CFC VIRTUOSO ハンズオン

2017年8月5日 @稲毛海岸

新田 清 <knitta@acm.org>

タイムテーブル

時間 (分)	項目	種類
0-15	オープンデータの共有と利用アプローチ	プレゼン
15-45	Virtuosoのインストールとサンプルデータのロード	ハンズオン
45-90	SPARQLの紹介	プレゼン
90-120	データ検索	ハンズオン

オープンデータの共有と利用アプローチ

オープンデータの共有形式

A) PDF

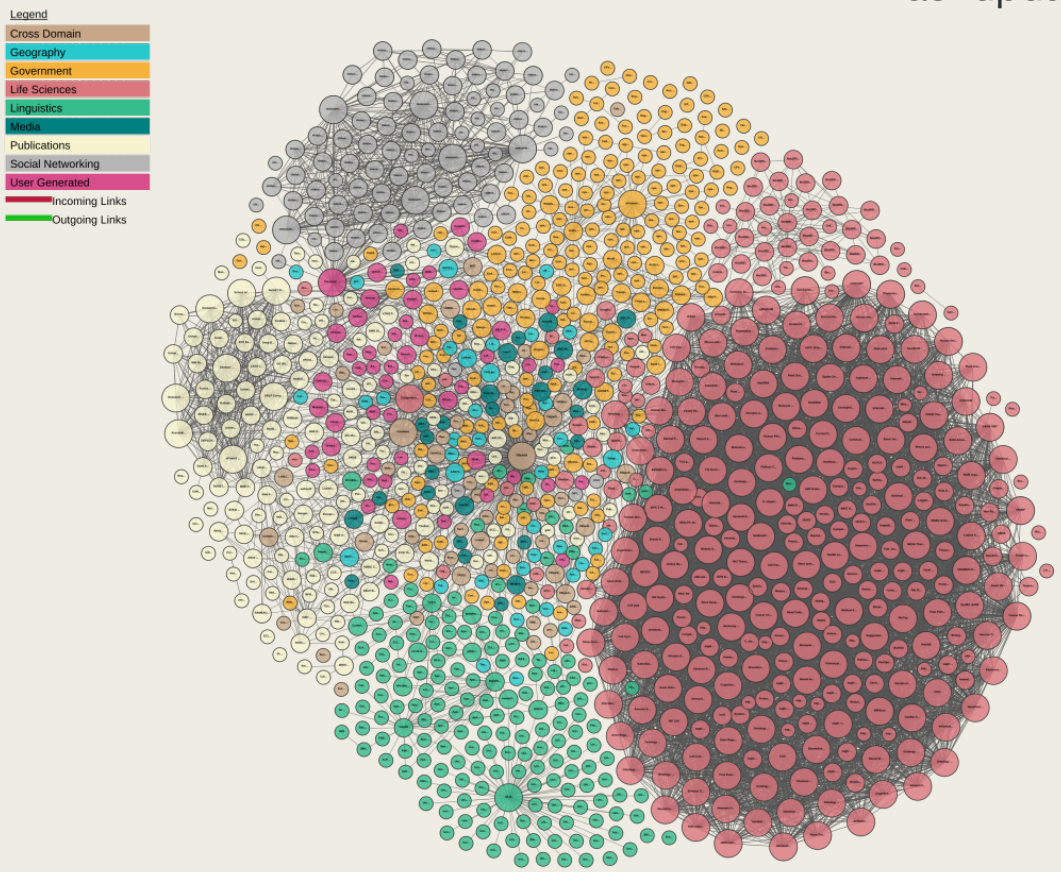
B) Excel, Word

C) Linked Open Data (LOD)

- オープンライセンス
- Resource Description Framework (RDF)
- LinkedDataルール
  - あらゆるデータの識別子としてURIを使用する。
  - 識別子にはHTTP URIを使用し、参照やアクセスを可能にする。
  - URIにアクセスされた際には有用な情報を標準的なフォーマットで提供する。
  - データには他の情報源における関連情報へのリンクを含め、ウェブ上の情報発見を支援する。

