SPARQL Query API

- Purpose
- Use Cases
  - Reusing data from VIVO
  - Writing a VIVO "face" application
- Specification
  - URL
  - HTTP Method
  - Parameters
  - Response Codes
  - Available content types
    - For SELECT or ASK queries
    - For CONSTRUCT or DESCRIBE queries
  - Limitation
- Examples
  - SELECT to JSON example
  - DESCRIBE to N3 example
- Enabling the SPARQL Query API

Purpose

Permits external applications to obtain data from the VIVO data model.

The results of the queries are not filtered, so access to the service should remain restricted if the VIVO instance contains any data which should remain private. Queries can be performed against the entire data model, or against specific graphs. .

By default, the SPARQL Query API is disabled in VIVO, for security reasons. See Enabling the API

Use Cases

Reusing data from VIVO

Data in VIVO is available to other applications via Linked Open Data - requests and responses. But some applications may work better with the sort of data sets that can be obtained from SPARQL queries.

Writing a VIVO "face" application

Various VIVO sites have written applications, in Drupal or other such frameworks, that display data from VIVO, and allow the user to edit their data. This API, used in conjunction with SPARQL Update API, allows such an application to freely read or modify VIVO data.

Specification

URL

`[vivo]/api/sparqlQuery`

Examples:

```
http://vivo.cornell.edu/api/sparqlQuery
```

```
http://localhost:8080/vivo/api/sparqlQuery
```

HTTP Method

The API supports HTTP GET or POST calls.

Parameters

name	value
email	the email address of a VIVO administrator account
password	the password of the VIVO administrator account
query	A SPARQL query

The syntax of the SPARQL query is described on the World Wide Web Consortium site at http://www.w3.org/TR/2013/REC-sparql11-query-20130321/

Response Codes

Code	Reason
200 OK	SPARQL query was successful.
400 Bad Request	HTTP request did not include a query parameter.
	The SPARQL query was syntactically incorrect.
	The type of the SPARQL query was not SELECT, ASK, CONSTRUCT, or DESCRIBE
403 Forbidden	HTTP request did not include an email parameter.
	HTTP request did not include a password parameter.
	The combination of email and password is not valid.
	The selected VIVO account is not authorized to use the SPARQL Query API.
406 Not Acceptable	The Accept header does not include any available content types.
500 Internal Server Error	VIVO could not execute the request; internal code threw an exception.

Available content types

The request may include an Accept header, to specify the preferred content type of the response. If no Accept header is provided, the preferred content type is assumed to be text/plain.

For SELECT or ASK queries

SELECT queries return rows of results, and each row may include an arbitrary number of values, depending on the query.

ASK queries return a single result, which is either true or false.

MIME type in the Accept header	Response format	Format description
text/plain	text	
text/csv	CSV	http://www.w3.org/TR/2013/REC-sparql11-results-csv-tsv-20130321
text/tab-separated-values	TSV	
application/sparql-results+xml	XML	http://www.w3.org/TR/2013/REC-rdf-sparql-XMLres-20130321
application/sparql-results+json	JSON	http://www.w3.org/TR/2013/REC-sparql11-results-json-20130321

For CONSTRUCT or DESCRIBE queries

CONSTRUCT and DESCRIBE queries return RDF.

MIME type in the Accept header	Response format	Format description
text/plain	N-Triples	http://www.w3.org/2001/sw/RDFCore/ntriples/
application/rdf+xml	RDF/XML	http://www.w3.org/TR/rdf-syntax-grammar/
text/n3	N3	http://www.w3.org/TeamSubmission/n3/
text/turtle	Turtle	http://www.w3.org/TeamSubmission/turtle/

application/json	JSON-LD	http://www.w3.org/TR/json-ld/

Limitation

Queries can be performed against specific graphs. However, the graphs that hold application data are not accessible to the API. "Application data" means data that controls the functioning of the VIVO application, such as user accounts, page definitions, or display parameters.

Examples

These examples use the UNIX `curl` command to issue queries to the API.

