SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

UMREŽAVANJE SADRŽAJA

3. DOMAĆA ZADAĆA

ZAGREB, 29. prosinca 2013.

SADRŽAJ

UVOD 1
Komentari vezani uz postojeću ontologiju i programski kod za izvršavanje upita nad njom 2
Komentar vlastitog rješenja 6

UVOD

Zadatak ove domaće zadaće je upoznavanje s ontologijama kao bazama znanja, njihovo postavljanje u repozitorije te izvršavanje upita nad ontologijama pomoću posebnog jezika SPARQL. Ontologija je skup formalno i eksplicitno definiranih koncepata u određenoj domeni te njihovi međusobni odnosi. Ontologija omogućuje da računala različite konfiguracije i programske podrške mogu učinkovito razmjenjivati podatke i informacije vezane uz određenu domenu te donositi određene zaključke na temelju tih podataka, bez posredovanja čovjeka. Najčešća primjena je u npr. web uslugama gdje se pomoću ontologije razmjenjuju strukturirani podatci o karakteristikama određene usluge i njezinim mogućnostima.

Zadaća je podijeljena u dva glavna dijela, prvi dio obuhvaća proučavanje postojeće ontologije i programskog koda koji izvršava upite nad tom ontologijom, a drugi dio obuhvaća izradu vlastite ontologije i modifikaciju inicijalnog programa s ciljem izvršavanja upita nad novom ontologijom. Prilikom generiranja upita nad bazom znanja, koristi se već spomenuti SPARQL.

Komentari vezani uz postojeću ontologiju i programski kod za izvršavanje upita nad njom

Postojeća ontologija je pohranjena u datoteci naziva University.owl. Nakon što dotičnu datoteku otvorimo u alatu Protege, uočavamo da ona posjeduje 40 klasa, 18 predikata i 16 instanci. Klase opisuju temeljne entitete i koncepte u nekoj domeni, predikati reprezentiraju odnose između tih entiteta ili njihova svojstva, dok instance predstavljaju pojedinačne realizirane klase (npr. klasa može biti *Sveučilište*, što predstavlja bilo koje sveučilište u svijetu, dok je jedna instanca npr. *Sveučilište u Zagrebu*). Klase definirane u ovoj datoteci su sljedeće: *City* (grad), *Organization* (organizacija), *Person* (osoba), *Settlement* (naselje), *Work* (posao). To su temeljne klase koje u sebi sadrže podklase (njihove podklase opet mogu sadržavati svoje podklase → hijerarhijska organizacija). Npr. klasa *Person* u sebi sadrži direktne podklase *Advisor*, *Associate, Employee*, *PhD*, *Student*, *Teacher*. Predikatima možemo modelirati odnose između pojedinih klasa. Npr. predikat *worksOn* pomaže u određivanju koja osoba radi na kojem poslu, povezujući klase *Person* i *Work*, gdje mu je klasa *Person* domena (*Domains*), a klasa *Work* doseg (*Ranges*). Instance su npr. *Department of Telecommunication*, što je jedna od realizacija klase *Department*, koja je podklasa klase *Organization*.

SPARQL je jezik za definiranje upita nad ontologijama te je preuzeo neka svojstva od SQL-a koji se koristi za dohvaćanje i modifikaciju podataka iz baze podataka. Tako i u SPARQL-u imamo SELECT dio naredbe kojim definiramo koje entitete, odnosno instance ili varijable želimo prikazati u rezultatu (definira se koliko će stupaca imati rezultat, ako se prikazuje tablično). Nakon SELECT dijela naredbe, slijedi WHERE dio gdje se definira koje će točno instance biti uključene u stvaranje ukupnog rezultata. To se određuje pomoću tzv. trostrukog uzorka (uzorak trojki): *subjekt predikat objekt*. Predikat je najčešće *object property* definiran u ontologiji, dok su subjekt i objekt najčešće nekakve varijable predznačene znakom „?“ čije će vrijednosti postati imena instanci koje povezuje predikat (odnosno njihove klase). Npr. ako imamo klasu `Patient` i klasu `Doctor`, te predikat `goesToDoctor` koji definira da pacijent ide doktoru, onda nakon izvršavanja upita `SELECT * WHERE {?X Prefiks:goesToDoctor ?Y}` dobivamo sve one instance (pacijente) za koje je definirano da posjećuju (idu kod) doktora, odnosno neke druge instance koja predstavlja doktora, te ujedno i te druge instance (doktore). Dakle, ispisuju se sve vrijednosti obiju varijabli, X i Y. Varijabla X je pritom subjekt, dok je varijabla Y objekt. U uzorku se ne moraju nužno navesti *objectProperties*, odnosno predikati, upit može izgledati i ovako: `SELECT ?X WHERE {?X a Prefiks:Doctor}`, gdje će se ispisivati sve one instance koje su nastale od klase `Doctor`, odnosno svi doktori.

