

# Linked Open Data 勉強会 2020

## ① LODの基礎・作成・公開

LODチャレンジ実行委員会

人工知能学会セマンティックWebとオントロジー研究会 企画委員



LOD challenge



# 内容

---

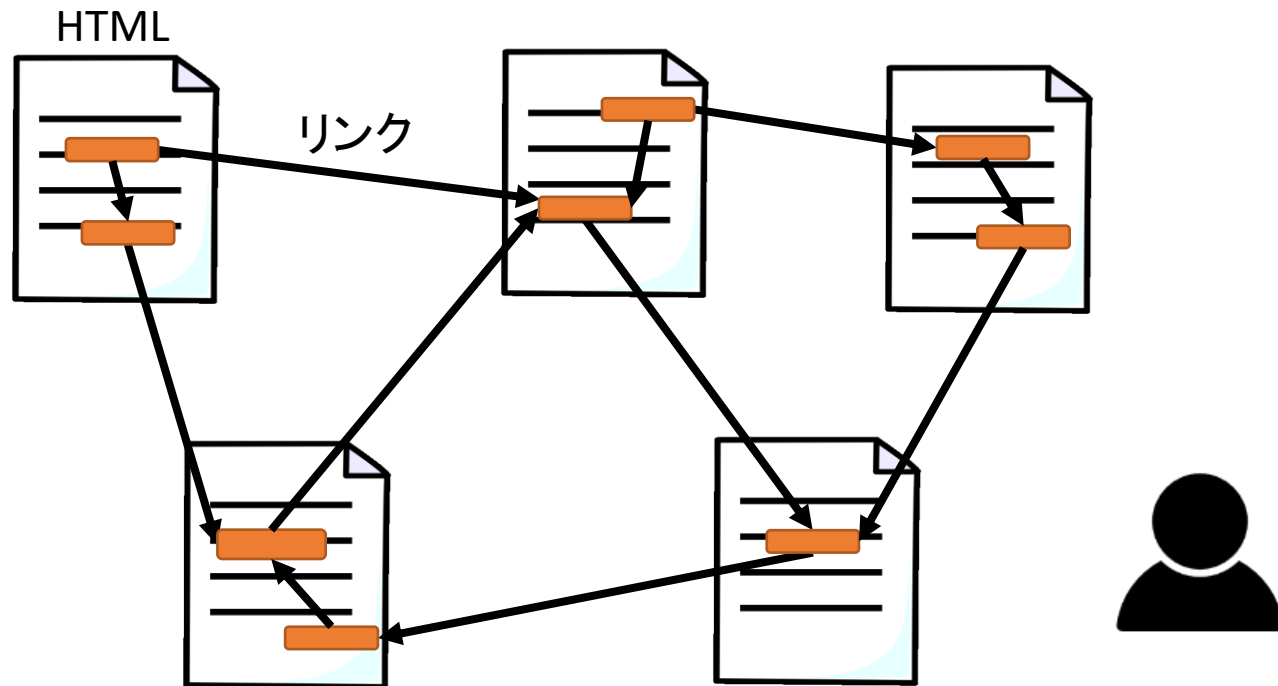
- LODの基礎
- RDF(初歩)
- RDF(クラス,インスタンス,サブクラス, サブプロパティ)
- RDF, LODの作成
- RDF, LODの格納・公開

# LODの基礎

概要

# 文書のWeb

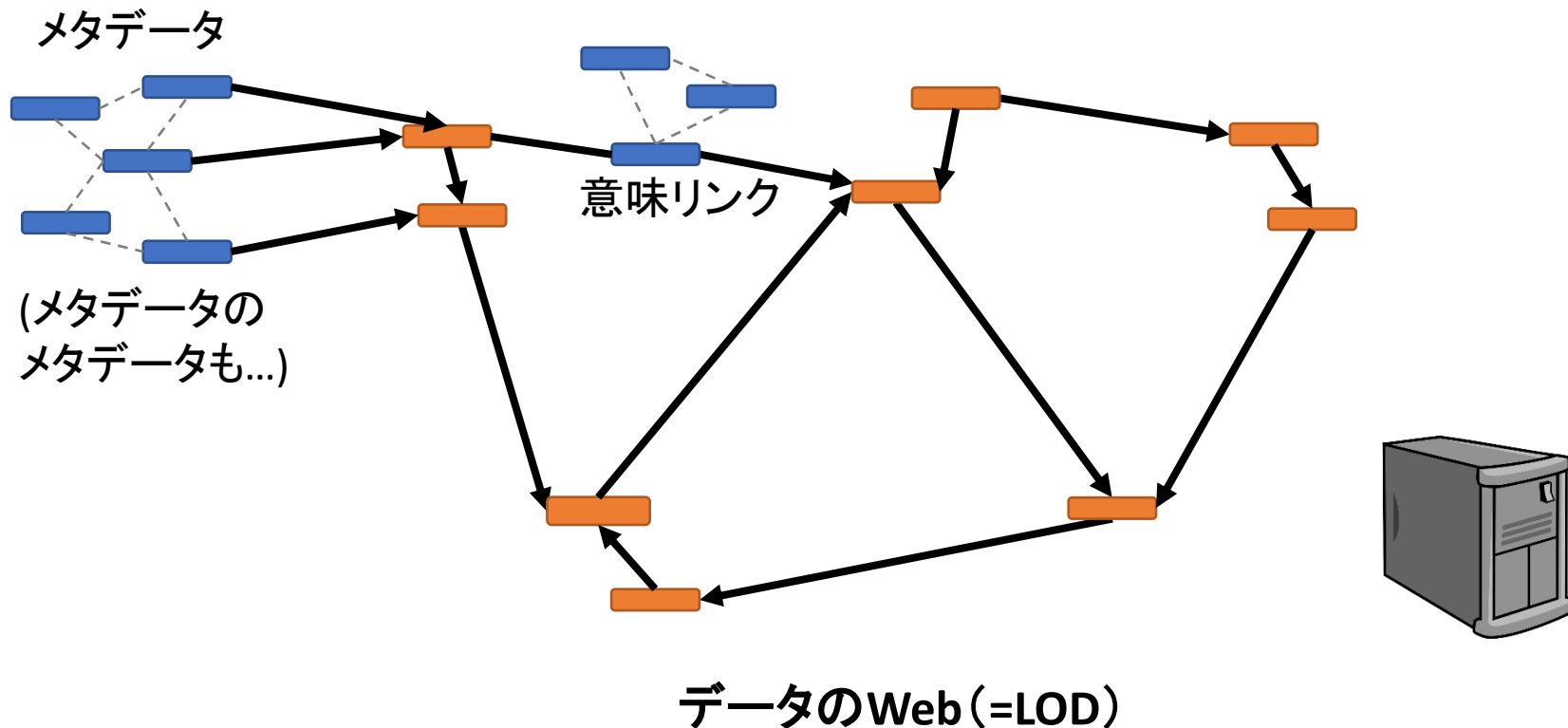
- HTML文書内に書かれたハイパーリンクにより文書同士をリンク
- 人間のためのWebであり, 文書に書かれた単語の意味や, リンクの意味をコンピュータが理解することはできない



従来のWebの構造

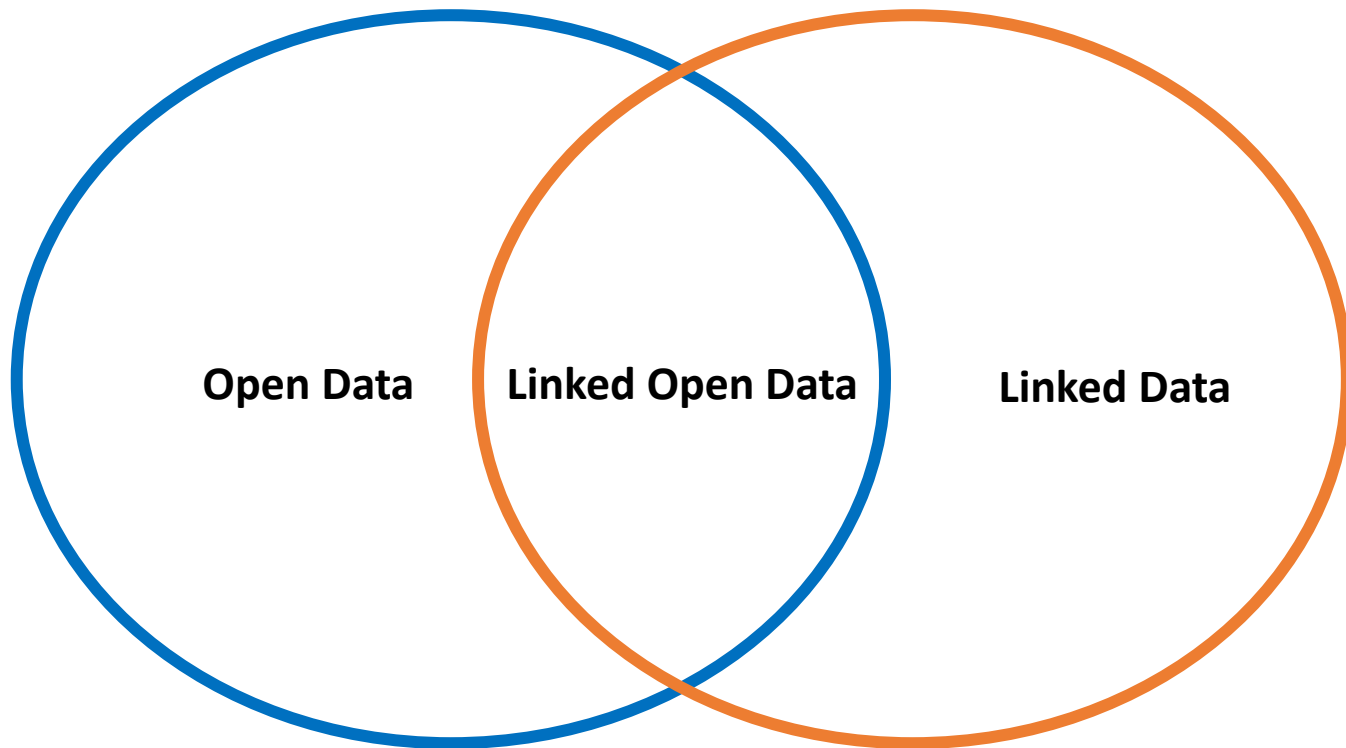
# データのWeb

- すべての文書/データにメタデータをつけて意味を付与する
- さらにリンクにも意味をつけることでWebが膨大な知識空間に  
＝機械処理可能な, コンピュータのためのデータのWeb



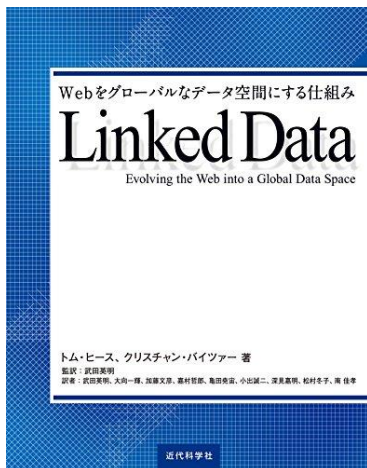
# Linked (Open) Data

- すべての事物に一意なURIを付与し, **データを主語・述語・目的語の三組(トリプル)で記述し**, 「データのWeb」を実現させる
- 厳密には, 後述する4つの原則にしたがって公開