LODクラウド

Last updated: 2017-02-20



Linking Open Data cloud diagram 2017, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak. http://lod-cloud.net/

PDF, Excel, Word形式の課題

- 個々のデータの取り出しに苦労する (PDF, Word)
  - *形式の異なるデータ毎に抽出処理を行う必要がある*
  - *手作業で抽出する場合、データ量に応じたコストが必要*
  - *抽出処理系を開発する場合、データ形式の種類に応じたコストが必要*
- 複数のサイトのデータを組み合わせて利用する際に、各サイトのデータの各項目の指し示す意味の特定に苦労する (PDF, Excel, Word)
  - *データを説明する各文書にアクセスし意味を確認する必要がある*

LOD/RDF形式の利点

- 個々のデータがあらかじめ切り出されている
  - *切り出しコストが発生しない*
  - *各データにURIが割り振られ、グローバルに外部から参照可能*
- 各データにhttpアクセスすることでその意味に関する可読情報を入手できる
- 各データが他サイトのデータとどのような関係（同一、抽象、具体、、、）にあるか明示されている
  - *周囲のデータとの関係からも意味の特定が容易になる*
  - *容易に複数サイトのデータを機械的にまとめて利用できる*

LODに関する動き

- データカタログサイト (http://www.data.go.jp/)
  - *日本政府が自治体公開データのポータルとして公開したもの*
  - *データそのものは多い*
- 共通語彙基盤 (https://imi.go.jp/goi/)
  - *公開データの横断活用を効率化する*
  - *グローバルなLODではメジャーなサイトの語彙が自然に選択されるに任されているが、日本語データのLODコミュニティはまだ立ち上げ期なのであると便利*

RDFデータ利用のアプローチ

I. リレーショナルDBに入れて利用
- *必要なデータセットの数が固定で少数の場合*
- *高い処理効率*
- *SQL等の一般的なスキル*

II. 専用のDBに入れて利用
- *必要なデータセットが不明確、多数、動的変更ありの場合*
- *柔軟なデータ表現*
- *トリプルストアへの格納*
- *SPARQLスキルの習得*

Virtuoso Open Source

- トリプルストアのひとつ
- OpenLink Software製で商用製品もあり処理効率やSPARQLサポート率の面で完成度が高い
- ライセンスはGNU General Public License (GPL) Version 2
- 小規模な計算機からでも稼働させることが可能
- ドキュメントやチュートリアルがそれなりにある
- 発音が難しい (^^;

本ハンズオンの目的

- データ提供される方
  - *LOD/RDF形式でオープンデータを公開する長所を理解する*
  - *SPARQLエンドポイントを立ち上げデータ提供する*
  - *提供データが実際にどのように利用されるのかを体験する*

- データ利用される方
  - *LOD/RDF形式データの長所を理解する*
  - *手間をかけずにLOD/RDFデータを利用する*
  - *実際にデータにふれながら、目的に合ったオープンデータを絞り込む*

リンク

- Resource Description Framework (RDF)
  https://www.w3.org/2001/sw/wiki/RDF
- LinkedData (ルール)
  https://www.w3.org/wiki/LinkedData
- LODクラウド
  http://lod-cloud.net/
- 日本政府データカタログサイト
  http://www.data.go.jp/
- 共通語彙基盤
  https://imi.go.jp/goi/
- Virtuoso Open Source
  http://vos.openlinksw.com/owiki/wiki/VOS/

VIRTUOSOの
インストールと
サンプルデータのロード

(ハンズオン)

AWS設定

- キーペア作成
  - *安全な場所に保管*
- セキュリティグループ作成
  - *sshとhttpを外部からアクセス許可するルールを追加*
- EC2 インスタンスを作成

