RDF baze podataka

Čerkez, Katarina

Master's thesis / Diplomski rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:532654

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Katarina Čerkez

RDF baze podataka

DIPLOMSKI RAD

Varaždin, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Katarina Čerkez

Matični broj: 0016123033

Studij: Organizacija poslovnih sustava

RDF baze podataka

DIPLOMSKI RAD

Mentorica:

Prof. dr. sc. Sandra Lovrenčić

Varaždin, rujan 2022.

Katarina Čerkez

Izjava o izvornosti

Izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autorica potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

U ovom radu objašnjen je pojam RDF-a (engl. *Resource Description Framework*) i temeljni koncepti istog koji uključuju resurse, svojstva i izjave. Zatim su navedene i objašnjene RDF serijalizacije s primjerima, kao i vrste spremnika, RDF kolekcije te je objašnjena RDF Schema čiji su najvažniji koncepti klase i svojstva. Svi ovi pojmovi potrebni su kako bi se izgradila RDF baza podataka, koja je u suštini grafovska baza podataka, stoga su objašnjeni pojmovi RDF baze podataka, grafovskih baza podataka te NoSQL baza podataka općenito, kojima grafovske baze podataka i pripadaju.

Nakon toga navedena su i opisana četiri sustava za upravljanje RDF bazama podataka koji spadaju među najpopularnije i najčešće korištene. Načinjena je usporedba tih sustava po određenim karakteristikama te je izrađena baza podataka s istom domenom u sva četiri sustava kako bi se demonstrirao rad u opisanim alatima i uvidjele razlike među njima.

Ključne riječi: RDF; RDF baze podataka; RDF Schema; trojke; SPARQL; resursi; klase; svojstva; sustavi za upravljanje RDF bazama podataka

Sadržaj

1. Uvod

Tema ovog rada su RDF baze podataka koje se temelje na RDF-u, odnosno Resource Description Frameworku. RDF okvir je za opis podataka o web resursima dizajniran da bude čitljiv i razumljiv računalu i pisan je u XML-u, a podaci i metapodaci opisuju se pomoću takozvanih trojki (engl. *triplestore*) koje se sastoje od subjekta, predikata i objekta kao jedinica prikaza informacija. Trojke se mogu prikazati kao graf, a grafovske baze podataka podvrsta su NoSQL baza podataka.

U ovom radu prvo će biti opisan sami RDF, što uključuje povijest RDF-a, temeljne koncepte, RDF serijalizacije, RDF kolekcije i RDF shemu. Zatim će biti riječi o samim RDF bazama podataka, odnosno o modelu podataka i SPARQL-u kao upitnom jeziku, o grafovskim i NoSQL bazama podataka te o najpopularnijim sustavima za upravljanje RDF bazama podataka.

Posljednji dio rada fokusirat će se na izradu iste RDF baze podataka u najpopularnijim sustavima i usporedbu najpopularnijih sustava za izradu RDF baza podataka s obzirom na određene karakteristike.

2. RDF

RDF (*Resource Description Framework*) nastao je 1999. godine kao standard za kodiranje metapodataka usko vezan s XML—om. Drugim riječima, za kodiranje podataka o podacima. Metapodaci se odnose na informacije koje su na neki način sekundarne nekom drugom sadržaju koji već postoji na regularnom webu, npr. informacije o tome tko je autor web stranice, kojeg datuma je objavljeno nešto i sl. Nakon što su 2004. ažurirani standardi i specifikacije RDF-a, on je postao znatno proširen i značajan. Više se ne koristi samo za kodiranje informacija o web resursima, već za informacije o vezama između stvari i pojava u stvarnome svijetu – ljudi, mjesta, koncepata itd. [1]

2.1. Općenito o RDF-u

RDF je preporuka W3C-a (*World Wide Web Consortium*) od 2004. godine i to je okvir za opis podataka o web resursima dizajniran da bude čitljiv i razumljiv računalu i pisan je u XML-u. S obzirom da koristi XML, RDF može razmjenjivati informacije između različitih vrsta računala koristeći različite vrste operacijskih sustava i aplikacijskih jezika. [2]

Kao što je već spomenuto, RDF služi za kodiranje informacija o resursima, a resursi mogu biti bilo što – dokumenti, ljudi, fizički objekti, apstraktni koncepti. RDF namijenjen je situacijama u kojima informacije na Webu trebaju biti procesuirane od strane aplikacija, a ne samo prikazivane ljudima. Informacije se mogu razmjenjivati između različitih aplikacija, tako da određenu informaciju ne mora koristiti samo aplikacija za koju je informacija originalno kreirana, već je mogu čitati i koristiti i ostale aplikacije. [3]

Upotreba RDF-a raznolika je, a neke od njih su [3]:

- Dodavanje strojno čitljivih informacija web stranicama koje omogućuje bolji prikaz na tražilicama
- Obogaćivanje nekog skupa podataka njegovim povezivanjem sa drugim skupovima podataka
- Međusobno povezivanje API *feedova* što klijentima olakšava pristup više informacija
- Izgradnja distribuiranih društvenih mreža povezivanjem RDF opisa ljudi na više web stranica
- Međusobno povezivanje skupova podataka unutar organizacije

Dakle, cilj RDF-a je omogućiti razmjenu podataka na Webu uz očuvanje njihovog originalnog značenja te omogućiti daljnju obradu i rekombinaciju informacija.

RDF može se promatrati i prikazati kao označeni usmjereni graf relacija između resursa i njihovih vrijednosti, o čemu će više riječi biti kasnije.

2.2. Temeljni koncepti

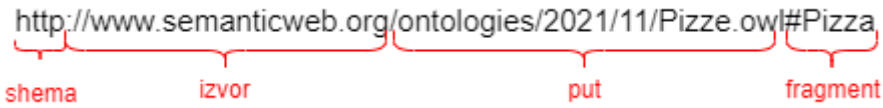
Temeljni koncepti RDF-a su resursi, svojstva i izjave, od čega se i sastoji RDF model podataka.

2.2.1. Resursi

Resurs predstavlja objekt kojeg opisujemo, a to može biti bilo što – ljudi, knjige, web stranica, riječ na web stranici, dokumenti, fizički objekti, apstraktni koncepti itd. Svaki resurs ima svoj jedinstveni identifikator na način da je imenovan IRI-jem (eng. *International Resource Identifier*). IRI kao identifikator ima svoje prednosti od kojih je najvažnija ta da služi kao globalna jedinstvena shema za imenovanje što razmjenu podataka čini jednostavnijom. Resurs imenovan IRI-jem naziva se referent ili označitelj. [3]

Postoji konstrukcijska shema za IRI koja se sastoji od sheme, izvora, puta, upita i fragmenta. Shema je ime IRI sheme koja označava tip IRI-ja (http, ftp itd.), izvor je obično ime domene te je izborni dio, put je glavni dio IRI-ja, može biti prazan ili organiziran hijerarhijski, upit je još jedan izborni dio koji osigurava dodatne informacije, obično parametre, a fragment je izborni dio koji omogućava drugu razinu identifikacije resursa. [3]

Na sljedećoj slici može se vidjeti primjer IRI-ja koji se sastoji od sheme, izvora, puta i fragmenta.



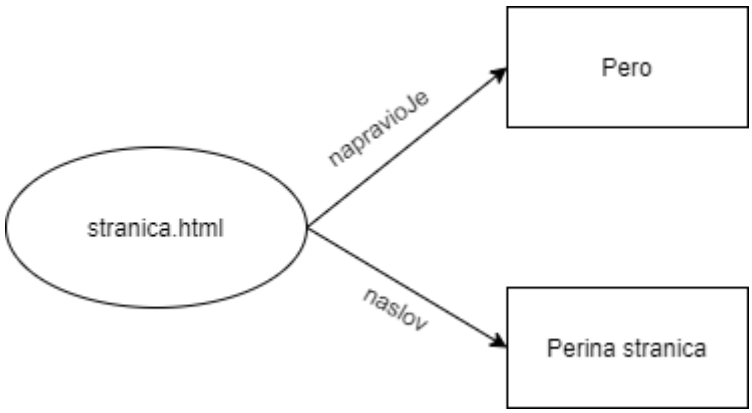
Slika 1 Primjer IRI-ja

2.2.2. Svojstva

Svojstva su zapravo specijalna vrsta resursa koja su također identificirana IRI-jima, a služe da opišu specifičnu karakteristiku, aspekt, atribut ili relaciju među resursima .[3]

2.2.3. Izjave

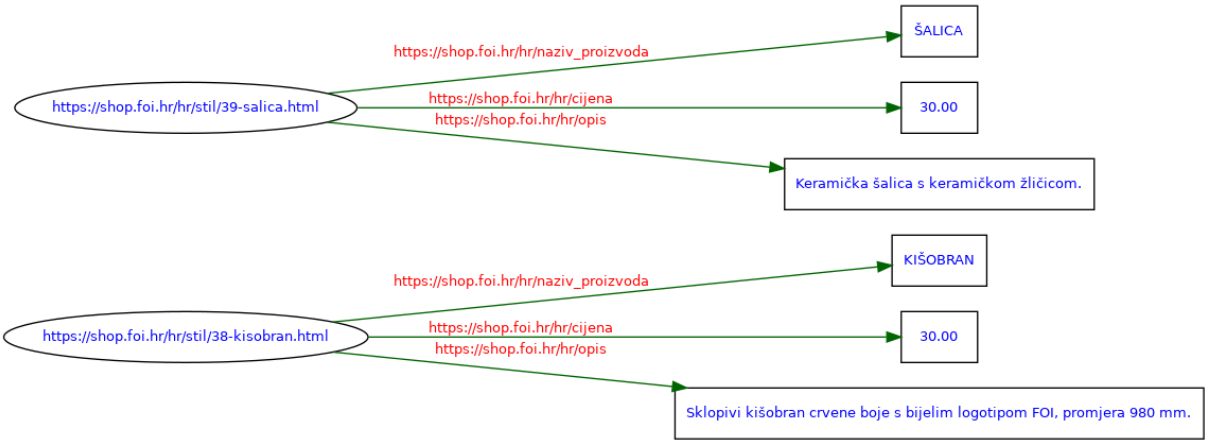
Posljednji temeljni koncept RDF-a su izjave, koje se koriste za opisivanje svojstava resursa. Izjava se sastoji od subjekta (resurs), predikata (svojstvo) i objekta (vrijednost svojstva; može biti resurs). RDF model podataka sastoji se od tih izjava koje se još nazivaju trojke (engl. *triple*), a može se prikazati pomoću grafa. Recimo da je Pero napravio html stranicu s naslovom „Perina stranica“. RDF model te izjave prikazan pomoću grafa izgledao bi kao na slici ispod. [4]



Slika 2 Primjer grafa RDF modela

Ovalni oblici na grafu prikazuju resurs (subjekt), strelice prikazuju svojstva (predikat), a vrijednosti svojstava (objekti) prikazuju se pravokutnicima. [4]

Na sljedećoj slici može se vidjeti još jedan primjer grafa RDF modela podataka koji sadrži dva proizvoda s FOI shopa kao resurse sa svojim svojstvima i njihovim vrijednostima.



Slika 3 Graf RDF modela podataka za proizvode FOI shopa

Vidimo da su resursi šalica i kišobran, te da imaju naziv proizvoda, cijenu i opis kao svojstva s pripadajućim vrijednostima.

2.3. RDF serijalizacije

Iako su grafovi lako čitljiv i jednostavan način prikaza informacija, nisu prikladni i lako čitljivi za računala koji nisu sposobni interpretirati graf kao što to mogu ljudi. Stoga je potrebno transformirati složene strukture podataka i informacije o trojkama u datoteke, odnosno napraviti serijalizaciju. Nekoliko je formata za serijalizaciju koji će biti objašnjeni u nastavku. [5]

2.3.1. Turtle obitelj

Turtle (*Terse RDF Triple Language*) je sintaksa, odnosno format serijalizacije koji obuhvaća N-Triples, Turtle, TriG i N-Quads. te omogućuje zapis RDF grafa u tekstualnom obliku. [3]

2.3.1.1. N-Triples

N-Triples najjednostavnija je RDF serijalizacija. Svaka trojka zapisana je u posebnoj liniji, a subjekt, predikat i objekt zapisani su tim redom dok kraj trojke označava točka i prijelaz u sljedeći red. Slijedi primjer zapisa N-trojki prema grafu prikazanom prethodno. [5]

```
<https://shop.foi.hr/hr/stil/39-salica.html>
<https://shop.foi.hr/hr/naziv_proizvoda> „ŠALICA“ .

<https://shop.foi.hr/hr/stil/39-salica.html>
<https://shop.foi.hr/hr/cijena> „30.00“ .

<https://shop.foi.hr/hr/stil/39-salica.html>  <https://shop.foi.hr/hr/opis>
„Keramička šalica s keramičkom žličicom.“ .


<https://shop.foi.hr/hr/stil/38-kisobran.html>
<https://shop.foi.hr/hr/naziv_proizvoda> „KIŠOBRAN“ .

<https://shop.foi.hr/hr/stil/39-kisobran.html>
<https://shop.foi.hr/hr/cijena> „30.00“ .

<https://shop.foi.hr/hr/stil/39-kisobran.html>
<https://shop.foi.hr/hr/opis> „Sklopivi kišobran crvene boje s bijelim
logotipom FOI, promjera 980 mm.“ .
```

N-Triples, odnosno N-trojke nisu naročito čitljive jer URI-ji ne smiju biti skraćeni, ali je lako njima manipulirati u datoteci jer je svaka trojka zaseban red i napisati kod koji će ih koristiti. RDF tekstualne datoteke kod N-Triples serijalizacije obično imaju ekstenziju .nt. [5]

2.3.1.2.Turtle

Kao što je već navedeno, Turtle je skraćenica za „Terse RDF Triple Language“, a N-Triples su podskup Turtle serijalizacije, što znači da svaka datoteka koja je validna u N-Triples serijalizaciji, validna je i u Turtle serijalizaciji. Razlika je u tome što Turtle omogućuje skraćene URI-je i prečace koji sprječavaju nepotrebno ponavljanje dijelova trojki. Primjerice, ako neke trojke dijele isti subjekt, on se može napisati jednom, a predikati i objekti se navesti razdvojeni točkom i zarezom. [5] Slijedi zapis gornjeg primjera u Turtle sintaksi.

```
@prefix op: <https://shop.foi.hr/hr/>

<https://shop.foi.hr/hr/stil/39-salica.html>
      op:naziv_proizvoda „ŠALICA“;
      op:cijena „30.00“;
      op:opis „Keramička šalica s keramičkom žličicom.“.
<https://shop.foi.hr/hr/stil/38-kisobran.html>
      op:naziv_proizvoda „KIŠOBRAN“;
      op:cijena „30.00“;
op:opis „Sklopivi kišobran crvene boje s bijelim logotipom FOI, promjera 980 mm.“.
```

Usporedbom dva navedena zapisa istog primjera može se vidjeti da su datoteke u Turtle serijalizaciji znatno čitljivije od N-trojki.

RDF tekstualne datoteke kod Turtle serijalizacije obično imaju ekstenziju .ttl.

2.3.2.JSON-LD

JSON-LD serijalizacija je koja omogućuje da se koristi JSON sintaksa za RDF grafove i skupove podataka. Ovaj format serijalizacije može se koristiti za transformaciju JSON dokumenata u RDF uz minimalne promjene jer nudi univerzalne identifikatore za JSON objekte pomoću kojih se JSON dokument može referirati na objekt u nekom drugom JSON dokumentu na Webu, kao i na tip podatka i jezik. [3] Slijedi primjer naveden za N-Triples i Turtle u JSON-LD formatu.

```
{
      „@context“: „https://shop.foi.hr/hr/“,
      „@id“: „https://shop.foi.hr/hr/stil/39-salica.html“,
      „naziv_proizvoda“: „ŠALICA“,
      „cijena“: „30.00“,
      „opis“: „Keramička šalica s keramičkom žličicom.“
}
```

```
{
      „@context“: „https://shop.foi.hr/hr/“,
      „@id“: „https://shop.foi.hr/hr/stil/38-kisobran.html“,
      „naziv_proizvoda“: „KIŠOBRAN“,
      „cijena“: „30.00“,
      „opis“: „Sklopivi kišobran crvene boje s bijelim logotipom FOI, promjera 980 mm.“
}
```

JSON-LD jedan je od novijih formata serijalizacije, a sve je popularniji zbog široke upotrebe JSON notacije u API-jima. JSON-LD datoteke obično imaju uobičajenu ekstenziju za JSON datoteke - .json. [5]

2.3.3. RDFa

RDFa koristi se za ugradnju RDF podataka unutar HTML i XML dokumenta, što posebno ima svrhu kod tražilica jer omogućuje da tražilice agregiraju podatke i koriste ih kako bi obogatili rezultate pretrage. [3]

Kako bi bilo moguće specificirati RDF trojke unutar HTML-a, koriste se četiri posebna RDFa atributa – *resource, property, typeof* i *prefix*.

2.3.4. RDF/XML

RDF/XML serijalizacija omogućuje XML sintaksu za RDF grafove. To je ujedno i najstariji format serijalizacije, nastao kasnih devedesetih godina prošlog stoljeća kad i sami RDF. Stoga je i dalje najuobičajeniji oblik RDF-a. [3]

Može biti teško čitljiv onima koji ne poznaju XML sintaksu, ali može se i generirati i bez znanja XML sintakse, koristeći alate poput XQuery-ja. Slijedi primjer korišten prethodno zapisan kao RDF/XML dokument. [5]

```
<?xml version="1.0"?>

 <rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:shop="https://shop.foi.hr/hr/">

 <rdf:Description
 rdf:about="https://shop.foi.hr/hr/stil/38-kisobran.html">
   <shop:naziv_proizvoda>KIŠOBRAN</shop:naziv_proizvoda>
   <shop:cijena>30.00</shop:cijena>
   <shop:opis>Sklopivi kišobran crvene boje s bijelim logotipom FOI, promjera
980 mm.</shop:opis>
 </rdf:Description>

 <rdf:Description
 rdf:about="https://shop.foi.hr/hr/stil/39-salica.html">
   <shop:naziv_proizvoda>ŠALICA</shop:naziv_proizvoda>
```

```
  <shop:cijena>30.00</shop:cijena>
  <shop:opis>Keramička šalica s keramičkom žličicom.</shop:opis>
</rdf:Description>
</rdf:RDF>
```

Kod RDF/XML dokumenta, RDF trojke specificiraju se XML elementom s oznakom *rdf:RDF*, a sadržaj tog elementa je skup trojki koji sadrži oznaku *rdf:Description*. Kod ovog formata serijalizacije koristi se mehanizam prostora imena (namespace) pri čemu bi vanjski prostori imena trebali biti RDF dokumenti koji definiraju resurse koji se mogu ponovno koristiti, što je odlika velikih i distribuiranih kolekcija znanja. [3]

RDF tekstualne datoteke kod RDF/XML serijalizacije obično imaju ekstenziju .rdf.

