SPARQL演習

トルヴェ　アントワン　（九州大学システム情報科学大学院　助教）
後藤　孝行　（九州大学共進化社会システム創成拠点　学術研究員）

２０１５年２月５日

SRPビル２階

連絡先：teapot-admin@isit.or.jp

本日のスケジュール

背景：セマンティックWebおよびオープンデータ（約１０分）

本題１：RDFの紹介（約３０分）

本題２：SPARQLの一般紹介（約１５分）

休憩：１０分

演習１：SPARQLを使ってみよう（約１時間）

演習２：初めてのSPARQLを使ったWebアプリ（約４５分）

休憩：１０分

演習３：地図アプリを作りましょう（約１時間）

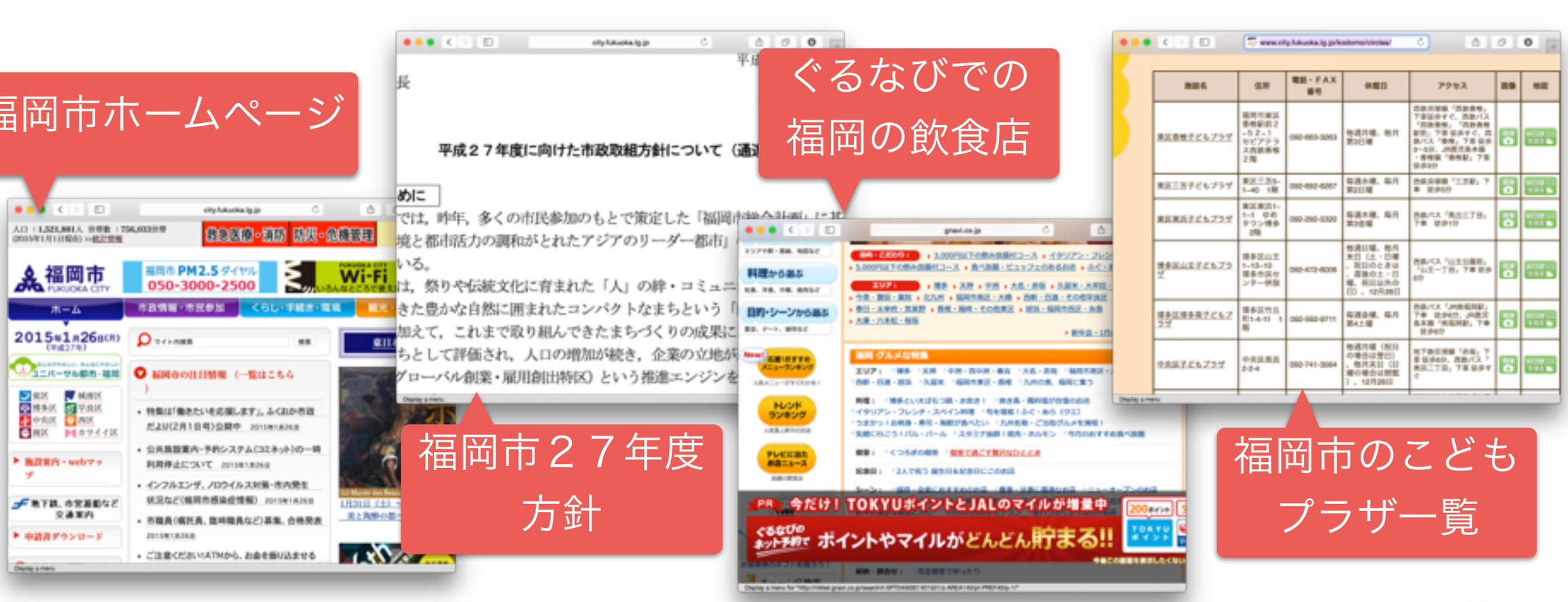
始める前に・・・


bodik.jp
オープンデータで
みんなを
ハッピーに！
OPENDATA CONTEST
2014-2015 FUKUOKA
オープンデータコンテスト 2014-2015 福岡
応募受付開始日
2014年 12月22日(月)
応募締切日
2015年 2月23日(月)
表彰式
2015年 3月6日(金)
募集部門
アプリケーション部門
アイデア部門
応募フォームはこちら ＞
ホーム
開催概要
実証概要
応募方法
応募規約
審査・表彰
受賞作品
お問い合わせ
お知らせ
2014.12.22
「オープンデータ・コンテスト2014-2015 Fukuoka」サイトオープン
開催概要
オープンデータコンテスト2014-2015 Fukuokaの開催概要です。
実証概要
今年度の実証実験の概要をご案内いたします。

背景セマンティックWeb

について

既存のWebの課題：検索



- インターネットは膨大な情報が閲覧可能
- が、正規化・構造化されていないため、
  検索は困難

セマンティックWebによる解決策

ページやページの中にあるエレメントに

属性情報を付加する

属性情報等は検索可能とする

標準技術を提案する

ページ属性情報

about: 福岡市
type: ホームページ
lastupdate: 2015-2-1

about: 福岡市
type: 市政方針資料
fiscal year: 2015

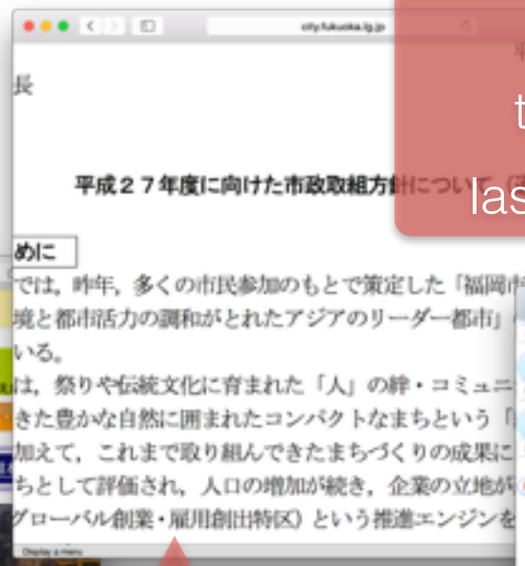
about: 福岡市
type: 飲食店一覧
lastupdate: 2015-2-5

about: 福岡市
type: こどもプラザー覧
type: 子育支援情報

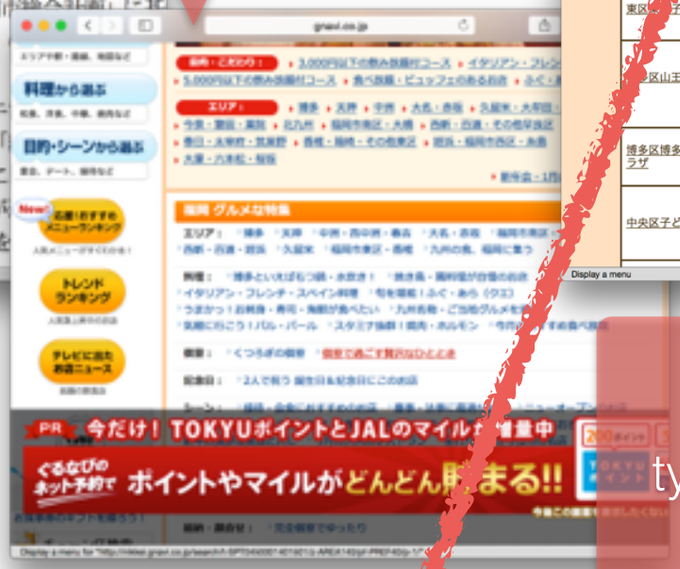
ページエレメント属性情報

about: 福岡市
type: ホームページ
lastupdate: 2015-2-1

about: 福岡市
type: 市政方針資料
fiscal year: 2015



about: 福岡市
type: 飲食店一覧
lastupdate: 2015-2-5



about: 福岡市
type: こどもプラザ一覧
type: 子育支援情報

東区三苫子どもプラザ	東区三苫5–1–40　1階	092-692-6267	毎週水曜、毎月第2日曜	西鉄貝塚線「三苫駅」下車　徒歩5分	画像	WEB

about: 東区東浜子どもプラザ
type: こどもプラザ
type: 子育支援情報
address: 東区東浜1–1–1ゆめタウン博多2階
telephone: 092-292-5320

更に！

団体（自治体等）、各個人が生成・蓄積する膨大なデータをWebで公開・構造化したら、より便利な社会へ！

しかし

データの利用条件（ライセンス）を明確する必要がある

オープンデータ

データ間の関係性を表さないと探索が困難

リンクトオープンデータ（LOD）

オープンデータ級

二次利用可能
ライセンス

★オープンライセンス

★★構造化データ

★★★オープン形式

エクセルよりも、
CSV等

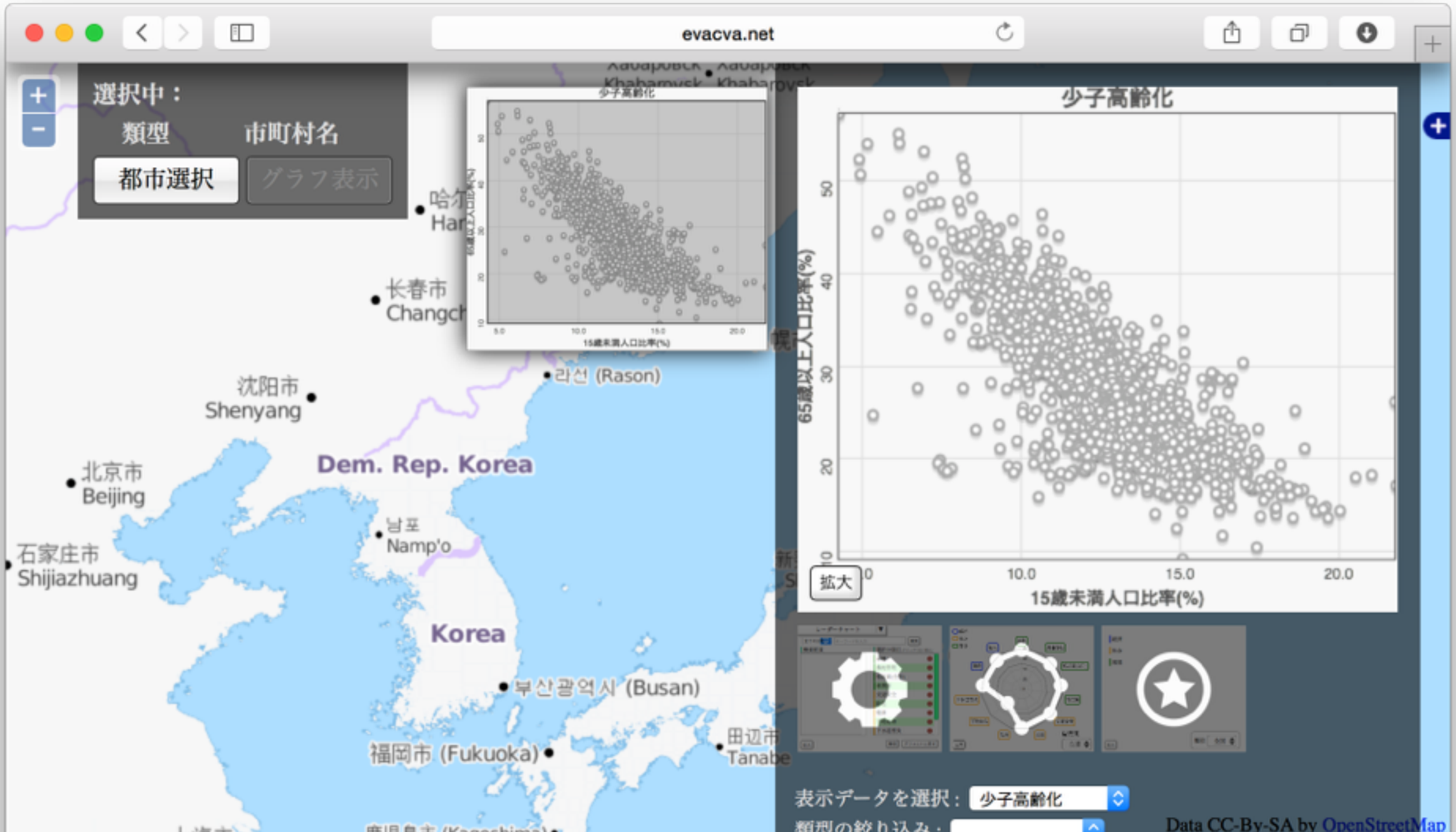
★★★★RDFを利用

★★★★★リンクRDFを利用

事例１：かなざわ育なび


kirakana.city.yokohama.lg.jp
かなざわ育なび.net
出産・子育て
保育園・幼稚園
医療機関
おでかけイベント
おでかけスポット
防災・減災
お知らせ
【募集！】かなざわ育なび.netに広告を出しませんか？
【UDCN並木ラボ】子育てしやすいまちを目指す新しいモビリティモデルの意見交換会
【イベント】３／７　キラキラかなざわっこ　春の祭典！
【イベント】けいきゅん ぽたんちゃん キャラ弁コンテスト2015
【お知らせ】先ほど一時的にかなざわ育なび.netの動作が不安定になりました。ご迷惑をおかけして申し訳ありません。
出産・子育てイベント
すべての出産・子育てイベントを見る
乳幼児健康診査
測定、診察、歯科健診、尿検査、育児相談
予防接種スケジューラー
金沢動物園のイベント
コアラのランチタイム
毎日 13:30 ～ 13:40
ゾウさんタイム
毎日 14:30 ～ 14:40
ゾウの調教
毎日 11:20 ～ 11:40
ARページ
子育て支援拠点とことこのイ

事例２：Evacva


evacva.net
選択中：
類型
市町村名
都市選択
グラフ表示
少子高齢化
65歳以上人口比率(%)
15歳未満人口比率(%)
拡大
表示データを選択：
少子高齢化
類型の絞り込み：
Data CC-By-SA by OpenStreetMap
哈尔滨市
长春市
沈阳市
Shenyang
北京市
Beijing
石家庄市
Shijiazhuang
Dem. Rep. Korea
Korea
라선 (Rason)
남포
Namp'o
부산광역시 (Busan)
福岡市 (Fukuoka)
田辺市
Tanabe

本題１ RDFについて

セマンティックWebの
データモデル

Resource Description Framework

リソース（＝物）　記述　枠組み

W3Cはインターネットで使われている技術（HTML、Javascript、CSS等）を標準化する団体。
１９９４年に結成されました。

- **W3C**の標準技術
- 複数の標準技術の集合（抽象的なセマンティックス、テキスト表現等）
- 現在のバージョンは**1.1**です

RDF1.1データモデル(1)

トリプル＝物３つ
ちなみに、**カープル＝物２つ／<u>シングル</u>トン**＝物１つ

全てデータは**トリプル**で表現する

主語

例：福岡

述語

＝プロパティ名

例：人口

目的語

＝プロパティ値

例：1521881人

RDF1.1データモデル(2)

主語

型：IRI

述語

型：IRI

目的語

型：IRI又は
リテラル

- IRIとは？
  - International（国際）Resource（リソース） Identifier（識別子）
  - 人物を指すユニークな識別子
  - URLはIRIの例です（Webサイトを指す）
- リテラルとは？
  - 単なる文字列
  - RDFではタイプや言語を付けることもできる

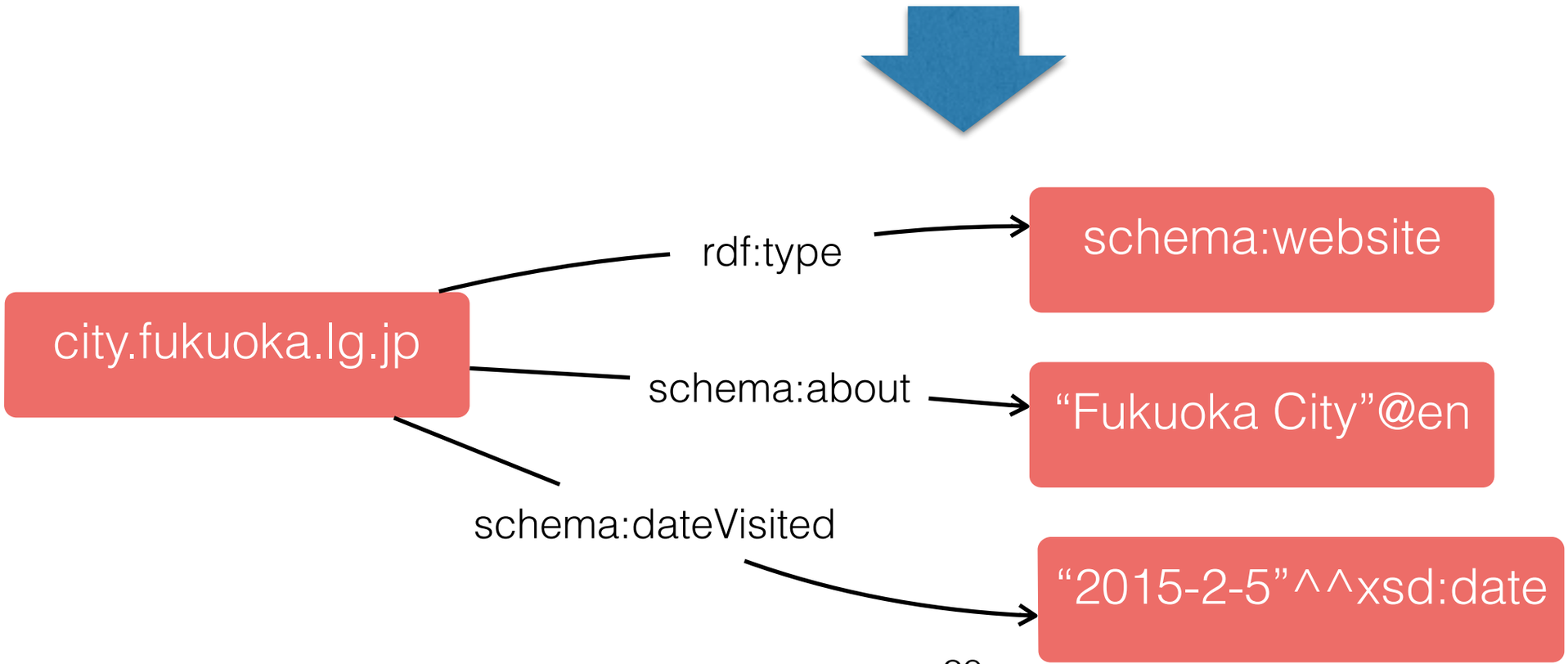
RDFモデルのサンプル

主語	述語	目的語	型・言語
city.fukuoka.lg.jp	rdf:type	schema:website	
city.fukuoka.lg.jp	schema:about	Fukuoka City	en
city.fukuoka.lg.jp	schema:dateVisited	2015-2-5	xsd:date

