INDEXING AND SEARCHING RDF DATASETS

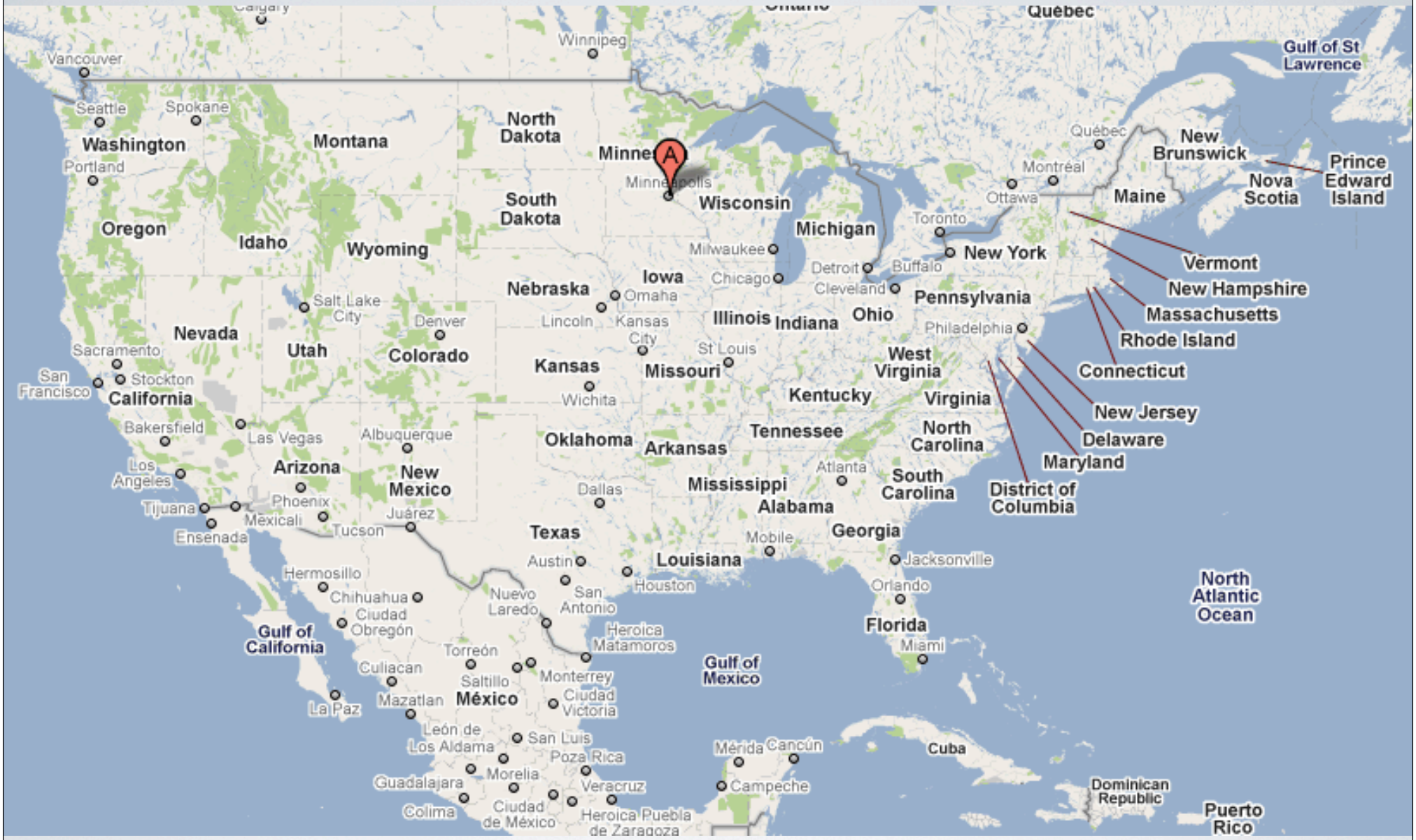
Improving Performance of Semantic Web Applications with Lucene, SIREn and RDF

Mike Hugo
Entagen, LLC

ENTAGEN
ACCELERATING INSIGHT

slides and sample code can be found at

https://github.com/mjhugo/rdf-lucene-siren-presentation



ENTAGEN
ACCELERATING INSIGHT


Groovy
HIBERNATE
JBoss™
GRAILS
Java
Sun Microsystems
ORACLE
Apache
Solr
MySQL®
Lucene
Spring
ENTAGEN
ACCELERATING INSIGHT

ENTAGEN OVERVIEW

- Founded in May 2008
- 17 Team Members located in two onshore offices: Boston & Minneapolis
- Life sciences & biomedical expertise, extensive network and focused technical specialization in enterprise class search, integration and analysis systems
- Custom Software Development
- Proprietary Software Applications: TripleMap, BrightScan, ExaSeq
- Professional Services/Strategic Consulting