2.4. Vrste spremnika

RDF koristi spremnike (engl. *containers*) kad postoji potreba za opisivanjem skupine resursa, npr. da je knjiga napisana od strane nekoliko autora, da određena skupina studenata pohađa neki kolegij i sl. Spremnik je zapravo resurs koji sadrži određene objekte. Postoje tri predefinirane vrste spremnika [6]:

- `rdf:Bag`
- `rdf:Seq`
- `rdf:Alt`

`rdf:Bag` je neuređeni spremnik, što znači da redoslijed članova, odnosno elemenata spremnika, nije bitan i da se čak mogu pojaviti dupli članovi. Primjer takvog spremnika je mapa s dokumentima, lista sa studentima i sl. Slijedi primjer RDF/XML koda za *Bag* spremnik sa studentima koji pohađaju neki predmet. [6]

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:s="http://exapmle.org/students/vocab#">
   <rdf:Description rdf:about="http://example.org/courses/6.001">
      <s:students>
         <rdf:Bag>
              <rdf:li rdf:resource="http://example.org/students/Amy"/>
              <rdf:li rdf:resource="http://example.org/students/Maria"/>
              <rdf:li rdf:resource="http://example.org/students/John"/>
         </rdf:Bag>
```

```
        </s:students>
      </rdf:Description>
</rdf:RDF>
```

`rdf:Seq` je uređeni spremnik, što znači da je redoslijed elemenata bitan, ali svejedno može sadržavati duple elemente. Primjer takvog spremnika je abecedni popis. Recimo, za primjer naveden prethodno za `rdf:Bag`, kod `rdf:Seq`, resursi bi bili navedeni abecednim redom – Amy, John pa onda Maria. [6]

`rdf:Alt` je spremnik koji sadrži skupinu alternativa za neku vrijednost svojstva. Takav spremnik može, primjerice, sadržavati prijevode neke knjige na više jezika. Slijedi primjer za `rdf:Alt` u kojem su alternative nosači zvuka na kojima se nalazi glazba Beatlesa. [7]

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:cd="http://www.recshop.fake/cd#">
  <rdf:Description rdf:about="http://www.recshop.fake/cd/Beatles">
      <cd:format>
         <rdf:Alt>
               <rdf:li>CD</rdf:li>
               <rdf:li>Record</rdf:li>
               <rdf:li>Tape</rdf:li>
         </rdf:Alt>
      </cd:format>
  </rdf:Description>
</rdf:RDF>
```

Spremnici nisu jedini način za opisivanje skupine resursa, već se samo nude kao opcija za standardizirani način za isto, u svrhu interoperabilnosti podataka koji uključuju skupine resursa.

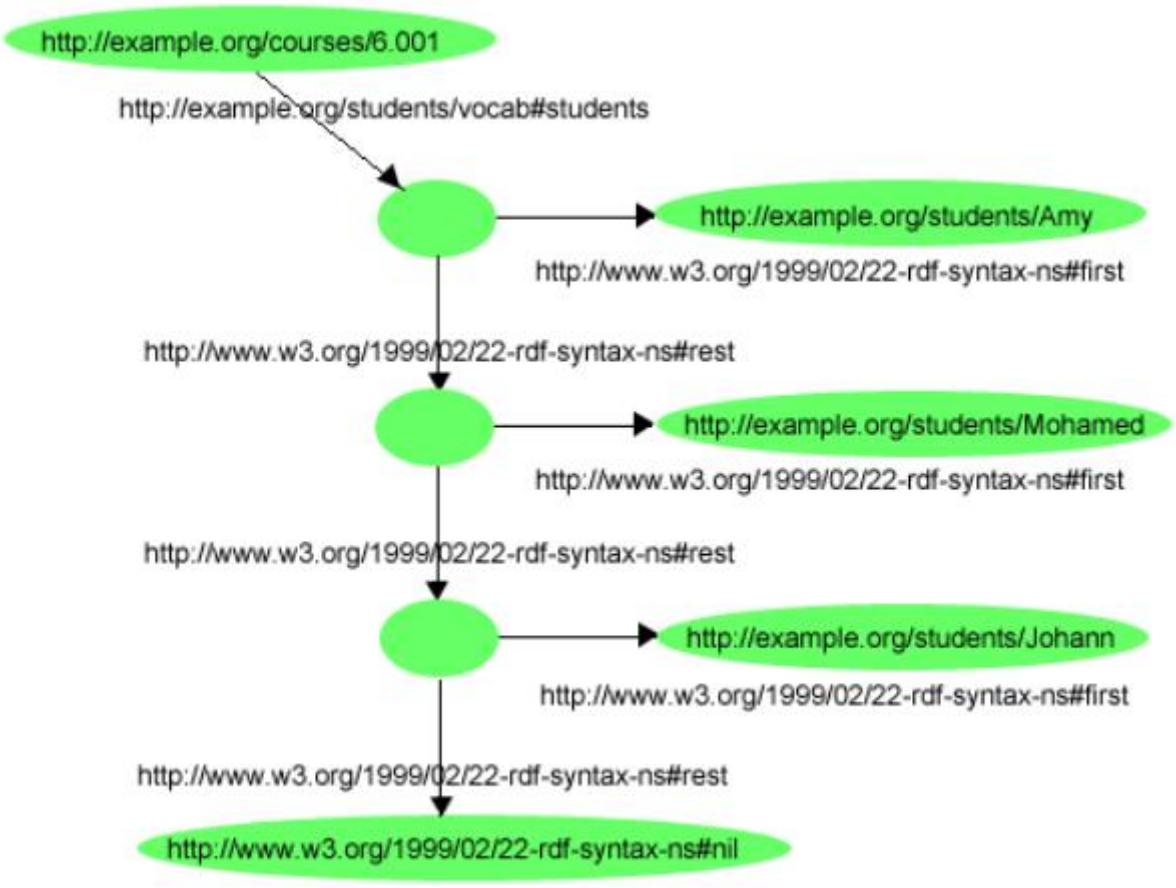
2.5. RDF kolekcije

S obzirom da nema načina da se opisani spremnici završe, odnosno da se definitivno utvrdi kako nema više članova spremnika osim onih koji su navedeni, RDF omogućuje da se opišu skupine koje sadržavaju samo specificirane članove, i to u obliku RDF kolekcija. [6]

RDF kolekcija skupina je predstavljena kao struktura liste u RDF grafu, koja se kreira pomoću predefiniranih elemenata [6]:

- `rdf:List`
- `rdf:first`
- `rdf:rest`
- `rdf:nil`

Na slici ispod može se vidjeti graf koji je vezan za prethodni primjer naveden za spremnike, a koji prikazuje da su studenti na predmetu 6.001 Amy, Mohamed i Johann. [6]



Slika 4 RDF kolekcija -struktura liste [7]

Svaki član kolekcije, odnosno student, je objekt sa svojstvom `rdf:first` čiji je subjekt resurs koji je u ovom slučaju prazan čvor. Taj resurs povezan je s ostatkom liste uz pomoć svojstva `rdf:rest`, a kraj liste prikazan je svojstvom `rdf:rest` koje ima resurs `rdf:nil` koji predstavlja praznu listu i definirano je da je tipa `rdf:List`. [6]

RDF/XML ima specijalnu notacija koja olakšava opisivanje kolekcija jer se u RDF/XML-u kolekcija može opisati pomoću svojstva rdf:parseType="Collection" koje sadrži skupinu ugnježđenih elemenata koji predstavljaju članove kolekcije. Sami rdf:parseType naglašava da se sadržaj elementa promatra na određeni način. RDF/XML dokument koji bi rezultirao grafom sa slike iznad bio bi dosta sličan primjeru spremnika navedenog ranije i izgledao bi ovako [6]:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:s="http://exapmle.org/students/vocab#">
   <rdf:Description rdf:about="http://example.org/courses/6.001">
      <s:students rdf:parseType="Collection">
            <rdf:Description
rdf:about="http://example.org/students/Amy"/>
            <rdf:Description
rdf:about="http://example.org/students/Mohamed"/>
            <rdf:Description
rdf:about="http://example.org/students/Johann"/>
      </s:students>
   </rdf:Description>
</rdf:RDF>
```

2.6. RDF Schema

RDF Schema definira vokabular i značenje podataka za RDF, odnosno definira koje oznake bi se trebale koristiti za svojstva RDF-a, kao i koje objekte ta svojstva mogu opisivati. Drugim riječima, mehanizam RDF sheme pruža osnovni tipski sustav za korištenje u RDF modelima. [8]

Vokabular ili rječnik koji RDF Schema definira može se podijeliti u [9]:

- Skup svojstava, koja su binarne relacije između subjekta i objekta (*rdfs:subPropertyOf, rdfs:subClassOf, rdfs:domain, rdfs:range, rdfs:type*)
- Skup klasa koje obilježavaju skup resursa; elementi klase nazivaju se instance klase
- Ostale funkcionalnosti, poput sustava klasa i svojstava za opisivanje listi
- Pomoćni rječnik koji se koristi za dokumentiranje, komentiranje itd.

Dva krucijalna konstrukta kod RDF sheme su *subClassOf* i *subPropertyOf*. Objekti mogu biti instance jedne ili više klasa što se označava korištenjem svojstva *type*, a ovi konstrukti omogućuju da se definira hijerarhija klasa i hijerarhija svojstava. [8]

Nadalje, konstrukti koji su također važni su *domain* i *range*, koji služe da se ograniče kombinacije klasa i svojstava te spomenuti *type* pomoću kojeg se resurs definira kao instanca određene klase. [4]

Ovaj sustav klasa i svojstava zapravo je jako sličan onom u objektno-orijentiranim jezicima kao što je Java. Međutim, razlika je to da umjesto da se klasa definira s obzirom na svojstva koja imaju njezine instance, RDF shema opisuje svojstva s obzirom na klase resursa na koje se to svojstvo odnosi. Prednost toga je to što je svakome dopušteno proširiti opis postojećeg resursa, što je jedan od principa samog Weba. [6]

Primjer RDF sheme preuzet sa w3schools.com prikazuje neke od konstrukta spomenutih prethodno.

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://www.animals.fake/animals#">

<rdf:Description rdf:ID="animal">
  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
</rdf:Description>

<rdf:Description rdf:ID="horse">
  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="#animal"/>
</rdf:Description>

</rdf:RDF>
```

Resurs „horse“ je potklasa klase „animal“. Koristeći konstrukt *rdfs:Class* umjesto *rdf:Description* možemo skratiti shemu i onda više nije potreban ni konstrukt *rdf:type*.

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://www.animals.fake/animals#">

<rdfs:Class rdf:ID="animal" />

<rdfs:Class rdf:ID="horse">
  <rdfs:subClassOf rdf:resource="#animal"/>
</rdfs:Class>

</rdf:RDF>
```

2.6.1. Klase

Resursi mogu biti podijeljeni u skupine koje se nazivaju klasama. Klase su i same resursi, a članovi klase nazivaju se instancama te klase. Opisuju se koristeći resurse *rdfs:Class* i *rdfs:Resource* i svojstva *rdf:type* i *rdf:subClassOf*, što se može vidjeti i u prethodnom primjeru. „Animal“ je klasa, skupina resursa koji imaju određena svojstva koja ih svrstavaju u životinje, a „horse“ je instanca te klase jer je to vrsta životinje i sve instance klase „horse“ automatski su i instance klase „animal“. Nadalje, u RDF Schemi sve klase su potklase klase *rdfs:Resource*, budući da su instance svih klasa zapravo resursi [10].

Temeljne klase su [10]:

- rdfs:Resource – klasa svih resursa
- rdfs:Class – klasa svih klasa
- rdfs:Literal – klasa svih literala (stringova, brojeva)
- rdf:XMLLiteral – klasa svih XML literala (potklasa od rdfs:Literal)
- rdfs:Datatype – klasa svih tipova podataka
- rdfs:Property – klasa svih svojstava
- rdfs:Statement – klasa svih reificiranih izjava
- rdfs:Container – natklasa svih spremnika

2.6.2. Svojstva

Osim svojevrsnog kategoriziranja stvari koje se žele opisati u klase i opisivanje tih klasa, potrebno je moći opisati i specifična svojstva koja karakteriziraju te klase. U RDF shemi svojstva se opisuju koristeći klasu *rdf:Property* i svojstva *rdfs:domain*, *rdfs:range*, *rdfs:subPropertyOf*, a sva svojstva su instance klase *rdf:Property*. [6]

rdfs:subPropertyOf povezuje svojstvo s jednim od njegovih nadsvojstava, baš kao što *rdfs:subClassOf* povezuje klasu s jednom od njezinih natklasa. [10]

rdfs:domain i *rdfs:range* ubrajaju se u temeljna svojstva za ograničavanje svojstava i instance su klase *rdf:Property*.. *rdfs:domain* služi da se odredi da su svi resursi koji imaju to svojstvo instance jedne ili više klasa, odnosno određuje domenu nekog svojstva. *rdfs:range* služi da se odredi da su sve vrijednosti tog svojstva instance jedne ili više klasa, odnosno određuju doseg nekog svojstva. [10]

Slijedi primjer RDF sheme preuzete iz RDF Primera koja opisuje motorna vozila gdje se jasno vidi upotreba klasa i svojstava.

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://example.org/schemas/vehicles">

<rdfs:Class rdf:ID="MotorVehicle" />

<rdfs:Class rdf:ID="PassengerVehicle">
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdfs:Class>


<rdfs:Class rdf:ID="Truck">
  <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdfs:Class>


<rdfs:Class rdf:ID="MiniVan">
  <rdfs:subClassOf rdf:resource="#Van"/>
  <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
</rdfs:Class>


<rdfs:Class rdf:ID="Person"/>
<rdfs:Datatype rdf:about="&xsd;integer"/>
<rdf:Property rdf:ID="registeredTo">
  <rdfs:domain rdf:resource="#MotorVehicle"/>
  <rdfs:range rdf:resource="#Person"/>
</rdf:Property>


<rdf:Property rdf:ID="rearSeatLegRoom">
  <rdfs:domain rdf:resource="#PassengerVehicle"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="driver">
  <rdfs:domain rdf:resource="#MotorVehicle"/>
</rdf:Property>

<rdf:Property rdf:ID="primaryDriver">
  <rdfs:subPropertyOf rdf:resource="#driver"/>
</rdf:Property>

</rdf:RDF>
```

3. RDF baze podataka

RDF baze podataka tip su grafovskih baza podataka i pohranjuju semantičke činjenice, a temelje se na RDF-u. Pohranjuju podatke kao mrežu objekata koji su međusobno povezani, zbog čega su RDF baze podataka pogodne za upravljanje velikim brojem podataka s velikim brojem veza, a uz to su fleksibilnije od relacijskih baza podataka. Sposobne su za izvršavanje moćnih semantičkih upita i otkrivanje novih informacija iz postojećih veza. [11]

Samim time što spadaju u grafovske baze podataka, RDF baze podataka zapravo su NoSQL baze podataka, koje su nerelacijske i namijenjene za pohranu velikih količina podataka. Stoga zahtijevaju upitni jezik koji je nešto napredniji od SQL-a kako bi bilo moguće vršiti semantičke upite nad podacima u skladu s idejom semantičkog weba. Taj upitni jezik zove se SPARQL. [12]

3.1. NoSQL baze podataka

NoSQL (eng. *Not only SQL*) baze podataka su nerelacijske baze podataka i u odnosu na tradicionalne relacijske baze podataka imaju bitne razlike. Najznačajnije su te da kod pohrane podataka u NoSQL-u fiksna shema nije obavezna te da je skalabilnost horizontalna, a ne vertikalna kao kod relacijskih baza podataka. U slučaju horizontalne skalabilnosti znatno je lakše upravljati podacima, što je jako važno kod nerelacijskih baza podataka budući da su namijenjene za pohranu velikih količina podataka (npr. Google, Facebook i slične platforme koje dnevno pohranjuju terabajte podataka). Upravo zbog potrebe za pohranom velikih količina podataka popularnost nerelacijskih baza podataka rapidno raste. [13]

Samog termina *NoSQL* dosjetio se Carlo Strozzi 1998.. godine kad je napravio bazu koja nema SQL sučelje. Taj termin ponovno je upotrijebljen 2009. od strane Erica Evansa kako bi se dalo ime bazama koje nisu relacijske i nakon toga bilježi se rast popularnosti NoSQL baza podataka. [13]

Četiri su glavne vrste NoSQL baza podataka [14]:

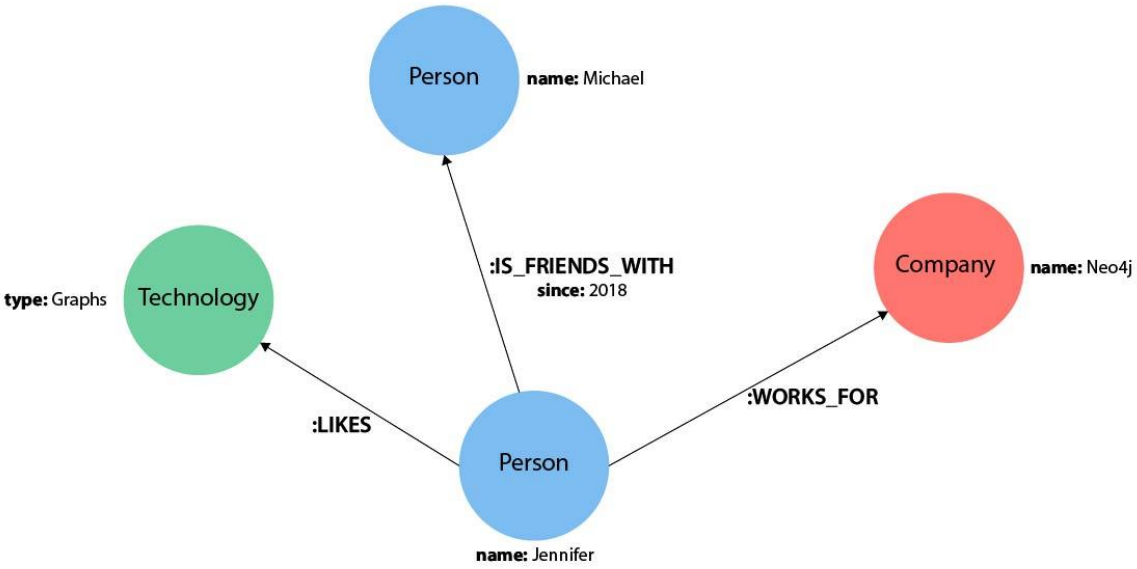
- Ključ-vrijednost baze podataka (eng. *Key-value databases*)
- Grafovske baze podataka (eng. *Graph databases*)
- Dokumentne baze podataka (eng. *Document databases*)
- Stupčaste baze podataka (eng. *Columnar databases*)

RDF baze podataka svrstavaju se upravo u grafovske baze podataka, stoga će biti detaljnije opisane.