さっきの福岡市ホームページ

RDFのグラフ形式

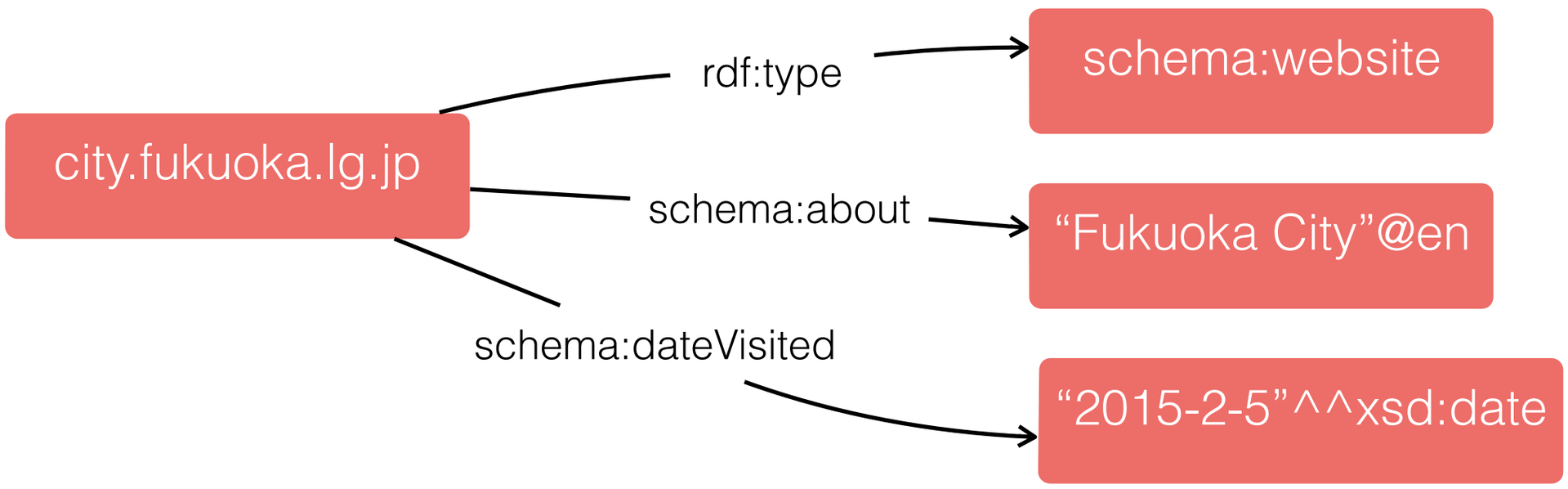
主語	述語	目的語	型・言語
city.fukuoka.lg.jp	rdf:type	schema:website	
city.fukuoka.lg.jp	schema:about	Fukuoka City	ja
city.fukuoka.lg.jp	schema:dateVisited	2015-2-5	xsd:date



RDFのグラフ形式

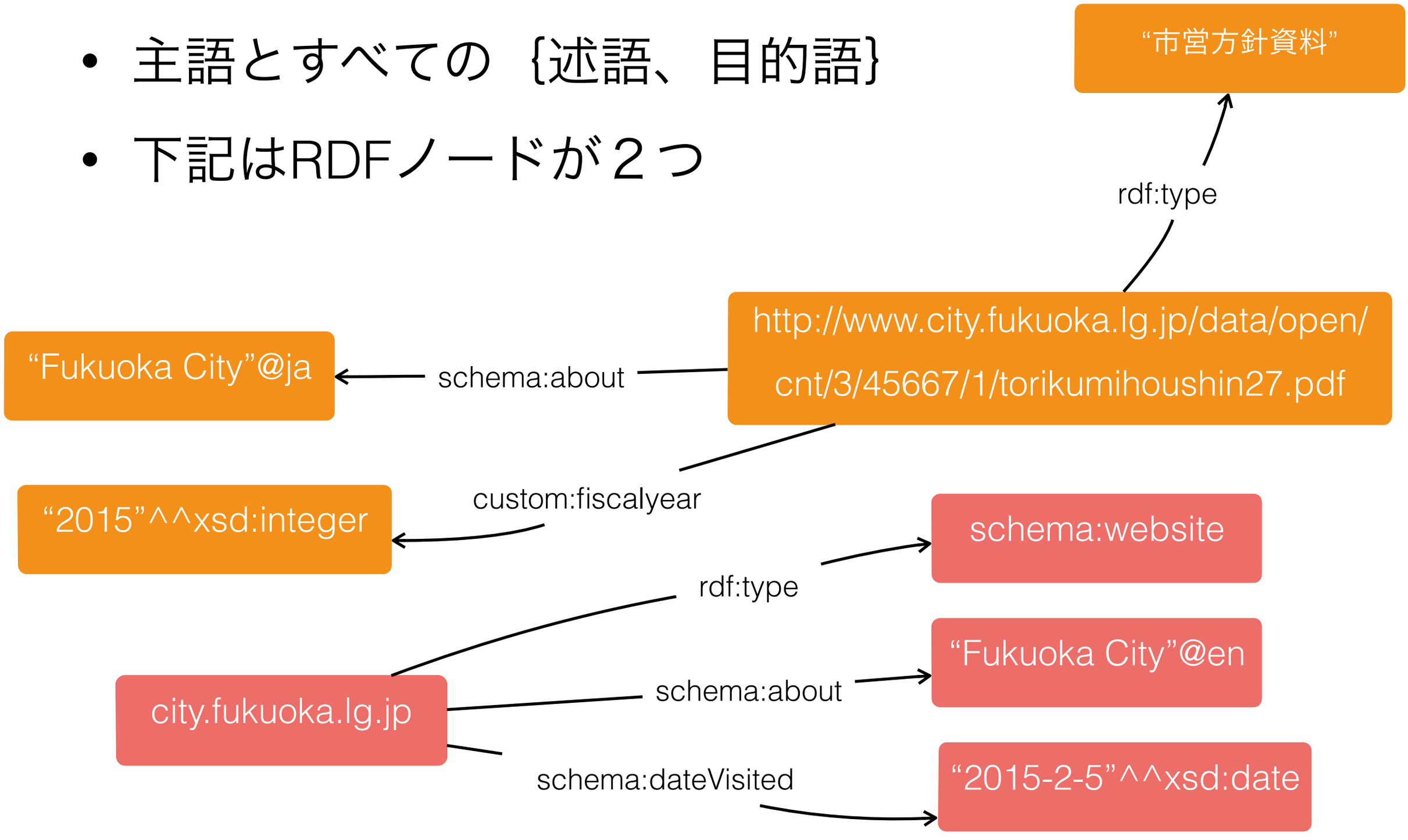
- 主語と目的語を矢印で繋ぐ
- 矢印は述語を表す（ラベル付き）
- 目的語の型や言語は下記のシンタックスで表す
  - 言語：“XXX”@lang
  - 型：”XXX”^^type

RDF/Turtle
シンタックス
（後で説明する）



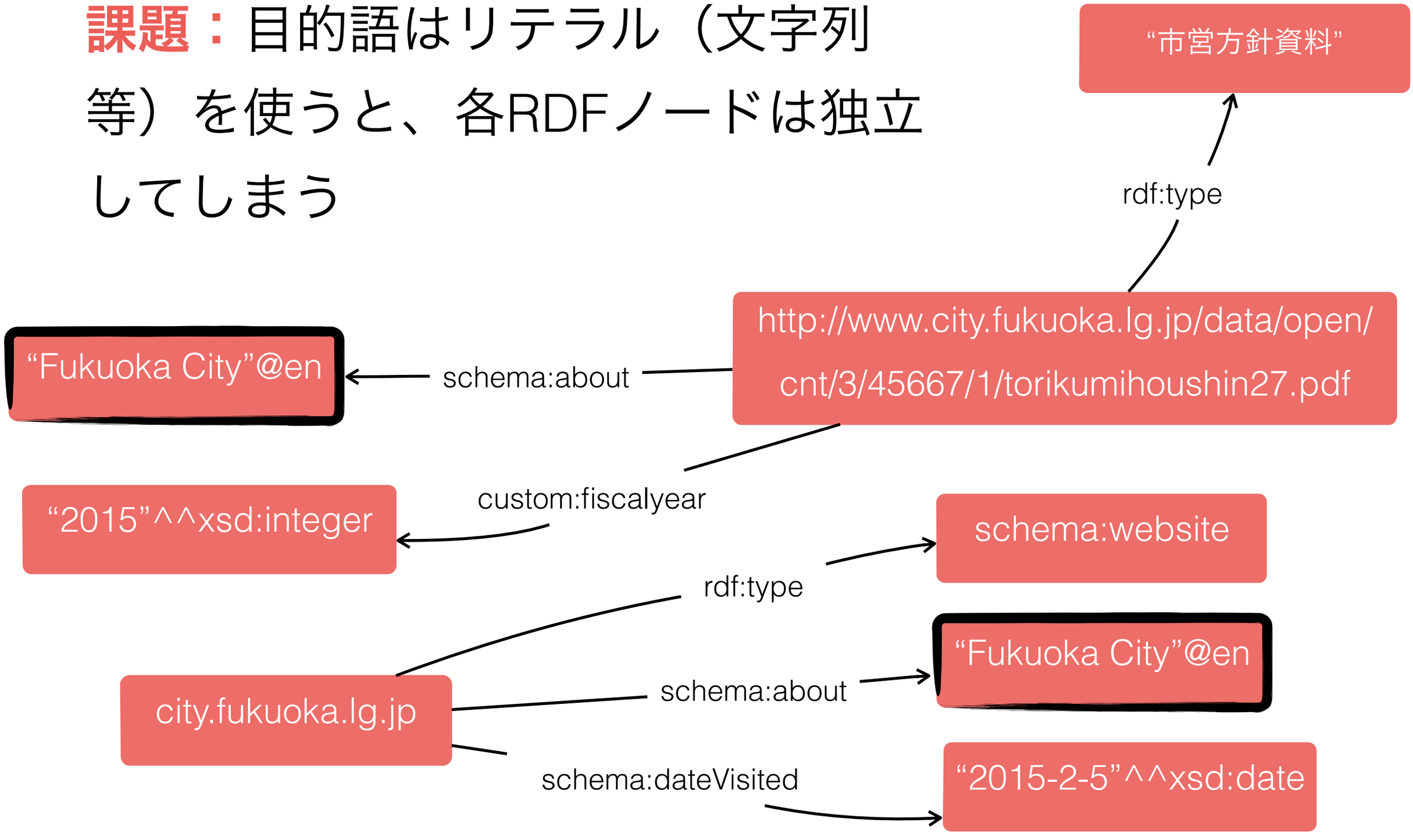
RDFノードとは

- 主語とすべての｛述語、目的語｝
- 下記はRDFノードが２つ



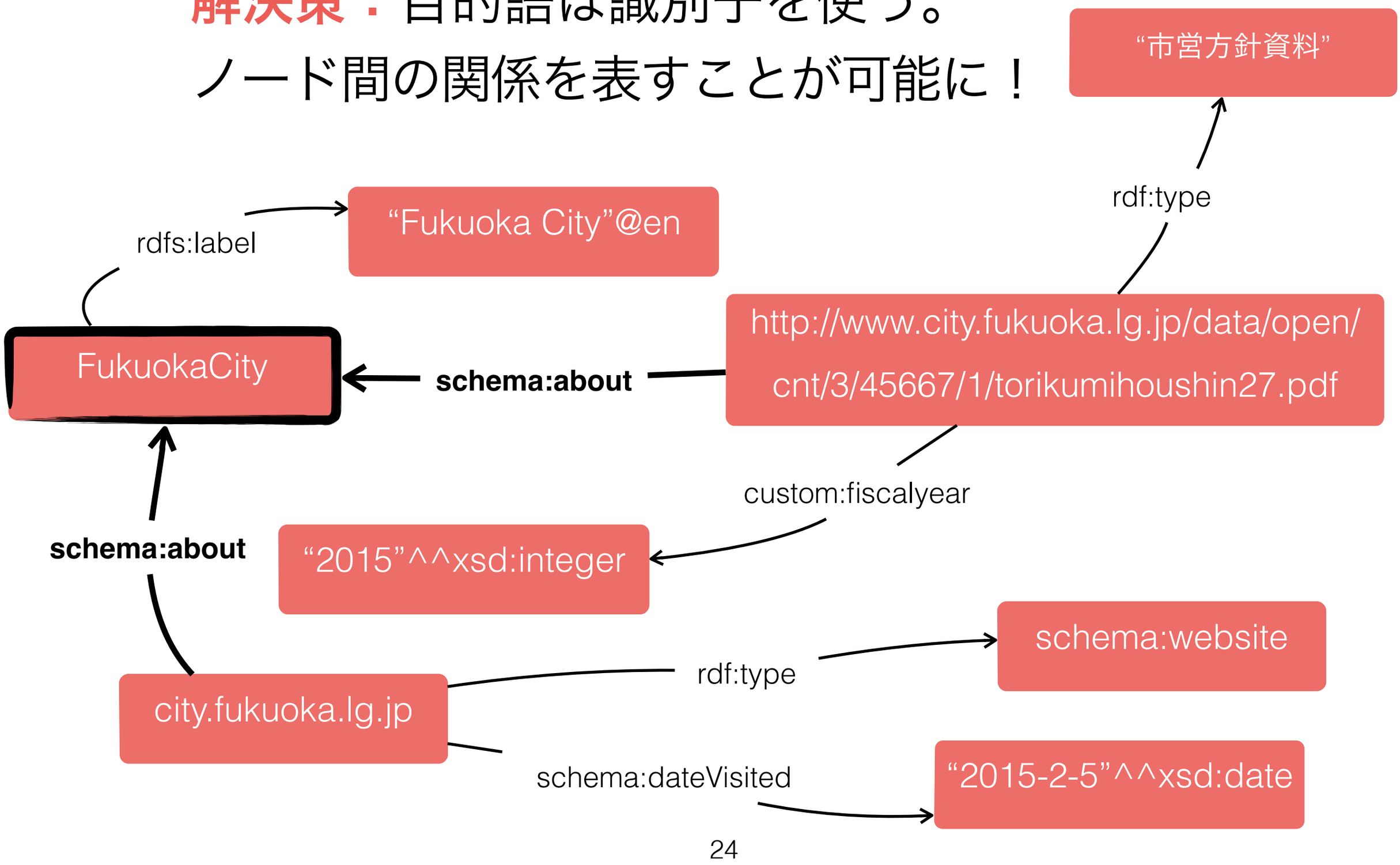
リンクRDF(1)

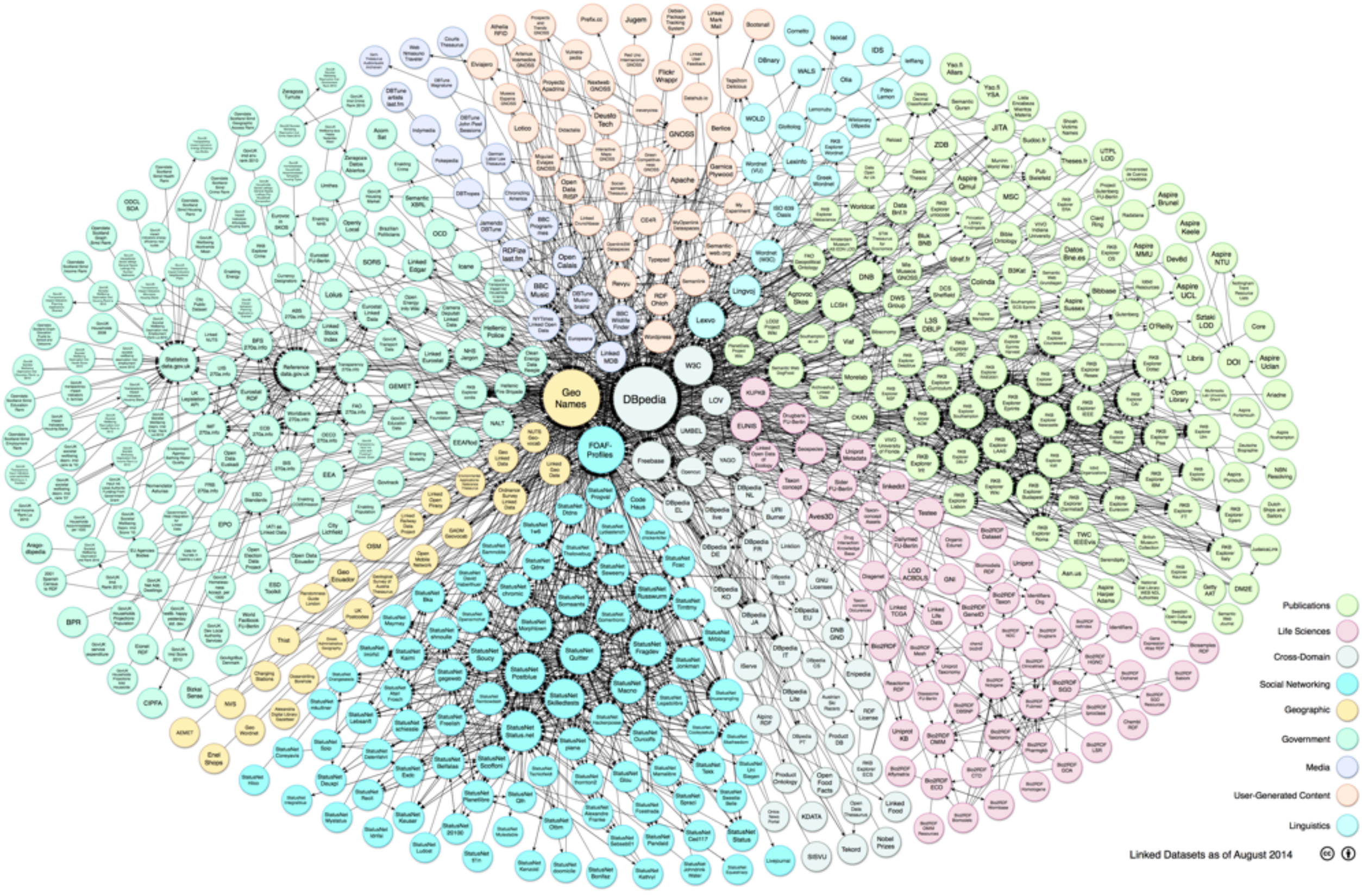
課題：目的語はリテラル（文字列等）を使うと、各RDFノードは独立してしまう



リンクRDF(2)

