SPARQLを利用したI-Scover DBの分析・処理の演習（初級）

富士通研究所　西野文人

I-Scoverデータに対するSPARQL検索

サンプルの取得

- LIMIT 句による個数制限

```
SELECT ?s ?p ?o WHERE {?s ?p ?o . } LIMIT 20
```

・「*」で出力内容を個別に指定せずにすべて表示可
・最後のトリプルのピリオドは省略可
・WHEREも省略可
・OFFSETによって指定された数以降の結果を出す

```
SELECT * {?s ?p ?o} LIMIT 20 OFFSET 20
```

・ORDER BY によって出力順序を制御する（ただし、時間がかかる）

```
SELECT * {?s ?p ?o} ORDER BY ?s LIMIT 20
```

・Subjectの指定 によってサンプル確認

```
SELECT * {<http://xxx> ?p ?o}
```

統計情報の取得

- COUNT：数を数える
- DISTINCT：確実にユニークにする

- トリプル数、サブジェクト数、述語数、オブジェクト数

```
SELECT  count(*)  count(distinct ?s) WHERE { ?s ?p ?o }
```

*のcountはトリプル数。
?sをユニークして数えることでサブジェクト数がとれる
Virtuosoでは上でうまくいくが、正しくは

```
SELECT
    (count(*) AS ?ntriples)
    (count(distinct ?s) AS ?nsubjs)
    (count(distinct ?p) AS ?npreds)
    (count(distinct ?o) AS ?nobjs)
WHERE { ?s ?p ?o}
```

資源とクラス

- 資源はクラスに分割される
- rdf:typeはインスタンスがどのクラスに属するかを表現する
- rdf:typeは　a と略記される。

- クラスの一覧の取得（Virtuosoは○）

```
SELECT DISTINCT ?class (COUNT(?class) AS ?count)
WHERE {?s a ?class} ORDER BY DESC(?count)
```

・グループ化によるクラスの一覧の取得

```
SELECT ?class (COUNT(?s) AS ?count)
WHERE {?s a ?class} GROUP BY ?class
ORDER BY DESC(?count)
```

クラスを指定したサンプル取得

- WHERE句の中で条件を指定すればよい（?s a ?o を　?s a iscover:Article　とする）

```
PREFIX iscover: <http://i-scover.ieice.org/terms/iscover#>
SELECT * WHERE {
  ?s a iscover:Article .
  ?s ?p ?o } LIMIT 100
```

- I-Scover Endpointでは PREFIX iscoverが設定されているので省略可
- 主語が共通の時（?s ?p1 ?o1 . ?s ?p2 ?o2 .）は、;で省略できる（?s ?p1 ?o1; ?p2 ?o2.）

```
SELECT * WHERE {?s a iscover:Article; ?p ?o } LIMIT 100
```

クラス指定サンプル取得（入れ子）

- ある主語のまわりの情報を出したい
- 解決策１　ORDER BY ?s とする（時間がかかる）
- 解決策２　まず１つの?s だけを取り出す

```
SELECT ?s WHERE {?s a iscover:Article; ?p ?o } LIMIT 1
```

- この結果の?sを使って全情報を取り出す

```
SELECT * WHERE {
   {SELECT ?s WHERE {?s a iscover:Article; ?p ?o } LIMIT 1}
  ?s ?p ?o }
```

練習問題1：述語一覧

- 述語一覧を作成する
- 述語ごとにトリプル数を求める
- その述語が使われているユニーク主語数を求める
- その述語が使われているユニーク目的語数を求める
- トリプル数の逆順に出力する

クラスを指定したサンプル取得

- WHERE句の中で条件を指定すればよい（?s a ?o を　?s a iscover:Article　とする）

```
PREFIX iscover: <http://i-scover.ieice.org/terms/iscover#>
SELECT * WHERE {
  ?s a iscover:Article .
  ?s ?p ?o } LIMIT 100
```