3.1.1. Grafovske baze podataka

Grafovske baze podataka najveću primjenu pronašle su u aplikacijama gdje se naglašava važnost odnosa između podataka jer su osmišljene i napravljene tako da pridaju jednaku važnost vezama između podataka kao i samim podacima. [14]

U grafovskim bazama podataka podaci se opisuju pomoću čvorova (eng. nodes) koji označavaju entitete i lukova (eng. edges) koji označavaju odnose, a i čvorovi i lukovi mogu imati svoja svojstva (eng. properties) dok lukovi imaju tip, smjer te početni i završni čvor. Podaci se pohranjuju jednom, a onda se mogu čitati i interpretirati na razne načine s obzirom na odnose između čvorova.. [15]



Slika 5 Primjer grafovskog modela podataka sa svojstvima (Izvor: https://neo4j.com/developer/cypher/syntax/)

Najveća snaga grafovskih baza podataka je to da bez obzira na veličinu skupa podataka mogu učinkovito upravljati složenim upitima jer koriste obrasce i točke na grafu kako bi prikupili informacije iz velikog broja čvorova i veza. [15]

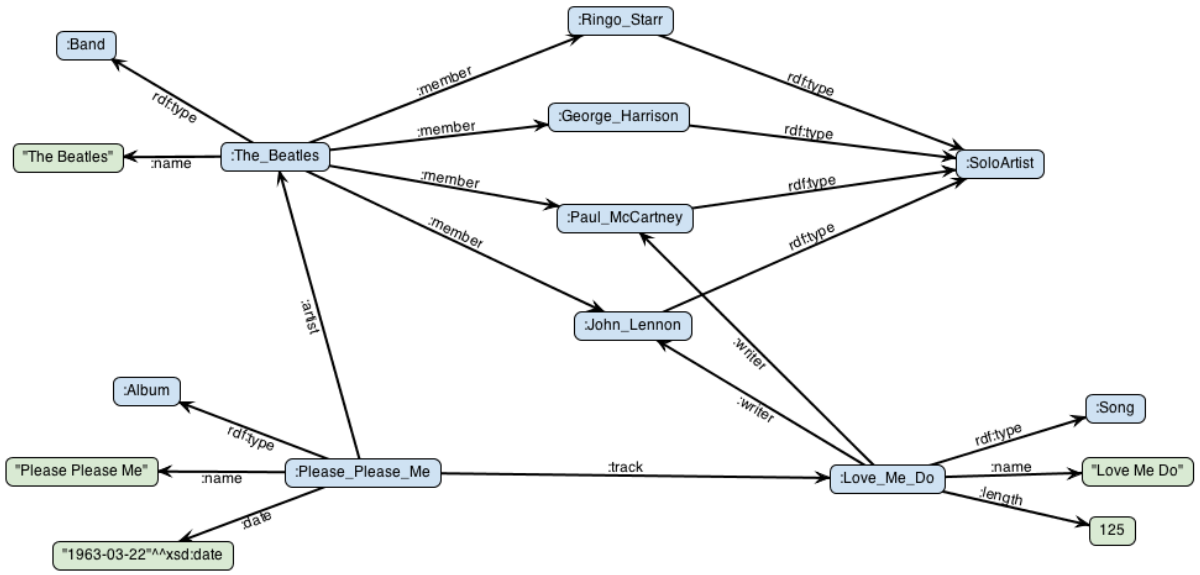
Sustavi za upravljanje grafovskim bazama podataka iznimno su popularni, a najpopularniji i možda najpoznatiji je Neo4j koji ima i deklarativni upitni jezik sličan SQL-u, a zove se Cypher. [15]

3.2. RDF model podataka

Osim grafovskog modela podataka za svojstvima, u grafovskim bazama podataka najčešće se koristi i upravo RDF model podataka.

RDF prikazuje entitete kao resurse, a resursi su na modelu podataka prikazani kao trojke koje se sastoje od subjekta, predikata i objekta, pri čemu je subjekt zapravo entitet koji se opisuje, predikat njegovo svojstvo, a objekt vrijednost tog svojstva. Kao što je već spomenuto u ranijim poglavljima, trojke čine RDF izjavu, a RDF graf, odnosno model podataka, upravo je niz tih izjava. [16]

U poglavlju o RDF-u i izjavama već su dani jednostavni primjeri RDF grafa, a na slici ispod može se vidjeti nešto složeniji graf s podacima o Beatlesima.



Slika 6 Primjer grafa RDF modela podataka (Izvor: https://docs.stardog.com/tutorials/rdf-graph-data-model)

Za razliku od ostalih tipova grafovskih baza podataka, RDF baze podataka podržavaju i ontologije kao opcionalne sheme modela, koje omogućuju formalni opis podataka i specificiraju klase i veze, kao i njihovu hijerarhiju. [11]

Za pretraživanje RDF modela podataka i vršenje upita koristi se SPARQL, upitni jezik koji se temelji na definiranju uzoraka i koji je sintaksom sličan SQL-u.

3.3. SPARQL

RDF je postao standard 1999. godine, a od tada do 2004. godine razvijalo se preko 12 upitnih jezika. 2008. godine, SPARQL (*SPARQL Protocol and RDF Query Language*) je postao službena W3C specifikacija i preporuka. Direktor W3C-a Tim Berners-Lee izjavio je kako je korištenje semantičkog weba bez SPARQL-a kao korištenje relacijskih baza podataka bez SQL-a. SPARQL nije dizajniran da radi upite nad relacijskim podacima, već nad podacima u RDF modelu podataka. Kod postavljanja upita, uvjeti se opisuju slično kao RDF trojke. [17]

Slika 7 SPARQL logo

Uzorci koji su definirani za pretraživanje slični su RDF-u, s tim da sadrže varijable, a vrijednosti tih varijabla vraćaju se kao odgovor na postavljeni upit ako je postignuto usklađivanje uzorka s postojećim trojkama. [18]

Osnovni obrasci za upite u SPARQL-u su: [18]

- SELECT – sličan kao SELECT u SQL-u, prema traženom uvjetu u upitu, odnosno prema uzorku, vraća konkretne vrijednosti varijabli
- ASK – odgovara na pitanje postoji li određeni RDF graf
- DESCRIBE – na temelju upita vraća RDF graf
- CONSTRUCT – omogućuje da se rezultati preoblikuju u RDF graf

Sintaksa SPARQL-a uglavnom se temelji na Turtle serijalizaciji. Kako bi se prikazala sintaksa, izvršeno je nekoliko jednostavnih upita u SPARQL Playground, na sljedećem primjeru RDF dokumenta pisanom upravo u Turtle sintaksi [19]:

```
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefix dbp: <http://dbpedia.org/property/> .
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tto: <http://example.org/tuto/ontology#> .
@prefix ttr: <http://example.org/tuto/resource#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ttr:Eve dbo:parent ttr:William ;
      dbp:birthDate "2006-11-03"^^xsd:date ;
      dbp:name "Eve" ;
      tto:sex "female" ;
      a dbo:Person .

ttr:John dbp:birthDate "1942-02-02"^^xsd:date ;
      dbp:name "John" ;
      tto:pet ttr:LunaCat , ttr:TomCat ;
      tto:sex "male" ;
      a dbo:Person .

ttr:LunaCat dbp:name "Luna" ;
      tto:color "violet" ;
      tto:sex "female" ;
      tto:weight 4.2 ;
      a tto:Cat .

ttr:RexDog dbp:name "Rex" ;
      tto:color "brown" ;
      tto:sex "male" ;
      tto:weight 8.8 ;
      a tto:Dog .

ttr:SnuffMonkey dbp:name "Snuff" ;
      tto:color "golden" ;
      tto:sex "male" ;
      tto:weight 3.6 ;
      a tto:Monkey .

ttr:TomCat dbp:name "Tom" ;
      tto:color "grey" ;
      tto:sex "male" ;
      tto:weight 5.8 ;
      a tto:Cat .

ttr:William dbo:parent ttr:John ;
      dbp:birthDate "1978-07-20"^^xsd:date ;
      dbp:name "William" ;
      tto:pet ttr:RexDog ;
      tto:sex "male" ;
      a dbo:Person .

dbo:Person rdfs:subClassOf tto:Creature .

tto:Animal a rdfs:Class ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
```

```
      rdfs:label "animal" ;
      rdfs:subClassOf tto:Creature .

tto:Cat a rdfs:Class ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "cat" ;
      rdfs:subClassOf tto:Animal .

tto:Creature a rdfs:Class ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "creature" .

tto:Dog a rdfs:Class ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "dog" ;
      rdfs:subClassOf tto:Animal .

tto:Monkey a rdfs:Class ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "monkey" ;
      rdfs:subClassOf tto:Animal .

tto:pet a rdf:Property ;
      rdfs:domain dbo:Person ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "domestic animal" ;
      rdfs:range tto:Animal .

tto:sex a rdf:Property ;
      rdfs:domain tto:Creature ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "sex" ;
      rdfs:range xsd:string .

tto:weight a rdf:Property ;
      rdfs:comment "weight in kilograms" ;
      rdfs:domain tto:Creature ;
      rdfs:isDefinedBy <http://example.org/tuto/ontology#> ;
      rdfs:label "weight" ;
      rdfs:range xsd:decimal .
```

Želimo li dohvatiti prvih deset redova, odnosno prvih deset trojki, upit je sljedeći:

```
SELECT DISTINCT * WHERE {
  ?s ?p ?o
}
LIMIT 10
```

Rezultat upita prikazan je na slici ispod. Također je korišten DISTINCT kako u rezultatu ne bi bilo duplikata, baš kao što to postoji i u SQL-u.

s	p	o
ttr:Eve	dbo:parent	ttr:William
ttr:Eve	dbp:birthDate	"2006-11-03"
ttr:Eve	dbp:name	"Eve"
ttr:Eve	tto:sex	"female"
ttr:Eve	rdf:type	dbo:Person
ttr:John	dbp:birthDate	"1942-02-02"
ttr:John	dbp:name	"John"
ttr:John	tto:pet	ttr:LunaCat
ttr:John	tto:pet	ttr:TomCat
ttr:John	tto:sex	"male"

Slika 8 Rezultat upita koji dohvaća prvih deset trojki

Želimo li dohvatiti osobe i njihove ljubimce, upit je sljedeći:

```
SELECT ?person ?pet WHERE {
    ?person rdf:type dbo:Person .
      ?person tto:pet ?pet .
}
```

Rezultat upita prikazan je na slici ispod.

person	pet
ttr:John	ttr:LunaCat
ttr:John	ttr:TomCat
ttr:William	ttr:RexDog

Slika 9 Rezultat upita koji dohvaća osobe s njihovim ljubimcima

Također možemo izvršiti upit pomoću kojeg možemo vidjeti koje potklase neka klasa ima. Naprimjer, želimo li vidjeti koje su potklase klase Creature, izvršit ćemo sljedeći upit:

```
SELECT ?subSpecies   WHERE {
  ?subSpecies rdfs:subClassOf tto:Creature .
}
```

Rezultat upita prikazan je na slici ispod.

subSpecies
dbo:Person
tto:Animal
tto:Cat
tto:Dog
tto:Monkey

Slika 10 Rezultat upita koji dohvaća potklase klase Creature

Kao što se može vidjeti, SPARQL zaista podsjeća na SQL i s njim dijeli i neke ključne riječi i naredbe, poput SELECT, DISTINCT, WHERE, LIMIT, ORDER BY, GROUP BY.

4. Sustavi za upravljanje RDF bazama podataka

U nastavku će biti opisani neki od najpopularnijih sustava za upravljanje RDF bazama podataka sa osnovnim informacijama i prednostima koje imaju.

4.1. AllegroGraph

AllegroGraph je multi-model sustav za upravljanje bazama podataka koji podržava dokumentne i grafovske, odnosno RDF baze podataka i sposoban je procesuirati i analizirati kompleksne i distribuirane podatke. Može skalirati milijarde trojki, odnosno dokumenata, a i dalje održati visoke performanse. Usklađen je s W3C standardima i ACID svojstvima i podržava JSON, JSON-LD, SPARQL, OWL, SHACL i Prolog. [20]

Slika 11 AllegroGraph logo

Važno je naglasiti da AllegroGraph ne pohranjuje nužno podatke kao čiste RDF trojke (subjekt-predikat-objekt), već kao petorke – trojkama se pridružuju dodatne informacije i kontekst, što olakšava upravljanje podacima. [21]

Dakle, svaka izjava u AllegroGraph bazi podataka zapravo je petorka i sastoji se od sljedećih polja [23]:

- subjekt (s)
- predikat (p)
- objekt (o)
- graf (g)
- jedinstveni identifikator trojki (i)

Uobičajenim dijelovima RDF trojke – subjektu koji je zapravo resurs, predikatu koji označava svojstvo i objektu koji označava vrijednost tog svojstva pridodaju se još i graf kako bi se bolje opisao kontekst te jedinstveni identifikator trojki, pomoću kojeg je svaka trojka jedinstvena, što olakšava referenciranje na istu. [21]

Jedna od najznačajnijih značajki AllegroGrapha je to što podržava nekoliko specijaliziranih tipova podataka u svrhu efikasne pohrane, manipuliranja i pretrage informacija. Naime, uključuje SNA (engl. Social Network Analysis) koja na trojke gleda kao na graf s relacijama i omogućuje odgovore na pitanja o povezanosti nekih individua, postojanju klastera unutar podataka i važnosti informacija. [21]

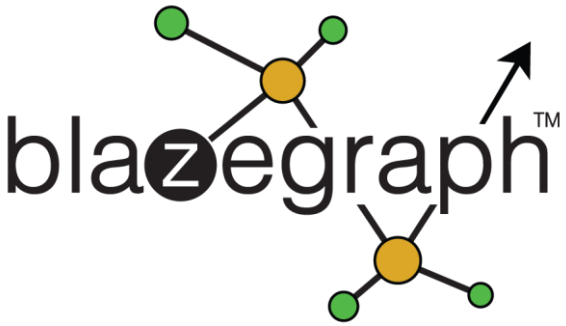
AllegroGraph također vješto rukuje s višedimenzionalnim podacima, točnije onima vezanim za lokacije, kao što su koordinate. Takvi podaci obično se nazivaju *geospatial* podacima. Također podržava efikasnu pohranu i dohvaćanje temporalnih podataka kao što su datum i vrijeme, vremenski intervali i slično. [21]

Kad se to troje kombinira, omogućuje se fleksibilno analiziranje velike količine strukturiranih i nestrukturiranih podataka. [20]

Još jedna važna značajka AllegroGrapha je Gruff – alat za vizualizaciju grafa podataka i grafičku izradu upita. Jako je koristan za praktični pregled podataka i informacija u grafu, omogućuje i dodavanje čvorova i veza u grafu. [22]

4.2. Blazegraph

Blazegraph je grafovska baza podataka otvorenog koda, pisana u Javi, skalabilna i vrlo visokih performansi. Koristi RDF i RDR kao standarde za modele podataka, a podržana je na zasebnom serveru, ali i na visoko dostupnim klasterima. U oba slučaja podržava ACID svojstva: atomarnost, konzistentnost, izolaciju i trajnost podataka. [23]



Slika 12 Blazegraph logo

Objavljena je 2016. godine, a prije toga bila je poznata kao Bigdata. Razvijana je tako da radi sa web skaliranim semantičkim grafovima, a podržava i pohranjene procedure kroz ekstenzije SPARQL-a, što omogućuje primjenjivanje kompleksnije logike na samu bazu podataka. Pohranjeni upit može se koristiti pozivanjem instance pripadajuće klase. [23]

Svojevrsni nedostatak Blazegrapha je to što ne podržava vizualizaciju podataka u obliku grafa, što je inače praktično i korisnicima pojednostavljuje upravljanje bazom podataka. Međutim, bez obzira na to, zbog visokih performansi i izražene skalabilnosti te jednostavnosti sučelja i same uporabe, stalno je među najpopularnijim sustavima za upravljanje grafovskim bazama podataka i u širokoj je upotrebi. [23]

4.3. Stardog

Stardog je grafovska baza podataka koja je također jako skalabilna, usklađena s ACID svojstvima i može skalirati i do trilijun trojki. Također omogućuje pristup SQL i NoSQL bazama podataka, kao i nestrukturiranim izvorima podataka. [24]

Slika 13 Stardog logo

Značajke koje Stardog čine jednim od najpopularnijih sustava za upravljanje grafovskim i RDF bazama podataka također su šifriranje značenja podataka uz same podatke, inkorporiranje pravila iz stvarnog svijeta kako bi implicitno znanje učinili eksplicitnim te fleksibilan pogled na podatke i mogućnost više slučajeva korištenja sa samo jednim modelom. [25]

Neke od prednosti koje Stardog ima nad nekim drugim sustavima su virtualni konektori koji se mogu povezati sa SQL i NoSQL bazama podataka, NLP pipeline pomoću kojeg se i nestrukturirani podaci mogu inkorporirati u graf, virtualizacija podataka te ugrađeno strojno učenje koje daje mogućnost prediktivne analize i pronalaženja sličnosti među podacima. [25]

Stardog se zbog svojih karakteristika često koristi za optimizaciju lanca nabave, praćenje u svrhu sprječavanja pranja novca i raznih prijevara te u farmaceutskoj industriji, a neki od korisnika su i NASA i Bosch. Zadnjih godina, Stardog je izrazito poboljšao performanse i skalabilnost te može učitati podatke brzinom od milijun trojki po sekundi. [25]

4.4. GraphDB

GraphDB je robusan, učinkovita i skalabilna grafovska baza podataka visokih performansi, kreirana od strane Ontotexta, a implementirana u Javi. Učitavanje podataka i evaluacija upita odvija se jako brzo čak i kad se radi o ogromnim ontologijama i bazama znanja. [26]

Slika 14 GraphDB logo

GraphDB omogućuje povezivanje teksta i podataka u velike grafove i lako eksperimentiranje s funkcijama poput umetanja i transformiranja bilo kojih tipova podataka u RDF format. Osim toga, moguće je vizualizirati ontologije i integrirati se s MongoDB bazom podataka u svrhu skaliranja velikih količina metapodataka i traženja semantičkih sličnosti među podacima. [25]

Obično se koristi u svrhu obogaćivanja i upravljanja metapodacima, semantičkog zaključivanja, semantičkog pretraživanja za poduzeća i skladištenje podataka, a neki od korisnika su BBC, Financial Times, Oxford University Press i Korea Telecom. Najkorišteniji je u poduzećima i situacijama gdje su ključni stabilnost i predviđanje za ostvarenje nekog važnog cilja. Postoje tri izdanja, a jedno od njih je GraphDB Free, koji je besplatan za korištenje. [25]

4.5. Usporedba sustava za upravljanje RDF bazama podataka

U tablici ispod može se vidjeti usporedba četiri iznad opisana sustava po određenim karakteristikama. Podaci su preuzeti sa stranice db-engines.com.