ENTAGEN
ACCELERATING INSIGHT

SELECT ENTAGEN CLIENTS

MERCK

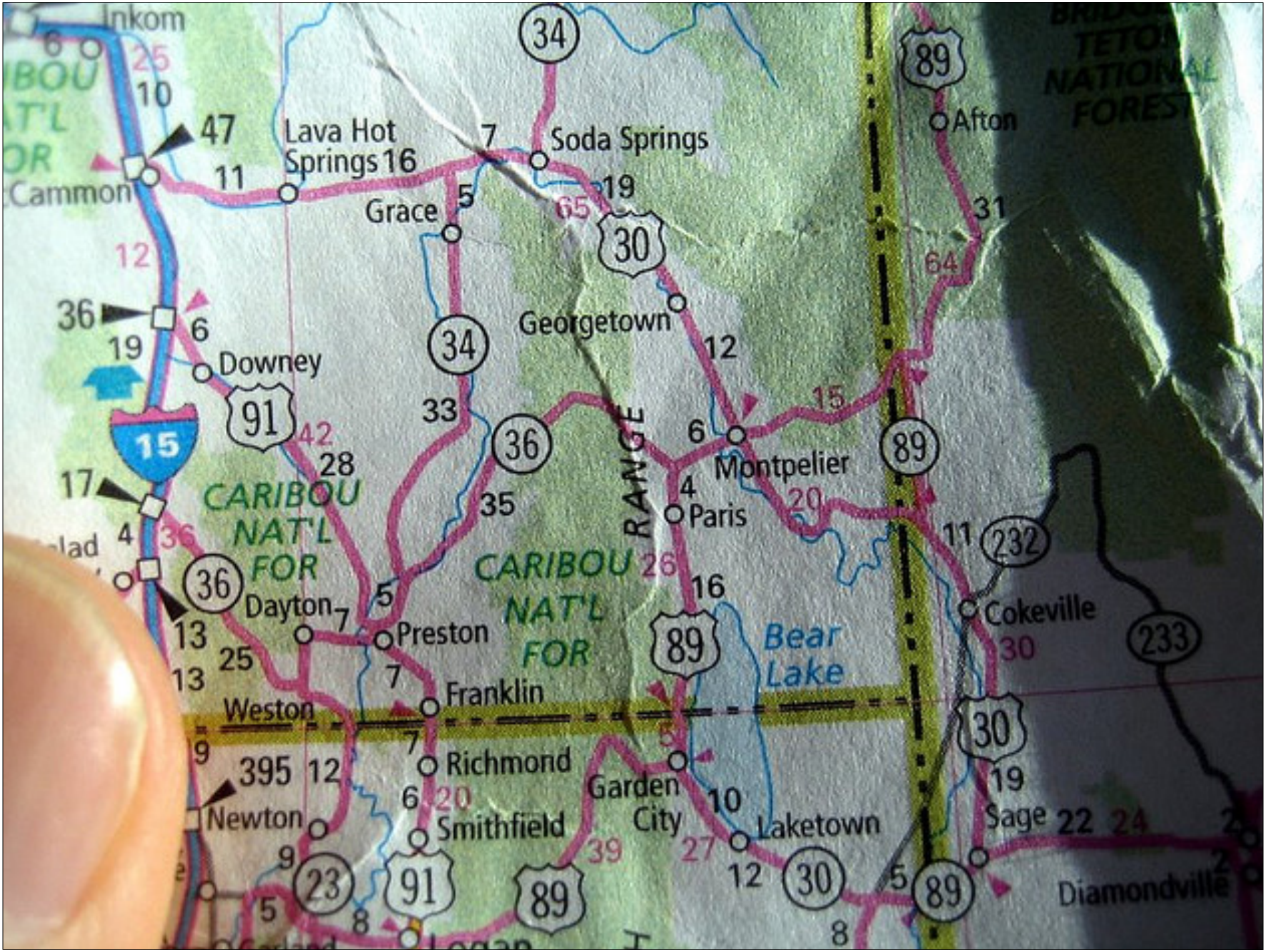
NOVARTIS

Boehringer Ingelheim

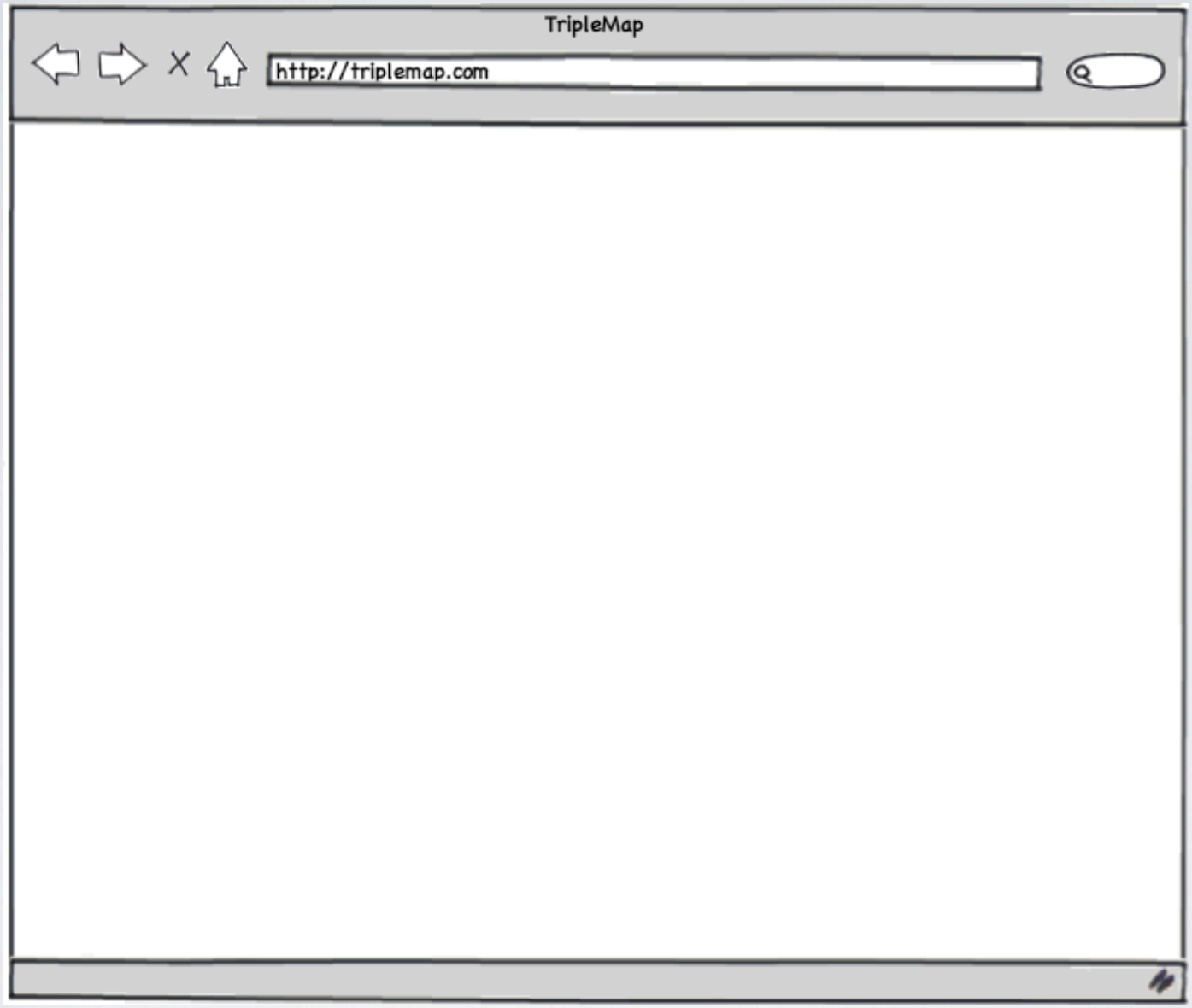
MILLIPORE

DANA-FARBER
CANCER INSTITUTE

AGENDA

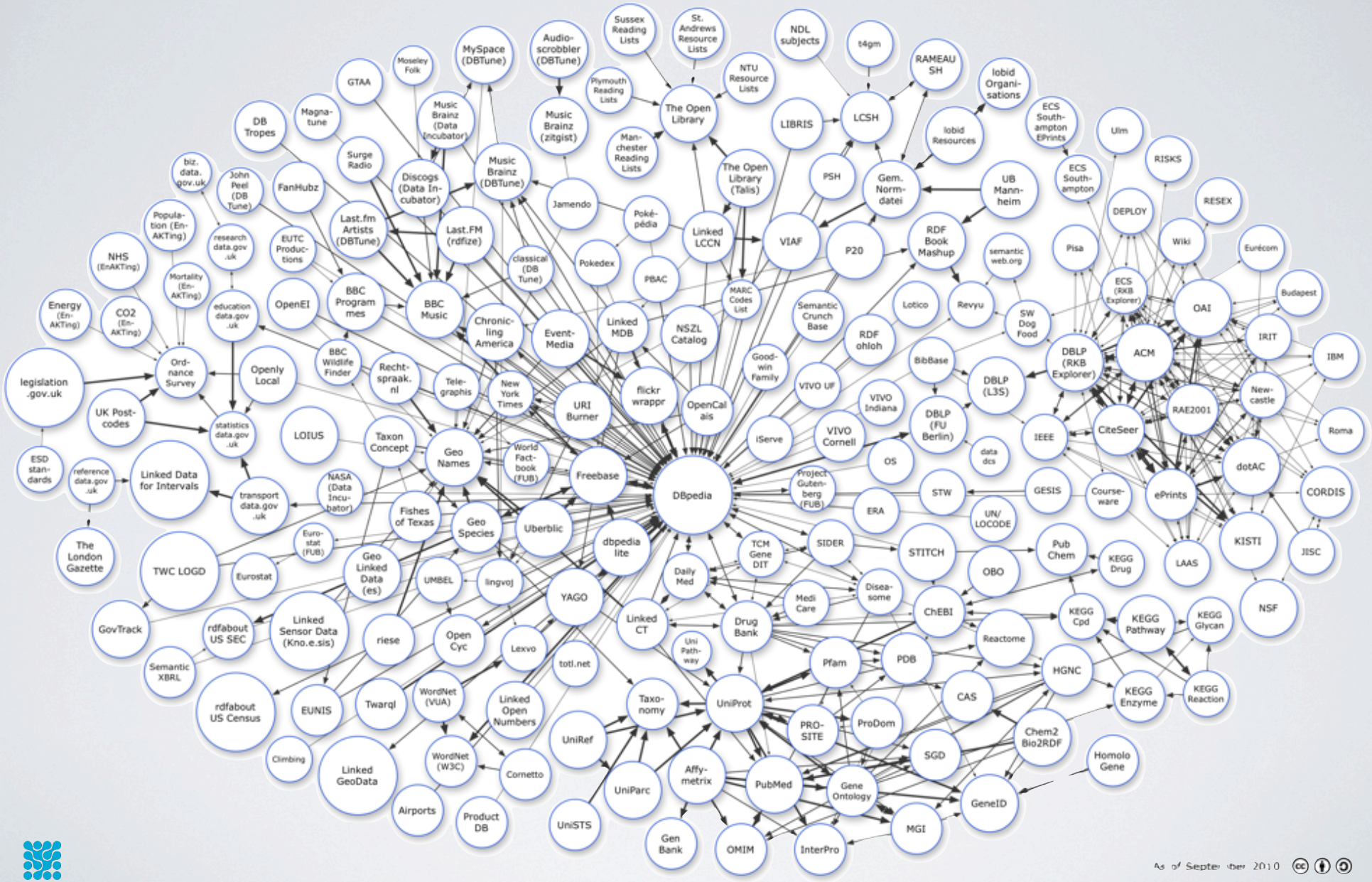

Inkom
34
89
Lava Hot Springs
Soda Springs
Afton
Grace
30
Georgetown
Downey
91
34
36
15
CARIBOU NAT'L FOR
RANGE
Montpelier
Paris
89
232
Dayton
Preston
CARIBOU NAT'L FOR
Bear Lake
Cokeville
233
Weston
Franklin
Richmond
Garden City
Laketown
Sage
Newton
Smithfield
23
91
89
30
Diamondville


