7 SPARQL support の機能開発

DBCLS Database Center for Life Science

守屋 勇樹
情報・システム研究機構 ライフサイエンス統合データベースセンター（DBCLS）

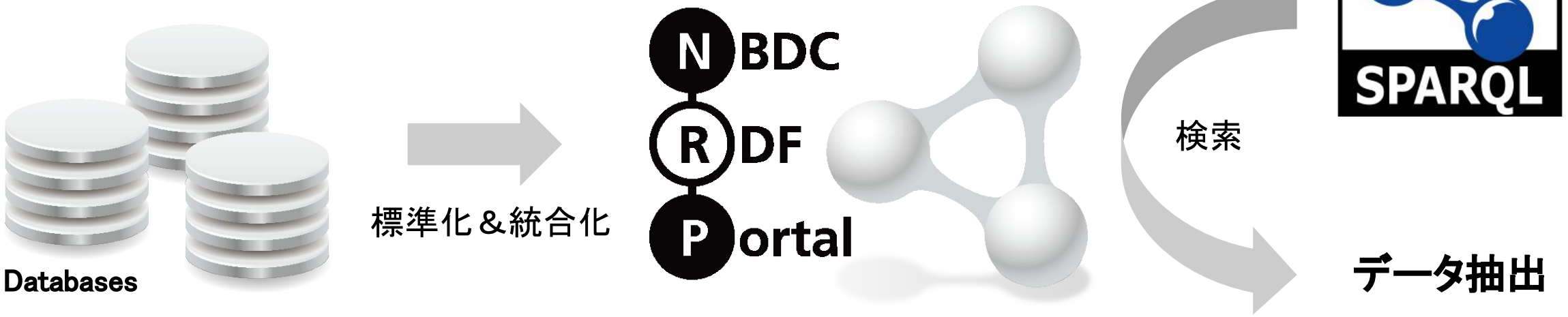
https://sparql-support.dbcls.jp/

SPARQL support とは

Web ブラウザ上で動作する SPARQL クライアント・SPARQL クエリエディタ

クエリ記述を補助する機能を持っている

開発の背景



これまで多くのライフサイエンス分野のデータが知識グラフ化(RDF化) され、SPARQLエンドポイントが公開されている

問い合わせ言語 SPARQL を使って検索

SPARQL support の UI

SPARQL クエリ

- RDF は ’主語・述語・目的語’ のトリプルでリソースの関係性を記述した知識グラフ
- 検索したいトリプルパターンを SPARQL クエリで記述する

1 2 3 4 + - table ?

```
1  # @endpoint https://db-dev.jpostdb.org/proxy/sparql
2  PREFIX : <http://rdf.jpostdb.org/entry/>↔
8  SELECT DISTINCT ?peptide ?uniprot ?seq ?begin
9  WHERE {
10   ?protein jpost:hasPeptideEvidence [
11     faldo:location/faldo:begin/faldo:position ?begin ;
12     jpost:hasPeptide ?peptide
13   ] ;
14             jpost:hasDatabaseSequence ?uniprot .
15   ?uniprot core:reviewed true .
16   ?peptide jpost:hasSequence/rdf:value ?seq .
17 }
```

基本はトリプルの集合なので
難しくは無いが
愚直にタイピングして
いくのは面倒

Run Query

結果

- Table, json 形式で検索結果が得られる

[8 bindings. -- 0.1 sec.]

Download

peptide	uniprot	seq	begin
:PEP791_1_1	uniprot:A2A5R2	"HLDVDLDRQSLSSIDR"	1509
:PEP791_1_7	uniprot:A2AJT4	"QRSPIALPVK"	209
:PEP791_1_10	uniprot:A2AJI0	"SASASPLTPCSAPR"	366
:PEP791_1_115	uniprot:A6H8H2	"STSLSALVR"	1319
:PEP791_1_132	uniprot:D3YXK2	"RSVVSFDK"	622

