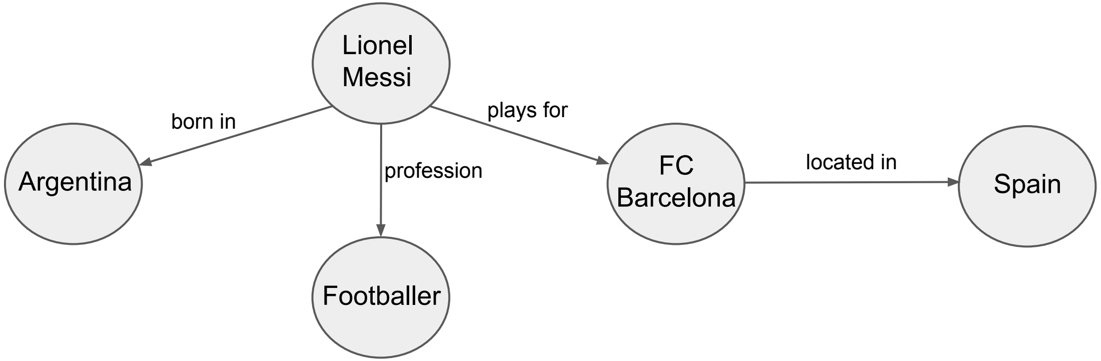
RDF and SPARQL

Resource Description Framework (RDF)

- Flexible domain independent data model for information representation.
- Resources: Objects (locations, persons, organizations, ...)
- Properties: Linking objects (located in, written by, ...)
- Literals: Atomic values (strings, dates, integers)
- Statements: Resource-property-resource or subject-property-object triples.
- “*Lionel Messi is an Argentine professional footballer who plays for Spanish club FC Barcelona.*” ⇒ Written as directed RDF graph:



RDF syntax: Terse RDF Triple Language (Turtle)

- Common machine interpretable syntax.
- File extension: .ttl
- Resources and properties are uniquely identified by URIs.
- “*Lionel Messi*” ⇒ in DBpedia: <http://dbpedia.org/resource/Lionel_Messi>
- Birth date (property) ⇒ in DBpedia: <http://dbpedia.org/ontology/birthDate>
- SPO triple (always ends with “.”):

```
<http://dbpedia.org/resource/Lionel_Messi>
              <http://dbpedia.org/property/birthPlace>
              <http://dbpedia.org/resource/Argentina> .
```

Turtle

- Prefixes: Abbreviations for URIs (at the beginning of a ttl file).

```
@prefix dbpedia: <http://dbpedia.org/resource/>.
@prefix dbpedia-property: <http://dbpedia.org/property/>.
@prefix dbpedia-owl: <http://dbpedia.org/ontology/>
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
```

```
dbpedia:Lionel_Messi dbpedia-property:birthPlace dbpedia:Argentina .
dbpedia:Lionel_Messi dbpedia-owl:birthDate "1987-06-24"^^xsd:date.
```

- Literals: Strings (potentially with language tag, e.g. “<string>”@en), integers, decimals, dates, times, dates with time.

SPARQL

- **S**PARQL **P**rotocol **A**nd **R**DF **Q**uery **L**anguage
- RDF query language for triple stores (Apache Jena, Virtuoso, RDF-3X, ...) with SQL-like syntax.
- Variables start with a “?” followed by a name.
- Example:

```
PREFIX dbr:  <http://dbpedia.org/resource/>
PREFIX dbo:  <http://dbpedia.org/ontology/>
SELECT ?place
WHERE {
        dbr:Lionel_Messi dbo:birthPlace ?place
}
```

- More keywords: LIMIT, DISTINCT, COUNT, GROUP BY, UNION, ...

Querying DBPedia (with YASGUI)

Go to http://yasgui.org to try out the examples

Note that YASGUI helps you with prefixes as you type them.

Common DBPedia prefixes that YASGUI supports with auto-complete:

- dbo: Ontology (e.g., classes like dbo:City, dbo:Person, …)
- dbr: Resource (e.g., instances like dbr:Edmonton or dbr:Neymar)
- dbp: Property (e.g., dbp:name)

Other common prefixes/ontologies you should know

schema: → refers to ontology concepts in http://schema.org . Check it out now!

ex: schema:City, schema:Airport, …
Should you use schema: or dbo: ? It is really up to you!

rdf: → refers to the standard RDF vocabulary

The most useful one is rdf:type which assigns a class to the object
ex: ?c rdf:type schema:City makes ?c bind to a city in DBPedia

Where is the FROM clause?