解決策：目的語は識別子を使う。
ノード間の関係を表すことが可能に！





http://lod-cloud.netより

語彙について

- RDFデータの使っている述語等は出来るだけ統一する必要がある
  - データベース間にリンクを貼るために
  - データベースを探索ために
- よく使われている語彙セットがあります
  - W3Cの標準語彙（少ない！）：RDF語彙集合、RDFスキーマ（RDFS）語彙集合
  - その他：Friend of a friend、Schema.org、Geo
- 人気語彙セットのランキング：http://www.prefix.cc
  - 名前空間（接頭辞）で整理する

自分のパソコンでサイトを見ても大丈夫ですよ！

語彙集合サンプル：RDFS

「RDFS」語彙セット

- RDFスキーマ
- W3C標準語彙
- RDFデータベースの構造を記述するための語彙

RDFデータベースのスキーマはRDFで記述する

prefix.cc

rdfs

http://www.w3.org/2000/01/rdf-schema#

+1
−1

Add alternative URI

ttl xml rdfa sparql txt json jsonld vann | lov | prefix.cc

NUI Galway
OÉ Gaillimh

DERI Galway

Display a menu

w3.org

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

<http://www.w3.org/2000/01/rdf-schema#> a owl:Ontology ;
        dc:title "The RDF Schema vocabulary (RDFS)" .

rdfs:Resource a rdfs:Class ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "Resource" ;
        rdfs:comment "The class resource, everything." .

rdfs:Class a rdfs:Class ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "Class" ;
        rdfs:comment "The class of classes." ;
        rdfs:subClassOf rdfs:Resource .

rdfs:subClassOf a rdf:Property ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "subClassOf" ;
        rdfs:comment "The subject is a subclass of a class." ;
        rdfs:range rdfs:Class ;
        rdfs:domain rdfs:Class .

rdfs:subPropertyOf a rdf:Property ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "subPropertyOf" ;
        rdfs:comment "The subject is a subproperty of a property." ;
        rdfs:range rdf:Property ;
        rdfs:domain rdf:Property .

rdfs:comment a rdf:Property ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "comment" ;
        rdfs:comment "A description of the subject resource." ;
        rdfs:domain rdfs:Resource ;
        rdfs:range rdfs:Literal .

rdfs:label a rdf:Property ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "label" ;
        rdfs:comment "A human-readable name for the subject." ;
        rdfs:domain rdfs:Resource ;
        rdfs:range rdfs:Literal .
```

Display a menu

このファイルは
なに？

RDFテキスト形式

- ここまでの発表ではRDFを抽象的にしか扱ってない
- が、RDFデータはテキスト形式で格納する必要もある
- 標準フォーマットはRDF1.1規格に記載されている：
  - RDF/Json
  - RDF/XML
  - RDF/Turtle（拡張子：ttl）

さっきのファイルの形式

Turtleシンタックス(1)

SPARQLで使いますので、覚えましょう！

- トリプルを効率よく表現できる
  - 特にRDF/XMLよりも文字数がすくなく、読みやすい
- トリプルをそのまま書きます（順序：主語、述語、目的語）
  - ピリオドで区切る「.」
  - リテラルはダブルクオートで区切る「””」／IRIは「<>」で区切る
  - 言語やリテラル型は「@」と「^^」を利用する（RDFグラフと同様）

```
<http://bodic.org/subject1> <http://www.w3.org/2000/01/rdf-schema#label> “名前”@ja.
<http://bodic.org/subject1> <http://www.w3.org/2000/01/rdf-schema#label> “name”@en.
<http://bodic.org/subject1>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://bodic.org/people>.
```

- 「PREFIX」キーワードで名前空間を定義し、IRIを短く書ける

```
PREFIX bodic: <http://bodic.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
bodic:subject1 rdfs:label “名前”@ja.
bodic:subject1 rdfs:label “name”@en.
bodic:subject1 rdf:type bodic:people.
```

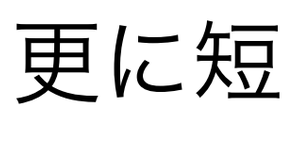
名前空間を使うと「<...>」が不要になる

Turtleシンタックス(2)

- 主語又は述語を結合し、更に短く書ける！
  - 主語の合同：セミコロンで｛述語、目的語｝を区切る
  - 述語の合同：コンマで目的語を区切る

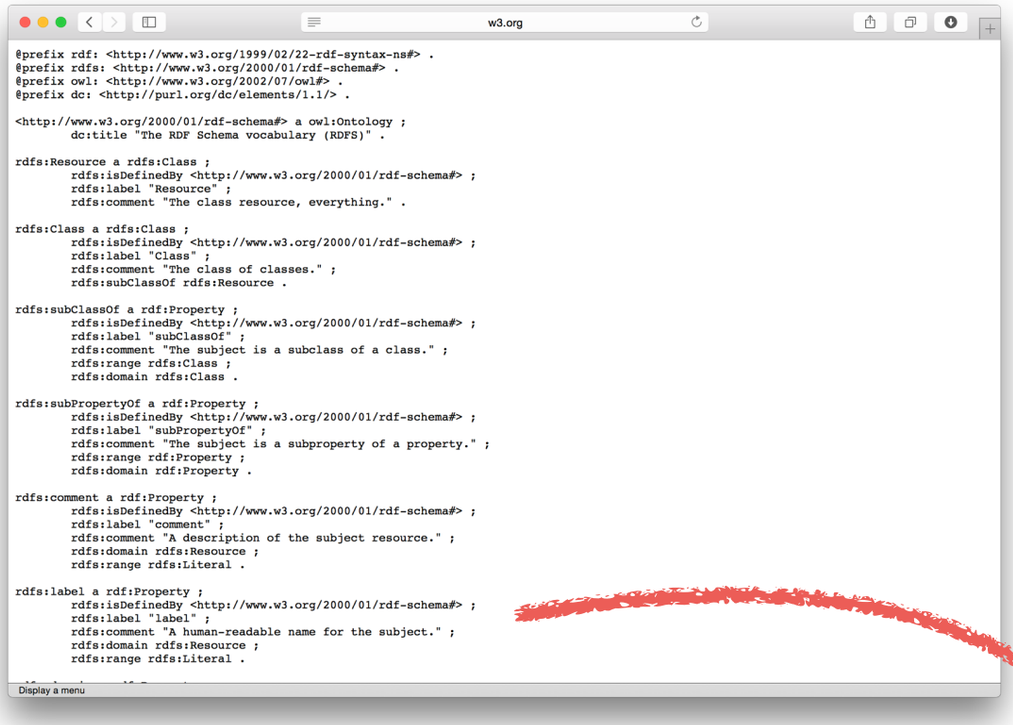
```
PREFIX bodic: <http://bodic.org/>
PREFIX rdf: < http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
bodic:subject1 rdfs:label “名前”@ja.

bodic:subject1 rdfs:label “name”@en.
bodic:subject1 rdf:type bodic:people.
```



```
PREFIX bodic: <http://bodic.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
bodic:subject1
  rdfs:label “名前”@ja,“name”@en;

  rdf:type bodic:people.
```



例：rdfs:label の定義

rdfs:labelという述語を
定義するRDFノード

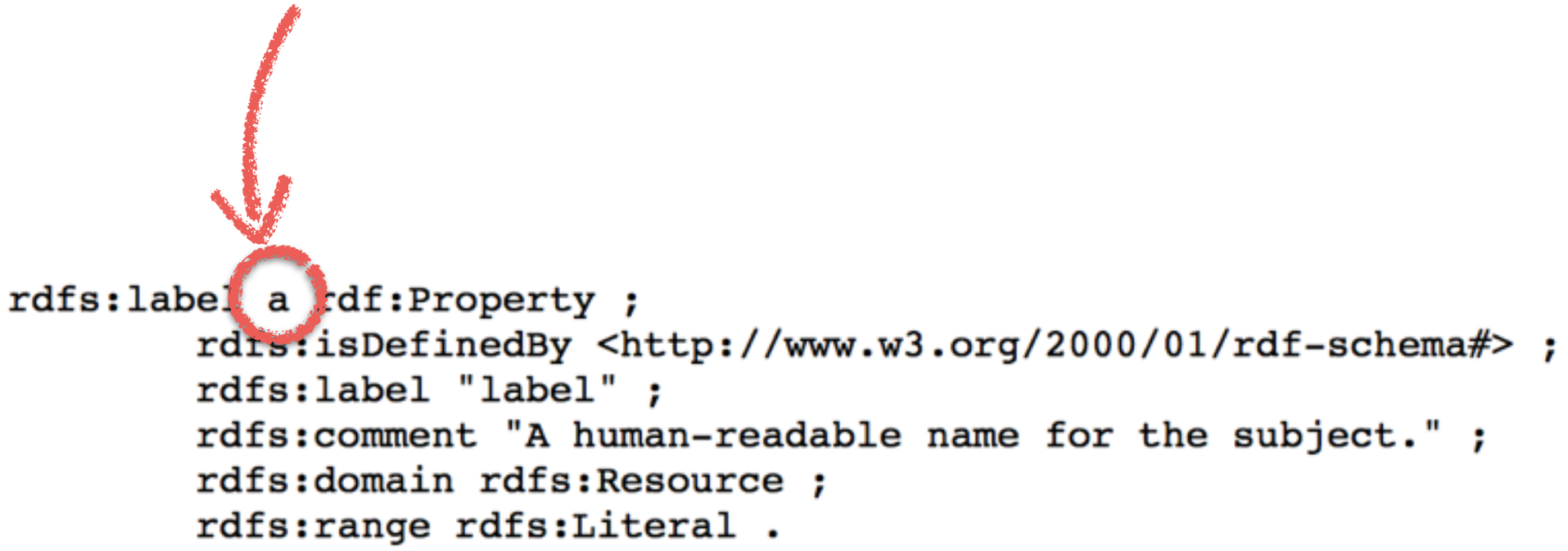
```
rdfs:label a rdf:Property ;
        rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
        rdfs:label "label" ;
        rdfs:comment "A human-readable name for the subject." ;
        rdfs:domain rdfs:Resource ;
        rdfs:range rdfs:Literal .
```

rdfs:label (<http://www.w3.org/2000/01/rdf-schema#label>)
名前を指定するような標準的な述語。

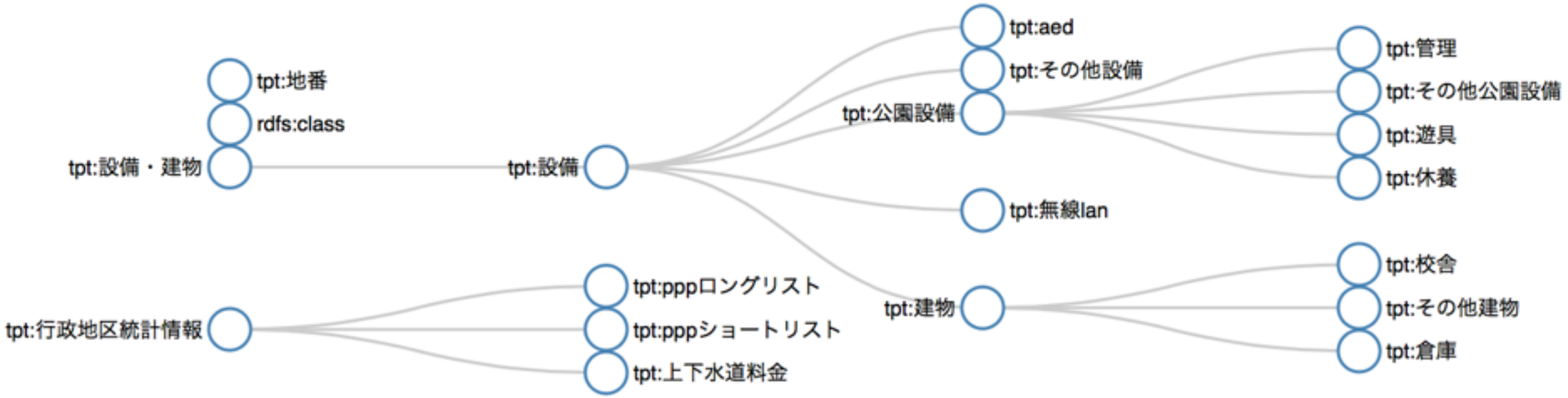
RDFノード型について

「a」キーワード又は「rdf:type」でノード型を指定

- Turtle「a」キーワードは「rdf:type」述語と同義
- RDFスキーマ語彙を使うと、ノード型の親子関係および利用可能述語を定義可能
- RDFで、データと同じデータベースの中にスキーマを定義するとのことです
- スキーマを守るのはデータ登録者の責任である（トリプルストアは確認してくれません）

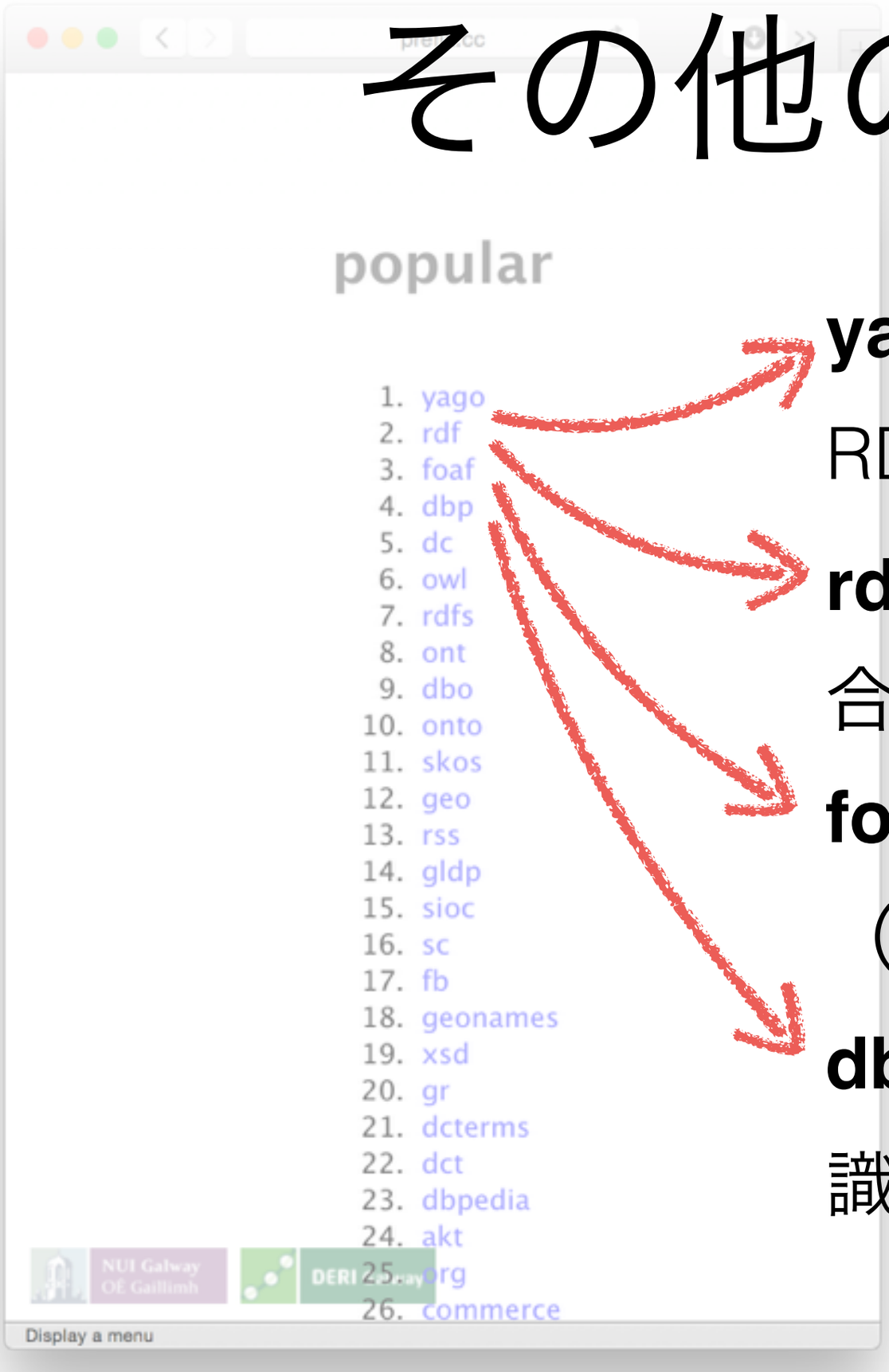


RDFノード型ツリーのサンプル



http://teapot.bodic.org/voc_doc.htmlより

その他の名前空間



yago. Wikipedia等から作られたRDF知識データベース

rdf. W3DのRDF1.1規格の語彙集合

foaf. 人間関係を表す述語（foaf=**F**riend **O**f **A** **F**riend）

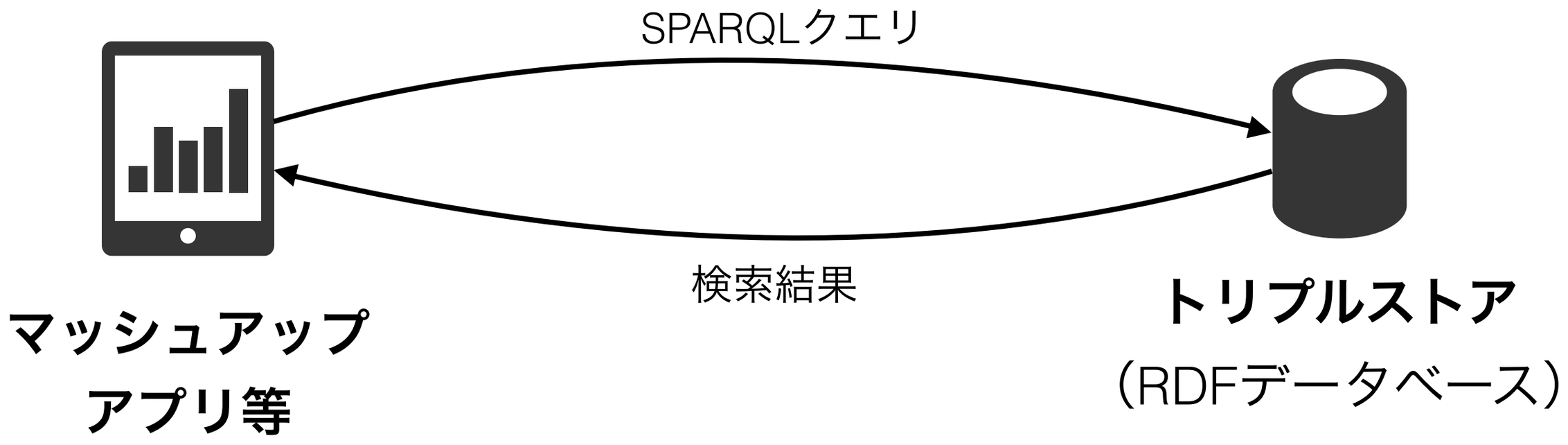
dbpd. Wikipediaから作られた知識データベース（DBペディア）

・・・

本題２ SPARQLクエリの基本

SPARQLとは

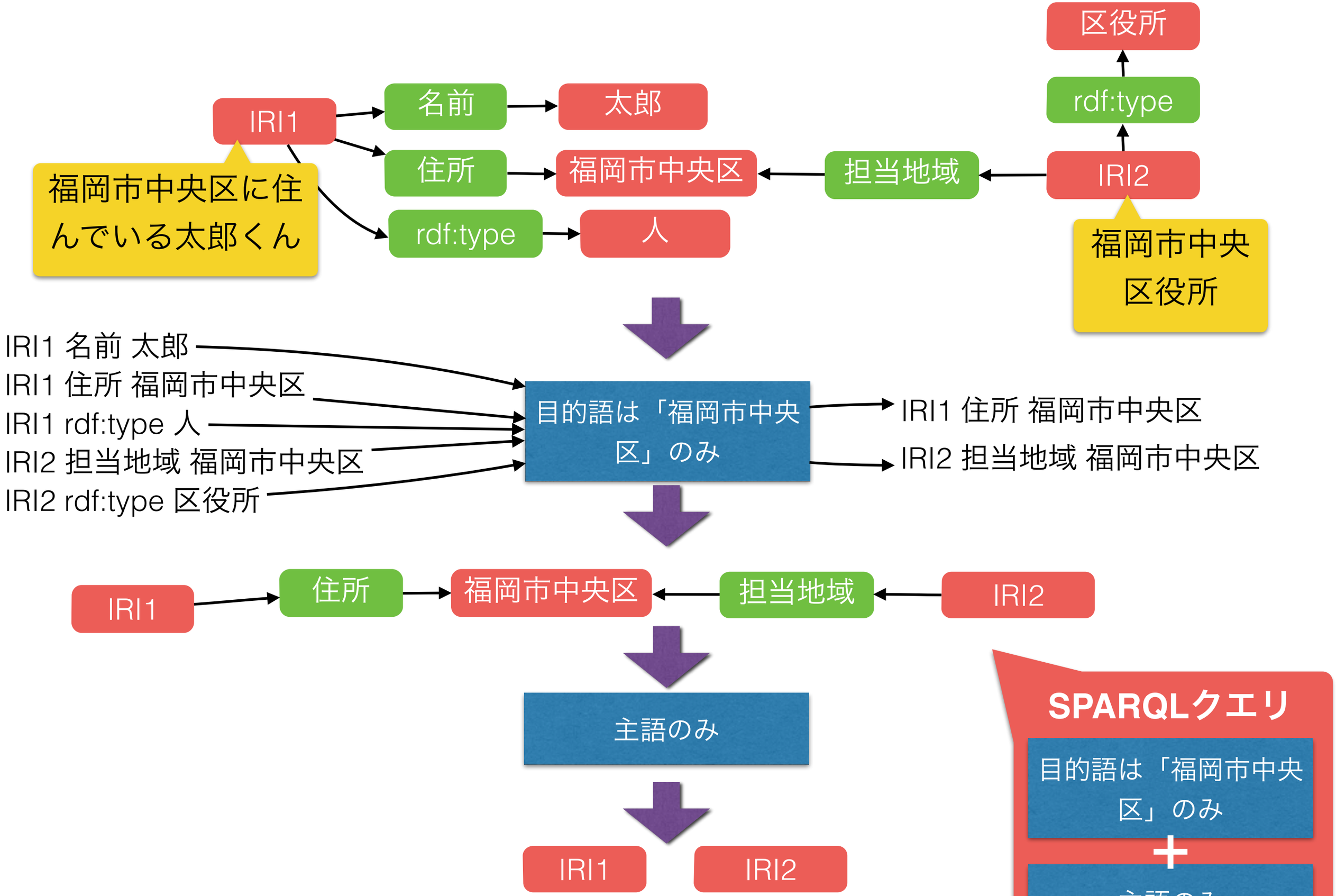
- RDF1.1のクエリ言語。W3Cの標準技術である
- クエリ言語とは
  - データの検索や操作を行うための言語
  - データモデルやデータベースシステムに合わせて設計



SPARQLの概念

- RDFトルプルを検索することで、**RDFグラフをマッチする**
- RDFトリプルを検索して、RDFトルプルを返す
- 検索条件はRDFトリプルで表現する
- RDF/Turtle Syntaxを利用

サンプル：次のスライド・・・


区役所
rdf:type
IRI2
担当地域
福岡市中央区
住所
太郎
名前
IRI1
福岡市中央区に住んでいる太郎くん
rdf:type
人
福岡市中央区役所
IRI1 名前 太郎
IRI1 住所 福岡市中央区
IRI1 rdf:type 人
IRI2 担当地域 福岡市中央区
IRI2 rdf:type 区役所
目的語は「福岡市中央区」のみ
IRI1 住所 福岡市中央区
IRI2 担当地域 福岡市中央区
IRI1
住所
福岡市中央区
担当地域
IRI2
主語のみ
IRI1
IRI2
SPARQLクエリ
目的語は「福岡市中央区」のみ
+
主語のみ

SPARQLシンタックス

- SPARQLクエリは主に２つに分けれる：
  - SELECT部分：返して欲しい変数を指定する（例：主語のみ）
  - 条件部分：検索条件を指定する（例：目的語は「福岡市中央区のみ」）
- RDF/Turtleシンタックスを利用
  - 特にRDF/Turtleで書かれたRDFトリプルで検索条件を表現する
- RDFトリプルの主語・述語・目的語は変数または固定値である
  - 固定値を使うと、条件とする
  - 変数を使うと、選択する
- 変数名は必ず「?」で始まる

リテラル（文字列等）：クゥオートを利用（”...”）
ＩＲＩ：ブラケットを利用（<...>）

目的語は「福岡市中央区」のみ
＋
主語のみ

SELECT部分