AMI	Amazon Linux AMI 2017.03.1 (HVM), SSD Volume Type - ami-3bd3c45c
インスタンスタイプ	t2.micro
インスタンス設定	全てデフォルト
ストレージ要領	20GB
タグ	なし
セキュリティグループ	上記作成したSGを選択

Virtuoso設定 (1)

- ソースコードをダウンロード (所用時間: 約3-4分)

```
$ mkdir wrk
$ cd wrk
$ wget https://github.com/openlink/virtuoso-opensource/releases/download/v7.2.4.2/virtuoso-opensource-7.2.4.2.tar.gz
$ tar zxvf virtuoso-opensource-7.2.4.2.tar.gz cd virtuoso-opensource-7.2.4.2
```

- ビルドのために以下のツールをインストール

```
$ sudo yum-config-manager --enable epel
$ sudo yum install gcc gmake autoconf automake libtool flex bison gperf gawk m4 make openssl-devel readline-devel ¥
wget git p7zip
```

Virtuoso設定 (2)

- ビルドとインストール (所用時間: 約10分)

```
$./configure --prefix=/usr/local/ -with-readline
$ make                  # 8-9m
$ sudo make install # 4s
```

- Virtuosoサーバを起動

```
$ cd /usr/local/var/lib/virtuoso/db/
$ sudo chown -R ec2-user .
$ virtuoso-t –df &
```

データのロード (1)

- ハンズオン用githubからサンプルデータをダウンロード

```
$ cd /home/ec2-user/wrk
$ git clone https://github.com/zambendorf/cfc2017aug.git
```

- Virtuosoで読めるようにリンクを設定

```
$ cd /usr/local/var/lib/virtuoso/db
$ sudo ln -s /home/ec2-user/wrk/cfc2017aug/dat
```

- bulk loaderの機能を設定

```
$ isql localhost:1111 dba dba
SQL>LOAD 'src/rdfloader.sql';
```

データのロード (2)

- isqlからデータ読み取り指示を出す

```
SQL> ld_dir ('dat', 'yagoFacts-h200k.ttl', 'http://ygfact.cfc2017.test#');
SQL> ld_dir ('dat', 'yagoTaxonomy.ttl', 'http://ygtaxo.cfc2017.test#');
SQL> rdf_loader_run ();
```

- isqlで確認

```
SQL> SPARQL SELECT * FROM <http://ygfact.cfc2017.test#> WHERE { ?s ?p ?o };
SQL> SPARQL SELECT * FROM <http://ygtaxo.cfc2017.test#> WHERE { ?s ?p ?o };
```

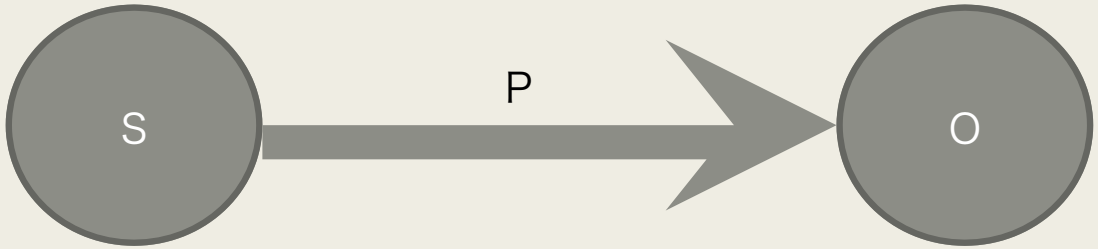
SPARQLの紹介

SPARQL概要

- RDF用クエリ言語でW3C勧告で仕様が定義されており、様々な実装がある
- SQLに類似した構文で問い合わせを行うが、基本要素がRDFトリプルとなる
- 変数を含む複数のトリプルからなるグラフパターンがクエリの骨格である
- RDFデータの実態に合わせた仕様として、OPTIONAL構文が用意されている
- SPARQL1.1で追加された仕様として、プロパティーパスがある
- SQLと同様なサブクエリ構文が使える
- データセットによりデータの分割管理が可能
- SQLと同様なソリューション・シーケンスやクエリ形式が使える