Slijede komentari vezani za upite izvršene nad postojećom ontologijom (upiti se nalaze u klasi `Main.java` u projektu `UniversityQuerying` te se neće navoditi posebno ovdje, već će se komentirati onim slijedom kojim su navedeni u klasi `Main.java`). Također, komentari se odnose na repozitorij `uniowlim` koji ima podršku za semantičko zaključivanje, a u slučaju da rezultati upita budu značajno drugačiji tijekom korištenja repozitorija `uni`, razlika će biti eksplicitno (znak „!“) navedena.

1. Upit. Dohvati sve one instance za koje je definirano da rade (`University:worksFor`) za neku drugu instancu, te da je ta druga instanca primjerak klase `Company`. Ujedno prikaži i te druge instance (odnosno kompanije). Rezultat je `KrunoslavTrzec` kao varijabla ?X, i `EricssonNikolaTesla` kao varijabla ?Y. Ovaj par vrijednosti se ispisuje 4 puta. Razlog zbog

kojeg se rezultat ispisuje 4 puta leži u činjenici da se koristi semantičko zaključivanje. Naime, predikat `worksFor` je podpredikat predikata `memberOf`, a `memberOf` je pak podpredikat predikata `affiliateOf`. Budući da podpredikat i podklasa obično nasljeđuju svojstva svojih nadpredikata ili nadklasa te ih eventualno dodatno proširuju, semantičko zaključivanje ih može u postupku rasuđivanja uzeti kao jednake svojim nadpredikatima ili nadklasama i proširiti konačan rezultat upita. Tako će se za instancu `KrunoslavTrzec` nakon rasuđivanja ustvrditi da je ujedno i `memberOf` instance `EricssonNikolaTesla` i `affiliateOf` instance `EricssonNikolaTesla`. Jedan od ukupno 4 rezultata je onaj koji nije zaključen, postavljen je unaprijed (predikat je eksplicitno definiran za instance koje se pojavljuju u rezultatu). Ostali su semantički zaključeni. Pri zaključivanju pomaže i činjenica da je klasa `Employee` ekvivalentna izrazu `worksFor some Organization`.

!U slučaju izvršavanja ovog upita nad repozitorijem koji nema podršku za semantičko zaključivanje, dobiva se samo jednom par `KrunoslavTrzec` i `EricssonNikolaTesla` jer je u ontologiji eksplicitno definirano da instanca `KrunoslavTrzec` radi (`worksFor`) za `EricssonNikolaTesla`, ostali nadpredikati nisu definirani za tu konkretnu instancu i za njih je potrebno semantičko rasuđivanje.

2. Upit. Ovaj upit je skoro identičan prvom upitu, samo se u drugom uzorku trojki u WHERE dijelu naredbe definira da varijabla ?Y treba biti realizacija klase `Faculty`, a ne `Company`. Dakle dohvatit će se sve one osobe (instance) koje rade (predikat `worksFor`) za instancu koja je nastala od klase `Faculty`. Rezultat je: `AleksandarAntonic` kao varijabla ?X, `FacultyOfElectricalEngineeringAndComputing` kao varijabla ?Y potom `IvanaPodnarZarko` te ponovno `FacultyOfElectricalEngineeringAndComputing`, zatim `KresimirPripuzic` te ponovno `FacultyOfElectricalEngineeringAndComputing`, zatim `MajaMatijasevic` i `FacultyOfElectricalEngineeringAndComputing`, potom `NedjelkoPeric` i `FacultyOfElectricalEngineering AndComputing`. Svaki do navedenih parova (?X ?Y) se ispisuje ukupno 4 puta.