Tablica 1 Usporedba AllegroGrapha, Blazegrapha, Stardoga i GraphDB-a

	AllegroGraph	**Blazegraph**	**Stardog**	**GraphDB**
Primarni model baze podataka	Dokumentne baze podataka, grafovske baze podataka, RDF baze podataka	Grafovske baze podataka, RDF baze podataka	Grafovske baze podataka, RDF baze podataka	Grafovske baze podataka, RDF baze podataka
Developer	Franz Inc.	Blazegraph	Stardog Union	Ontotext
Godina nastajanja	2004.	2006.	2010.	2000.
Licenca	Komercijalna	Open Source	Komercijalna	Komercijalna
Operacijski sustavi	Linux, OS X, Windows	Linux, OS X, Windows	Linux, macOS, Windows	Svi operacijski sustavi s Java CM, Linux, OS X, Windows
Shema podataka	Da (RDFS)	Slobodna shema	Slobodna shema i OWL/RDFS	Slobodna shema i OWL/RDFS
Podrška za XML	Ne	-	Ne	Ne
SQL	SPARQL kao upitni jezik	SPARQL kao upitni jezik	Da, kompatibilan sa svim većim SQL varijantama kroz BI/SQL Server	SQL-u se pristupa kroz SPARQL korištenjem Apache Calcite kroz JDBC/ODBC
API-ji i druge metode pristupa	RESTful HTTP API, SPARQL	Java API, RESTful HTTP API, SPARQL QUERY, SPARQL UPDATE, TinkerPop 3	GraphQL, HTTP API, Jena RDF API, OWL, RDF4J API, Sesame REST HTTP Protocol, SNARL, SPARQL, Spring Data, Stardog Studio, TinkerPop 3	GeoSPARQL, GraphQL, GraphQL Federation, Java API, JDBC, RDF4J API, RDFS, RIO, Sail API, Sesame REST HTTP Protocol, SPARQL 1,1
Podržani programski jezici	C#, Clojure, Java, Lisp, Perl, Python, Ruby, Scala	.Net, C, C++, Java, JavaScript, PHP, Python, Ruby	.Net, Clojure, Groovy, Java, JavaScript, Python, Ruby	.Net, C#, Clojure, Java, JavaScript (Node.js), PHP, Python, Ruby, Scala

Okidači	Da	Ne	Da	Ne
Metode particioniranja	Uz Federaciju	Horizontalno particioniranje (sharding)	Ne podržava	Ne podržava
Metode replikacije	Multi-source, Source-replica	Da	Multi-source u HA-Cluster	Multi-source
Transakcijski koncepti	ACID	ACID	ACID	ACID
Istodobnost	Da	Da	Da	Da
Vanjski ključevi	Ne	Da (veze u grafovima)	Da (veze u grafovima)	Da (provjera ograničenja)

U tablici su navedene neke najvažnije karakteristike po kojima se mogu usporediti opisani sustavi za RDF baze podataka. Na osnovu njih može se zaključiti zašto ovi sustavi spadaju u najpopularnije sustave za RDF baze podataka, ali i koje su razlike među njima. Odabrane su jer obuhvaćaju važne informacije za bilo koju bazu podataka, poput upitnog jezika kojeg koriste, sheme podataka, okidača, vanjskih ključeva itd. Jedna od najznačajnijih snaga tradicionalnih relacijskih baza podataka upravo je referencijalni integritet i upotreba okidača za manipuliranje podacima. Od četiri uspoređena sustava jedino AllegroGraph nema referencijalni integritet preko vanjskih ključeva, dok Blazegraph i Stardog osiguravaju isti preko veza u grafovima, a GraphDB postavljanjem i provjerom ograničenja (engl. constraint). Što se tiče upitnog jezika, svi koriste SPARQL, a prednost Stardoga i GraphDB-a nad drugim dvjema bazama je ta da imaju i podršku za SQL.

Važna je i shema podataka jer je karakteristika nerelacijskih baza podataka da imaju fleksibilnu shemu. AllegroGraph koristi RDF shemu, dok je kod ostalih sustava shema slobodna, a Stardog i GraphDB podržavaju i OWL/RDFS shemu. Budući da je RDF vezan za XML, bitna karakteristika sustava je i podrška za XML, što označava neki oblik procesiranja podataka u XML formatu (npr. podršku za XPath ili XQuery). Za Blazegraph nije definirano ima li podršku, a ostala tri nemaju podršku za XML, međutim AllegroGraph ima mogućnost *bulk loadinga* XML datoteka, odnosno učitavanja velike količine podataka odjednom u obliku XML datoteke, dok Stardog ima opciju importa i exporta XML podataka.

Značajna karakteristika je i ona vezana za API i ostale metode pristupa jer je važno koje mehanizme komunikacije sustavi koriste kako bi se uvidjelo u koju svrhu ih se efikasno

može koristiti. Primjerice, GraphDB između ostalog koristi GeoSPARQL koji služi za ispitivanje geoprostorno povezanih podataka.

Bitna karakteristika nekog sustava za upravljanje bazama podataka je i koje metode particioniranja koriste jer se kod RDF baza podataka radi s velikim brojem trojki i sve su kompleksnije. Kako bi se poboljšala skalabilnost i performanse pri izvršavanju upita, koriste se metode particioniranja, odnosno pohranjivanja različitih podataka na različitim čvorovima. AllegroGraph koristi FedShard tehnologiju koju su sami razvili, a Blazegraph koristi horizontalno particioniranje koje se još naziva *sharding*, pri čemu se veliki skupovi podataka razdvajaju u manje skupove kojima se brže i lakše upravlja. GraphDB i Stardog ne podržavaju nikakve metode particioniranja.

Osim particioniranja, značajna karakteristika je i replikacija podataka, odnosno, pohranjivanja istih podataka na više čvorova u svrhu poboljšanja dostupnosti podataka. Sva četiri sustava podržavaju neke metode replikacije.

Sustavi uspoređeni u tablici visoko su rangirani po popularnosti kao RDF sustavi za baze podataka prema db-engines.com, dapače, u samom su vrhu – Graph DB na petom mjestu, Stardog na šestom, AllegroGraph na sedmom, a Blazegraph na osmom mjestu. Svi su impelmentirani u Javi, osim AllegroGrapha za kojeg nije definirano te su sva četiri usklađena s ACID svojstvima, odnosno osigurava se valjanost podataka kroz atomarnost, konzistentnost, izolaciju i durabilnost. Također, sva četiri sustava kao primarni model baze podataka imaju grafovske i RDF baze podataka, dok je svojevrsna prednost AllegroGrapha ta da ima i dokumentne baze podataka kao primarni model.

5. Izrada RDF baze podataka

U ovom poglavlju bit će prikazan rad s četiri sustava za upravljanje RDF bazama podataka opisanima u prethodnom poglavlju. Za svaki sustav bit će izrađena baza u istoj domeni kako bi se prikazalo kako svaki od njih radi i pritom će se sustavi usporediti s obzirom na primijećene karakteristike.

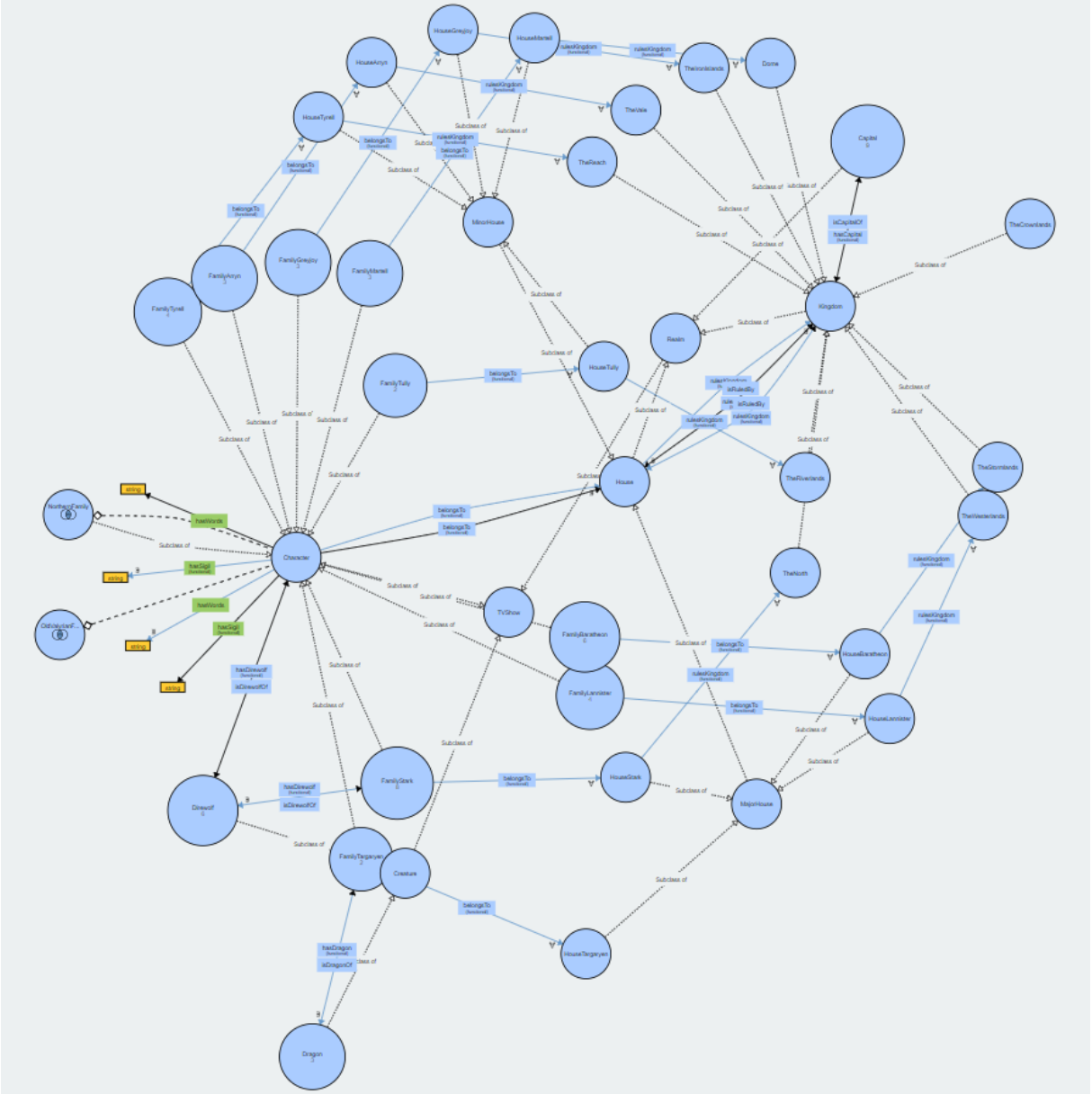
5.1. Opis domene i model podataka

Kako bi se prikazale karakteristike svakog sustava i način na koji se radi baza podataka i upiti nad njom, potrebno je odrediti domenu iste. Budući da RDF baze podataka podržavaju ontologije kao sheme za model podataka, upravo će ontologija biti korištena kako bi se pokazale mogućnosti opisanih sustava i ukazalo na prednosti ili nedostatke i na razlike između njih.

Primjer ontologije koji će se koristiti je jednostavna ontologija vezana za televizijsku serije Game of Thrones.

Ontologija se sastoji od većeg broja klasa, između ostalog klasa koje označavaju obitelji u seriji, a njihove instance likove iz serije, odnosno članove pojedine obitelji. Na sličan način opisano je koja kraljevstva postoje i slično. Objektna i podatkovna svojstva detaljnije opisuju veze između klasa, primjerice, odnosi između likova, tko pripada kojoj obitelji, tko vlada kojim kraljevstvom i slično.

Na slici ispod može se vidjeti vizualizacija ontologije u alatu WebVOWL, odnosno njen grafički prikaz.



Slika 15 Vizualizacija ontologije u WebVOWL alatu

Zapis ove ontologije u RDF dokumentu je opsežan, pa će biti prikazani samo neki dijelovi dokumenta. Zaglavlje sadržava XML namespace koji su sadržani u dokumentu.

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones"
     xml:base="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:GameOfThrones="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#">
    <owl:Ontology rdf:about="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones">
        <rdfs:comment>Ontologija o seriji Game of Thrones</rdfs:comment>
    </owl:Ontology>
```

Slika 16 Zaglavlje RDF dokumenta

Sljedeći isječak koda prikazuje definiranje i opis klase *Character* kako bi se prikazalo kako se u RDF dokumentu opisuje neka klasa.

```
<owl:Class rdf:about="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#Character">
    <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#TVShow"/>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#belongsTo"/>
            <owl:someValuesFrom rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#House"/>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#hasSigil"/>
            <owl:someValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#hasWords"/>
            <owl:someValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

Slika 17 Klasa u RDF dokumentu

Definirana je klasa *Character* koja je potklasa klase *TVShow* i ima određene restrikcije, odnosno svojstva poput pripadanja nekoj kući (*belongsTo, House*).

Još će biti prikazan opis objektnog svojstva *hasDirewolf*.

```
<owl:ObjectProperty rdf:about="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#hasDirewolf">
    <owl:inverseOf rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#isDirewolfOf"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#Character"/>
    <rdfs:range rdf:resource="http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#Direwolf"/>
</owl:ObjectProperty>
```

Slika 18 Objektno svojstvo u RDF dokumentu

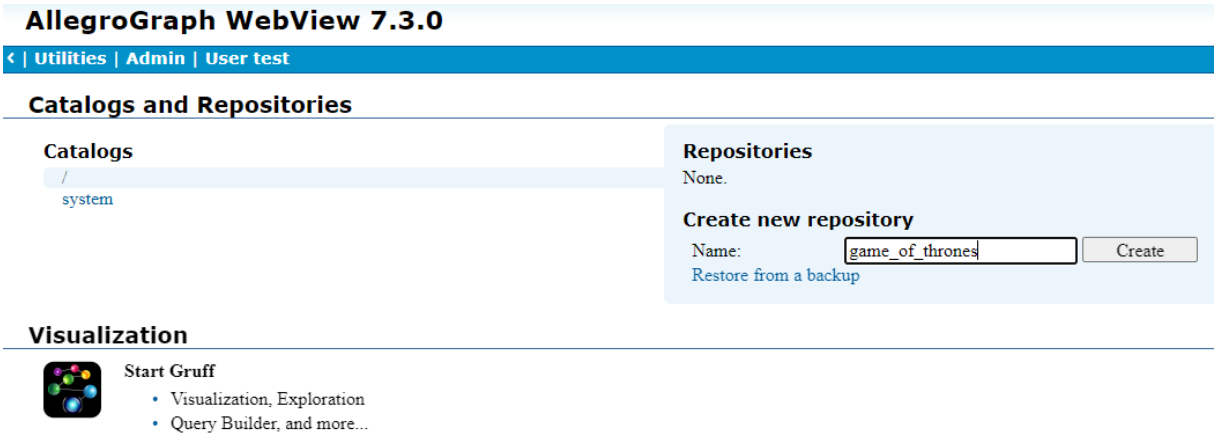
Definirano je koji je tip svojstva, koje mu je svojstvo inverzno te koja je domena i doseg.

Na ovaj način definirane su i ostale klase i objektna te podatkovna svojstva ove ontologije koja će biti korištena kao shema baze podataka.

5.2. Baza podataka u sustavu AllegroGraph

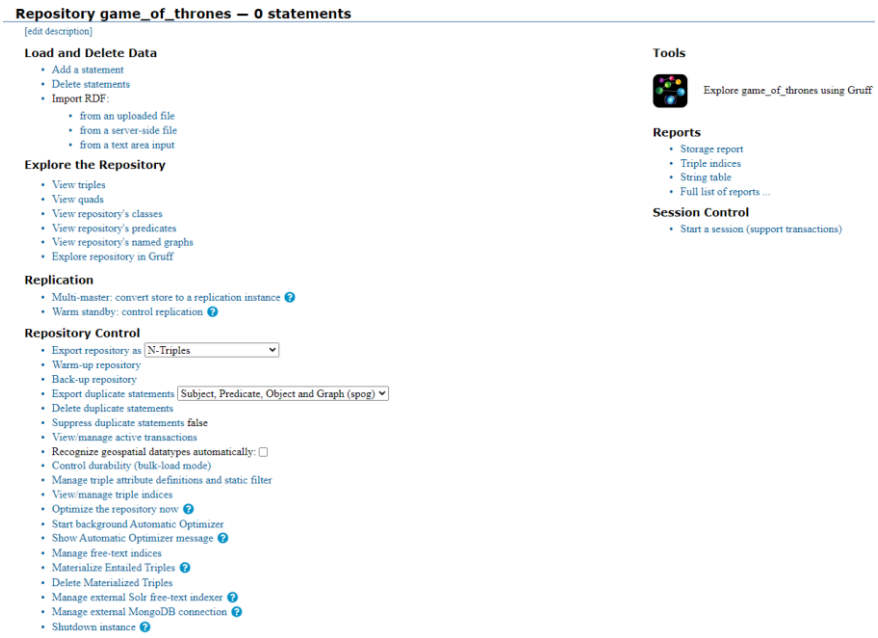
AllegroGraph se nativno pokreće na Linux operacijskom sustavu, ali postoje razni načini na koje mu se može pristupiti i na računalima s Windows operacijskim sustavima. Jedan od njih je i pokretanje sustava u Docker kontejneru, što je učinjeno i u ovom slučaju. Instaliran je Docker Desktop, preuzeta je Docker slika AllegroGraph sustava te je pokrenut Docker kontejner, nakon čega se serveru AllegroGrapha pristupa preko adrese localhost:10035.

Na toj adresi otvara se AllegroGraph WebView, a na početku je potrebno kreirati repozitorij u kojeg ćemo učitati ontologiju i manipulirati podacima.



Slika 19 Prikaz AllegroGraph alata

Nakon što je repozitorij kreiran, otvara se prozor s brojnim mogućnostima, od učitavanja i brisanja podataka do vizualizacije u alatu Gruff.