プロテオームデータベースから、実験で同定されたペプチド配列と、タンパク質上でのポジションを取得するクエリの例

SPARQL support の機能

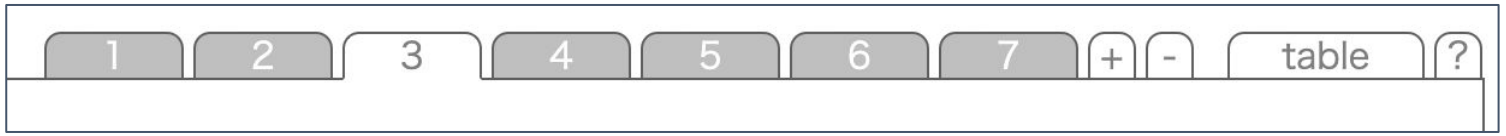
詳細は https://sparql-support.dbcls.jp/sparql-support_j.html#sparql_support

自動補間：変数や関数、句、PREFIX などをショートカットキー操作で補間

```
?h|              -->   ?hoge|
r|               -->   rdf:|
F|               -->   FILTER (|)
PREFIX obo:|     -->   PREFIX obo: <http://purl.obolibrary.org/obo/>|
<id>|            -->   <http://identifiers.org/>|
PREFIX #2|       -->   PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
                      PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
                      PREFIX dct: <http://purl.org/dc/terms/>|
COPY #1|         -->   クエリ全体をコピー
```

Z Shell のような補間で
クエリ作成のコストを削減

クエリ管理：複数のクエリをブラウザに保存して管理



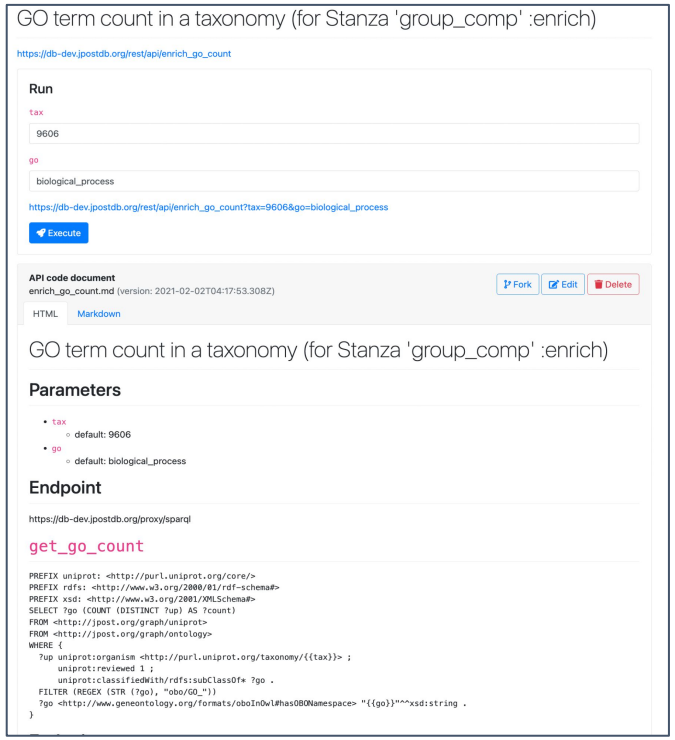
エンドポイント指定：クエリに埋め込み

```
# @endpoint https://example.org/sparql
```

