RESTful API documentation for RDF2Any

1. Convert API

Actual convert APIs. These APIs will convert the RDF ResultSet to a desired output format , viz., JSON, XML, CSV, RDB , etc

1.1 Convert to JSON

GET **/v1.0/convert/json**

1.1.1 description

This method returns the RDF ResultSet in JSON format. There are two kinds of convert.
The first one is a generic convert in which it converts to standard JSON output of a SPARQL resultset. The other one is a class convert in which the convert is based on a particular class. For class convert it is essential to pass paramaters “for_class” which specifies for which class the convert is based on, and in the “query”, there should be column in the resultset “concept” which has objects of the class.

1.1.2 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
query	true	String	contains the SPARQL query as a string. The query should have an output variable “concept” if a class conversion is being called.
json_output_format	false	String	can have value {“virtuoso”, “sesame”}. Determines the JSON serialization structure based on this value. By DEFAULT its “virtuoso”
for_class	false	String	contains the uri of the class for which the conversion is done. This specifies class conversion needs to be done.
properties	false	String	comma separated uris of properties of the class which are required to be present in the final output.

			To get all the properties , “all” should be passed.

1.1.3 Example request

1.1.3.1 Generic convert

GET
http://localhost:8081/rdf2any/v1.0/convert/json?dataset=http://dbpedia.org/sparql&query=select+distinct+%3FConcept+where+%7B%5B%5D+a+%3FConcept%7D+LIMIT+10


```
{
   "head": {
      "vars": [
         "Concept"
      ],
      "link": []
   },
   "results": {
      "bindings": [
         {
            "Concept": {
               "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#Property",
               "type": "uri"
            }
         },
         {
            "Concept": {
               "value": "http://www.openlinksw.com/schemas/virtrdf#QuadMapFormat",
               "type": "uri"
            }
         },
        …
       ….
      ],
      "distict": false,
      "ordered": true,
      "time_taken": 1.459
   }
}
```


1.1.3.1 Class convert

GET
http://localhost:8081/rdf2any/v1.0/convert/json?dataset=http://dbpedia.org/sparql&query=PREFIX%20rdf%3A%3Chttp%3A%2F%2Fwww.w3.org%2F1999%2F02%2F22-rdf-syn

tax-ns%23%3E%20%0APREFIX%20rdfs%3A%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%20%0ASELECT%20%3Fconcept%20%3Flabel%20WHERE%20%0A%7B%20%3Fconcept%20rdf%3Atype%20%3Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FCity%3E.%0A%20%3Fconcept%20rdfs%3Alabel%20%3Flabel.%0AFILTER(langMatches(lang(%3Flabel)%2C%20%22EN%22))%7D%0A%20LIMIT%2010&for_class=http://dbpedia.org/ontology/City&properties=http%3A%2F%2Fdbpedia.org%2Fontology%2Fcountry%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FpopulationTotal%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FareaCode


```
{
   "class_name": "city",
   "class_uri": "http://dbpedia.org/ontology/City",
   "objects": [
      {
         "hasClass": {
            "dataset": "http://dbpedia.org/sparql",
            "indexCreated": true,
            "label": "city",
            "properties": [
               {
                  "count": 2770,
                  "label": "area code",
                  "multiplePropertiesForSameNode": true,
                  "range": {
                     "label": "string",
                     "uri": "http://www.w3.org/2001/XMLSchema#string"
                  },
                  "type": "data",
                  "uri": "http://dbpedia.org/ontology/areaCode"
               },
               {
                  "count": 13095,
                  "label": "population total",
                  "multiplePropertiesForSameNode": true,
                  "range": {
                     "label": "nonNegativeInteger",
                     "uri": "http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
                  },
                  "type": "data",
                  "uri": "http://dbpedia.org/ontology/populationTotal"
               },
               {
                  "count": 176,
                  "label": "country",
                  "multiplePropertiesForSameNode": true,
                  "range": {
                     "label": "country",
                     "uri": "http://dbpedia.org/ontology/Country"
                  },
```

```
            "type": "object",
            "uri": "http://dbpedia.org/ontology/country"
        }
    ],
    "uri": "http://dbpedia.org/ontology/City"
},
"name": "Léry, Quebec",
"properties": [
    {
        "objects": [
            {
                "additionalValue": "",
                "value": "450 and 579"
            }
        ],
        "predicate": {
            "count": 2770,
            "label": "area code",
            "multiplePropertiesForSameNode": true,
            "range": {
                "label": "string",
                "uri": "http://www.w3.org/2001/XMLSchema#string"
            },
            "type": "data",
            "uri": "http://dbpedia.org/ontology/areaCode"
        }
    },
    {
        "objects": [
            {
                "additionalValue": "",
                "value": "http://dbpedia.org/resource/Canada"
            }
        ],
        "predicate": {
            "count": 176,
            "label": "country",
            "multiplePropertiesForSameNode": true,
            "range": {
                "label": "country",
                "uri": "http://dbpedia.org/ontology/Country"
            },
            "type": "object",
            "uri": "http://dbpedia.org/ontology/country"
        }
    },
    {
        "objects": [
            {
                "additionalValue": "",
```

```
                    "value": "2307"
                }
            ],
            "predicate": {
                "count": 13095,
                "label": "population total",
                "multiplePropertiesForSameNode": true,
                "range": {
                    "label": "nonNegativeInteger",
                    "uri": "http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
                },
                "type": "data",
                "uri": "http://dbpedia.org/ontology/populationTotal"
            }
        }
    ],
    "uri": "http://dbpedia.org/resource/L%C3%A9ry,_Quebec"
  },
…],
  "time_taken": 15.056
}
```


