

初めてのSPARQL

Trouv  Antoine (トルヴェ Antoine)

2015/08/27

九州先端科学技術研究所・九州大学

- ・ **セミナーの目的**

- ・ SPARQLで基本的なクエリーを書けるようになること
- ・ SPARQLの背景について理解すること

- ・ **セミナーの概要**

- ・ セマンティックウェブについて (約15分)
- ・ RDFデータモデルについて (約30分)
- ・ SPARQL演習 (約2時間)

The RDF **Data** **Model**

RDFの背景

福岡市ホームページ



Gnavi(福岡)



福岡市27年度方針

施設名	住所	電話・FAX番号	休園日	アクセス	園舎	地図
東区東区子どもプラザ	福岡市東区東区2-1-1	092-980-3003	毎週月曜、毎月第3日曜	西鉄東区線「東区駅」、下車徒歩5分	あり	あり
東区三宮子どもプラザ	東区三宮5-40-1	092-980-4207	毎週水曜、毎月第2日曜	西鉄東区線「三宮駅」、下車徒歩5分	あり	あり
東区東区子どもプラザ	東区東区1-1-1	092-290-0300	毎週水曜、毎月第3日曜	西鉄バス「東区三丁目」、下車徒歩1分	あり	あり
東区山王子どもプラザ	東区山王5-1-1	092-672-6006	毎週日曜、毎月第1日曜、第3日曜	西鉄バス「山王駅」、下車徒歩5分	あり	あり
東区東区子どもプラザ	東区東区1-1-1	092-980-0711	毎週金曜、毎月第4土曜	西鉄バス「東区東区」、下車徒歩5分	あり	あり
中央区子どもプラザ	中央区東区2-2-4	092-741-9964	毎週月曜（祝日の場合は休園）、毎月第1日曜（日曜の場合は休園）、12月28日	地下鉄東区線「東区」、下車徒歩5分、西鉄バス「東区三丁目」、下車徒歩5分	あり	あり

福岡保育園一覧

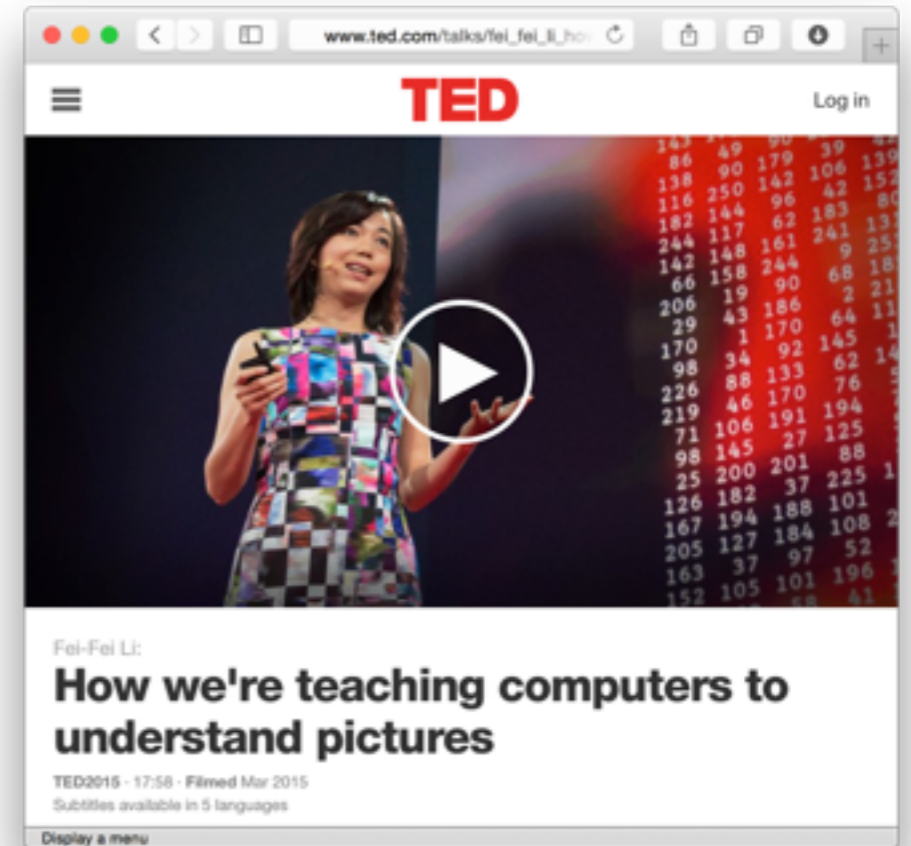
- WWWには膨大の情報がある
- しかしながら、ほとんどは**構造のない情報**
- 人間には問題なくその情報を解析できるが**コンピュータは違う**



コンピュータがWWW情報を理解するため、どうすればいいですか？

解決策 1

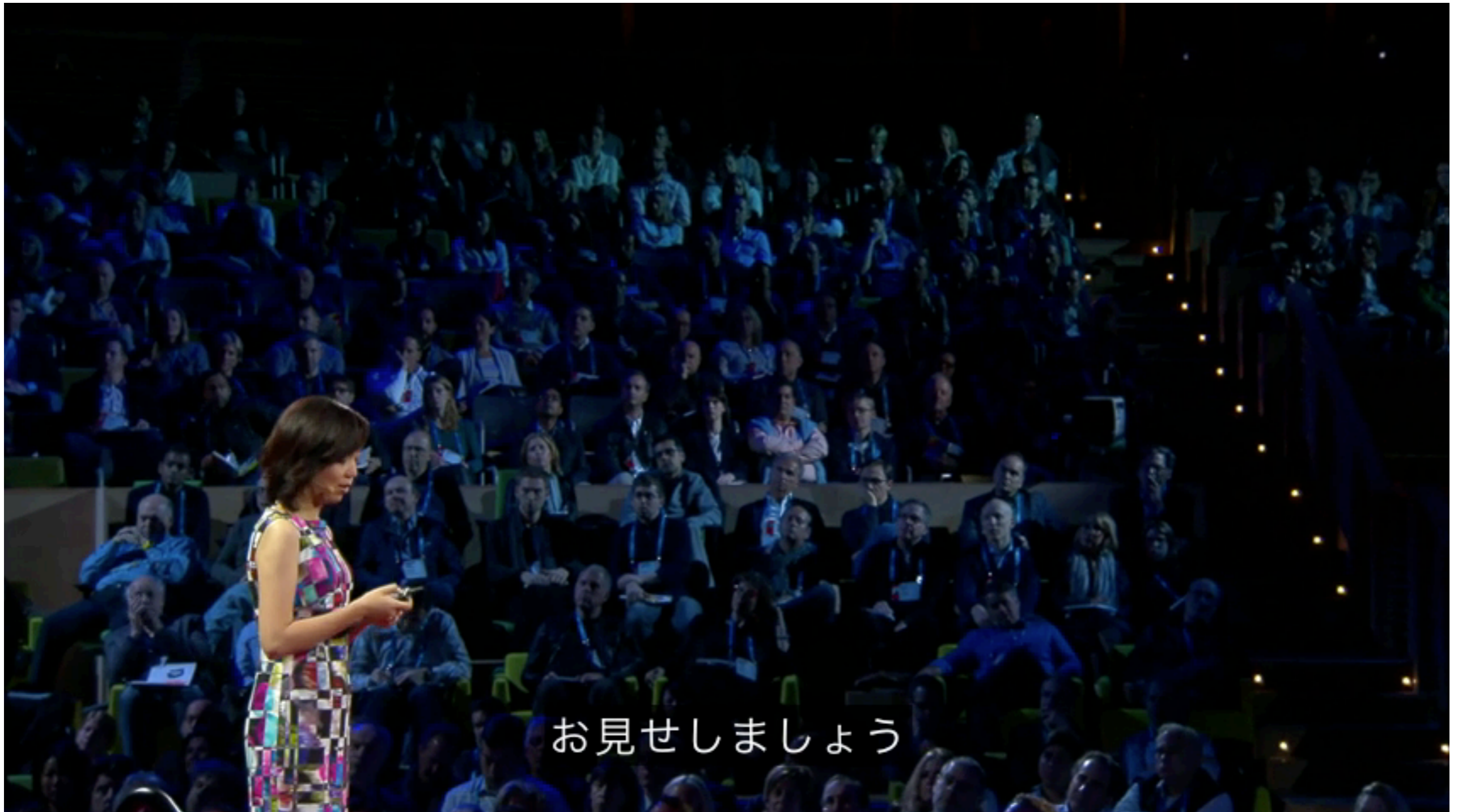
アルゴリズムなど（例：機械学習）を使って、コンピュータを賢くする



解決策 2

WWWにある情報を手動（半自動）で構造化する

解決策 1 について：先端研究の現状



お見せしましょう

From <http://www.ted.com/talks/fei-fei-li-how-we-re-teaching-computers-to-understand-pictures?language=en>

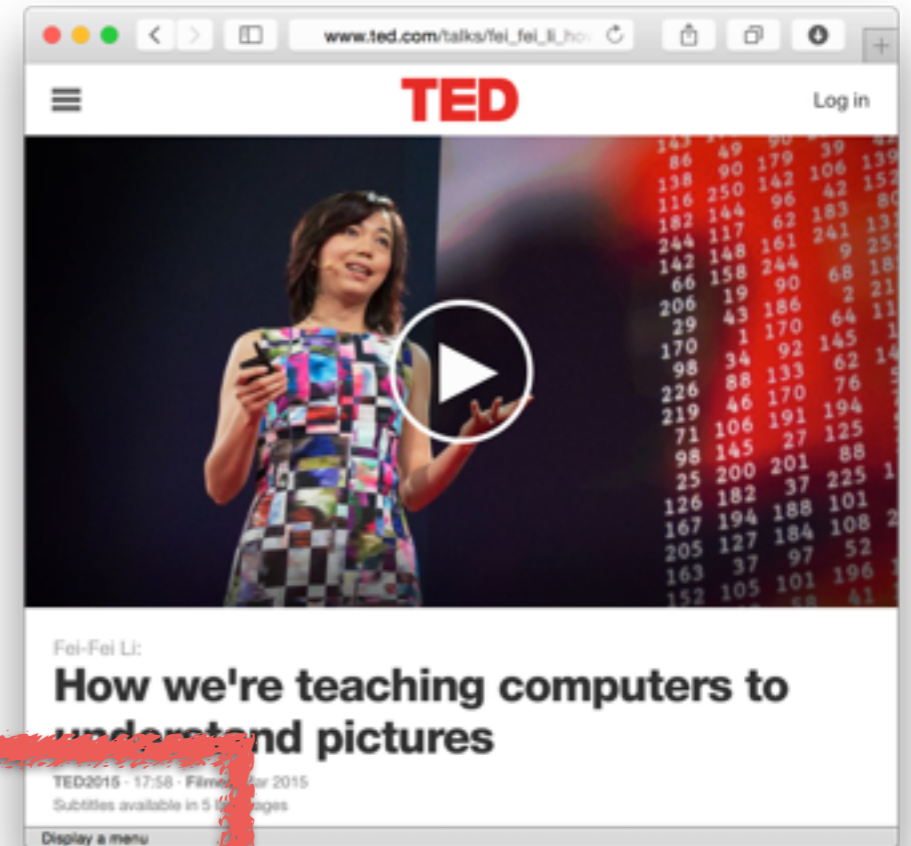
Fei-Fei Li

Stanford's Artificial Intelligence Lab and Vision Lab

コンピュータがWWW情報を理解するため、どうすればいいですか？

解決策 1

アルゴリズムなど（例：機械学習）を使って、コンピュータを賢くする



解決策 2

WWWにある情報を手動（半自動）で構造化する

WWWにある情報をKey Valueで保存

リソースID

http://city.fukuoka.lg.jp

is about Fukuoka city

is a Web page

last seen 2015-2-1

キー

値

福岡市ホームページ



R Resource D Description F Framework

リソース (=物)

記述

枠組み

英語について

1: singleton / 2: couple / 3: triple

Data is expressed as triples

主語

リソースのID

述語

プロパティ

目的語

プロパティ値

例: 前のスライドのサンプル、 トルプルで表現した場合

http://city.fukuoka.lg.jp **is about** Fukuoka city

http://city.fukuoka.lg.jp **is a** Web page

http://city.fukuoka.lg.jp **last seen** 2015-2-1

W3Cについて

- ・ 1994年に設立
- ・ WWWで使われている技術の規格を管理する
 - ・ HTML, XML, Javascript, CSS, **RDF**
- ・ RDFはW3C規格である
 - ・ 最新版はRDF 1.1 (2014/2/25に発表)
 - ・ RDF規格の中に更に諸々な規格が定義されている



W3C®

- ・ Tim Berners-Lee, head of the W3C.
- ・ He developed the early version of the www in 1989 (while working at CERN, France)

RDFの実例

主語

述語

目的語

言語・タイプ

http://
city.fukuoka.lg.jp

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

http://schema.org/
WebSite

http://
city.fukuoka.lg.jp

http://schema.org/
about

http://dbpedia.org/
resource/Fukuoka

http://
city.fukuoka.lg.jp

http://schema.org/
lastReviewed

“2015-2-1”

http://www.w3.org/
2001/
XMLSchema#date

http://
city.fukuoka.lg.jp

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

“Fukuoka city official
homepage”

en

http://
city.fukuoka.lg.jp

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

“福岡市公式ホームペー
ジ”

ja

このアドレスIRI:
Internationalized
Resource Identifier
国際リソースID

RDFの実例

Subject

Predicate

Object

Language /
Type

http://
city.fukuoka.lg.jp

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

http://schema.org/
WebSite

These addresses are
**IRI: Internationalized
Resource Identifier**
(superset of URI)

http://
city.fukuoka.lg.jp

http://schema.org/
about

http://dbpedia.org/
resource/Fukuoka

IRIは読みにくい!

http://
city.fukuoka.lg.jp

http://www.w3.org/
2001/XMLSchema#date

“2015-2-1”

http://www.w3.org/
2001/
XMLSchema#date

http://
city.fukuoka.lg.jp

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

“Fukuoka city official
homepage”

en

http://
city.fukuoka.lg.jp

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

“福岡市公式ホームペー
ジ”

ja



QnameとCURIE: IRIが読みやすくなるように

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

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

共通プレフィックス



プレフィックス

rdfs:type
rdfs:label

ローカル部分

- ・ CURIE: スラッシュ「/」を使える
- ・ Qname: スラッシュ「/」を使えない

CURIEを使ったRDF事例

主語

述語

目的語

言語・タイプ

http://city.fukuoka.lg.jp

rdfs:type

schema:WebSite

http://city.fukuoka.lg.jp

schema:about

db:Fukuoka

http://city.fukuoka.lg.jp

schema:lastReviewed

“2015-2-1”

xsd:date

http://city.fukuoka.lg.jp

rdfs:label

“Fukuoka city official
homepage”

en

http://city.fukuoka.lg.jp

rdfs:label

“福岡市公式ホームペー
ジ”

ja

I use well-used prefix here. In the real world one should define them before use.