# Linked Dataの基本原則

1. あらゆる事物にURIを付与すること
2. 誰でも事物の内容が確認できるように、URIはHTTP経由で参照できること
3. URIを参照した時は、標準の技術(RDFやSPARQL等)を使用して関係する有用な情報を利用できるようにすること
4. より多くの事物を発見できるように、他のURIへのリンクを含めること



詳しく理解するには下記の本がおすすめ

**Linked Data: Webをグローバルなデータ空間にする仕組み**

トム・ヒース、クリスチャン・バイツァー 著

監訳：武田英明

訳者：武田英明、大向一輝、加藤文彦、嘉村哲郎、亀田亮宙、小出誠二、深見嘉明、松村冬子、南 佳孝



# LODクラウド図

- 一つの円が数千トリプル～数十億トリプルからなるLOD

## Legend

Cross Domain

Geography

Government

Life Sciences

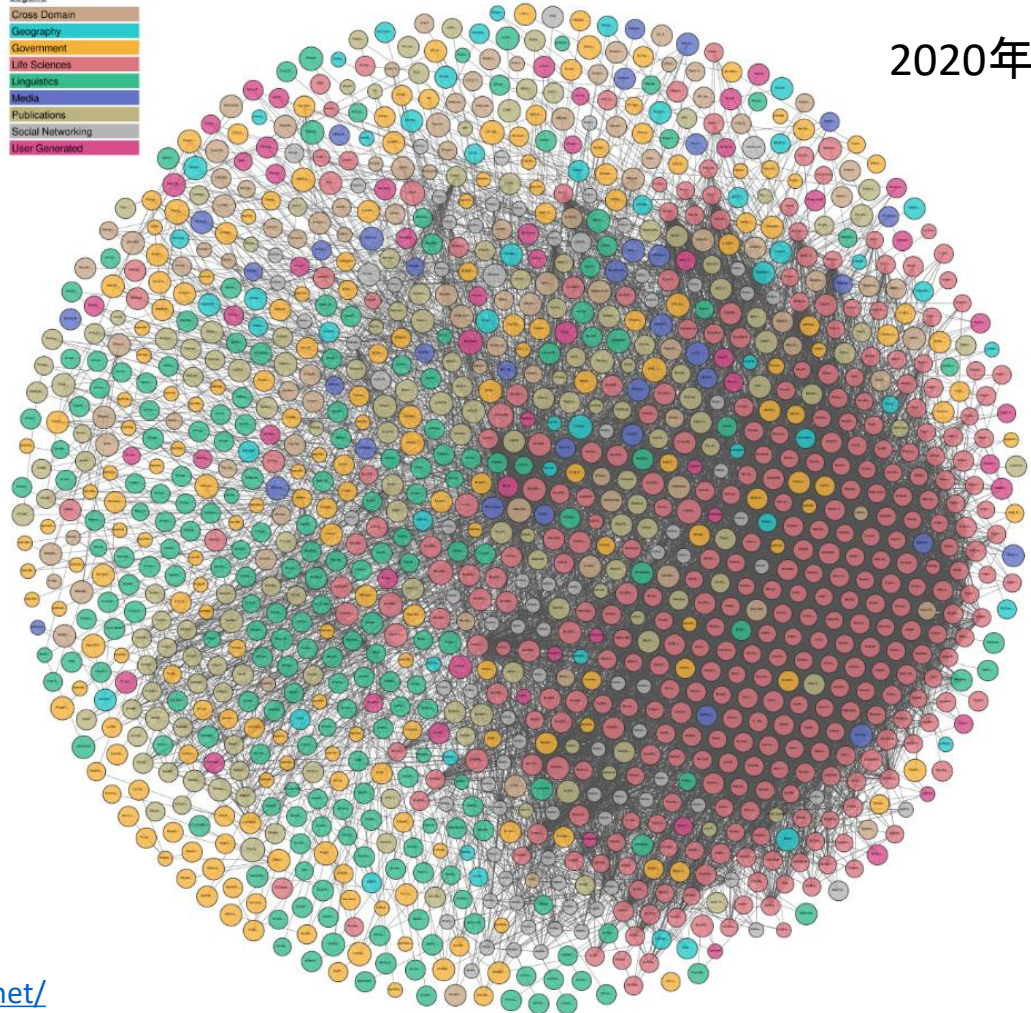
Linguistics

Media

Publications

Social Networking

User Generated



一定の基準を満たしたものののみ図示

The Linked Open Data Cloud, <https://lod-cloud.net/>



# RDF

LODを理解する上で最も重要な技術

# RDF (Resource Description Framework) とは

---

すべての情報を

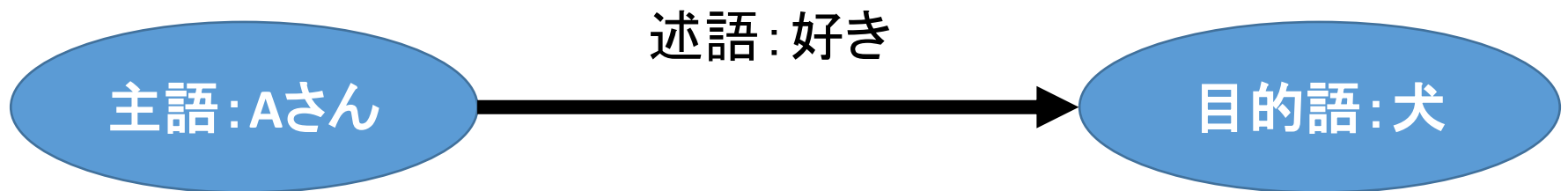
- ・主語(リソース)
- ・述語(プロパティ)
- ・目的語(リソースorリテラル)

の三組み(トリプルという)で記述する方法を提供

# 結局RDFはなにか

その実体は、ラベル付き有向グラフ (※)

情報 「Aさんは犬が好き」



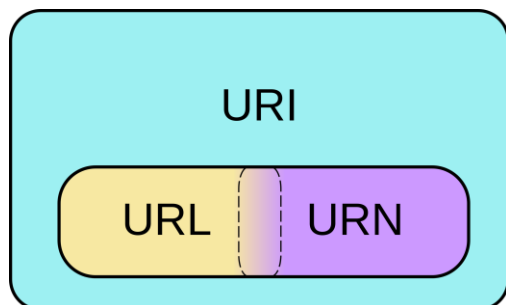
このようにグラフ形式で図示したものをRDFグラフと呼ぶ

(※)厳密にはRDFは枠組みなので、RDFの枠組みに従ったデータがラベル付き有向グラフで表現できる

# リソースとは

URI※で識別可能な全ての事物

- ・人物, 書籍, イベントなどの実世界の物事
- ・趣味, 嗜好, 信頼性などの事柄など...



by Peter Krauss, Mysid, Rjgodoy, Surachit /  
CC 表示-継承 3.0

※補足:

**URI: Uniform Resource Identifier**

リソースを指し示す統一識別子

**URL: Uniform Resource Locator**

リソースの場所を識別する

**URN: Uniform Resource Name**

リソースの名前を識別する

# リソースとは

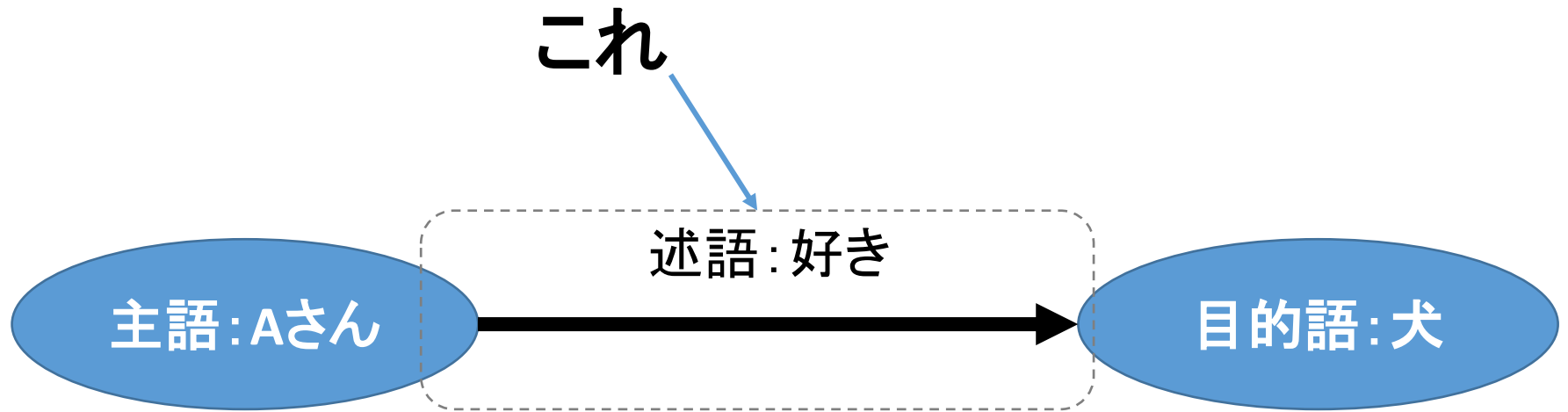
つまり、

事物に一意なURIを振り当ててしまえば、RDFでそれはリソースとして認識される



※本当はブラウザでアクセスできることが望ましいが  
とりあえず内部で重複がなければこれでよい(後で説明)

# プロパティとは



プロパティにも一意なURIを振り当てる

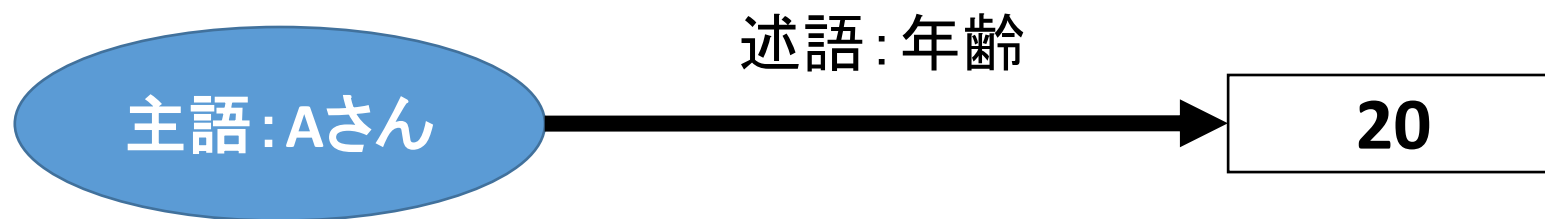
<http://example.com/terms/好き>

# リテラルとは

文字列や数値のこと.

プロパティを持たない.