Parovi se ispisuju po 4 puta iz skoro identičnog razloga kao i u prethodnom upitu, predikat `worksFor` je samo podpredikat od predikata `memberOf` i `affiliateOf` te će se i oni uključiti u proces semantičkog rasuđivanja, odnosno u konačni rezultat. Za svaku osobu koja se pojavljuje u rezultatu upita se automatski zaključuje ne samo da radi (`worksFor`), nego da je i `memberOf` i `affiliateOf` fakulteta. Prvi ispis je obično onaj koji nije semantički zaključen.

!Tijekom korištenja repozitorija koji nema podršku za semantičko zaključivanje, svaki par se ispisuje samo jednom, a ne četiri puta.

3. Upit. Dohvati sve one instance koje su realizacija klase `PhD` (dakle sve one osobe koje imaju doktorat). Ovaj upit nema predikat u sebi (umjesto njega se koristi ključna riječ `a`). Rezultat je: `IvanaPodnarZarko, KresimirPripuzic, KrunoslavTrzec, MajaMatijasevic i Nedjelko Peric`.

!Tijekom izvršavanja ovog upita u repozitoriju koji **nema** podršku za semantičko zaključivanje, rezultat će biti prazan.

Razlog zbog kojeg je rezultat prazan u slučaju repozitorija bez zaključivanja je sljedeći: u ontologiji nije eksplicitno definirano da klasa `PhD` ima članove (`Members`), ali je za nju definirano da je ekvivalent (`Equivalent to`) sljedećem izrazu: `doctoralDegreeFrom` *some* `University`. Predikat `doctoralDegreeFrom` je definiran za one individue (instance) koje se pojavljuju u rezultatu ovog upita kada se koristi repozitorij s podrškom za zaključivanje. Dakle, da bi se generirao rezultat, potrebno je zaključivanje jer nije eksplicitno definirano da klasa `PhD` ima članove, već se to mora zaključivati putem semantičkog rasuđivanja (posredstvom predikata `doctoralDegreeFrom`).

4. Upit. Dohvati sve one instance za koje je definirano da im pripada (predikat `affiliateOf`) specifična instanca zvana `KresimirPripuzic`. Ovdje se kao subjekt koristi specifična instanca, a ne varijabla. Rezultat je: `FacultyOfElectricalEngineeringAndComputing`, `DistributedInformationSystemGroup`, `DepartmentOfTelecommunications`.

!Tijekom izvršavanja ovog upita u repozitoriju koji **nema** podršku za semantičko zaključivanje, rezultat će biti prazan.

Razlog praznog rezultata za repozitorij bez semantičkog zaključivanja je slične naravi kao i u prethodnom upitu. Naime, za instancu `KresimirPripuzic` nije eksplicitno definiran predikat `affiliateOf`, odnosno da je on eksplicitno `affiliateOf` neke druge instance, već je to učinjeno putem podpredikata, odnosno predikata `memberOf` i `worksFor` koji su podpredikati predikata `affiliateOf`. Stoga će se rezultat generirati samo u slučaju repozitorija sa semantičkim zaključivanjem jer je potrebno rasuđivanje da bi se podpredikati povezali sa svojim glavnim predikatom. Za instancu `KresimirPripuzic` je definirano da je član instanci `DepartmentOfTelecommunications` i `DistributedInformationSystemGroup`, te da radi za `FacultyOfElectricalEngineeringAndComputing`, stoga će se te instance pojaviti u rezultatu.

5. Upit. Dohvati sve one instance za koje je definirano da su na čelu (predikat `headOf`) neke druge instance, ujedno prikaži i te druge instance. Dakle prikazat će se sve osobe koje su na čelu nekog odjela ili organizacije, te dotične organizacije ili odjeli. Rezultat je: `IvanaPodnarZarko` i `DistributedInformationSystemGroup`, `MajaMatijasevic` i `DepartmentOfTelecommunication` te `NedjelkoPeric` i `FacultyOfElectricalEngineeringAndComputing`. Ovi parovi se ispisuju po dva puta.