More on that later with
turtle and SPARQL

A Real RDF Example with CURIE

Subject	Predicate	Object	Language / Type
<code>http://city.fukuoka.lg.jp</code>	<code>rdfs:type</code>	<code>schema:WebSite</code>	
<code>http://city.fukuoka.lg.jp</code>	<code>schema:about</code>	<code>db:Fukuoka</code>	
<code>http://city.fukuoka.lg.jp</code>	<code>schema:lastReviewed</code>	<code>"2015-2-1"</code>	<code>xsd:date</code>
<code>http://city.fukuoka.lg.jp</code>	<code>rdfs:label</code>	<code>"Fukuoka city official homepage"</code>	<code>en</code>
<code>http://city.fukuoka.lg.jp</code>	<code>rdfs:label</code>	<code>"福岡市公式ホームページ"</code>	<code>ja</code>



大分良くなりました！

More on that later with
turtle and SPARQL

I use well-used prefix here. In the real world one should define them before use.

リソースのIRI

主語

述語

目的語

言語・タイプ

http://city.fukuoka.lg.jp

rdfs:type

schema:WebSite

http://city.fukuoka.lg.jp

schema:about

db:Fukuoka

http://city.fukuoka.lg.jp

schema:lastReviewed

"2015-02-11"

xsd:date

http://city.fukuoka.lg.jp

en

http://city.fukuoka.lg.jp

ja

- ・このIRIはサイトのURLではなく実世界に「福岡市」というリソースを示す
- ・誰でもIRIを作っても構いませんが、できるだけ既存のIRIを使った方がデータの利用者にとって使いやすい
- ・IRIとしてURLを使うことが多い（アクセスするとリソースについての情報が表示）

リソースはウェブサイトですのでIRIとしてサイトURLを使うのは無難

語彙におけるIRI

主語

述語

目的語

言語・タイプ

http://city.fukuoka.lg.jp

述語はIRIである

schema:WebSite

これはschema.orgという語彙の言葉を示すIRI

http://city.fukuoka.lg.jp

db:Fukuoka

http://city.fukuoka.lg.jp

schema:lastReviewed

“2015-2-1”

xsd:date

http://city.fukuoka.lg.jp

rdfs:label

“Fukuoka city official
home”

これは日付というタイプ（型）を示すIRI

http://city.fukuoka.lg.jp

rdfs:label

“福岡市公式
ページ”

ja

- ・ IRIは人物に加えて、語彙を示すこともある
- ・ 語彙は意味がちゃんと定義されている言葉・概念のこと（人間言語に依存せずに定義する）
- ・ その言葉は主に述語とタイプとして使う
- ・ 自分の語彙を定義しても構いませんが、既存の語彙を使った方がデータ利用者に優しい

RDFの細かい機能：述語の重複

主語	述語	目的語	言語・タイプ
http://city.fukuoka.lg.jp	rdfs:type	schema:WebSite	
http://city.fukuoka.lg.jp	schema:about	db:Fukuoka	
http://city.fukuoka.lg.jp	schema:lastReviewed	“2015-2-1”	xsd:date
http://city.fukuoka.lg.jp	rdfs:label	“Fukuoka city official homepage”	en
http://city.fukuoka.lg.jp	rdfs:label	“福岡市公式ホームページ”	ja

- ・ RDFデータは何度も同じ述語を指定しても大丈夫です
 - ・ よくあるユースケース：名前を複数言語で入れたいときに

リテラルの言語・タイプについて

主語

述語

目的語

言語・タイプ

http://city.fukuoka.lg.jp

rdfs:type

schema:WebSite

タイプの例

http://city.fukuoka.lg.jp

schema:about

db:Fukuoka

- ・ リテラルはIRIではない者
- ・ 引用符に囲むが、文字列ではないものがある
 - ・ 言語を指定すると「言語付き文字列」として特別に扱う（言語はISO 639で指定）
 - ・ 更にタイプも指定できる

"2015-2-1"

xsd:date

"Fukuoka city official homepage"

en

"福岡市公式ホームページ"

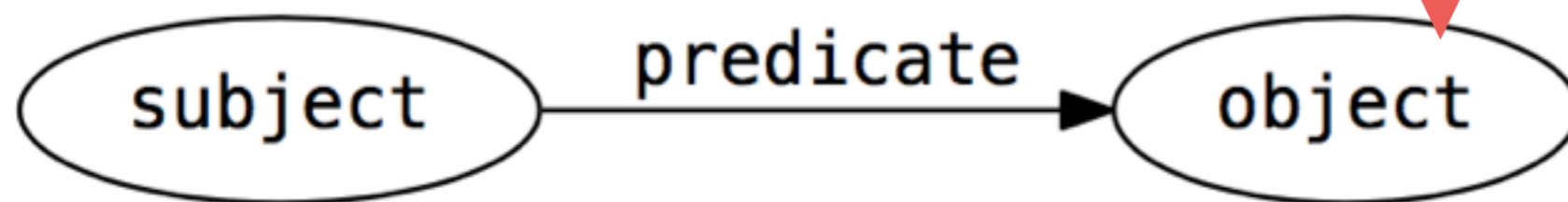
ja

言語の例

- ・ RDF規格はXML標準タイプを含む:
 - ・ xsd:integer, xsd:decimal, xsd:float, xsd:double, xsd:string, xsd:boolean, xsd:dateTime, xsd:date, xsd:time

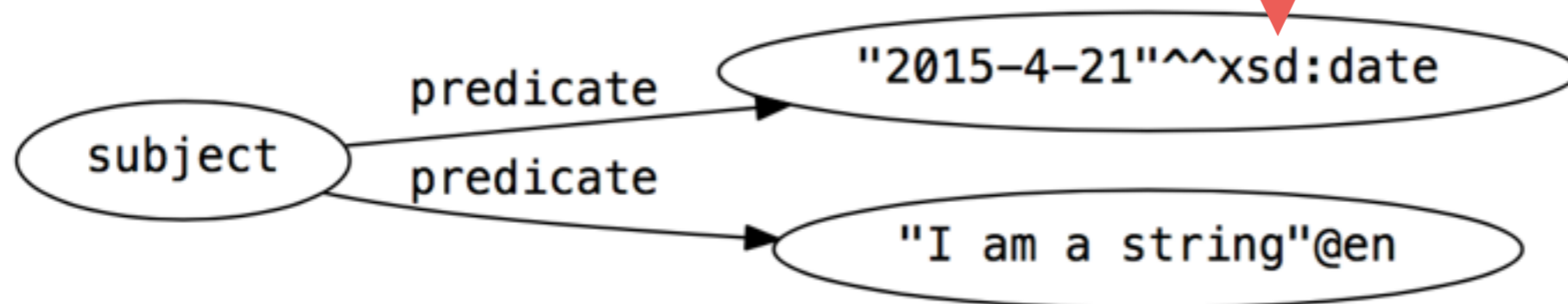
RDFグラフ

It is possible (and common usage) to **represent RDF data graphically**, as below:



I generate the graphs with Graphviz

Literals' tags are represented with “^^” for datatypes, and @ for language, as below:



This is the same syntax as in SPARQL and Turtle, as we will see soon

An Example of Graph

主語

述語

目的語

言語・タイプ

http://city.fukuoka.lg.jp

rdfs:type

schema:WebSite

http://city.fukuoka.lg.jp

schema:about

db:Fukuoka

http://city.fukuoka.lg.jp

schema:lastReviewed

“2015-2-1”

xsd:date

http://city.fukuoka.lg.jp

rdfs:label

“Fukuoka city official homepage”

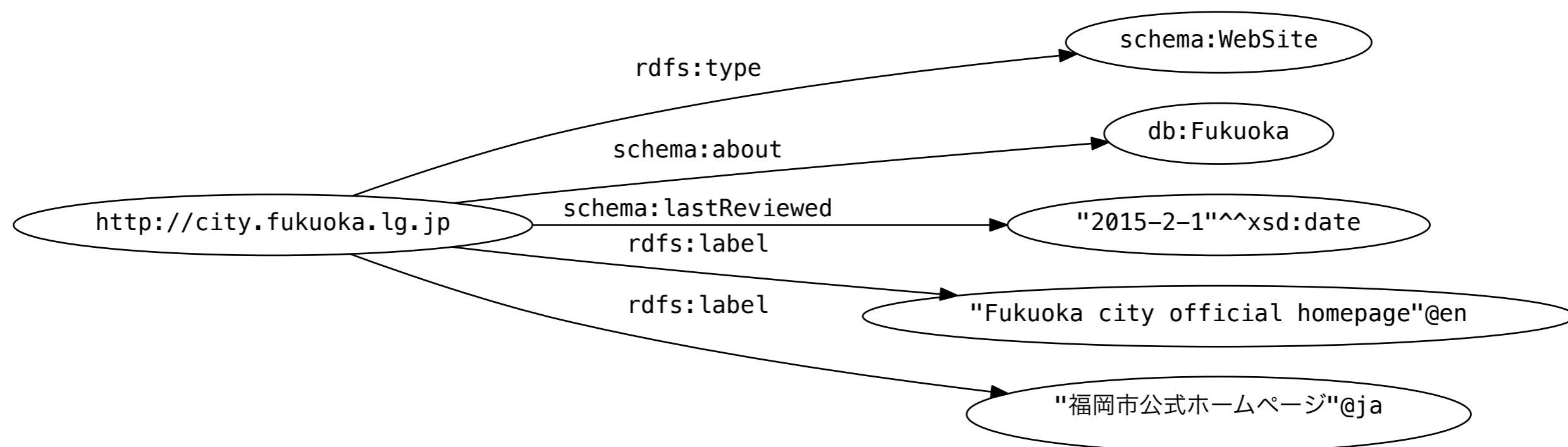
en

http://city.fukuoka.lg.jp

rdfs:label

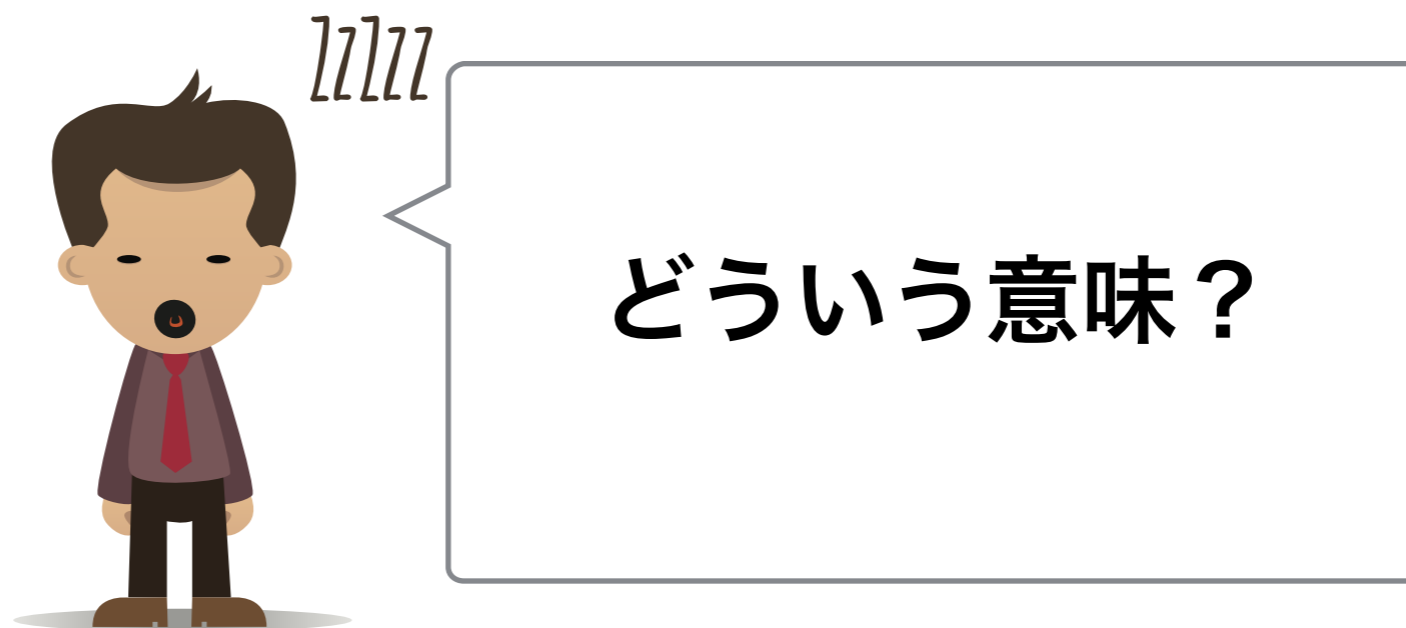
“福岡市公式ホームページ”