Slika 20 Prikaz repozitorija u alatu AllegroGraph

Podaci se mogu učitati iz datoteke, dodavanjem trojki ili preko SPARQL upita. Odabirom opcije „Import RDF from an uploaded file“ učitana je .rdf datoteka s 599 trojki, a odabirom opcije „View Triples, prikazuju se sve trojke.

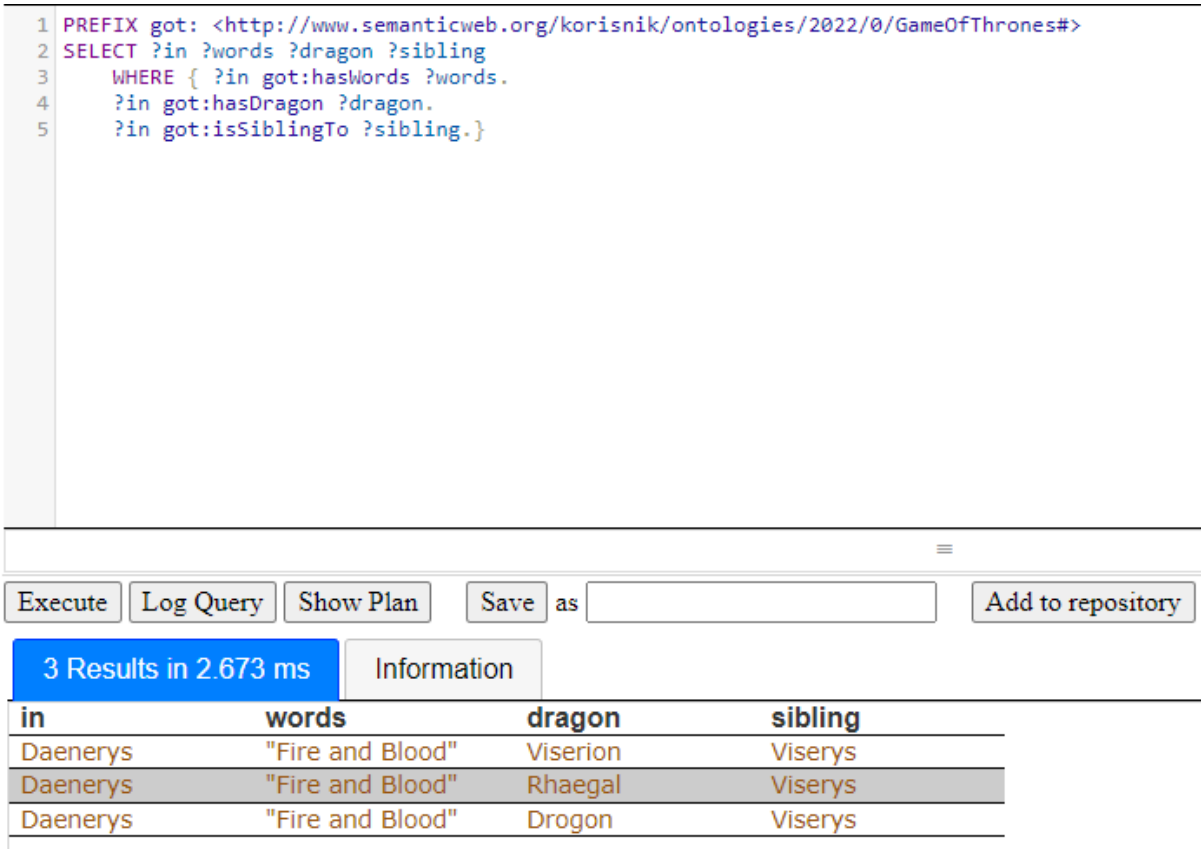
599 Results in 3.073 ms | Information

s	p	o
GameOfThrones	rdf:type	owl:Ontology
GameOfThrones	rdfs:comment	"Ontologija o seriji Game of Thrones"
belongsTo	rdf:type	owl:ObjectProperty
belongsTo	rdf:type	owl:FunctionalProperty
belongsTo	rdfs:domain	Character
belongsTo	rdfs:range	House
hasCapital	rdf:type	owl:ObjectProperty
hasCapital	owl:inverseOf	isCapitalOf
hasCapital	rdf:type	owl:FunctionalProperty
hasCapital	rdfs:domain	Kingdom
hasCapital	rdfs:range	Capital
hasChild	rdf:type	owl:ObjectProperty
hasChild	owl:inverseOf	hasParent
hasDirewolf	rdf:type	owl:ObjectProperty
hasDirewolf	owl:inverseOf	isDirewolfOf
hasDirewolf	rdf:type	owl:FunctionalProperty
hasDirewolf	rdfs:domain	Character

Slika 21 Prikaz trojki u alatu AllegroGraph

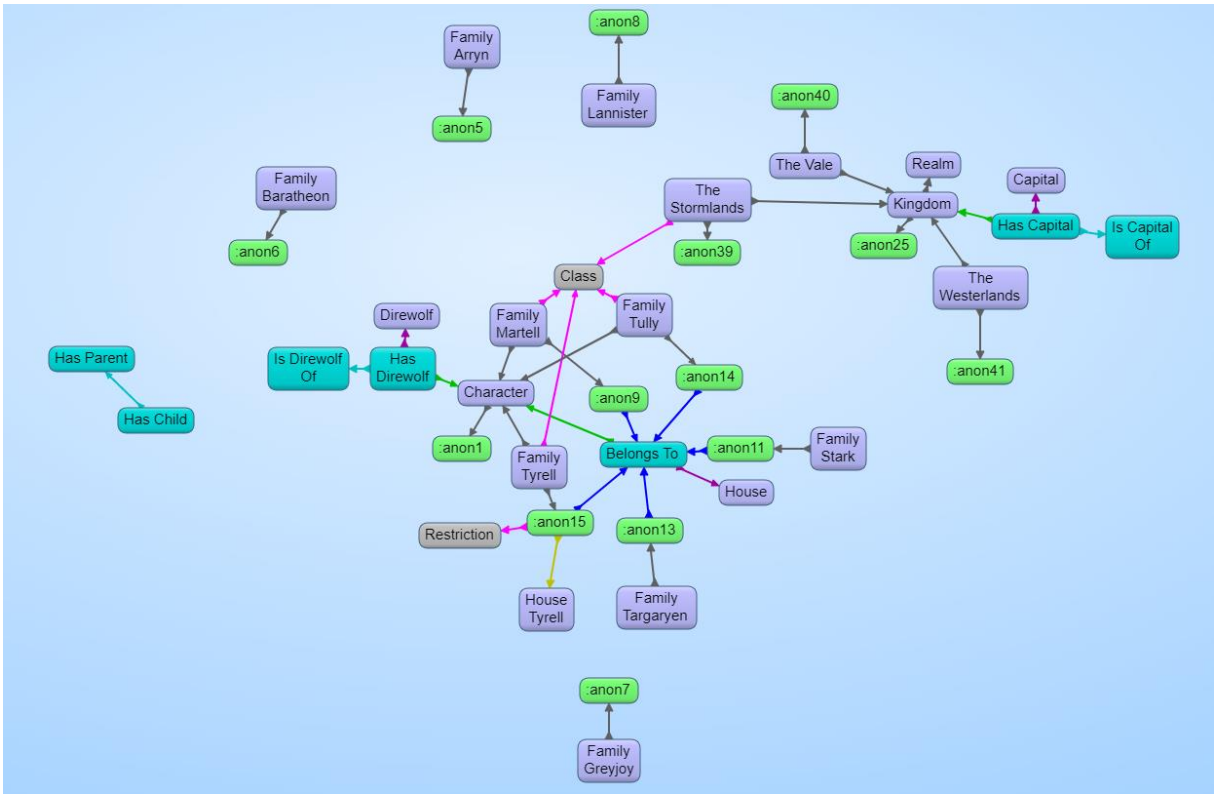
U AllegroGraphu upiti se mogu pisati u SPARQL, GraphQL i Prolog upitnim jezicima.

Na slici ispod može se vidjeti jedan SPARQL upit s rezultatima.



Slika 22 SPARQL upit u alatu AllegroGraph

Također je moguće vizualizirati trojke kao graf pomoću alata Gruff koji nudi mnoštvo opcija za istraživanje trojki preko grafičkog prikaza. Slijedi prikaz dijela podataka u alatu Gruff.

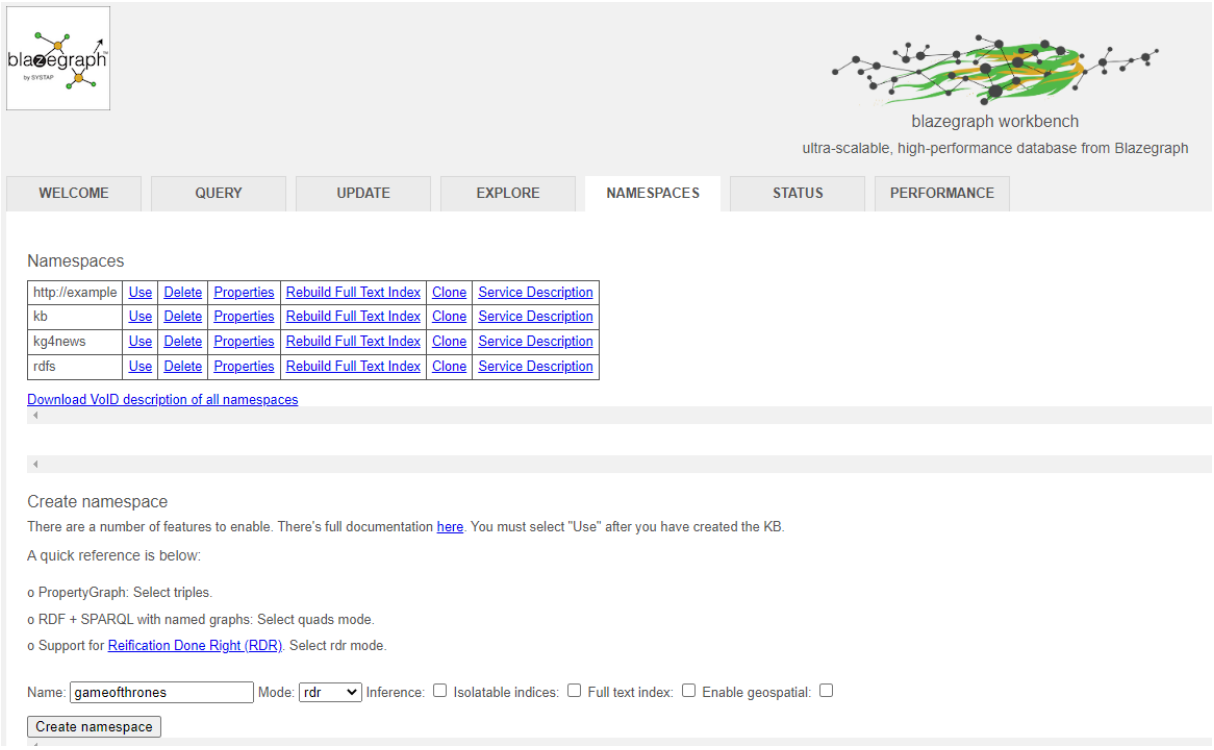


Slika 23 Vizualizacija trojki u alatu Gruff

AllegroGraph doista je moćan alat za RDF baze podataka, jer nudi razne način unosa podataka, odnosno izjava, tri upitna jezika za postavljanje upita, vizualizaciju podataka u obliku grafa i razne druge mogućnosti za upravljanje trojkama u repozitoriju.

5.3. Baza podataka u sustavu Blazegraph

Blazegraph kao opciju ima Blazegraph Workbench kojem se jednostavno u pretraživaču pristupi preko web adrese. Na početnoj stranici naglašeno je da je potrebno kreirati namespace prije nego se podaci mogu uvesti, stoga je kreiran namespace imena gameofthrones.



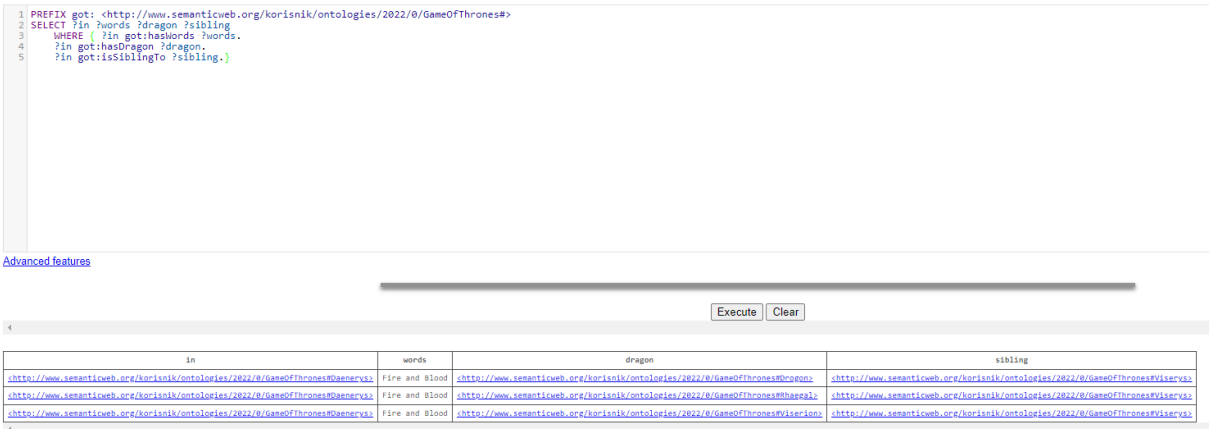
Slika 24 Kreiranje namespacea u alatu Blazegraph

Nakon što se kreira namespace, potrebno je učitati podatke, a za to treba otići na tab Update, gdje je moguće učitati RDF datoteku, napraviti SPARQL update ili unijeti putanju do neke datoteke ili URL.

U ovom slučaju učitana je ista .rdf datoteka kao i u AllegroGraph. Nakon toga mogu se početi postavljati upiti. Na slici 21. može se vidjeti dio rezultata upita kojim su prikazane sve trojke. Kao što se može vidjeti, u Blazegraphu rezultati su prikazani s IRI-jima, za razliku od rezultata u AllegroGraphu, što ih čini malo teže čitljivima.

Slika 25 Prikaz trojki u alatu BlazeGraph

Također je postavljen isti upit kao i u AllegroGraphu kako bi se usporedili rezultati, koji su, naravno jednaki, ali kao što je već spomenuto, teže čitljivi.

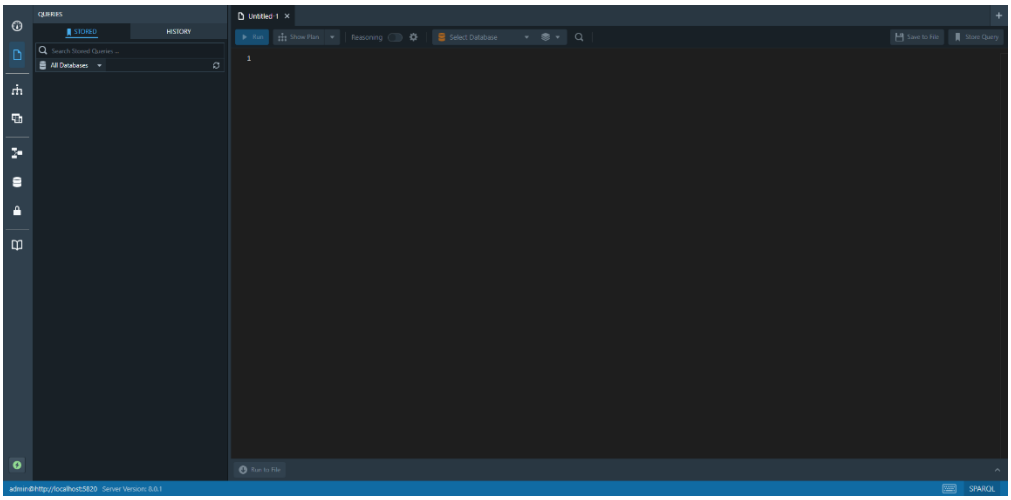


Slika 26 Rezultat upita u alatu Blazegraph

Blazegraph je prilično jednostavan za rad i na prvu djeluje kao da ne nudi puno opcija, ali ipak je moguće vrlo brzo učitati i pretraživati trojke, kao i vršiti upite. Korisno je i što pri pisanju upita postoje namespace shortcuti, kako bi se olakšalo pisanje koda. Ono što je nedostatak je zasigurno izostanak vizualizacije pomoću grafa kojeg većina alata za RDF baze podataka ima. Iako mogućnost grafičkog prikaza nije nužna, zasigurno je praktična i olakšava rad s podacima.

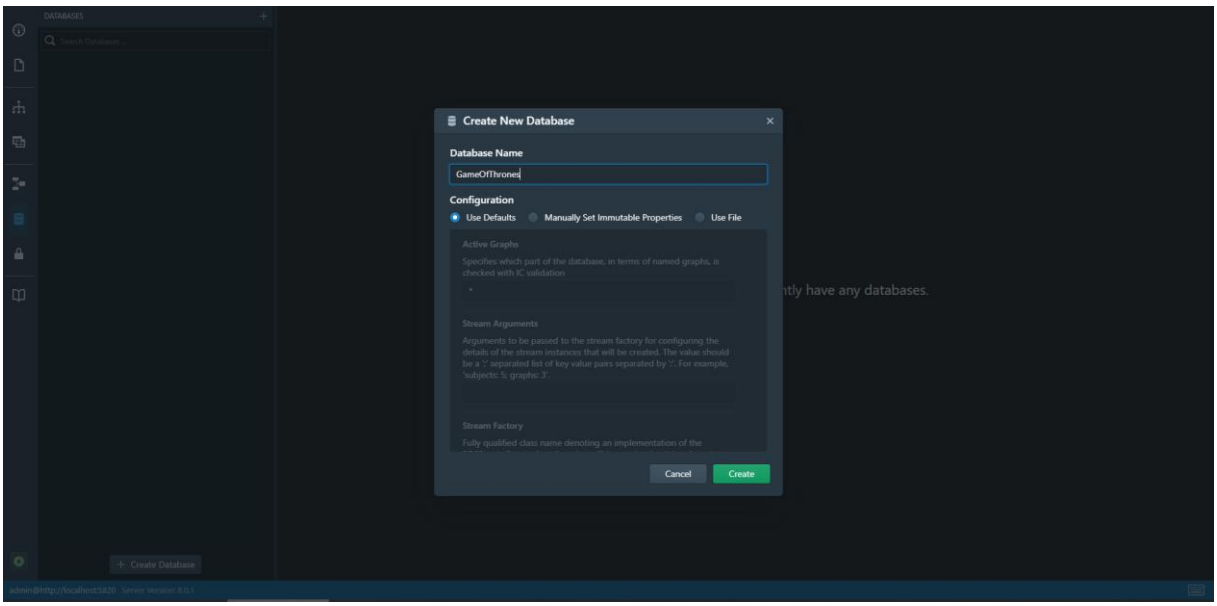
5.4. Baza podataka u sustavu Stardog

Za pokretanje Stardoga bilo je potrebno spustiti i licencu osim ostalih instalacijskih datoteka i uraditi razne konfiguracije i pokrenuti server, zatim pristupiti na Stardog Studio i povezati se sa serverom. Otvara se početno sučelje Stardog Studia koje odmah na prvi pogled djeluje ljepše i funkcionalnije od prethodnih sustava i lako je za upravljanje.



