Ontologia sobre Filmes

Projeto de MAC0444

CAIQUE BARROS MARINHO	8940848
MATEUS AGOSTINHO DOS ANJOS	9298191
PEDRO SOLA PIMENTEL	9298079
VINICIUS PERCHE DE TOLEDO AGOSTINI	4367487

1 DESCRIÇÃO DO PROJETO

Projeto da matéria Sistemas Baseados em Conhecimento (MAC0444) em que modelamos, utilizando a ferramenta Protègè, uma pequena ontologia a fim de praticarmos os conceitos vistos em aula. A partir desta modelagem podemos obter informações como diretor, atores, ano de lançamento e duracao, sobre os filmes que foram inseridos na ontologia.

2 CONSTRUÇÃO DA ONTOLOGIA

A explicação da modelagem da ontologia seguirá o seguinte padrão:

Classe:
Nome da Classe que foi criada no Protègè.
Object Properties:
Lista de Object Properties que estão relacionadas com a Classe citada - Descrição de como o campo desta propriedade foi preenchido.
Data Properties:
Lista de Data Properties que estão relacionadas com a Classe citada - Descrição de como o campo desta propriedade foi preenchido.
Descrição:
Aqui vai um exemplo sobre como está preenchido um indivíduo que pertence a esta classe.

Portanto, a ontologia foi modelada da seguinte forma:

Classe:
Filme
Object Properties:
dirigido_por - Preenchido via parser.
Data Properties:
ano_lancamento - Preenchido via parser.
duracao - Preenchido via parser.
titulo - Preenchido via parser.
Descrição:
O Filme F é dirigido_por Diretor D.
O Filme F tem ano_lancamento igual a 2008.
O Filme F tem duracao igual a 128 (minutos).
O Filme F tem titulo igual a "O melhor filme do mundo".

Esta Classe contém os indivíduos que estão marcados
no predicado 'filme' no arquivo filmes.pl

Classe:
Diretor
Object Properties:
dirige - Inferido a partir da inversa 'dirigido_por'
Data Properties:
primeiro_nome - Preenchido via parser.
nome_familia - Preenchido via parser.
Descrição:
O Diretor D dirige o Filme F. (note que é inversa de 'dirigido_por')
O Diretor D tem primeiro_nome igual a "João".
O Diretor D tem nome_familia igual a "Pereira".

Esta Classe contém os indivíduos que estão marcados
no predicado 'diretor' no arquivo filmes.pl

Classe:
Estrela
Object Properties:
Data Properties:
Descrição:
Esta classe tem como subclasses "Ator" e "Atriz", portanto serve para se fazer consultas em que o sexo da Pessoa não importa.

Classe:
Ator
Object Properties:
atua - Preenchido via parser.
Data Properties:
primeiro_nome - Preenchido via parser.
nome_familia - Preenchido via parser.
Descrição:
O Ator Ar atua a Atuacao 1337.
O Ator Ar tem primeiro_nome igual a "Pedro".
O Ator Ar tem nome_familia igual a "Oliveira".

Esta Classe contém os indivíduos que estão marcados
no predicado 'ator' no arquivo filmes.pl

Classe:
Atriz
Object Properties:
atua - Preenchido via parser.
Data Properties:
primeiro_nome - Preenchido via parser.
nome_familia - Preenchido via parser.
Descrição:
A Atriz Az atua a Atuacao 1693.
A Atriz Az tem primeiro_nome igual a "Patricia".
A Atriz Az tem nome_familia igual a "Silva".

Esta Classe contém os indivíduos que estão marcados
no predicado 'atriz' no arquivo filmes.pl

Classe:
Personagem
Object Properties:
Data Properties:
nome_do_personagem - Preenchido via parser.
Descrição:
O Personagem P tem nome_do_personagem igual a "Ratao_Esperto".