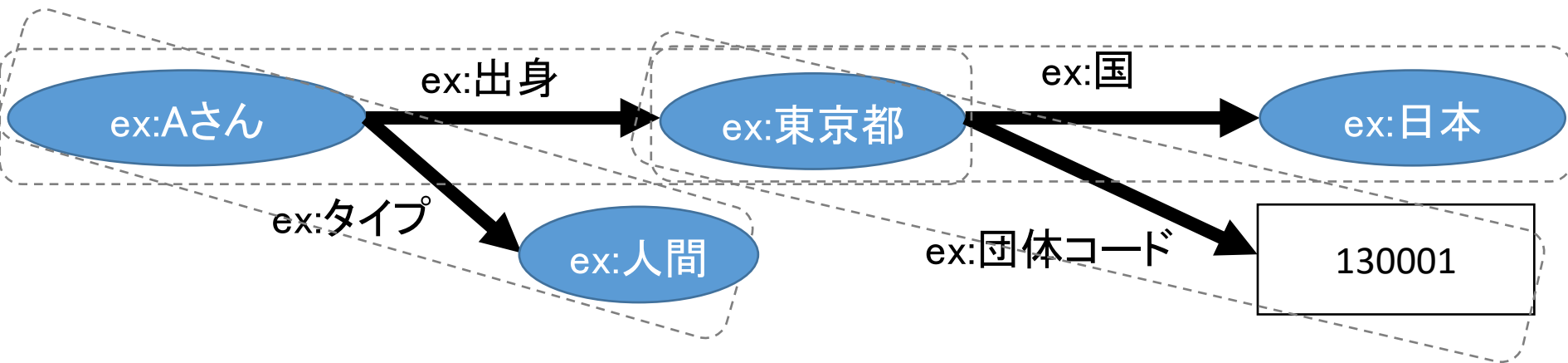
URIを振り当てない.





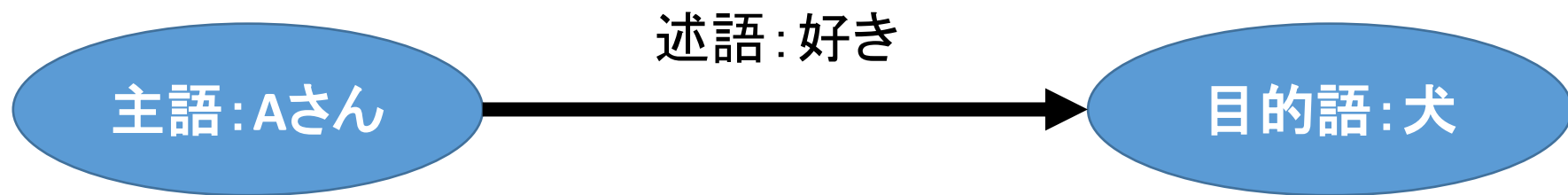
# トリプルの連鎖の例

あるトリプルの目的語のリソースは別のトリプルの主語になりうる。  
リテラルは主語にならない。



4トリプル

# シリアライゼーションフォーマット



↑これは図なのでデータとして記述する必要がある。

XML, N-Triples, Turtle, JSON等で書くことができる。

記述方法が異なるが、結局RDFグラフとして図示すると全て同じになることが特徴！ (= 情報の統一化)

(シリアライゼーションフォーマットの相互変換ツールも充実している)

# N-Triples

- <URI>をスペースで分けて並べる最も原始的な記法。
- トリプルはピリオドで区切る。
- 拡張子は.ntとすることが多い

<http://example.com/terms/Aさん> <http://example.com/terms/名前> "Aさん"@ja .

<http://example.com/terms/Aさん> <http://example.com/terms/好き> <http://example.com/terms/犬> .

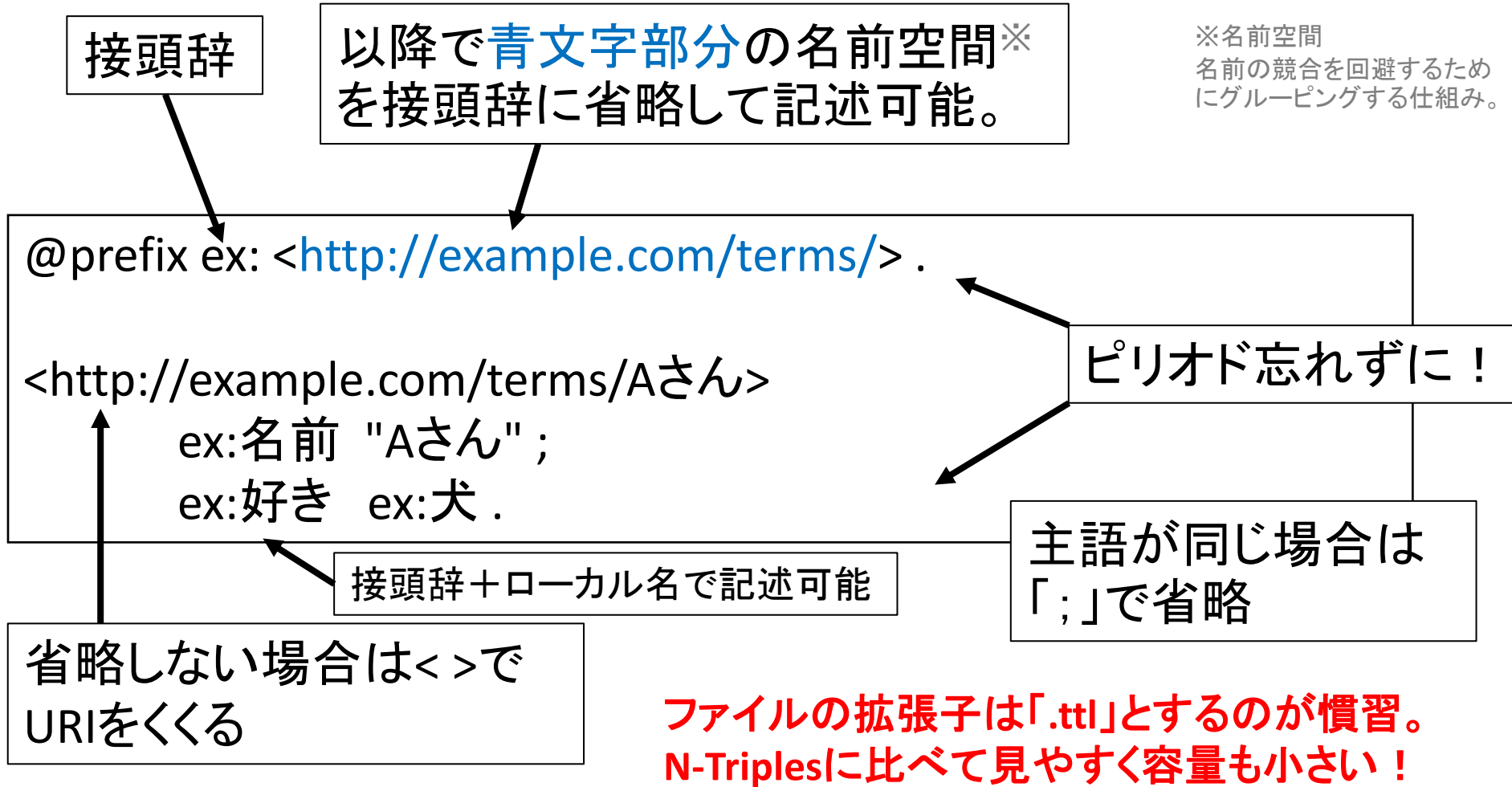
リテラルの言語指定をする  
場合@言語コード

ピリオド忘れずに！

ファイルの拡張子は「.nt」とするのが慣習。  
EXCELデータやCSVからの変換が楽。

N-Triplesは原始的で理解しやすいけど長い！

# Turtle



# XMLタグの入れ子構造でRDFグラフを記述

## RDF/XML

以降で青文字部分の名前空間を接頭辞「rdf:」に省略できる

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<rdf:RDF
```

```
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
```

```
  <rdf:Description rdf:about="http://example.com/terms/Aさん">
```

```
    <http://example.com/terms/好き>
```

名前空間定義していないからURIをフルスペルで書いている

```
    <rdf:Description rdf:about="http://example.com/terms/犬" />
```

```
  </http://example.com/terms/好き>
```

```
</rdf:Description>
```

緑文字部分の行はリソース記述

```
</rdf:RDF>
```

ファイルの拡張子は「.rdf」とするのが慣習。  
XMLエディタで開きたい場合は「.xml」

# JSON-LD

JSONでのデータ交換や、HTMLのscriptタグへの埋め込みによく利用される  
拡張子は.jsonや.jsonld