SPARQL
LUCENE
SIREN
TripleMap.com
ENTAGEN
ACCELERATING INSIGHT



ENTAGEN
ACCELERATING INSIGHT

LINKING OPEN DATA

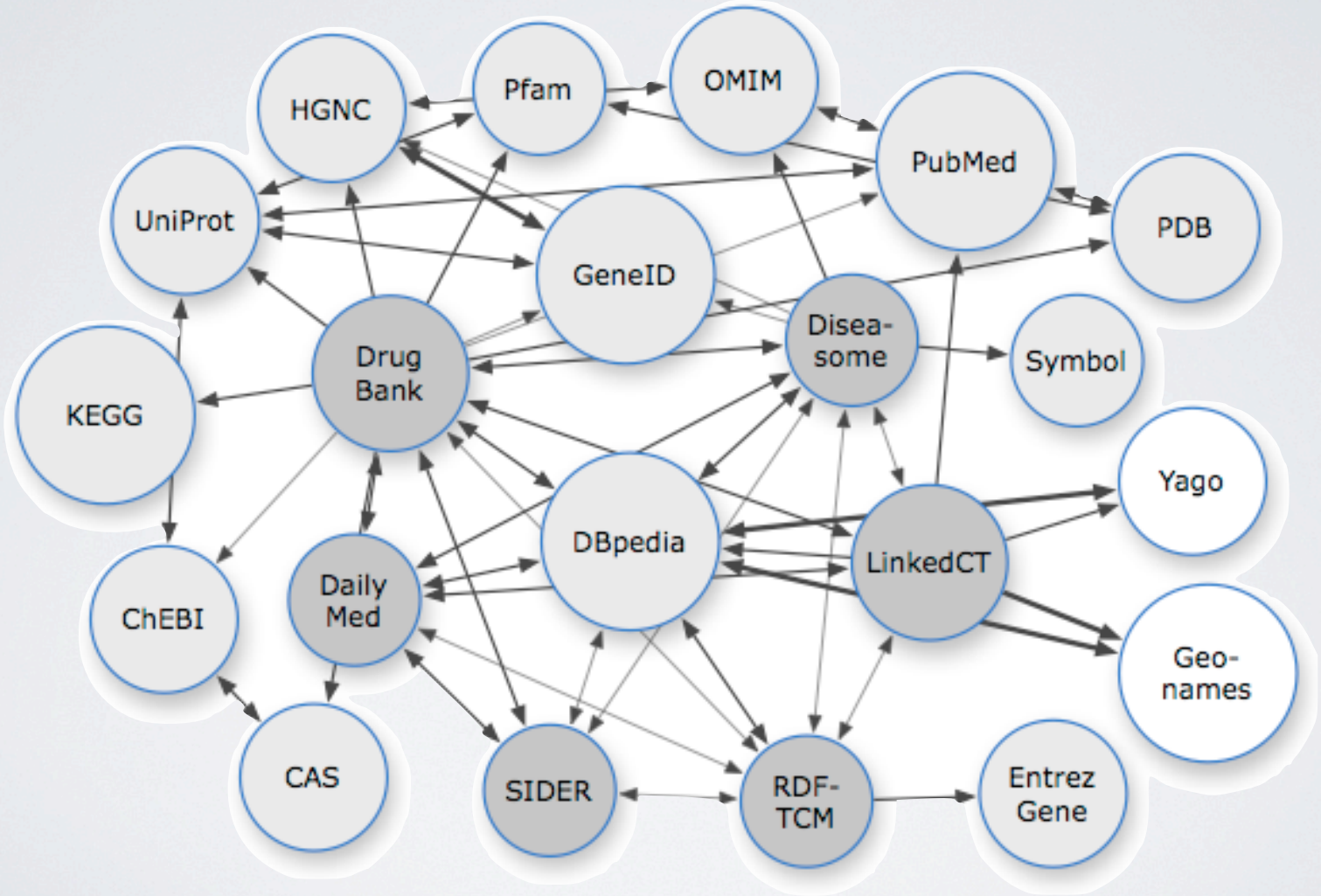


As of September 2010

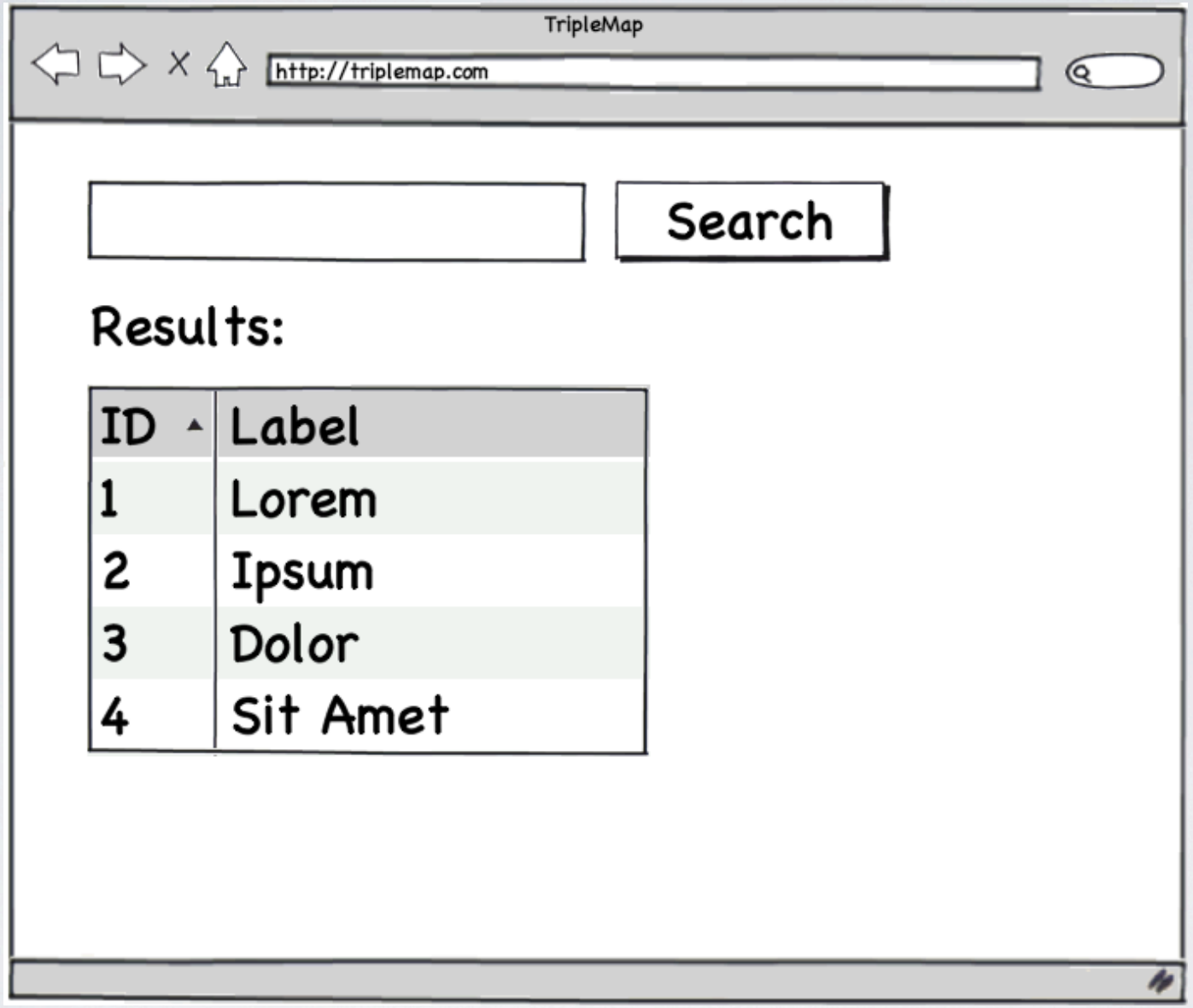
ENTAGEN
ACCELERATING INSIGHT

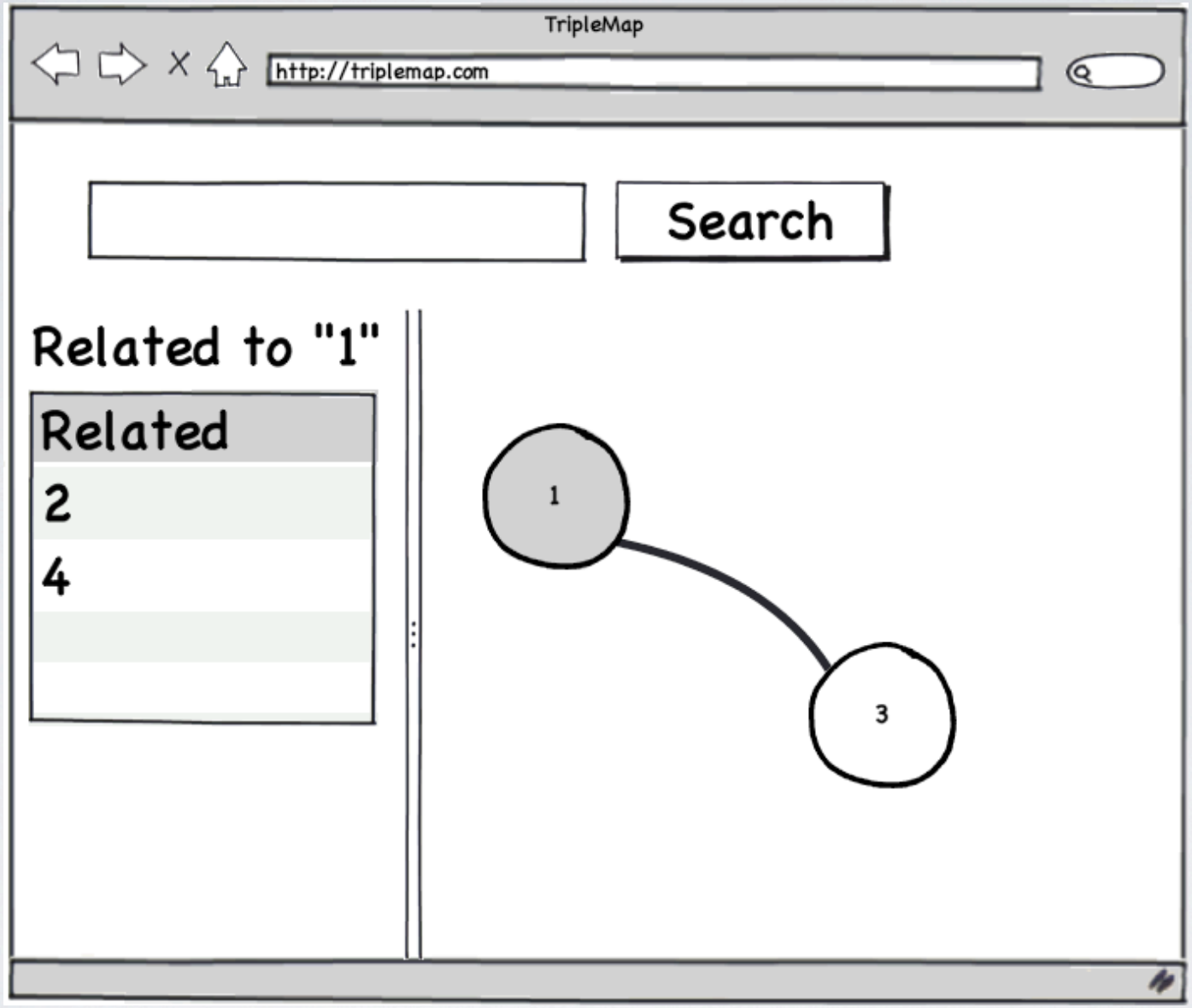
LINKING OPEN **DRUG** DATA

http://esw.w3.org/HCLSIG/LODD

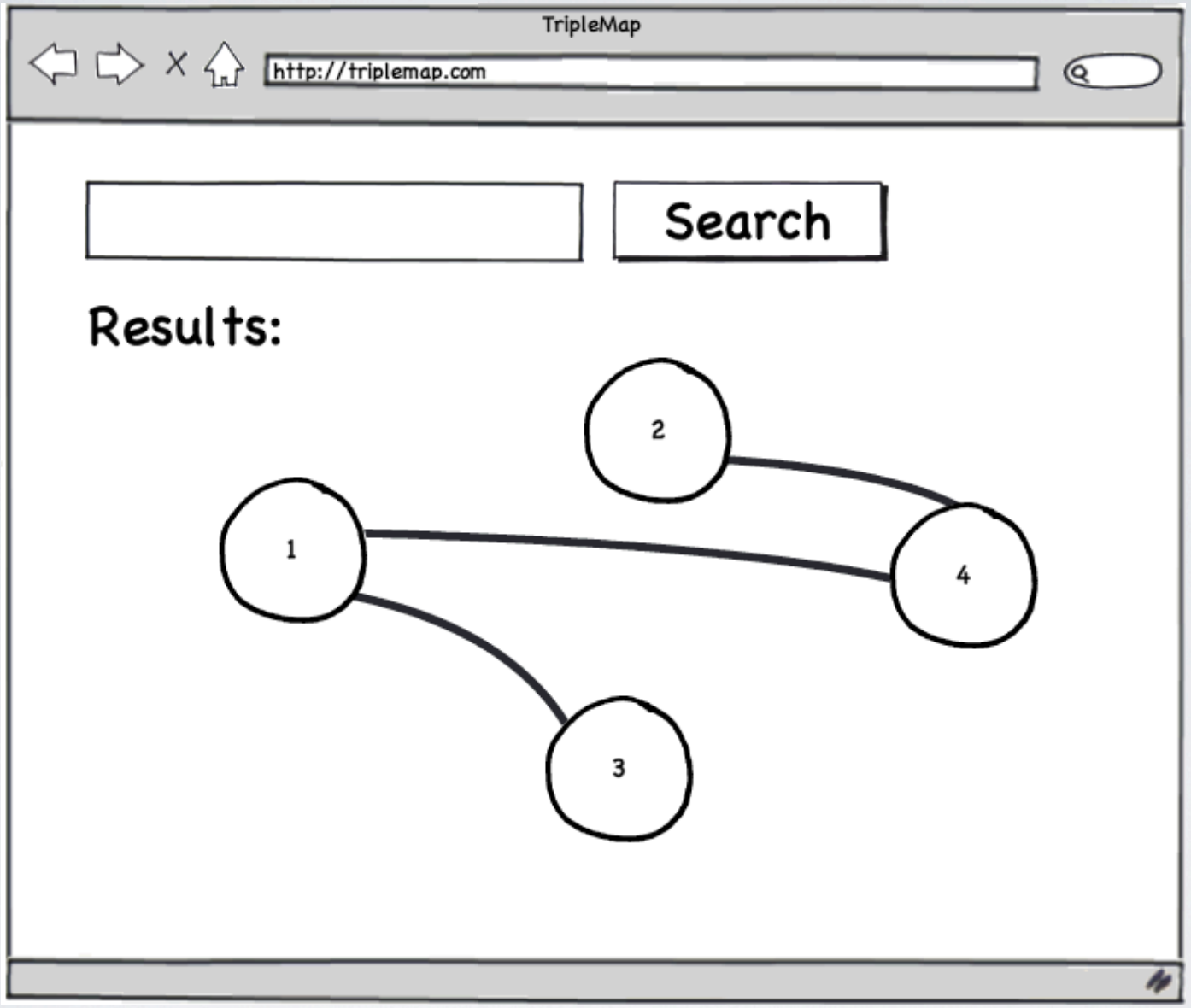


ENTAGEN
ACCELERATING INSIGHT





ENTAGEN
ACCELERATING INSIGHT



ENTAGEN
ACCELERATING INSIGHT

Ready, GO!

```
select id, label
from targets
where label = '${queryValue}'
```

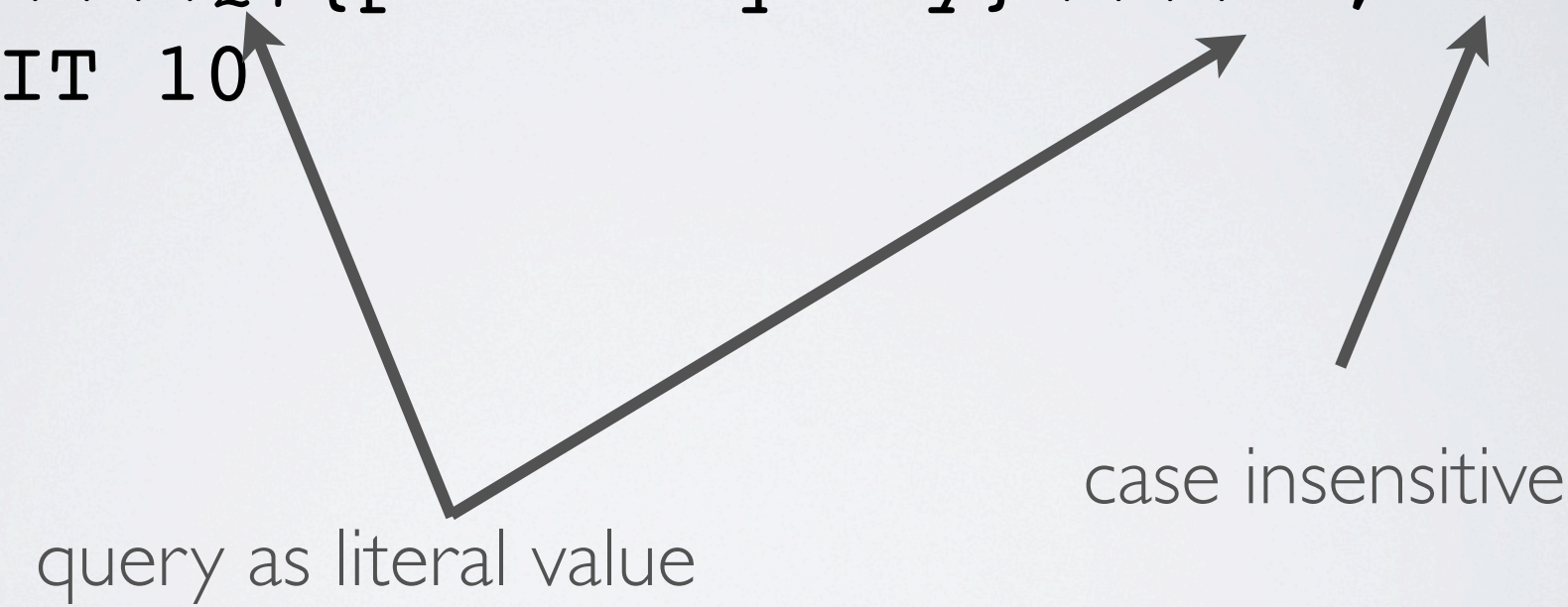
ENTAGEN
ACCELERATING INSIGHT

```
select id, label
from targets
where label
 ilike '%${queryValue}%'
```

```
SELECT ?uri ?type ?label WHERE {
   ?uri rdfs:label ?label .
   ?uri rdf:type ?type .
   FILTER (?label = '${params.query}')
} LIMIT 10
```

```
SELECT ?uri ?type ?label WHERE {
   ?uri rdfs:label ?label .
   ?uri rdf:type ?type .
   FILTER regex(?label,
      '\\\\Q${params.query}\\\\E', 'i')
} LIMIT 10
```

```
SELECT ?uri ?type ?label WHERE {