O Personagem não possui a divisão em primeiro_nome e nome_familia, pois alguns personagens não tem informações desse tipo, por exemplo, temos o personagem que é letter_writer ou poderíamos ter algo como homem_aranha. Além disso nós criamos um personagem 'nao_informado' para atribuir às atuações que não tinham personagem informado.

Esta Classe contém os indivíduos que estão marcados
no predicado 'ator' ou 'atriz' no arquivo filmes.pl

Classe:

Atuacao

Object Properties:

personagem_da_atuacao- Preenchido via parser.
filme_da_atuacao - Preenchido via parser.

Data Properties:

Descrição:

A Atuacao 1337 tem personagem_da_atuacao igual a P.
A Atuacao 1337 tem filme_da_atuacao igual a F.

Criamos a Classe Atuacao para que fosse possível atribuirmos um personagem e um filme, que estão ligados, a uma determinada estrela, ou seja, para que conseguíssemos traduzir um predicado de aridade 3 para a modelagem da Ontologia. (Utilizamos a Reificação)

Esta Classe foi preenchida com números para identificar cada uma das atuações.

Outras Considerações:

Note que as Classes Ator e Atriz são disjuntas, portanto um Ator não pode ser também Atriz. Além disso uma Pessoa não pode ser um Filme já que as Classes Person e Project são disjuntas.

Já as classes Diretor e Estrela não são disjuntas, portanto uma Estrela que também dirigiu um filme aparecerá como indivíduo na classe Diretor, porém não houve duplicação de indivíduos, apenas preenchemos tal indivíduo como parte das duas Classes.

3 RELAÇÃO ENTRE ONTOLOGIA CONSTRUÍDA E ONTOLOGIAS REFERENCIADAS

Neste projeto utilizamos a ontologia rdf em algumas consultas para escolher apenas indivíduos de uma determinada Classe, para isso usamos rdf:type.

Já a ontologia FOAF nos permitiu organizar melhor o projeto, já que fizemos o *direct import* dela (NÃO editamos o arquivo da ontologia FOAF) e começamos a construir nossa ontologia nos apoiando em algumas classes e propriedades já criadas como as classes foaf:Project, foaf:Person, e as propriedades foaf:made e foaf:maker.

Devido a isso, o reasoner do Protègè conseguiu inferir algumas informações importantes como foaf:made e foaf:maker a partir de dirige e atua, por exemplo, o que facilita algumas consultas que pretendem listar quais Pessoas participaram de um filme, sem se importar com a área específica de atuação. Além das inferências a ontologia foaf nos forneceu as Classes Project e Person, facilitando modelar a disjunção de Filme e Atores, por exemplo.

4 CONSULTAS FEITAS NO SPARQL

Utilizamos os seguintes prefixos para as consultas (incluindo os prefixos padrões que já aparecem na aba SPARQL Query):

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns>
PREFIX owl:  <http://www.w3.org/2002/07/owl>
PREFIX rdfs:  <http://www.w3.org/2000/01/rdf-schema>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema>
PREFIX :
<http://www.semanticweb.org/root/ontologies/2019/10/projeto_mac0444>
```

Cada consulta será exibida de acordo com o seguinte padrão:
(Note que: id. = identificador)

IDENTIFICADOR DA CONSULTA
Para consultas em que foram utilizadas mais de uma combinação de parâmetros, aqui virá uma breve explicação sobre eles.

Parâmetros:

<id. no enunciado> = <id. na consulta> = <valor do parâmetro>

Consulta em SPARQL:

Aqui virá o código da consulta em SPARQL

Resultado da consulta:

Aqui virá a imagem relativa a saída da consulta em questão

CONSULTA 1.

Parâmetros:

diretor D = ?diretor = :eric_schaeffer

Consulta em SPARQL:

```
SELECT ?TituloDoFilme
WHERE {
    VALUES ?diretor {:eric_schaeffer}.
    ?Filme :dirigido_por ?diretor.
    ?Filme :titulo ?TituloDoFilme.
}
ORDER BY ?TituloDoFilme
```

Resultado da consulta:

	TituloDoFilme
1.	"Fall"
2.	"If Lucy Fell"

CONSULTA 2.

Parâmetros:

Filme $\mathrm{F}_t = ?titulo = \ 'Her'$

Consulta em SPARQL:

```
SELECT ?PrimeiroNome ?UltimoNome ?Personagem
WHERE {
    VALUES ?titulo {"Her"}
    ?filme :titulo ?titulo.
    ?atuacao :filme_da_atuacao ?filme.
    ?estrela :atua ?atuacao.
    ?atuacao :personagem_da_atuacao ?Personagem.
    ?estrela :primeiro_nome ?PrimeiroNome.
    ?estrela :nome_familia ?UltimoNome.
}
ORDER BY ?PrimeiroNome ?UltimoNome
```

Resultado da consulta:

	PrimeiroNome	UltimoNome	Personagem
1.	"Amy"	"Addams"	amy
2.	"Chris"	"Pratt"	paul
3.	"Gabe"	"Gomez"	letter_writer3
4.	"Joaquin"	"Phoenix"	theodore
5.	"Lisa"	"Pitts"	letter_writer2
6.	"Lynn"	"Adrianna"	letter_writer1
7.	"Matt"	"Letscher"	charles
8.	"Rooney"	"Mara"	catherine
9.	"Scarlett"	"Johansson"	samantha

CONSULTA 3.

Parâmetros:

Ator X = ?atorUm = :john_turturro
Ator Y = ?atorDois = :john_goodman

Consulta em SPARQL:

```
SELECT ?Filmes ?Diretor ?Ano
WHERE {
    VALUES (?atorUm ?atorDois) {(:john_turturro :john_goodman)}.
    ?atorUm :atua ?atuaUm.
    ?atorDois :atua ?atuaDois.
    ?atuaUm :filme_da_atuacao ?Filmes.
    ?atuaDois :filme_da_atuacao ?Filmes.
    ?Filmes :dirigido_por ?Diretor.
    ?Filmes :ano_lancamento ?Ano.
}
ORDER BY ?Ano
```

Resultado da consulta:

	Filmes	Diretor	Ano
1.	barton_fink	joel_coen	1991
2.	the_big_lebowski	joel_coen	1998
3.	o_brother_where_art_thou	joel_coen	2000

CONSULTA 4.
Note que nesta consulta:

scarlett_johansson aparece em todos os filmes do diretor woody_allen, enquanto emily_mortimer aparece apenas em 1 deles.

Parâmetros:

Filme F = ?f = :match_point
Ator X = ?X = :emily_mortimer
Ator Y = ?Y = :scarlett_johansson

Consulta em SPARQL:

```
SELECT DISTINCT ?Filme ?Duracao
WHERE {
    VALUES (?f ?X ?Y) {(:match_point :emily_mortimer
:scarlett_johansson)}.
    ?f :dirigido_por ?diretor.
    ?Filme :dirigido_por ?diretor.
    ?X :atua ?atuacaoX.
    ?Y :atua ?atuacaoY.
    {?atuacaoX :filme_da_atuacao ?Filme.}
    UNION
    {?atuacaoY :filme_da_atuacao ?Filme.}
    ?Filme :duracao ?Duracao.
}
ORDER BY ?Duracao
```

Resultado da consulta:

	Filme	Duracao
1.	scoop	96
2.	match_point	124

CONSULTA 4.
Note que nessa consulta:

michael_patrick_bell não aparece em nenhum dos filmes de woody_allen, já jody_halse aparece em apenas 1 deles.

Parâmetros:

Filme F = ?f = :match_point
Ator X = ?X = :michael_patrick_bell
Ator Y = ?Y = :jody_halse

Consulta em SPARQL:

```
SELECT DISTINCT ?Filme ?Duracao
WHERE {
    VALUES (?f ?X ?Y) {(:match_point :michael_patrick_bell
:jody_halse)}.
    ?f :dirigido_por ?diretor.
    ?Filme :dirigido_por ?diretor.
    ?X :atua ?atuacaoX.
    ?Y :atua ?atuacaoY.
    {?atuacaoX :filme_da_atuacao ?Filme.}
    UNION
    {?atuacaoY :filme_da_atuacao ?Filme.}
    ?Filme :duracao ?Duracao.
}
ORDER BY ?Duracao
```

Resultado da consulta:

	Filme	Duracao
1.	scoop	96

CONSULTA 5.
Note que esta consulta lista todas as combinações.
(Não foi possível colocar uma boa imagem que mostrasse todas)

Parâmetros:

Consulta em SPARQL:

```
SELECT ?Pessoa ?FilmeAt ?FilmeDir
WHERE {
    ?Pessoa rdf:type :Diretor.
    ?FilmeDir :dirigido_por ?Pessoa.
    ?Pessoa :atua ?atuacao.
    ?atuacao :filme_da_atuacao ?FilmeAt.
}
ORDER BY ?Pessoa
```

Resultado da consulta:

	Pessoa	FilmeAt	FilmeDir
1.	denzel_washington	the_great_debaters	the_great_debaters
2.	eric_schaeffer	fall	fall
3.	eric_schaeffer	fall	if_lucy_fell
4.	eric_schaeffer	if_lucy_fell	fall
5.	eric_schaeffer	if_lucy_fell	if_lucy_fell
6.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	peggy_sue_got_married
7.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	rumble_fish
8.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_cotton_club
9.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_godfather
10.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_godfather_part_ii
11.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_godfather_part_iii
12.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_outsiders
13.	george_lucas	hearts_of_darkness_a_filmmaker_s_apocalypse	star_wars_episode_i__the_phantom_menace
14.	harold_ramis	ghostbusters	groundhog_day
15.	harold_ramis	groundhog_day	groundhog_day

CONSULTA 5.
Note que esta consulta lista apenas um exemplo por Pessoa.

Parâmetros:

Consulta em SPARQL:

```
SELECT ?Pessoa (SAMPLE(?filme_atuado) AS ?FilmeAt)
(SAMPLE(?filme_dirigido) AS ?FilmeDir)
WHERE {
    ?Pessoa rdf:type :Diretor.
    ?filme_dirigido :dirigido_por ?Pessoa.
    ?Pessoa :atua ?atuacao.
    ?atuacao :filme_da_atuacao ?filme_atuado.
}
GROUP BY ?Pessoa
ORDER BY ?Pessoa
```

Resultado da consulta:

	Pessoa	FilmeAt	FilmeDir
1.	denzel_washington	the_great_debaters	the_great_debaters
2.	eric_schaeffer	fall	if_lucy_fell
3.	francis_ford_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	the_godfather_part_iii
4.	george_lucas	hearts_of_darkness_a_filmmaker_s_apocalypse	star_wars_episode_i__the_phantom_menace
5.	harold_ramis	groundhog_day	groundhog_day
6.	ivan_reitman	ghostbusters	ghostbusters
7.	joel_coen	crimewave	intolerable_cruelty
8.	robert_redford	the_horse_whisperer	the_horse_whisperer
9.	roman_coppola	hearts_of_darkness_a_filmmaker_s_apocalypse	cq
10.	sam_raimi	the_hudsucker_proxy	crimewave
11.	sofia_coppola	the_godfather	marie_antoinette
12.	spike_jonze	torrance_rises	her
13.	stephen_hillenburg	the_spongebob_squarepants_movie	the_spongebob_squarepants_movie
14.	va_g_rdos	an_american_rhapsody	an_american_rhapsody
15.	woody_allen	scoop	scoop

CONSULTA 6.
Mostra que o filtro de Ano funciona, pois não exibe o filme barton_fink de 1991.

Parâmetros:

Ano N_1 = ?$N1$ = 1994
Ano N_2 = ?$N2$ = 2004
Ator X = ?X = :john_turturro
Ator Y = ?Y = :john_goodman

Consulta em SPARQL:

```
SELECT ?Ano ?Diretor ?Filme
WHERE {
    VALUES (?N1 ?N2) {(1994 2004)}
    VALUES (?X ?Y) {(:john_turturro :john_goodman)}.
    ?Filme :ano_lancamento ?Ano.
    FILTER (?Ano >= ?N1  ?Ano <= ?N2).
    ?Filme :dirigido_por ?Diretor.
    ?X :atua ?atuacaoX.
    ?Y :atua ?atuacaoY.
    ?atuacaoX :filme_da_atuacao ?Filme.
    ?atuacaoY :filme_da_atuacao ?Filme.
}
ORDER BY ?Ano
```

Resultado da consulta:

	Ano	Diretor	Filme
1.	1998	joel_coen	the_big_lebowski
2.	2000	joel_coen	o_brother_where_art_thou

CONSULTA 6.
Mostra que são capturados filmes de diretores diferentes.

Parâmetros:

Ano N_1 = ?$N1$ = 1994
Ano N_2 = ?$N2$ = 2004
Ator X = ?X = :issac_freeman
Ator Y = ?Y = :jerry_douglas

Consulta em SPARQL:

```
SELECT ?Ano ?Diretor ?Filme
WHERE {
    VALUES (?N1 ?N2) {(1994 2004)}
    VALUES (?X ?Y) {(:issac_freeman :jerry_douglas)}.
    ?Filme :ano_lancamento ?Ano.
    FILTER (?Ano >= ?N1  ?Ano <= ?N2).
    ?Filme :dirigido_por ?Diretor.
    ?X :atua ?atuacaoX.
    ?Y :atua ?atuacaoY.
    ?atuacaoX :filme_da_atuacao ?Filme.
    ?atuacaoY :filme_da_atuacao ?Filme.
}
ORDER BY ?Ano
```

Resultado da consulta:

	Ano	Diretor	Filme
1.	2000	d_a_pennebaker	down_from_the_mountain
2.	2000	joel_coen	o_brother_where_art_thou

CONSULTA 7.
Lista todas as combinações de pares (ator, atriz) para dois filmes diferentes. (Na imagem aparece apenas um deles)

Parâmetros:

Duração $M_1 = ?M1 = 126$
Duração $M_2 = ?M2 = 126$

Consulta em SPARQL:

```
SELECT ?Atriz ?Ator ?Filme ?Duracao
WHERE {
    VALUES (?M1 ?M2) {(126 126)}
    FILTER (?Duracao >= ?M1  ?Duracao <= ?M2).
    ?Filme :duracao ?Duracao.
    ?atuacaoAtor :filme_da_atuacao ?Filme.
    ?atuacaoAtriz :filme_da_atuacao ?Filme.
    ?Ator rdf:type :Ator.
    ?Ator :atua ?atuacaoAtor.
    ?Atriz rdf:type :Atriz.
    ?Atriz :atua ?atuacaoAtriz.
}
ORDER BY ?Filme
```

Resultado da consulta:

	Atriz	Ator	Filme	Duracao
1.	amy_addams	chris_pratt	her	126
2.	gabe_gomez	chris_pratt	her	126
3.	lisa_renee_pitts	chris_pratt	her	126
4.	lynn_adrianna	chris_pratt	her	126
5.	rooney_mara	chris_pratt	her	126
6.	scarlett_johansson	chris_pratt	her	126
7.	amy_addams	joaquin_phoenix	her	126
8.	gabe_gomez	joaquin_phoenix	her	126
9.	lisa_renee_pitts	joaquin_phoenix	her	126
10.	lynn_adrianna	joaquin_phoenix	her	126
11.	rooney_mara	joaquin_phoenix	her	126
12.	scarlett_johansson	joaquin_phoenix	her	126
13.	amy_addams	matt_letscher	her	126
14.	gabe_gomez	matt_letscher	her	126
15.	lisa_renee_pitts	matt_letscher	her	126

CONSULTA 7.
Mostra um exemplo menor, de apenas 1 filme.

Parâmetros:

Duração $M_1 = ?M1 = 149$
Duração $M_2 = ?M2 = 149$

Consulta em SPARQL:

```
SELECT ?Atriz ?Ator ?Filme ?Duracao
WHERE {
    VALUES (?M1 ?M2) {(149 149)}
    FILTER (?Duracao >= ?M1  ?Duracao <= ?M2).
    ?Filme :duracao ?Duracao.
    ?atuacaoAtor :filme_da_atuacao ?Filme.
    ?atuacaoAtriz :filme_da_atuacao ?Filme.
    ?Ator rdf:type :Ator.
    ?Ator :atua ?atuacaoAtor.
    ?Atriz rdf:type :Atriz.
    ?Atriz :atua ?atuacaoAtriz.
}
ORDER BY ?Filme
```

Resultado da consulta:

	Atriz	Ator	Filme	Duracao
1.	audrey_tautou	alfred_molina	the_da_vinci_code	149
2.	audrey_tautou	ian_mckellen	the_da_vinci_code	149
3.	audrey_tautou	jean_reno	the_da_vinci_code	149
4.	audrey_tautou	jean_yves_berteloot	the_da_vinci_code	149
5.	audrey_tautou	jürgen_prochnow	the_da_vinci_code	149
6.	audrey_tautou	paul_bettany	the_da_vinci_code	149
7.	audrey_tautou	tom_hanks	the_da_vinci_code	149

CONSULTA 8.

Parâmetros:

Primeiro Nome $X_p = ?Xp = "Ewan"$
Último Nome $X_u = ?Xu = "McGregor"$

Consulta em SPARQL:

```
SELECT ?Diretor
WHERE {
    SELECT ?Diretor (count(?Diretor) as ?count)
    WHERE {
        VALUES ?Xp {"Ewan"}.
        VALUES ?Xu {"McGregor"}.
        ?x rdfs:subClassOf :Estrela.
        ?ator rdf:type ?x.
        ?ator :primeiro_nome ?Xp.
        ?ator :nome_familia ?Xu.
        ?ator :atua ?atuacao.
        ?atuacao :filme_da_atuacao ?Filme.
        ?Filme :dirigido_por ?Diretor.
    }
    GROUP BY ?Diretor
    ORDER BY DESC (?count)
    LIMIT 1
}
```

Resultado da consulta:

	Diretor
1.	george_lucas

CONSULTA 9.

Parâmetros:

Ator X = ?ator = :scarlett_johansson

Consulta em SPARQL:

```
SELECT ?Filme ?Ano ?Personagem
WHERE {
    VALUES ?ator {:scarlett_johansson}.
    ?ator :atua ?atuacao.
    ?atuacao :filme_da_atuacao ?Filme.
    ?atuacao :personagem_da_atuacao ?Personagem.
    ?Filme :ano_lancamento ?Ano.
}
ORDER BY ?Ano
LIMIT 1
```

Resultado da consulta:

	Filme	Ano	Personagem
1.	north	1994	laura_nelson

CONSULTA 10.

Parâmetros:

Diretor D = ?diretor = :joel_coen

Consulta em SPARQL:

```
SELECT ?Filme ?Duracao
WHERE {
    VALUES ?diretor {:joel_coen}.
    ?diretor :dirige ?Filme.
    ?Filme :duracao ?Duracao
}
ORDER BY DESC (?Duracao)
LIMIT 1
```

Resultado da consulta:

	Filme	Duracao
1.	no_country_for_old_men	122