ja



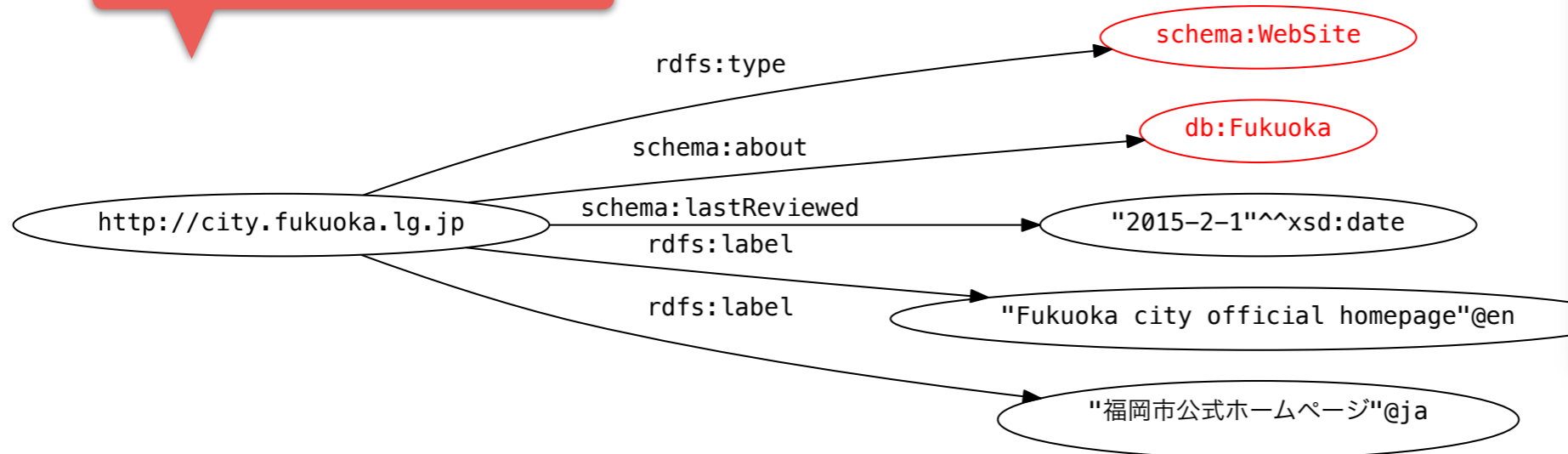
LODについて

- ・ 同じIRIを再利用すると、リソース間にリンクを貼ることができる
- ・ 特に同じIRIが目的語と主語として使われている時
- ・ **LOD**は**L**inked **O**pen **D**ataの略語です

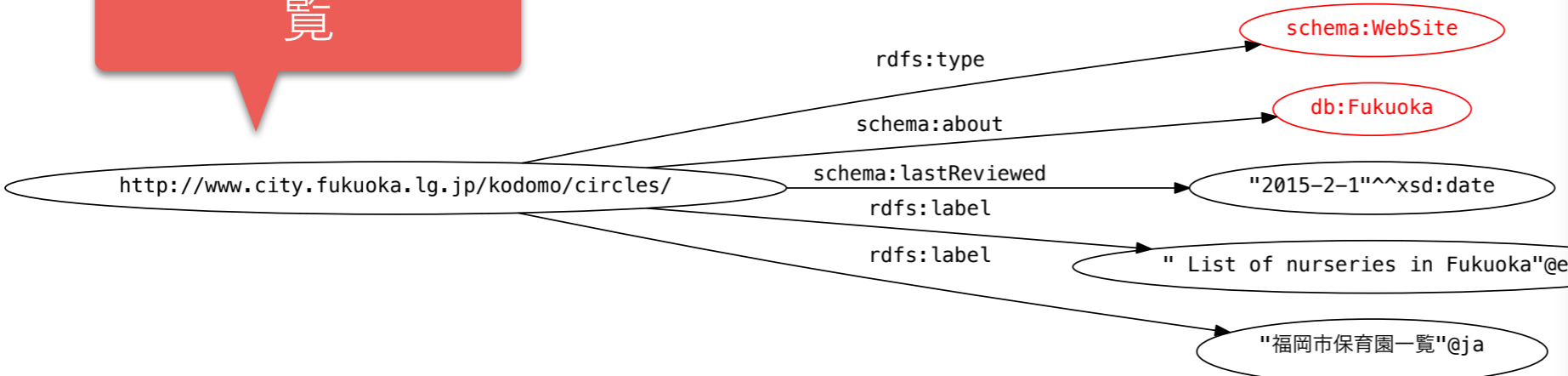


LODグラフの例 (1)

福岡市ホームページ

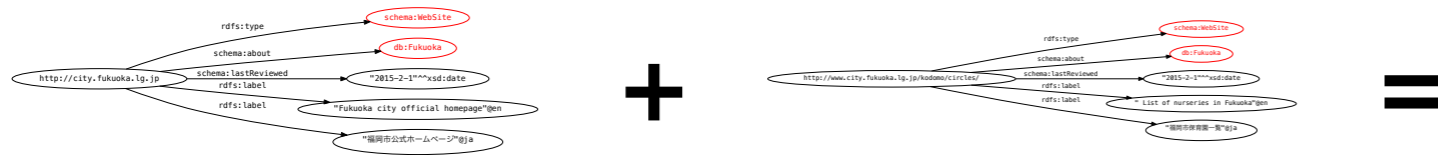


福岡市保育園一
覧

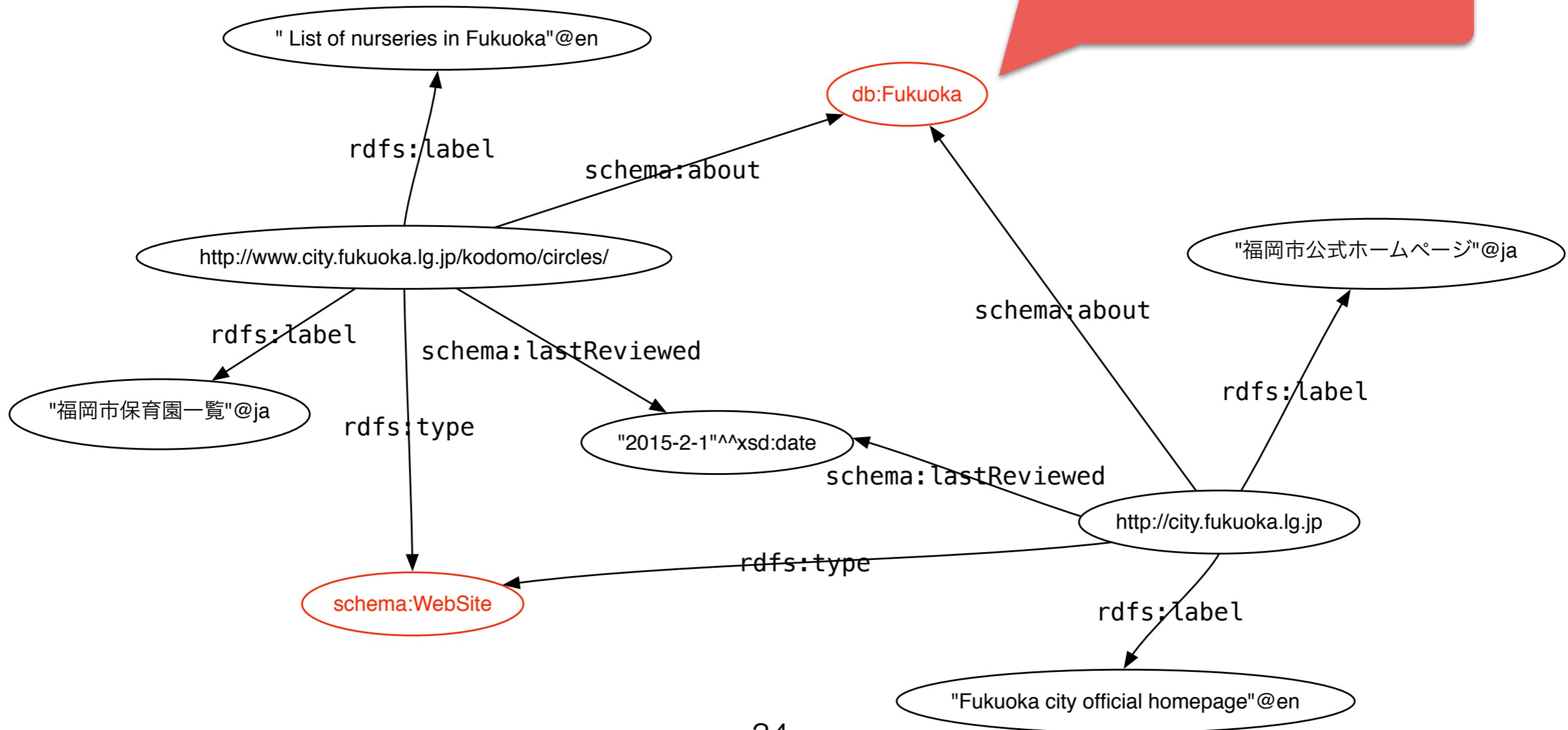


施設名	住所	電話・FAX 番号	休園日	アクセス	画像	地図
東区東区子どもプラザ	福岡市東区 東区東区2 -5-2-1 セゾプラザ 西鉄東区 2階	090-660-3060	毎週月曜、毎月 第2日曜	西鉄東区線「西鉄東区」 下車徒歩5分。西鉄バス 「西鉄東区」下車徒歩5分。 西鉄バス「東区」下車徒歩 5分。西鉄バス「東区」下車 徒歩5分。		
東区三宮子どもプラザ	東区三宮 1-40-1階	090-660-4007	毎週水曜、毎月 第1日曜	西鉄東区線「三宮駅」下 車徒歩5分		
東区東区子どもプラザ	東区東区1- 1-1 4F セゾプラザ 西鉄東区 2階	090-260-5320	毎週水曜、毎月 第3日曜	西鉄バス「東区三丁」下 車徒歩1分		
博多区山王子どもプラザ	博多区山王 1-13-10 博多区山王 センター併設	090-472-6006	毎週月曜、毎月 末日（土・日曜 、祝日の場合は 、西鉄の土・日 曜、祝日の場合は センター併設）	西鉄バス「山王公園」下 車徒歩5分		
博多区博多区子どもプラザ	博多区博多 1-4-11 1 階	090-660-6715	毎週水曜、毎月 第4土曜	西鉄バス「博多駅前」下 車徒歩4分。西鉄バス「博 多駅前」下車徒歩5分		
中央区中央子どもプラザ	中央区中央 2-2-4	090-741-3664	毎週月曜（毎月 の第1日曜） 、毎月末日（日 曜の場合は休園）	地下鉄有明線「中央」下 車徒歩4分。西鉄バス「 中央二丁目」下車徒歩5分		