```
SELECT ?s
WHERE { ?s ?p “福岡市中央区”}
```

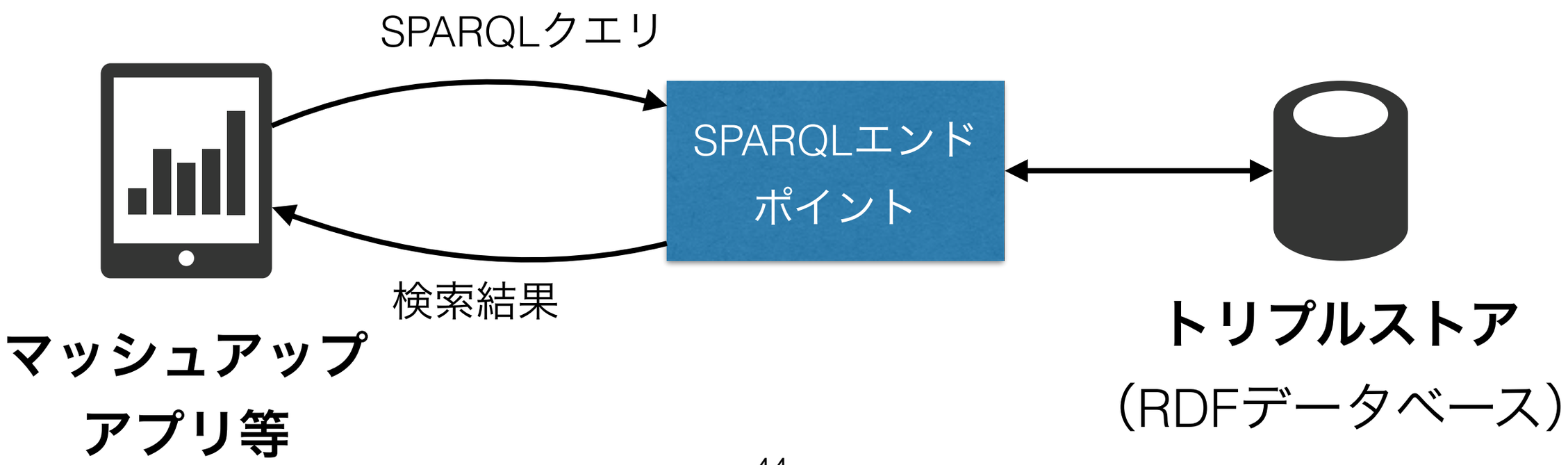
条件部分

関係データベースとの比較

	関係データベース（MySQL等）	RDF
クエリ言語	SQL	SPARQL
データベース	関係データベース	トリプルストア（グラフストア）
データモデル	関係モデル	RDF1.1モデル

SPARQLエンドポイント

- SPARQLクエリを受ける場所
  - 普段はHTTP API経由
  - Webページ上のフォームを提供するエンドポイントも多い（裏はHTTP APIを利用）
- マッシュアップアプリ等がクライエント
  - サーバ側（PHP、Ruby等）
  - クライエント側（AJAX）



今から演習

演習 1 SPARQLを使って

みよう

エンドポイント１

東日本大震災アーカイブ

http://fukushima.archive-disasters.jp/

東日本大震災アーカイブ Ful

SPARQLでさがす

このSPARQLエンドポイントには、東日本大震災アーカイブ Fukushimaメタデータを RDFに変換したメタデータを蓄積しており、SPARQLを利用して検索することがで

①SPARQL検索を選択

PREFIX(名前空間接頭辞)追加:
Dublin Core　DCMI Metadata Terms　NDL Metadata Terms　Data Catalog Vocabulary　Geo vocabulary　GeoNames Ontology　JMP2.0　The RDF Vocabulary (RDF)　The RDF Schema vocabulary (RDFS)　Creative Commons　Friend of a Friend(FOAF)　The Ontology for Media Resources　NDL東日本大震災アーカイブメタデータ　PREMIS　vCard　「東日本大震災アーカイブ」基盤構築事業 福島プロジェクトメタデータ

サンプルSPARQLクエリ:
全検索　全データのタイトルを取得　URIと緯度経度情報を取得

SPARQLクエリ

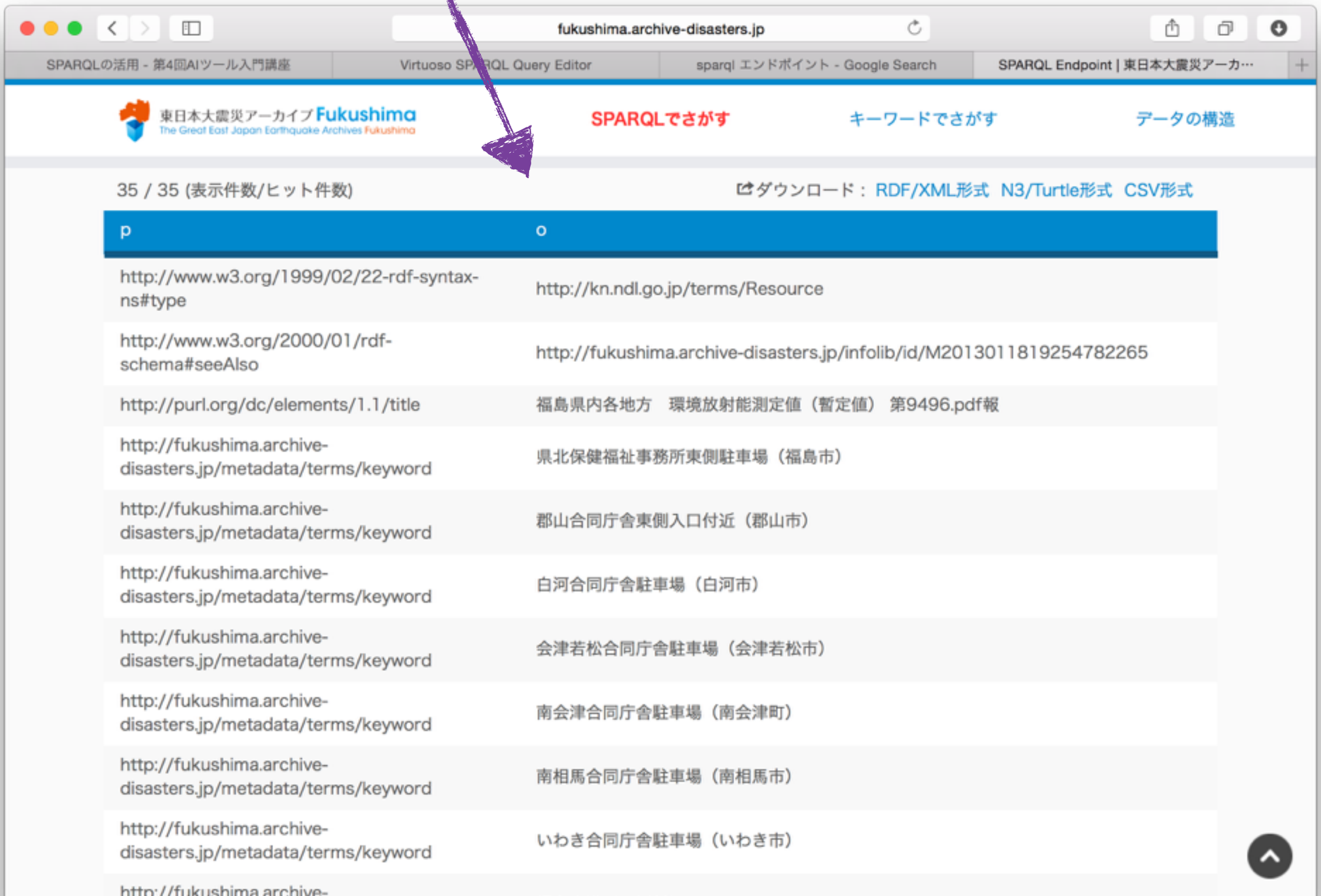
```
1 select ?p ?o { <http://fukushima.archive-disasters.jp/id/resource/M20130118192547822 65> ?p ?o }
```

②SPARQLクエリを入力

SPARQL検索

③SPARQLクエリを実行する

④検索結果を確認する


fukushima.archive-disasters.jp
SPARQLの活用 - 第4回AIツール入門講座
Virtuoso SPARQL Query Editor
sparql エンドポイント - Google Search
SPARQL Endpoint | 東日本大震災アーカ…
東日本大震災アーカイブ Fukushima
The Great East Japan Earthquake Archives Fukushima
SPARQLでさがす
キーワードでさがす
データの構造
35 / 35 (表示件数/ヒット件数)
ダウンロード： RDF/XML形式 N3/Turtle形式 CSV形式
p
o
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://kn.ndl.go.jp/terms/Resource
http://www.w3.org/2000/01/rdf-schema#seeAlso
http://fukushima.archive-disasters.jp/infolib/id/M20130118192547822 65
http://purl.org/dc/elements/1.1/title
福島県内各地方　環境放射能測定値（暫定値） 第9496.pdf報
http://fukushima.archive-disasters.jp/metadata/terms/keyword
県北保健福祉事務所東側駐車場（福島市）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
郡山合同庁舎東側入口付近（郡山市）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
白河合同庁舎駐車場（白河市）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
会津若松合同庁舎駐車場（会津若松市）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
南会津合同庁舎駐車場（南会津町）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
南相馬合同庁舎駐車場（南相馬市）
http://fukushima.archive-disasters.jp/metadata/terms/keyword
いわき合同庁舎駐車場（いわき市）

サンプルクエリを利用

①「すべてのタイトルを取得」を選択

サンプルSPARQLクエリ:

全検索　全データのタイトルを取得　URIと緯度経度情報を取得

SPARQLクエリ

