SPARQLを利用したI-Scover DBの分析・処理の演習（中級）

富士通研究所　西野文人

文献キーワード

URIで求めるだけならば簡単

```
SELECT  ?kw (count(?kw) AS ?count) WHERE {
    [] a iscover:Article;
          dcterms:subject ?kw
} GROUP BY ?kw ORDER BY DESC(?count) LIMIT 300
```

グループ化とラベル

単純にラベルを求めただけではダメ
　?kwに対して?labelが複数あったとき何を出力すべきか不明

エラー例）

```
SELECT  ?label (count(?kw) AS ?count) WHERE {
    [] a iscover:Article;      dcterms:subject ?kw .
    ?kw skos:prefLabel ?label .
} GROUP BY ?kw ORDER BY DESC(?count) LIMIT 300
```

サンプルによる出力）

```
SELECT  SAMPLE(?label) (count(?kw) AS ?count)
WHERE {
    [] a iscover:Article; dcterms:subject ?kw .
    ?kw skos:prefLabel ?label .
} GROUP BY ?kw ORDER BY DESC(?count) LIMIT 300
```

被引用文献数

```
SELECT ?article (count(?article) as ?nrefer) WHERE {
  ?article a iscover:Article .
  ?referrer dcterms:reference/cito:cites ?article
  } group by ?article order by desc(?nrefer) limit 100
```

文字列処理

要約に文字列"Linked Data"を含む文献

```
SELECT * WHERE {
  ?article a iscover:Article;  dcterms:abstract ?abstract.
   FILTER(CONTAINS(?abstract, "Linked Data"))
   } limit 10
```

データ整備

例）名前に#を含む

```
SELECT * WHERE {
  ?person a iscover:Person; foaf:name ?name.
   FILTER(CONTAINS(?name, "#"))
   }
```

例）否定（vcard:org を持たないPerson）

```
SELECT * WHERE {
?person a iscover:Person; foaf:name ?name
MINUS {?person vcard:org ?org}
} limit 100
```

データ整備（同姓同名）

例）vcard:orgを持たないPersonと同姓同名の所属先

```
SELECT sample(?name) ?person sample(?org2)  ?person2
WHERE {
?person a iscover:Person; foaf:name ?name
MINUS {?person vcard:org ?org}
?person2 a iscover:Person; foaf:name ?name; vcard:org/
foaf:name ?org2
} group by ?person ?person2 order by ?person limit 100
```

Named Graph

3つ組の集合を区別したいという要求
　例）データソースの違いや権利関係の違いなど
Triple(3つ組)ではなくQuad(4つ組)で表現される

```
SELECT DISTINCT ?g WHERE {
  GRAPH ?g {?s ?p ?o} }
```

現在のI-Scoverは、内部データを除いて一つのNamed Graph

Federated クエリ

SERVICE句：他のSPARQLエンドポイントを指定してクエリを実行。利用できるエンドポイントが限られる

例）紫綬褒章を受賞している（同姓同名の）会員（未チェック[空白問題がある]）

```
SELECT * WHERE {
   SERVICE <http://ja.dbpedia.org/sparql> {
[] dcterms:subject  <http://ja.dbpedia.org/resource/
Category:紫綬褒章受章者>;  foaf:name ?name }
   ?s a iscover:Person; foaf:name ?name}
```

I-Scoverエンドポイントは、Federatedクエリを許可しない

SPARQL結果を使った作業

主なライブラリと可視化ツール

RDFライブラリ
　Jena(Java), rdflib(Python), ARC2(PHP), PerlRDF(Perl), RDF.qb(Ruby) など

SPARQLクライアントライブラリ
　ARQ(Java), SPARQLWrapper(Python) など

可視化ツール
　d3.js, c3.js, vis.js,
　d3sparql, sgvizler

SPARQL1.1クエリ結果JSON フォーマット

SPARQL結果（SELECT）をJSONフォーマットによるシリアル化は、配列としてシリアル化され、個々のエレメントはクエリ結果の1つの「列」になる
例）

```
{
 "head": { "vars": [ "book" , "title" ]
 } ,
 "results": {
  "bindings": [
   {
    "book": { "type": "uri" , "value": "http://example.org/book/
book6" } ,
    "title": { "type": "literal" , "value": "Harry Potter and the Half-
Blood Prince" }
   } ,
```