   ?uri rdfs:label ?label .
   ?uri rdf:type ?type .
   FILTER regex(?label,
      '\\\\Q${params.query}\\\\E', 'i')
} LIMIT 10
```



ENTAGEN
ACCELERATING INSIGHT

DEMO

Baseline SPARQL Query Performance

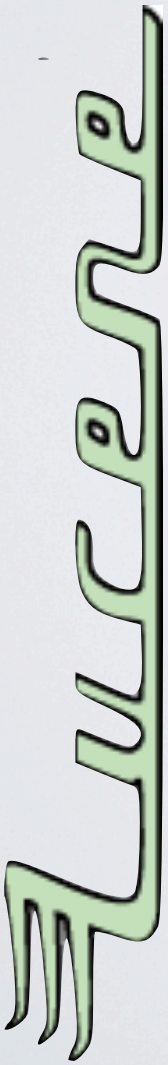
FASTER!

Java API

Indexing and Searching Text

http://wiki.apache.org/lucene-java/PoweredBy

indexing



storage

ENTAGEN
ACCELERATING INSIGHT

Document

Document

field	value
ID	2
name	"Mike Hugo"
company	"Entagen"
bio	"lorem ipsum dolor sum etc..."

ENTAGEN
ACCELERATING INSIGHT

Index

field	value
id	2
name	"Mike Hugo"
company	"Entagen"
bio	"lorem ipsum dolor sum etc.."

Indexed not Stored

ENTAGEN
ACCELERATING INSIGHT

Query: **name: mike**

ENTAGEN
ACCELERATING INSIGHT

Query: **name: mike**

Matching Documents:

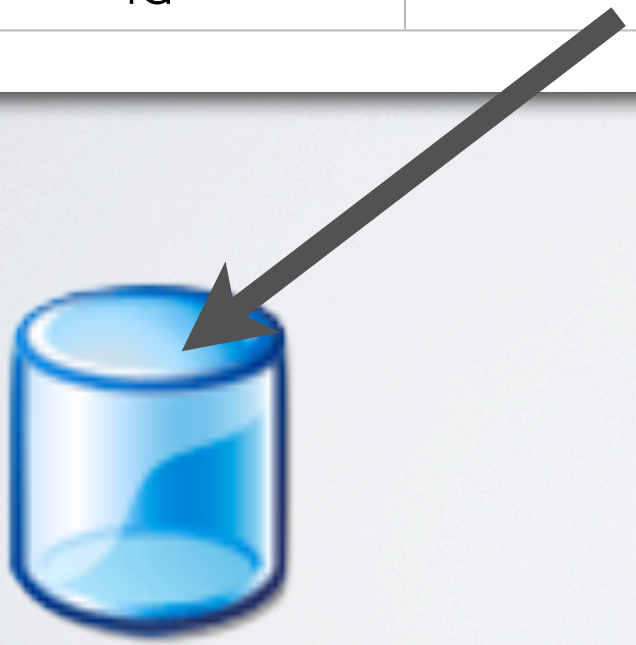
field	value
id	2

ENTAGEN
ACCELERATING INSIGHT

field	value
id	2

ENTAGEN
ACCELERATING INSIGHT

field	value
id	2



ENTAGEN
ACCELERATING INSIGHT

field	value
id	2

field	value
ID	2
name	"Mike Hugo"
company	"Entagen"
bio	"lorem ipsum dolor sum etc..."

