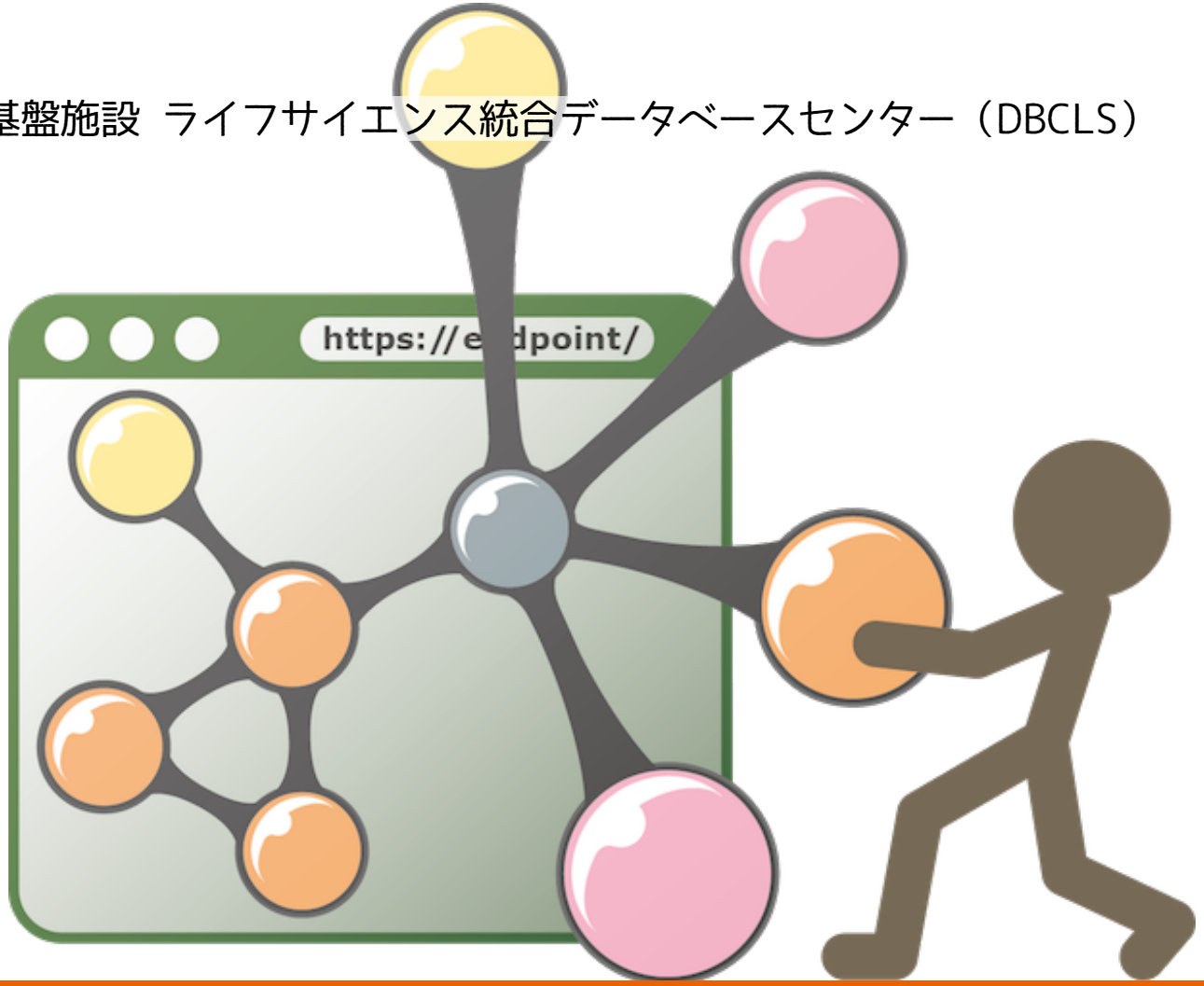
3 Endpoint browser

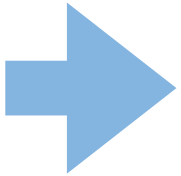
守屋 勇樹
情報・システム研究機構 データサイエンス共同利用基盤施設 ライフサイエンス統合データベースセンター（DBCLS）