  - I-Scover Endpointでは PREFIX iscoverが設定されているので省略可
  - 主語が共通の時（?s ?p1 ?o1 . ?s ?p2 ?o2 .）は、;で省略できる（?s ?p1 ?o1; ?p2 ?o2.)

```
SELECT * WHERE {?s a iscover:Article; ?p ?o } LIMIT 100
```

FILTERで絞り込む

- 結果を条件で絞り込むにはFILTERを利用する
- 例）2005年4月１日〜2006年3月31日に絞り込む

```
SELECT (count(*) AS ?count) WHERE {
  ?s a iscover:Article ;
     dcterms:issued ?date .
  FILTER (?date > xsd:date("2005-04-01")
       &&?date <= xsd:date("2006-03-31"))
}
```

特定の要件の設定

- 変数への値の設定はVALUES構文を使う
- 例）2005年4月１日〜2006年3月31日あるいは2006年4月１日〜20067年3月31日にマッチするもの

```
SELECT  ?fyear (count(?s) as ?count) WHERE {
VALUES (?start ?end ?fyear) {
     ("2005-04-01" "2006-03-31" "2005年度")
     ("2006-04-01" "2007-03-31" "2006年度")
}
  ?s a iscover:Article ;
  dcterms:issued ?date.
  FILTER (?date > xsd:date(?start) && ?date<= xsd:date(?end))
}
```

GROUPINGによる集計

- ある変数の値でGROUPINGをすることで集計可能
- 年ごとの集計には変数に年を設定→BINDで定義

```
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
```

IF文を使っての年度の設定

- year関数とmonth関数によるyear, monthの取得
- monthの値によりyearないしyear-1 をfyearに設定
- fyear によるグルーピング

```
SELECT ?fyear (count(?s) AS ?count) WHERE {
  ?s a iscover:Article ;
     dcterms:issued ?date .
  BIND(year(?date) AS ?year)
  BIND(month(?date) AS ?month)
  BIND(IF(?month > 3, ?year, ?year-1) AS ?fyear)
} GROUP BY ?fyear ORDER BY ?fyear
```

5年ごとの集計

- ５年単位で一つの値になるようにする

```
SELECT
  CONCAT(?fyear5*5, "~", ?fyear5*5+4)
  (count(?s) AS ?count) WHERE {
  ?s a iscover:Article ;
     dcterms:issued ?date .
  BIND(year(?date) AS ?year)
  BIND(month(?date) AS ?month)
  BIND(IF(?month > 3, ?year/5, (?year-1)/5) AS ?fyear5)
} GROUP BY ?fyear5 ORDER BY ?fyear5
```

練習問題２：出版物の文献

- 「技術研究報告書」の年別文献件数
  - 「技術研究報告書」がどう表現されているか
  - 文献と「技術研究報告書」の関係を把握
  - 年を変数化してグルーピング

文献と著者

- 「文献」と「著者」は直接結ばれていない
  - 「文献」と「文献発表時の著者」という概念が結ばれている
  - 「文献発表時の著者」は「著者」と（その当時の所属）「機関」概念の両方の値を持つ
- 「文献」には「著者順」という概念がある
  - 複数のリンクではなく、リスト構造としている

文献の構造の確認

```
SELECT * WHERE {
  {select ?article {?article a iscover:Article} limit 1}
   ?article ?p ?o}
```

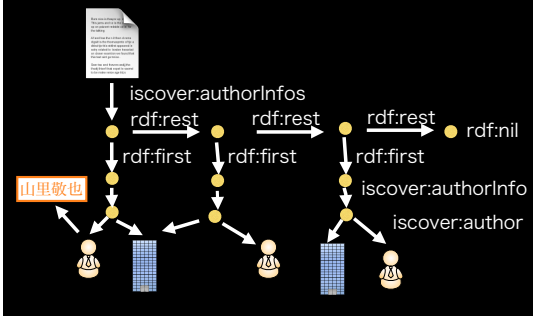
http://i-scover.ieice.org/terms/iscover#authorInfos nodeID://b25423970