How does YASGUI know to query dbpedia?

```
PREFIX dbr:  <http://dbpedia.org/resource/>
PREFIX dbo:  <http://dbpedia.org/ontology/>
SELECT ?place
WHERE {
        dbr:Lionel_Messi dbo:birthPlace ?place
}
```

In SPARQL, the query always runs on the default RDF graph of the endpoint

You can specify the exact graph with a FROM command though

Your turn

Find all cities in Alberta (dbr:Alberta)

Cities in Alberta

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT * WHERE {
  ?city rdf:type schema:City .
  ?city dbo:isPartOf dbr:Alberta
}
```

ALWAYS restrict the type of what you are looking for!

(unless you are looking for "things" of different kinds)

How can we know the right predicate to use?

How to find which predicate to use?

There is no other way…

Explore the data!

If you want something about a city in dbr:Alberta:

- Check the DBPedia page of one such city, like dbr:Edmonton
- Check the page dbr:Alberta!!!

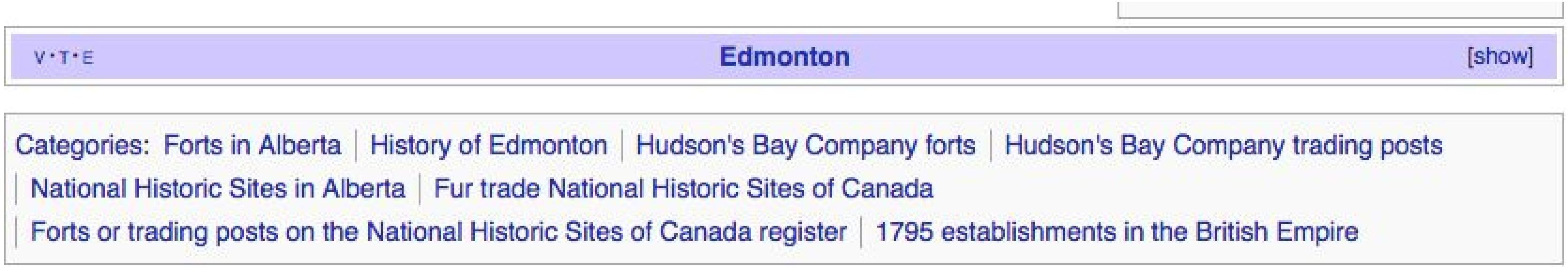
What does dbr:Edmonton and dbr:Alberta stand for?

Your turn

Find all national historic sites in Alberta

Note: because DBPedia comes from Wikipedia all category pages are also part of DBPedia

https://en.wikipedia.org/wiki/List_of_National_Historic_Sites_of_Canada_in_Alberta



V·T·E Edmonton [show]

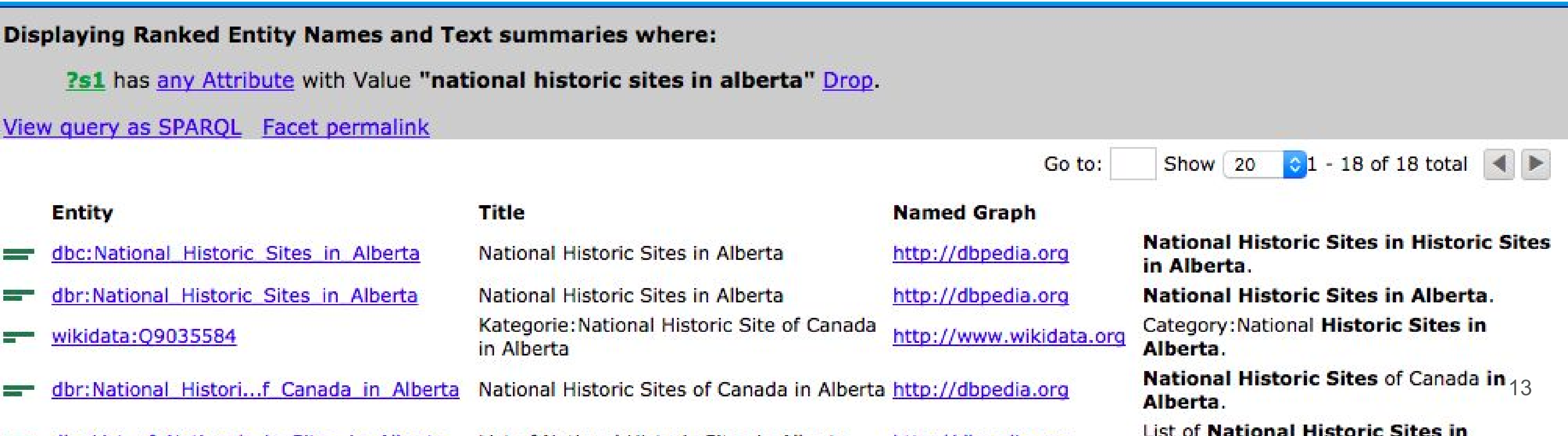
Categories: Forts in Alberta | History of Edmonton | Hudson's Bay Company forts | Hudson's Bay Company trading posts | National Historic Sites in Alberta | Fur trade National Historic Sites of Canada | Forts or trading posts on the National Historic Sites of Canada register | 1795 establishments in the British Empire

Finding category pages in DBPedia

Head to http://wiki.dbpedia.org/OnlineAccess

Find the “Public Faceted Web Service Interface”

Search for the category you are looking for (or anything else)



Displaying Ranked Entity Names and Text summaries where:

?s1 has any Attribute with Value **"national historic sites in alberta"** Drop.

View query as SPARQL Facet permalink

Go to: Show 20 1 - 18 of 18 total

Entity	Title	Named Graph	
dbc:National_Historic_Sites_in_Alberta	National Historic Sites in Alberta	http://dbpedia.org	**National Historic Sites in Historic Sites in Alberta**.
dbr:National_Historic_Sites_in_Alberta	National Historic Sites in Alberta	http://dbpedia.org	**National Historic Sites in Alberta**.
wikidata:Q9035584	Kategorie:National Historic Site of Canada in Alberta	http://www.wikidata.org	Category:National **Historic Sites in Alberta**.
dbr:National_Histori...f_Canada_in_Alberta	National Historic Sites of Canada in Alberta	http://dbpedia.org	**National Historic Sites** of Canada **in Alberta**.
			List of **National Historic Sites in**

Back to your turn

Find all national historic sites in Alberta

All national historic sites in Alberta