概要

- 様々なデータがRDF化されてきている
- けど、SPARQLを書くのは大変

SPARQLクエリの例



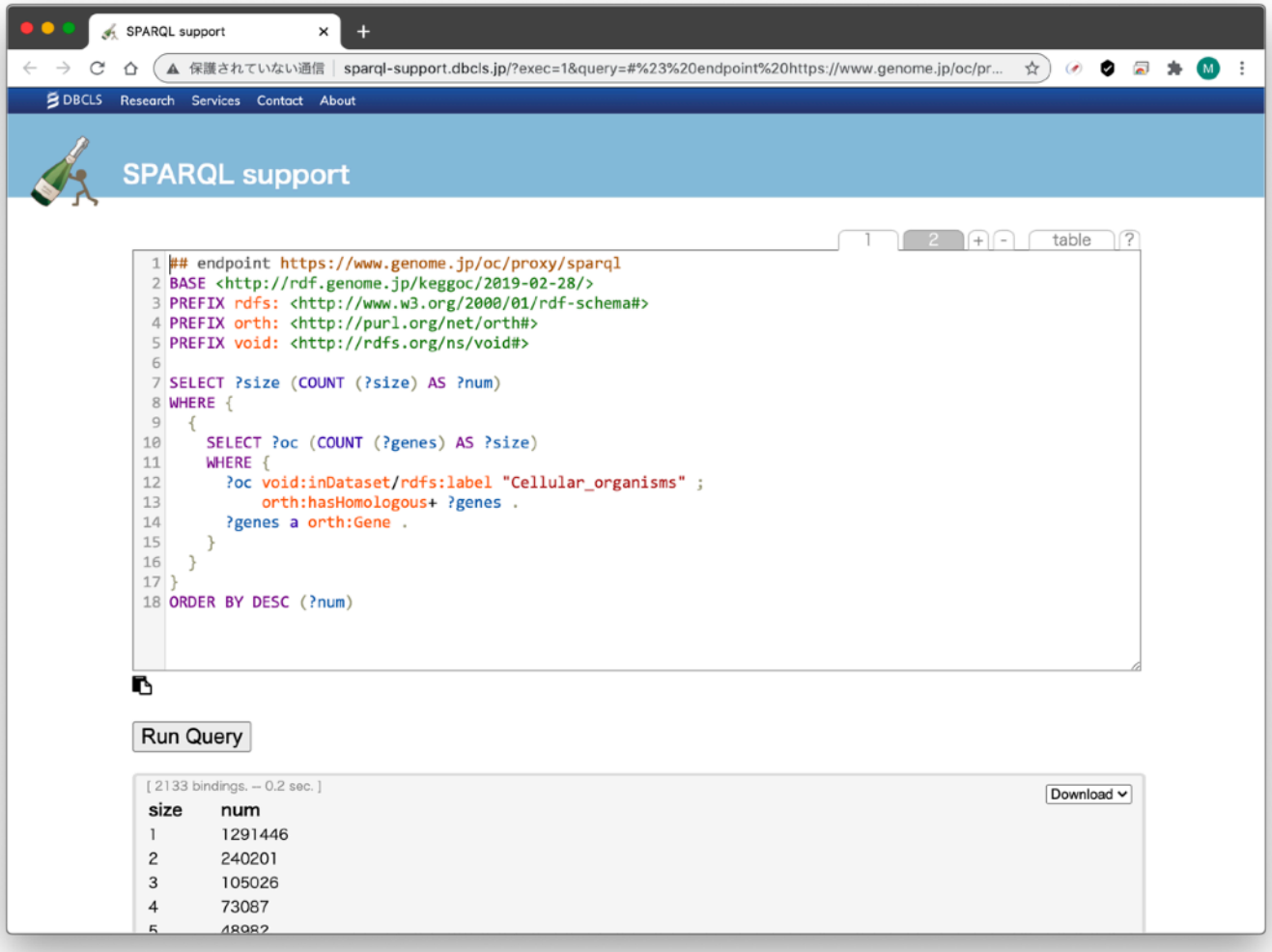
```
## endpoint https://www.genome.jp/oc/proxy/sparql
BASE <http://rdf.genome.jp/keggoc/2019-02-28/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX orth: <http://purl.org/net/orth#>
PREFIX void: <http://rdfs.org/ns/void#>

SELECT ?size (COUNT (?size) AS ?num)
WHERE {
  {
    SELECT ?oc (COUNT (?genes) AS ?size)
    WHERE {
      ?oc void:inDataset/rdfs:label "Cellular_organisms" ;
          orth:hasHomologous+ ?genes .
      ?genes a orth:Gene .
    }
  }
}
ORDER BY DESC (?num)
```