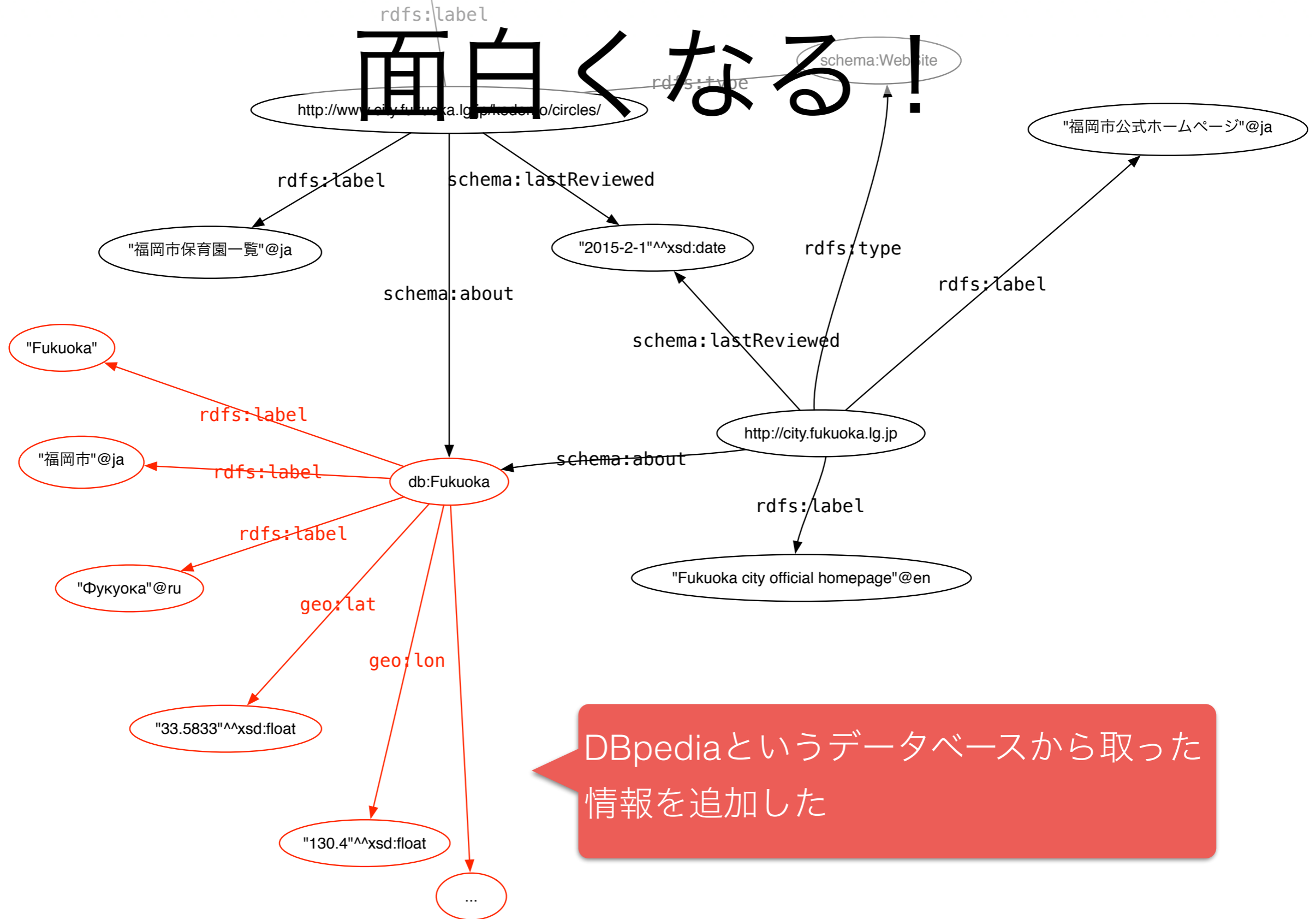
LODグラフの例 (1)



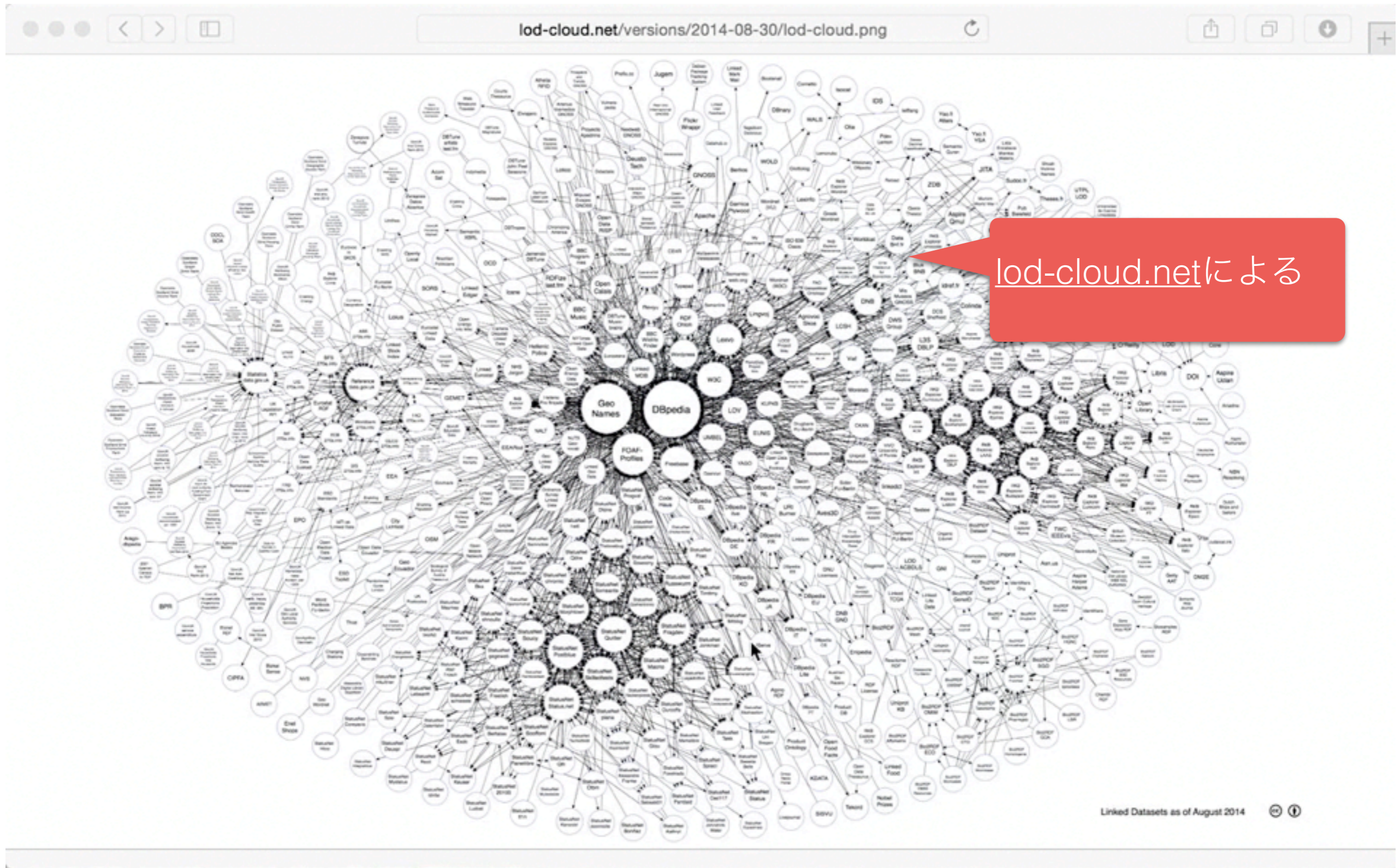
両方福岡市について



更に情報源を増やすとより



LODグラフの実例

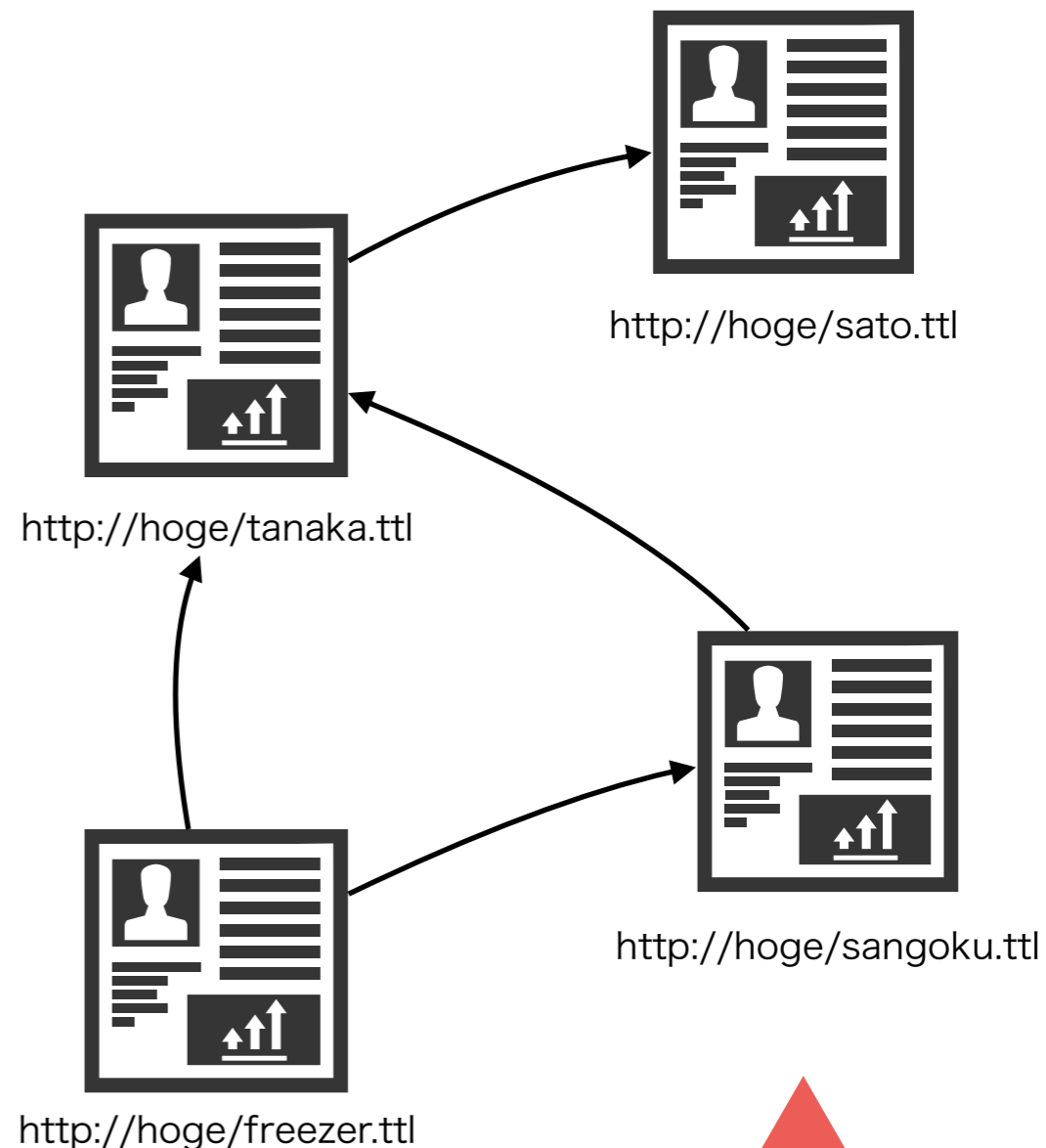


RDFデータ

保管について

どこでRDFを保管すればいい？

- ・ データが小さい場合：**テキストファイル**
 - ・ テキスト形式でRDFファイルをオンラインで置くだけでセマンティックウェブに参加できる！
 - ・ そのようなファイルを検索できるツールもある！
- ・ データが多い場合：**データベース**
 - ・ RDFデータベース：**グラフストア**又は**トリプルストア**という
 - ・ NoSQLデータベースの1種類