Svaki rezultat se ispisuje po dva puta iz razloga što su dvije različite klase definirane kao ekvivalent predikatu `headOf`, a te dvije klase su `Head` i `Dean`. Kod tih dvaju klasa predikat `headOf` ima različite raspone (u slučaju klase `Head`, objekt odnosno raspon predikata `headOf` je `Organization`, a u slučaju klase `Dean` raspon je `Faculty`). Dakle na temelju predikata `headOf` rasuđuju se dva različita tipa klasa.

!Kod uni repozitorija svaki par se ispisuje jedanput.

6. Upit. Dohvati sve one instance za koje je definirano da su suradnici (predikat `associateOf`) neke druge instance, dakle sve one osobe koje surađuju s nekom organizacijom. Rezultat je: `KrunoslavTrzec` (dva puta se ispisuje).

Rezultat se ispisuje dva puta jer se dodatno zaključuje da je klasa `Associate` ekvivalent izrazu `associateOf some Organization`. To je dodatno zaključivanje koje se javlja uz rezultat koji se ne mora semantički zaključiti (predikat associateOf je eksplicitno definiran za instancu KrunoslavTrzec).

Također, predikat associateOf je podpredikat predikata affiliateOf, pa se semantičkim zaključivanjem definira da je `KrunoslavTrzec` ujedno i `affiliateOf facultyOfElectricalEngineeringAndComputing`.

!Kod `uni` repozitorija, rezultat se ispisuje samo jednom.

7. Upit. Dohvati sve one instance za koje je definirano da su predavači (predikat `teacherOf`) na nekoj drugoj instanci, dakle sve one osobe koje predaju na određenim kolegijima. Rezultat je: `IvanaPodnarZarko`, `KresimirPripuzic` i `KrunoslavTrzec` (neke se vrijednosti ispisuju više puta, npr. `KresimirPripuzic` 4 puta, ostali po 2 puta).

Za instance koje se pojavljuju po 2 puta u rezultatu došlo je do semantičkog zaključivanja i to na temelju toga što je za klasu `Teacher` definirano da je ekvivalentna izrazu `teacherOf some Course` pa preko nje dolazi do dodatnog zaključivanja (predikat `teacherOf` zaključuje da je instanca pripadnik klase `Teacher`, a ta klasa je ekvivalentna izrazu `teacherOf some Course`). Uz to, predikat `teacherOf` je podpredikat predikata `worksOn` pa semantičko zaključivanje dodatno zaključuje da te instance ujedno i rade (`worksOn`) na instanci koju predaju.

Instanca `KresimirPripuzic` predaje dva predmeta, pa se pojavljuje još dva puta u ispisu.

!`uni` repozitorij ispisuje svaku osobu samo jednom, osim KresimirPripuzic (ispisuje 2 puta) jer on predaje dva predmeta (eksplicitno definirano u ontologiji).

8. Upit. Dohvati sve one instance za koje je definirano u ontologiji da rade (predikat `worksOn`) na specifičnoj instanci naziva `ContentNetworking`, dakle sve one osobe koje rade na kolegiju „ContentNetworking“. Rezultat je: `AleksandarAntonic`, `KresimirPripuzic`, `KrunoslavTrzec` i `IvanaPodnarZarko`.

!Tijekom izvršavanja ovog upita u repozitoriju koji **nema** podršku za semantičko zaključivanje, rezultat će biti prazan.

Rezultat je prazan u slučaju repozitorija bez podrške za semantičko zaključivanje jer nije eksplicitno navedeno u ontologiji da neke instance rade (`worksOn`) na instanci `ContentNetworking`. Potrebno je semantičko zaključivanje da bi se generirali rezultati, i to na temelju sljedećega: instanca `AleksandarAntonic` je tipa `TeachingAssistant` i za njega je definiran izraz `teachingAssistantOf ContentNetworking`, a za instancu `TeachingAssistant` je pak definirano da je ekvivalent `teachingAssisatntOf some Course`, a `ContentNetworking` je tipa `Course`; za instancu `KresimirPripuzic` je definirano da je `teacherOf` instance `ContentNetworking`, a predikat `teacherOf` je podpredikat predikata `worksOn`; za instancu `IvanaPodnarZarko` vrijedi isto kao i za instancu `KresimirPripuzic`; za instancu `KrunoslavTrzec` vrijedi isto.