ENTAGEN
ACCELERATING INSIGHT

Simplest Solution

Lucene index of rdfs:label

Build the Index

```
String queryLabels = """
    SELECT ?uri ?label
    WHERE {
        ?uri rdfs:label ?label .
    }
"""
```

Build a SPARQL query to find all the rdfs:label properties

```
sparqlQueryService.executeForEach(repo
    def doc = new Document()
    String uri = it.uri.stringValue()
    String label = it.label.stringValu

    doc.add(new Field(SUBJECT_URI_FIE
            Field.Store.YES, Field.Ind
```

ENTAGEN
ACCELERATING INSIGHT

```
sparqlQueryService.executeForEach
  (repository, queryLabels) {
    String uri = it.uri.stringValue()
    String label = it.label.stringValu

    new Document()

    doc.add(new Field(SUBJECT_URI_FIE
            Field.Store.YES, Field.In
    doc.add(new Field(LABEL_FIELD, la
            Field.Store.NO, Field.Ind

    writer.addDocument(doc)
```

Execute the SPARQL query

```
liqlQueryService.executeForEach(reposito
 String uri = it.uri.stringValue()
 String label = it.label.stringValue()

 Document doc = new Document()
 doc.add(new Field(SUBJECT_URI_FIELD,
         uri,
         Field.Store.YES,
         Field.Index.ANALYZED))
 doc.add(new Field(LABEL_FIELD,
         label,
         Field.Store.NO,
         Field.Index.ANALYZED))

 writer.addDocument(doc)
```

Instantiate a new Lucene Document

```
Document doc = new Document()
doc.add(new Field(SUBJECT_URI_FIELD,
        uri,
        Field.Store.YES,
        Field.Index.ANALYZED))
doc.add(new Field(LABEL_FIELD,
        label,
        Field.Store.NO
        Field.Index.AN

writer.addDocument(doc)

ly {
```

key

value

Add the Subject URI to the Document

```
            Field.Store.YES,
            Field.Index.ANALYZED));
 doc.add(new Field(LABEL_FIELD,
     value label,
            Field.Store.NO,
            Field.Index.ANALYZED));


 writer.addDocument

ly {
ter.close()   // Close index
```

key

Add the Label field to the document (but don't store it)

```
                Field.Store.NO,
                Field.Index.ANALYZED))

    writer.addDocument(doc)
}
inally {
 writer.close()   //
```

Add the document to the Index

Query the Index

```
query = {
Query query = new QueryParser(
    Version.LUCENE_CURRENT,
    LABEL_FIELD,
    new StandardAnalyzer())
            .parse(params.query);

def s                                    ime
List                                     teQuery(query)
def e                                    ime

render(view: 'index', model: [results:
```

query this field

for this value

Create a Lucene Query from user input

```
IndexSearcher searcher = luceneSearche
ScoreDoc[] scoreDocs =
    searcher.search(query, 10).scoreDo
List results = []
def connection = r                    ctior
scoreDocs.each {
    Document doc =                    t.doc
    String uri = doc[SUBJECT_URI_FIELD
    Map labelAndType = sparqlQueryServ
    results << [uri: uri, type: labelA
}
connection.close()
return results
```

Search the index (limit 10) for matching documents

```
List results = []
def connection = repository.connectior
scoreDocs.each {
  Document doc = searcher.doc(it.doc)
  String uri = doc[SUBJECT_URI_FIELD]
  Map labelAndType =
    spa
      g                    i, connection)
  resu
          uri: uri,
          type: labelAndType.type,
          label: labelAndType.label])
}
connection.close()
return results
```

For each matching document, get the doc and extract the Subject URI

```
List results = []
def connection = repository.connectior
scoreDocs.each {
  Document doc = searcher.doc(it.doc)
  String uri = doc[SUBJECT_URI_FIELD]
  Map labelAndType =
   sparqlQueryService.
     getLabelAndType(uri, connection)
  results.add([
          uri: uri
          type: la
          label: l                    ])
}
connection.close()
return results
```

Using the Subject URI, load properties from the triplestore

```
List results = []
def connection = repository.connectior
scoreDocs.each {
  Document doc                                .doc)
  String uri =                                IELD]
  Map labelAnd
   sparqlQueryService.
     getLabelAndType(uri, connection)
  results.add([
        uri: uri,
        type: labelAndType.type,
        label: labelAndType.label])
}
connection.close()
return results
```

return results containing Subject URI, Type, and Label

DEMO

Lucene Index of Searchable Labels

WHAT ABOUT ENTITY RELATIONSHIPS?

WHAT ABOUT OTHER PROPERTIES?

SIREn

{ Semantic Information Retrieval Engine }

Lucene Extension

Indexing and Searching Semi-Structured Data

ENTAGEN

ACCELERATING INSIGHT

SIREn

{ Semantic Information Retrieval Engine }

Document

ENTAGEN

ACCELERATING INSIGHT

SIREn

{ Semantic Information Retrieval Engine }

Document

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Build the Index

```
Connection connection = repository.conn
y {
  String subjectUris = """
     SELECT distinct ?uri
     WHERE {
         ?uri ?p ?o .
     }
  """

  sparqlQueryService.executeForEach(rep
      def doc = ne

      String subje                    ring
      doc.add(new Field(SUBJECT_URI_FIE
              subjectUri,
              Field.Store.YES,
```

Select all Subject URIs from the triplestore

ENTAGEN
ACCELERATING INSIGHT

```
"""
sparqlQueryService.executeForEach(
  repository, subjectUris) {
    def doc = new Document()

    String subjectUri = it.uri.stringV
    doc.add(new Field(SUBJECT_URI_FIEI
            subjectUri,
            Field.Store.YES,

    StringWr                    er =
    NTriplesWriter nTriplesWriter =
        new NTriplesWriter(triplesStri
    connection.exportStatements(
```

Execute the Sparql Query
For each URI, create a new Document

ENTAGEN
ACCELERATING INSIGHT

```
def doc = new Document()

String subjectUri = it.uri.stringValue
doc.add(new Field(SUBJECT_URI_FIELD,
        subjectUri,
        Field.Store.YES,
        Field.Index.ANALYZED))

StringWriter triplesStringWriter = new
NTriplesWriter nT
    new NTriplesW                    ngWr
connection.exportStatements(
        new URIImpl(subjectUri),
        null, null, false,
```

Add the Subject URI to the Document

ENTAGEN
ACCELERATING INSIGHT

```
        Field.Index.ANALYZED))

StringWriter triplesStringWriter = new
NTriplesWriter nTriplesWriter =
    new NTriplesWriter(triplesStringWr
connection.exportStatements(
        new URIImpl(subjectUri),
        null, null, false,
        nTriplesWriter)

doc.add(new Field(T
        triplesStri                    ()
        Field.Store.
        Field.Index.ANALYZED))
```

Get an NTriples string from the triplestore

ENTAGEN
ACCELERATING INSIGHT

```
            new URIImpl(subjectUri),
            null, null, false,
            nTriplesWriter)

doc.add(new Field(TRIPLES_FIELD,
            triplesStringWriter.toString()
            Field.Store.NO,
            Field.Index.ANALYZED))

writer.addDocumen
```

Add the NTriples string to the document

ENTAGEN
ACCELERATING INSIGHT

```
doc.add(new Field(TRIPLES_FIELD,
        triplesStringWriter.toString(
        Field.Store.NO,
        Field.Index.ANALYZED))

writer.addDocument(doc)
```

Add the document to the index

Query the Index

```
SirenCellQuery predicate =
  new SirenCellQuery(
    new SirenTermQuery(
      new Term(TRIPLES_FIELD,
      RDFS.LABEL.stringValue())));
predicate.constraint = PREDICATE_CELL

SirenCellQuery object =
  new SirenCellQue
    new SirenTermQu
      new Term(TRIPLES_FIELD,
      params.query.toLowerCase()))
object.constraint = OBJECT_CELL
```

query the Triples field

```
SirenCellQuery predicate =
  new SirenCellQuery(
    new SirenTermQuery(
        new Term(TRIPLES_FIELD,
        RDFS.LABEL.stringValue())));
predicate.constraint = PREDICATE_CELL

SirenCellQuery object =
  new SirenCellQue
    new SirenTermQuery(
        new Term(TRIPLES_FIELD,
        params.query.toLowerCase()))
object.constraint = OBJECT_CELL
```

for a predicate

```
SirenCellQuery predicate =
  new SirenCellQuery(
    new SirenTermQuery(
      new Term(TRIPLES_FIELD,
      RDFS.LABEL.stringValue())));
predicate.constraint = PREDICATE_CELL

SirenCellQuery object
  new SirenCellQuery(
    new SirenTermQuery(
      new Term(TRIPLES_FIELD,
                                Case()))
object.constraint = OBJECT_CELL
```

of rdfs:label *

*** note: could be any predicate!**

```
SirenCellQuery object =
  new SirenCellQuery(
   new SirenTermQuery(
            new Term(TRIPLES_FIELD,
             params.query.toLowerCase()))
object.constraint = OBJECT_CELL

Query query = new SirenTupleQuery()
query.add(predica
        SirenTupl                T)
query.add(object,
        SirenTupleClause.Occur.MUST)
```

query the Triples field

```
SirenCellQuery object =
   new SirenCellQuery(
    new SirenTermQuery(
            new Term(TRIPLES_FIELD,
            params.query.toLowerCase()))
object.constraint = OBJECT_CELL

Query query = new SirenTupleQuery()
query.add(predicate
        SirenTupleClause.Occur.MUST)
query.add(object,
        SirenTupleClause.Occur.MUST)
```

for an object

ENTAGEN
ACCELERATING INSIGHT

```
List results = executeQuery(query)
```

```
SirenCellQuery object =
  new SirenCellQuery(
   new SirenTermQuery(
            new Term(TRIPLES_FIELD,
              params.query.toLowerCase()))
object.constraint = OBJECT_CELL

Query query = new SirenTupleQuery()
query.add(predica
        SirenTupl                    T)
query.add(object,
        SirenTupleClause.Occur.MUST)

List results = executeQuery(query)
```

matching the user input

ENTAGEN
ACCELERATING INSIGHT

```
                params.query.toLowerCase())
object.constraint = OBJECT_CELL
```

```
Query query = new SirenTupleQuery()
query.add(predicate,
          SirenTupleClause.Occur.MUST)
query.add(object,
          SirenTupleClause.Occur.MUST)
```

```
List results = exe
```

Build query of with both pieces - predicate and object

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query: **“imatinib”**

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

triples field

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

predicate = rdfs:label

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

predicate = rdfs:label

object = “imatinib”

ENTAGEN
ACCELERATING INSIGHT

```
List executeQuery(Query query) {
IndexSearcher searcher = sirenSearcherM
ScoreDoc[] scoreDocs =
  searcher.search(query, 10).scoreDocs
List results = []
def connection = repos                    n
scoreDocs.each {
    Document doc = sea                    c
    String uri = doc[S                    D
    Map labelAndType = sparqlQueryServi
        getLabelAndType(uri, connectio
    results.add([
            uri: uri,
            type: labelAndType.type,
            label: labelAndType.label
```

Search the index (limit 10) for matching documents

ENTAGEN
ACCELERATING INSIGHT

```
List results = []
def connection = repository.connection
scoreDocs.each {
    Document doc = searcher.doc(it.doc)
    String uri = doc[SUBJECT_URI_FIELD]
    Map labelAndType = sparqlQueryServic
        get                    connectio
    results.
                                   pe.type,
            label: labelAndType.label])
}
connection.close()
return results
```

For each matching document, get the doc and extract the Subject URI

```
 connection = repository.connection
reDocs.each {
 Document doc = searcher.doc(it.doc)
 String uri = doc[SUBJECT_URI_FIELD]
 Map labelAndType = sparqlQueryService.
      getLabelAndType(uri, connection)
 results.add([
         uri: uri,
         type: lab
         label: la

nection.close()
urn results
```

Using the Subject URI, load properties from the triplestore

ENTAGEN
ACCELERATING INSIGHT

```
String uri = doc[SUBJECT_URI_FIELD]
Map labelAndType = sparqlQueryService.
    getLabelAndType(uri, connection)
results.add([
        uri: uri,
        type: labelAndType.type,
        label: labelAndType.label])

nection.close()
urn results
```

return results containing Subject URI, Type, and Label

ENTAGEN
ACCELERATING INSIGHT

DEMO

SIREn Index of RDF Entities

FLEXIBILITY

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

predicate = rdfs:label

object = “imatinib”

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

object = “imatinib”

ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" <DB00619> drugbank:target <targets/1588> .