```
{
  "@context": {
    "ex": "http://example.com/terms/",
  },
  "@id": "http://example.com/terms/Aさん",
  "ex:好き": {
    "@id": "http://example.com/terms/犬"
  }
}
```

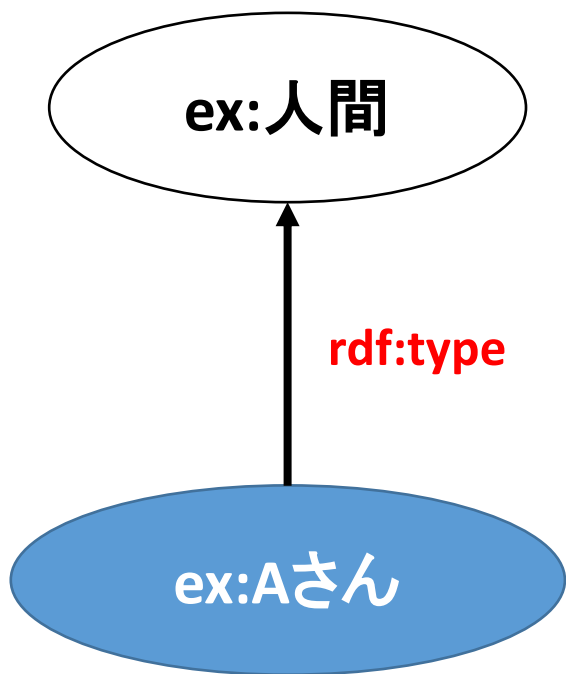
他にもRDF/JSONがある  
<https://www.w3.org/TR/rdf-json/>

# RDF(つづき)

クラス, インスタンス, サブクラス, サブプロパティ



# クラスとインスタンス



「人間」さんは実在しない。  
概念, カテゴリ, グループのニュアンス。  
これをクラス(Class)という

「Aさん」は実在する。  
人間を具体化させたものの一例。  
人間クラスのインスタンスという。

クラス-インスタンスの関係はrdf:typeプロパティを使用する

# クラスとインスタンス

「ex:人間」を単なるリソースではなく  
クラスとして定義する！

書きたい情報「ex:人間はクラスである」



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

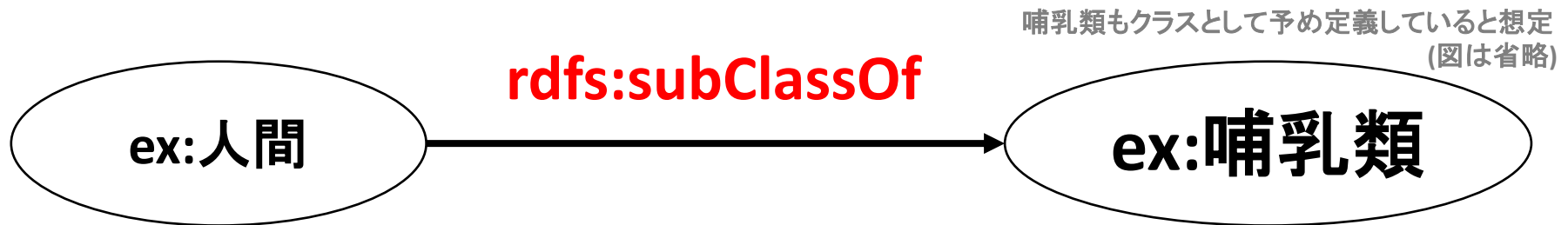
RDFSはクラスを定義するための用語をRDFで提供している。  
↑アクセスしてみるとRDFSのTurtleが見れる

クラスの指定には「rdf:type」を使う決まり

# クラスの階層関係

「ex:人間」を「ex:哺乳類」のサブクラスとして定義する！

書きたい情報「ex:人間はex:哺乳類のサブクラスである」



階層関係を意味するプロパティとして  
rdfs:subClassOfが一般的に使用される。

# クラスの階層関係

「ex:人間」を「ex:哺乳類」のサブクラスとして定義する！

書きたい情報「ex:人間はex:哺乳類のサブクラスである」

**rdfs:subClassOf**

哺乳類もクラスとして予め定義していると想定  
(図は省略)

ex:人間

ex:哺乳類

この定義をした上で

「ex:Aさん rdf:type ex:人間 .」と書くと,

「ex:Aさん rdf:type ex:哺乳類 .」を

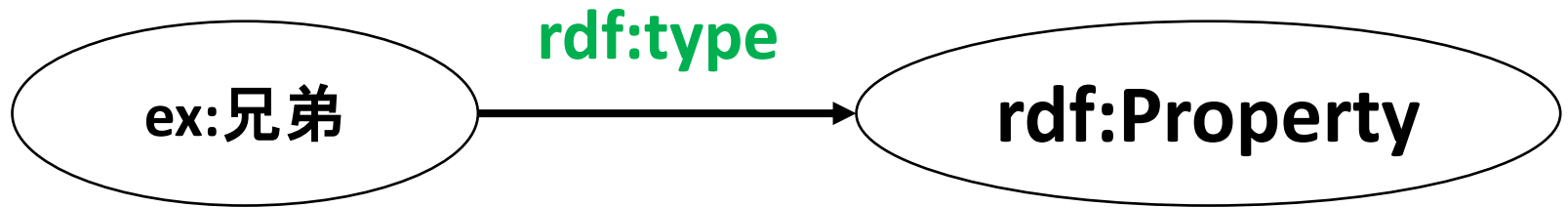
記述しなくても当然導ける＝【推論可能】

推論については今回は説明しません

# プロパティ

「ex:兄弟」をリソースではなくプロパティとして定義する！

書きたい情報「ex:兄弟はプロパティである」



# プロパティの階層関係

「ex:兄弟」を「ex:親族」のサブプロパティとして定義する！

書きたい情報「ex:兄弟はex:親族のサブプロパティである」



兄弟プロパティを使って「A ex:兄弟 B .」と書くと,  
「A ex:親族 B .」が当然導ける＝【推論可能】

# RDF, LODの構築



# LODの例: WikiData

- データをWikipediaのように皆で編集できる

Wikidata logo and navigation links: Main page, Community portal, Project chat, Create a new Item, Create a new Lexeme, Recent changes, Random Item, Query Service, Nearby, Help, Donate, Tools, What links here, Related changes, Special pages, Permanent link, Page information, Cite this page, Concept URI.

Item: **person** (Q215627)

being that has certain capacities or attributes constituting personhood (avoid use with P31; use Q5 for humans)

persons

▼ In more languages

Configure

Language	Label	Description	Also known as
English	person	being that has certain capacities or attributes constituting personhood (avoid use with P31; use Q5 for humans)	persons
Japanese	人	個性を持った人物に認められる特定の属性を持った存在を指す抽象的用語	個人 人物
Korean	개인	존재에 대한 추상적인 정의 (범주(P31)의 항목으로 쓰지 마시고 사람(Q5)을 대신 쓰세요)	자연인 인 (법률) 인물 사람

All entered languages

Statements

subclass of

subject

▼ 0 references

agent

▼ 0 references

<https://www.wikidata.org/wiki/Q215627>

# LODの例: WikiData

- 様々なシリアライゼーションフォーマットで利用可能

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ontolx: <http://www.w3.org/ns/lemon/ontolx#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix wikibase: <http://wikiba.se/ontology#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix schema: <http://schema.org/> .
@prefix cc: <http://creativecommons.org/ns#> .
@prefix geo: <http://www.opengis.net/ont/geosparql#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix wd: <http://www.wikidata.org/entity/> .
@prefix data: <https://www.wikidata.org/wiki/Special:EntityData/> .
@prefix s: <http://www.wikidata.org/entity/statement/> .
@prefix ref: <http://www.wikidata.org/reference/> .
@prefix v: <http://www.wikidata.org/value/> .
@prefix wdt: <http://www.wikidata.org/prop/direct/> .
@prefix wdt_n: <http://www.wikidata.org/prop/direct-normalized/> .
@prefix p: <http://www.wikidata.org/prop/> .
@prefix ps: <http://www.wikidata.org/prop/statement/> .
@prefix ps_v: <http://www.wikidata.org/prop/statement/value/> .
@prefix ps_n: <http://www.wikidata.org/prop/statement/value-normalized/> .
@prefix pq: <http://www.wikidata.org/prop/qualifier/> .
@prefix pq_v: <http://www.wikidata.org/prop/qualifier/value/> .
@prefix pq_n: <http://www.wikidata.org/prop/qualifier/value-normalized/> .
@prefix pr: <http://www.wikidata.org/prop/reference/> .
@prefix pr_v: <http://www.wikidata.org/prop/reference/value/> .
@prefix pr_n: <http://www.wikidata.org/prop/reference/value-normalized/> .
@prefix wdn: <http://www.wikidata.org/prop/novalue/> .

data:Q215627 a schema:Dataset ;
  schema:about wd:Q215627 ;
  cc:license <http://creativecommons.org/publicdomain/zero/1.0/> ;
  schema:softwareVersion "1.0.0" ;
  schema:version "1260618037"^^xsd:integer ;
  schema:dateModified "2020-08-19T13:41:15Z"^^xsd:dateTime ;
  wikibase:statements "57"^^xsd:integer ;
  wikibase:identifiers "14"^^xsd:integer ;
  wikibase:sitelinks "77"^^xsd:integer .

wd:Q215627 a wikibase:Item .

<https://it.wikiquote.org/wiki/Persona> a schema:Article ;
  schema:about wd:Q215627 ;
  schema:inLanguage "it" ;
  schema:isPartOf <https://it.wikiquote.org/> ;
  schema:name "Persona"@it .

<https://it.wikiquote.org/> wikibase:wikiGroup "wikiquote" .

<https://ja.wikiquote.org/wiki/%E4%BA%E9%96%93> a schema:Article ;
  schema:about wd:Q215627 ;
  schema:inLanguage "ja" ;
  schema:isPartOf <https://ja.wikiquote.org/> ;
  schema:name "人間"@ja .

...
```

<https://www.wikidata.org/entity/Q215627.ttl>

# WikiDataへの移動

- Wikipediaのページから移動が可能

The screenshot shows the Wikipedia article for '人間' (Human). In the left sidebar, the link 'このページを引用' (Cite this page) is highlighted with a red box. A callout box with a red border points to this link, containing the text: 'ページ情報', 'ウィキデータ項目', and 'このページを引用'.

ウィキペディア  
フリー百科事典

メインページ  
コミュニティ・ポータル  
最近の出来事  
新しいページ  
最近の更新  
おまかせ表示  
練習用ページ  
アップロード (ウィキメディア・コモンズ)

ヘルプ  
ヘルプ  
井戸端  
お知らせ  
バグの報告  
寄付  
ウィキペディアに関するお問い合わせ

ツール  
リンク元  
関連ページの更新状況  
ファイルをアップロード  
特別ページ  
この版への固定リンク  
ページ情報  
このページを引用  
ウィキデータ項目  
このページを引用  
ウィキデータ項目  
ブックの新規作成  
PDF形式でダウンロード  
印刷用バージョン  
他のプロジェクト  
コモンズ  
ウィキクオート  
他言語版

ログインしていません トーク 投稿記録 アカウント作成 ログイン

ページ ノート 閲覧 編集 履歴表示 Wikipedia内を検索

## 人間

出典: フリー百科事典『ウィキペディア (Wikipedia)』

この記事は中立的な観点に基づく疑問が提出されているか、議論中です。そのため、中立的でない偏った観点から記事が構成されているおそれがあり、場合によっては記事の修正が必要です。議論はノート「人間観の遷移がキリスト教及びヨーロッパの視点ばかりで偏っています。」節を参照してください。(2018年12月)

その他の用法については「人間 (曖昧さ回避)」、「人」、「人類」をご覧ください。

人間 (にんげん、英: human being<sup>[1]</sup>) とは、以下の概念を指す。

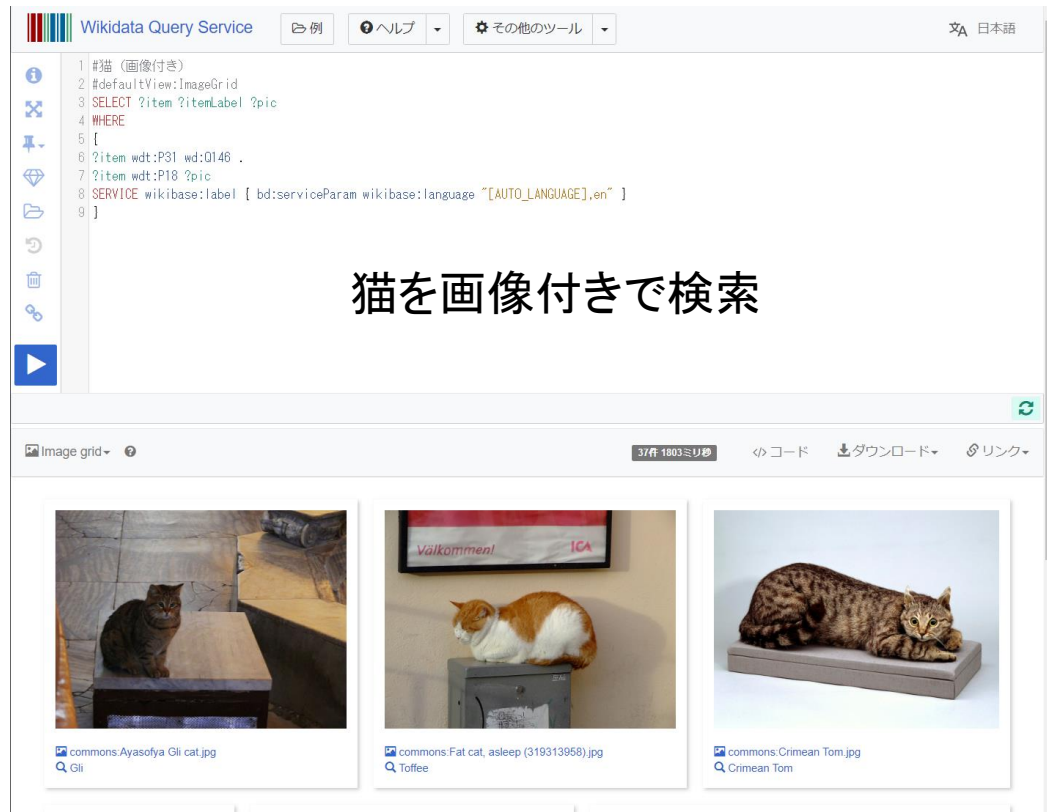
- 人のすむところ。世の中<sup>[2]</sup>。世間。人が生きている人と人の関係の世界。またそうした人間社会の中で脆くはかないさまを概念的に表すことば。
- (社会的なありかた、人格を中心にとらえた) 人。また、その全体<sup>[2]</sup>。
- ひとがら。「人物」<sup>[2]</sup>。