```
1 PREFIX dc: <http://purl.org/dc/elements/1.1/>
2 select ?s ?o from <http://fukushima.archive-disasters.jp/rdf/resource>
3 where {?s dc:title ?o}
4
```

②クエリが自動的に導入される↓

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?s ?o from <http://fukushima.archive-disasters.jp/rdf/resource>
where {?s dc:title ?o}
```

名前空間「dc」を定義

省略可能ですが、述語名などに「http://purl.org/dc/elements/1.1/」の代わりに「dc」を使える様になるため、便利。

クエリ対象ノード識別子の名前空間

指定する名前空間の中しかノードを検索しない（省略可能）

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?s ?o from <http://fukushima.archive-disasters.jp/rdf/resource>
where {?s dc:title ?o}
```

クエリの条件

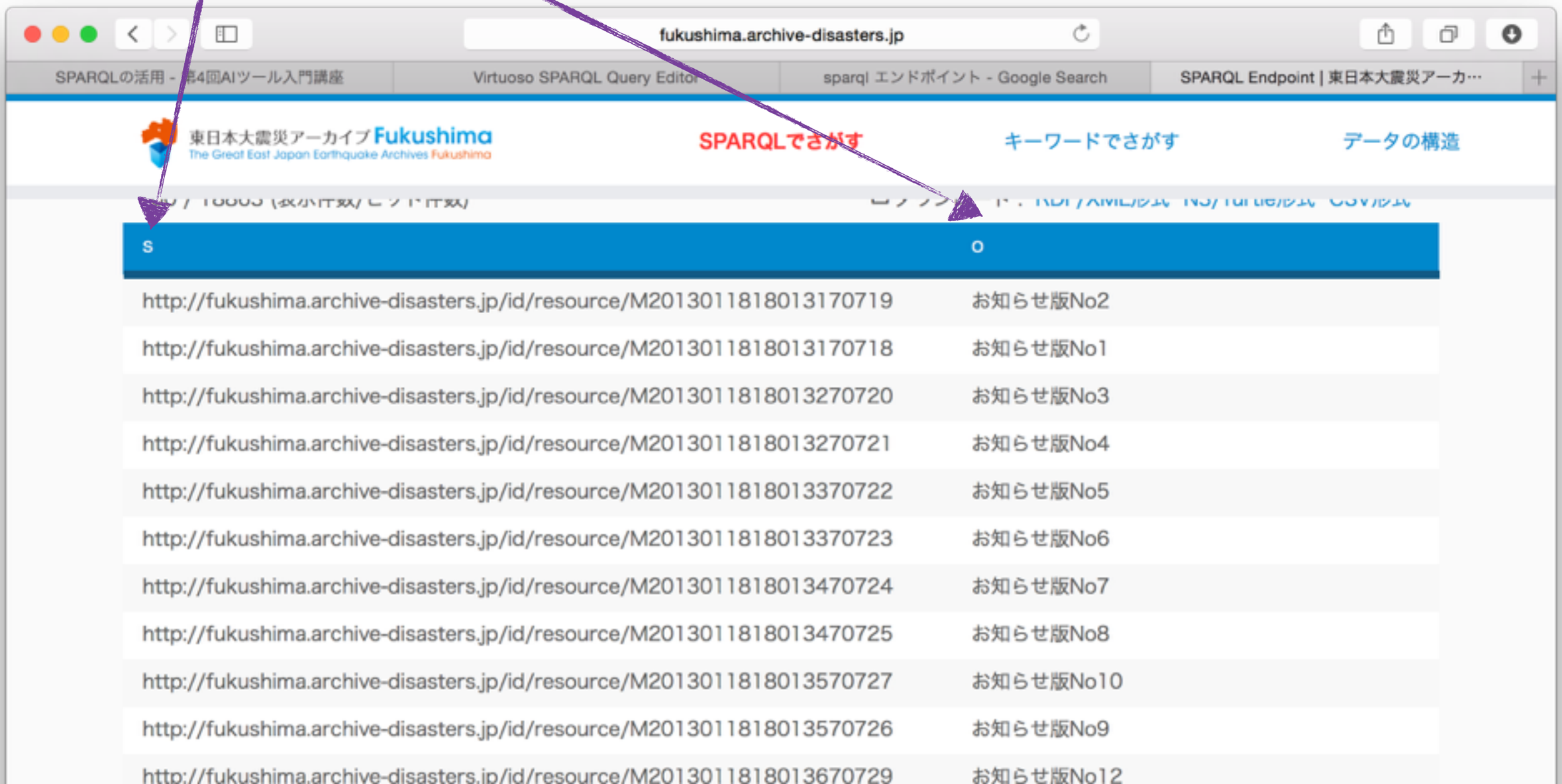
述語が「http://purl.org/dc/elements/1.1/title」のすべてのトルプルを探す

返して欲しい変数

クエリは他の変数を使ってもいいですが、そちらに書いてある変数のみ返される。

サンプルクエリの結果

```
select ?s ?o where { ?s dc:title ?o }
```



s	o
http://fukushima.archive-disasters.jp/id/resource/M20130118180131707 19	お知らせ版No2
http://fukushima.archive-disasters.jp/id/resource/M20130118180131707 18	お知らせ版No1
http://fukushima.archive-disasters.jp/id/resource/M20130118180132707 20	お知らせ版No3
http://fukushima.archive-disasters.jp/id/resource/M20130118180132707 21	お知らせ版No4
http://fukushima.archive-disasters.jp/id/resource/M20130118180133707 22	お知らせ版No5
http://fukushima.archive-disasters.jp/id/resource/M20130118180133707 23	お知らせ版No6
http://fukushima.archive-disasters.jp/id/resource/M20130118180134707 24	お知らせ版No7
http://fukushima.archive-disasters.jp/id/resource/M20130118180134707 25	お知らせ版No8
http://fukushima.archive-disasters.jp/id/resource/M20130118180135707 27	お知らせ版No10
http://fukushima.archive-disasters.jp/id/resource/M20130118180135707 26	お知らせ版No9
http://fukushima.archive-disasters.jp/id/resource/M20130118180136707 29	お知らせ版No12

問題１

タイトルが**「お知らせ版No2」**になっている

ノードについてすべての情報（述語、目的語）

を検索してください

答１

案１：前の検索結果を活かせる

前の結果では「dc:title」が「お知らせ版No2」となっているノードの識別子がありましたので、そのまま使う

案１：
```
select ?p ?o
where {<http://fukushima.archive-disasters.jp/id/resource/M20130118180131707 19> ?p ?o}
```

案２：
```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?p ?o
where {?s dc:title "お知らせ版No2" . ?s ?p ?o }
```

案２：タイトルの文字列から検索

「?s dc:title "お知らせ版No2"」を検索条件として、すべての述語と目的語を検索

各クエーリを説明しましょう・・・

案1クエリの説明

変数「p」と「o」を返す

```
select ?p ?o
where {<http://fukushima.archive-disasters.jp/id/resource/
M20130118180131707 19> ?p ?o}
```

主語を指定する

検索条件のトリプルは主語を指定するので、主語が指定したもののトルプルのみ検索する。述語と目的語は変数を指定するので、絞り込み条件に使わない。

案２クエリの説明

変数「p」と「o」を返す

検索トルプルで使う「s」は返さない。

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
select ?p ?o
where {?s dc:title "お知らせ版No2" . ?s ?p ?o }
```

条件トリプル２：情報取得

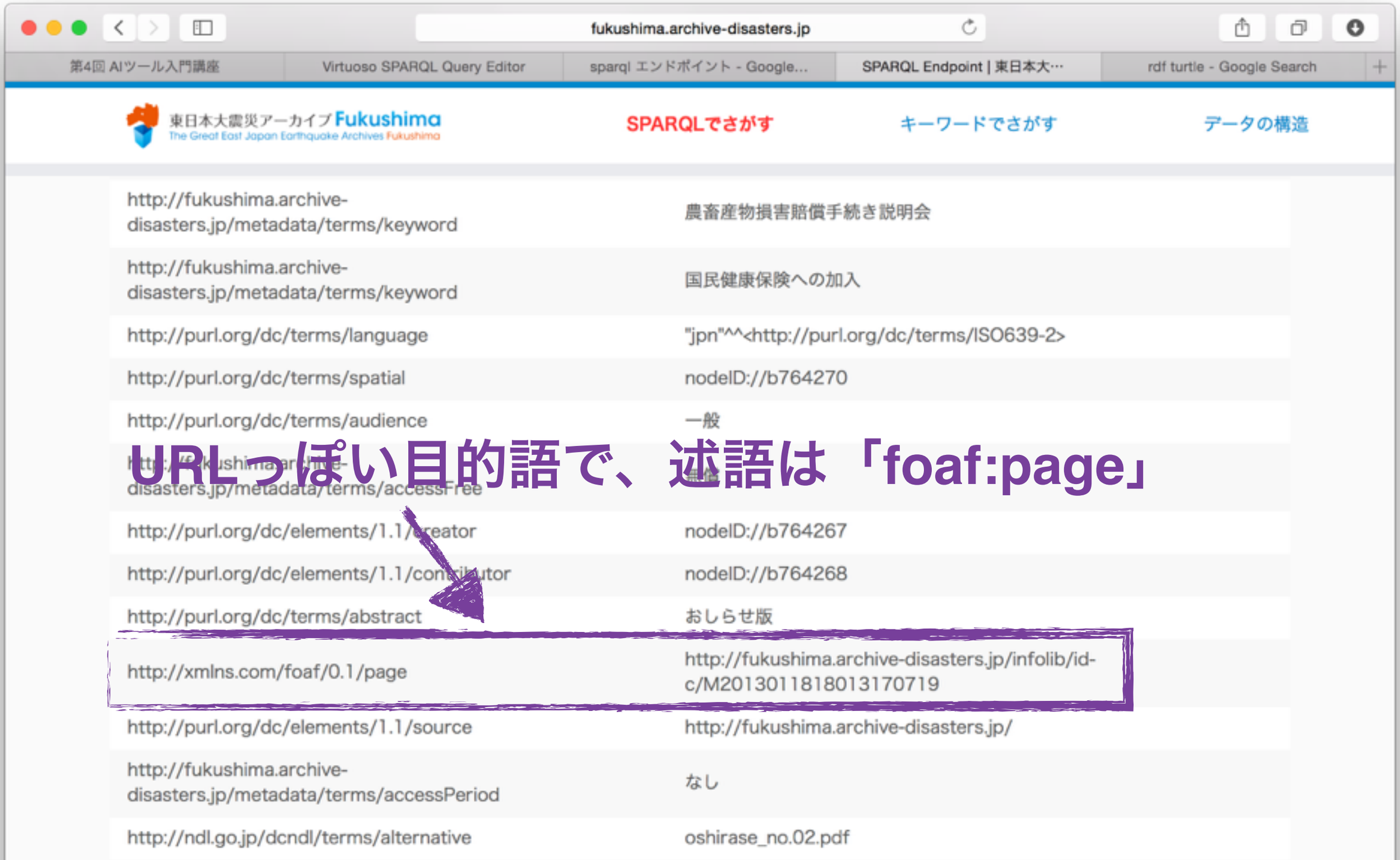
条件トルプル１で選択した「s」に対して、すべての述語と目的語を選択し、それぞれ「p」と「o」に格納。

条件トリプル１：絞り込み条件

述語＝「dc:title」と目的語＝「お知らせ版No2」のトリプルのすべて選択して、それらの主語は変数「s」に格納する。

で？

検索結果を見ると・・・



URLっぽい目的語で、述語は「foaf:page」

URLをアクセスすると・・・

RDFストアの探索について

悩み：初めてSPARQLエンドポイントを見る時に、使っている語彙（述語、ノード型等）が分からない

解決策：とりあえず、語彙の閲覧を出来る様なクエリを投げる！

データベースの中に使われているノード型をリストアップしましょう

下記のクエリーを使うと全てのノードのノード型を検索できる

```
select ?type
where {?s a ?type . }
```

- しかし、重複している。重複をなくすため、「**DISTINCT**」キーワードを利用できる

```
select distinct ?type
where {?s a ?type . }
```

問題２

東日本大震災アーカイブから利用されている
すべての述語をリストアップして下さい

エンドポイント２
公共施設等情報のオープンデータ実証

http://teapot.bodic.org

ニックネームは「Teapot」
（ティーポット）

問題３

まず、データベースの構成を探索しましょう！

URL: teapot.bodic.org/test_api.html

利用されているノード型および述語を閲覧しましょう

「a」キーワードは使えない場合は「rdf:type」述語を直接つかいましょう！

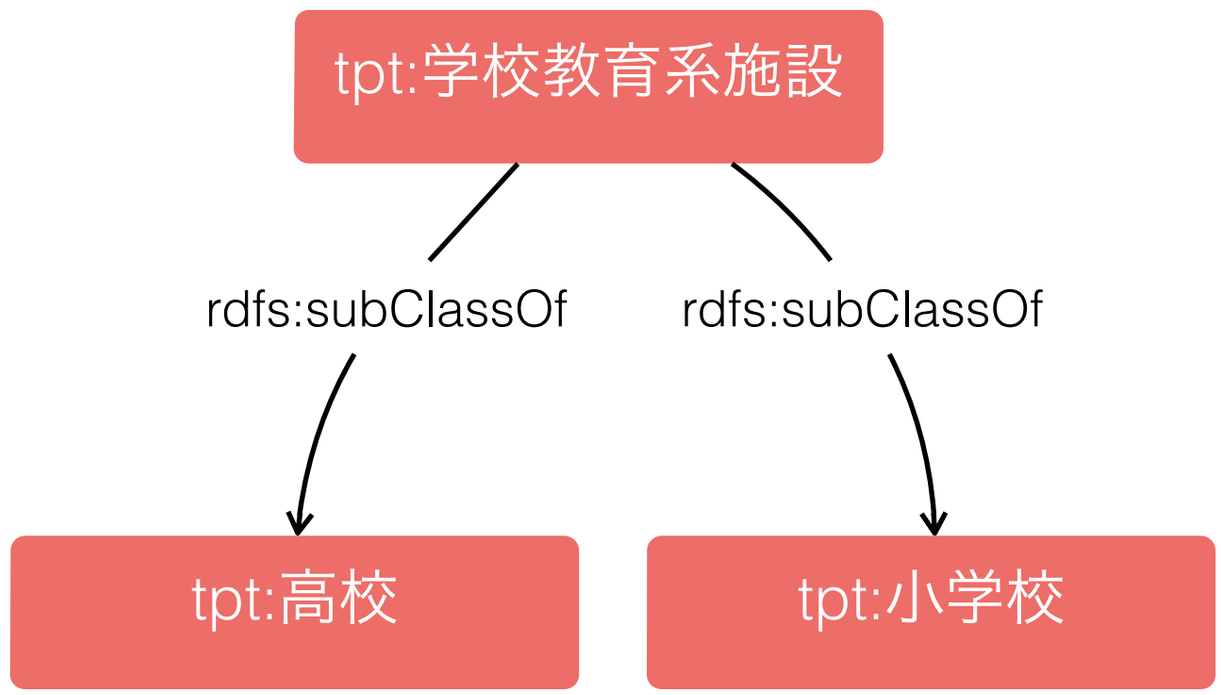
・・・ノード型が沢山あります！

階層的ノード型の探索

TeapotはRDFS語彙を利用して、ノード型の親子関係を表す（述語：「rdfs:subClassOf」）

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?t1 ?t2　{ ?t1 rdfs:subClassOf ?t2.　}
```