- URIを打ち込むのも面倒
- RDFやクエリによって使うURIは様々なのでコピペすらも面倒
- 使われてる述語も覚えてない

概要

- そこで SPARQL support (https://sparql-support.dbcls.jp/) を開発



- 様々な自動補完
- クエリ管理
- 検索時間表示
- 述語検索
- 複数行コメントアウト

などの機能を持った
SPARQLクライアント

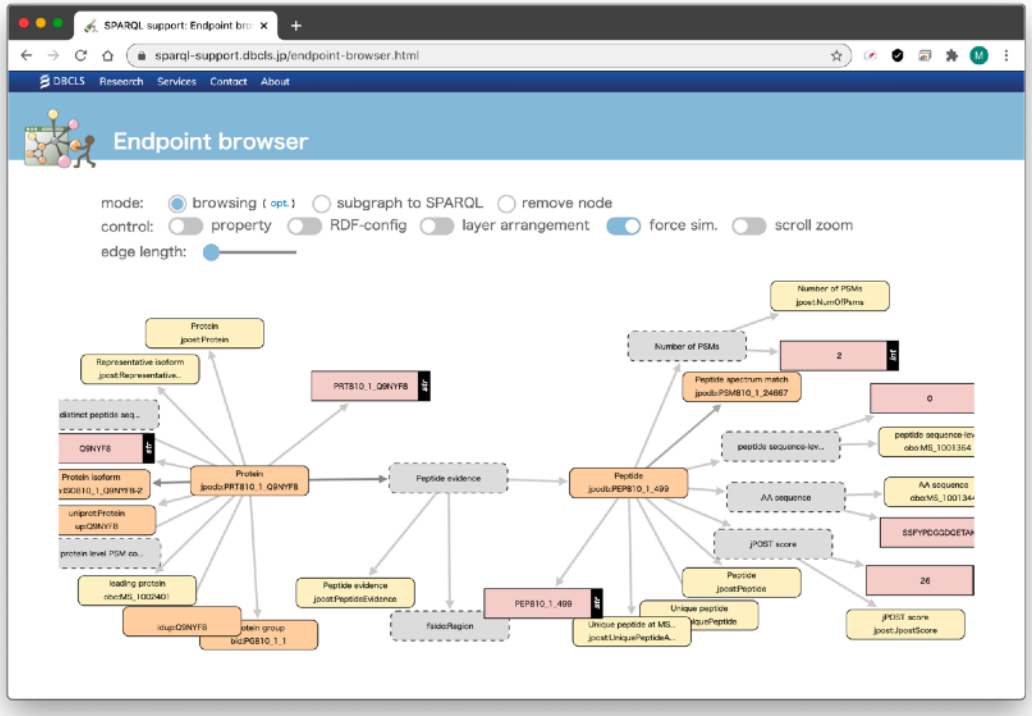
詳細はこちら

概要

- でもSPARQLクエリを書くにはデータ構造を考えないといけないが
  - RDFの構造が複雑
  - そもそも構造を知らない、覚えてない
  - 構造を記述したSchema図が提供されてない
  - ver. upでRDFの構造が変わった、オントロジーが変わった……

そこで Endpoint browser を開発

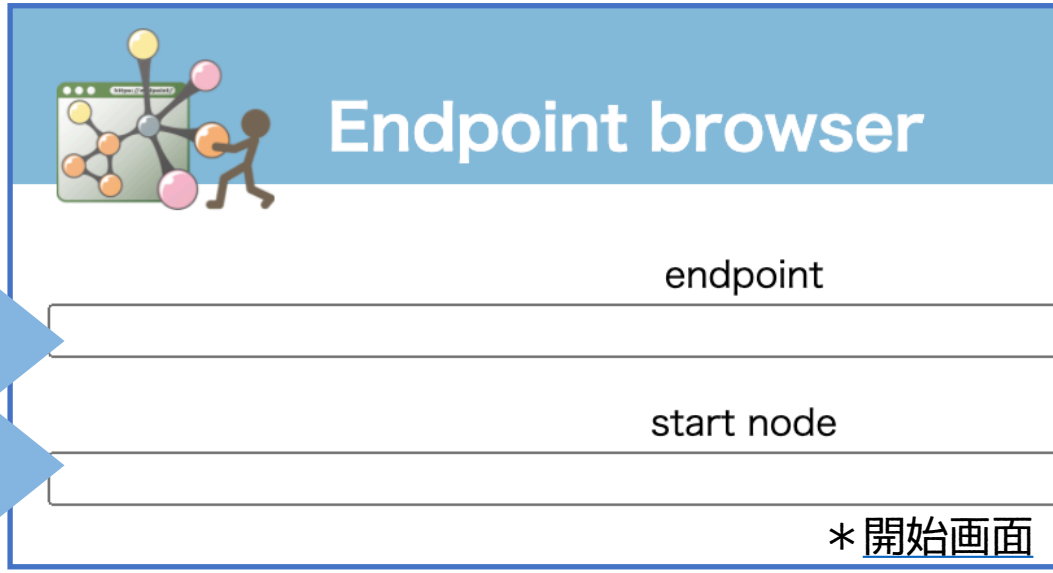
(https://sparql-support.dbcls.jp/endpoint-browser.html)



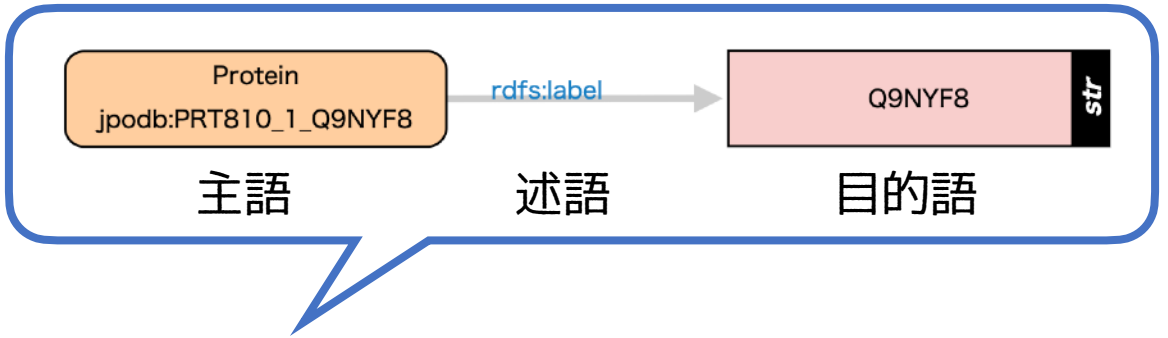
Endpoint browserでできること

- エンドポイントに入ったRDFデータで、
  - 段階的にネットワークグラフ状に可視化
  - クリック操作だけでRDFデータの探索
  - クリック操作で簡単なSPARQLクエリを実行
  - エンドポイントを跨いだ検索
  - RDF-configの生成