1.2 Convert to CSV

GET **/v1.0/convert/csv-converter.csv**

1.2.1 Description

This method returns the RDF ResultSet in CSV format. There are two kinds of convert.
The first one is a generic convert in which it puts all the returned rows to the the columns of the result. The other one is a class convert in which the convert is based on a particular class. For class convert it is essential to pass parameters “for_class” which specifies for which class the convert is based on, and in the “query”, there should be column in the resultset “concept” which has objects of the class.

1.2.2 Input Parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
query	true	String	contains the SPARQL query as a string. The query should have an output

			variable “concept” if a class conversion is being called.
for_class	false	String	contains the uri of the class for which the conversion is done. This specifies class conversion needs to be done.
properties	false	String	comma separated uris of properties of the class which are required to be present in the final output. To get all the properties , “all” should be passed.

1.2.3 Example request

1.2.3.1 Generic convert

GET
http://localhost:8081/rdf2any/v1.0/convert/csv-converter.csv?dataset=http://dbpedia.org/sparql&query=select+distinct+%3FConcept+where+%7B%5B%5D+a+%3FConcept%7D+LIMIT+100

```
subject,label
http://dbpedia.org/resource/2005_in_South_Korean_football,2005 in South Korean football@en
http://dbpedia.org/resource/Category:2005_in_South_Korean_football,2005 in South Korean football@en
http://wikidata.dbpedia.org/resource/Q4605218,2005 in South Korean football@en
http://dbpedia.org/resource/2010_in_South_Korean_football,2010 in South Korean football@en
http://dbpedia.org/resource/Category:2010_in_South_Korean_football,2010 in South Korean football@en
http://wikidata.dbpedia.org/resource/Q4619094,2010 in South Korean football@en
http://dbpedia.org/resource/List_of_Federal_Roads_in_Sarawak,List of Federal Roads in Sarawak@en
http://wikidata.dbpedia.org/resource/Q6570751,List of Federal Roads in Sarawak@en
http://dbpedia.org/resource/Wang_Jingyao,Wang Jingyao@en
http://wikidata.dbpedia.org/resource/Q4018111,Wang Jingyao@en
 ……..
```

1.2.3.2 Class convert

GET
http://localhost:8081/rdf2any/v1.0/convert/csv-converter.csv?dataset=http://dbpedia.org/sparql&query=PREFIX%20rdf%3A%3Chttp%3A%2F%2Fwww.w3.org%2F1999%2F02%2F22-rdf-syntax-ns%23%3E%20%0APREFIX%20rdfs%3A%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%20%0ASELECT%20%3Fconcept%20%3Flabel%20WHERE%20%0A%7B%20%3Fconcept%20rdf%3Atype%20%3Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FCity%3E.%0A%20%3Fconcept%20rdfs%3Alabel%20%3Flabel.%0AFILTER(lang