SPARQL のノーコード・ローコード化（DBCLS の取り組み）

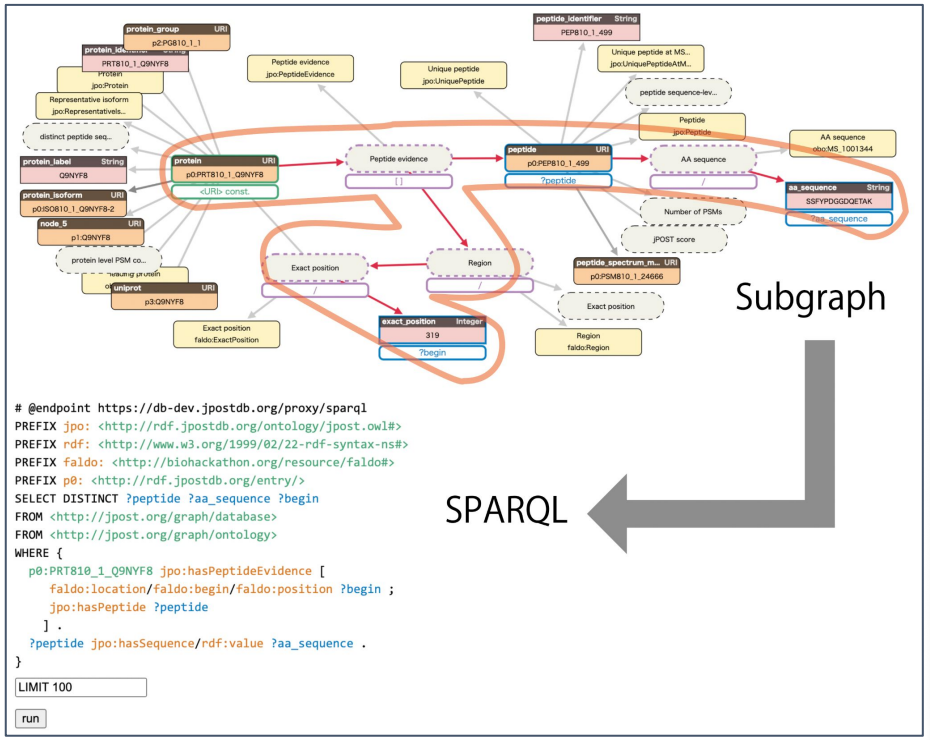
SPARQList

SPARQL と JavaScript で REST API 化（API ユーザはノーコード）



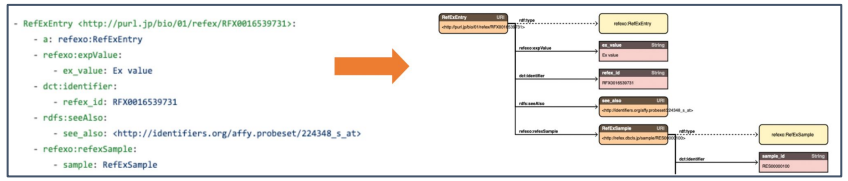
Endpoint browser

視覚的にトリプルパターンを選択して、SPARQL クエリを自動生成



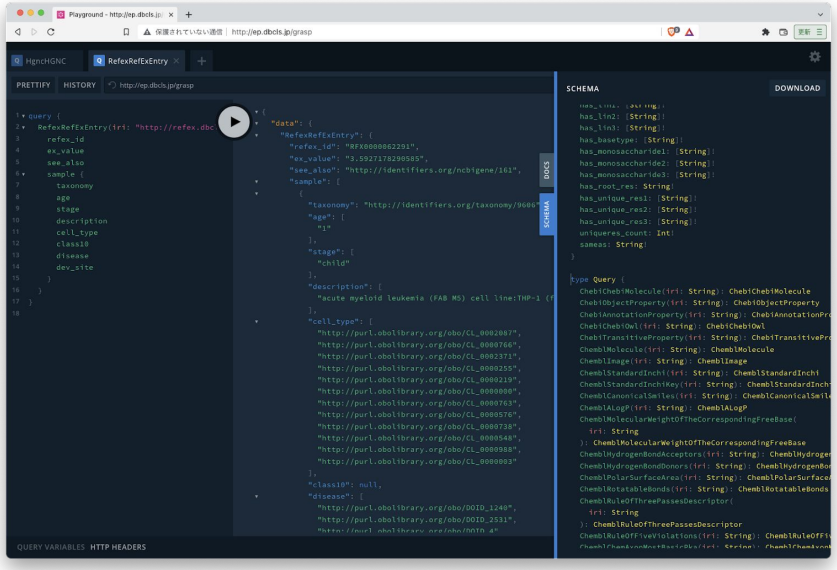
RDF-config

簡潔にしたデータモデル Yaml から SPARQL クエリや Schema 図を自動生成



Grasp

GraphQL で RDF からデータ取得



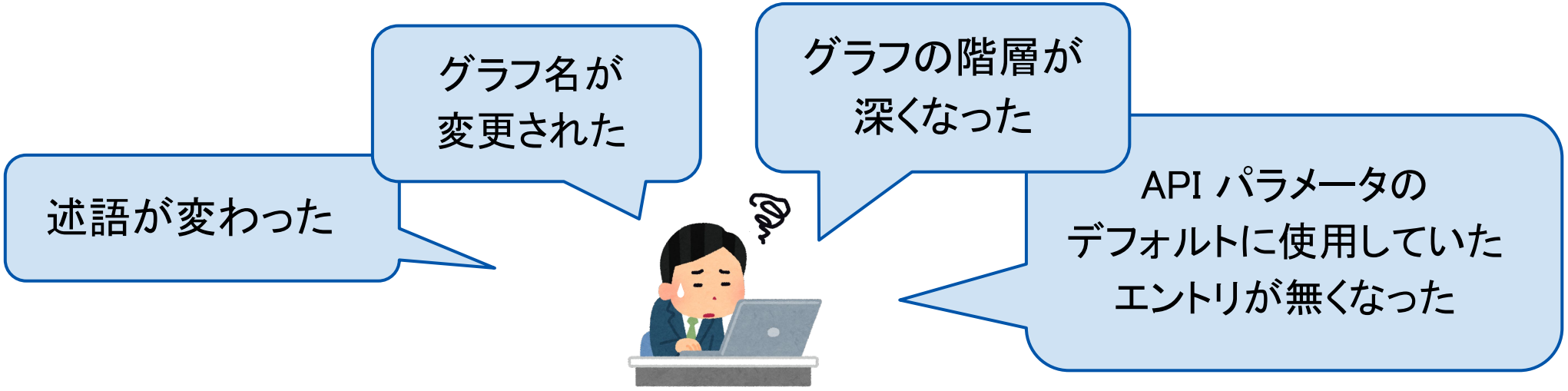
いろいろな取り組みで SPARQL クエリ記述の削減を目指しているが、REST API の開発やメンテナンス、自動生成されたシンプルなクエリの編集などで、しばらくの間は、開発者はこれまで通り SPARQL クエリを記述していく必要がある

課題

RDF portal への集約で統合して検索できるようになった

- SPARQL クエリの複雑化
- 全ての更新箇所の把握が困難

データ更新したらクエリが正常に動作しなくなることも 'DBあるある' だが、デバッグが煩雑



クエリのどこに問題があるかを体系的に調査するには、本来は SPARQL 構文解析が必要

- 高コスト

複数行コメントアウト

SPARQLは1行コメントアウトのみ
初期から =b から =e までコメントアウトする機能を開発していた

```
# @endpoint https://db-dev.jpostdb.org/proxy/sparql
PREFIX jpost: <http://rdf.jpostdb.org/ontology/jpost.owl#>↔
SELECT DISTINCT ?peptide ?aa_sequence ?exact_position ?protein
FROM <http://jpost.org/graph/database>
FROM <http://jpost.org/graph/ontology>
WHERE {
  # 1行コメント
  =b 複数行コメント
  ?protein jpost:hasPeptideEvidence [
    faldo:location/faldo:begin/faldo:position ?exact_position ;
    jpost:hasPeptide ?peptide
  ] . =e
  ?peptide jpost:hasSequence/rdf:value ?aa_sequence .