RDFトリプルとトリプルパターン

- 主語(Subject)、述語(Predicate)、目的語(Object)からなるグラフモデル
- 3つの要素から成り立っているのでトリプルと呼ぶ
- データにはIRI (I)、リテラル (L)、ブランク (B)の種類があり、SはIかB、PはI、OはIかBかL、の値をとることができる
- ブランクは'_:a'のように'_:'の接頭辞をつけて表す
- トリプルパターンはS, P, Oの少なくともひとつが変数であるもの
- 変数は'?name'のように'?'の接頭辞をつけて表す



シンプルなクエリ

- データ
  <http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial" .
- クエリ
  SELECT ?title WHERE
  { <http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> ?title . }
- クエリ結果

title
"SPARQL Tutorial"

複数マッチ

- データ

```
_:a   foaf:name    "Johnny Lee Outlaw" .
_:a   foaf:mbox    <mailto:jlow@example.com> .
_:b   foaf:name    "Peter Goodguy" .
_:b   foaf:mbox    <mailto:peter@example.org> .
_:c   foaf:mbox    <mailto:carol@example.org> .
```

- クエリ

```
SELECT ?name ?mbox
WHERE  { ?x foaf:name ?name .
         ?x foaf:mbox ?mbox }
```

- クエリ結果

name	mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

RDFグラフの構築

- データ

```
_:a   org:employeeName    "Alice" .
_:a   org:employeeId      12345 .
_:b   org:employeeName    "Bob" .
_:b   org:employeeId      67890 .
```

- クエリ

```
CONSTRUCT { ?x foaf:name ?name }
WHERE   { ?x org:employeeName ?name }
```

- クエリ結果

```
 _:x foaf:name "Alice" .
 _:y foaf:name "Bob" .
```

オプションのパターン・マッチング

- データ

```
_:a   rdf:type          foaf:Person .
_:a   foaf:name          "Alice" .
_:a   foaf:mbox          <mailto:alice@example.com> .
_:a   foaf:mbox          <mailto:alice@work.example> .
_:b   rdf:type          foaf:Person .
_:b   foaf:name          "Bob" .
```

- クエリ

```
SELECT  ?name  ?mbox
WHERE { ?x foaf:name   ?name .
         OPTIONAL { ?x   foaf:mbox   ?mbox }
        }
```

- クエリ結果

name	mbox
"Alice"	<mailto:alice@example.com>
"Alice"	<mailto:alice@example.com>
"Bob"	

代替のマッチング

- データ

```
_:a   dc10:title      "SPARQL Query Language Tutorial" .
_:a   dc10:creator    "Alice" .
_:b   dc11:title      "SPARQL Protocol Tutorial" .
_:b   dc11:creator    "Bob" .
_:c   dc10:title      "SPARQL" .
_:c   dc11:title      "SPARQL (updated)" .
```

- クエリ

```
SELECT ?title
WHERE  { { ?book dc10:title  ?title } UNION
         { ?book dc11:title  ?title } }
```

- クエリ結果

title
"SPARQL Query Language Tutorial"
"SPARQL Protocol Tutorial"
"SPARQL"
"SPARQL (updated)"

プロパティー・パス

- 1リンク
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows/foaf:name ?name .
- 2リンク
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows/foaf:knows/foaf:name ?name .
- 逆プロパティー・パス
  <mailto:alice@example> ^foaf:mbox ?x
- 逆パス・シーケンス
  ?x foaf:knows/^foaf:knows ?y .
  FILTER(?x != ?y)
- 任意の長さのマッチ
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows+/foaf:name ?name .
- 任意の長さのパス・マッチ
  <http://example/thing> rdf:type/rdfs:subClassOf* ?type
- 否定のプロパティー・パス
  ?x !(rdf:type|^rdf:type) ?y