Matches(lang(%3Flabel)%2C%20%22EN%22))%7D%0A%20LIMIT%2010&for_class=http://dbpedia.org/ontology/City&properties=http%3A%2F%2Fdbpedia.org%2Fontology%2Fcountry%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FpopulationTotal%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FareaCode

uri,name,area_code,population_total,country
http://dbpedia.org/resource/L%C3%A9ry,_Quebec,Léry, Quebec,450 and
579,2307,http://dbpedia.org/resource/Canada
http://dbpedia.org/resource/Alipur_Chatha,Alipur Chatha,,,http://dbpedia.org/resource/Pakistan
http://dbpedia.org/resource/Alipurduar,Alipurduar,03564,127342,http://dbpedia.org/resource/India
http://dbpedia.org/resource/Ambikapur,_India,Ambikapur, India,7774,214575,http://dbpedia.org/resource/India
http://dbpedia.org/resource/Ardal,Ardal,,8162,http://dbpedia.org/resource/Iran
http://dbpedia.org/resource/Arra,_India,Arra, India,,19911,http://dbpedia.org/resource/India
http://dbpedia.org/resource/Babai,_Madhya_Pradesh,Babai, Madhya
Pradesh,,14587,http://dbpedia.org/resource/India
http://dbpedia.org/resource/Badarpur,_Assam,Badarpur, Assam,,11291,http://dbpedia.org/resource/India
….
….

1.3 Convert to RDB

GET **/v1.0/convert/rdb-converter.sql**

1.3.1 Description

This method returns the RDF ResultSet in an upload .sql script file. There are two kinds of convert.
The first one is a generic convert where it puts all the returned rows to the table “things”. The other one is a class convert in which the convert is based on a particular class. For class convert it is essential to pass paramaters “for_class” which specifies for which class the convert is based on, and in the “query”, there should be column in the resultset “concept” which has objects of the class.

1.3.2 Input Parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
query	true	String	contains the SPARQL query as a string. The query should have an output variable “concept” if a class conversion is being called.

for_class	false	String	contains the uri of the class for which the conversion is done. This specifies class conversion needs to be done.
properties	false	String	comma separated uris of properties of the class which are required to be present in the final output. To get all the properties , “all” should be passed.

1.3.3 Example request

1.3.3.1 Generic convert

GET
http://localhost:8081/rdf2any/v1.0/convert/rdb-converter.sql?dataset=http://dbpedia.org/sparql&query=select+distinct+%3FConcept+where+%7B%5B%5D+a+%3FConcept%7D+LIMIT+100

```
DROP TABLE IF EXISTS things;

CREATE TABLE things
(
ID int,
subject varchar(1000),
label varchar(1000),
PRIMARY KEY ID
);


INSERT INTO rdf_table VALUES(1,'http://dbpedia.org/resource/2005_in_South_Korean_football','2005 in South Korean football@en');
INSERT INTO rdf_table VALUES(2,'http://dbpedia.org/resource/Category:2005_in_South_Korean_football','2005 in South Korean football@en');
INSERT INTO rdf_table VALUES(3,'http://wikidata.dbpedia.org/resource/Q4605218','2005 in South Korean football@en');
INSERT INTO rdf_table VALUES(4,'http://dbpedia.org/resource/2010_in_South_Korean_football','2010 in South Korean football@en');
INSERT INTO rdf_table VALUES(5,'http://dbpedia.org/resource/Category:2010_in_South_Korean_football','2010 in South Korean football@en');
INSERT INTO rdf_table VALUES(6,'http://wikidata.dbpedia.org/resource/Q4619094','2010 in South Korean football@en');
INSERT INTO rdf_table VALUES(7,'http://dbpedia.org/resource/List_of_Federal_Roads_in_Sarawak','List of Federal Roads in Sarawak@en');
INSERT INTO rdf_table VALUES(8,'http://wikidata.dbpedia.org/resource/Q6570751','List of Federal Roads in Sarawak@en');
INSERT INTO rdf_table VALUES(9,'http://dbpedia.org/resource/Wang_Jingyao','Wang Jingyao@en');
```