}
LIMIT 100
```

残った部分で検索することでデバッグ作業

```
## endpoint https://data.allie.dbcls.jp/sparql
select (str(?year) as ?PubYear) (SUM(?PubMedCount) AS ?NumOfPMID)
where {
   =b {
        SELECT DISTINCT ?year ?PubMedCount
        WHERE {
            [] allie:publishedIn ?year .
            BIND (0 as ?PubMedCount)
        }
    } UNION {
        SELECT ?year (count (?year) as ?PubMedCount)
        WHERE {
            [] allie:hasShortFormRepresentationOf
                 / rdfs:label "CDS"@en ; =e
               allie:hasLongFormRepresentationOf
                 / rdfs:label "clinical decision support"@en ;
 =b              allie:appearsIn
                   / allie:hasMemberOf
                   / allie:publishedIn ?year .
        } group by ?year
    }
    {
        SELECT (MIN(?year) as ?minyear) (MAX(?yea
        WHERE {
            [] allie:hasShortFormRepresentationOf
                 / rdfs:label "CDS"@en ;
               allie:hasLongFormRepresentationOf
                 / rdfs:label "clinical decision suppo
               allie:appearsIn
                 / allie:hasMemberOf
                 / allie:publishedIn ?year .
        }
    }
    FILTER(?year >= ?minyear && ?maxyear >= ?year) =e
}
```