SELECT to JSON example

This example reads 5 arbitrary triples from the data model, returning the result as JSON.

```
curl -i -d 'email=testAdmin@mydomain.edu' -d 'password=Password' -d 'query=SELECT ?s ?p ?o WHERE {?s ?p ?o}
LIMIT 5' -H 'Accept: application/sparql-results+json' 'http://localhost:8080/vivo/api/sparqlQuery'
```

The response looks like this:

```
{
  "head": {
    "vars": [ "s" , "p" , "o" ]
  } ,
  "results": {
    "bindings": [
      {
        "s": { "type": "bnode" , "value": "b0" } ,
        "p": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#rest" } ,
        "o": { "type": "bnode" , "value": "b1" }
      } ,
      {
        "s": { "type": "bnode" , "value": "b0" } ,
        "p": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#first" } ,
        "o": { "type": "uri" , "value": "http://purl.obolibrary.org/obo/ERO_0000006" }
      } ,
      {
        "s": { "type": "bnode" , "value": "b2" } ,
        "p": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#rest" } ,
        "o": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#nil" }
      } ,
      {
        "s": { "type": "bnode" , "value": "b2" } ,
        "p": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#first" } ,
        "o": { "type": "bnode" , "value": "b3" }
      } ,
      {
        "s": { "type": "uri" , "value": "http://vivoweb.org/ontology/core#FacultyMember" } ,
        "p": { "type": "uri" , "value": "http://vitro.mannlib.cornell.edu/ns/vitro/0.
7#hiddenFromDisplayBelowRoleLevelAnnot" } ,
        "o": { "type": "uri" , "value": "http://vitro.mannlib.cornell.edu/ns/vitro/role#public" }
      }
    ]
  }
}
```

DESCRIBE to N3 example

This example reads all of the properties for a particular individual in the model, returning the result as N3.

```
curl -i -d 'email=vivo_root@mydomain.edu' -d 'password=Password' -d 'query=DESCRIBE <http://dbpedia.org/resource
/Connecticut>' -H 'Accept: text/n3' 'http://localhost:8080/vivo/api/sparqlQuery'
```

The response looks like this:

```
@prefix vitro:   <http://vitro.mannlib.cornell.edu/ns/vitro/0.7#> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://dbpedia.org/resource/Connecticut>
      a       <http://vivoweb.org/ontology/core#StateOrProvince> ,
              <http://purl.obolibrary.org/obo/BFO_0000006> ,
              <http://vivoweb.org/ontology/core#Location> ,
              owl:Thing ,
              <http://vivoweb.org/ontology/core#GeopoliticalEntity> ,
              <http://purl.obolibrary.org/obo/BFO_0000002> ,
              <http://vivoweb.org/ontology/core#GeographicRegion> ,
              <http://purl.obolibrary.org/obo/BFO_0000001> ,
              <http://purl.obolibrary.org/obo/BFO_0000141> ,
              <http://vivoweb.org/ontology/core#GeographicLocation> ,
              <http://purl.obolibrary.org/obo/BFO_0000004> ;
      <http://www.w3.org/2000/01/rdf-schema#label>
              "Connecticut"@en ;
      <http://purl.obolibrary.org/obo/BFO_0000050>
              <http://aims.fao.org/aos/geopolitical.owl#United_States_of_America> ;
      vitro:mostSpecificType
              <http://vivoweb.org/ontology/core#StateOrProvince> .
```

Enabling the SPARQL Query API

Before enabling the SPARQL Query API, you should secure the URL `api/sparqlQuery` with HTTPS. Otherwise, email/password combinations will be sent across the network without encryption. Methods for securing the URL will depend on your site's configuration.

By default, the SPARQL Query API is disabled in VIVO for all users except the root user. To enable it for non-root users, you must edit the RDF file `[vivo]/home/rdf/auth/everytime/permission_config.n3` to authorize your site administrators to use the API. Find the permissions for `auth:ADMIN` and include the following permission:

permission_config.n3

```
auth:hasPermission simplePermission:UseSparqlQueryApi;
```

After editing this file you need to restart tomcat.