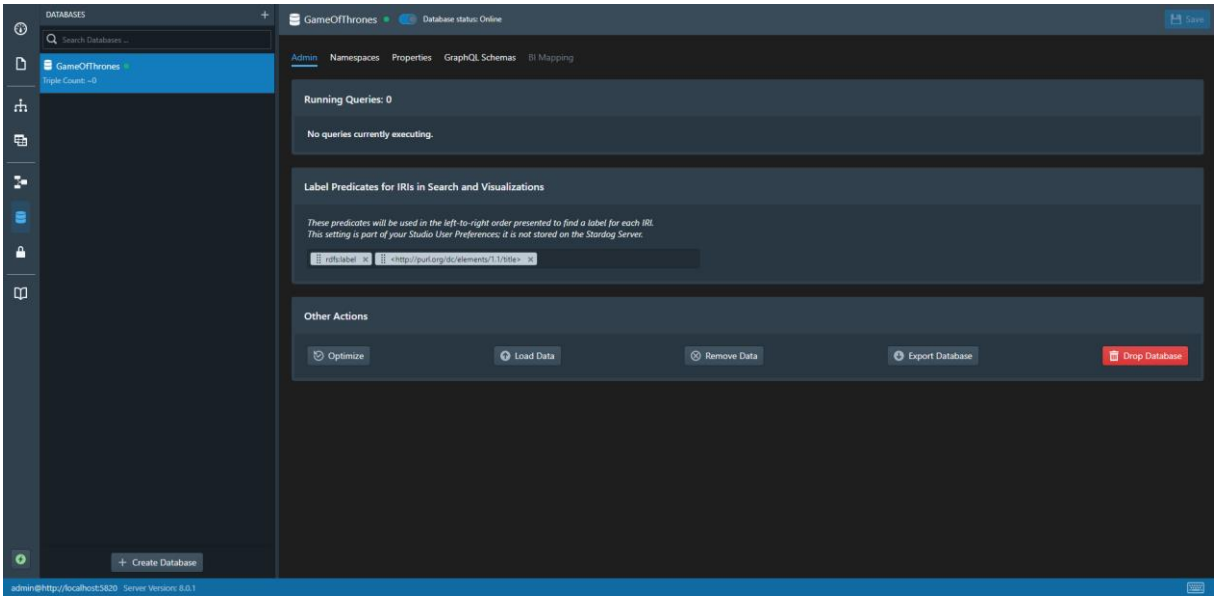
Slika 27 Početno sučelje Stardog Studia

Kako bi se kreirala baza podataka, potrebno je iz lijevog izbornika odabrati ikonicu baze podataka i kliknuti na *Create Database* nakon čega se otvara prozor prikazan na slici ispod.



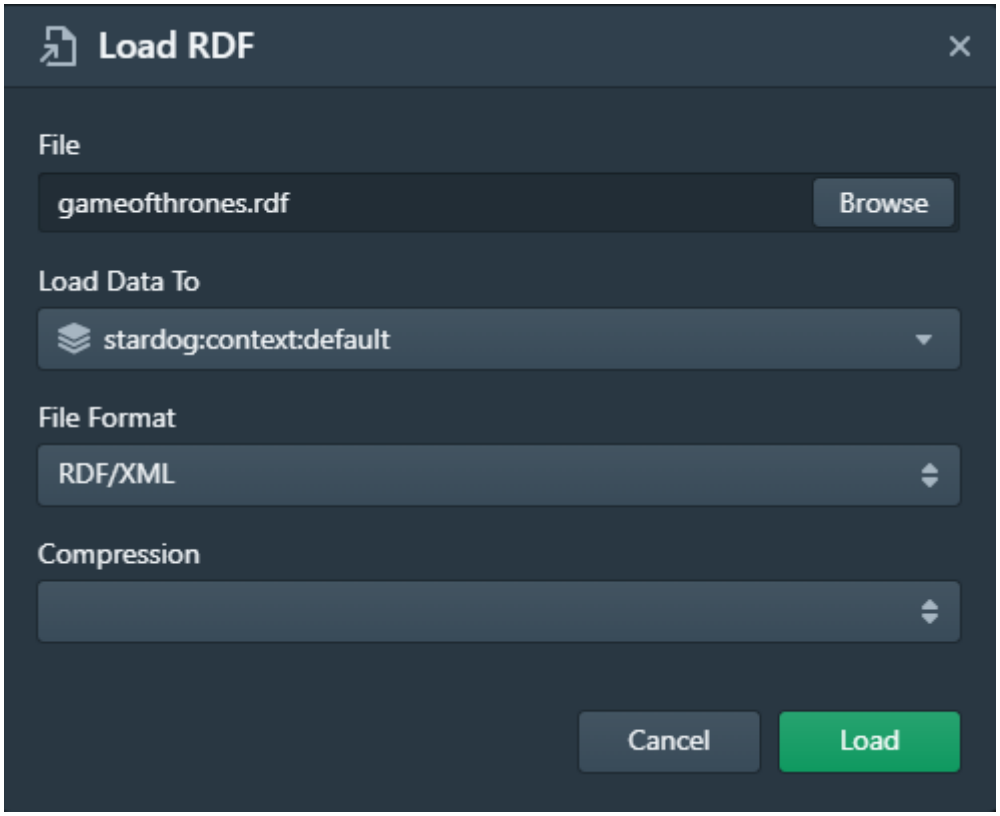
Slika 28 Kreiranje baze podataka u Stardog Studiu

Sad je kreirana baza podataka imena *GameOfThrones* koja je prazna, a kako bi se učitali podaci, potrebno je kliknuti na gumb Load Data.



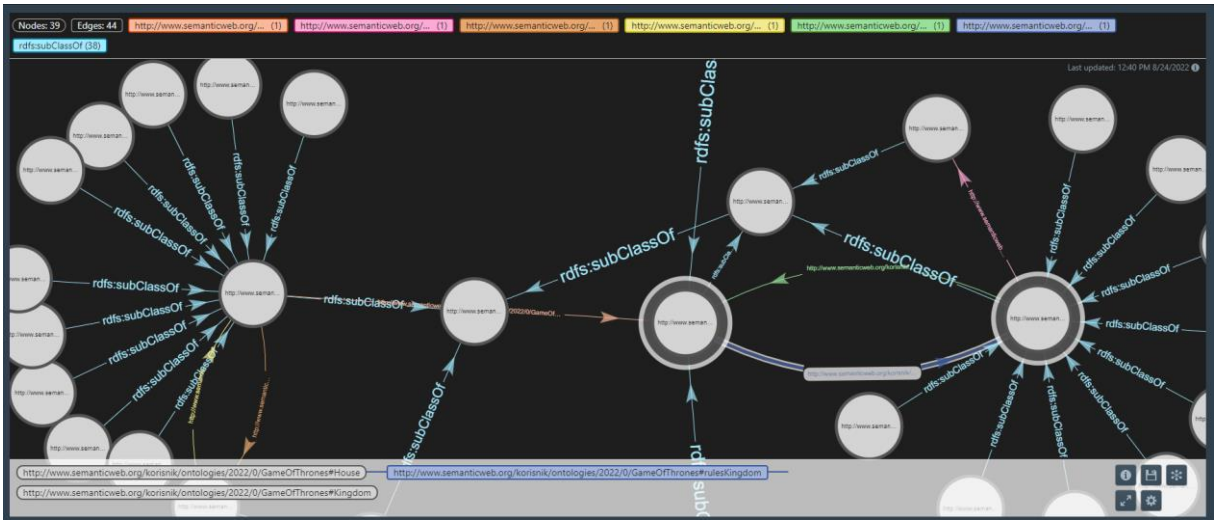
Slika 29 Kreirana baza podataka u Stardog Studiu

Odabrana je .rdf datoteka i učitani su podaci te sada baza podataka sadrži 599 trojki.



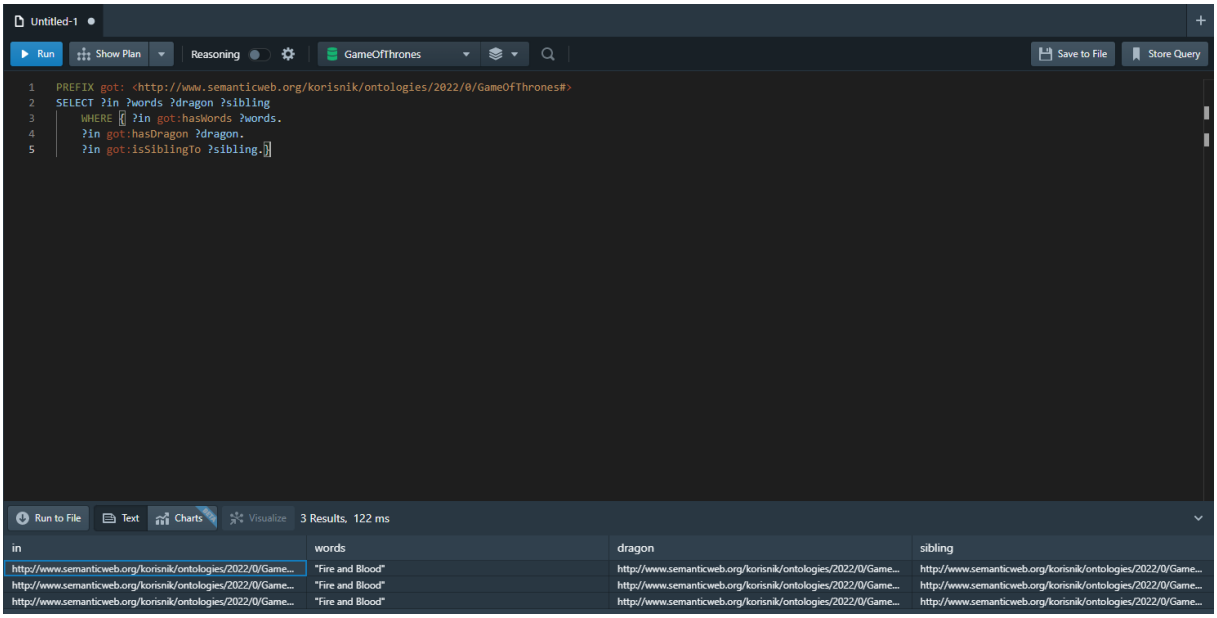
Slika 30 Učitavanje podataka u bazu u Stardog Studiu

Stardog nudi i moćnu opciju vizualizacije grafa baze podataka pri čemu se lako kretati po grafu i vidjeti pojedine veze između čvorova.



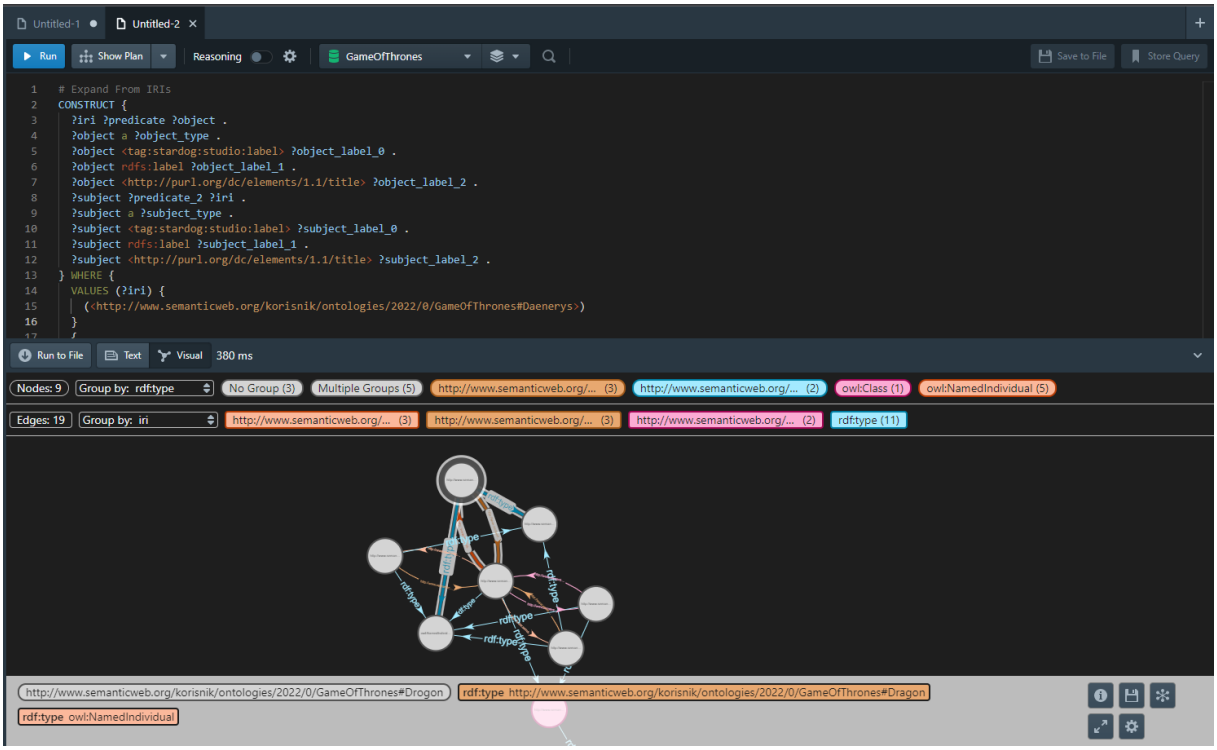
Slika 31 Vizualizacija grafa baze podataka u Stardog Studiu

Pri izvršavanju SPARQL upita rezultati se mogu prikazati tekstualno u obliku tablice i vizualno pomoću grafa. Na slici ispod prikazan je rezultat upita postavljenog i u prethodna dva sustava.



Slika 32 Izvršavanje upita u Stardog Studiu

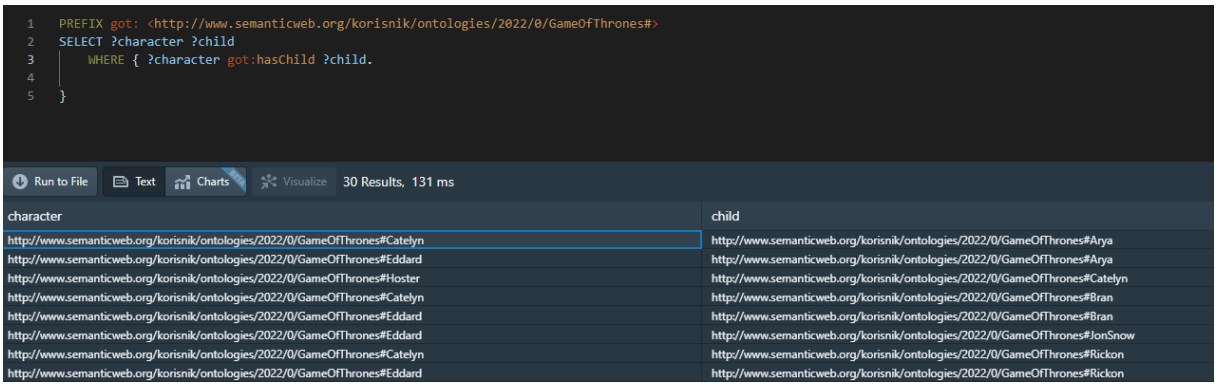
I u Stardogu rezultati se prikazuju sa IRI-jem što ih čini teže čitljivima u odnosu na AllegroGraph. Kako bi se rezultat prikazao u obliku grafa, potrebno je kliknuti na bilo koji IRI u tablici, zatim na *Visualize*.



Slika 33 Rezultati upita u obliku grafa u Stardog Studiu

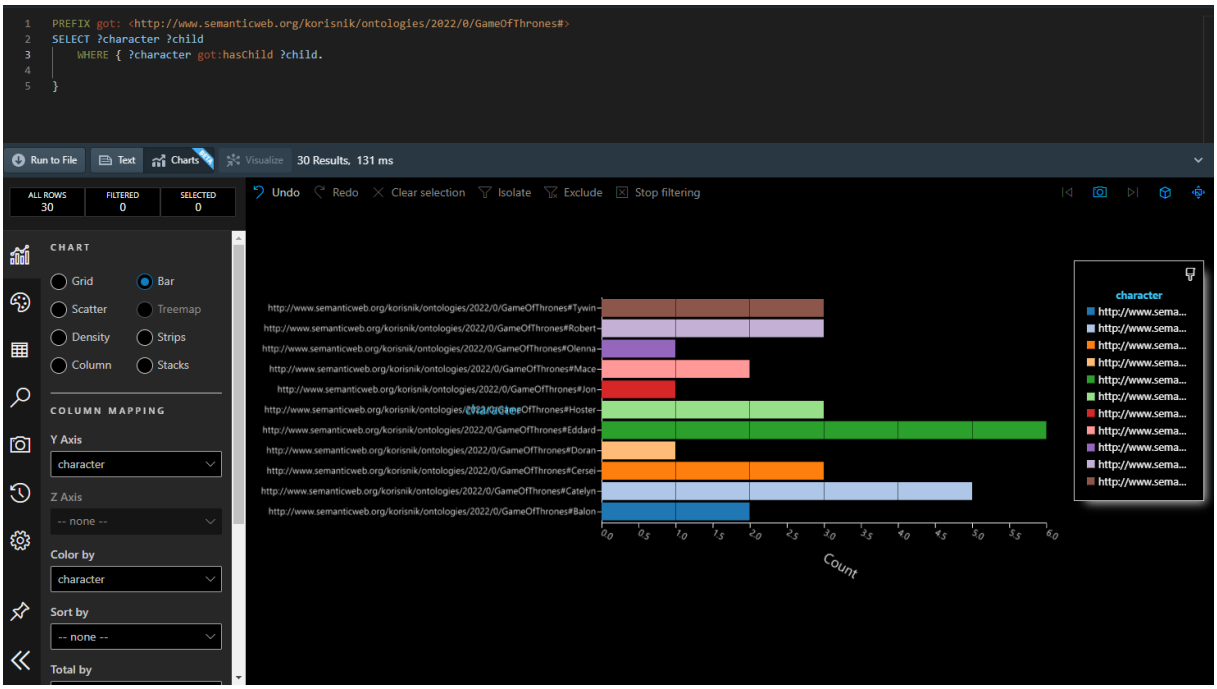
Kao i kod grafa na kojem su prikazani svi podaci, vidljiv je broj čvorova i veza, a moguće ih je grupirati na više načina. Kretanjem po grafu i klikom na čvorove i veze dobije se detaljniji uvid u rezultate.