目次 [非表示]

- 1 概説
- 2 人間観の遷移
  - 2.1 旧約聖書
  - 2.2 古代ギリシャ
  - 2.3 キリスト教
  - 2.4 中世〜近世
  - 2.5 近代
  - 2.6 現代
- 3 教育と人間
- 4 性質
- 5 歴史
- 6 生活
- 7 人間の特徴と人間論
  - 7.1 人間と遊び
- 8 人間の線引き
  - 8.1 線引き、差別、区別
  - 8.2 様々な基準と概念的な戯れ
- 9 関連語
- 10 脚注
  - 10.1 注

# WikiDataのSPARQLエンドポイント

- SPARQLクエリを利用できるエンドポイント
- SPARQLについては後編で説明



<https://query.wikidata.org/>

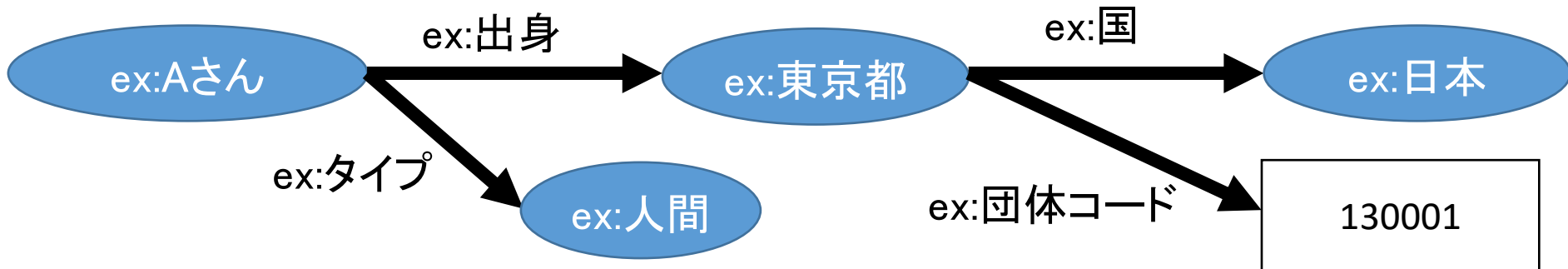
# 参考：WikiDataページを編集

---

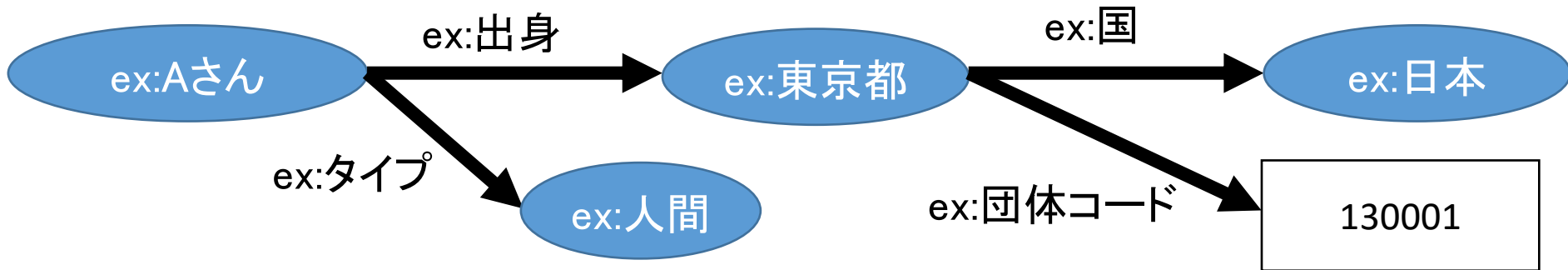
- 無いデータは自分で作れる
- *WikiData*を編集してみよう！
  - <https://www.slideshare.net/KoujiKozaki/wikidata-75383827>
  - 事前にWikiData右上からアカウント作成が必要

# RDFの作成: 題材

- Aさんは人間である
- Aさんは東京都出身
- 東京都の国は日本
- 東京都の団体コードは130001



# テキストエディタでTurtleを書く場合



```
@prefix ex: <http://example.com/terms/> .  
ex:Aさん
```

```
    ex:タイプ ex:人間 ;  
    ex:出身 ex:東京都 .
```

```
ex:東京都
```

```
    ex:国 ex:日本 ;  
    ex:団体コード "130001" .
```

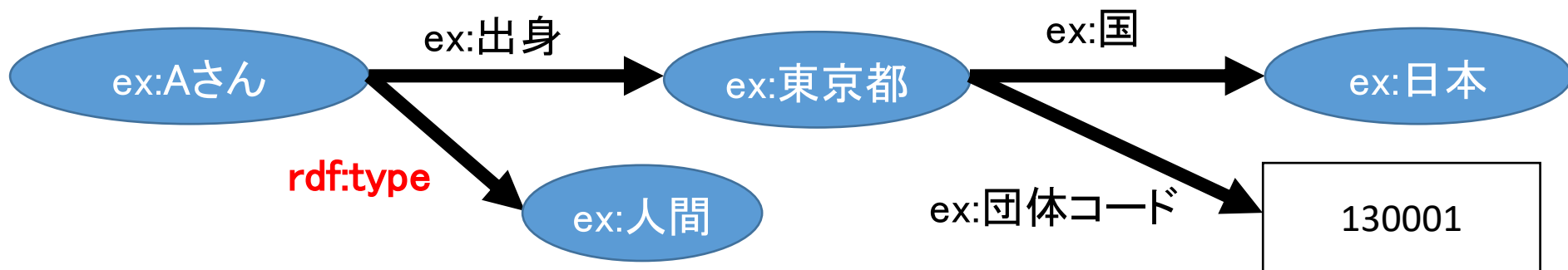
A screenshot of an EmEditor window titled 'C:\Users\s-egami\Downloads\test.ttl - EmEditor'. The window shows the Turtle code entered in the previous block. The file name 'test.ttl' is circled in red in the tab bar. The status bar at the bottom indicates '172 バイト (172 バイト), 8 行。 Text 6行, 10桁 UTF-8 (BOM無し)' and '0 文字 0/8 行'.

このようにTurtleを記述し、テキストファイルとして保存



# よりRDFらしく: 規定の語彙を使用

## クラス-インスタンスの関係をrdf:typeで追加

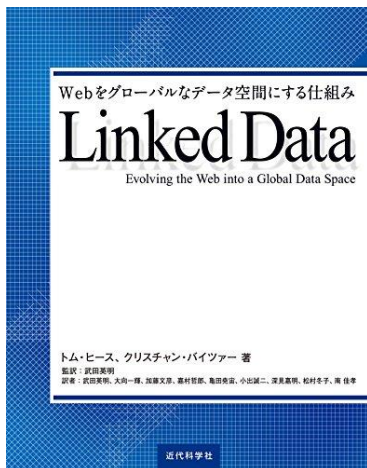


```
@prefix ex: <http://example.com/terms/> .  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
ex:Aさん  
    rdf:type ex:人間 ;  
    ex:出身 ex:東京都 .  
  
ex:東京都  
    ex:国 ex:日本 ;  
    ex:団体コード "130001" .
```

※Turtleでは「rdf:type」を「a」と省略して書くこともできます

# (再掲) Linked Dataの基本原則

1. あらゆる事物にURIを付与すること
2. 誰でも事物の内容が確認できるように、URIはHTTP経由で参照できること
3. URIを参照した時は、標準の技術(RDFやSPARQL等)を使用して関係する有用な情報を利用できるようにすること
4. より多くの事物を発見できるように、他のURIへのリンクを含めること



詳しく理解するには下記の本がおすすめ

**Linked Data: Webをグローバルなデータ空間にする仕組み**

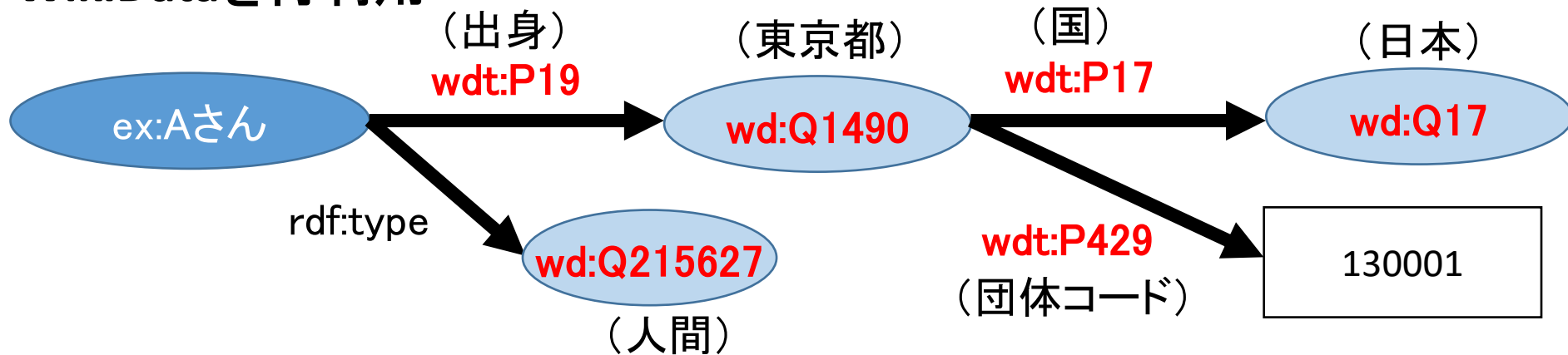
トム・ヒース、クリスチャン・バイツァー 著

監訳：武田英明

訳者：武田英明、大向一輝、加藤文彦、嘉村哲郎、亀田亮宙、小出誠二、深見嘉明、松村冬子、南 佳孝

# よりLODらしく

## WikiDataを再利用



```
@prefix ex: <http://example.com/terms/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix wdt: <http://www.wikidata.org/prop/direct/> .
@prefix wd: <http://www.wikidata.org/entity/> .

ex:Aさん
    rdf:type wd:Q215627 ;
    wdt:P19 wd:Q1490 .

wd:Q1490
    wdt:P17 wd:Q17 ;
    wdt:P429 "130001" .
```

```
C:\Users\%s-egami\Downloads\test.ttl - EmEditor
test.ttl x
1 @prefix ex: <http://example.com/terms/> .
2 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3 @prefix wdt: <http://www.wikidata.org/prop/direct/> .
4 @prefix wd: <http://www.wikidata.org/entity/> .
5 ex:Aさん
6     rdf:type wd:Q215627 ;
7     wdt:P19 wd:Q1490 .
8
9 wd:Q1490
10     wdt:P17 wd:Q17 ;
11     wdt:P429 "130001" .
Text 4行, 48桁 UTF-8 (BOM無し) 0文字 0/11行
```