1.3.3.2 Class convert

GET
http://localhost:8081/rdf2any/v1.0/convert/rdb-converter.sql?dataset=http://dbpedia.org/sparql&query=PREFIX%20rdf%3A%3Chttp%3A%2F%2Fwww.w3.org%2F1999%2F02%2F22-rdf-syntax-ns%23%3E%20%0APREFIX%20rdfs%3A%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%20%0ASELECT%20%3Fconcept%20%3Flabel%20WHERE%20%0A%7B%20%3Fconcept%20rdf%3Atype%20%3Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FCity%3E.%0A%20%3Fconcept%20rdfs%3Alabel%20%3Flabel.%0AFILTER(langMatches(lang(%3Flabel)%2C%20%22EN%22))%7D%0A%20LIMIT%2010&for_class=http://dbpedia.org/ontology/City&properties=http%3A%2F%2Fdbpedia.org%2Fontology%2FleaderName%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2Fcountry%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2Fabstract%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FpopulationTotal%2Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FareaCode

```
-- START table creation scripts properties pointing to other classes


DROP TABLE IF EXISTS countrys CASCADE;
CREATE TABLE countrys
(
id int PRIMARY KEY,
uri varchar(300),
name text
);


DROP TABLE IF EXISTS persons CASCADE;
CREATE TABLE persons
(
id int PRIMARY KEY,
uri varchar(300),
name text
);


-- END table creation scripts properties pointing to other classes


-- START Table creation section for main class table


DROP TABLE IF EXISTS citys CASCADE;
CREATE TABLE citys
(
id int PRIMARY KEY,
uri varchar(300),
```

```
name text
);


-- END Table creation section for main class table


-- START table creation scripts for normalized property tables

DROP TABLE IF EXISTS cityhasabstracts CASCADE;
CREATE TABLE cityhasabstracts
(id int PRIMARY KEY,
city_id int,
hasabstract text,
hasabstractLang varchar(6)
);

DROP TABLE IF EXISTS cityareacodes CASCADE;
CREATE TABLE cityareacodes
(id int PRIMARY KEY,
city_id int,
areacode text
);

DROP TABLE IF EXISTS citypopulationtotals CASCADE;
CREATE TABLE citypopulationtotals
(id int PRIMARY KEY,
city_id int,
populationtotal int
);

DROP TABLE IF EXISTS citycountrys CASCADE;
CREATE TABLE citycountrys
(id int PRIMARY KEY,
city_id int,
country_id int REFERENCES countrys(id)
);

DROP TABLE IF EXISTS cityleadernames CASCADE;
CREATE TABLE cityleadernames
(id int PRIMARY KEY,
city_id int,
leadername_id int REFERENCES persons(id)
);


-- END table creation scripts for normalized property tables
```

```
--1. ########################################
INSERT INTO citys (ID, uri, name) VALUES (1, 'http://dbpedia.org/resource/L%C3%A9ry,_Quebec','Léry, Quebec');
INSERT INTO cityhasabstracts(id,city_id,hasabstract, hasabstractLang) VALUES(1, 1, 'Léry est une ville dans la municipalité régionale de comté de Roussillon au Québec (Canada), située dans la région administrative de la Montérégie.', 'FR');
INSERT INTO cityhasabstracts(id,city_id,hasabstract, hasabstractLang) VALUES(2, 1, 'Léry es una ciudad de la provincia de Quebec, Canadá. Es una de las ciudades que conforman la Comunidad metropolitana de Montréal y se encuentra en el condado regional de Roussillon y a su vez, en la región del Valle del Alto San Lorenzo en Montérégie. Hace parte de las circunscripciones electorales de Châteauguay a nivel provincial y de Châteauguay−Saint-Constant a nivel federal.', 'ES');
INSERT INTO cityhasabstracts(id,city_id,hasabstract, hasabstractLang) VALUES(3, 1, 'Léry is a small town situated along the south shore of Lake Saint-Louis in Quebec, Canada. The population as of the Canada 2011 Census was 2,307. Located on Route 132 west of Châteauguay and east of Beauharnois in the administrative region of Montérégie, the town is home to the Bellevue Golf Club, with its two 18-hole courses.', 'EN');
INSERT INTO cityareacodes(id,city_id,areacode) VALUES(1, 1, '450 and 579');
INSERT INTO countrys(id, uri, name) VALUES (1, 'http://dbpedia.org/resource/Canada', 'Canada');
INSERT INTO citycountrys(id,city_id,country_id) VALUES(1, 1, 1);
INSERT INTO persons(id, uri, name) VALUES (1,
'http://dbpedia.org/resource/Ch%C3%A2teauguay_(provincial_electoral_district)', 'Châteauguay (provincial electoral district)');
INSERT INTO cityleadernames(id,city_id,leadername_id) VALUES(1, 1, 1);
INSERT INTO persons(id, uri, name) VALUES (2,
'http://dbpedia.org/resource/Ch%C3%A2teauguay%E2%80%94Saint-Constant', 'Châteauguay—Saint-Constant');
INSERT INTO cityleadernames(id,city_id,leadername_id) VALUES(2, 1, 2);
INSERT INTO citypopulationtotals(id,city_id,populationtotal) VALUES(1, 1, 2307);

--2. ########################################
INSERT INTO citys (ID, uri, name) VALUES (2, 'http://dbpedia.org/resource/Alipur_Chatha','Alipur Chatha');
INSERT INTO cityhasabstracts(id,city_id,hasabstract, hasabstractLang) VALUES(4, 2, 'Alipur (en urdu: علی پور ,
también Alipur Chatta) es una localidad de Pakistán, en la provincia de Punyab.', 'ES');
INSERT INTO cityhasabstracts(id,city_id,hasabstract, hasabstractLang) VALUES(5, 2, 'Ali Pur Chattha (Urdu: علی پور
چٹھہ ) is a city and one of the 36 union councils of Wazirabad Tehsil of Gujranwala District in the Punjab province of Pakistan. It contains ruins of the historical city of Akālgarh.', 'EN');
INSERT INTO countrys(id, uri, name) VALUES (2, 'http://dbpedia.org/resource/Pakistan', 'Pakistan');
INSERT INTO citycountrys(id,city_id,country_id) VALUES(2, 2, 2);

…….
…….
```