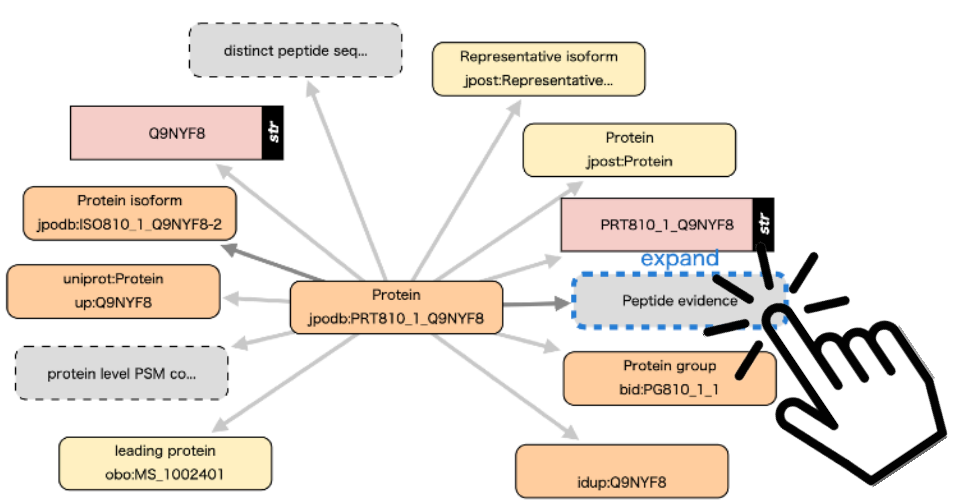
- 用意するものは、
  - エンドポイントのURL
  - 最初のノードのURI（またはリテラル）

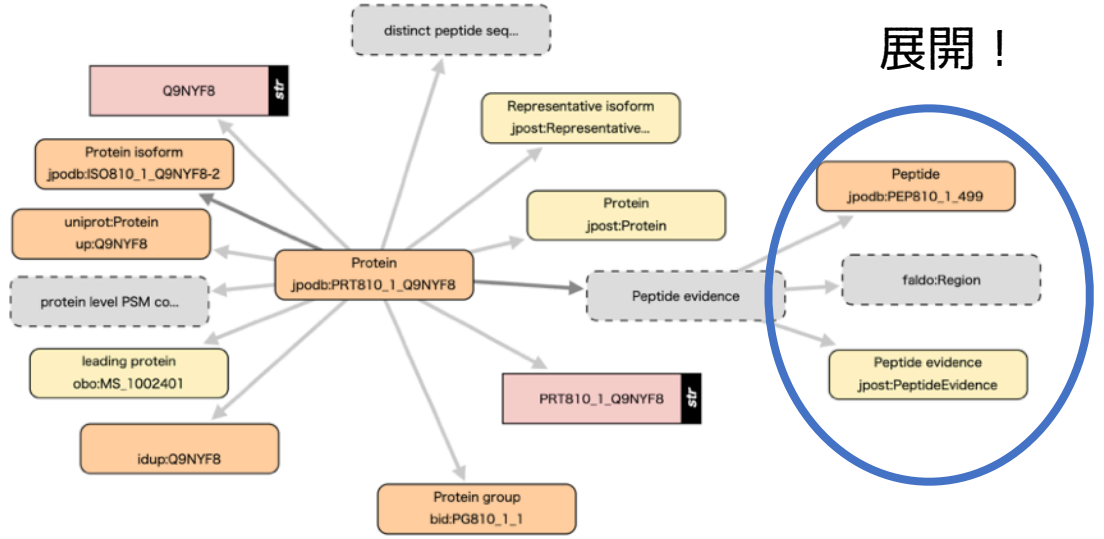


Endpointの探索



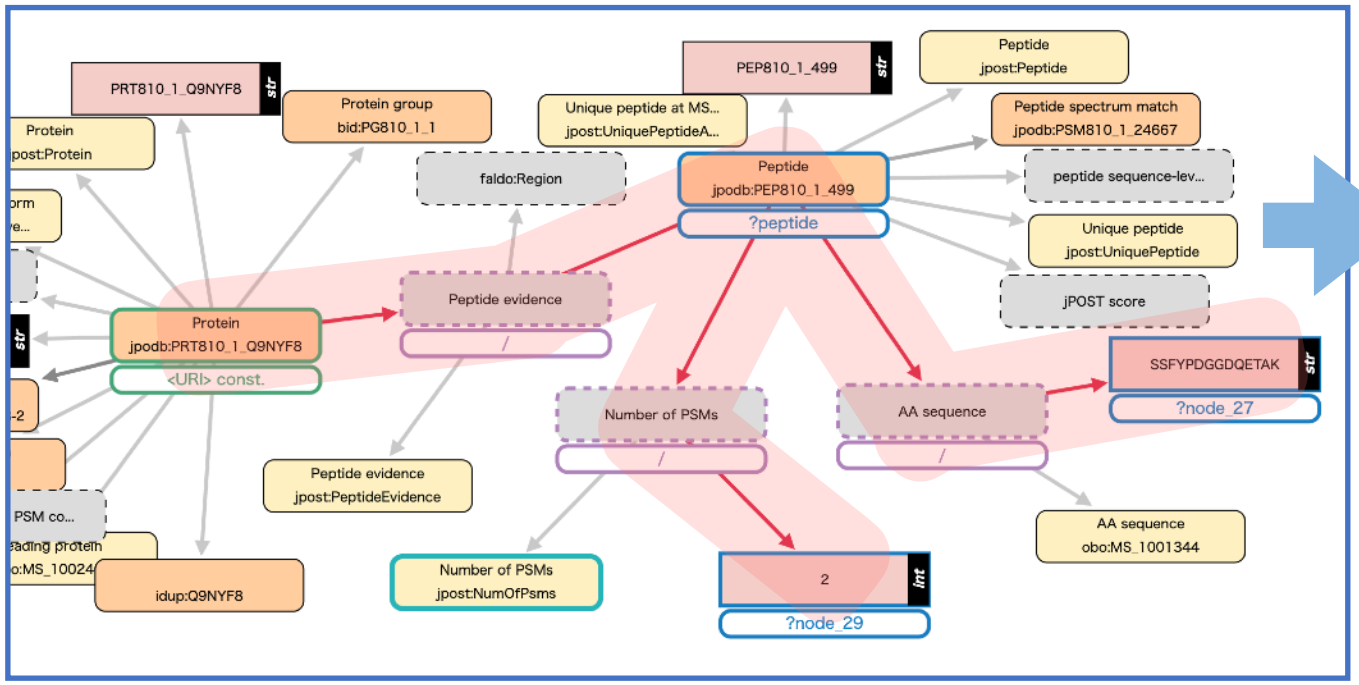
- 最初のノードを主語とする、複数のトリプルから構築されたネットワークグラフが表示
- ノードをクリックしていくことで、ネットワークが展開