# WikiDataでどうやって語彙を調べるの？

- WikiDataプロパティの一覧 & 検索

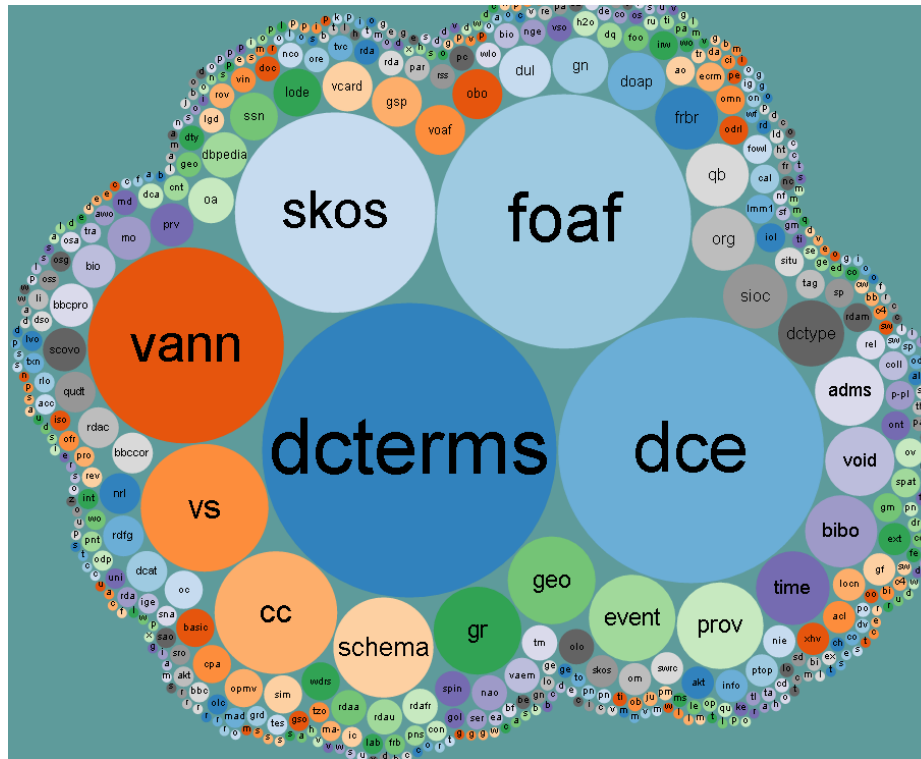
- [https://www.wikidata.org/wiki/Wikidata:List\\_of\\_properties/ja](https://www.wikidata.org/wiki/Wikidata:List_of_properties/ja)

The screenshot shows the Wikidata page for 'List of properties' in Japanese. The page title is 'ウィキデータ:プロパティの一覧'. The sidebar on the left contains links for 'Main page', 'Community portal', 'Project chat', 'Create a new Item', 'Create a new Lexeme', 'Recent changes', 'Random Item', 'Query Service', 'Nearby', 'Help', 'Donate', and 'Tools'. The main content area has a search bar and a list of other languages. Below this, there is a section for 'Other languages' with a list of language links. The main content area also includes a search box and a button 'プロパティを検索'. Below the search box is a grid of property categories: 'art, art collection', 'asymmetric relation', 'author's rights', 'award', 'causality', 'E-commerce', 'identity, equality', 'judiciary', 'link rot', 'MediaWiki page', 'number of entities', 'obsolete Wikidata property', 'occurrence', 'orderable Wikidata property', and 'ownership'. The URL at the bottom is 'https://www.wikidata.org/wiki/Wikidata:List\_of\_properties/fo'.

## その他: Linked Open Vocabularies

## どのような語彙があるのか検索できる

<https://lov.linkeddata.es/dataset/lov/>



# RDFグラフの可視化

- ARC2によるRDFグラフの視覚化
  - Turtle, Microdata, JSON-LD, RDF/XML, TriGに対応
  - By KANZAKI, Masahide
  - <https://www.kanzaki.com/works/2009/pub/graph-draw>
- W3C RDF Validation Service
  - RDF/XMLに対応
  - 日本語を含むトリプルを視覚化したい場合には、「Graph format」を「SVG-embedded」または「SVG-link」とする
  - <http://www.w3.org/RDF/Validator/>

# RDFグラフの可視化

The Web KANZAKI music & knowledge sharing

? Help modified 2020-03-24

## RDFグラフの視覚化 Turtle, Microdata, JSON-LD, RDF/XML, TriG

テキストエリアにTurtle、Microdata、JSON-LD、RDF/XMLもしくはTriGで記述したRDFを入力し、「グラフ描画」ボタンを選択してください。MicrodataはHTML断片でもかまいません。URIをコンパクトに描画したい場合は「URIを修飾名表記する」チェックを有効にしてください。

```
1 @prefix ex: <http://example.com/terms/> .
2 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3 @prefix wdt: <http://www.wikidata.org/prop/direct/> .
4 @prefix wd: <http://www.wikidata.org/entity/> .
5 ex:Aさん
6   rdf:type wd:Q215627 ;
7   wdt:P19 wd:Q1490 .
8
9 wd:Q1490
10  wdt:P17 wd:Q17 ;
11  wdt:P429 "130001" .
12
```

Turtleを貼り付け

入力形式 : ☒ Turtle ☐ Microdata ☐ JSON-LD ☐ RDF/XML ☐ TriG +

出力形式 : ☒ PNG ☐ 小PNG (約1112 x 1890 px以内) ☐ SVG

描画方向 : ☒ 左から右 ☐ 上から下 ☐ 下から上 ☐ 右から左

☒ URIを修飾名表記する ☐ 空白ノードIDを表示する ☐ 型付ノードグラフ (型付ノードグラフとは)

☐ クラスを強調 ☐ Turtle変換のみ (グラフ描画なし) ☐ compact

グラフ描画/Draw



<https://www.kanzaki.com/works/2009/pub/graph-draw>

# フォーマットの変換

- シリアライゼーションフォーマット変換ツールを利用し、各種システムで扱いやすいのデータに変換可能
- EASYRDF Converter
  - Turtle, N-Triples, RDF/XML, RDF/JSON, JSON-LD等に対応
  - <https://www.easyrdf.org/converter>

**EASYRDF** Documentation Examples **Converter** Support Downloads

## Converter

Input Data:

```
@prefix ex: <http://example.com/terms/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix wdt: <http://www.wikidata.org/prop/direct/> .
@prefix wd: <http://www.wikidata.org/entity/> .
ex:Aさん
  rdf:type wd:Q215627 ;
  wdt:P19 wd:Q1490 .
```

or URI:

(This URI is also used as the Base URI, when text is put in the input data box)

Input Format:

Output Format:

☐ Raw output

## Output

Number of triples parsed: 4

```
{
  "http://example.com/terms/Aさん": [
    {
      "http://www.w3.org/1999/02/22-rdf-syntax-ns#type": [
        {
          "type": "uri",
          "value": "http://www.wikidata.org/entity/Q215627"
        }
      ],
      "http://www.wikidata.org/prop/direct/P19": [
        {
          "type": "uri",
          "value": "http://www.wikidata.org/entity/Q1490"
        }
      ]
    }
  ],
  "http://www.wikidata.org/entity/Q1490": [
    {
      "http://www.wikidata.org/prop/direct/P17": [
        {
          "type": "uri",
          "value": "http://www.wikidata.org/entity/Q17"
        }
      ]
    }
  ],
  "http://www.wikidata.org/prop/direct/P429": [
    {
      "type": "literal",
      "value": "130001"
    }
  ]
}
```

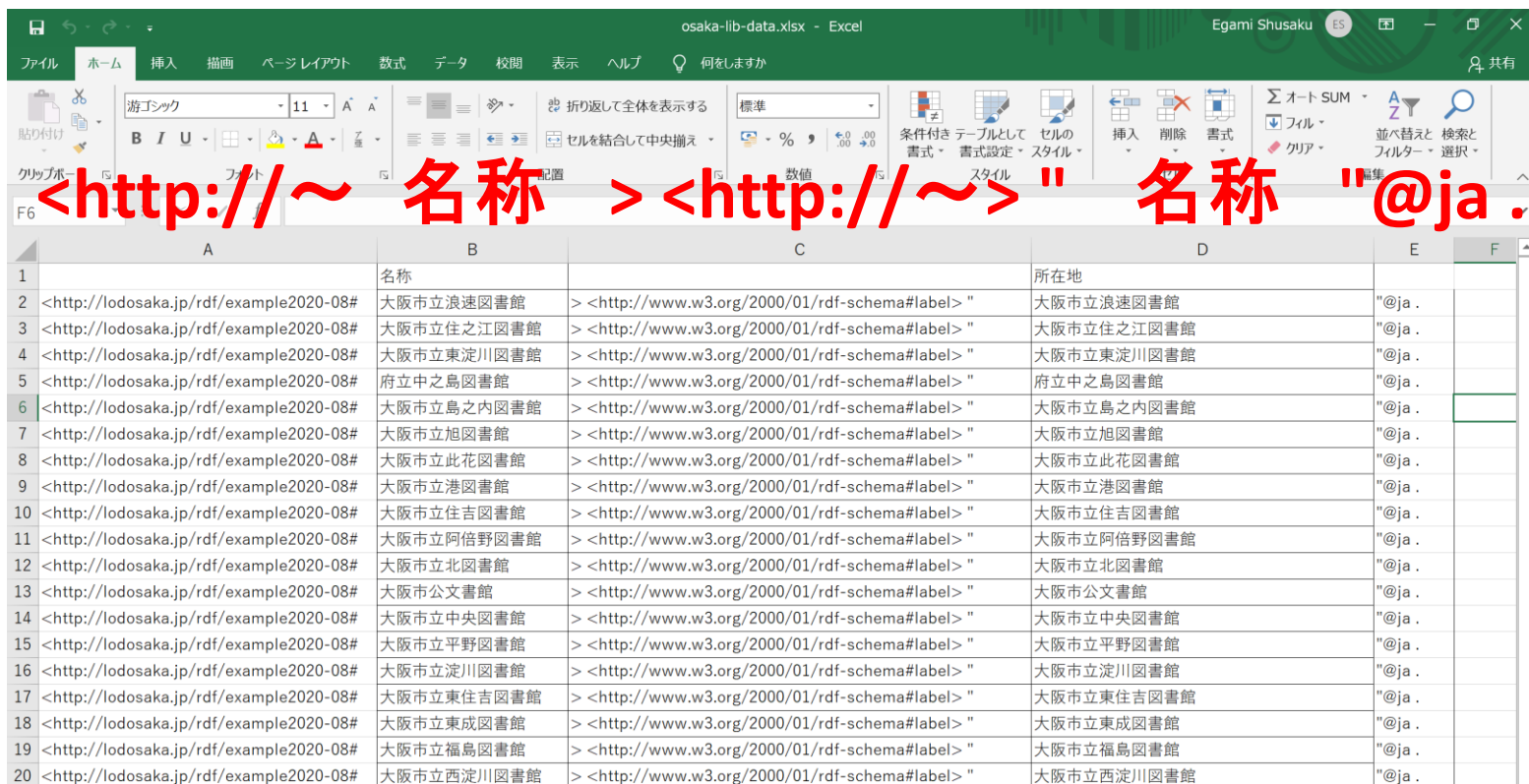
```
{
  - http://example.com/terms/Aさん: [
    - http://www.w3.org/1999/02/22-rdf-syntax-ns#type: [
      - {
        type: "uri",
        value: "http://www.wikidata.org/entity/Q215627"
      }
    ],
    - http://www.wikidata.org/prop/direct/P19: [
      - {
        type: "uri",
        value: "http://www.wikidata.org/entity/Q1490"
      }
    ]
  ],
  - http://www.wikidata.org/entity/Q1490: [
    - http://www.wikidata.org/prop/direct/P17: [
      - {
        type: "uri",
        value: "http://www.wikidata.org/entity/Q17"
      }
    ],
    - http://www.wikidata.org/prop/direct/P429: [
      - {
        type: "literal",
        value: "130001"
      }
    ]
  ]
}
```