1.4 Configured convert

GET **/v1.0/convert/configured-converter**

1.4.1 Description

This method is useful in having a generic type of serialization for class convert. For the configured convert. Four parts are essential

1) *variable_dictionary* : this will define the variables for the property uris. The variable definitions are comma separated.
   variable_name1::*property_uri1,variable_name2*::*property_uri2*
2) *head* : This will constitute the head of the serialization output. It will be printed once in the top and is static
3) *footer* : This will constitute the footer of the serialization output. It will be printed once in the end and is static.
4) *body* : This will constitute the body of the serialization output. This part will be looped with every object of the resultset. Three programmable items are supported in body.
   a) *print* : this is a simple print of a variable. variable names, URI and NAME are reserved for uri and name of the object. OBJECT_COUNTER returns a unique integer counter for the object.
      $[=*variable_name*]
      For language literals the language and text can be separately printed.
      $[=variable_name.text] gives the text and $[=variable_name.lang] returns the language.
   b) if condition : This checks whether the property has some particular property.
      $[if *property_variable_name*] *some body here* $[end]
   c) for each loop : This loops over the values of a particular property.
      $[for *property* : *property_variable_name*] *some body here* $[end]

To get a simple xml output of say

```
<elements>
    <element uri="http://some.example.uri.">
        <name>example</name>
        <abstracts>
            <abstract>this is the abstract 1</abstract>
            <abstract>this is the abstract 2</abstract>
        </abstracts>
    </element>
</elements>
```

the following values should be passed

1) variable_dictionary : abstracts::http://dbpedia.org/ontology/abstract
2) head : <elements>
3) body : <element uri="$[=URI]">\n<name>$[=NAME]</name>\n $[if abstracts]<abstracts>\n $[for abstract: abstracts] \n<abstract>$[=abstract]</abstract> $[end] </abstracts> $[end] \n</element>
4) footer : </elements>

In the above example we can notice that at first it checks if there are any abstracts for the object. If there aren't then it won't print the html tag <abstracts>. Then it loops into all the abstracts and prints them .
Using these straightforward programming items, most of the serializations like, XML, YAML, CSV , etc can be achieved.

1.4.2 Input Parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
query	true	String	contains the SPARQL query as a string. The query should have an output variable "concept" if a class conversion is being called.
for_class	true	String	contains the uri of the class for which the conversion is done. This specifies class conversion needs to be done.
properties	false	String	comma separated uris of properties of the class which are required to be present in the final output. To get all the properties , "all" should be passed.
variable_dictionary	false	String	this will define the variables for the property uris. The definitions are comma separated. *variable_name1*::*property_uri1,variable_name2*::*property_uri2*
header	false	String	This will constitute the head of the serialization output. It will be printed once and is static
body	true	String	This will constitute the body of the serialization output. This part will be looped with every object of the resultset
footer	false	String	This will constitute the footer of the serialization output. It will be printed once in the end and is static.