```
[
  {
    "t1": {
      "type": "uri",
      "value": "http://teapot.bodic.org/type/小学校"
    },
    "t2": {
      "type": "uri",
      "value": "http://teapot.bodic.org/type/学校教育系施設"
    }
  },
  {
    "t1": {
      "type": "uri",
      "value": "http://teapot.bodic.org/type/高校"
    },
    "t2": {
      "type": "uri",
      "value": "http://teapot.bodic.org/type/学校教育系施設"
    }
  }
]
```

なんでこんなわかり難い
ことをするのか？

演習で実感しましょう

問題４

全ての小学校の名前を検索しましょう

施設名は「rdfs:label」
述語で格納している

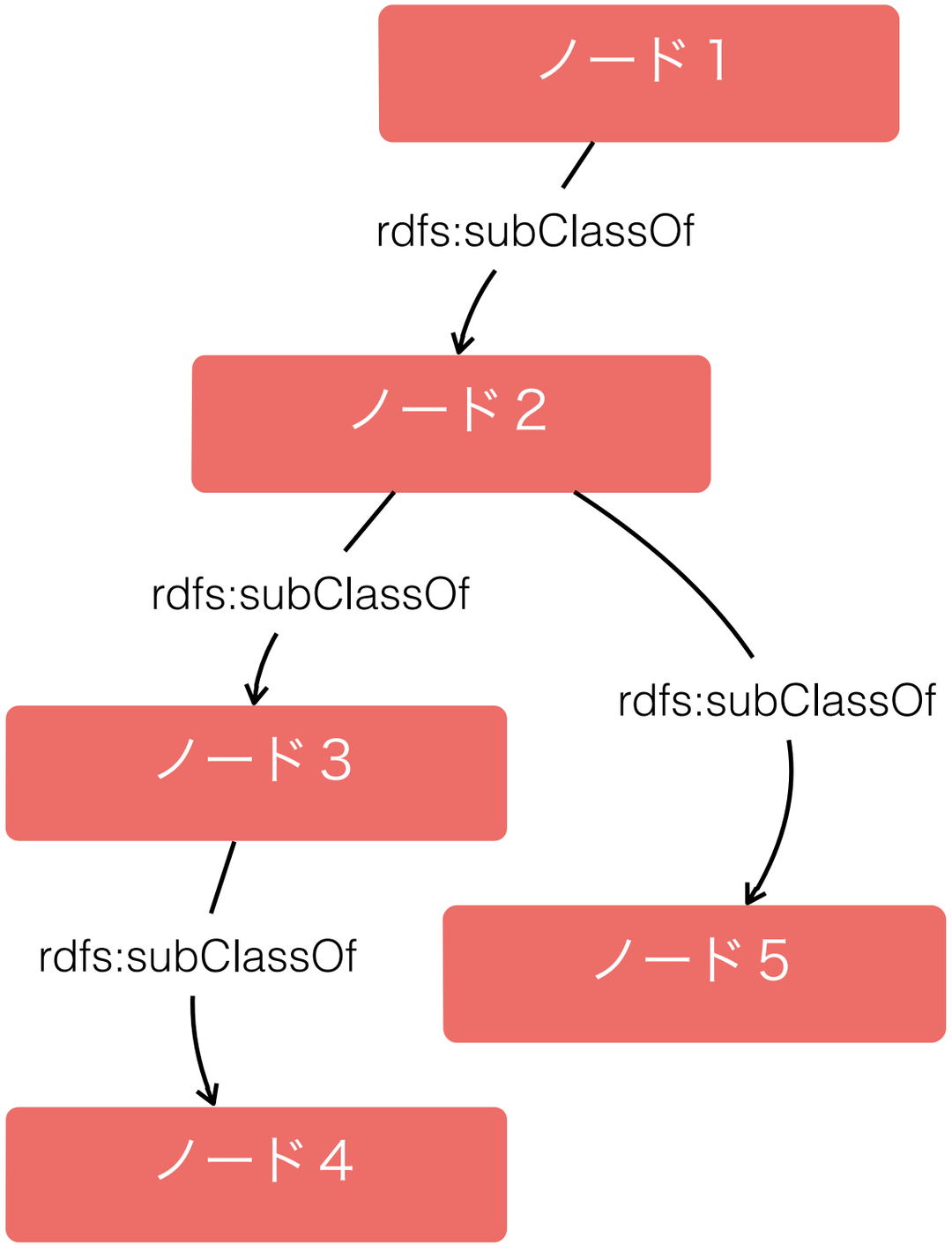
問題５

構造化ノード型を利用して、より正確なクエリを書きましょう！

全ての学校教育系施設の名前を検索しましょう

SPARQLでRDFグラフを辿る

SPARQLの「*」や「+」オペレータを使うと、グラフを辿る



prefix rdfs: <…>
select ?s{ ノード2 rdfs:subClassOf ?s. }
結果：ノード3、5

prefix rdfs: <…>
select ?s{ ノード2 rdfs:subClassOf+ ?s. }
結果：ノード3、4、5

prefix rdfs: <…>
select ?s{ ノード2 rdfs:subClassOf* ?s. }
結果：ノード2、3、4、5

問題６

施設の全てのサブタイプを検索しましょう！

SPARQLで文字列をフィルタをかける

SPARQLで「**FILTER**」キーワードを使うと諸々はフィルターをかけられる

例１（比較演算）：select ?s { ?s foaf:age ?age .

FILTER (age < 10 && age > 5) }

年齢 ∈]5,10[

例２（正規表現）：select ?s { ?s foaf:name ?name .

FILTER (regexp(?name, “taro”)) }

“kentaro”, ”kantaro”, “taro” …

例３（正規表現）：select ?s { ?s foaf:name ?name .

FILTER (regexp(?name, “^山”)) }

○山田、山口

×中山

問題7

中央区にある全ての施設を検索してください。

住所の文字列は「http://teapot.bodic.org/predicate/所在地（文）」述語で格納している

その他の便利なSPARQL(1)

OPTIONAL キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix tpp: <http://teapot.bodic.org/predicate/>
select ?name ?address
{ ?s rdfs:label ?name ; OPTIONAL { tpp:所在地（文）?address } }
```

「tpp:所在地（文）」述語がなくても、
「rdfs:name」さえあれば合致する

その他の便利なSPARQL(2)

UNION キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix tpt: <http://teapot.bodic.org/type/>
select ?name
{
  ?s rdfs:label ?name.
  { ?s rdf:type tpt:小学校 } UNION { ?s rdf:type tpt:高校 }
}
```

高校と小学校の名前を検索する

その他の便利なSPARQL(3)

COUNT キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
select COUNT(?name)
{
  ?s rdfs:label ?name.
}
```

rdf:labelのあるノードの数を返す

その他の便利なSPARQL(4)

GROUP BY キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
select COUNT(?name)
{
  ?s rdfs:label ?name ; rdf:type ?type
} group by ?type
```

各タイプ毎にrdfs:labelのある
ノードの数を返す

その他の便利なSPARQL(5)

LIMIT / OFFSET キーワード

```
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
select COUNT(?name)
{
  ?s rdfs:label ?name ; rdf:type ?type
} group by ?type offset 50 limit 10
```

合致するデータの中から５０個目から１０個を返す（５９個目まで）

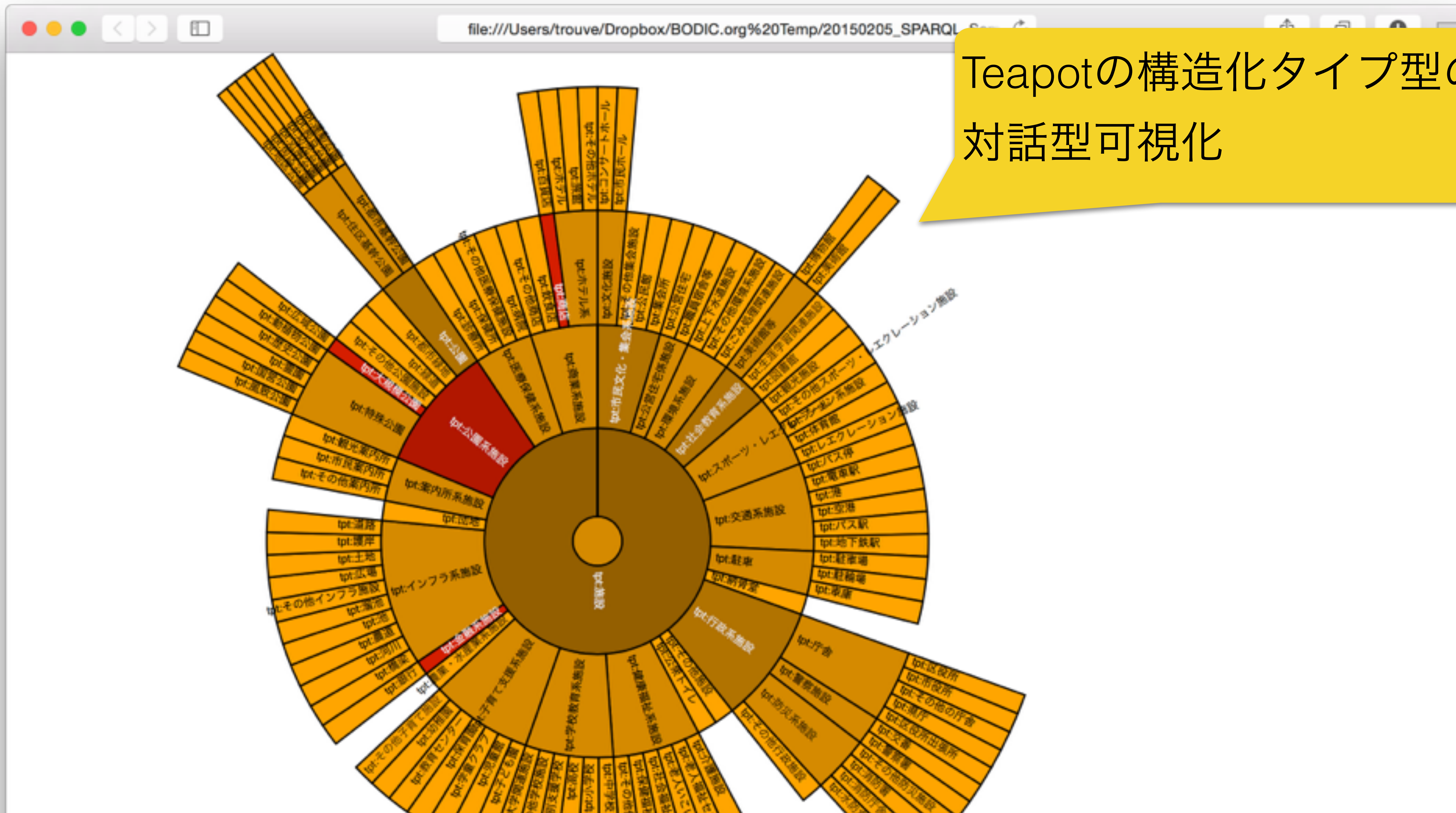
演習２初めてのWebアプリケーション

演習について

- 演習の教科書下記のリンクにあります：
  - http://www.bodic.org/?p=119
- 各演習はファイル３つあります：
  - stepX.html：ページの構成
  - stepX.js：ジャヴァスクリプトソースコード
  - stepX.css：スタイルシート
- 基本的にStepXはStep(X-1)のファイルを書き加えると想定する
  - できるだけ自分で考えてコードを書いてください
  - 分からない時のみ教科書からコピーペーストできる

一番編集することが多い

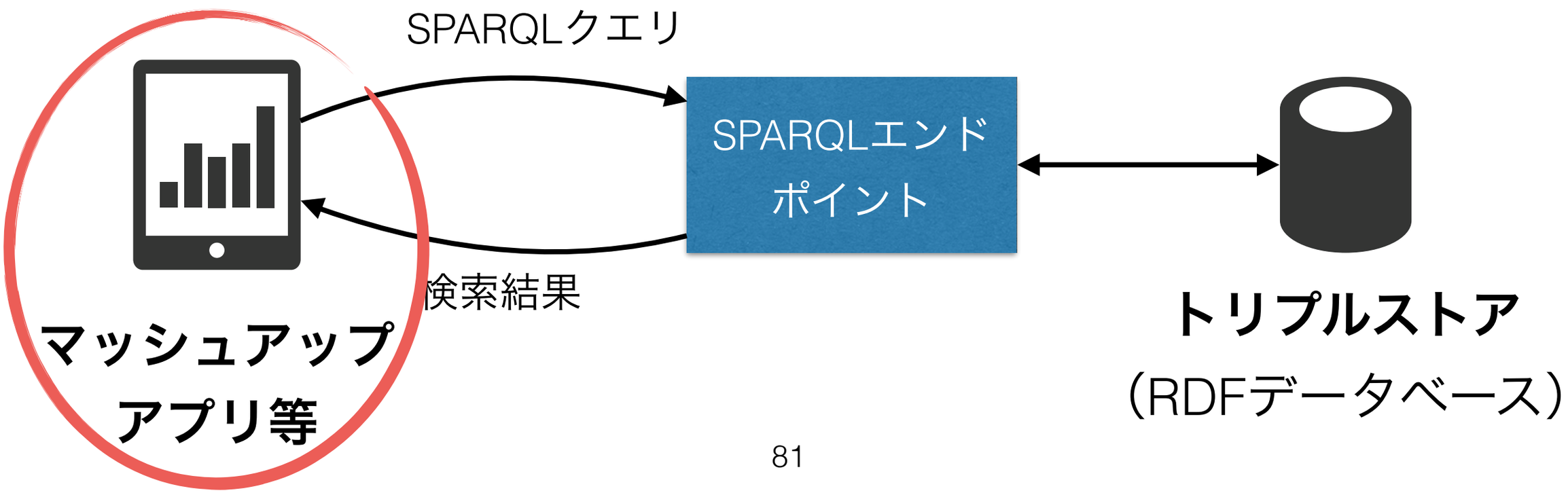
この演習で出来上がるもの



Teapotの構造化タイプ型の対話型可視化

Webアプリケーションとは

- Web Browserで利用できるアプリケーション
- クライエント側で利用する技術
  - 言語はHTML、CSS、Javascript
  - その他：AJAX（次のスライドで説明する・・・）
- サーバ側は今日しません



AJAXについて

Asynchronous **J**avascript **A**nd **X**ML

- クライエント側から非同期的に遠隔内容をアクセスするような技術

- 本演習はAJAX経由でクエリを投げて、動的にWebページ内容を更新する

JQueryについて

- Webブラウザがjavascript経由で提供している機能を簡単に利用できるようなライブラリ
- 本演習では下記の機能を活躍する：
  - DOMエレメントへの選択（セレクタ）
  - AJAX機能（POST関数）
  - DOMエレメントの内容を変更
- Jquery機能は「$」オブジェクトから利用できる
  - DOMエレメント選択：$(セレクタ)
  - AJAX POST：$.post(…)
  - DOMエレメント内容変更：$(…).append()

Document **O**bject **M**odel (**DOM**)

HTML等で使われているエレメントの規格および操作APIの仕様

演習２ Step 1

SPARQLの検索結果を取得しよう

http://www.bodic.org/?p=242

教科書の指示に基づいて、Teapotサーバへクエリを投げて、得られた結果はページに表示しましょう

例えば、全てのノード型親子関係を表すトリプルを取得する

クエリー結果を一部ずつロード

- 課題：データが大量に合致した場合は、結果は数メガを超えます
  - サーバ側はGzip圧縮使うことも多いですが、ブラウザによってサポートしていない
- 解決策：データを一気にダウンロードするではなく、一部ずつ行う
- 実装方法：SPARQLの「limit」と「offset」キーワードを利用する
- 例：