TurtleをRDF/JSONに変換した例



# EXCELでN-Triplesを作成する

- テキストエディタにコピー→タブを一括削除→保存



The screenshot shows an Excel spreadsheet with a table of library data. The table has columns for ID, Name, and Location. Red text is overlaid on the first row of data, showing the process of converting the data into N-Triples format. The red text is: `<http://~ 名称 > <http://~> " 名称 "@ja .`

	A	B	C	D	E	F
1		名称		所在地		
2	<http://lodosaka.jp/rdf/example2020-08#	大阪市立浪速図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立浪速図書館	"@ja .	
3	<http://lodosaka.jp/rdf/example2020-08#	大阪市立住之江図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立住之江図書館	"@ja .	
4	<http://lodosaka.jp/rdf/example2020-08#	大阪市立東淀川図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立東淀川図書館	"@ja .	
5	<http://lodosaka.jp/rdf/example2020-08#	府立中之島図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	府立中之島図書館	"@ja .	
6	<http://lodosaka.jp/rdf/example2020-08#	大阪市立島之内図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立島之内図書館	"@ja .	
7	<http://lodosaka.jp/rdf/example2020-08#	大阪市立旭図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立旭図書館	"@ja .	
8	<http://lodosaka.jp/rdf/example2020-08#	大阪市立此花図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立此花図書館	"@ja .	
9	<http://lodosaka.jp/rdf/example2020-08#	大阪市立港図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立港図書館	"@ja .	
10	<http://lodosaka.jp/rdf/example2020-08#	大阪市立住吉図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立住吉図書館	"@ja .	
11	<http://lodosaka.jp/rdf/example2020-08#	大阪市立阿倍野図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立阿倍野図書館	"@ja .	
12	<http://lodosaka.jp/rdf/example2020-08#	大阪市立北図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立北図書館	"@ja .	
13	<http://lodosaka.jp/rdf/example2020-08#	大阪市公文書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市公文書館	"@ja .	
14	<http://lodosaka.jp/rdf/example2020-08#	大阪市立中央図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立中央図書館	"@ja .	
15	<http://lodosaka.jp/rdf/example2020-08#	大阪市立平野図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立平野図書館	"@ja .	
16	<http://lodosaka.jp/rdf/example2020-08#	大阪市立淀川図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立淀川図書館	"@ja .	
17	<http://lodosaka.jp/rdf/example2020-08#	大阪市立東住吉図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立東住吉図書館	"@ja .	
18	<http://lodosaka.jp/rdf/example2020-08#	大阪市立東成図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立東成図書館	"@ja .	
19	<http://lodosaka.jp/rdf/example2020-08#	大阪市立福島図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立福島図書館	"@ja .	
20	<http://lodosaka.jp/rdf/example2020-08#	大阪市立西淀川図書館	> <http://www.w3.org/2000/01/rdf-schema#label> "	大阪市立西淀川図書館	"@ja .	

<https://github.com/koujikozaiki/RDF-FederatedSample>

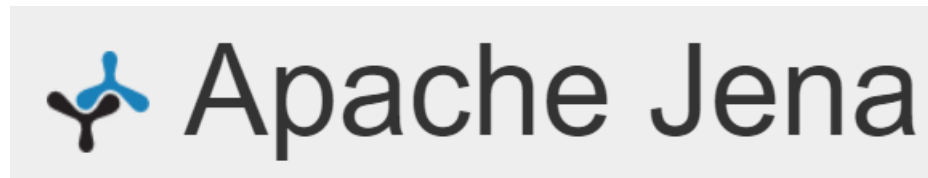
# Open Refine (旧Google Refine)

- スプレッドシートのようなUIでデータの整形やRDFへの変換などが可能
- <https://openrefine.org/>
- 詳しくはこちらをご参照ください
  - 「Open Refine (旧Google Refine) の使い方 ～導入・基本編～」 <http://togotv.dbcls.jp/ja/20130212.html>

# Javaからフレームワークを使用する

## Apache Jena

- Linked DataのためのオープンソースJavaフレームワーク
  - <https://jena.apache.org/>



- 以下の3つの機能を持つ
  - RDFの構築, 検索
  - RDFの格納(トリプルストア)とエンドポイントの立ち上げ
  - オントロジーの利用と推論

# Apache JenaでRDFの作成

Jenaをダウンロードしてパスを通す必要あり

```
1 package lesson;
2
3 import java.io.FileNotFoundException;
4 import java.io.FileOutputStream;
5 import org.apache.jena.rdf.model.*;
6 import org.apache.jena.vocabulary.RDF;
7
8 public class Test {
9     public static void main(String[] args) {
10         //RDFモデルを作成。このmodel内にトリプルが作成されていく
11         Model model = ModelFactory.createDefaultModel();
12         String ex = "http://example.com/";
13         String wd = "http://www.wikidata.org/entity/";
14         String wdt = "http://www.wikidata.org/prop/direct/";
15         //プロパティを作成。引数はURI
16         Property birth_of_place = model.createProperty(wdt + "P19");
17         Property country = model.createProperty(wdt + "P17");
18         Property dantai_code = model.createProperty(wdt + "P429");
19         //リソースを作成。引数はURI
20         Resource a_san = model.createResource(ex + "Aさん");
21         Resource tokyo = model.createResource(wd + "Q1490");
22         Resource person = model.createResource(wd + "Q215627");
```

```
23         Resource japan = model.createResource(wd + "Q17");
24         //リソースにプロパティとその値を追加する (トリプルを作る)
25         //第 1 引数にプロパティ, 第 2 引数に値
26         a_san.addProperty(RDF.type, person);
27         a_san.addProperty(birth_of_place, tokyo);
28         tokyo.addProperty(country, japan);
29         tokyo.addProperty(dantai_code, "130001");
30         //名前空間のPrefixを設定
31         model.setNsPrefix("ex", ex);
32         model.setNsPrefix("wd", wd);
33         model.setNsPrefix("wdt", wdt);
34         //モデルの書き出し。Turtleを指定
35         //モデルの書き出し。
36         try {
37             FileOutputStream fout = new FileOutputStream("test.ttl");
38             model.write(fout, "TTL");
39         } catch (FileNotFoundException e) {
40             e.printStackTrace();
41         }
42     }
43 }
```

- 詳しくはこちらもご参考ください

- 「Javaのプログラムを作成してLODを効率的に処理してみよう」

<https://github.com/KnowledgeGraphJapan/LODws2nd/blob/master/ApacheJena%E3%83%8F%E3%83%B3%E3%82%BA%E3%82%AA%E3%83%B3.pdf>

# その他の言語

---

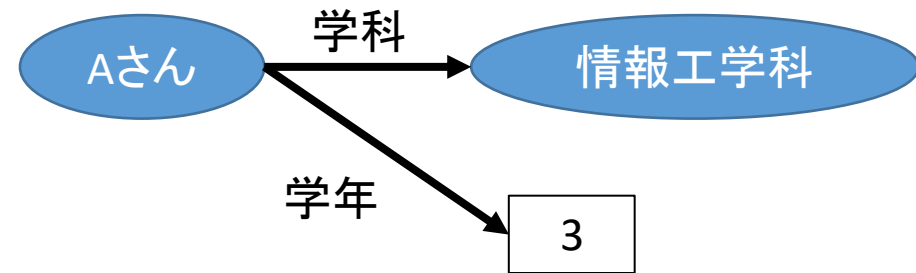
- Python
  - RDFLib: <https://github.com/RDFLib/rdfliib>
- PHP
  - EASYRDF: <https://www.easyrdf.org/>
- JS
  - RDFJS: <https://rdf.js.org/>
- Ruby
  - Ruby RDF: <https://github.com/ruby-rdf>
- C
  - Redland: <http://librdf.org/>

# プログラムでRDFを作成する利点

- ファイルの入出力処理と組み合わせると、既存のデータをRDFに自動変換するプログラムを自在に作成できる。

述語		
	学科	学年
Aさん	情報工学科	3
Bさん	機械工学科	2
Cさん	生命科学科	2

主語                      目的語



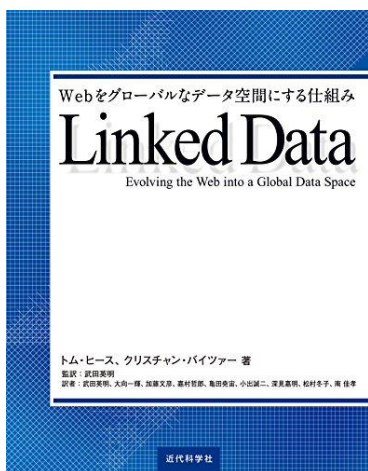
## その他

複雑なデータの変換，インスタンスマッチング，機械学習応用など，将来的に色々やりたいことがある人はプログラムの利用に慣れていると便利です。

RDFの格納・公開→LOD

# (再掲) Linked Dataの基本原則

1. あらゆる事物にURIを付与すること
2. 誰でも事物の内容が確認できるように、URIはHTTP経由で参照できること
3. URIを参照した時は、標準の技術(RDFやSPARQL等)を使用して関係する有用な情報を利用できるようにすること
4. より多くの事物を発見できるように、他のURIへのリンクを含めること