Stardog nudi i opciju izrade grafikona s obzirom na rezultate upita što može biti vrlo korisno. Na slici ispod može se vidjeti upit kojim se prikazuju likovi u seriji i njihova djeca.



Slika 34 Izvršavanje drugog upita u Stardog Studiu

Odabere li se opcija *Charts*, nudi se više različitih grafikona kojima se mogu prikazati dobiveni rezultati. Kao primjer je naveden grafikon na kojem je prikazano koliko koji lik ima djece.

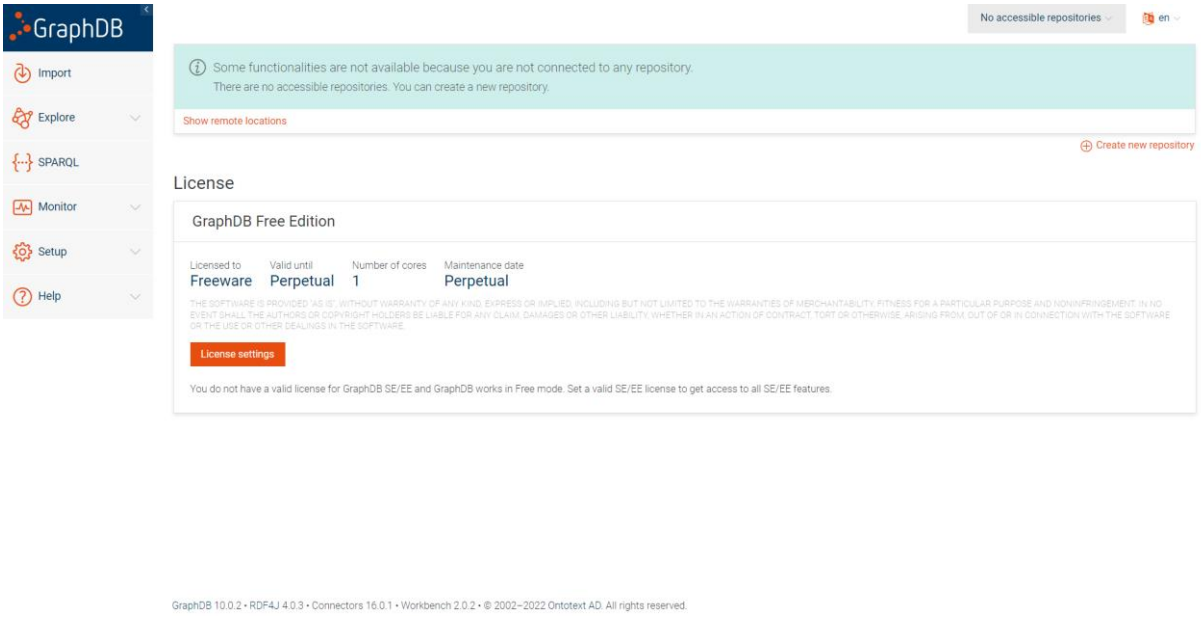


Slika 35 Grafikon s obzirom na rezultate upita u Stardog Studiu

Ova opcija može biti vrlo korisna za slikovito prikazivanje velikog broja trojki, osobito u svrhu korištenja podataka za izradu različitih statistika i provedbu istraživanja.

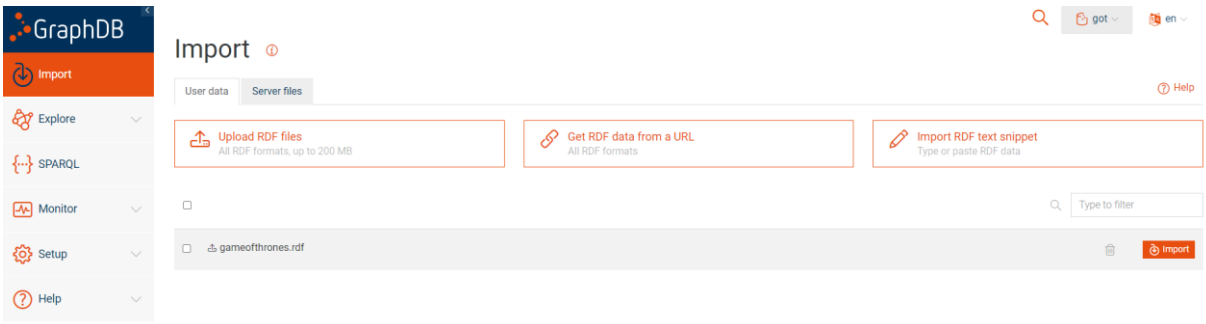
5.5. Baza podataka u sustavu GraphDB

Za potrebe demonstracije rada s bazom u alatu GraphDB korištena je besplatna verzija alata koju je lako preuzeti sa službene stranice, nakon ispunjavanja traženog obrasca. GraphDB Workbench otvara se na adresi localhost:7200.



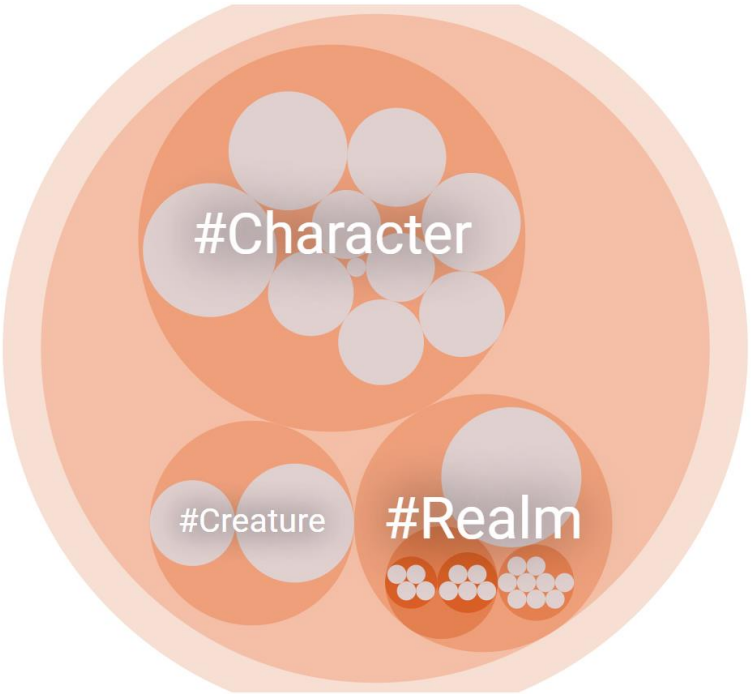
Slika 36 Početno sučelje GraphDB

Izbornik na lijevoj strani nudi opcije učitavanja podataka, istraživanja podataka pomoću vizualizacije grafovima, postavljanje SPARQL upita i raznih postavki. Prije svega je potrebno kreirati novi repozitorij, a nakon toga učitati podatke iz datoteke, u ovom slučaju .rdf datoteke otprije.



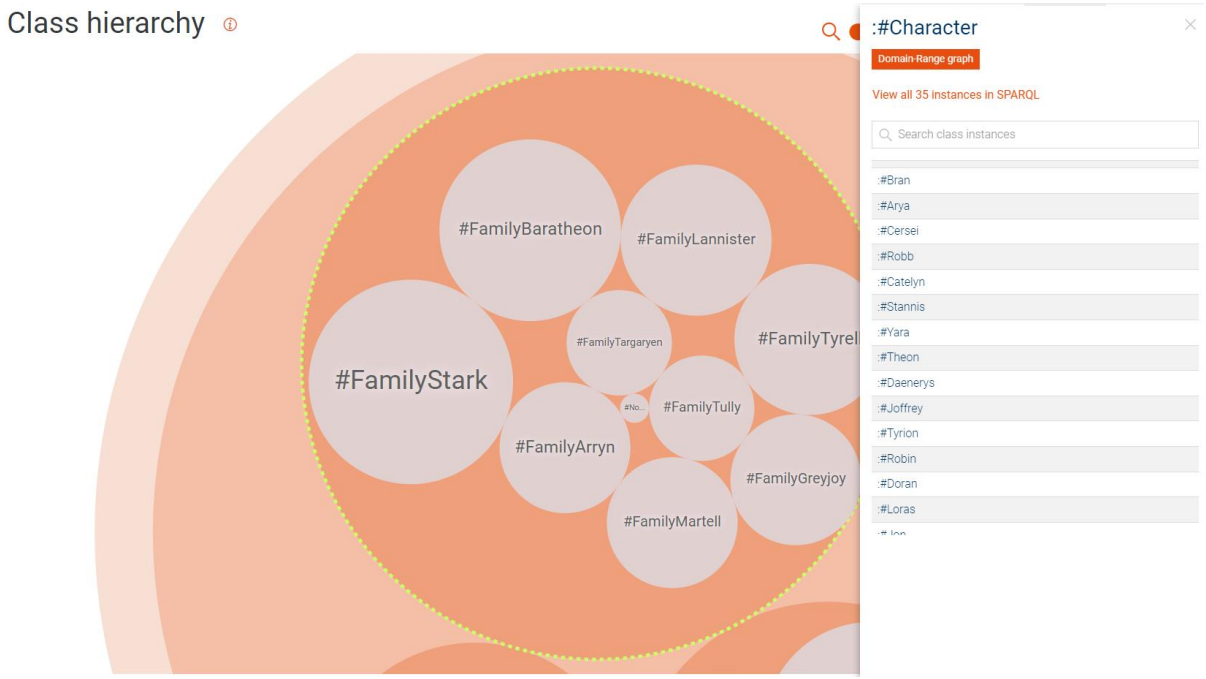
Slika 37 Učitavanje RDF datoteke u GraphDB

GraphDB nudi dosta opcija kad je u pitanju vizualizacija podataka pomoću grafa. Moguće je grafički pregledavati hijerarhiju klasa koje se nalaze u ontologiji, odnosno u RDF/XML dokumentu.



Slika 38 Hijerarhija klasa u grafičkom prikazu u GraphDB

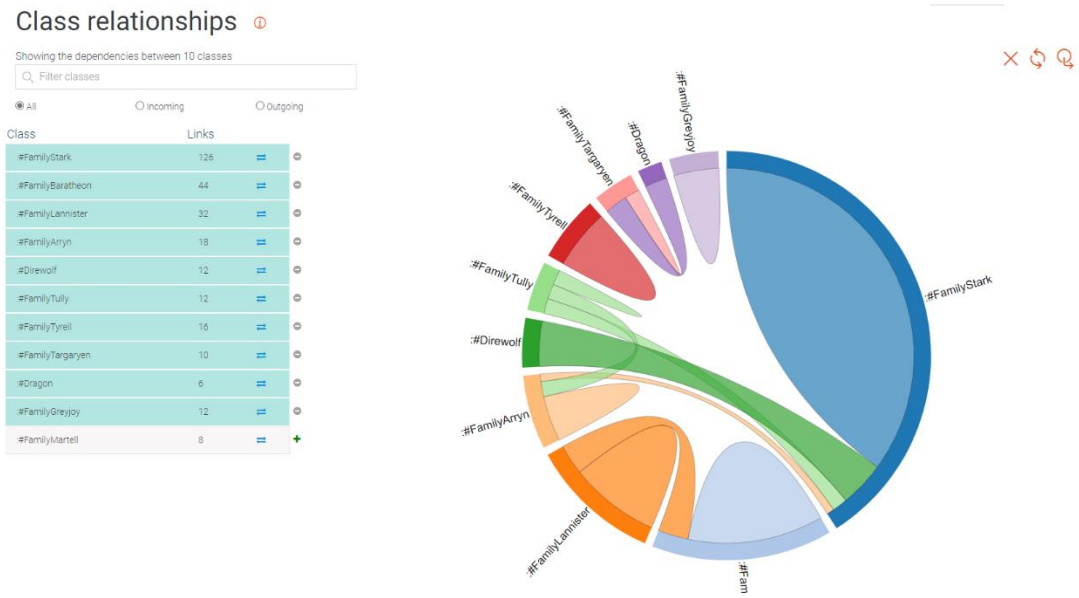
Fokusirajući se u pojedine klase, mogu se vidjeti njezine potklase, a klikom na neku od njih vide se i njezine instance.



Slika 39 Detalji klase u hijerarhijkom prikazu u GraphDB

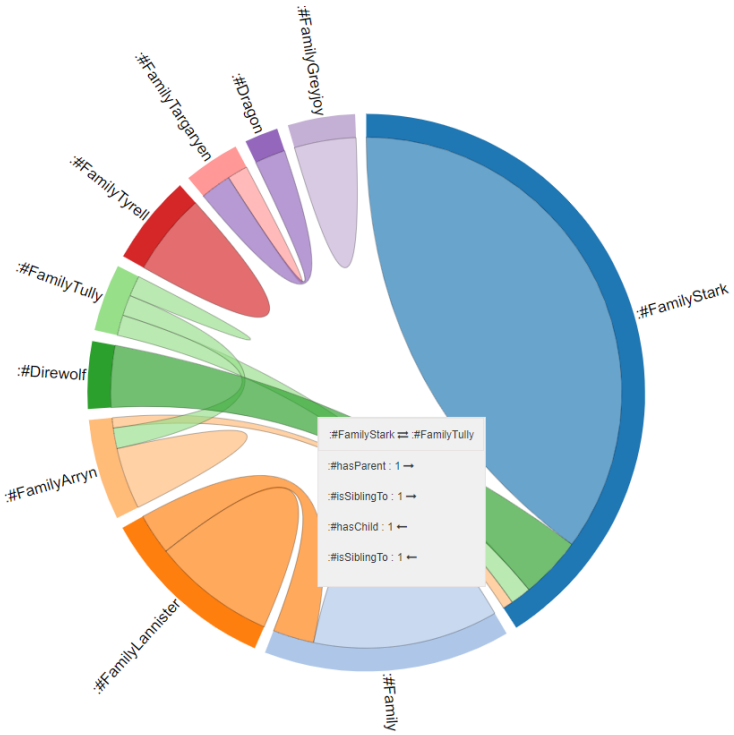
Ova opcija nudi uvid u način na koji je baza podataka izgrađena na slikovit, intuitivan i pregledan način, osobito kad se radi o kompleksnijoj bazi podataka s velikim brojem trojki.

Osim klasa, moguće je grafički vizualizirati i odnose, odnosno veze, među pojedinim klasama. Grafički je različitim bojama prikazano koje klase su u međusobnom odnosu, a vidi se i broj veza kojeg pojedina klasa ima.



Slika 40 Grafički prikaz veza u GraphDB

Klikne li se li na neku od veza na grafu, može se vidjeti točno kojim su svojstvima povezane neke klase i koliko takvih veza postoji.



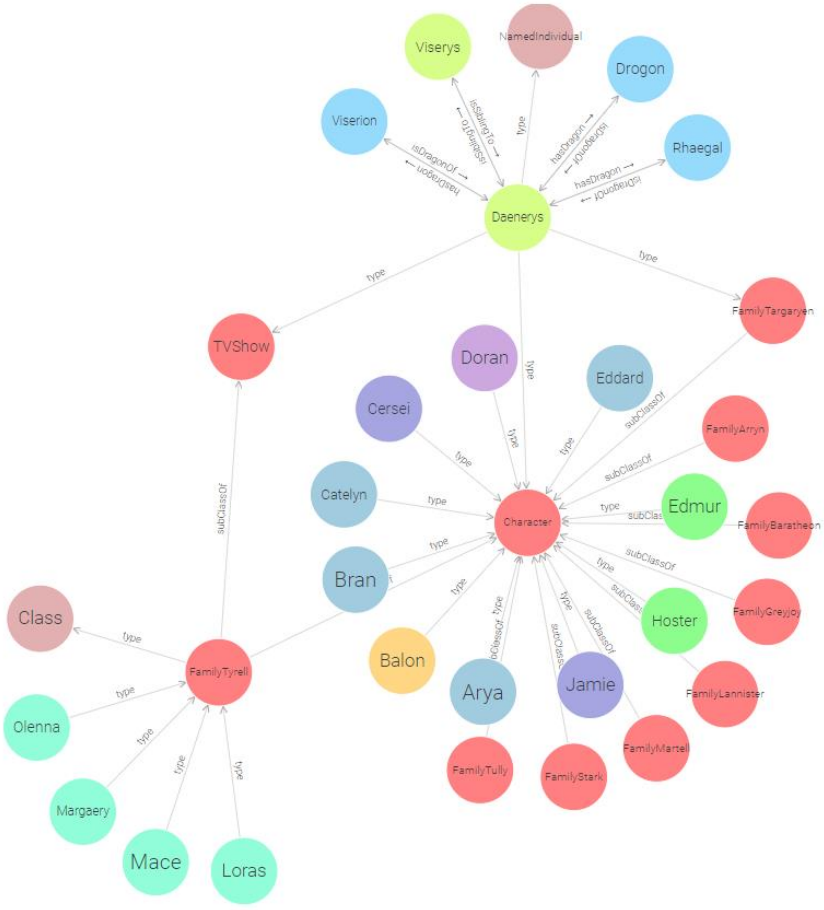
Slika 41 Detaljniji prikaz veza u GraphDB

Čvorove i veze moguće je grafički prikazati tako što se upiše link točno određenog resursa iz baze podataka kojeg se želi grafički prikazati. Na slici ispod može se vidjeti resurs *Character* prikazan kao graf.

Slika 42 Grafički prikaz klase *Character*

Graf se može proširiti tako što se proširi pojedini čvor pa se na grafu prikažu i veze koje taj čvor ima s drugim čvorovima u grafu. Na slici 43 prošireni su čvorovi *Daenerys* i *Family Tyrell* na grafu prikazanom na slici 42.

Na ovaj način se mogu proširiti svi čvorovi i tako se vizualno prikazati cijela RDF baza podataka, međutim, to izgleda dosta nepregledno i sve veze na grafu su ispresijecane pa je lakše trojke pretraživati po grafu tako što se upiše koji točno resurs želimo prikazati, a onda proširivati čvorove po potrebi.