9. Upit. Dohvati sve one instance za koje je definirano da rade (predikat `worksOn`) na onim instancama (ili jednoj instanci) koje su realizacija klase `Research`, dakle sve one osobe koje rade na projektima koji pripadaju odjelu ili kategoriji „Research“. Rezultat je: `AleksandarAntonic` i `ContentMatching`, `AleksandarAntonic` i `PublishSubscribeSystem`, `AleksandarAntonic` i `SemanticWeb`, `IvanaPodnarZarko` i `PublishSubscribeSystem`, `KresimirPripuzic` i `PublishSubscribeSystem`, `KrunoslavTrzec` i `SemanticWeb`. Ovi parovi se ispisuju po 4 puta.

Višestruki ispis rezultat je dakako semantičkog zaključivanja, a to se zaključivanje provodi na temelju činjenice da je klasa `ResearchStaff` ekvivalentna izrazu `worksOn some Research`. Nakon zaključivanja konstatira se da su osobe koje se pojavljuju u rezultatu pripadnici klase `ResearchStaff`, pa se provodi dodatno zaključivanje i zbog toga višestruki ispis.

!U repozitoriju `uni` svaki par se ispisuje samo jednom.

10. Upit. Dohvati sve one instance za koje je definirano da imaju savjetnika (predikat `hasAdvisor`), ujedno prikaži i te druge instance koji su savjetnici. Rezultat je: `AleksandarAntonic` i `IvanaPodnarZarko` (dva puta se ispisuje ovaj par).

Predikat `hasAdvisor` ima svoj inverzni predikat, `isAdvisorOf`. Na temelju toga će semantičko zaključivanje dva puta ispisati par koji čini rezultat jer ako `AleksandarAntonic` ima mentora (`hasAdvisor`) onda taj mentor ujedno ima i tog studenta kao savjetnika (inverzni predikat `isAdvisorOf`) i to se dodatno zaključuje za instancu `IvanaPodnarZarko` (a student kojem je ona mentor je `AleksandarAntonic`) pa se rezultat zapravo dva puta ispisuje. Za instancu `AleksandarAntonic` je eksplicitno definirano da ima kao mentora instancu `IvanaPodnarZarko`.

!Kod repozitorija uni rezultat (par) se ispisuje samo jednom.

Repozitorij `uni` ne podržava semantičko zaključivanje i proces generiranja rezultata je kod njega vrlo jednostavan. Za svaki predikat u svakom upitu se provjerava koja instanca u ontologiji ima eksplicitno definiran taj predikat i neki objekt vezan uz taj predikat koji zadovoljava uvjet u uzorku trojki te se to uključuje u rezultat, odnosno odgovor na upit. Dakle pretražuje se baza znanja i uspoređuje se koje trojke odgovaraju predanim uzorcima.

Komentar vlastitog rješenja

Ontologija (`XX.owl`) kreirana u ovom dijelu zadaće predstavlja domenu koja obuhvaća desetak timskih sportova u kojima postoje golmani, napadači i obrambeni igrači. Ti sportovi se mogu igrati u nekoliko različitih turnira koji se mogu održavati u raznim gradovima. U tim sportovima (i natjecanjima), osim igrača, mogu sudjelovati i sudci (glavni, te dva pomoćna sudca). U skladu s opisanim entitetima, kreirane su sljedeće klase: *City*, *Player*, *Referee*, *Sport*, *Tournament*. Klasa *City* nema podklasa, dok ostale klase imaju. Tako klasa *Player* ima podklase *Attacker*, *Defender* i *Goalkeeper*. Klasa *Referee* ima podklase *MainReferee*, *FrontSecondaryReferee* i *BackSecondaryReferee*. Klasa *Sport* ima 11 podklasa od kojih svaka predstavlja jedan sport. Konačno, klasa *Tournament* ima podklase *ClubLeague*, *EurpeanTournament* i *WorldTournament*. Odnosi između tih klasa definirani su pomoću tzv. predikata, odnosno *Object Properties*. Imamo sljedeće predikate: *livesIn* (igrač živi u gradu), *isMainRefereeIn* (sudac je glavni sudac u sportu), *isHeldInTournament* (sport se igra na turniru), *isHeldinCity* (turnir se održava u gradu), *isFrontSecondaryRefereeIn* (sudac je prednji pomoćni sudac u sportu), is *BackSecondaryRefereeIn* (sudac je zadnji pomoćni u sportu), *hasExperienceInTournament* (sudac ima iskustva u turniru), *goalkeeperIn* (golman brani u sportu), *defendsIn* (igrač igra obranu u sportu), *attacksIn* (igrač napada u sportu). Zadnja tri predikata su podpredikati predikata *playsIn*. Uz navedene predikate i klase,