Friend of Friend (FOAF)プロジェクトについて

- ・ FOAF語彙は人間関係をできるような語彙
- ・ FOAFプロジェクトは分散SNSを構築しようとしている

RDFのテキスト形式

- **N-Triple**

- トリプルを並べるだけ

- **Turtle / N3**

- N-Tripleを読みやすくしたもの
- SPARQLはTurtleに近いシンタックスを使う

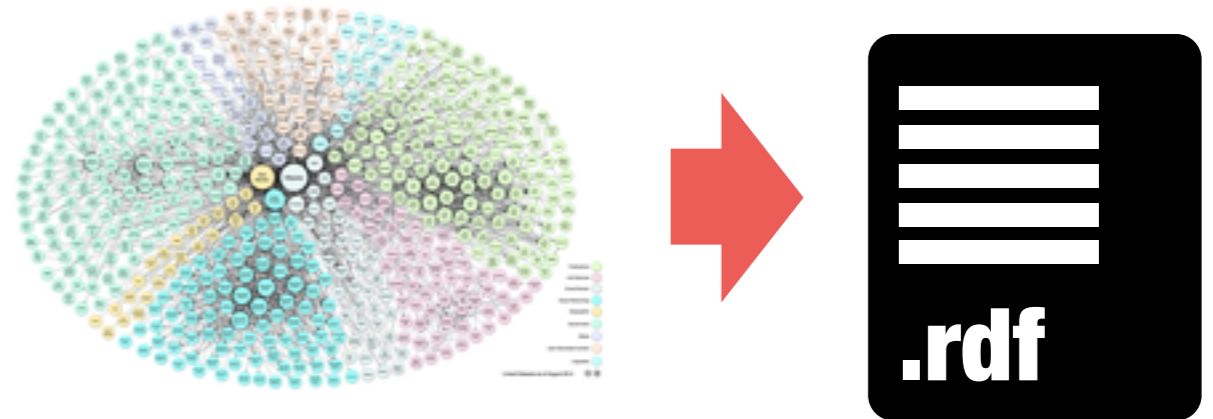
- **RDF/XML**

- Turtleが出る前に一番一般的なテキスト形式でした
- Turtleと比べると文字数が多い

- **MicroData / RDFa**: HTMLページにRDFデータを組み込めるため

- SEOに効果的！

グーグルはRDFaとMicroDataを解析しています！



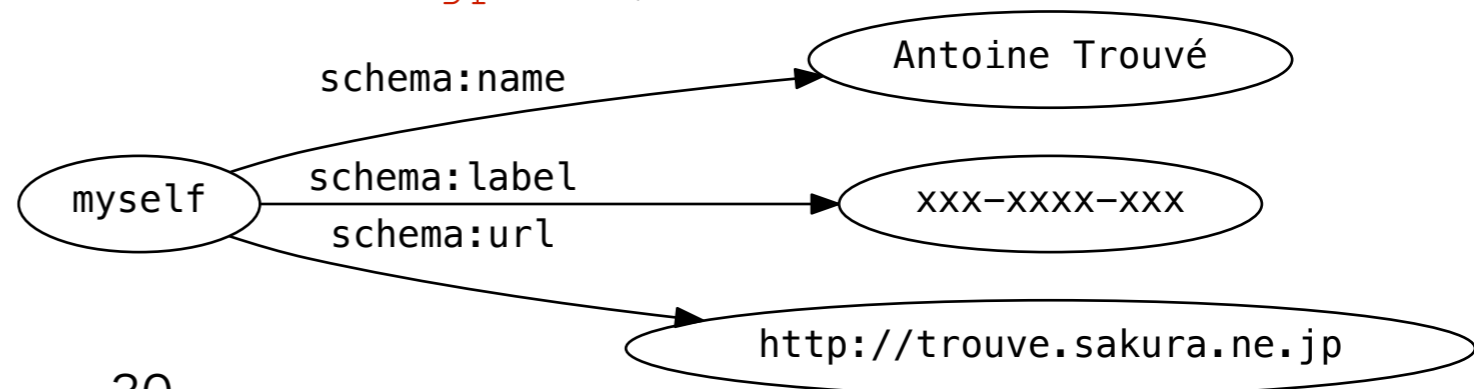
Microdata と RDFa

- HTMLページにRDFトリプルを組み込むためのフォーマット
- Microdata
 - CSSクラス名を使う
 - シンプルだが、表現できないRDF情報がある
 - schema.org consortium が提案(メンバー: Google, Microsoft, Yahoo等)
- RDFa (especially RDFa Lite)
 - W3C規格
 - 例:

```
<p about="myself" vocab="http://schema.org/" typeof="Person">  
My name is  
<span property="name">Antoine Trouvé</span>,  
my phone number is  
<span property="telephone">xxx-xxxx-xxx</span>  
and my homepage is  
<a property="url" href="http://trouve.sakura.ne.jp/"></a>  
</p>
```

schema.org語彙を利用する場合

同等グラフ



グーグルにおけるRDFa/ Microdata情報の扱い

The screenshot shows a Google search interface. The search bar contains the text 'arstechnica'. Below the search bar, there are navigation links for 'Web', 'News', 'Images', 'Videos', 'Maps', 'More', and 'Search tools'. The search results indicate 'About 1,760,000 results (0.72 seconds)'. The top result is for 'Ars Technica' with the URL 'arstechnica.com/'. Below this, there is a search bar for 'Results from arstechnica.com'. The search results are organized into a grid of six categories:

- OpenForum**: The Server Room - Agora, Classifieds - The Observatory
- Newer Stories**: Older Stories - Newer Stories →, ArsTechnica ... Recent ...
- Infinite Loop**: Infinite Loop / The Apple Ecosystem, Free Apple Logic ...
- Risk Assessment**: Risk Assessment / Security & Hactivism, Feature Story ...
- Technology Lab**: Information Technology Technology Lab ... Technology ...
- Opposable Thumbs**: Opposable Thumbs / Gaming & Entertainment, One year later ...

グーグルのユーザーが見る
検索結果をコントロールで
きる

RDFa / Microdata活用事例

・ 検索エンジン

- ・ SindiceはRDFa/Microdataデータを検索エンジン
- ・ Google, Yahoo, Bingは解析し、検索結果に反映している

・ **Facebook** はRDFaを利用している (Open Graph API)

- ・ ウェブサイトからの情報抽出
- ・ Likeボタンを実装するため

SEOに効きます！

・ **Browser support**

- ・ FirefoxなどはRDFa/Microdataを検索できるようにプラグインが存在している

・ RDFaとMicrodataの間の**変換**

- ・ <http://rdf-translator.appspot.com>

・ **RDFa ? Microdata ?**どっちを使えばいい？

- ・ RDFaの方が複雑だが、より複雑な情報を表現できる
- ・ 現在はすべてのツールが両方サポートしているので、どちらでもいい！

ハンズオン：SPARQL
クエリー書きましょう！

SPARQLにおける背景

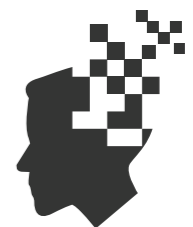
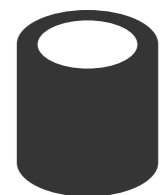
SPARQL Protocol And RDF Query Language

- ・ SPARQLはW3C規格である
 - ・ SPARQL 1.0 (15/1/2008)
 - ・ SPARQL 1.1 (21/3/2013)
- ・ 現在のRDFデータベースは大体対応している

RDFは2000年3月に
発表されたので、
SPARQLの方が大分若い

SPARQLはRDFのSQLである

	関係データベース	RDF
クエリー言語	SQL	SPARQL
データベース種	関係データベース	RDFデータベース
データモデル	関係モデル	RDF 1.1




DBpediaについて

- ・ DBpediaプロジェクトの目的
 - ・ Wikipediaにある情報を構造化して、公開しましょう。
- ・ その成果は
 - ・ dbpedia.orgというサイト
- ・ そのデータはRDFで公開し、SPARQL検索が可能



DBpedia SPARQL インタフェース

<http://dbpedia.org/sparql>



The screenshot shows the Virtuoso SPARQL Query Editor interface in a web browser. The browser's address bar displays `dbpedia.org/sparql`. The page title is "Virtuoso SPARQL Query Editor". In the top right corner, there are links for "About", "Namespace Prefixes", "Inference rules", and "ISPARQL".

The "Default Data Set Name (Graph IRI)" field contains the text `http://dbpedia.org`.

The "Query Text" area contains the SPARQL query: `select * { ?s ?p ?o }`.

At the bottom of the interface, there are several configuration options:

- A note: "(Security restrictions of this server do not allow you to retrieve remote RDF data, see [details](#).)"
- Results Format:** A dropdown menu is set to "HTML". A note next to it says "(The CXML output is disabled, see [details](#))".
- Execution timeout:** A text input field contains "30000" milliseconds. A note next to it says "(values less than 1000 are ignored)".
- Options:** A checkbox labeled "Strict checking of void variables" is checked.

At the very bottom, there is another note: "(The result can only be sent back to browser, not saved on the server, see [details](#))" and a button labeled "メニューを表示".

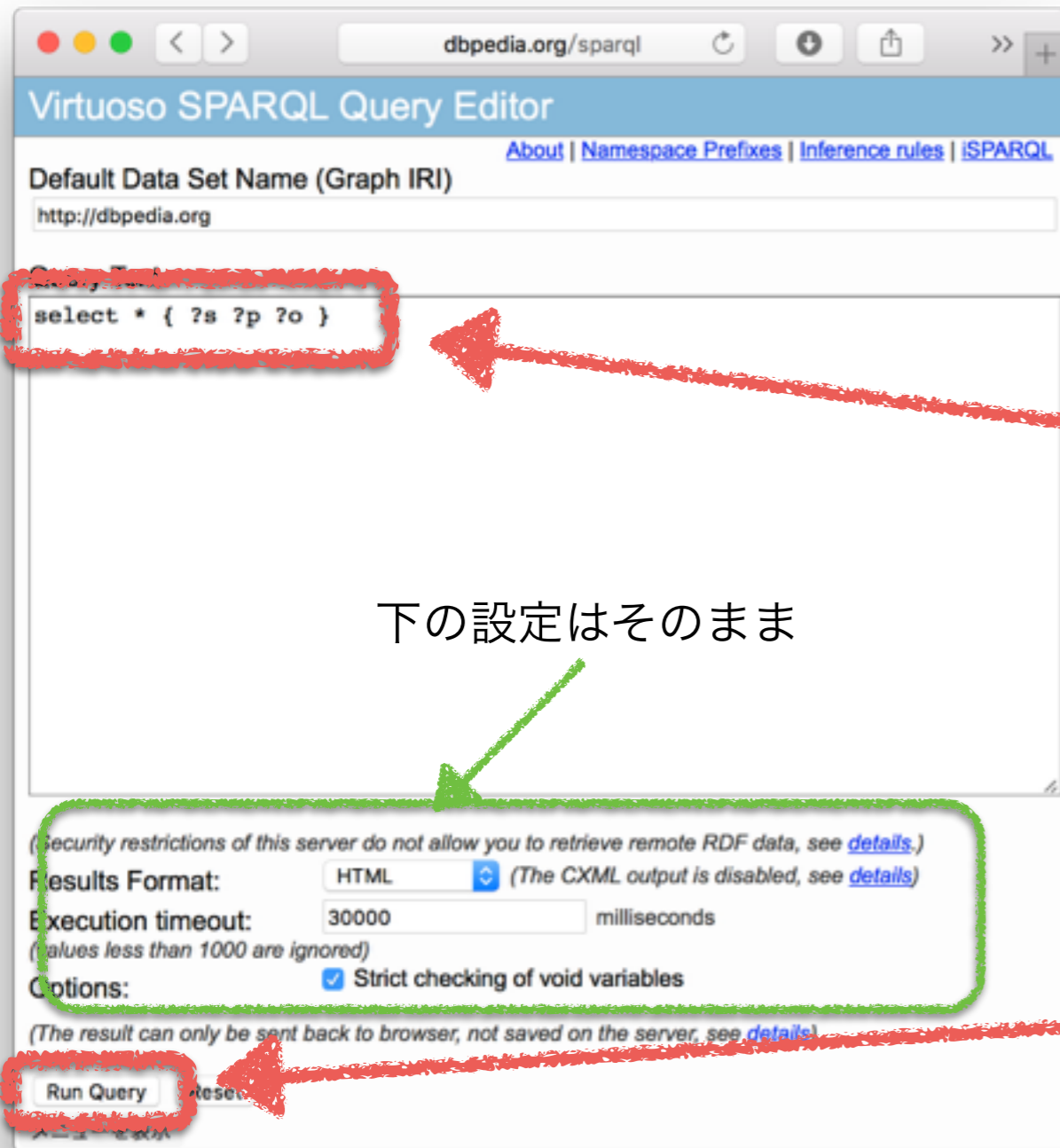
演習 1

下記のクエリーを書いてください：

```
select * { ?s ?p ?o }
```

下の設定はそのまま

クエリーを記入したら「Run Query」を押す。



演習 1

変数s,p,oはそれぞれ全ての主語,述語,目的語とする

```
select * { ?s ?p ?o }
```

Virtuoso SPARQL Query Editor

Default Data Set Name (Graph IRI)
http://dbpedia.org

Query Text
select * { ?s ?p ?o }

Results Format: HTML

Execution timeout: 30000

Options: Strict checking of vo

Run Query Reset

s	p	o
http://dbpedia.org/ontology/deathDate	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/birthDate	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/acceleration	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/averageAnnualGeneration	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/birthYear	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/co2Emission	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/deathYear	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/diameter	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/displacement	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/foalDate	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/fuelCapacity	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/fuelConsumption	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty
http://dbpedia.org/ontology/height	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#FunctionalProperty

検索結果

LIMIT, OFFSETについて

- ・ 先ほどのクエリは合致した結果が多い
- ・ 表示側もサーバー側も大量のデータを扱うのは負担がかかる
- ・ その負担を減らすために、「LIMIT」や「OFFSET」 SPARQLキーワードを使う
 - ・ LIMIT：検索結果の件数の上限
 - ・ OFFSET：全ての検索結果の中から何件目から欲しいのかを指定
- ・ 例：
 - ・ SELECT * { ?s ?p ?o } LIMIT 10 → 1件目～10件目
 - ・ SELECT * { ?s ?p ?o } LIMIT 10 OFFSET 5 → 5件目～15件目
 - ・ SELECT * { ?s ?p ?o } OFFSET 20 → 20件目～最後まで

演習 2 : LIMIT/OFFSETを使おう！

下記のクエリーを

```
select * { ?s ?p ?o }
```

「LIMIT」と「OFFSET」キーワードを使って、0から100件目までのみ戻すように変更してください。

基本的なSPARQL

シンタックスについて

クエリーの種類を指定する動詞

- ・ SELECTはクエリー
- ・他にもあります：ASK、DESCRIBE、CONSTRUCT

SELECT * { <グラフパターン> }

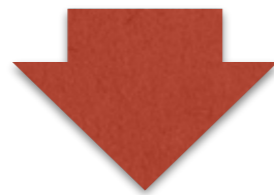
変数リスト。「*」は「全ての変数」と意味する

グラフパターン

- ・ Turtleに近いシンタックスを使ったトリプル。
- ・ クエリーの条件や合致したい情報を表現するため。

Turtleシンタックスの例

Subject	Predicate	Object	Language / Type
http://city.fukuoka.lg.jp	http://www.w3.org/2000/01/rdf-schema#type	http://schema.org/Website	
http://city.fukuoka.lg.jp	http://schema.org/about	http://dbpedia.org/resource/Fukuoka	
http://city.fukuoka.lg.jp	http://schema.org/lastReviewed	"2015-2-1"	http://www.w3.org/2001/XMLSchema#date
http://city.fukuoka.lg.jp	http://www.w3.org/2000/01/rdf-schema#label	"Fukuoka city official homepage"	en
http://city.fukuoka.lg.jp	http://www.w3.org/2000/01/rdf-schema#label	"福岡市公式ホームページ"	ja