```
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/>
PREFIX dbc: <http://dbpedia.org/resource/Category:>
SELECT * WHERE {
  ?site dct:subject dbc:National_Historic_Sites_in_Alberta .
}
```

dct:subject is a very common way of finding things

Your turn

All national historic sites in Alberta which are forts

```
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/>
PREFIX dbc: <http://dbpedia.org/resource/Category:>
SELECT * WHERE {
  ?site dct:subject dbc:National_Historic_Sites_in_Alberta .
  ?site dct:subject dbc:Forts_in_Alberta .
}
```

Filters in SPARQL

- Filters: Remove results that do not match the filter.

  ```
  PREFIX dbr:  <http://dbpedia.org/resource/>
  PREFIX dbo:  <http://dbpedia.org/ontology/>
  SELECT ?player ?number
  WHERE {
          ?player dbo:number ?number .
          ?player dbo:team dbr:FC_Barcelona .
          FILTER (?number = "10") .
  }
  ```
- ≥, ≤, = are supported for numerical data types and date/time.
- regex() for using regular expressions (e.g. FILTER (regex(?team,"Barcelona"))).
- str() for converting numerical data types into strings that can be used within regex().

Your turn

Find cities in Alberta with a population over 100,000

Find neighborhoods in Edmonton with a population over 5,000

More SPARQL

- Union and Optional:

```
SELECT ?player ?number
WHERE {
        {?player dbo:team dbr:FC_Bayern_Munich . }
        UNION
        {?player dbo:team dbr:FC_Barcelona . }
        OPTIONAL
        {?player dbo:number ?number . }
}
```

- Return results for the optional graph pattern only if available (if not the ?number is empty).

Negation

Specify constraints that the answer should not have

```
PREFIX dbr:  <http://dbpedia.org/resource/>
PREFIX dbo:  <http://dbpedia.org/ontology/>
SELECT ?player ?number
WHERE {
        ?player dbo:number ?number .
        ?player dbo:team dbr:FC_Barcelona .
        FILTER (?number = "10") .
        MINUS {?player dbo:birthPlace ?city .
                  ?city dbo:country dbr:Argentina}
}
```

Exclude players born in a city in Argentina

Aggregation

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?city (COUNT(?hospital) as ?cnt) WHERE {
  ?city rdf:type schema:City .
  ?city dbo:isPartOf dbr:Alberta .
  ?hospital dbo:region ?city .
  ?hospital rdf:type schema:Hospital
}
```

Check the documentation for the other functions!

Nested queries

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?bigCity ?population WHERE {
    ?bigCity dbo:populationTotal ?population .
    ?bigCity rdf:type schema:City .
    {
            SELECT (MAX(?population2) as ?max) WHERE {
                    ?city rdf:type schema:City .
                    ?city dbo:isPartOf dbr:Alberta .
                    ?city dbo:populationTotal ?population2
            }
   } .
   FILTER (?population > 10 * ?max)
}
```

Highest population of
a city in Alberta

Evaluation of nested queries

SPARQL evaluates the sub-queries first, and keeps their results

When executing the outer query, an answer is produced if we can find triples in the graph that match all predicates in the outer query and whose variable assignments agree with the shared variables in the sub-query