1.4.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/convert/configured-converter?dataset=http://dbpedia.org/sparql&query=PREFIX%20rdf%3A%3Chttp%3A%2F%2Fwww.w3.org%2F1999%2F02%2F22-rdf-syntax-ns%23%3E%20%0APREFIX%20rdfs%3A%3Chttp%3A%2F%2Fwww.w3.org%2F2000%2F01%2Frdf-schema%23%3E%20%0ASELECT%20%3Fconcept%20%3Flabel%20WHERE%20%0A%7B%20%3Fconcept%20rdf%3Atype%20%3Chttp%3A%2F%2Fdbpedia.org%2Fontology%2FCity%3E.%0A%20%3Fconcept%20rdfs%3Alabel%20%3Flabel.%0AFILTER(langMatches(lang(%3Flabel)%2C%20%22EN%22))%7D%0A%20LIMIT%2010&for_class=http://dbpedia.org/ontology/City&properties=http%3A%2F%2Fdbpedia.org%2Fontology%2Fabstract&variable_dictionary=abstracts%3A%3Ahttp%3A%2F%2Fdbpedia.org%2Fontology%2Fabstract&header=%3Celements%3E&footer=%3C%2Felements%3E&body=%3Celement+uri%3D%E2%80%9D%24%5B%3DURI%5D%E2%80%9D%3E%5Cn%3Cname%3E%24%5B%3DNAME%5D%3C%2Fname%3E%5Cn+%24%5Bif+abstracts%5D%3Cabstracts%3E%5Cn+%24%5Bfor+abstract%3A+abstracts%5D+%5Cn%3Cabstract%3E%24%5B%3Dabstract%5D%3C%2Fabstract%3E+%24%5Bend%5D+%3C%2Fabstracts%3E+%24%5Bend%5D+%5Cn%3C%2Felement%3E&oq=%3Celement+uri%3D%E2%80%9D%24%5B%3DURI%5D%E2%80%9D%3E%5Cn%3Cname%3E%24%5B%3DNAME%5D%3C%2Fname%3E%5Cn+%24%5Bif+abstracts%5D%3Cabstracts%3E%5Cn+%24%5Bfor+abstract%3A+abstracts%5D+%5Cn%3Cabstract%3E%24%5B%3Dabstract%5D%3C%2Fabstract%3E+%24%5Bend%5D+%3C%2Fabstracts%3E+%24%5Bend%5D+%5Cn%3C%2Felement%3E

<elements>
<element uri="http://dbpedia.org/resource/L%C3%A9ry,_Quebec">
<name>Léry, Quebec</name>
<abstracts>
<abstract>Léry est une ville dans la municipalité régionale de comté de Roussillon au Québec (Canada), située dans la région administrative de la Montérégie.@fr</abstract> <abstract>Léry es una ciudad de la provincia de Quebec, Canadá. Es una de las ciudades que conforman la Comunidad metropolitana de Montréal y se encuentra en el condado regional de Roussillon y a su vez, en la región del Valle del Alto San Lorenzo en Montérégie. Hace parte de las circunscripciones electorales de Châteauguay a nivel provincial y de Châteauguay−Saint-Constant a nivel federal.@es</abstract>
<abstract>Léry is a small town situated along the south shore of Lake Saint-Louis in Quebec, Canada. The population as of the Canada 2011 Census was 2,307. Located on Route 132 west of Châteauguay and east of Beauharnois in the administrative region of Montérégie, the town is home to the Bellevue Golf Club, with its two 18-hole courses.@en</abstract>
</abstracts>
</element>
<element uri="http://dbpedia.org/resource/Alipur_Chatha">
<name>Alipur Chatha</name>
<abstracts>
<abstract>Alipur (en urdu: علی پور , también Alipur Chatta) es una localidad de Pakistán, en la provincia de Punyab.@es</abstract> \n<abstract>Ali Pur Chattha (Urdu: علی پور چٹھہ) is a city and one of the 36 union councils of

Wazirabad Tehsil of Gujranwala District in the Punjab province of Pakistan. It contains ruins of the historical city of Akālgarh.@en</abstract> </abstracts>
</element>
….
….
</elements>

2. Builder API

These APIs help in query builder actions. They have APIs to search for classes, objects and retrieve properties of a particular class.

2.1 Class search

GET **/v1.0/builder/classes**

2.1.1 Description

This API returns a list of classes in a dataset matching a search string.