- ・ トリプルはドット「.」で区切る
- ・ IRIはアングルブラケット「<...>」の中に入れる
- ・ リテラル（文字列など）は引用符の中に入れる
- ・ 言語とタイプはそれぞれ「^^」と「@」で指定する
 - ・ タイプ："2"^^integer
 - ・ 言語："Fukuoka"@en

```
<http://city.fukuoka.lg.jp>
  <http://www.w3.org/2000/01/rdf-schema#type> <http://schema.org/Website> .
<http://city.fukuoka.lg.jp>
  <http://schema.org/about> <http://dbpedia.org/resource/Fukuoka> .
<http://city.fukuoka.lg.jp>
  <http://schema.org/lastReviewed> "2015-2-1"^^<http://www.w3.org/2001/XMLSchema#date> .
<http://city.fukuoka.lg.jp>
  <http://www.w3.org/2000/01/rdf-schema#label> "Fukuoka city official homepage"@en .
<http://city.fukuoka.lg.jp>
  <http://www.w3.org/2000/01/rdf-schema#label> "福岡市公式ホームページ"@ja .
```

条件のつけ方について

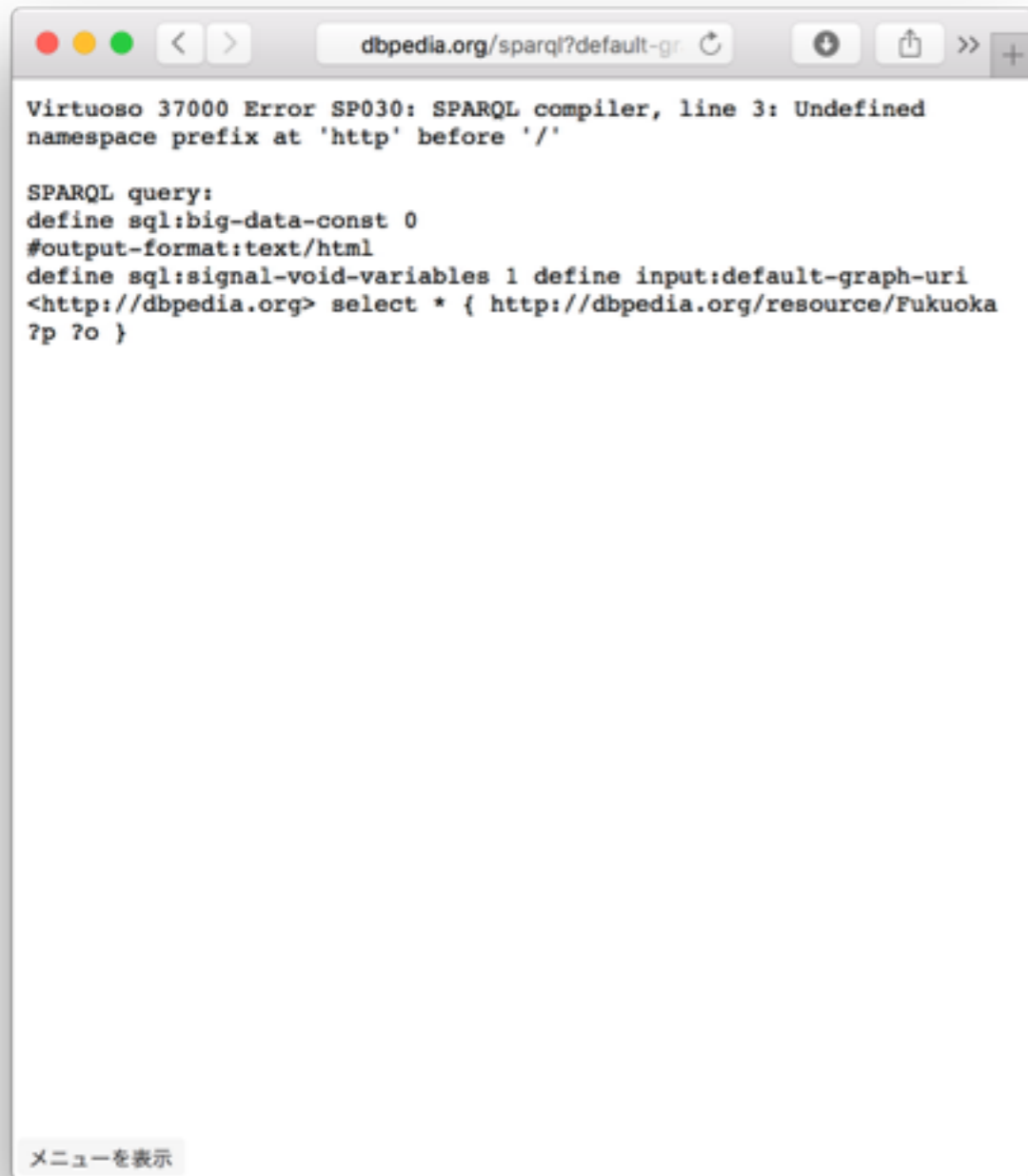
SELECT * { <グラフパターン> }

- ・ グラフパターンの中にある主語、述語、目的語は2種類があります：
 - ・ 変数（「？」で始まる）は全てのデータを合致する
 - ・ 定数（リテラル又はIRI）は合致トリプルに対する条件となる
- ・ 例：
 - ・ 「?s ?p ?o」は全てのトリプルを合致する
 - ・ 「?s 名前 ?o」は述語が「名前」となっているトリプルのみ合致する

演習 3 : 条件をつけてみよう

- DBpediaでは福岡市のIRIは
 - <http://dbpedia.org/resource/Fukuoka>
- ですが、下記のクエリー
 - `select * { ?s ?p ?o }`
- のトリプルパターンの主語に条件を追加することで、福岡市に関わる全てのトリプルを取得してください

演習 3 : 注意点



```
Virtuoso 37000 Error SP030: SPARQL compiler, line 3: Undefined namespace prefix at 'http' before '/'

SPARQL query:
define sql:big-data-const 0
#output-format:text/html
define sql:signal-void-variables 1 define input:default-graph-uri
<http://dbpedia.org> select * { http://dbpedia.org/resource/Fukuoka
?p ?o }
```

このエラーある方はIRIの周りに「<…>」をつけてるかどうか。

演習 4 : 述語に条件を付けてみよう !

<http://dbpedia.org/resource/Fukuoka> ?p ?o

述語

• 下記のクエリー

```
select * {  
  <http://dbpedia.org/resource/Fukuoka>  
  ?p ?o }
```

- のトリプリパターンの述語に条件を追加することで、福岡市の名前を指定するトリプルを取得してください。
- DBpediaだと都市名は「<http://www.w3.org/2000/01/rdf-schema#label>」という述語で指定されている

演習 4 : 答え

```
select * {  
  <http://dbpedia.org/resource/Eukuoka>  
  <http://www.w3.org/2000/01/rdf-schema#label>  
  ?o  
}
```

|||||

IRIは読みにくい!



QnameとCURIE: IRIが読みやすくなるように

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

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

共通プレフィックス



プレフィックス

rdfs:type
rdfs:label

ローカル部分

- ・ CURIE: スラッシュ「/」を使える
- ・ Qname: スラッシュ「/」を使えない

SPARQLでQNAMEの指定の仕方

トリプルパターンの中に

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

がある場合

1

クエリーの前にプレフィックスを宣言する：

prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

2

トリプルパターンの中にIRIの代わりにCURIEを使う

rdfs:type

メモ：

- ・ CURIEは「<>」や「'''」が必要ない
- ・ プレフィックス名は自由だけど、Good Practiceがあります

演習 5 : QNAMEを使ってみよう !

- ・ 下記のクエリーは

```
select * {  
  <http://dbpedia.org/resource/Fukuoka>  
  <http://www.w3.org/2000/01/rdf-schema#label>  
  ?o  
}
```

- ・ IRIをQNAMEに変更してください

「http://dbpedia.org/resource/」 → 「dbpedia」

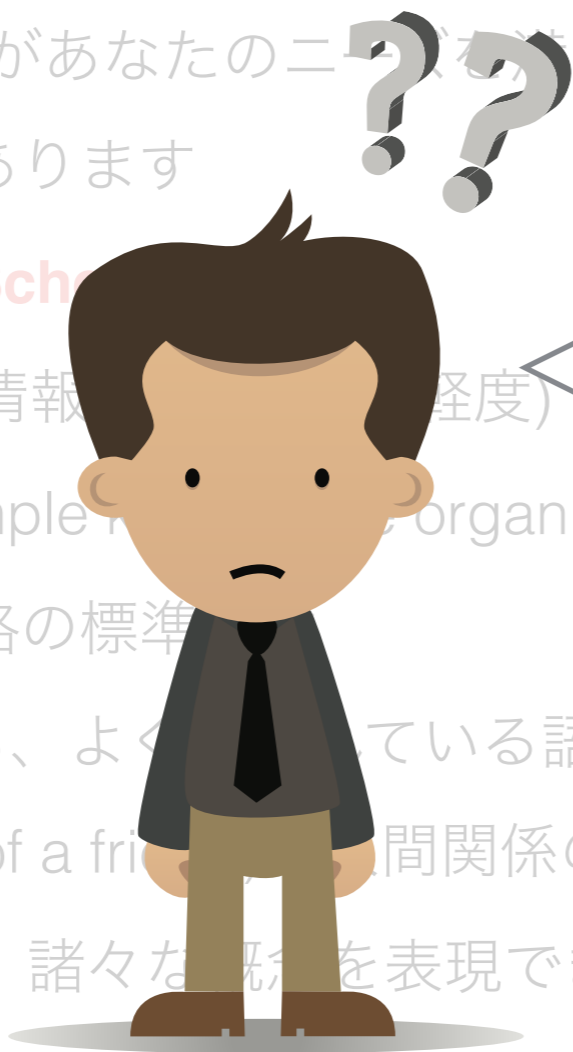
「http://www.w3.org/2000/01/rdf-schema#」 → 「rdfs」

語彙について

- データによって諸々な語彙が必要になる
 - 自分で語彙を決めても問題ありません
 - しかし、誰かがあなたのニーズを満たす語彙を作ったことがあるかも
- W3C規格語彙があります
 - **RDF**と **RDF Schema** (RDFS)
 - **geo** : 地理的情報 (e.g. 緯度 / 軽度)
 - **SKOS** : 「simple knowledge organization system」
 - **XSD**, XML規格の標準語彙
- 規格ではなくても、よく使われている語彙もある
 - **Foaf** (Friend of a friend) : 人間関係の表現
 - **Schema.org** : 諸々な概念を表現できるような語彙 (RDFaやMicrodataでよく使われている)
 - **Dublin Core** : 図書データ
 - **DBPedia**, **Yago** : 図鑑などデータ

語彙について

- データによって諸々な語彙が必要になる
 - 自分で語彙を決めても問題ありません
 - しかし、誰かがあなたのニーズを満たす語彙を作ったことあるかも
- W3C規格語彙があります
 - **RDF**と**RDF Schema**
 - **geo** : 地理的情報 (経度)
 - **SKOS** : 「simple knowledge organization system」
 - **XSD**, XML規格の標準
- 規格ではなくても、よく使われている語彙もある
 - **Foaf** (Friend of a friend) 人間関係の表現
 - **Schema.org** : 諸々な概念を表現できるような語彙 (RDFaやMicrodataでよく使われている)
 - **Dublin Core** : 図書データ
 - **DBPedia, Yago** : 図鑑などデータ



どうやって語彙を探せばいいのか

語彙の探し方について

1 データベースにクエリーを投げることにによって、語彙を閲覧する（演習10と11を参考）

2 グーグルやprefix.ccで語彙の定義を確認し、関連語彙を探索する

prefix.ccを見てみましょう

prefix.cc で語彙を探索

RDFSの場合

- RDFデータのスキーマを表現するため
- W3C規格



popular

1. yago
2. rdf
3. foaf
4. dbp
5. dc
6. owl
7. rdfs
8. ont
9. dbo
10. onto
11. skos
12. geo
13. rss
14. gldp
15. sioc
16. sc
17. fb
18. geonames
19. xsd
20. gr
21. dcterms
22. dct
23. dbpedia
24. akt
25. org
26. commerce

prefix.ccではランキングを見れる


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

Browser window showing the Turtle syntax for the RDF Schema vocabulary (RDFS) on w3.org. The code defines various classes and properties, including rdfs:Resource, rdfs:Class, rdfs:subClassOf, rdfs:subPropertyOf, rdfs:comment, and rdfs:label.

```
@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 .

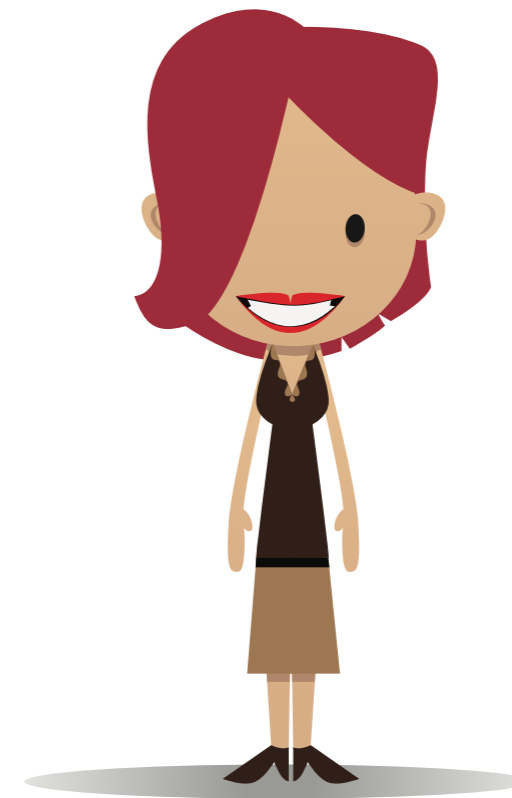
rdfs:subClassOf a rdf:Property ;
  rdfs:isDefinedBy <http://www.w3.org/2000/01/rdf-schema#> ;
  rdfs:label "subClassOf" ;
  rdfs:comment "The subject is a subclass of a class." ;
  rdfs:range rdfs:Class ;
  rdfs:domain rdfs:Class .

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



語彙のTurtle式を 取得する