ノードは class instance literal type blank で色が分かれています

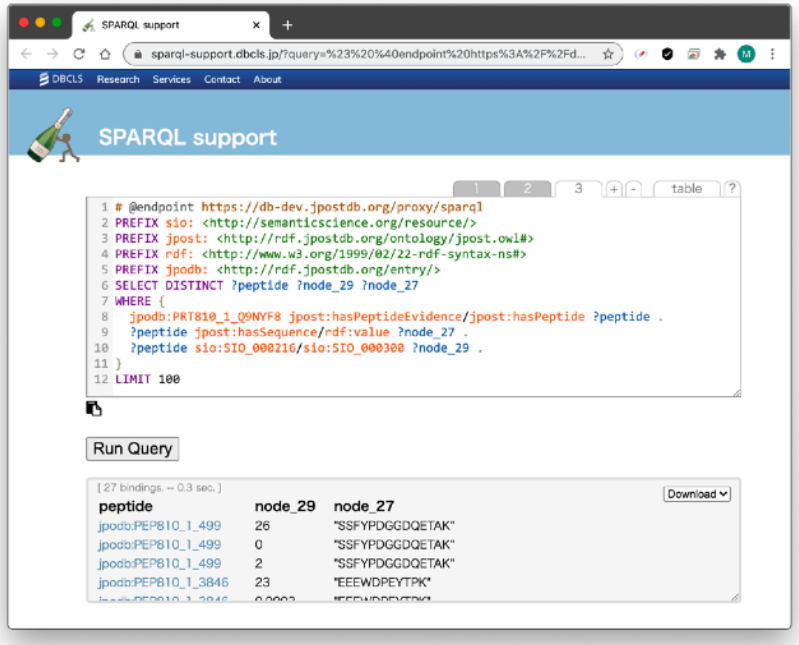
SPARQLクエリの発行と実行




```
# @endpoint https://db-dev.jpostdb.org/proxy/sparql
PREFIX sio: <http://semanticscience.org/resource/>
PREFIX jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX jpodb: <http://rdf.jpostdb.org/entry/>
SELECT DISTINCT ?peptide ?node_29 ?node_27
WHERE {
  jpodb:PRT810_1_Q9NYF8 jpost:hasPeptideEvidence/jpost:hasPeptide ?peptide .
  ?peptide jpost:hasSequence/rdf:value ?node_27 .
  ?peptide sio:SIO_000216/sio:SIO_000300 ?node_29 .
}
```