2.1.2 Input Parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
search	true	String	search string which will match the class
force_uri_search	false	Boolean	if set true, then it forces to search for string in the uri patterns of classes which do not have any labels. Set this to true only if you want to search for classes which do not probably have any labels. By default its false.

2.1.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/classes?dataset=http://dbpedia.org/sparql&search=anim


```
{
   "dataset": "http://dbpedia.org/sparql",
   "search_string": "anim",
   "searched_items": [
      {
```

```
    "labels": {
      "it": "animale",
      "pt": "animal",
      "fr": "animal",
      "en": "animal"
    },
    "sequence": 1,
    "uri": "http://dbpedia.org/ontology/Animal"
  },
  {
    "labels": {
      "it": "personaggio animanga",
      "en": "animanga character"
    },
    "sequence": 2,
    "uri": "http://dbpedia.org/ontology/AnimangaCharacter"
  },
  {
    "labels": {
      "de": "anime",
      "it": "anime",
      "en": "anime"
    },
    "sequence": 3,
    "uri": "http://dbpedia.org/ontology/Anime"
  },
  {
    "labels": {
      "it": "cartone animato",
      "fr": "dessin animé"
    },
    "sequence": 4,
    "uri": "http://dbpedia.org/ontology/Cartoon"
  },
  {
    "labels": {
      "fr": "animateur de télévision"
    },
    "sequence": 5,
    "uri": "http://dbpedia.org/ontology/TelevisionHost"
  },
  {
    "labels": {
      "en": "animacy attribute"
    },
    "sequence": 6,
    "uri": "http://www.ontologyportal.org/SUMO#AnimacyAttribute"
  },
  {
    "labels": {
```

```
            "en": "vanimo group language"
          },
          "sequence": 7,
          "uri": "http://www.ontologyportal.org/SUMO#VanimoGroupLanguage"
        }
    ]
}
```


DEPRECATED

GET

http://localhost:8081/rdf2any/v0.9/builder/classes?dataset=http://dbpedia.org/sparql&search=anim


```
{
    "head": {
        "vars": [
            "class",
            "label"
        ],
        "link": []
    },
    "results": {
        "bindings": [
            {
                "class": {
                    "value": "http://dbpedia.org/ontology/Animal",
                    "type": "uri"
                },
                "label": {
                    "value": "animal",
                    "xml:lang": "en",
                    "type": "literal"
                }
            },
            {
                "class": {
                    "value": "http://dbpedia.org/ontology/AnimangaCharacter",
                    "type": "uri"
                },
                "label": {
                    "value": "animanga character",
                    "xml:lang": "en",
                    "type": "literal"
                }
            },
            {
                "class": {
                    "value": "http://dbpedia.org/ontology/Anime",
```

```
                "type": "uri"
            },
            "label": {
                "value": "anime",
                "xml:lang": "en",
                "type": "literal"
            }
        },
        ....
       .....
    ],
    "distict": false,
    "ordered": true,
    "time_taken": 3.456
  }
}
```


2.2 Objects search

GET **/v1.0/builder/objects**

2.2.1 Description

This API returns a list of objects matching the search string. Objects of a particular triple rule can also be searched, then it will return list of objects of a particular class, and property.

2.2.3 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
search	true	String	search string which will match the object
classes	false	String	Will contain comma separated classes. Will search for objects of these classes. This is required if the variables *for_class* and *for_property* are not passed
for_class	false	String	uri of the class needs to be passed here. This is required if we need to search for objects of a particular class and its property
for_property	false	String	uri of the property needs to be passed here. This is required if we need to

			search for objects of a particular class and its property

2.2.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/objects?search=germ&dataset=http%3A%2F%2Fdbpedia.org%2Fsparql&classes=http%3A%2F%2Fdbpedia.org%2Fontology%2FCountry&for_class=http%3A%2F%2Fdbpedia.org%2Fontology%2FCity&for_property=http%3A%2F%2Fdbpedia.org%2Fontology%2Fcountry


```
{
   "head": {
      "vars": [
         "object",
         "label"
      ],
      "link": []
   },
   "results": {
      "bindings": [
         {
            "label": {
               "value": "Germany",
               "xml:lang": "en",
               "type": "literal"
            },
            "object": {
               "value": "http://dbpedia.org/resource/Germany",
               "type": "uri"
            }
         }
      ],
      "distict": false,
      "ordered": true,
      "time_taken": 7.532
   }
}
```


2.3 Class properties

GET **/v1.0/convert/properties**

2.3.1 Description

This API returns the properties of a particular class. The properties have types “data” for DataType range properties and “object” for ObjectType range properties.