もう少し楽にデバッグ作業できたら…

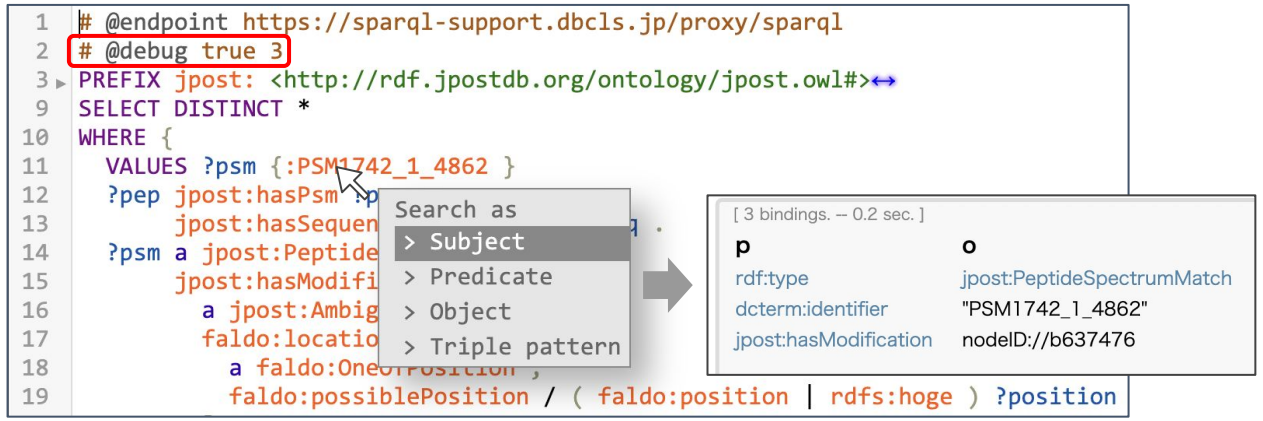
複雑なクエリでは複数箇所コメントアウトしたり、省略した主語や行末のセミコロンを書き換える必要がある

デバッグ検索機能

コメントアウトとは逆に、選んだ URI やトリプルパターンを対象に検索

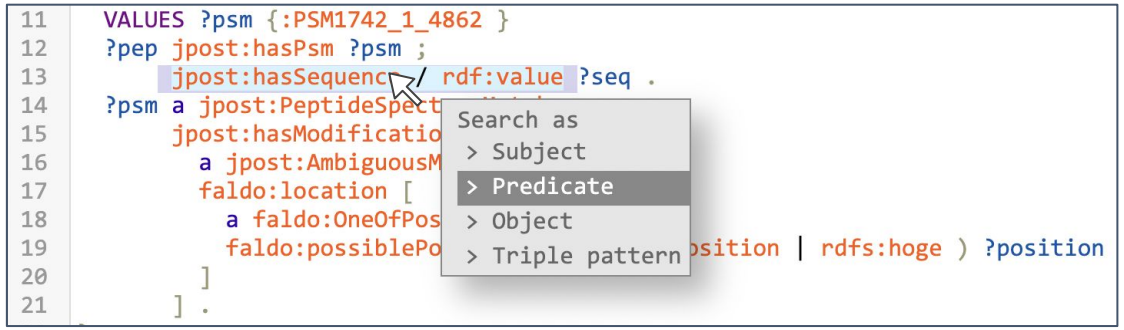
- ‘# @debug true’でモード指定
- 短縮 URI を右クリックで検索
- プロパティパスとか、ブロック単位を範囲選択して右クリックで検索