LIMIT 100

run

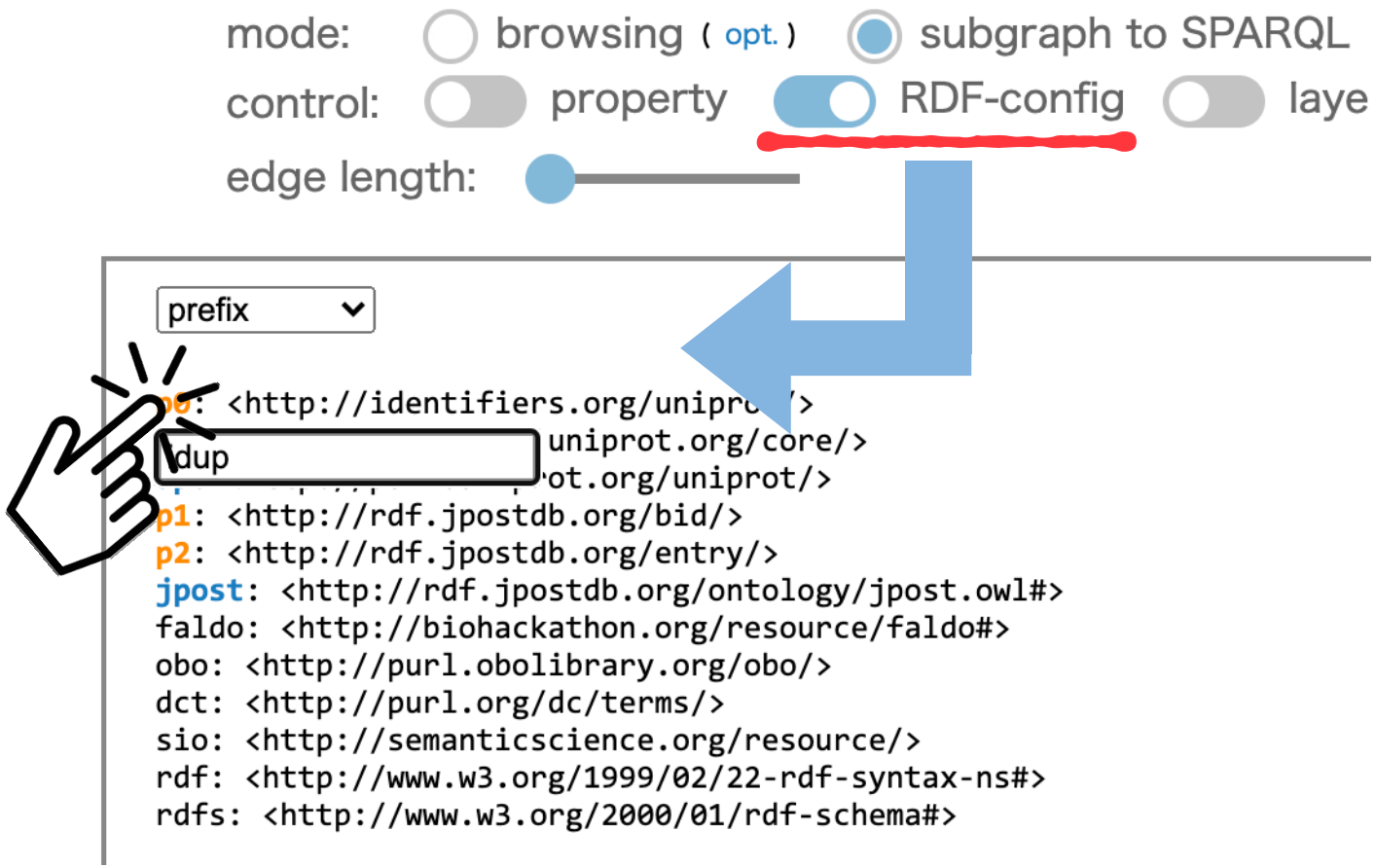


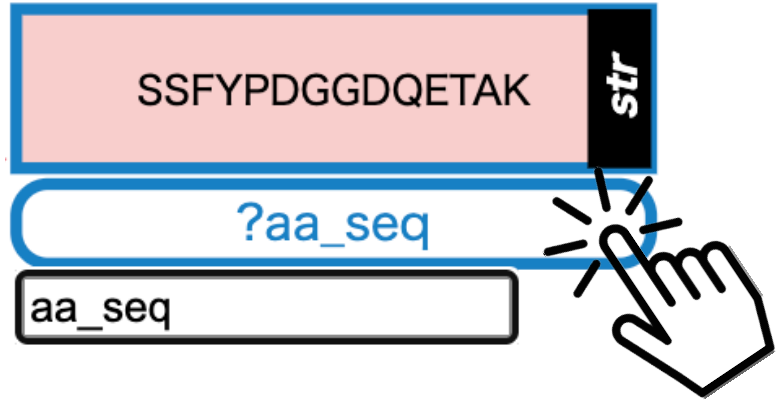
- ノードをクリックして部分グラフを選択するだけでSPARQLクエリが構築
- そのままSPARQL supportで実行

