SPARQL

Daniel Reinholdt

Trondheim 30.09.16

Oversikt

Hva er SPARQL?

- SPARQL er et spørrespråk (query language) som kan brukes til å hente og manipulere data som er lagret i RDF format.

Fordeler med SPARQL

- Spørringer gir muligheten til å hente ut spesifikk informasjon fra RDF grafer.
- Man kan gjøre spørringer mot eksterne RDF servere og få sanntids resultater tilbake.
- Man kan sette opp automatiserte spørringer mot RDF datasett som genererer rapporter.
- Det åpner for applikasjonsutvikling på et høyere nivå fordi man kan arbeide med SPARQL spørringsresultater og ikke bare RDF enkeldata.

Joseki endpoint

- Joseki er et webbasert SPARQL endepunkt med et brukergrensesnitt som kan brukes til å skrive inn spørringer mot en RDF datalagringsserver. Mer spesifikt så har Joseki endepunktet egen web server, som verter en java servlet.
- Link: `http://www.sparql.org/sparql.html`

FOAF

- *Friend of a Friend* forkortet FOAF dokument er et prosjekt som ble satt i gang av Dan Brickley for å lage et sosialt nettverk ved å bruke semantic web teknologi. Dokumentet er ment for å beskrive en person dvs. Navn, nettside, email, venner, interesser osv.

tripple pattern and graph pattern

- Alle SPARQL spørringsformene er basert på to grunnleggende prinsipper: tripple pattern og graph pattern
- På samme måte som RDF er bygd på tripple pattern så er SPARQL bygd på tripple pattern som også er skrevet som subject, predicate og object.
- SPARQL kan inkludere variabler, noen eller alle subjectene, predicatene og object verdiene i SPARQL kan være variabel.
- Eksempel: <http://danbri.org/foaf.rdf#danbri>foaf:name ?name. Subjectet er Dan birckley URI, predicate er foaf:name, og objektet er ?name. Name er i dette tilfelle en variabel. Variabler kan ha prefikset ? Eller $.

tripple pattern and graph pattern fortsetter

- Graph pattern er når man har en sammling av flere tripplepattern i en spørring. Eksempel på tripple pattern:
  {
  ?who foaf:name ?name.
  ?who foaf:interest ?interest.
  ?who foaf:knows ?others.
  }
- Graph pattern forsøker å finne en ressurs som innholder disse 3 egenskapene, søkeprossesen stopper og går videre til en ny ressurs dersom den nåværende ressursen ikke har alle de nødvendige egenskapene definert. I graph pattern så må en variable alltid være bundet til samme verdi uansett hvor den dukker opp.

SPARQL spørring

- SELECT query
- CONSTRUCT query
- DESCRIBE query
- ASK query

Select Eksempel

```
1: base <http://danbri.org/foaf.rdf>
2: prefix foaf: <http://xmlns.com/foaf/0.1/>
3: select *
4: from <http://danbri.org/foaf.rdf>
5: where
6: {
7:   <#danbri> ?property ?value.
8: }
```

Avansert Select Eksempel

```
base <http://danbri.org/foaf.rdf> prefix foaf: <http://xmlns.com/foaf/0.1/>
select * from <http://danbri.org/foaf.rdf>
where
{
        <#danbri> foaf:knows ?friend.
        ?friend foaf:name ?name.
        ?friend foaf:mbox ?email.
        ?friend foaf:homepage ?homepage.
}
```

Select med funksjoner

```
base <http://danbri.org/foaf.rdf> prefix foaf: <http://xmlns.com/foaf/0.1/>
select * from <http://danbri.org/foaf.rdf>
where
{
        <#danbri> foaf:knows ?friend.
        optional {?friend foaf:name ?name.}
        optional {?friend foaf:mbox ?email. }
        optional {?friend foaf:homepage ?homepage.}
}
```

- Optional finner også de vennene som ikke har data på alle punktene i graph pattern (name, email, homepage).
- Andre funksjoner: distinct, order by, limit, Filter med regex.

Construct Eksempel

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
construct {
?person a foaf:Person.
?person foaf:name ?name.
?person foaf:mbox ?email.
}
from <http://danbri.org/foaf.rdf>

where
{
        ?person a foaf:Person.
        ?person foaf:name ?name.
        ?person foaf:mbox ?email.
}
```

- Construct returnerer en ny RDF graph. Brukes for å kunne transformere en gitt graph til en ny graph med annen ontology. F.eks FOAF til vCard.

Describe Eksempel

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
describe ?x
from <http://danbri.org/foaf.rdf>
where
{
   ?x foaf:mbox <mailto:timbl@w3.org>.
}
```

- Describe query brukes for å få en beskrivelse av data graph'en vi skal gjøre spørringer mot.

Ask Eksempel

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
ask
from <http://danbri.org/foaf.rdf>
where
{
        ?x foaf:mbox <mailto:danbri@danbri.org>.
}
```

- Ask brukes for å kunne få ut en boolean verdi, altså gjøre en spørring som enten returnerer True eller False.

SPARQL 1.1

- **Aggregate function:** gir muligheten til å jobbe med resultat tabell, dette innebærer at man kan telle resultatene (Count), hente ut gjennomsnitt fra resultatene (AVG), finne maks og minimum (MIN/MAX) osv.
- **Negation:** finne alle verdier som ikke har en gitt verdi (NOT EXISTS)
- **Expressions in SELECT:** gir muligheten til å sette inn et uttrykk i SELECT setningen (subqueries). F.eks SELECT fn:concat(?first, , ?last) AS ?name
- **Update:** gir muligheten til å oppdatere, sette inn eller fjerne et utsagn i RDF.
- **Load:** gir muligheten til å kopiere alle tripplene (utsagnene) fra en graph til en annen gitt graph.
- **Clear:** gir muligheten til å slette alle tripples (utsagn) fra en gitt graph.

Oppsummering

- SPARQL er et query language som brukes for å hente ut data fra RDF dokumenter.
- SPARQL bygd på tripple pattern som er skrevet som subject, predicate og object. Likt som RDF.
- Nøkkel ord som finnes i SPARQL: Select, construct, describe og ASK. Select er det mest brukte.
- SPARQL 1.1 gir ny funksjonalitet: Aggregate function, Negation, Expressions in SELECT, Update, Load, Clear.