postoji i nekoliko instanci: *Classic_Soccer* (instanca klase *Soccer*), *FIFA_World_Cup* (instanca klase *WorldTournament*), *Josip_Simunic* (instanca klase *Defender*), *Mario_Mandzukic* (instanca klase *Attacker*), *Marko_Markic* (nije eksplicitno definiran kao član određene klase), *Pierluigi_Collina* (instanca klase *MainReferee*) i *Zagreb* (instanca klase *City*).

Slijede komentari upita izvršenih nad vlastitom ontologijom. Semantičko rasuđivanje u svim upitima se provodi pretraživanjem baze znanja kako bi se našle one trojke (subjekt – predikat – objekt) koje odgovaraju zadanim uzorcima u upitu, a to se najčešće obavlja uspoređivanjem svih trojki sa zadanim uzorcima te vraćanjem onih koje zadovoljavaju upit. Rezultat je najčešće pronalazak svih onih instanci koje su definirane kao domena određenog predikata te svih onih instanci koje su definirane kao doseg (ili moguće vrijednosti) istog predikata. Povezivanje uzoraka trojki u WHERE dijelu naredbe se provodi slijedno i to s lijeva na desno (pojedini trostruki uzorci su odvojeni znakom „.“) te se rezultat jednog trostrukog uzorka može odmah koristiti u sljedećem, i to u obliku varijabli. Primjer takvog upita je 2. upit gdje se varijabli Y postavi vrijednost nakon obrade prvog trostrukog uzorka `(?X XX:livesIn ?Y)`, a potom se ta varijabla koristi u drugom trostrukom uzorku `(?Z XX:isHeldinCity ?Y)`. Važno je napomenuto da se komentari odnose na upite izvršene nad repozitorijem koji **ima** podršku za semantičko zaključivanje, a u slučaju se rezultati razlikuju tijekom izvršavanja upita koji nema podršku za zaključivanje, razlike će biti eksplicitno navedene.

1. Upit. Dohvati sve instance za koje vrijedi: instanca X živi (predikat *livesIn*)u instanci Y. Znak „*“ u SELECT dijelu naredbe ukazuje na to da se trebaju ispisati sve instance (varijable) koje sudjeluju u upitu, odnosno koje su ili subjekt ili objekt, a koje zadovoljavaju upit. U ovom slučaju ispisat će se sadržaj varijabli X i Y (rezultati su prikazani u nastavku, pod `Query Result`:). Rezultati će biti isti i u slučaju repozitorija s podrškom za zaključivanje, i u slučaju repozitorija bez podrške za zaključivanje.

```
PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select * where

{?X XX:livesIn ?Y .}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Josip_Simunic

Y: http://www.fer.hr/umrsad/XX.owl#Zagreb


X: http://www.fer.hr/umrsad/XX.owl#Mario_Mandzukic

Y: http://www.fer.hr/umrsad/XX.owl#Zagreb
```

2. Upit. Dohvati samo one instance (gradove) za koje je definirano da netko u njima živi (predikat *livesIn*) i da se u njima održava (*isHeldinCity*) neki turnir. Konkretno, riječ je o varijabli Y, koja je objekt u ovom upitu. Dva puta se ispisuje *Zagreb* jer je definirano u ontologiji da dva igrača žive u njemu. Rezultat je isti i kod repozitorija bez podrške za zaključivanje.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>
```