2.3.2 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
class	true	String	uri of the class for which properties are to be retrieved

2.3.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/properties?dataset=http://dbpedia.org/sparql&class=http%3A%2F%2Fdbpedia.org%2Fontology%2FCity


```
{
   "rdfClass": {
      "dataset": "http://dbpedia.org/sparql",
      "indexCreated": true,
      "label": "city",
      “uri” : “http://dbpedia.org/ontology/City",
      "properties": [
         {
            "count": 795,
            "label": "synonym",
            "multiplePropertiesForSameNode": true,
            "range": {
               "label": "string",
               "uri": "http://www.w3.org/2001/XMLSchema#string"
            },
            "type": "data",
            "uri": "http://dbpedia.org/ontology/synonym"
         },
         {
            "count": 136526,
            "label": "has abstract",
            "multiplePropertiesForSameNode": true,
            "range": {
               "label": "langString",
               "uri": "http://www.w3.org/1999/02/22-rdf-syntax-ns#langString"
            },
            "type": "data",
            "uri": "http://dbpedia.org/ontology/abstract"
         },
```

.....
....
        ]
    }
}

2.4 Class subclasses

GET **/v1.0/convert/classes/subclasses**

2.4.1 Description

This API returns the subclasses of a particular class. The subclasses are defined by the property rdfs:subclassOf using RDFS schema

2.4.2 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
class	true	String	uri of the class for which subclasses are to be retrieved

2.4.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/classes/subclasses?dataset=http://dbpedia.org/sparql&class=http://dbpedia.org/ontology/Animal


```
{
   "dataset": "http://dbpedia.org/sparql",
   "label": "animal",
   "subclasses": [
      {
         "label": "amphibian",
         "uri": "http://dbpedia.org/ontology/Amphibian"
      },
      {
         "label": "arachnid",
         "uri": "http://dbpedia.org/ontology/Arachnid"
      },
      {
         "label": "bird",
```

```
            "uri": "http://dbpedia.org/ontology/Bird"
        },
        {
            "label": "crustacean",
            "uri": "http://dbpedia.org/ontology/Crustacean"
        },
      .....
    ],
    "uri": "http://dbpedia.org/ontology/Animal"
}
```


2.5 Class examples

GET **/v1.0/convert/classes/examples**

2.5.1 Description

This API returns example objects of a particular class. It also returns the total objects of that particular class

2.5.2 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset
class	true	String	uri of the class for which example objects are to be retrieved
limit	false	Integer	Limit of the no. of example objects to be retrieved. Defaults to 5

2.5.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/classes/examples?dataset=http://dbpedia.org/sparql&class=http://dbpedia.org/ontology/Actor


```
{
    "dataset": "http://dbpedia.org/sparql",
    "label": "actor",
    "sample_objects": [
        {
            "label": "Alex Reid (actress)",
            "uri": "http://dbpedia.org/resource/Alex_Reid_(actress)"
```

```
      },
      {
         "label": "Henri Cogan",
         "uri": "http://dbpedia.org/resource/Henri_Cogan"
      },
      {
         "label": "Åke Fridell",
         "uri": "http://dbpedia.org/resource/%C3%85ke_Fridell"
      },
      {
         "label": "Aaron Lawrence (entrepreneur)",
         "uri": "http://dbpedia.org/resource/Aaron_Lawrence_(entrepreneur)"
      },
      {
         "label": "Alexis Conran",
         "uri": "http://dbpedia.org/resource/Alexis_Conran"
      }
   ],
   "total_objects": 6501,
   "uri": "http://dbpedia.org/ontology/Actor"
}
```


3. Administrative API

Contains APIs for administrative purposes, like creation of indexes, etc.

3.1 Property index creation

GET **/v1.0/builder/properties/indexes/create**

3.1.1 Description

This API is called to start the index creation of properties of classes of a particular dataset.

3.1.2 Input parameters

variable	required	data type	description
dataset	true	String	SPARQL endpoint of the dataset. Indexes for properties of classes will be created for this dataset

3.1.3 Example request

GET
http://localhost:8081/rdf2any/v1.0/builder/properties/indexes/create?dataset=http://dbpedia.org/sparql