```
@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 the object." ;
  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 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 .
```

??

ちょっと待って、語彙は Turtleで記述するの？

語彙のTurtle式を取得する

Display a menu

RDFで語彙を記述する

- ・ RDFデータのスキーマをRDFで表現できる
 - ・ **オントロジ**という
- ・ オントロジ自体を表現するために標準語彙があります
 - ・ **RDFS**: W3C規格による基本的なオントロジ語彙
 - ・ **OWL**: W3C規格による複雑なオントロジ語彙
 - ・ **SPIN**: SPARQLでオントロジ条件を表現できる語彙
- ・ オントロジ語彙では複雑な概念が記述可能！
 - ・ モデルドリブン開発へ
- ・ オントロジはRDFデータに組み込むと誰でもSPARQLでデータの構成を探索できるようになる！

プログラムの理論はオントロジ語彙で表現できる箇所が多い

演習 5 の結果を詳しく見る：

言語について

0
"Fukuoka"@en
"فوكوكا ، فوكوكا"@ar
"Fukuoka"@de
"Fukuoka"@es
"Fukuoka"@fr
"Fukuoka"@it
"福岡市"@ja
"Fukuoka (stad)"@nl
"Fukuoka"@pl
"Fukuoka (cidade)"@pt
"Фукьюока"@ru
"福岡市"@zh

ISO 639規格に従う言語指定文

演習6：あなたの故郷の情報 を取得しよう！

- ・ 今までは主語と述語に条件を付けましたが、今回は目的語につけてみよう！
- ・ 「rdfs:label」の目的語としてあなたの故郷の名前を入れることによって、あなたの故郷のIRIを検索してみよう！
- ・ 注意点
 - ・ グラフパタンの主語は条件の代わりに変数になる
 - ・ 名前は言語の指定を忘れずに！

下記のような結果が欲しい

s
http://dbpedia.org/resource/Poitiers
http://www.wikidata.org/entity/Q6616

僕の故郷はフランス語で「Poitiers」ですので、リテラルは”Poitiers”@frです。

FILTERキーワード

- ・ FILTERキーワードは合致するリテラルに更に条件を付けることができます：

```
select * {  
  <グラフパターン> .  
  FILTER (条件)  
}
```

条件の書き方は通常のプログラミング言語に似ています：

- ・ 変数とリテラルの関係性を表現できる (=,<,>,! =など)
- ・ SPARQLは様々な関数も定義する。例 (x,y,zは変数、リテラルまたは式)：

- ・ lang(x) : xの言語
- ・ abs(x) : xの絶対値 (数値のみ)
- ・ isIRI(x) : xはIRIかどうかを確認
- ・ IF(x,y,z) : 分岐。
- ・ IRI(x) : IRIへ変換 (文字列のみ)

演習 7 : 初めてのFILTER

- ・ 下記のクエリー

```
prefix dbpedia:<http://dbpedia.org/resource/>  
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>
```

```
select * { dbpedia:Fukuoka rdfs:label ?o }
```

- ・ を変更し、FILTER文を使って日本語の名前のみ取得する

o
"Fukuoka"@en
"فوكوكا ، فوكوكا"@ar
"Fukuoka"@de
"Fukuoka"@es
"Fukuoka"@fr
"Fukuoka"@it
"福岡市"@ja



下記の結果を求める

o
"福岡市"@ja

演習 8 : より複雑なグラフパターン を使ってみよう！

- ・ グラフパターンでは複数のトリプルを指定できます。
- ・ 「.」で区切る。例：{ s1 p1 o1 . s2 p2 o2 }
- ・ 例えば、下記のグラフパターンは福岡の名前と人口を合致する：

```
dbpedia:Fukuoka rdfs:label ?name .
```

```
dbpedia:Fukuoka http://dbpedia.org/property/populationTotal ?pop .
```

- ・ 問題：上のグラフパターン使ったクエリーを実行してみよう

演習 8 の結果

name	pop
"Fukuoka"@en	1483052
"فوكوكا ، فوكوكا"@ar	1483052
"Fukuoka"@de	1483052
"Fukuoka"@es	1483052
"Fukuoka"@fr	1483052
"Fukuoka"@it	1483052
"福岡市"@ja	1483052
"Fukuoka (stad)"@nl	1483052
"Fukuoka"@pl	1483052
"Fukuoka (cidade)"@pt	1483052
"Фукуюка"@ru	1483052
"福岡市"@zh	1483052

```
prefix dbpedia:<http://dbpedia.org/resource/>
prefix dbpedia-p:<http://dbpedia.org/property/>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

select * {
  dbpedia:Fukuoka rdfs:label ?name .
  dbpedia:Fukuoka dbpedia-p:populationTotal ?pop .
}
```

人口は何度も出てきます。

それは、「dbpedia:Fukuoka rdfs:label ?name .」
が何度も合致するから：合致した各名前に人口をクエ
リーしてしまう。

演習 8 の改善

```
prefix dbpedia:<http://dbpedia.org/resource/>
prefix dbpedia-p:<http://dbpedia.org/property/>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

select * {
  dbpedia:Fukuoka rdfs:label ?name .
  dbpedia:Fukuoka dbpedia-p:populationTotal ?pop .
}
```



```
prefix dbpedia:<http://dbpedia.org/resource/>
prefix dbpedia-p:<http://dbpedia.org/property/>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

select * {
  dbpedia:Fukuoka rdfs:label ?name .
  dbpedia:Fukuoka dbpedia-p:populationTotal ?pop .
  FILTER(lang(?name)="ja")
}
```

name	pop
"Fukuoka"@en	1483052
"فوكوكا ، فوكوكا"@ar	1483052
"Fukuoka"@de	1483052
"Fukuoka"@es	1483052
"Fukuoka"@fr	1483052
"Fukuoka"@it	1483052
"福岡市"@ja	1483052
"Fukuoka (stad)"@nl	1483052
"Fukuoka"@pl	1483052
"Fukuoka (cidade)"@pt	1483052
"Фукуюока"@ru	1483052
"福岡市"@zh	1483052



name	pop
"福岡市"@ja	1483052

グラフパターンを短くする

主語 の再利用

```
<http://city.fukuoka.lg.jp>  
  rdfs:type schema:WebSite ;  
  schema:about db:Fukuoka ;  
  schema:lastReviewed "2015-2-1"^^xsd:date ;  
  rdfs:label "Fukuoka city official homepage"@en ;  
  rdfs:label "福岡市公式ホームページ"@ja .
```

セミコロンで述語・目的語を句切る

最後だけドットを付ける

述語 の再利用

```
<http://city.fukuoka.lg.jp>  
  rdfs:type schema:WebSite ;  
  schema:about db:Fukuoka ;  
  schema:lastReviewed "2015-2-1"^^xsd:date ;  
  rdfs:label  
    "Fukuoka city official homepage"@en,  
    "福岡市公式ホームページ"@ja .
```

コロンで目的語を句切る

演習 9 : 前のクエリを短く しましょう！

```
prefix dbpedia:<http://dbpedia.org/resource/>
prefix dbpedia-p:<http://dbpedia.org/property/>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

select * {
  dbpedia:Fukuoka rdfs:label ?name .
  dbpedia:Fukuoka dbpedia-p:populationTotal ?pop .
  FILTER(lang(?name)="ja")
}
```

セミコロンを使って主語
を再利用しましょう

DISTINCTキーワード

- ・ 変数リストの中に「DISTINCT」を指定すると、何度も同じ結果が合致しても一回しか戻さない
- ・ 例：下記のクエリーはDBpediaのデータベースが使っているの全ての述語を返す

```
select DISTINCT ?p {  
    ?s ?p ?o  
}
```

演習10：DBpediaの語彙を 探索してみよう（1）

- ・ DISTINCTキーワードを使って、福岡市のデータに使っている全ての述語を取得しましょう！

演習11：DBpediaの語彙を 探索してみよう（2）

- ・ 今までは下記の述語を使用しました：
 - ・ `rdfs:label`：都市の名前
 - ・ `dbpedia-p:PopulationTotal`：人口
- ・ 都市の国を表すプロパティを見つけてください
- ・ ヒント：福岡市のデータを見て探索するのはおすすめ
 - ・ 「福岡は日本に位置する」という概念を表すトリプルを見つける

rdf:typeについて

- ・ 「<http://www.w3.org/1999/02/22-rdf-syntax-ns#>」 はW3CのRDF規格の標準的な語彙
 - ・ 普段は「rdf」というプレフィックスを使う
- ・ 「rdf:type」 はリソースのタイプを表すような述語
- ・ タイプの概念はオブジェクト指向言語に似ている
 - ・ オントロジーを使ってオブジェクトのメンバーなどを定義できる
- ・ ただし、RDFでは、あるリソースは複数のタイプを持つことができる
- ・ SPARQLで「<http://www.w3.org/1999/02/22-rdf-syntax-ns#/type>」のかわりに「a」キーワードを利用できる
 - ・ 例：?s a <<http://dbpedia.org/ontology/City>>

演習12：タイプを見てみよう

- ・ 福岡市のリソースのすべてのタイプを取得する
- ・ リマインダー：福岡市のIRIは「<http://dbpedia.org/resource/Fukuoka>」

t
http://www.w3.org/2002/07/owl#Thing
http://dbpedia.org/ontology/Place
http://dbpedia.org/ontology/Location
http://www.wikidata.org/entity/Q486972
http://www.wikidata.org/entity/Q515
http://dbpedia.org/ontology/City
http://dbpedia.org/ontology/PopulatedPlace
http://dbpedia.org/ontology/Settlement
http://www.w3.org/2003/01/geo/wgs84_pos#SpatialThing
http://schema.org/City
http://schema.org/Place
http://umbel.org/umbel/rc/City
http://umbel.org/umbel/rc/Location Underspecified

このような結果が欲しい！

演習13：より複雑なグラフパターンを使ってみよう

- ・ 下記の述語を使って：

`http://dbpedia.org/ontology/country`

- ・ 日本のすべての都市の日本語の名前と人口を取得する

- ・ ヒント、注意点：

- ・ 日本のIRIは「`dbpedia:Japan`」

- ・ 都市以外にも「`http://dbpedia.org/ontology/country`」を使うリソースがあるので、都市（英語：“city”）のみ合致するように工夫する必要がある

name	pop
"我孫子市"@ja	132999
"昭島市"@ja	112521
"秋田市"@ja	323232
"奄美市"@ja	44561
"青森市"@ja	293442
"旭市"@ja	68265
"浅口市"@ja	36985
"熱海市"@ja	28870

このような結果が欲しい！

演習14：数値にFILTERをかけよう！

- ・ 演習12のクエリーを変更して、FILTERキーワードを使って人口が10万を超えてる都市だけ取得する
- ・ ヒント：FILTERを複数使いたい場合、やり方が2つあります
 - ・ FILTERキーワードを複数使って、ドットで区切る
 - ・ 例：FILTER(条件1) . FILTER(条件2)
 - ・ FILTERの条件はブール理論を使う
 - ・ 例：FILTER(条件1 && 条件2)

演習15：ソートする

- ・ 「ORDER BY」 キーワードをクエリーの後ろにつけると結果をソートできる。例 (?popは人口とする場合) :
 - ・ SELECT * { … } ORDER BY ?pop : 人口の少ない都市から
 - ・ SELECT * { … } ORDER BY DESC(?pop) : 人口の多い都市から
- ・ 演習13のクエリーを人口の多い都市から結果を出すように変更してください

演習16：GROUP BYと COUNTとは？

下記のクエリーは何を検索しているのかを当ててみて
ください：

```
prefix dbpedia:<http://dbpedia.org/resource/>  
prefix dbpedia-p:<http://dbpedia.org/property/>  
prefix dbpedia-o:<http://dbpedia.org/ontology/>  
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>
```

```
select ?country COUNT(DISTINCT ?s) {  
  ?s a dbpedia-o:City ;  
  dbpedia-o:country ?country ;  
  dbpedia-p:populationTotal ?pop .  
  FILTER(?pop > 100000)  
} GROUP BY ?country
```

演習17 : 「as」 キーワード