Query:

object = “imatinib”

OR

object = “gleevec”

ENTAGEN
ACCELERATING INSIGHT

MORE THAN LITERALS

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

predicate = brandName

ENTAGEN
ACCELERATING INSIGHT

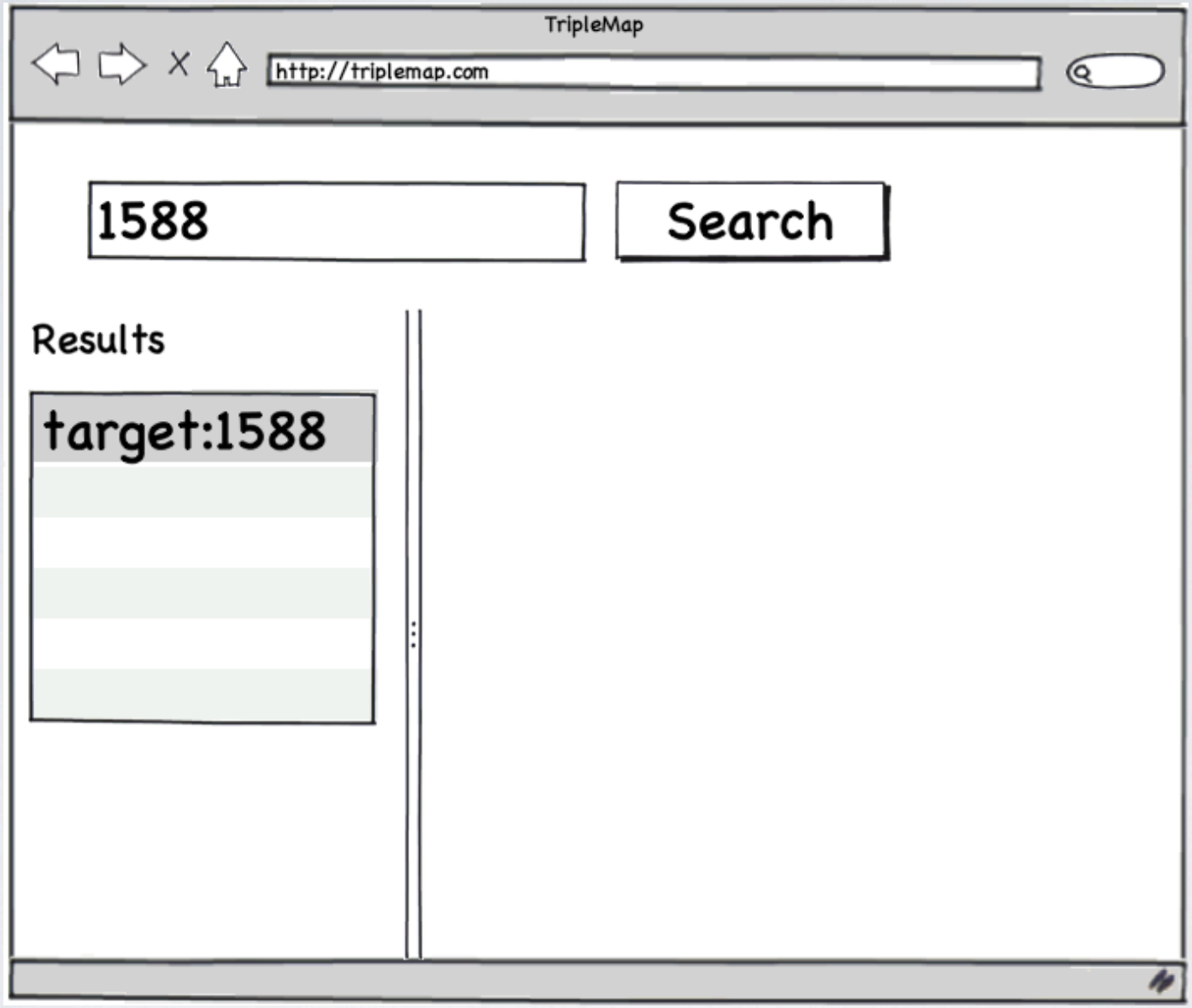
field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

Query:

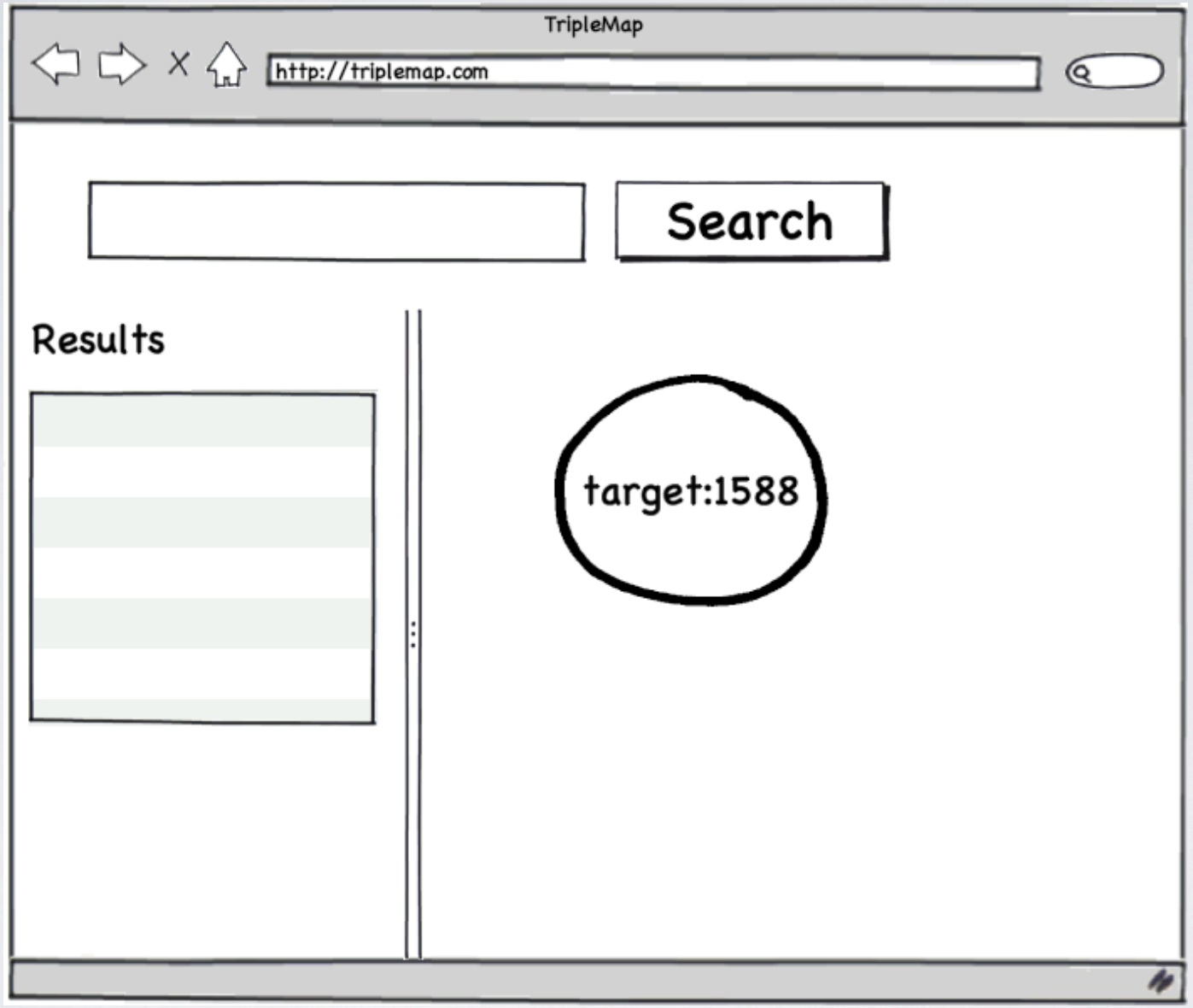
predicate = target

ENTAGEN
ACCELERATING INSIGHT

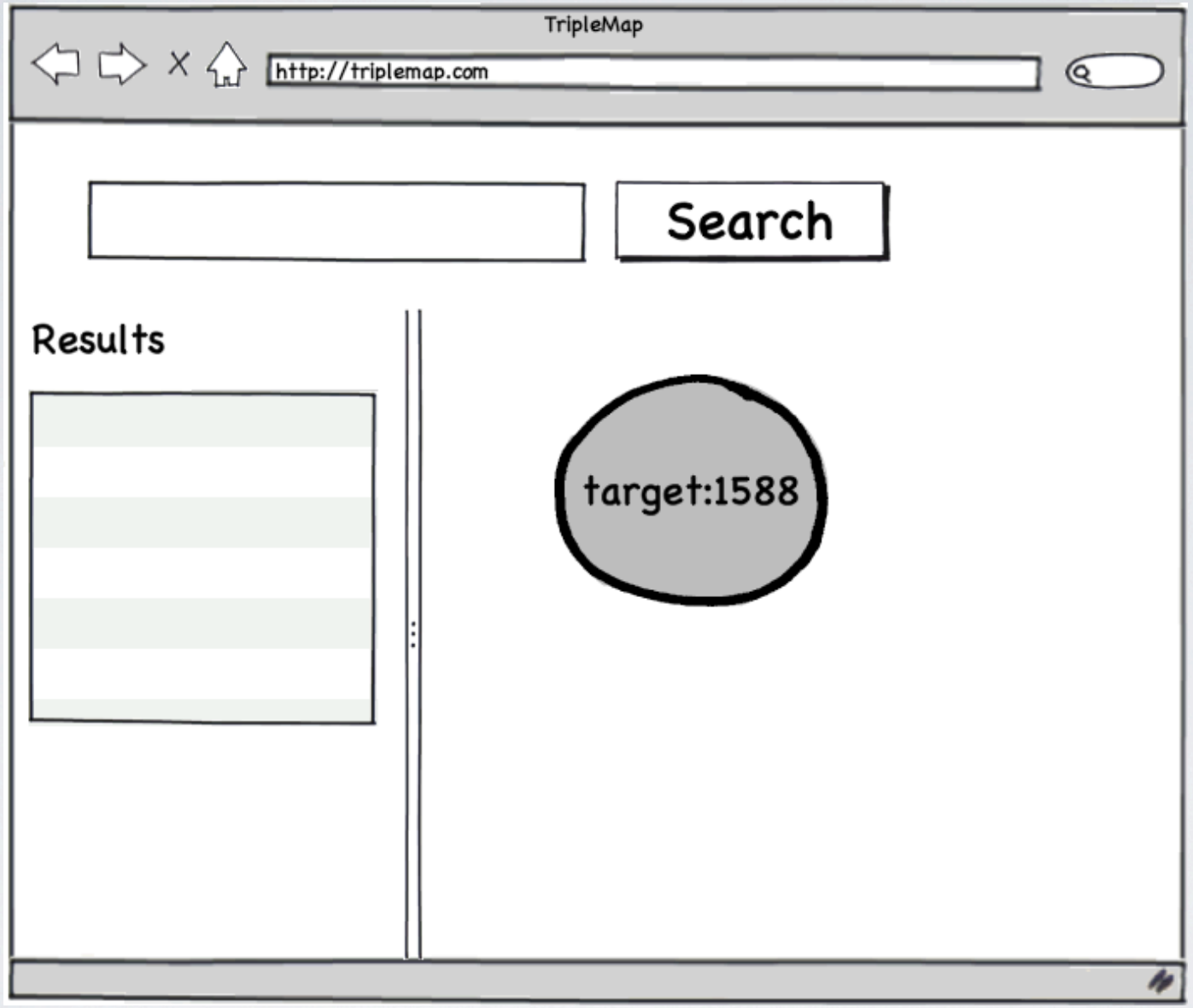
RELATIONSHIPS



ENTAGEN
ACCELERATING INSIGHT



ENTAGEN
ACCELERATING INSIGHT



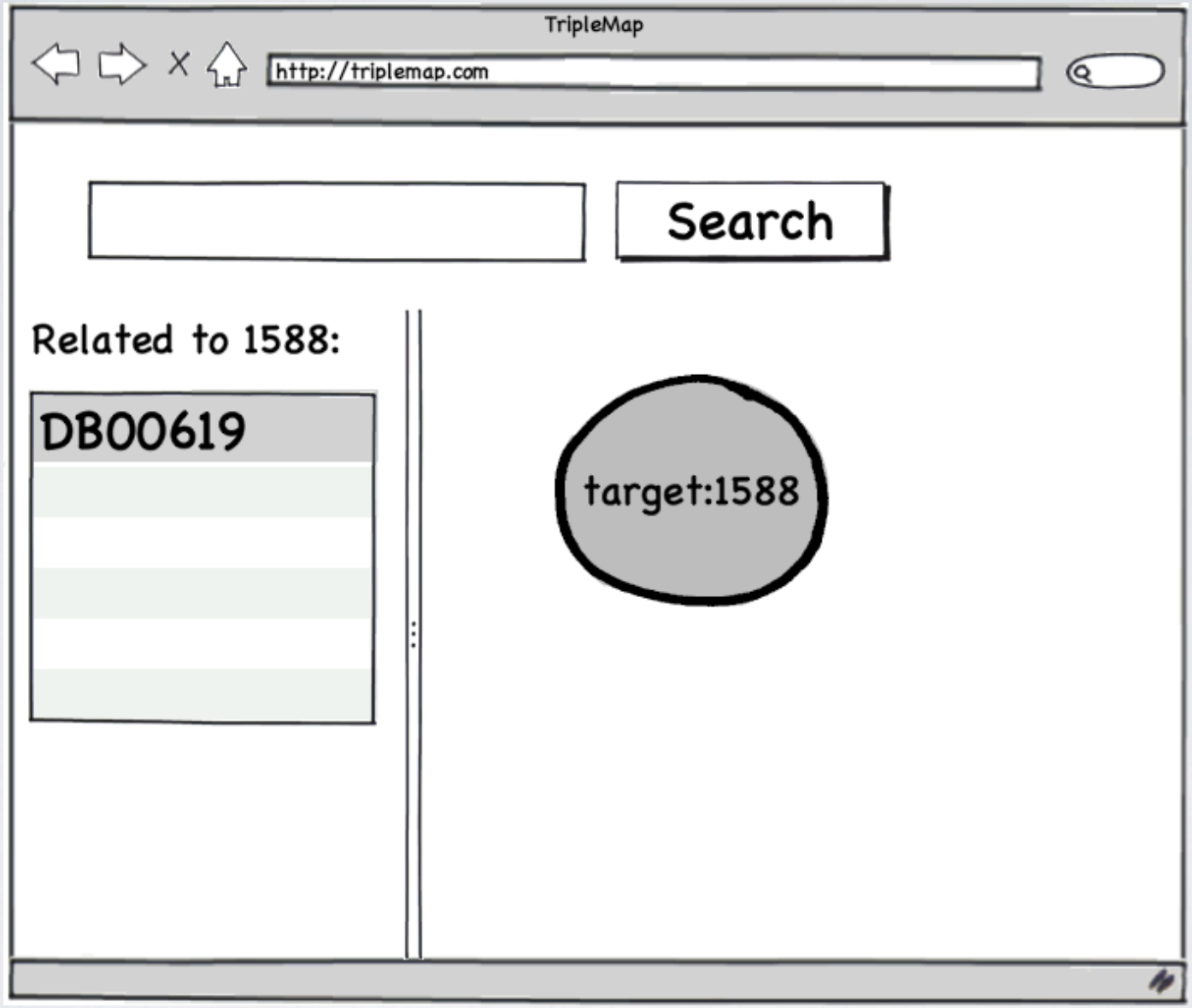
ENTAGEN
ACCELERATING INSIGHT

field	value
URI	<DB00619>
triples	<DB00619> rdfs:label "Imatinib" . <DB00619> rdf:type <drugbank:drugs> . <DB00619> drugbank:brandName "Gleevec" . <DB00619> drugbank:target <targets/1588> .

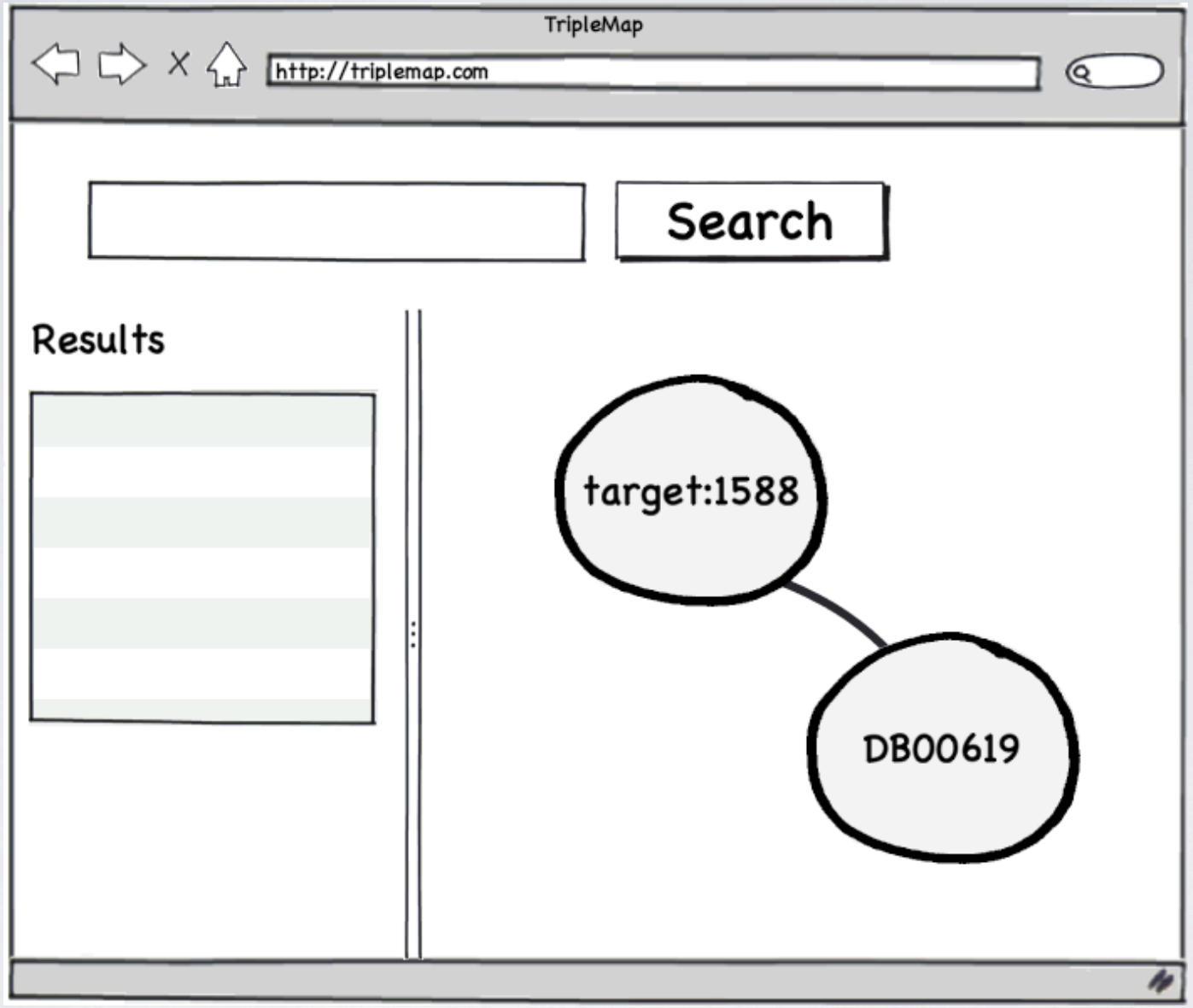
Query:

object = <targets/1588>

ENTAGEN
ACCELERATING INSIGHT



ENTAGEN
ACCELERATING INSIGHT



ENTAGEN
ACCELERATING INSIGHT

DEMO

Searching SIREn Index for Relationships

SIREn

{ Semantic Information Retrieval Engine }

Distributed
Indexing and Searching Semi-Structured Data

ENTAGEN
ACCELERATING INSIGHT

scale

ENTAGEN
ACCELERATING INSIGHT

Home About Se

Search interface type: Simple Advanced **Guru** Query Language Documentation

keyword(s)

ntriple(s) * <name> "Tim Berners Lee"

filter(s)