短縮 URI

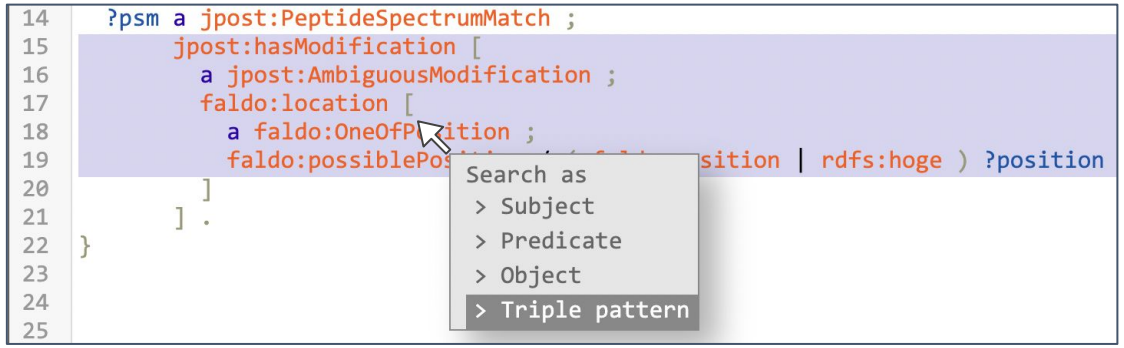


例）選択した URI を主語としたトリプルがエンドポイントで取得できるか検証

プロパティパス



トリプルパターン

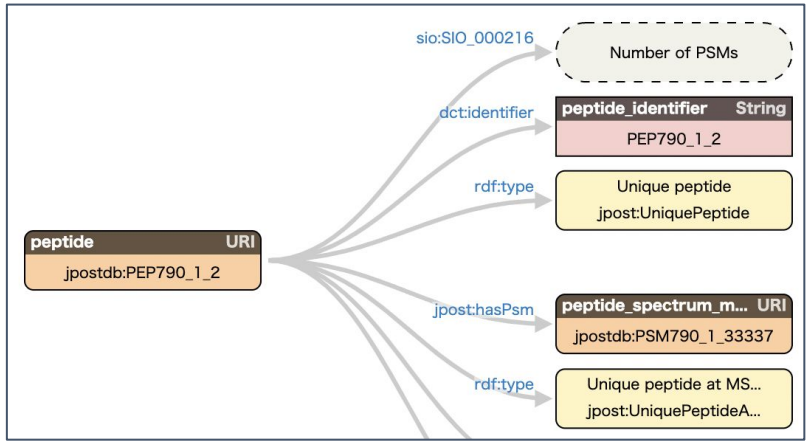


主語や blank node の閉じ括弧、最後のピリオドなどを補完して検索

その他の更新（トーゴーの日2018以降に追加）

peptide	seq	begin
:PEP790_1_2	"TEGATVFGYQVMDPER"	127
:PEP790_1_3	"EAPQTLAQEVDR"	386
:PEP790_1_4	"PVNYHFAPR"	283
:PEP790_1_5	"LSQAFHAHLEETHGQIER"	40
:PEP790_1_6	"NWEEDDGEEYCTAAADLADAQAVCDK"	39

結果から Endpoint broweser



結果から DESCRIBE

DESCRIBE <http://rdf.jpostdb.org/entry/PEP790_1_2>

s	p	o
:DS790_1	jpost:hasPeptide	http://rdf.jpostdb.org/entry/PEP
http://rdf.jpostdb.org/entry/PEP790_1_2	rdf:type	jpost:UniquePeptideAtMsLevel
http://rdf.jpostdb.org/entry/PEP790_1_2	rdf:type	jpost:UniquePeptide
http://rdf.jpostdb.org/entry/PEP790_1_2	rdf:type	jpost:Peptide
http://rdf.jpostdb.org/entry/PEP790_1_2	dcterm:identifier	"PEP790_1_2"
http://rdf.jpostdb.org/entry/PEP790_1_2	sio:SIO_000216	_:vb312452
http://rdf.jpostdb.org/entry/PEP790_1_2	jpost:hasPsm	:PSM790_1_33337

クリップボードへコピー

copy to clipboard　○ raw query　◉ query URL　○ short URL　☐ auto run

クエリや、クエリにアクセスするための URL をクリップボードへコピーすることで、クエリの共有を促進

- 長いクエリを Slack や Docs に書かなくて良くなる

詳細は https://sparql-support.dbcls.jp/sparql-support_j.html#sparql_support

使い方

https://sparql-support.dbcls.jp/

アクセスするだけで利用可能

https://github.com/moriya-dbcls/sparql-support

github からダウンロードして、ローカルでも使える

https://github.com/dbcls/sparql-proxy

SPARQL-proxy に内包しているので、Docker で立てることも可能

募集

欲しい新機能などの相談は随時募集中