Python言語からの利用

準備：pip install SPARQLWrapper

```
from SPARQLWrapper import SPARQLWrapper, JSON
query = """PREFIX iscover: <http://i-scover.ieice.org/terms/iscover#>  PREFIX
dcterms: <http://purl.org/dc/terms/> SELECT ?year (count(?s) AS ?count)
WHERE {?s a iscover:Article ; dcterms:issued ?date BIND(year(?date) AS ?year)}
GROUP BY ?year ORDER BY ?year"""
sparql = SPARQLWrapper("http://i-scover-api.ieice.org/iscover/api/sparql")
sparql.setQuery(query)
sparql.setReturnFormat(JSON)
results = sparql.query().convert()
for result in results["results"]["bindings"]:
  print result["year"]["value"], result["count"]["value"]
```

R言語からの利用準備

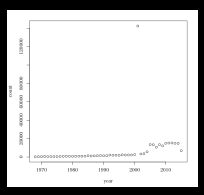
R言語とは：統計解析向きプログラミング言語
Rのインストール：http://cran.r-project.org からダウンロード
　Macの場合 R-3.2.4.pkg(2016.4.11現在)
パッケージSPARQLのインストール：
　Macの場合 アプリケーションRを起動、「パッケージとデータ」から「パッケージインストーラ」を選択、CRAN（バイナリ）の「一覧を取得」し、SPARQLを選択して「選択をインストール」

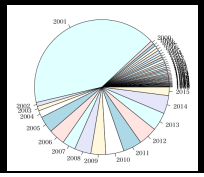
RからのSPARQL利用

```
> library(SPARQL)
> endpoint <- 'http://i-scover-api.ieice.org/iscover/api/sparql'
> q <- "PREFIX iscover: <http://i-scover.ieice.org/terms/
iscover#>
PREFIX dcterms: <http://purl.org/dc/terms/>
SELECT ?year (count(?s) AS ?count) WHERE {
      ?s a iscover:Article ;
      dcterms:issued ?date .
  BIND(year(?date) AS ?year)
} GROUP BY ?year ORDER BY ?year"
> res <- SPARQL(endpoint, q)$results
```

Rでの統計処理やグラフ描画

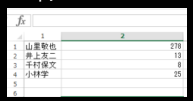
統計処理
```
> sum(res$count)
> summary(res)
```
散布図
```
> plot(res)
```
Pie Chart
```
> pie(res$count, label=res$year)
```





Excelでの利用(各研究者の論文数)

EXCEL2013以降
1) カラムAに研究者名を入れる(日本語、空白なし)
2) B1セルに以下を入力
=FILTERXML(WEBSERVICE("http://i-scover-api.ieice.org/iscover/api/sparql?query=" & ENCODEURL("PREFIX iscover: <http://i-scover.ieice.org/terms/iscover#> PREFIX dcterms: <http://purl.org/dc/terms/>SELECT (count(?a) as ?n) WHERE {?p a iscover:Person; foaf:name """ & RC[-1] & """@ja. ?a iscover:authorInfos/rdf:rest*/rdf:first/iscover:authorInfo/iscover:author ?p} GROUP BY ?p")),"//binding/*")
3) B1セルの内容をB2以下にコピー



まとめ

SPARQL は強力なクエリー言語
- 組み合わせ爆発が起こりやすい
- データ加工が弱い
- モジュール性が悪い

すべてをSPARQLですまそうとすべきではない
- 様々なプログラミング言語の中でSPARQLが扱える
- データ取得の部分とデータ加工は分離すべき

あるデータに対して、どんな操作を行えるか、そのためのSPARQLはどれか(SPARQLを意識せずに)、結果をどう加工するかを 管理・実行できる環境を提供すると良い。

FUJITSU

shaping tomorrow with you

Green Policy Innovation