クリックしていくと変数、定数、空白ノードなどの型を選択できます

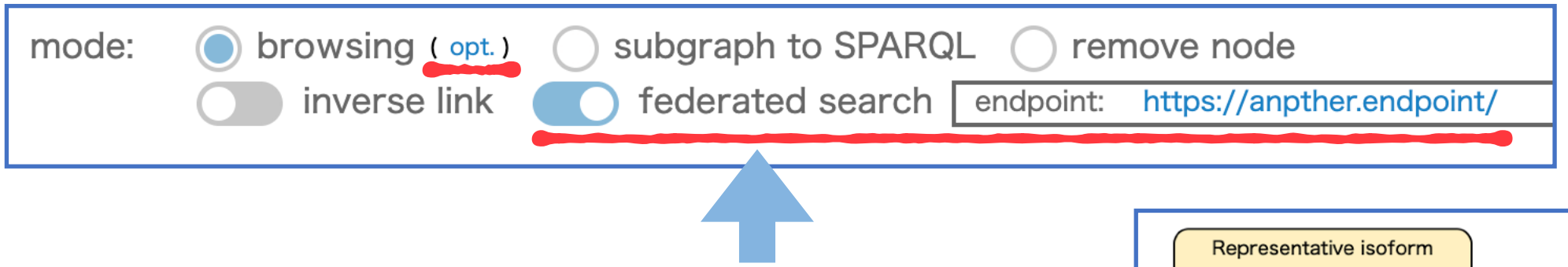
prefixや変数名の自動提案と変更

- pefixや変数名は、ある程度自動で提案
- 自動でつけられたものは分かりにくい場合もあるので変更



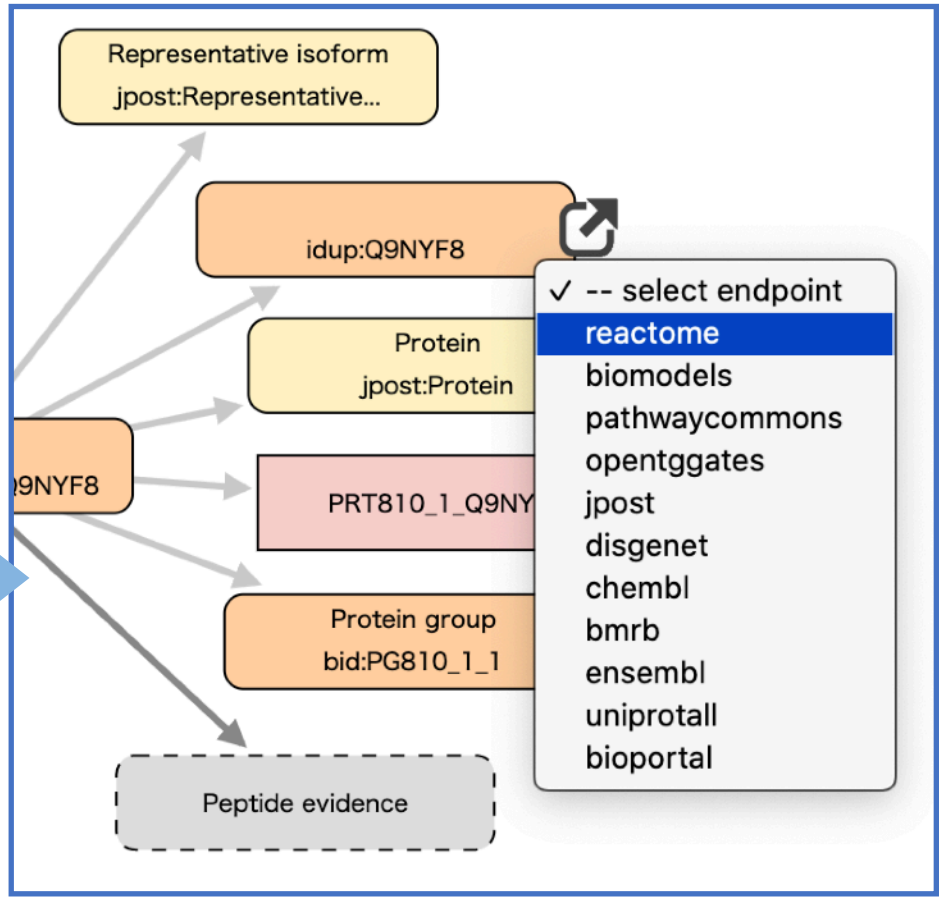


エンドポイントを跨いだ検索



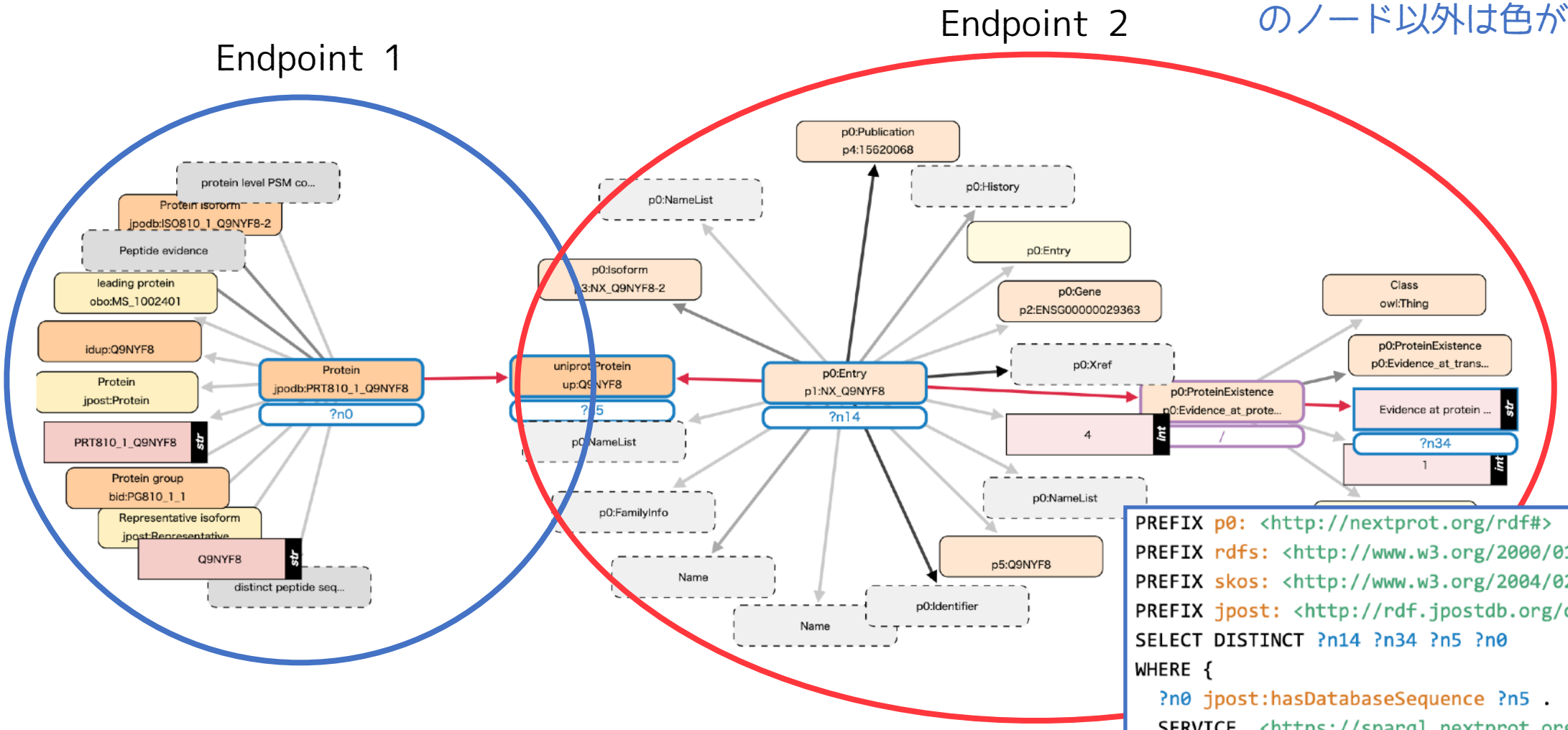
- 他のエンドポイントに切り替えてネットワークグラフを展開することで、エンドポイントを跨いだ検索