詳しく理解するには下記の本がおすすめ

**Linked Data: Webをグローバルなデータ空間にする仕組み**

トム・ヒース、クリスチャン・バイツァー 著

監訳：武田英明

訳者：武田英明、大向一輝、加藤文彦、嘉村哲郎、亀田亮宙、小出誠二、深見嘉明、松村冬子、南 佳孝

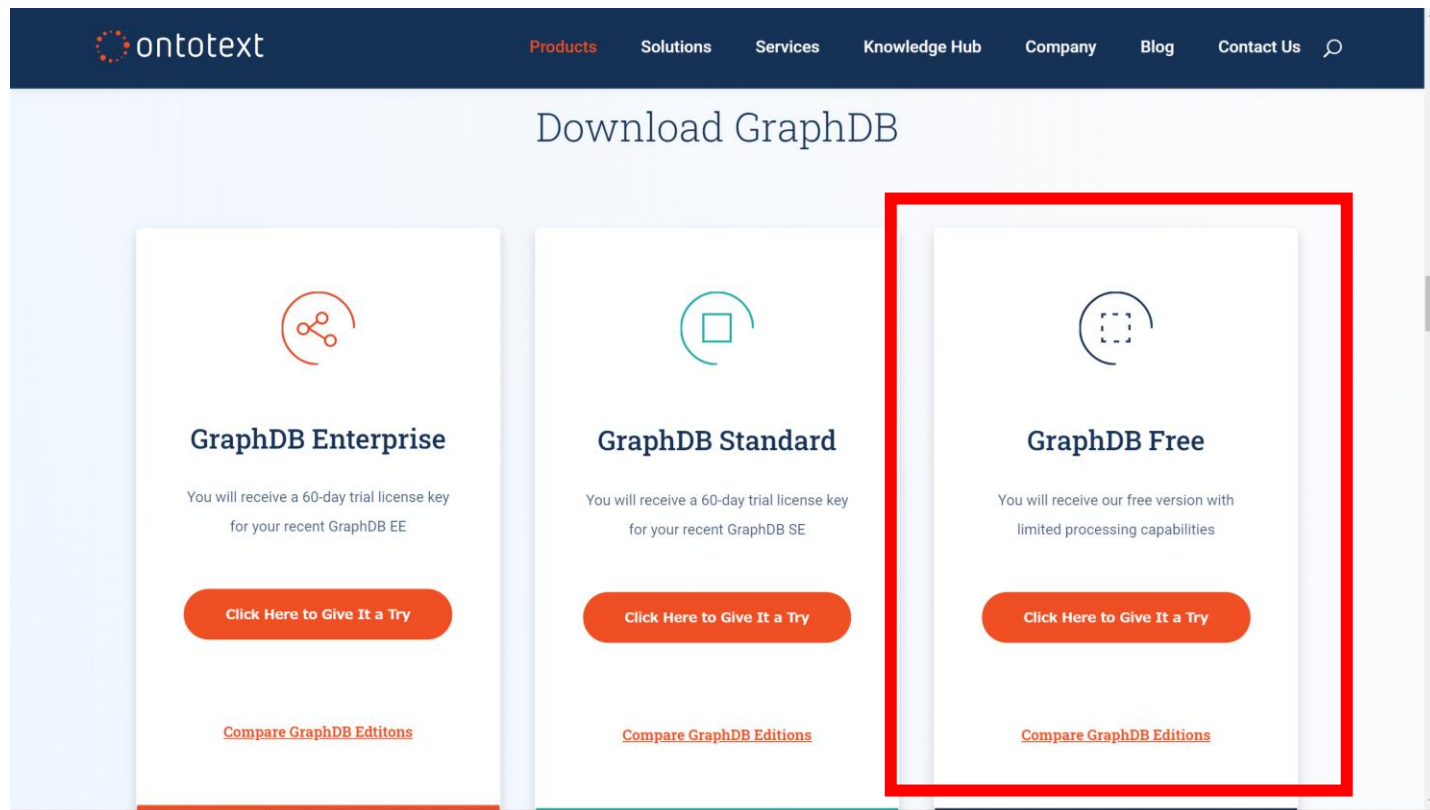


# トリプルストア

- RDFデータを格納し, SPARQLによる検索などを可能にするデータベース
    - RDFストア, RDFデータベースとも呼ばれる
  - オープンソース, 商用ともに様々なトリプルストアが存在
    - AllegroGraph
    - Amazon Neptune
    - Blazegraph
    - Dydra
    - GraphDB by Ontotext
    - MarkLogic
    - OpenLink Virtuoso
    - Oracle
    - Stardog
    - ...
- 詳しくはこちら  
[https://en.wikipedia.org/wiki/Comparison\\_of\\_triplestores](https://en.wikipedia.org/wiki/Comparison_of_triplestores)

# RDFデータをトリプルストアに格納

- GraphDB by Ontotextのフリー版で説明
  - 下記ボタンの遷移先フォームから必要情報を入力して送信するとDLリンクが送られてきます



<https://www.ontotext.com/products/graphdb/>

# リポジトリの作成

- Setup > Repositories > Create Repositories

**GraphDB** FREE

Import  
Explore  
SPARQL  
Monitor  
Setup  
**Repositories**  
Users and Access  
My Settings  
Connectors  
Namespaces  
Autocomplete  
RDF Rank  
Help

## Create Repository

Repository properties

Repository ID\* LOD-WS-2020

Repository title LOD-WS-2020

Type GRAPHDB-FREE

Storage folder storage

Ruleset RDFS-Plus (Optimized) [Upload custom ruleset](#)

☒ Disable owl:sameAs

☐ Supports SHACL validation

Base URL http://example.org/owlim#

Entity index size 10000000

☒ Use predicate indices ☒ Cache literal language tags

☐ Use context index ☒ Enable literal index

☐ Check for inconsistencies ☐ Throw exception on query time-out

☐ Read-only

Entity ID bit-size 32

# インポート

- Import > RDF > Upload RDF Files > ファイル選択 > Import

The screenshot displays the GraphDB web interface. On the left is a sidebar with navigation options: Import, RDF (highlighted), Tabular (OntoRefine), Explore, SPARQL, Monitor, Setup, and Help. The main area is titled 'Import' and contains three tabs: 'User data' and 'Server files'. Below these tabs are three import methods: 'Upload RDF files' (All RDF formats, up to 200 MB), 'Get RDF data from a URL' (All RDF formats), and 'Import RDF text snippet' (Type or paste RDF data). A list of imports is shown below, with a table containing one entry: 'test.ttl', which was imported successfully in less than a second. The table has columns for selection, status, name, and actions. The 'Import' button is visible in the bottom right corner of the table.

GraphDB FREE

Import

RDF

Tabular (OntoRefine)

Explore

SPARQL

Monitor

Setup

Help

Import

User data Server files

Help

Upload RDF files  
All RDF formats, up to 200 MB

Get RDF data from a URL  
All RDF formats

Import RDF text snippet  
Type or paste RDF data

Import

test.ttl

Imported successfully in less than a second.

Import

# データの確認

- Explore > Graph overview > The default graph

GraphDB FREE

Import

Explore

Graphs overview

Class hierarchy

Class relationships

Visual graph

Similarity

SPARQL

Monitor

Setup

Help

nil

Source: <http://www.openrdf.org/schema/sesame#nil>

subject predicate object context all Explicit only Show Blank Nodes Download as Visual graph

	subject	predicate	object	context
1	ex:Aさん	rdf:type	wd:Q215627	<a href="http://www.ontotext.com/explicit">http://www.ontotext.com/explicit</a>
2	ex:Aさん	wdt:P19	wd:Q1490	<a href="http://www.ontotext.com/explicit">http://www.ontotext.com/explicit</a>
3	wd:Q1490	wdt:P17	wd:Q17	<a href="http://www.ontotext.com/explicit">http://www.ontotext.com/explicit</a>
4	wd:Q1490	wdt:P429		

Visual graph

Aさん → P19 → Q1490 → P17 → Q17

可視化も

# SPARQL検索

- 検索が可能 (SPARQLは後編で説明)

GraphDB

SPARQL Query & Update

```
select * where {  
  ?s ?p ?o .  
} limit 100
```

Run

Table Raw Response Pivot Table Google Chart

Filter query results

Showing results from 1 to 80 of 80. Query took 0.2s, moments ago.

	s	p
1	rdf:type	rdf:type

keyboard shortcuts

# LOD公開ツール

---

- トリプルストアのエンドポイントを外向けに公開設定するのはハードルが高いという人向けのツール
- Simple LODI
  - <https://github.com/uedayou/simplelodi>
- LOD Smart Index
  - <https://www.mirko.jp/LODSI/>

# とりあえず公開するだけなら

- **DropBox, Google Drive, OneDriveなどで公開**
  - 厳密にLinked Dataの基本原則を満たすことはできないが、データを公開して共有することはできる
- **GitHubでRDFデータを公開**
  - こちらも厳密にLinked Dataの基本原則を満たすことはできないが、データを公開して共有することはできる
  - 複数人での管理や変更履歴を残したい場合に便利



# 今回説明しなかった関連技術

時間の都合上説明できなかったもので、より詳しく知りたい方は下記の資料をご参考にされると良いと思います

- RDFの細かな部分
  - RDF入門: <http://www.asahi-net.or.jp/~ax2s-kmttn/internet/rdf/rdf-primer.html>
  - RDF 1.1入門: <http://www.asahi-net.or.jp/~ax2s-kmttn/internet/rdf/NOTE-rdf11-primer-20140225.html>
- RDFスキーマ
  - RDFスキーマ -- リソース表現の語彙定義: <https://kanzaki.com/docs/sw/rdf-schema.html>
- SPARQL
  - 本勉強会の後編で
- オントロジー
  - オントロジーとは?: <https://www.slideshare.net/KoujiKozaki/ss-124638776>
- OWL
  - ウェブ・オントロジー言語OWL: <https://kanzaki.com/docs/sw/webont-owl.html>
- 記述論理
  - セマンティックWebと記述論理: <http://www.sw.cei.uec.ac.jp/kaneiwa/SemWeb.pdf>
- 推論
  - Reasoning: <http://graphdb.ontotext.com/documentation/standard/reasoning.html>

# その他参考資料

- 古崎晃司, つながったデータの作り方, 情報の科学と技術「特集: RDFとSPARQL～検索とデータ可視化」, 70(8), pp.406-412 (2020)
  - [https://www.jstage.jst.go.jp/article/jkg/70/8/70\\_406/\\_article/-char/ja/](https://www.jstage.jst.go.jp/article/jkg/70/8/70_406/_article/-char/ja/)
- 古崎晃司, ウェブの情報資源活用のための技術: ナレッジグラフとしてのLOD活用, 情報の科学と技術「特集: ウェブを基盤とした社会」, 70(6), pp.303-308 (2020)
  - [https://www.jstage.jst.go.jp/article/jkg/70/6/70\\_303/\\_article/-char/ja/](https://www.jstage.jst.go.jp/article/jkg/70/6/70_303/_article/-char/ja/)