集約

- 集約の種類: COUNT、SUM、MIN、MAX、AVG、GROUP_CONCATとSAMPLE
- データ

```
:org1 :affiliates :auth1, :auth2 .
:auth1 :writesBook :book1, :book2 .
:book1 :price 9 .
:book2 :price 5 .
:auth2 :writesBook :book3 .
:book3 :price 7 .
:org2 :affiliates :auth3 .
:auth3 :writesBook :book4 .
:book4 :price 7 .
```

- クエリ

```
SELECT (SUM(?lprice) AS ?totalPrice)
WHERE {
  ?org :affiliates ?auth .
  ?auth :writesBook ?book .
  ?book :price ?lprice .}
GROUP BY ?org
HAVING (SUM(?lprice) > 10)
```

- クエリ結果

totalPrice
21

サブクエリ

- データ

```
:alice :name "Alice", "Alice Foo", "A. Foo" .
:alice :knows :bob, :carol .
:bob :name "Bob", "Bob Bar", "B. Bar" .
:carol :name "Carol", "Carol Baz", "C. Baz" .
```

- クエリ

```
SELECT ?y ?minName
WHERE {
  :alice :knows ?y .
  { SELECT ?y (MIN(?name) AS ?minName)
    WHERE { ?y :name ?name . }
    GROUP BY ?y
  }
}
```

- クエリ結果

y	minName
:bob	"B. Bar"
:carol	"C. Baz"

RDFデータセット

- データ

```
g:graph1 dc:publisher "Bob" .
g:graph1 dc:date "2004-12-06"^^xsd:date .
g:graph2 dc:publisher "Bob" .
g:graph2 dc:date "2005-01-10"^^xsd:date .

_:a foaf:name "Alice" .
_:a foaf:mbox <mailto:alice@work.example> .
_:b foaf:name "Bob" .
_:b foaf:mbox <mailto:bob@oldcorp.example.org> .

_:a foaf:name "Alice" .
_:a foaf:mbox <mailto:alice@work.example> .
_:b foaf:name "Bob" .
_:b foaf:mbox <mailto:bob@newcorp.example.org> .
```

- クエリ

```
SELECT ?name ?mbox ?date
WHERE {
  ?g dc:publisher ?name ;
     dc:date ?date .
  GRAPH ?g
   { ?person foaf:name ?name ; foaf:mbox ?mbox }
}
```

- クエリ結果

name	mbox	date
"Bob"	<mailto:bob@oldcorp.example.org>	"2004-12-06"^^xsd:date
"Bob"	<mailto:bob@newcorp.example.org>	"2005-01-10"^^xsd:date

ソリューション・シーケンス

修飾子	説明
order	ソリューションを順序付け
projection	ある変数を選択
distinct	確実にシーケンス中のソリューションをユニーク
reduced	Distinctではないソリューションの排除を許可
offset	全体のソリューションのシーケンス中のどこからソリューションが始まるかを制御
limit	ソリューションの数を制限

クエリ形式

クエリ形式	説明
SELECT	クエリ・パターンにバインドされた変数の、すべてまたはサブセットを返す
CONSTRUCT	1組のトリプル・テンプレートに変数を代入して構築したRDFグラフを返す
ASK	クエリ・パターンがマッチするかどうかを示すブール値を返す
DESCRIBE	発見した資源に関して記述したRDFグラフを返す

リンク

- SPARQL
  https://www.w3.org/TR/sparql11-overview/
- SPARQLクエリ言語 (日本語訳)
  http://www.asahi-net.or.jp/~ax2s-kmtn/internet/rdf/REC-sparql11-query-20130321.html

データ検索

(ハンズオン)