- nodeID:// は空白ノード（ユニバーサルな識別子がいらないもの）

```
SELECT * WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos [?p ?o] }
```

- ?s ?p [?p2 ?o2] は ?s ?p ?temp . ?temp ?p2 ?o2 の意

I-Scoverの著者構造



文献の筆頭著者

```
SELECT * WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos [rdf:first
[iscover:authorInfo [iscover:author ?o]]]
}
```

- 第２著者が知りたければ

```
SELECT * WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos [rdf:rest [rdf:first
[iscover:authorInfo [iscover:author ?o]]]]
}
```

プロパティ・パス

SequencePath elt1 / elt2
　elt2が後続するelt1のシーケンス・パス。

- 第２著者

```
SELECT * WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos/rdf:rest/rdf:first/
iscover:authorInfo/iscover:author ?o}
```

全著者を取り出す

ZeroOrMorePath elt* 0以上のeltでマッチ

- 全著者：0以上のrdf:restの後にrdf:firstがある

```
SELECT * WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos/rdf:rest*/rdf:first/
iscover:authorInfo/iscover:author ?o}
```

プロパティー・パス構文

- ^elt　逆パス
- elt1/elt2　シーケンス
- elt1 | elt2　代替パス
- elt*　0個以上
- elt+　1個以上
- elt?　0または１個

最終著者を取り出す

ZeroOrMorePath elt* 0以上のeltでマッチ

- 全著者：0以上のrdf:restの後にrdf:firstがある

```
SELECT ?article ?o WHERE {
   {select ?article {?article a iscover:Article} limit 1}
    ?article iscover:authorInfos/rdf:rest* ?last .
     ?last rdf:rest rdf:nil ;
        rdf:first/iscover:authorInfo/iscover:author ?o
}
```

- 注）　elt*のObjectに空白ノードは不許可なので変数?lastが必要になる

共著者数

```
SELECT ?article count(?authornode) WHERE {
   {select ?article {?article a iscover:Article} limit 10}
    ?article iscover:authorInfos/rdf:rest*/rdf:first ?
authornode
} GROUP BY ?article
```

著者ごとにスコアを割り振って集計

```
SELECT ?author (SUM(?score) AS ?sum) where {
  {select ?article {?article a iscover:Article} limit 10}
  ?article iscover:authorInfos/rdf:rest*/rdf:first/
iscover:authorInfo/iscover:author ?author
{SELECT  ?article (1.0/count(?author) AS ?score)
WHERE {
  ?article iscover:authorInfos/rdf:rest*/rdf:first/
iscover:authorInfo/iscover:author ?author
} GROUP BY ?article}
} GROUP BY ?author
ORDER BY DESC(?sum)
```

名前での表示

```
SELECT ?name ?sum WHERE {
{ … }
 ?author foaf:name ?name }  ORDER BY DESC(?sum)
```

- 日本語名と英語名を持っているとそれぞれでてくる

名前表示の優先順位

- 日本語名と英語名を分けて取り出し、日本語名を優先して表示する
  - langMatches で言語ごとにフィルタリング
  - 存在しない場合に備えてOPTIONALにする
  - 優先順位を与えるにはCOALESCEを使う

```
SELECT COALESCE(?jname, ?ename) ?sum WHERE {
{ … }
 OPTIONAL{?author foaf:name ?jname
FILTER(langMatches(lang(?jname), “ja”)) }
OPTIONAL{?author foaf:name ?ename
FILTER(langMatches(lang(?ename), “en”)) }
} ORDER BY DESC(?sum)
```

練習問題3: 著者順番号

- 文献＄Paperが与えられたときに、筆頭著者なら 1 xxx、第３著者なら 3 yyy のように、著者順を含めて表示せよ

練習問題4: 共著者一覧と共著回数

- ある人物$Personに対して、共著者一覧と共著回数を表示せよ。
- 上記に対して、表記が同じ共著者は同一人物とみなし、名前の表記でグルーピングしてその和を表示せよ