```
select ・・・　 where {・・・}
offset 1000 limit 500
```

1000個目から１５００個目まで返す

演習２ Step２

検索結果を分割して取得しよう

教科書：http://www.bodic.org/?p=249

演習２Step１を書き加えて、検索結果を分割して
１０個ずつ取得するように書き換えてください。

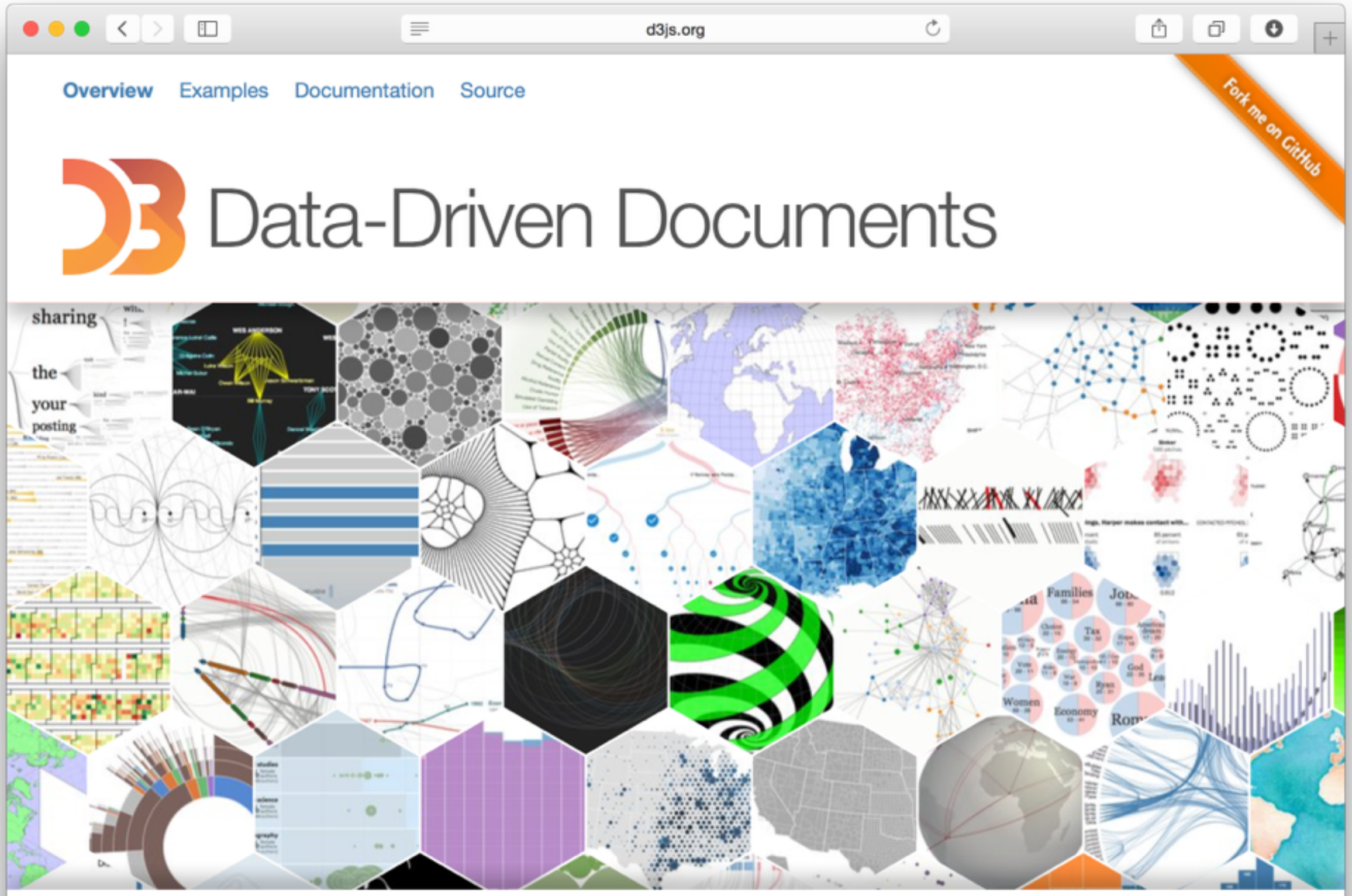
演習２ Step３

特定のデータを表示させよう

教科書：http://www.bodic.org/?p=276

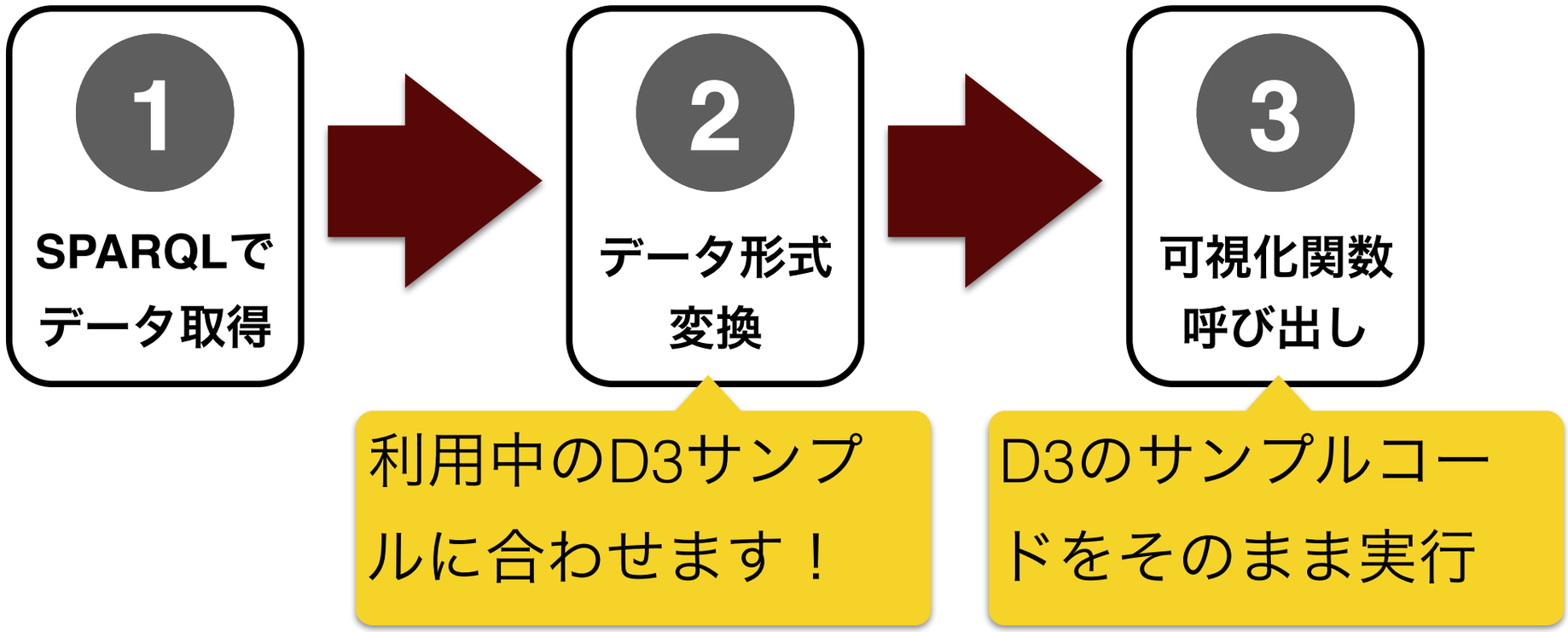
演習２Step２を書き加えてに、Teapotデータベースで利用されているノード型識別子を表示してください（生JSONではなく）。１０個ずつ取得してください。

D3について

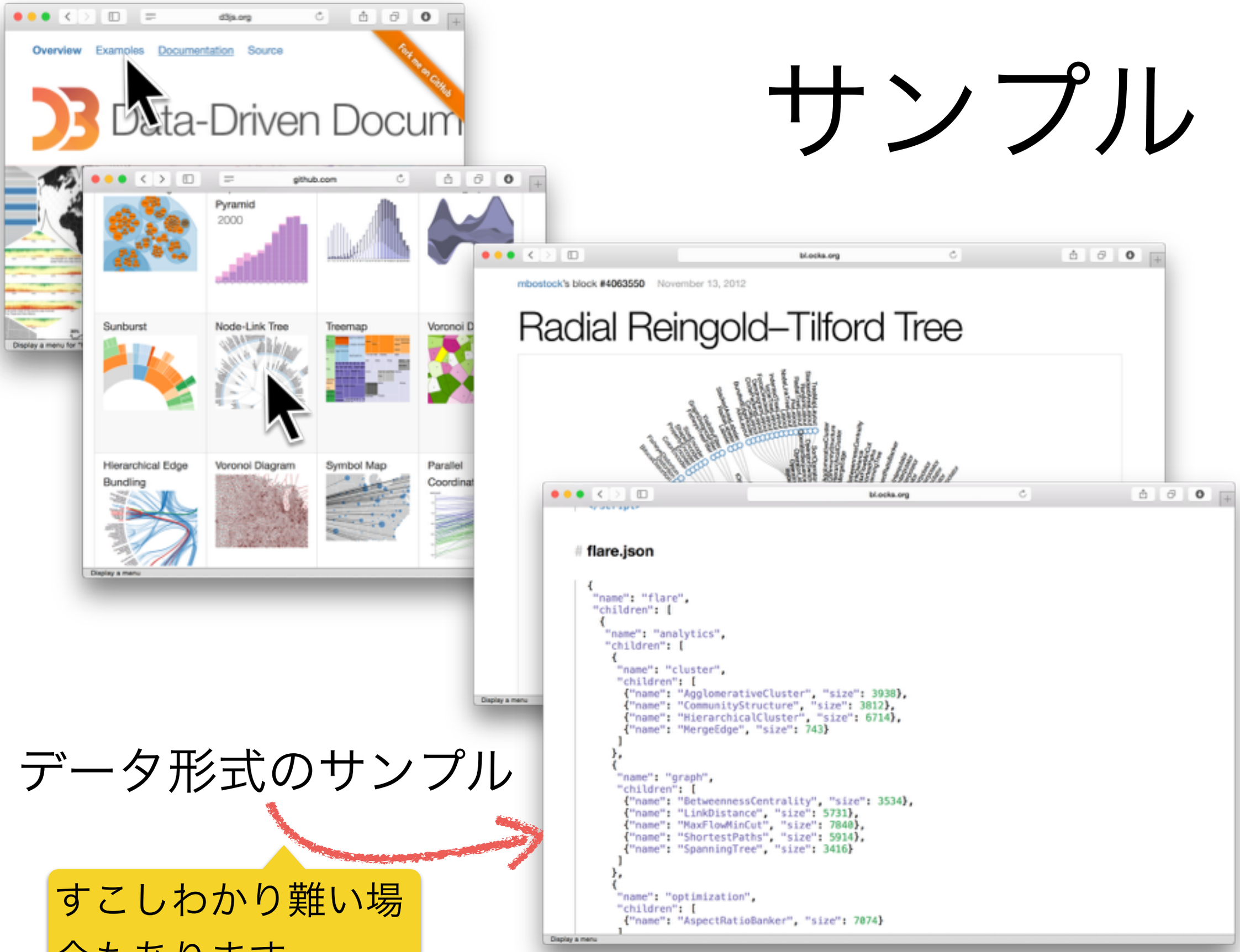

d3js.org
Overview
Examples
Documentation
Source
Fork me on GitHub
Data-Driven Documents

D3の使い方について

- D3のサイトに諸々なサンプルが置いてあります。そのままコピペしてもいいです！



サンプル



データ形式のサンプル

すこしわかり難い場合もあります・・・

演習２ Step４

SPARQLの検索結果を可視化しよう

教科書：http://www.bodic.org/?p=278

演習２Step３を書き加えて、Teapotのノード型を可視化しましょう！タイプは１０

「type_sunburst.lib.js」では、定義している下記の関数をご利用ください：

- データ変換：`makeTypeTree(sparql_res)`
- グラフ生成：`sunburst(selector, json)`

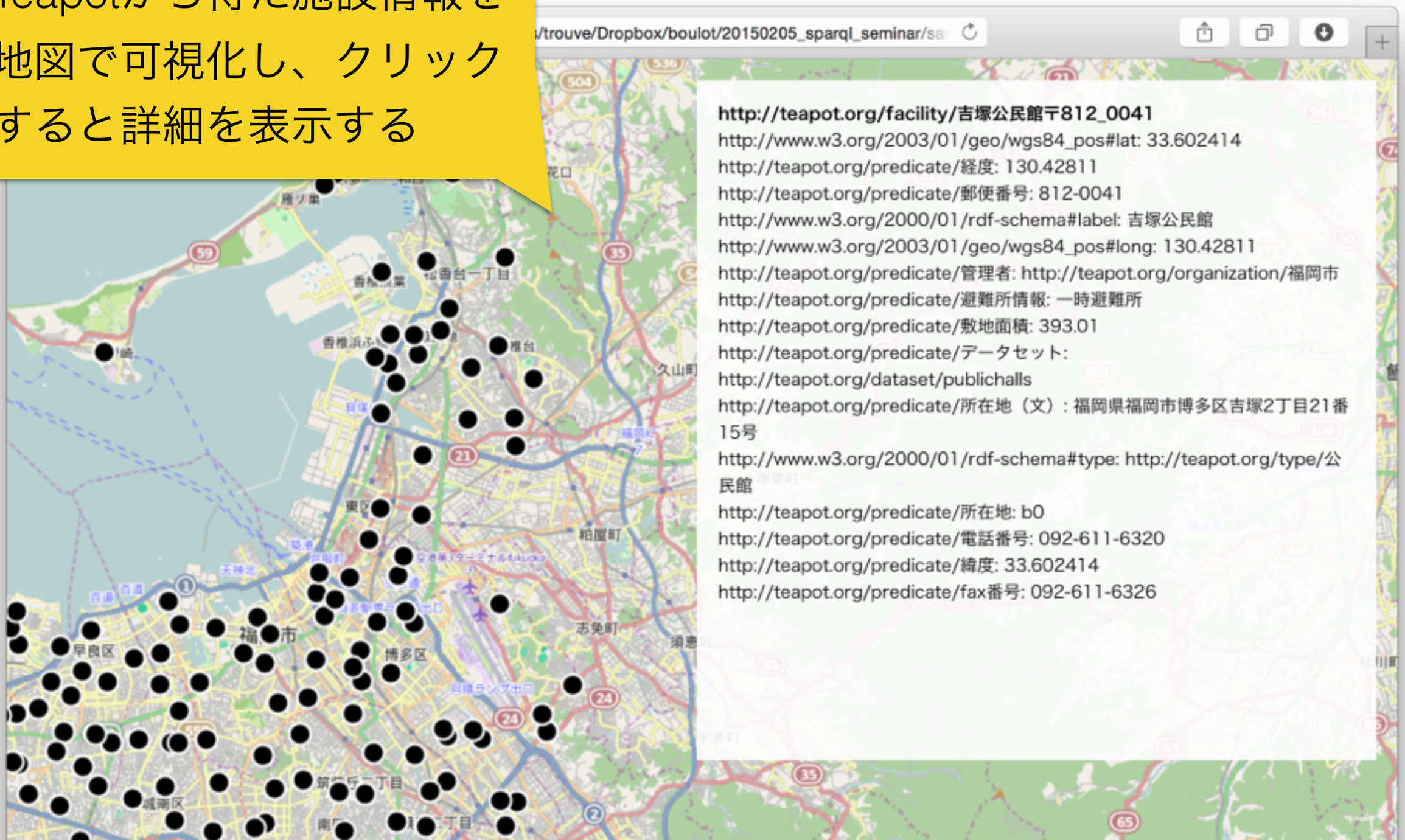
クエリ変数名：
- 親クラス：?superclass
- 子クラス：?class

「json」は「makeTypeTree」の出力

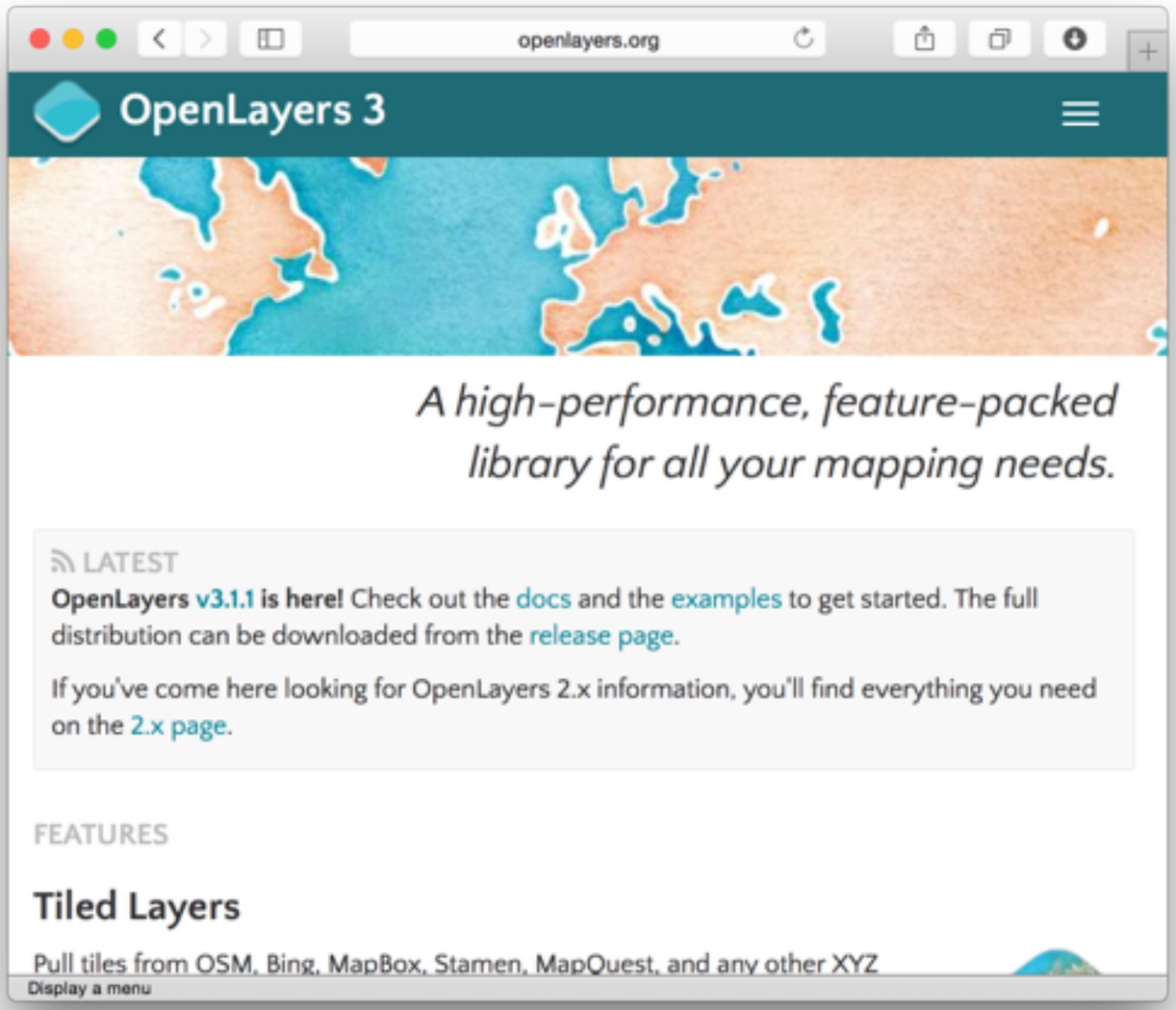
演習３地図アプリを作りましょう

この演習で出来上がるもの

Teapotから得た施設情報を地図で可視化し、クリックすると詳細を表示する



地図表示：OpenLayers3



- レイヤー管理・表示ライブラリ
- 今回利用するレイヤー種
  - 地図レイヤー：地図を表示する
  - ベクターレイヤー：アイコンやベクター形式地図データを表示する

今回のユースケース

上：ベクタレイヤー

下：地図レイヤー

今回のユースケース

上：ベクタレイヤー

下：地図レイヤー

オープンレイヤーの初期化(1)

OL3オブジェクトを生成する

対象DOMエレメントのID

各レイヤーの初期化

```
var map = new ol.Map({
  target: "地図",
  layers: [
    new ol.layer.Tile({
      source: new ol.source.OSM()
    }),
  ]
});
```

今回は地図タイルレイヤーを使う。利用可能なライヤー種はAPI説明へ↓

http://openlayers.org/en/v3.1.1/apidoc/

地図タイルについて

- GoogleMap等が表示する地図画像は資格単位で整理している：タイルという
- 表示する時にタイルが一個ずつ取得する
- ネットワークが遅い時に見える

今回はOpenStreetMapが提供している地図タイルを利用する

オープンレイヤーの初期化(2)

```
var map = new ol.Map({
  target: "地図",

  layers: [...],
  view: new ol.View({
    center: ol.proj.transform([経度, 緯度], 'EPSG:4326', 'EPSG:3857'),

    zoom: 10
  }),
});
```

表示する場所と拡大ファクターの設定

世界座標での経度・緯度を指す標準的な番号

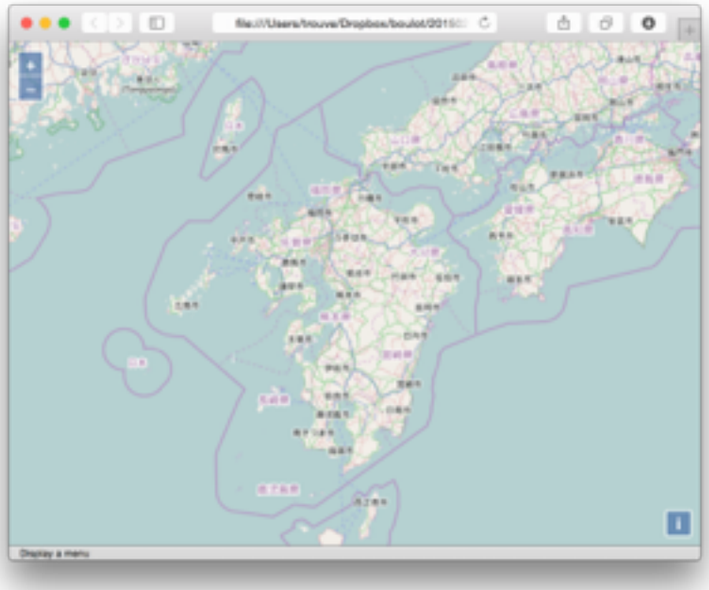
演習３ Step 1

地図データを表示しよう

教科書：http://www.bodic.org/?p=121

教科書の指示に基づいて、地図を作成し、福岡周辺を表示してください。

福岡市の緯度・経度は
Wikipedia等にありますよ！



地図上にピンを表示しましょう！

上：ベクターレイヤー

下：地図レイヤー

フィチャーについて

- ピンはOpenLayersの言葉で「フィチャー」というオブジェクトの1種類です。
- フィチャはベクターレイヤーで表示する
- フィチャーの主な属性は
  - ステイル（見た目）
  - 位置（緯度・経度）
- 今回は黒い円形を表示する

ベクターレイヤーの初期化