```
select ?Y where

{?X XX:livesIn ?Y . ?Z XX:isHeldinCity ?Y}


Query Result:

Y: http://www.fer.hr/umrsad/XX.owl#Zagreb

Y: http://www.fer.hr/umrsad/XX.owl#Zagreb
```

3. Upit. Dohvati one instance za koje vrijedi da žive (predikat *livesIn*) u nekoj drugoj instanci (gradu) i da igraju napad (predikat *attacksIn*) u nekoj trećoj instanci. Rezultat je *Mario_Mandzukic* jer je za njega definirano da živi u Zagrebu i da igra napad u klasičnom nogometu. Rezultat je isti u slučaju repozitorija bez zaključivanja.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X XX:livesIn ?Y . ?X XX:attacksIn ?Z}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Mario_Mandzukic
```

4. Upit. Dohvati one instance za koje vrijedi da igraju obranu (predikat *defendsIn*) u specifičnoj instanci zvanoj *Classic_Soccer*. Rezultat je *Josip_Simunic* jer je za njega definirano u ontologiji da igra u sportu (instanci klase *Soccer*) *Classic_Soccer*. Rezultat je isti u kod repozitorija bez zaključivanja.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select * where

{?X XX:defendsIn XX:Classic_Soccer}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Josip_Simunic
```

5. Upit. Dohvati one instance za koje je definirano da su glavni sudci (predikat *isMainRefereeIn*) u nekom sportu. Rezultat je *Pierluigi_Collina* jer je definirano u ontologiji da je on glavni sudac u klasičnom nogometu. Rezultat je isti u slučaju repozitorija bez zaključivanja.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X XX:isMainRefereeIn ?Y .}
```

```
Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Pierluigi_Collina
```

6. Upit. Dohvati sve one instance za koje vrijedi da imaju iskustva (*hasExperienceInTournament*) u nekoj drugoj instanci (turniru), da su glavni sudci (*isMainRefereeIn*) u nekom sportu (treća instanca) te da se taj sport igra (predikat *isHeldInTournament*) na nekom turniru (četvrta instanca). U rezultat će biti uključene sve varijable zbog znaka „*“ u SELECT dijelu naredbe, odnosno rezultat bi trebao biti *Pierluigi_Collina, FIFA_World_Cup, Classic_Soccer i ponovno FIFA_World_Cup* (jer se isti sadržaj nalazi u drugoj varijabli). Rezultat je isti u repozitoriju bez podrške za zaključivanje.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select * where

{?X XX:hasExperienceInTournament ?Y . ?X XX:isMainRefereeIn ?Z . ?Z XX:isHeldInTournament ?C
.}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Pierluigi_Collina

Y: http://www.fer.hr/umrsad/XX.owl#FIFA_world_cup

Z: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer

C: http://www.fer.hr/umrsad/XX.owl#FIFA_world_cup
```

7. Upit. Dohvati sve one instance za koje vrijedi da su realizacija klase *Goalkeepr*. Rezultat je *Marko Markic*.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X a XX:Goalkeepr}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Marko_Markic
```

!Tijekom izvođenja upita nad repozitorijem koji nema podršku za zaključivanje, ne dobivamo nikakve rezultate. Razlog tomu je činjenica da u ontologiji nije eksplicitno definirano da klasa *Goalkeeper* ima članove, već je za nju definirano da je ekvivalent sljedećem izrazu: *goalkeeperIn some Sport*, a za instancu *Marko_Markic* je definiran sljedeći izraz pod *Object Property Assertion*: *goalkeeperIn Classic_Soccer*. Dakle na temelju tih svojstava repozitorij zaključuje da je *Marko_Markic* zapravo golman (odnosno da je realizacija klase *Goalkeeper*).

