API de dados sobre um endpoint SPARQL

2024-05-07 @jcr

Operações CRUD em SPARQL

Neste documento, descreve-se como fazer as operações CRUD (Create, Retrieve, Update e Delete) sobre um conjunto de indivíduos e respetivos atributos através de um endpoint SPARQL.

Como exemplo, iremos usar uma ontologia muito simples, que carateriza um grupo de advogados sentados numa sala. Esta ontologia foi previamente carregada no GraphDB e responde no seguinte endpoint: `http://localhost:7200/repositories/advogados`.

O respetivo modelo em TTL é o seguinte:

```
A1: a :Advogado,
  :Pessoa, owl:NamedIndividual>;
  :bebida "margarita";
  :carro "crossover";
  :cor "verde";
  :dir :A2;
  :idade 40;
  :nome "Otávio";
  :área "imobiliária" .
```

(R)etrieve: Listar

Para listar um conjunto de indivíduos e respetivos atributos vamos usar a query `SELECT`:

```
PREFIX : <http://www.di.uminho.pt/prc2021/advogados#>
SELECT ?adv ?n ?i ?a ?b ?c ?cor
WHERE {
    ?adv a :Advogado ;
        :nome ?n ;
        :idade ?i ;
        :área ?a ;
        :bebida ?b ;
        :carro ?c ;
        :cor ?cor .
}
```

(R)etrieve: Consultar

Para se obter a informação completa de um determinado advogado usamos também uma query `SELECT`:

```
PREFIX : <http://www.di.uminho.pt/prc2021/advogados#>
SELECT * WHERE{
    :A1 ?p ?o
}
```

Neste caso, iríamos obter os triplos referentes ao advogado com identificador `A1`.

(C)reate: Criar

Para criarmos um novo advogado vamos ter de usar uma query especial introduzida na nova versão, 1.1, do SPARQL, a query `INSERT DATA`:

```
PREFIX : <http://www.di.uminho.pt/prc2021/advogados#>
INSERT DATA {
    <triplos a inserir>
}
```

Para realizar este tipo de query temos de alterar o endpoint para `http://localhost:7200/repositories/advogados/statements`.

O método do pedido também deverá ser alterado para `POST`.

(D)elete: Remover/Apagar

Para apagarmos os triplos associados a um advogado vamos ter de usar uma query especial introduzida na nova versão, 1.1, do SPARQL, a query `DELETE... WHERE...`:

```
PREFIX : <http://www.di.uminho.pt/prc2021/advogados#>
DELETE {
    :Ax ?p ?o .
}
WHERE {
    :Ax ?p ?o .
}
```

Para realizar este tipo de query temos de alterar o endpoint para `http://localhost:7200/repositories/advogados/statements`.

O método do pedido também deverá ser alterado para `POST`.

(U)pdate: Alterar/Editar

Para alterarmos os triplos associados a um advogado vamos ter de usar uma query especial introduzida na nova versão, 1.1, do SPARQL, a query `DELETE... INSERT... WHERE...`, que como o modelo apresenta, esta query irá apagar os triplos antigos e inserir os novos:

```
PREFIX : <http://www.di.uminho.pt/prc2021/advogados#>
DELETE {
    ?s ?p ?o .
}
INSERT {
    ?s ?p ?o .
}
WHERE {
    ?s ?p ?o .
}
```

Para realizar este tipo de query temos de alterar o endpoint para `http://localhost:7200/repositories/advogados/statements`.

O método do pedido também deverá ser alterado para `POST`.

Operações POST sobre um endpoint

Estas operações, resultantes da extensão do SPARQL, requerem um endpoint diferente e algumas configurações de baixo nível que se podem abstrair com a utilização do pacote `SPARQLWrapper`.

Eis um modelo para fazer este tipo de pedidos:

```
def sparql_query(query):
    sparql =
SPARQLWrapper("http://localhost:7200/repositories/advogados/statements")
    sparql.setMethod('POST')
    sparql.setQuery(query)
    sparql.setReturnFormat(JSON)
    return sparql.query().convert()
```

Operações GET sobre um endpoint

Estas operações já têm sido realizadas em aulas anteriores mas, podemos tirar também partido do pacote `SPARQLWrapper` para simplificar um pouco a sua especificação:

```
def sparql_get_query(query):
    sparql = SPARQLWrapper("http://localhost:7200/repositories/advogados")
    sparql.setMethod('GET')
    sparql.setQuery(query)
    sparql.setReturnFormat(JSON)
    return sparql.query().convert()
```