SEARCH Group By Dataset: Sorted by: relevance

Quick filters (All options)

Time range:

Any date

Today Yesterday Last week

Last month Last year

Format:

Any format

RDF RDFA MICROFORMAT XFN

HCARD HCALENDAR HLISTING

HRESUME LICENSE GEO ADR

Sindice search:* <name> "Tim Berners Lee" found 699 documer

http://dig.csail.mit.edu/2007/01/camp/data (RDF)

2009-08-12 - 387 triples in 43.2 kB

http://dig.csail.mit.edu/2007/01/camp/data (Search) Inspect: (Cache) (Live)

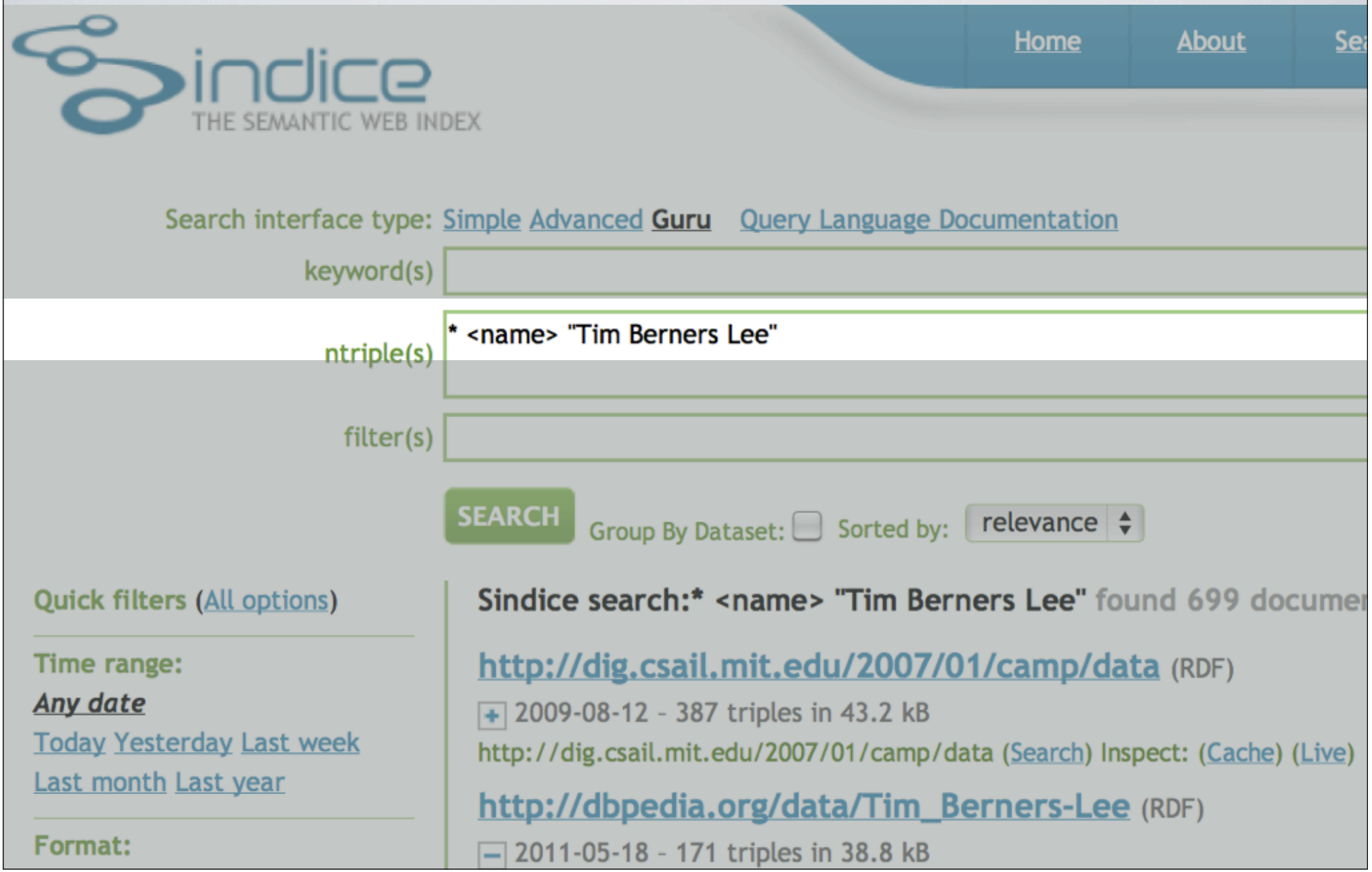
http://dbpedia.org/data/Tim_Berners-Lee (RDF)

2011-05-18 - 171 triples in 38.8 kB

SIGMA FULL CONTENT GRAPH TRIPLES(890)

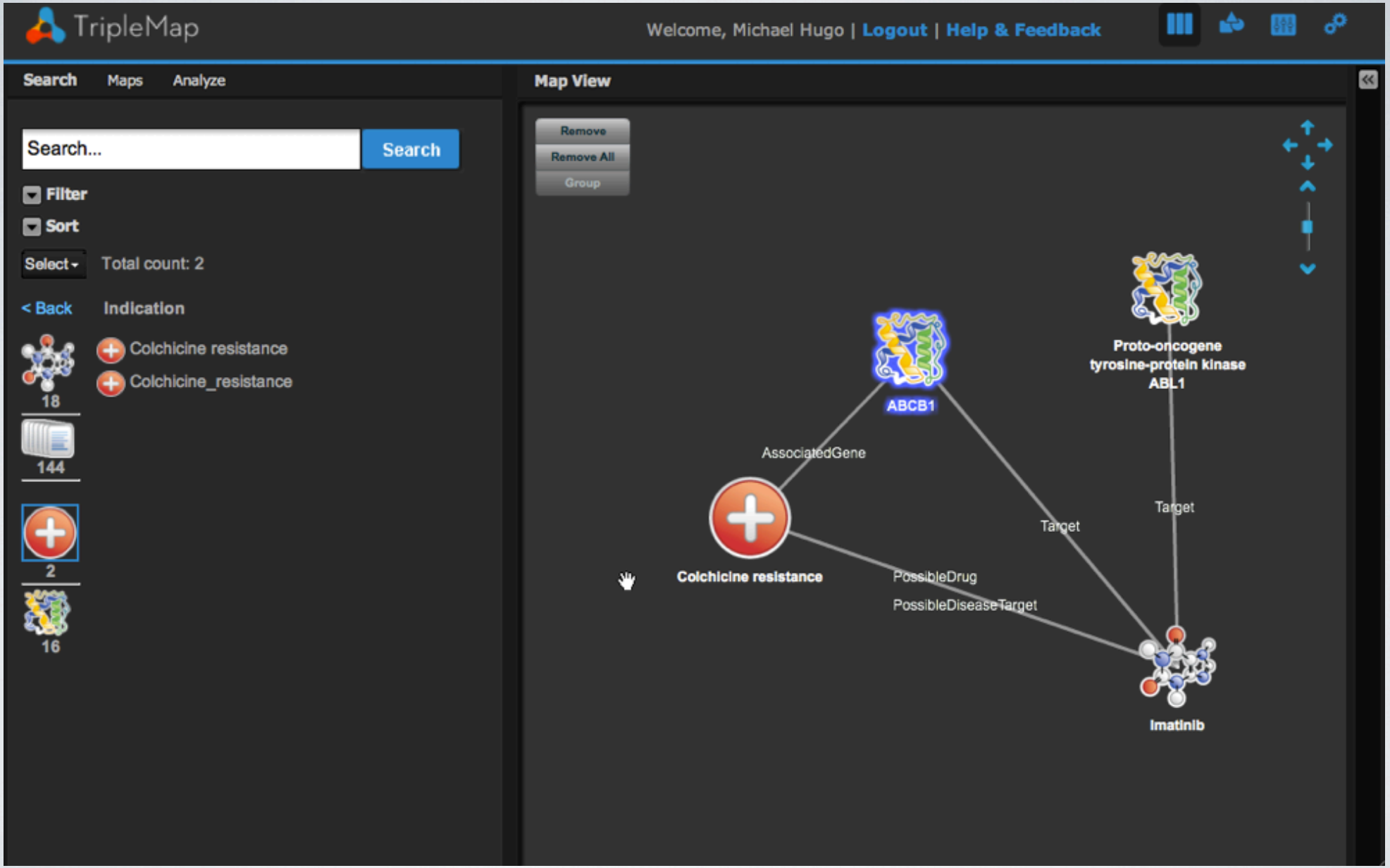
See all we know about "Tim Ber

Query Parser



DEMO

SIREn in action on TripleMap.com



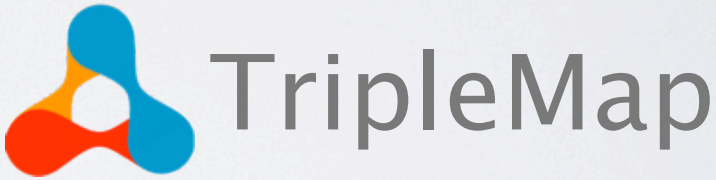
ENTAGEN
ACCELERATING INSIGHT


SPARQL
LUCENE
SIREN
TripleMap.com
ENTAGEN
ACCELERATING INSIGHT

QUESTIONS?

mike@entagen.com / twitter: @piragua

http://www.entagen.com



http://www.triplemap.com