Slika 43 prošireni grafički prikaz klase *Character*

U GraphDB upiti se također pišu u SPARQL jeziku. Kao i kod alata AllegroGraph, rezultati upita su čitko napisani bez IRI-ja, što olakšava pretraživanje rezultata.

```
PREFIX got: <http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#>
SELECT ?in ?words ?dragon ?sibling
    WHERE { ?in got:hasWords ?words.
    ?in got:hasDragon ?dragon.
    ?in got:isSiblingTo ?sibling.}
```

Run

Press Alt+Enter to autocomplete

Table | Raw Response | Pivot Table | Google Chart | Download as

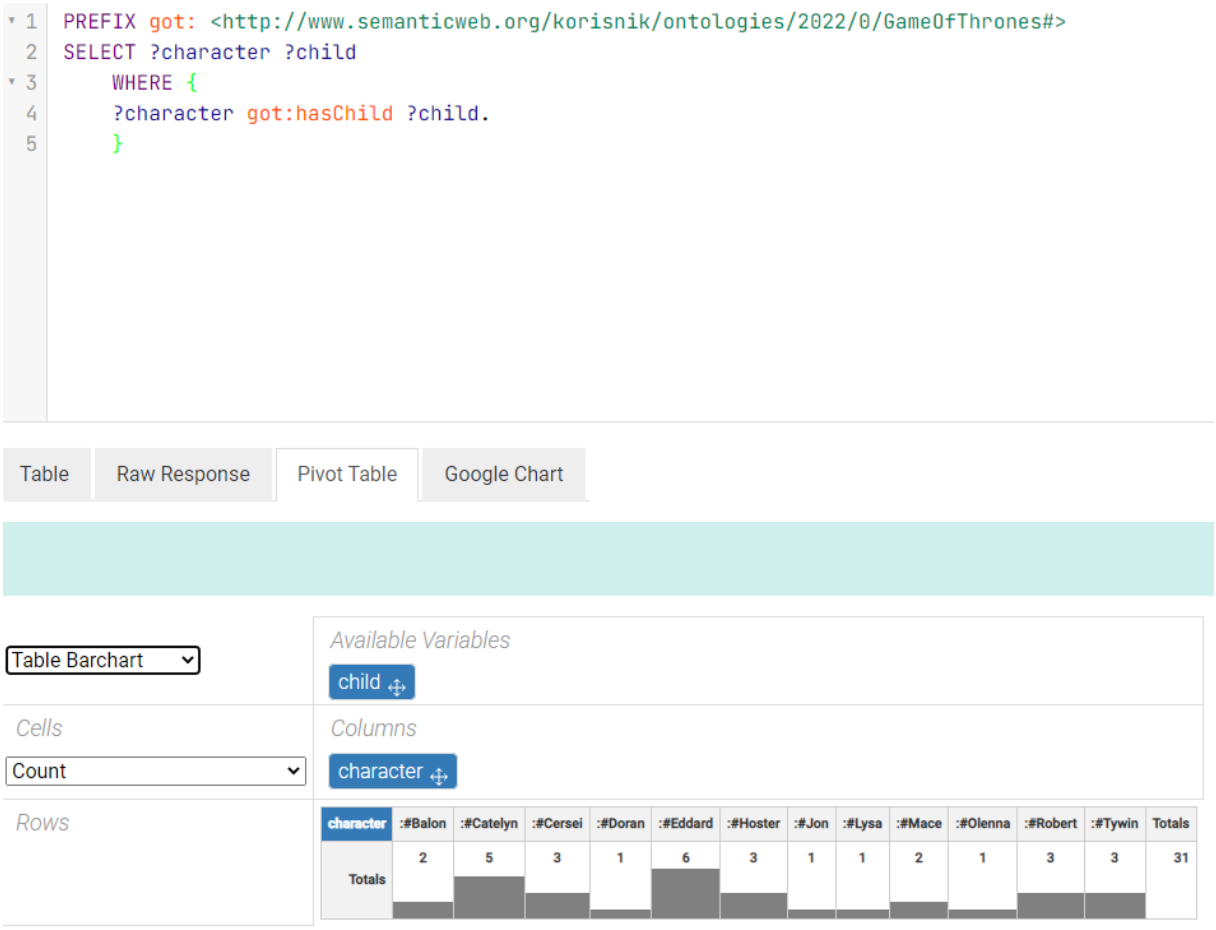
Filter query results

Showing results from 1 to 3 of 3. Query took 0.1s, minutes ago.

	in	words	dragon	sibling
1	:#Daenerys	"Fire and Blood"	:#Drogon	:#Viserys
2	:#Daenerys	"Fire and Blood"	:#Rhaegal	:#Viserys
3	:#Daenerys	"Fire and Blood"	:#Viserion	:#Viserys

Slika 44 Izvršavanje upita u alatu GraphDB

Slično kao i kod Stardoga, rezultati upita mogu se prikazati na više načina, a mogu se i prikazati pomoću tablica u kojima se podaci broje, zbrajaju i sl. s obzirom na varijable deklarirane u upitu. Na slici ispod prikazana je tablica za isti primjer upita kao i kod demonstracije grafikona kod Stardoga.



Slika 45 Pivot tablica s obzirom na rezultate upita u GraphDB

Stardog je možda bogatiji kad su u pitanju vrste grafikona i odabir onoga što će se prikazati i na koji način, međutim i GraphDB zasigurno pruža korisnu opciju za manipuliranje podacima i analiziranje istih.

5.6. Usporedba sustava s obzirom na primjer

Nakon što je ista ontologija učitana u četiri navedena sustava i isti upiti su napravljeni nad bazom podataka, sustavi se mogu usporediti na više načina. Što se tiče same instalacije sustava i lakoće pristupa, Blazegraphu je najlakše pristupiti jer je jedini Open Source dok su ostali alati komercijalni. Blazegraph Workbench je lako dostupan u browseru, dok se u ovom primjeru za AllegroGraph prvo preuzeo Docker Desktop i Docker slika AllegroGraph sustava

nakon čega je pokrenut Docker kontejner i serveru AllegroGrapha se pristupilo preko adrese localhost:10035.

Za Stardog bilo je potrebno napraviti određene konfiguracije na računalu, preuzeti instalacijske datoteke, ali i licencu te pokrenuti Stardog Server na lokalnom računalu i povezati se sa serverom pomoću Stardog Studia. Za GraphDB korištena je besplatna verzija alata koju je moguće preuzeti sa službene stranice, ali prije toga je potrebno ispuniti obrazac.

Kako bi se vidjelo koliko brzo sustavi izvršavaju upite, napravljena su tri upita u svakom sustavu. Prvo je izvršen upit kojim se prikazuju sve trojke u sustavu, njih 599:

```
PREFIX                                                                  got:
<http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#>

SELECT * WHERE {?s ?p ?o}
```

Zatim je izvršen upit prikazan i na slikama gdje se pretražuju individue s obzirom na određene uvjete odnosno svojstva koja imaju:

```
PREFIX                                                                  got:
<http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#>

SELECT ?in ?words ?dragon ?sibling

    WHERE { ?in got:hasWords ?words.

    ?in got:hasDragon ?dragon.

    ?in got:isSiblingTo ?sibling.}
```

Nakon toga je načinjen upit s filtriranjem kako bi se vidjelo koliko brzo u tom slučaju sustav pronađe traženi rezultat:

```
PREFIX                                                                  got:
<http://www.semanticweb.org/korisnik/ontologies/2022/0/GameOfThrones#>

SELECT ?character ?words

WHERE {?character got:hasWords ?words.

       FILTER (?words="Hear me roar")

}
```

U tablici ispod može se vidjeti brzina izvršavanja upita za svaki sustav. Prilikom izvršavanja upita, u svakom sustavu se uz rezultate upita prikazuje i vrijeme koje je prošlo da se rezultati dohvate, a upravo to vrijeme je prikazano u tablici. Budući da je u GraphDB vrijeme

prikazano u sekundama, u tablici su prikazane zaokružene vrijednosti u milisekundama. Upit 1 označava upit kojem su dohvaćene sve trojke, upit 2 označava upit kojim se traži koja individua koja ima uvjetom zadana svojstva, a to su *words, dragon* i *sibling*, a upit 3 označava upit u kojem je upotrijebljena ključna riječ FILTER za pronalaženje tražene individue, odnosno lika.

Tablica 2 Brzina izvršavanja upita

	AllegroGraph	**Blazegraph**	**Stardog**	**GraphDB**
Upit 1	3.073 ms	619 ms	118 ms	200 ms
Upit 2	2.673 ms	145 ms	122 ms	100 ms
Upit 3	4.764 ms	152 ms	126 ms	100 ms

Prema rezultatima prikazanima u tablici, AllegroGraph ima uvjerljivo najbrže vrijeme izvršavanja upita, dok Blazegraph ima najsporije. Iako se radi o milisekundama, kad se usporede AllegroGraph i Blazegraph, razlika je poprilično velika.

Što se tiče samog prikaza rezultata upita, na slikama se moglo vidjeti kako AllegroGraph i GraphDB rezultate prikazuju bez IRI-ja, dok Stardog i Blazegraph prikazuju i IRI. Kad se govori o čitljivosti rezultata, prednost imaju AllegroGraph i GraphDB, ali IRI također sadrži važne informacije o resursu, pa se ne može reći da je prikazivanje resursa s IRI-jem nedostatak u tom smislu.

Budući da je RDF model podataka zapravo graf, bitna značajka sustava za upravljanje RDF bazama podataka je i vizualizacija istih pomoću grafa. Blazegraph je po tome u nedostatku u odnosu na ostale sustave jer nema tu mogućnost. AllegroGraph ima mogućnost grafičkog prikaza pomoću alata Gruff gdje se kretanjem po grafu mogu vidjeti resursi i svojstva i odnosi među čvorovima, kreirati ili brisati trojke itd. U Stardogu se model podataka također može prikazati u obliku grafa, kao i rezultati samog upita. Osim prikaza rezultata u obliku grafa, moguće je izraditi različite grafikone s obzirom na rezultate, kao što je prikazano na slici 35. To je zasigurno prednost Stardoga jer ta opcija može biti vrlo korisna ne samo za iščitavanje nekih rezultata, već i za upotrebu u provođenju nekih istraživanja ili izradi statistike za određenu svrhu.

Kad se radi o vizualnom prikazu podataka, najviše opcija ima GraphDB. Moguće je vizualno prikazati hijerarhiju klasa, veze između njih, domene i dosege, kojim su točno svojstvima povezane neke klase i slično. Upisivanjem IRI-ja točno određenog resursa iz baze podataka dobiva se grafički prikaz čvorova i veza, a svaki čvor se može proširiti i na taj način

se dobiti kompletan graf modela podataka. U GraphDB se također mogu izraditi tablice nalik na grafikone gdje se mogu sistematizirati neki podaci, međutim u tom području je Stardog znatno moćniji.

6. Zaključak

RDF je preporuka W3C-a (*World Wide Web Consortium*) od 2004. godine i to je okvir za opis podataka o web resursima dizajniran da bude čitljiv i razumljiv računalu i pisan je u XML-u. Ima raznoliku upotrebu, a temeljni koncepti su resursi, svojstva i izjave.

Izjave se sastoje od subjekta (resurs), predikata (svojstvo) i objekta (vrijednost svojstva; može biti resurs). RDF model podataka sastoji se od tih izjava koje se još nazivaju trojke (engl. *triple*), a može se prikazati pomoću grafa.

RDF Schema definira vokabular i značenje podataka za RDF, odnosno pruža osnovni tipski sustav za korištenje u RDF modelima, a bitni koncepti su klase i svojstva.

RDF baze podataka tip su grafovskih baza podataka i temelje se na RDF-u, a pogodne su za upravljanje velikim brojem podataka s velikim brojem veza te su jako fleksibilne, a kao upitni jezik koristi se SPARQL.

U ovom radu opisana su četiri sustava za upravljanje RDF bazama podataka – AllegroGraph, Blazegraph, Stardog i GraphDB. Jedni su od najpopularnijih i najkorištenijih sustava za upravljanje RDF bazama podataka, a razlog tome je što imaju sposobnost da jako brzo obrađuju velike količine trojki i izvršavaju upite. Bilo je jednostavno snaći se u alatima i svaki omogućuje učitavanje trojki ili kreiranje istih na brz i jednostavan način, kao i postavljanje upita u SPARQL-u. Svojevrsni nedostatak kojeg ima Blazegraph u odnosu na ostala tri opisana sustava je taj da nema sposobnost vizualizacije podataka u obliku grafa, što je vrlo korisno i pomaže da se podaci lakše i brže pregledavaju te stvara jasniju sliku o samoj bazi podataka.

Osobito su se istakli Stardog i GraphDB zbog korisniku privlačnog i funkcionalnog sučelja te jako dobrim grafičkim prikazima podataka, dok AllegroGraph i GraphDB rezultate upita prikazuju bez IRI-ja, što je mala prednost zbog čitljivosti, ali ne može se reći da je to nedostatak druga dva sustava jer IRI sadrži bitne informacije o resursu. GraphDB pružio je najviše mogućnosti kad se radi o vizualizaciji podataka, dok se AllegroGraph ističe kao sustav koji ima znatno brže vrijeme izvršavanja upita u odnosu na ostale sustave.

RDF baze podataka ističu se zbog svoje fleksibilnosti i mogućnosti da se pomoću trojki opiše bilo što u stvarnom svijetu pa i apstraktni pojmovi. Iako na prvu djeluju apstraktno, osobito u odnosu na relacijske baze podataka, zapravo mogu biti jednostavne i intuitivne. Budući da popularnost NoSQL baza podataka raste zbog sve veće potrebe za pohranjivanjem i upravljanjem velikih količina podataka, nema razloga pomisliti kako RDF baze podataka neće biti u sve široj upotrebi.

Popis literature

[1] J. Tauberer, „What Is RDF“, 2006. [Na internetu]. Dostupno: https://www.xml.com/pub/a/2001/01/24/rdf.html [pristupano 2.8.2022.]

[2] W3Schools (bez dat.), *XML RDF* [Na internetu]. Dostupno: https://www.w3schools.com/xml/xml_rdf.asp [pristupano 2.8.2022.]

[3] G. Schreiber, Y. Raimond, „RDF 1.1 Primer“, 2014. [Na internetu]. Dostupno: https://www.w3.org/TR/rdf11-primer/ [pristupano 2.8. 2022.]

[4] M. Klein, „XML, RDF, and Relatives“, *IEEE Inteligent Systems,* sve. 16, izd. 2, str. 26-28, ožu./tra. 2001. [Na internetu]. Dostupno: https://ieeexplore.ieee.org/abstract/document/877487 [3.8.2022.]

[5] „Linked Data Basics: RDF Serializations and Triplestores“, 2019. [Na internetu]. Dostupno: https://heardlibrary.github.io/digital-scholarship/lod/serialization/ [pristupano 4.8.2022.]

[6] F. Manola, E. Miller, „RDF Primer“, 2004. [Na internetu]. Dostupno: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.458.872&rep=rep1&type=pdf [pristupano 4.8.2022.]

[7] „RDF Container Elements“ (bez dat.) [Na internetu]. Dostupno: https://w3schools.sinsixx.com/rdf/rdf_containers.asp.htm [pristupano 4.8.2022.]

[8] S. Decker i sur., „The Semantic Web: the roles of XML and RDF“, *IEEE Internet Computing*, sve. 4, izd. 5, str. 63-73, ruj./lis. 2000. [Na internetu]. Dostupno: https://ieeexplore.ieee.org/abstract/document/877487 [pristupano 5.8.2022.]

[9] M. Arenas, C. Guiterrez, J. Perez, „Foundations of RDF Databases“, *Reasoning Web International Summer School*, str. 158-204, 2009. [Na internetu]. Dostupno: https://link.springer.com/chapter/10.1007/978-3-642-03754-2_4 [pristupano 5.8.2022.]

[10] D. Brickley, R.V. Guha, „RDF Schema 1.1“, 2014. [Na internetu]. Dostupno: https://www.w3.org/TR/rdf-schema/ [pristupano 5.8.2022.]

[11] „What is an RDF Triplestore?“ (bez dat.) [Na internetu]. Dostupno: https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/ [pristupano 11.8.2022.]

[12] „RDF Database“, 2017. [Na internetu]. Dostupno: https://www.techopedia.com/definition/32092/rdf-database [pristupano 11.8.2022.]

[13] „NoSQL“, 2020., [Na internetu]. Dostupno: https://www.w3resource.com/mongodb/nosql.php [pristupano 12.8.2022.]

[14] A. Stojanović, „Osvrt na NoSQL baze podataka – Četiri osnovne tehnologije“, Polytechnic & Design, 2016. [Na internetu]. Dostupno: Hrcak, https://hrcak.srce.hr/ [pristupano 12.8.2022.]

[15] „What is a graph database?“, (bez dat.) [Na internetu]. Dostupno: https://neo4j.com/developer/graph-database/ [pristupano 12.8.2022.]

[16] K. Rabuzin, M. Šestak, „Grafovske baze podataka – pregled istraživanja i budućih trendova“, Sveučilište u Zagrebu, Fakultet organizacije i informatike, Varaždin, 2018. [Na internetu]. Dostupno: bib.irb.hr [pristupano 12.8.2022.]

[17] B. DuCharme, Learning SPARQL: Querying and Updating with SPARQL 1.1, O'Reilly Media, Inc., 2013.

[18] S. Lovrenčić, „SPARQL“, nastavni materijali na predmetu Baze znanja i semantički Web [Moodle], Sveučilište u Zagrebu, Fakultet organizacije i informatike, Varaždin, 2021.

[19] SIB Swiss Institute of Bioinformatics, SPARQL Playground [Web aplikacija]. Dostupno: http://sparql-playground.sib.swiss/ [pristupano 13.8.2022.]

[20] „AllegroGraph“ (bez dat.) [Na internetu]. Dostupno: https://allegrograph.com/products/allegrograph/ [pristupano 17.8.2022.]

[21] „AllegroGraph 7.3.0 Introduction“, 2022. [Na internetu]. Dostupno: https://franz.com/agraph/support/documentation/current/agraph-introduction.html#ai-overview [pristupano 17.8.2022.]

[22] „Gruff: A Triple-Store Browser, Query Manager, and Editor“, 2022. [Na internetu]. Dostupno: https://franz.com/agraph/support/documentation/current/gruff.html [pristupano 17.8.2022.]

[23] „Blazegraph“, *Database of Databases* (bez dat.) [Na internetu]. Dostupno: https://dbdb.io/db/blazegraph [pristupano 18.8.2022.]

[24] „How Knowledge Graphs Work“ (bez dat.) [Na internetu]. Dostupno: https://www.stardog.com/knowledge-graph/ [pristupano 23.8.2022.]

[25] *DB-Engines* (bez dat.) [Na internetu]. Dostupno: https://db-engines.com/en/ [pristupano 4.8.2022.]

[26] „Ontotext GraphDB“ (bez dat.) [Na internetu]. Dostupno: https://www.ontotext.com/products/graphdb/ [pristupano 24.8.2022.]

Popis slika

Popis tablica