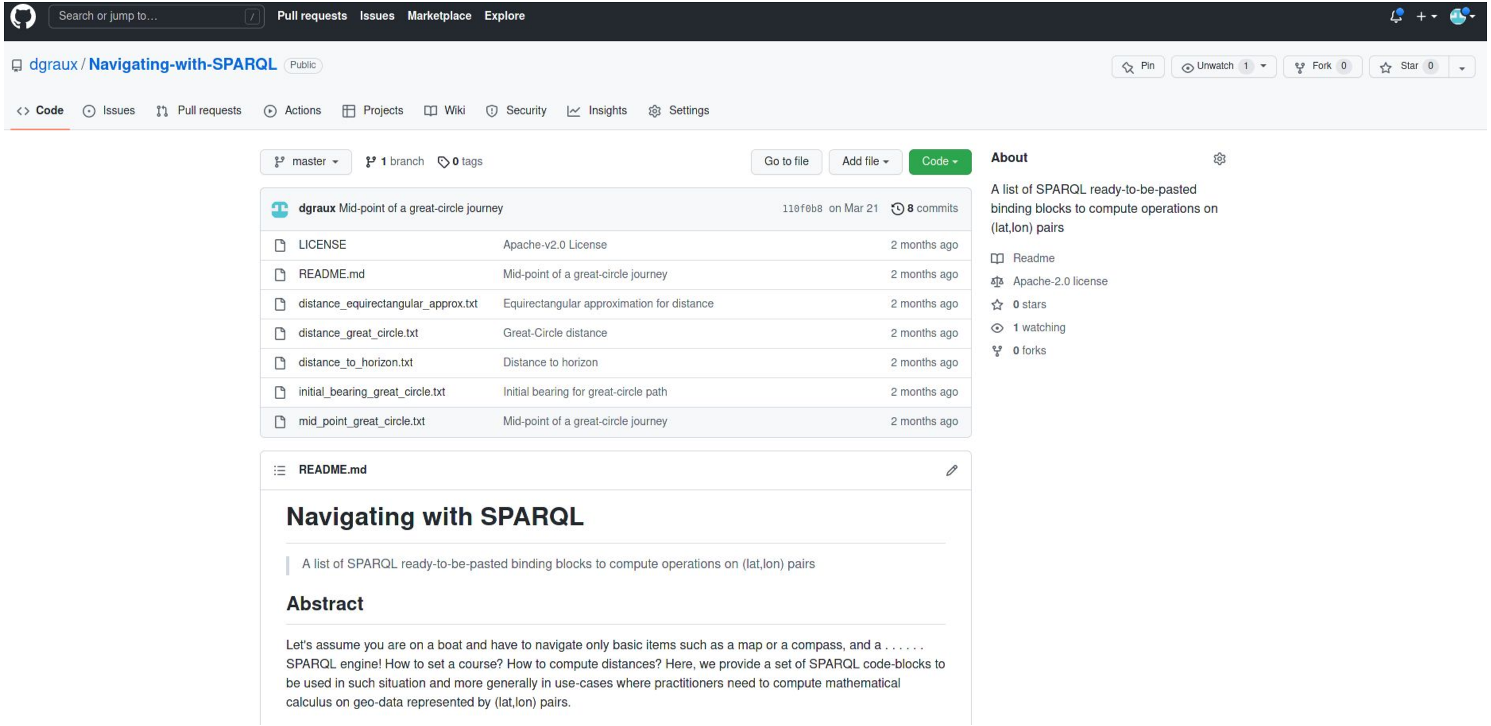
These correspond to the **equirectangular** projection!

$$x = dellor \; . \; \cos\left(\frac{lar_1 + lar_2}{2}\right)$$

$$y = dellar$$

$$d = \text{E\_radius} \; . \; \sqrt{x^2 + y^2}$$

<https://github.com/dgraux/Navigating-with-SPARQL>



Conclusion

Navigating with SPARQL...

- **... possible, but complicated!** 😉
- Binding blocks are available:
  <https://github.com/dgraux/Navigating-with-SPARQL>
- Results to be taken as a coding showcase
  to show the potential of SPARQL 1.1