シンプルなクエリ

- クエリ

SPARQL

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT * FROM <http://ygfact.cfc2017.test#>
WHERE { :Konami ?p ?o };
```

- クエリ結果

p	o
created	Nagano_Winter_Olympics_'98
created	Frogger_Beyond
created	Konami_80's_Arcade_Gallery
created	Ring_of_Red
created	Disney_Sports_Football
created	Disney_Sports_Soccer
created	Mikie
created	TwinBee_Yahho!

複数マッチ

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT * FROM <http://ygfact.cfc2017.test#>
WHERE { :Bolo_Airfield ?p1 ?o . `iri('http://yago-knowledge.org/resource/Glory_to_the_Filmmaker!')` ?p2 ?o };
```

- クエリ結果

p1	o	p2
isLocatedIn	Japan	isLocatedIn

RDFグラフの構築

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
CONSTRUCT {?s rdf:type <http://ygfact.cfc2017.test#/JapaneseThings>}
FROM <http://ygfact.cfc2017.test#> WHERE { ?s ?p :Japan };
```

- クエリ結果

S	P	O
Aso_Mining_forced_labor_controversy	rdf:type	JapaneseThings
152_(film)	rdf:type	JapaneseThings
Air_Next	rdf:type	JapaneseThings
Andy_Ologun	rdf:type	JapaneseThings
Art_Tower_Mito	rdf:type	JapaneseThings

オプションのパターン・マッチング

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT DISTINCT ?s, ?o2 FROM <http://ygfact.cfc2017.test#>
WHERE { ?s :participatedIn ?o1 OPTIONAL {?s :owns ?o2} } limit 5;
```

- クエリ結果

s	o2
Israel	Maccabi_Kiryat_Gat_F.C.
United_States	Chicago_Spurs
United_States	Global_Memory_Net
Kingdom_of_Saxony	NULL
Kingdom_of_Saxony	NULL

代替のマッチング

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT * FROM <http://ygfact.cfc2017.test#>
WHERE {{:Google ?p ?o1} UNION {:Microsoft ?p ?o2}} LIMIT 5;
```

- クエリ結果

p	o1	o2
owns	Google_Search	NULL
owns	Google_Maps	NULL
owns	Google_Mars	NULL
owns	Pyra_Labs	NULL
created	NULL	Fine_Artist

集約

- クエリ

```
SPARQL SELECT ?p, COUNT(*) AS ?c
FROM <http://ygfact.cfc2017.test#>
WHERE { ?s ?p ?o }
GROUP BY ?p ORDER BY DESC(?c) LIMIT 5;
```

- クエリ結果

o	c
isLocatedIn	32463
hasGender	19691
isAffiliatedTo	10434
playsFor	9265
wasBornIn	5030

プロパティー・パス

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?o, COUNT(*) AS ?c FROM <http://ygtaxo.cfc2017.test#>
WHERE { ?s rdfs:subClassOf :wordnet_person_100007846 . ?s rdfs:subClassOf* ?o }
GROUP BY ?o ORDER BY DESC(?c) LIMIT 5;
```

- クエリ結果

o	c
owl#Thing	134952
wordnet_physical_entity_100001930	89968
wordnet_person_100007846	44984
yagoLegalActor	44984
yagoLegalActorGeo	44984

サブクエリ

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/> SELECT ?o, COUNT(*) FROM <http://ygfact.cfc2017.test#>
WHERE { ?s2 :wasBornIn ?o .
  {SELECT ?o, COUNT(*) AS ?c FROM <http://ygfact.cfc2017.test#>
   WHERE { ?s :livesIn ?o } GROUP BY ?o ORDER BY DESC(?c) LIMIT 14}
} GROUP BY ?o;
```

- クエリ結果

o	c
London	91
Toronto	33
Los_Angeles	47
New_York_City	89
Mumbai	5

RDFデータセット

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?src, ?p, COUNT(*) AS ?c
FROM NAMED <http://ygfact.cfc2017.test#>
FROM NAMED <http://ygtaxo.cfc2017.test#>
WHERE {GRAPH ?src { ?s ?p ?o}} GROUP BY ?src ?p ORDER BY DESC(?c) LIMIT 5 ;
```