- 色々なエンドポイントで使われいている identifiers.org や purl のURIノードからはエンドポイントの候補リストから検索可能



Federatedクエリの発行

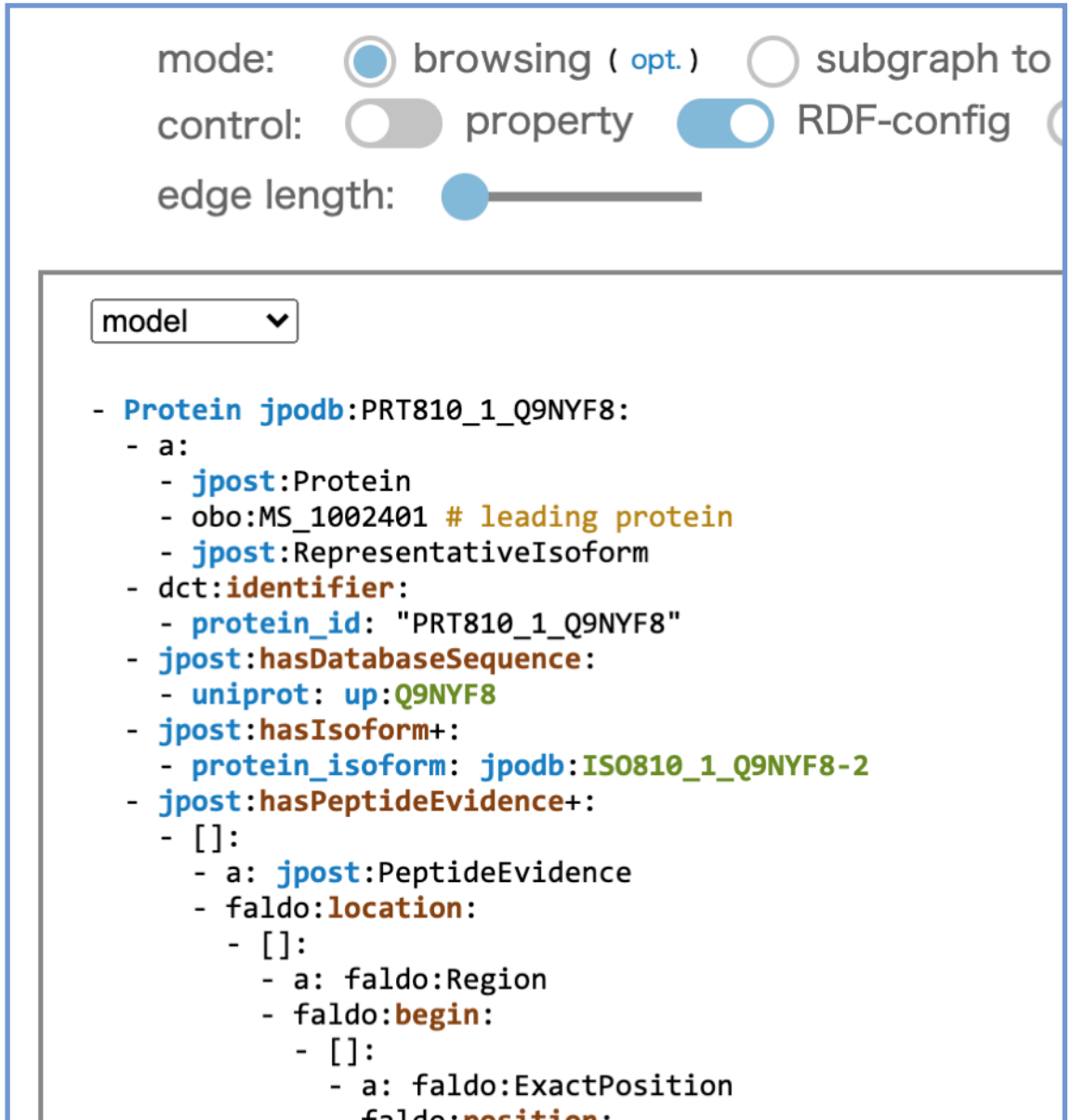
フォーカスしているエンドポイントのノード以外は色が薄くなります




```
PREFIX p0: <http://nextprot.org/rdf#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>
SELECT DISTINCT ?n14 ?n34 ?n5 ?n0
WHERE {
  ?n0 jpost:hasDatabaseSequence ?n5 .
  SERVICE  <https://sparql.nextprot.org/> {
     ?n5 ^skos:exactMatch ?n14 .
  ?n14 p0:existence/rdfs:label ?n34 .
  }
}
```

- エンドポイントを跨いでSPARQLクエリを発行

RDF-configの生成



- RDF-configのyamlファイルを生成
  - ラベルから変数名の自動提案
  - prefix、変数名、出現回数の制約などを簡単に編集
  - スキーマ図などのRDF-config出力チェック機能

RDF-configについては4番のポスターを見て下さい

作成チュートリアルはこちら

Docker

- 公開されたエンドポイント向けのサービスだが、
  Dockerを使ってローカルでサービスを構築することで
  非公開エンドポイントでも利用可能

まとめ

- クリックで、RDFデータを探索できる
  SPARQLクエリを生成できる
  RDF-configを作れる