8. Upit. Dohvati sve one instance za koje je definirano da su prednji pomoćni sudac (predikat *isFrontSecondaryRefereeIn*) u nekom sportu (drugoj instanci). Rezultat je prazan za obje vrste repozitorija jer takva instanca nije uopće definirana u ontologiji.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X XX:isFrontSecondaryRefereeIn ?Y .}


Query Result:
```

9. Upit. Dohvati one instance za koje vrijedi da se održavaju (*isHeldInTournament*) na nekom turniru (druga instanca) te da se taj turnir održava (*isHeldinCity*) u nekom gradu (treća instanca). Rezultat je *Classic_Soccer* jer se on održava na turniru *FIFA_World_Cup*, a taj turnir se pak održava u *Zagrebu*. Rezultat je isti za obje vrste repozitorija.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X XX:isHeldInTournament ?Y . ?Y XX:isHeldinCity ?Z .}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer
```

10. Upit. Odaberi one instance za koje vrijedi da igraju obranu (predikat *defendsIn*) u nekoj drugoj instanci (sportu) i da se taj sport održava (predikat *isHeldInTournament*) na nekom turniru (treća instanca). Rezultat je *Josip_Simunic* i *Classic_Soccer*. Isti se rezultat dobiva i u slučaju repozitorija bez podrške za zaključivanje.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X ?Y where

{?X XX:defendsIn ?Y . ?Y XX:isHeldInTournament ?Z}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Josip_Simunic

Y: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer
```

11. Upit. Odaberi one instance za koje je definirano da igraju (predikat *playsIn*) u nekoj drugoj instanci (sportu) te ujedno prikaži i te druge instance (sportove). Rezultat je prikazan u nastavku:

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X ?Y where

{?X XX:playsIn ?Y}


Query Result:

X: http://www.fer.hr/umrsad/XX.owl#Josip_Simunic

Y: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer


X: http://www.fer.hr/umrsad/XX.owl#Mario_Mandzukic

Y: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer


X: http://www.fer.hr/umrsad/XX.owl#Marko_Markic

Y: http://www.fer.hr/umrsad/XX.owl#Classic_Soccer
```

!Tijekom izvođenja upita nad repozitorijem koji nema podršku za zaključivanje dobivamo prazan rezultat. Razlog tomu je činjenica da u ontologiji nije eksplicitno definirano za neku instancu da *playsIn* (igra) u nekoj drugoj instanci, nego su podpredikati predikata *playsIn* definirani za instance koje se pojavljuju u rezultatu kad se koristi semantičko zaključivanje, a da bi se dobili ti rezultati potrebno je semantičko zaključivanje. Dakle semantičko zaključivanje donosi zaključak da iako ni za jednu instancu nije deinirnao da *playsIn*, ipak postoje instance koje imaju podpredikate tog predikata (npr. *attacksIn*) pa će se pojaviti u rezultatu. Smatra se da je podpredikat isti kao i predikat (obuhvaća sva njegova svojstva), samo što ga dodatno proširuje. Slično vrijedi i za klase.

12. Upit. Dohvati one instance za koje vrijedi da su pomoćni stražnji sudci (predikat *isBackSecondaryRefereeIn*) u nekom sportu. Takve instance nisu definirane u ontologiji, pa je rezultat prazan (za obje vrste repozitorija).

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?X where

{?X XX:isBackSecondaryRefereeIn ?Y .}


Query Result:
```

13. Upit. Dohvati one instance za koje vrijedi da se specifična instanca zvana *FIFA_world_cup* održava (predikat *isHeldinCity*) u njima. Dakle, dohvaćaju se svi oni gradovi za koje vrijedi da se u njima održava *FIFA_world_cup*. Ispravan rezultat bi trebao biti *Zagreb*, i taj se rezultat dobiva prilikom postavljanja upita kod obje vrste repozitorija.

```
Query:

PREFIX XX: <http://www.fer.hr/umrsad/XX.owl#>

select ?Y where

{XX:FIFA_world_cup XX:isHeldinCity ?Y .}


Query Result:

Y: http://www.fer.hr/umrsad/XX.owl#Zagreb
```

Uspoređujući upite izvršene nad repozitorijima s i bez podrške za semantičko zaključivanje, vidimo da postoji razlika u 7. i 11. upitu. Te razlike su objašnjene ispod samih upita. U ostalim upitima rezultat je isti jer su predikati koji se koriste u njima eksplicitno definirani u ontologiji za neke instance, a i same instance koje se pojavljuju u rezultatima su pripadnici određenih klasa i za njih su u ontologiji eksplicitno definirani predikati koji se pojavlju u upitu.