```
var vectorSource = new ol.source.Vector({ features: [] });
var map = new ol.Map({
  target: "地図",
  layers: [
    new ol.layer.Tile({
      source: new ol.source.OSM()
    }),
    new ol.layer.Vector({
      source: vectorSource
    })
  ],
  view: new ol.View({...}),
});
```

フィチャー格納オブジェクト

ベクターレイヤー

フィチャーの導入方法

```
var pinStyle = new ol.style.Style({
  image: new ol.style.Circle({
    radius: 8,
    fill: new ol.style.Fill({color: 'black'}),
    stroke: new ol.style.Stroke({color: 'white', width: 2})
  })
});
```

①スタイルオブジェクトを作成

再利用できる！

```
var iconFeature = new ol.Feature({
  geometry:  new ol.geom.Point(
    ol.proj.transform( [経度,緯度], 'EPSG:4326', 'EPSG:3857' )

  )
});
iconFeature.setStyle(pinStyle);
```

②フィチャーオブジェクトを作成

```
vectorSource.addFeature(iconFeature)
```

③フィチャー格納オブジェクトに追加

自動的にベクタレイヤが更新される。

演習３ Step２

地図にピンを表示させよう

教科書：http://www.bodic.org/?p=123

演習３Step１を書き加えて、地図上にTeapotにある公民館を地図上で黒い円形で表しましょう！

・フィチャスタイルオブジェクトを作成する

・SPARQLで公民館の緯度・経度を取得する（述語は「tpp:緯度」と「tpp:経度」）

・検索結果毎にフィチャーオブジェクトを作成し、フィチャー格納オブジェクトに追加する

対話機能：クリック

- OpenLayersはイベントベースでクリック等を処理する
- 例えば、クリックを処理するための「map.on(“click”, …)」を利用する：

```
map.on('click', function(evt) {
  // クリックした位置にFeatureがあるかどうかを確認
  var feature = map.forEachFeatureAtPixel(evt.pixel,
      function(feature, layer) {
        return feature;
      });
}
```

今回1個目のフィチャーしか処理しない

クリックしたところでフィチャーがあるかどうかを確認

フィチャー情報格納

```
var feature = new ol.Feature({
  geometry: new ol.geom.Point(
    ol.proj.transform( [lon,lat], 'EPSG:4326', 'EPSG:3857' )
  ),
  longitude: ...,
  latitude: ...,
  iri: ...
});
```

緯度・経度・
識別子を格納

フィチャー作成時に
情報を格納できる！

```
feature.get("iri")
feature.get("longitude")
```

後は「feature.get()」を使う
と、格納した情報を読み出せる

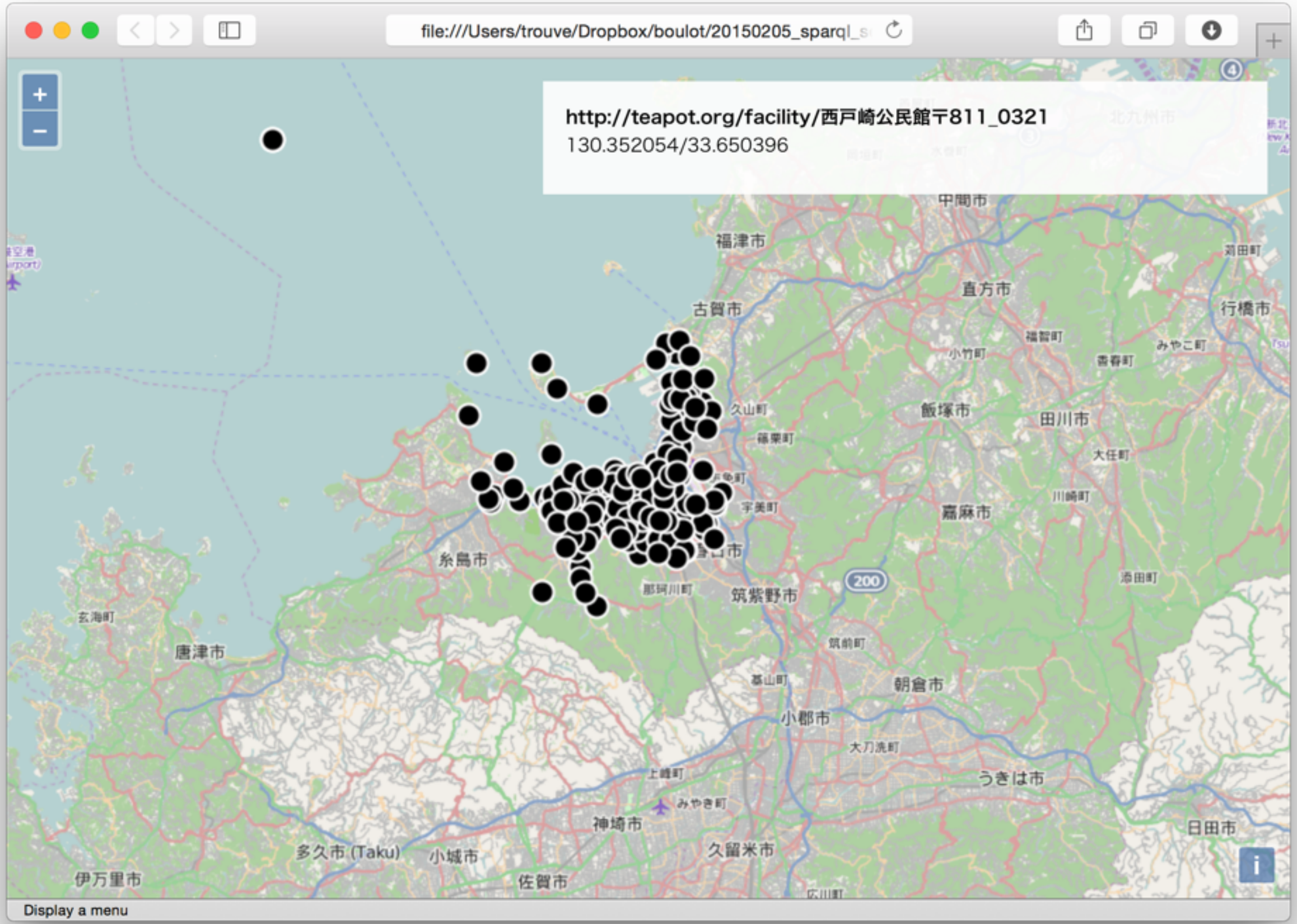
演習３ Step３

クリックしてピンの情報を表示しよう

教科書：http://www.bodic.org/?p=125

演習３Step２を書き加えて、公民館ピンをクリックすると識別子・緯度・経度を表示してください。

①Webページのどこかに表示するためなエレメント（div等）を追加する
②フィチャーオブジェクトに緯度・経度情報を格納する
③クリックイベント処理コードを追加する
④1で追加したエレメントの中に緯度・経度を表示する（Jquery等を使う）


file:///Users/trouve/Dropbox/boulot/20150205_sparql_s
http://teapot.org/facility/西戸崎公民館〒811_0321
130.352054/33.650396
中間市
福津市
古賀市
直方市
行橋市
飯塚市
田川市
嘉麻市
糸島市
筑紫野市
唐津市
朝倉市
小郡市
うきは市
神埼市
久留米市
日田市
多久市 (Taku)
小城市
佐賀市
伊万里市
200
Display a menu

演習３ Step４

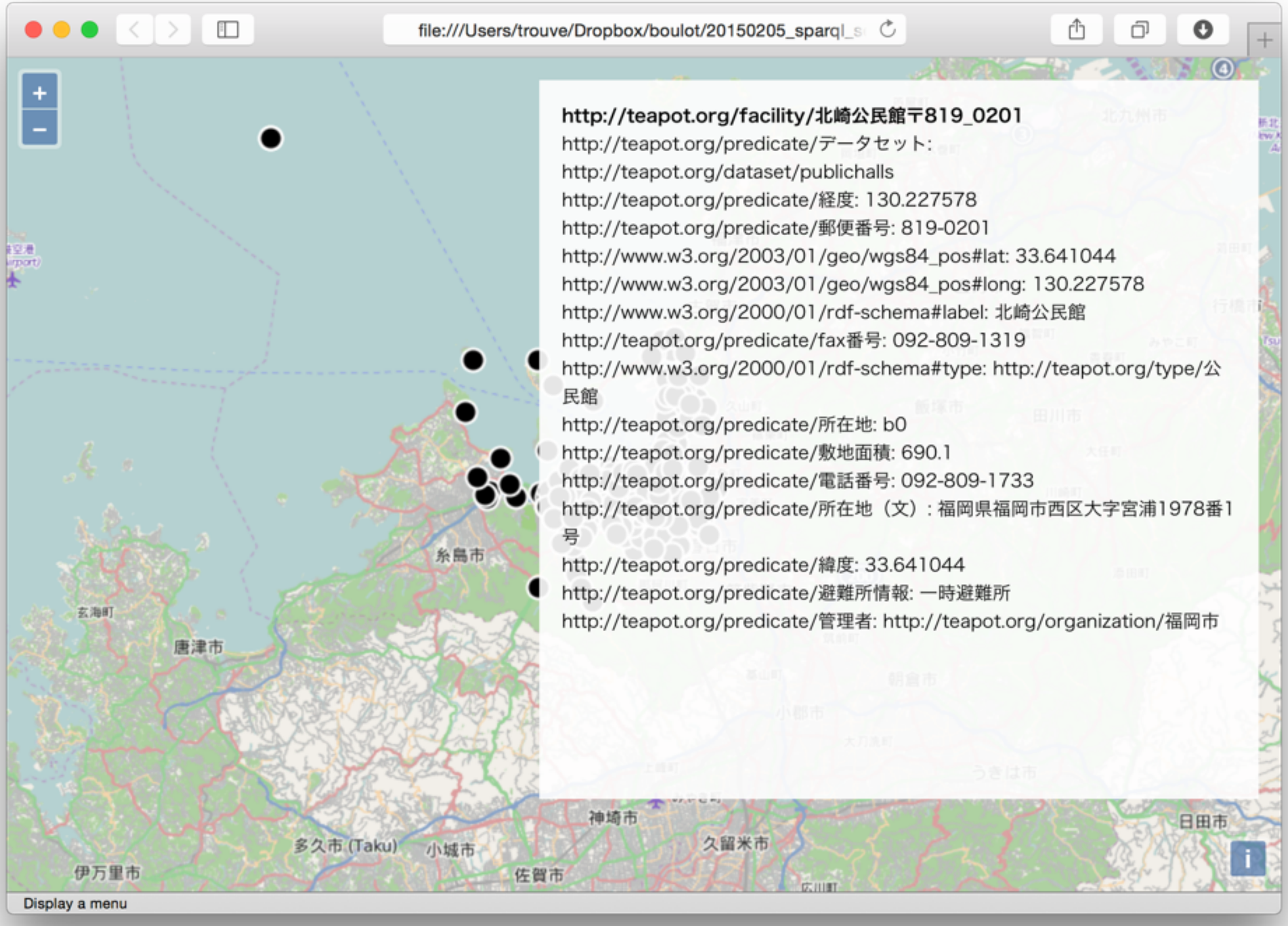
施設の詳細情報を表示しよう

教科書：http://www.bodic.org/?p=127

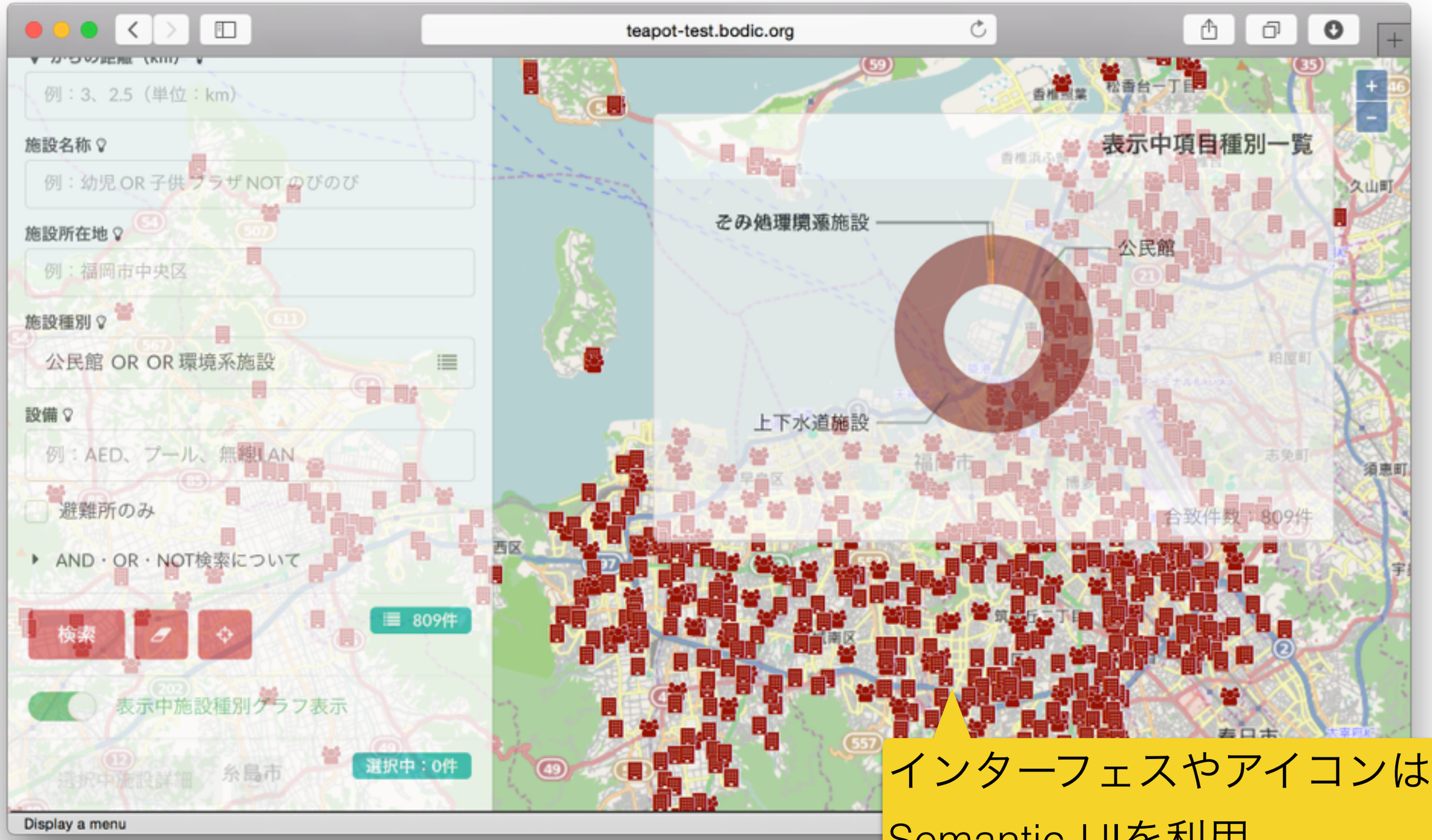
演習３Step３を書き加えて、公民館ピンをクリックするとTeapotにある全ての情報を表示してください。

・フィチャーをクリックするとTeapotにSPARQLクエリを投げる

・得られた情報は全て演習Step３で追加した情報表示エレメントに表示する。


file:///Users/trouve/Dropbox/boulot/20150205_sparql_s
http://teapot.org/facility/北崎公民館〒819_0201
http://teapot.org/predicate/データセット:
http://teapot.org/dataset/publichalls
http://teapot.org/predicate/経度: 130.227578
http://teapot.org/predicate/郵便番号: 819-0201
http://www.w3.org/2003/01/geo/wgs84_pos#lat: 33.641044
http://www.w3.org/2003/01/geo/wgs84_pos#long: 130.227578
http://www.w3.org/2000/01/rdf-schema#label: 北崎公民館
http://teapot.org/predicate/fax番号: 092-809-1319
http://www.w3.org/2000/01/rdf-schema#type: http://teapot.org/type/公民館
http://teapot.org/predicate/所在地: b0
http://teapot.org/predicate/敷地面積: 690.1
http://teapot.org/predicate/電話番号: 092-809-1733
http://teapot.org/predicate/所在地（文）: 福岡県福岡市西区大字宮浦1978番1号
http://teapot.org/predicate/緯度: 33.641044
http://teapot.org/predicate/避難所情報: 一時避難所
http://teapot.org/predicate/管理者: http://teapot.org/organization/福岡市
糸島市
玄海町
唐津市
伊万里市
多久市 (Taku)
小城市
佐賀市
神埼市
久留米市
日田市
Display a menu

今回の例を少してを加えれば



終わる前に・・・


bodik.jp
オープンデータで
みんなを
ハッピーに！
OPENDATA CONTEST
2014-2015 FUKUOKA
オープンデータコンテスト 2014-2015 福岡
応募受付開始日
2014年 12月22日(月)
応募締切日
2015年 2月23日(月)
表彰式
2015年 3月6日(金)
募集部門
アプリケーション部門
アイデア部門
応募フォームはこちら ＞
ホーム
開催概要
実証概要
応募方法
応募規約
審査・表彰
受賞作品
お問い合わせ
お知らせ
2014.12.22
「オープンデータ・コンテスト2014-2015 Fukuoka」サイトオープン
開催概要
オープンデータコンテスト2014-2015 Fukuokaの開催概要です。
実証概要
今年度の実証実験の概要をご案内いたします。



ありがとうございました