```
prefix dbpedia:<http://dbpedia.org/resource/>
prefix dbpedia-p:<http://dbpedia.org/property/>
prefix dbpedia-o:<http://dbpedia.org/ontology/>
prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>

select ?country (group_concat(?name, ",") as ?names) {
  ?s dbpedia-o:country ?country ;
  rdfs:label ?name ;
  dbpedia-p:populationTotal ?pop .
  FILTER(lang(?name)="ja" && ?pop > 100000) .
} GROUP BY ?country
```

SERVICEを使った

結合クエリー

同時複数のSPARQL
エンドポイントにク
エリーを投げる

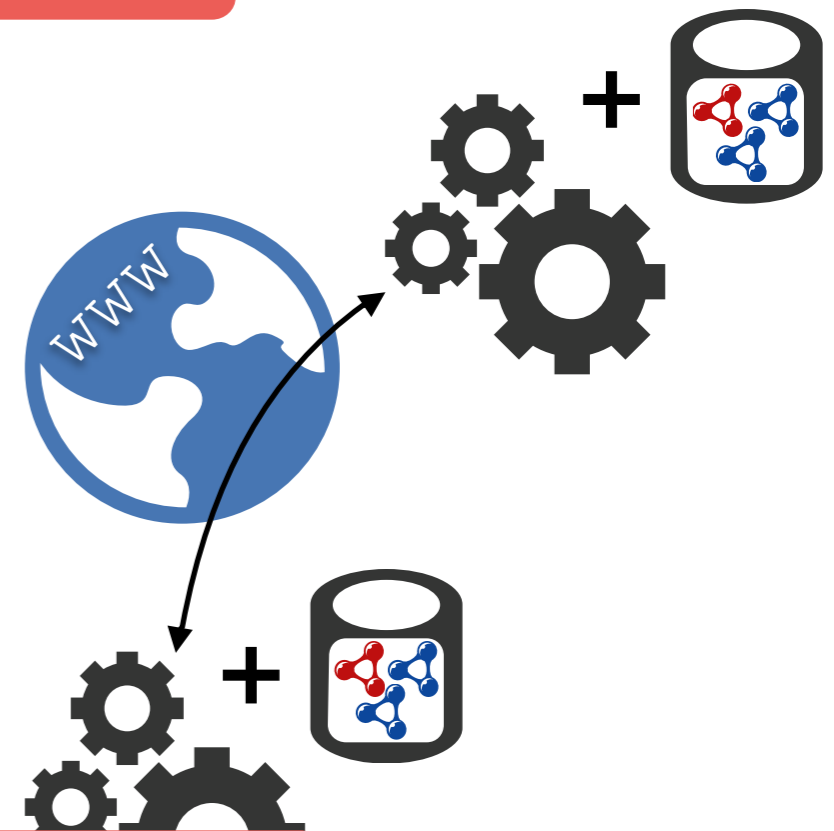
```
SELECT ?facility_name ?city_name
WHERE {
  ?s a bodic:PublicFacility ;
    rdfs:label ?facility_name ;
    bodic:locatedIn ?city_iri .
  SERVICE <http://dbpedia.org/sparql> {
    ?city_iri rdfs:label ?city_name
  }
}
```

dbpediaのSPARQLエン
ドポイントのURL

・ SPARQLクエリーの中から **マッシュアップ**
できる

・ しかし、SPARQLエンドポイントに対して
負担が大きいため、公開エンドポイントは
SERVICEを禁止することが多い

Jenaのようなツールから簡単
にローカルで使えます



SERVICEを使った

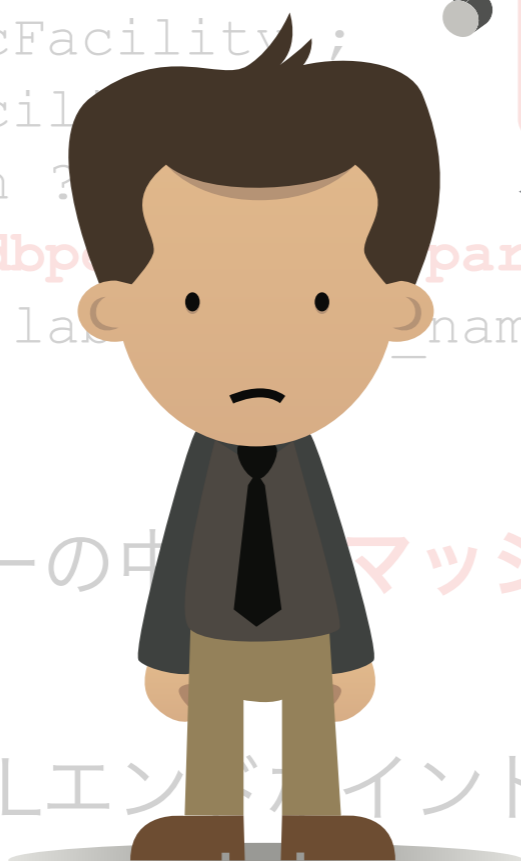
結合クエリー

同時複数のSPARQL
エンドポイントにク
エリーを投げる

```
SELECT ?facility_name ?city_name
WHERE {
  ?s a bodic:PublicFacility;
  rdfs:label ?facility_name;
  bodic:locatedIn ?city_iri;
  SERVICE <http://dbpedia.org/sparql>
  ?city_iri rdfs:label ?city_name
}
```



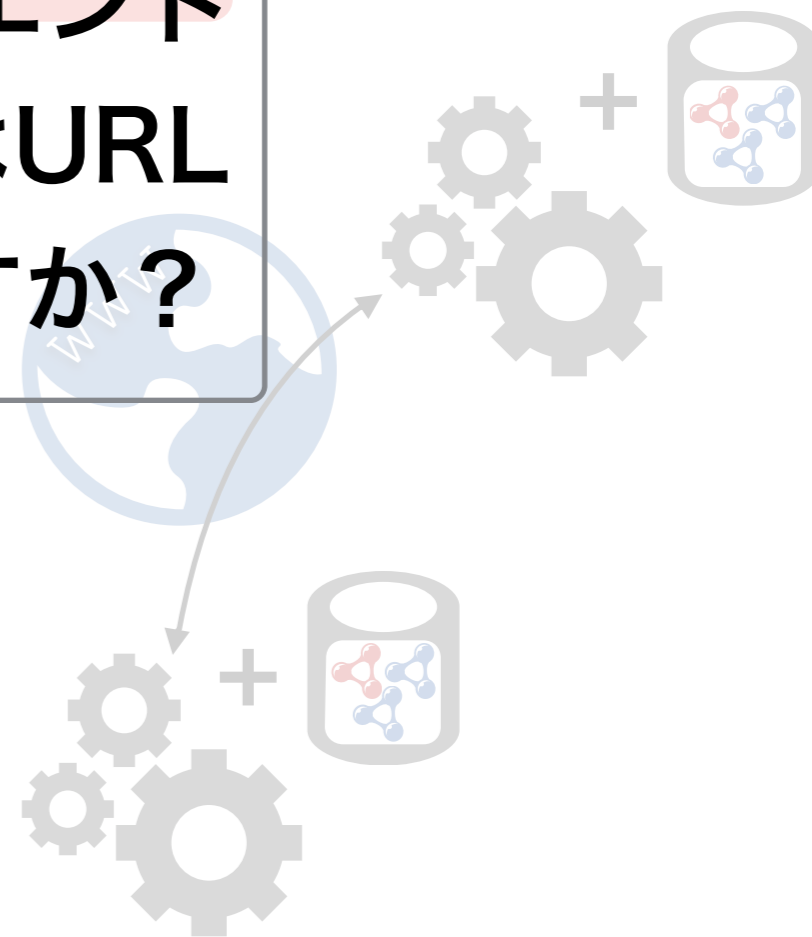
dbpediaのSPARQLエン
ドポイントのURL
**SPARQLエンド
ポイントはURL
がありますか？**



・ SPARQLクエリーの中でマッシュアップ
できる

・ しかし、SPARQLエンドポイントに対して
負担が大きいため、公開エンドポイントは
SERVICEを禁止する

Jenaのようなツールから簡単
にローカルで使えます



SPARQLはクエリー言語
だけではない

SPARQL Protocol And RDF Query Language

- ・ SPARQL 1.1では2つのプロトコルを定義している
 - ・ SPARQLクエリーを投げるためHTTP REST API
 - ・ SPARQLエンドポイント間で通信するためのプロトコル
- ・ 簡単にHTTP REST APIを見ましょう

SPARQL HTTP REST APIについて

- ・ SPARQLエンドポイントサーバーAPIが<http://endpoint.org>とすると
- ・ SPARQL検索
 - ・ <http://endpoint.org/sparql>
 - ・ 読み込み専用クエリーを投げるため (SELECT/CONSTRUCT/ASK/DESCRIBE)
 - ・ <http://endpoint.org/update>
 - ・ SPARQL UPDATE クエリーを投げるため (LOAD/INSERT/DELETE/DROP/COPY/MOVE)
- ・ 直接RDFグラフを扱うAPI(http://endpoint.org?graph=graph_name)
 - ・ **GET** グラフを取得
 - ・ **PUT** グラフを書き換える
 - ・ **POST** グラフを更新
 - ・ **DELETE** グラフを削除

グラフとは

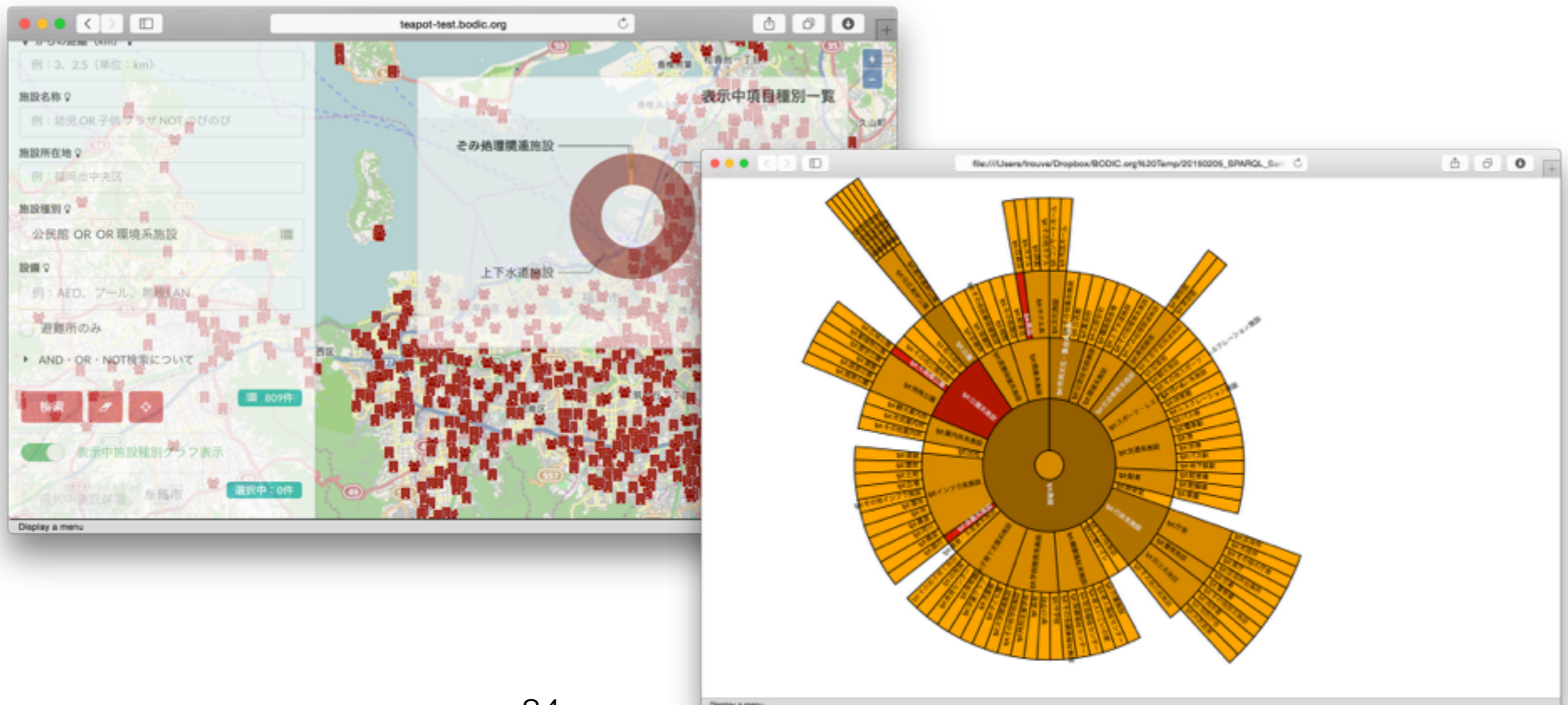
- ・ RDFストアはトリプルをグラフ単位で管理する (関係データベースのテーブルのように)
- ・ SPARQLでGRAPHを指定しないと「default」というグラフから検索する

まとめ

- ・ 本日は基本的なSPARQLクエリーを勉強しました
- ・ まだ勉強できなかった機能が沢山あります：
 - ・ 結合クエリー演習
 - ・ キーワード：BIND、UNION、MINUSなど
 - ・ その他の関数
 - ・ クエリー種：CONSTRUCT、ASK、DESCRIBE、SPARQL UPDATEクエリー群
 - ・ オントロジ
- ・ SPARQLはセマティックウェブ以外でもよく使われている
 - ・ NoSQLデータベースの1種類として
 - ・ 向き不向きがありますが、ソフトウェア者のツールボックスにあると便利

次回（予定）

- ・ 地図とグラフでSPARQL結果を可視化する
- ・ クライアント側（AJAX）のウェブ開発に関する演習



お疲れさまでした！