- クエリ結果

src	p	c
http://ygtaxo.cfc2017.test#	rdfs:subClassOf	570459
http://ygfact.cfc2017.test#	isLocatedIn	32463
http://ygfact.cfc2017.test#	hasGender	19691
http://ygfact.cfc2017.test#	isAffiliatedTo	10434
http://ygfact.cfc2017.test#	playsFor	9265

ソリューション・シーケンス
(GROUP BY)

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT ?src, ?p, COUNT(*) AS ?c
FROM NAMED <http://ygfact.cfc2017.test#>
FROM NAMED <http://ygtaxo.cfc2017.test#>          2031
WHERE {GRAPH ?src { ?s ?p ?o}} GROUP BY ?src ?p ORDER BY DESC(?c) LIMIT 5 OFFSET 5;
```

- クエリ結果

src	p	c
http://ygtaxo.cfc2017.test#	wasBornIn	5030
http://ygfact.cfc2017.test#	created	4213
http://ygfact.cfc2017.test#	hasWebsite	3616
http://ygfact.cfc2017.test#	actedIn	2311
http://ygfact.cfc2017.test#	hasWonPrize	2031

ソリューション・シーケンス (DISTINCT)

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
SELECT DISTINCT ?s, ?o2 FROM <http://ygfact.cfc2017.test#>
WHERE { ?s :participatedIn ?o1 OPTIONAL {?s :owns ?o2} } limit 5;
```

- クエリ結果

s	o2
Israel	Maccabi_Kiryat_Gat_F.C.
United_States	Chicago_Spurs
United_States	Global_Memory_Net
Kingdom_of_Saxony	NULL
Kingdom_of_Saxony	NULL

クエリ形式
(SELECT)

- クエリ

SPARQL

```
PREFIX : <http://yago-knowledge.org/resource/>
SELECT * FROM <http://ygfact.cfc2017.test#>
WHERE { :Konami ?p ?o };
```

- クエリ結果

p	o
created	Nagano_Winter_Olympics_'98
created	Frogger_Beyond
created	Konami_80's_Arcade_Gallery
created	Ring_of_Red
created	Disney_Sports_Football
created	Disney_Sports_Soccer
created	Mikie
created	TwinBee_Yahho!

クエリ形式
(CONSTRUCT)

- クエリ

```
SPARQL PREFIX : <http://yago-knowledge.org/resource/>
CONSTRUCT {?s rdf:type <http://ygfact.cfc2017.test#/JapaneseThings>}
FROM <http://ygfact.cfc2017.test#> WHERE { ?s ?p :Japan };
```

- クエリ結果

S	P	O
Aso_Mining_forced_labor_controversy	rdf:type	JapaneseThings
152_(film)	rdf:type	JapaneseThings
Air_Next	rdf:type	JapaneseThings
Andy_Ologun	rdf:type	JapaneseThings
Art_Tower_Mito	rdf:type	JapaneseThings

クエリ形式
(ASK)

- クエリ
  SPARQL PREFIX : <http://yago-knowledge.org/resource/>
  ASK FROM <http://ygfact.cfc2017.test#> { ?s1 ?p1 :United_States . :United_States ?p2 ?o2 };
- クエリ結果
  1
- クエリ
  SPARQL PREFIX : <http://yago-knowledge.org/resource/>
  ASK FROM <http://ygfact.cfc2017.test#> { ?s1 ?p1 :United_States . :United_States ?p1 ?o2 };
- クエリ結果
  0

課題

- (Q-1) データセットの中で最も多くのプレディケイトを持つサブジェクトは何か？
- (Q-2) そのサブジェクトで最も多くのオブジェクトを持つプレディケイトは何か？
- (Q-3) そのプレディケイトが接続する任意のサブジェクト間で、そのプレディケイト以外のプレディケイトはあるか？
- (Q-4) そのようなプレディケイトで最もトリプル数の多いものは何か？
- (Q-5) ygfactとygtaxoをつなぐグラフはあるか？

EOP