MILESTONE 1

CSE488 - Ontologies and the Semantic Web

Submitted by:

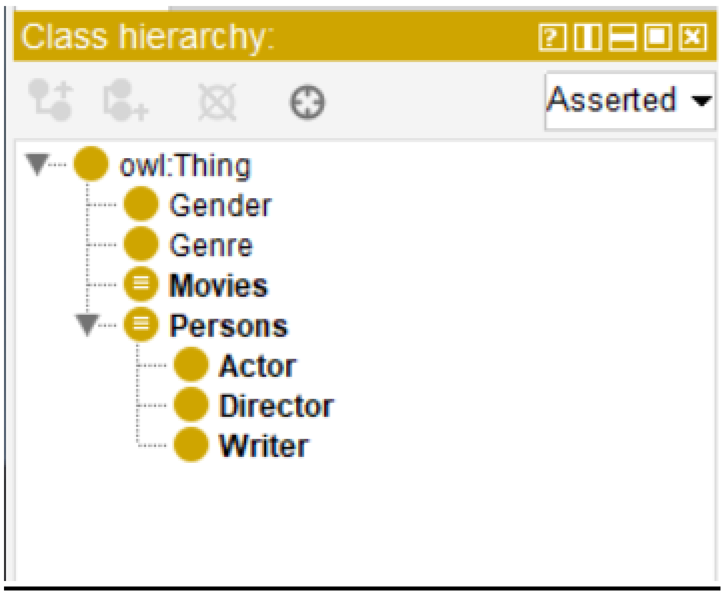
- Laila Yehia Mohamed	19P7649
- Maria Mourad Elia	19P4894
- Menna Tallah Ashraf Salama	19P3575
- Sara Mohamed Taha	19P9266
- Yasmin Haitham Abdelmoaty	18P3102

Submitted to:

- Dr. Ensaf Hussein Mohamed
- Eng. Eman Khaled

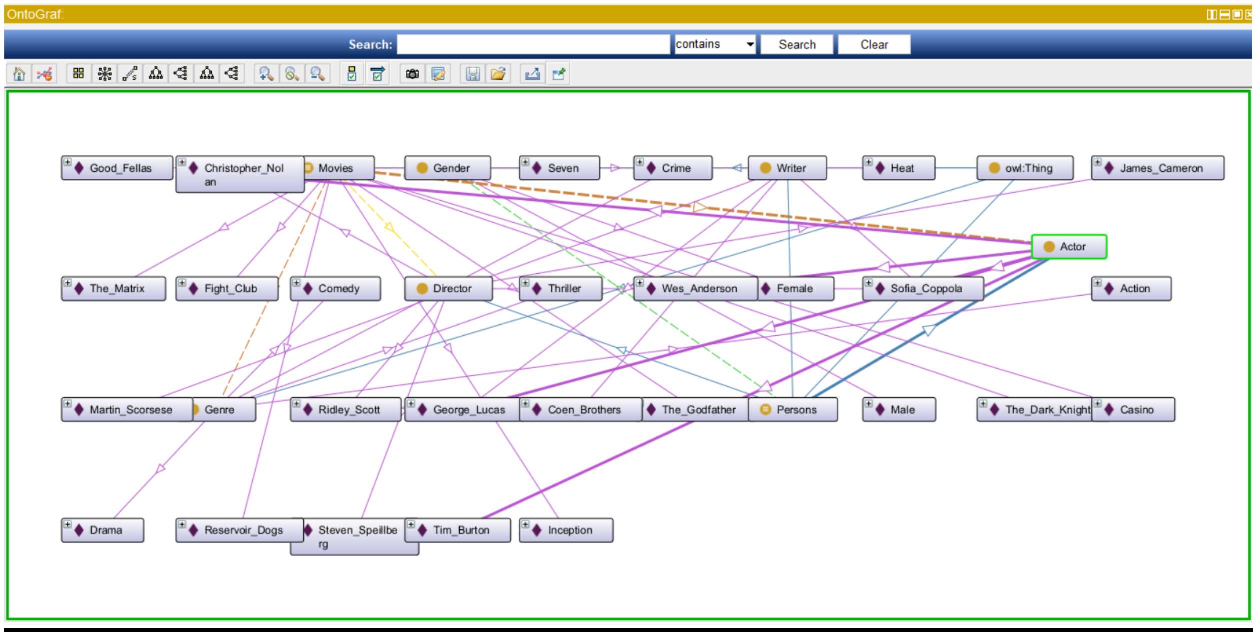
MAY 9, 2024

Class Hierarchy:

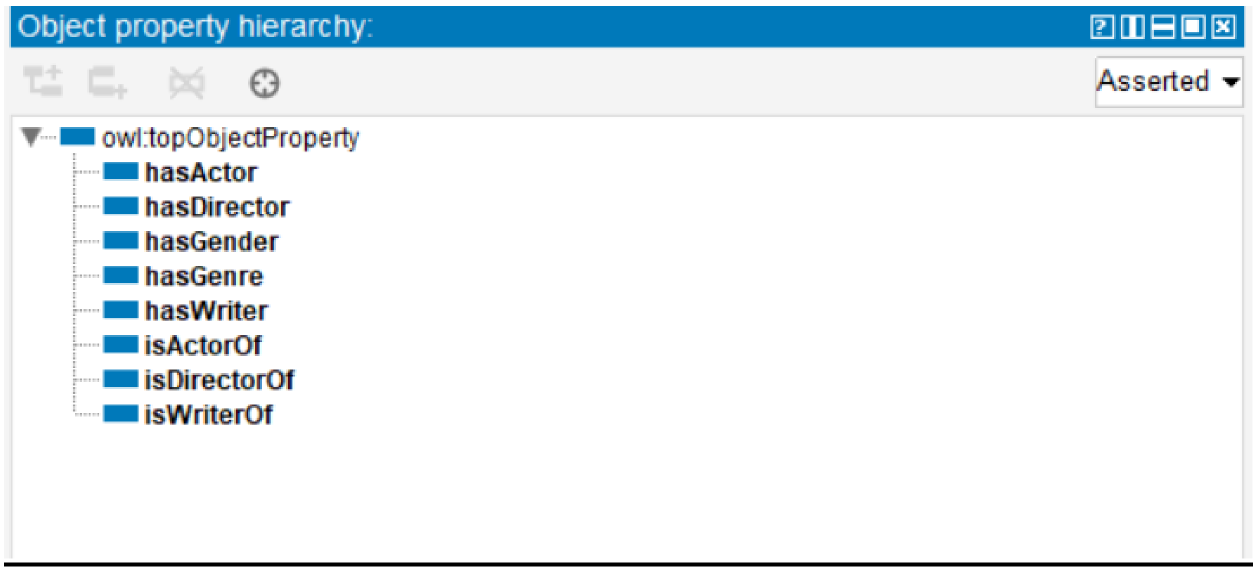


```
owl:Thing
    Gender
    Genre
    Movies
    Persons
        Actor
        Director
        Writer
```

OntoGraf:


OntoGraf:
Search:
contains
Search
Clear
Good_Fellas
Christopher_Nolan
Movies
Gender
Seven
Crime
Writer
Heat
owl:Thing
James_Cameron
Actor
The_Matrix
Fight_Club
Comedy
Director
Thriller
Wes_Anderson
Female
Sofia_Coppola
Action
Martin_Scorsese
Genre
Ridley_Scott
George_Lucas
Coen_Brothers
The_Godfather
Persons
Male
The_Dark_Knight
Casino
Drama
Reservoir_Dogs
Steven_Speilberg
Tim_Burton
Inception

Object Properties:

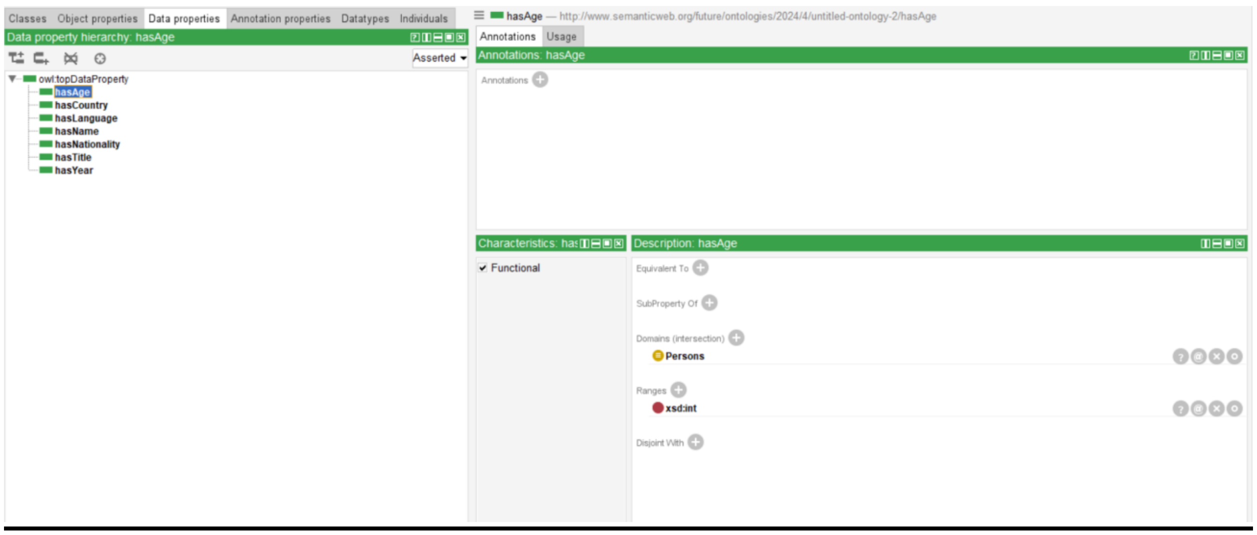


Object property hierarchy:

Asserted

- owl:topObjectProperty
  - hasActor
  - hasDirector
  - hasGender
  - hasGenre
  - hasWriter
  - isActorOf
  - isDirectorOf
  - isWriterOf

Data Properties:



hasAge — http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/hasAge

Data property hierarchy: hasAge

- owl:topDataProperty
  - hasAge
  - hasCountry
  - hasLanguage
  - hasName
  - hasNationality
  - hasTitle
  - hasYear

Annotations: hasAge

Characteristics: hasAge

- ✓ Functional

Description: hasAge

Equivalent To

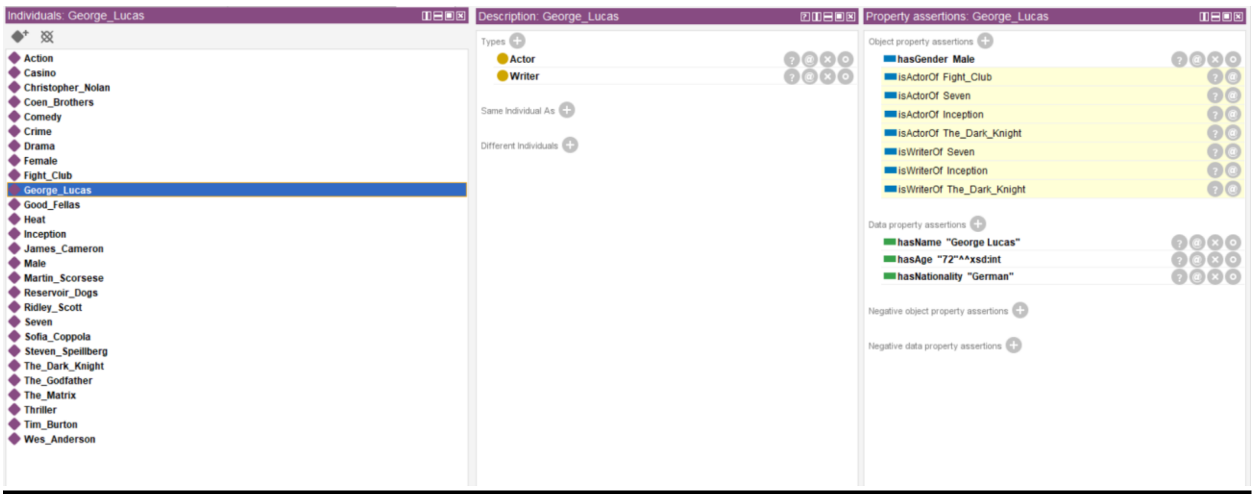
SubProperty Of

Domains (intersection)
Persons

Ranges
xsd:int

Disjoint With

Individuals:



Individuals: George_Lucas

- Action
- Casino
- Christopher_Nolan
- Coen_Brothers
- Comedy
- Crime
- Drama
- Female
- Fight_Club
- George_Lucas
- Good_Fellas
- Heat
- Inception
- James_Cameron
- Male
- Martin_Scorsese
- Reservoir_Dogs
- Ridley_Scott
- Seven
- Sofia_Coppola
- Steven_Speillberg
- The_Dark_Knight
- The_Godfather
- The_Matrix
- Thriller
- Tim_Burton
- Wes_Anderson

Description: George_Lucas

Types
- Actor
- Writer

Same Individual As

Different Individuals

Property assertions: George_Lucas

Object property assertions
- hasGender Male
- isActorOf Fight_Club
- isActorOf Seven
- isActorOf Inception
- isActorOf The_Dark_Knight
- isWriterOf Seven
- isWriterOf Inception
- isWriterOf The_Dark_Knight

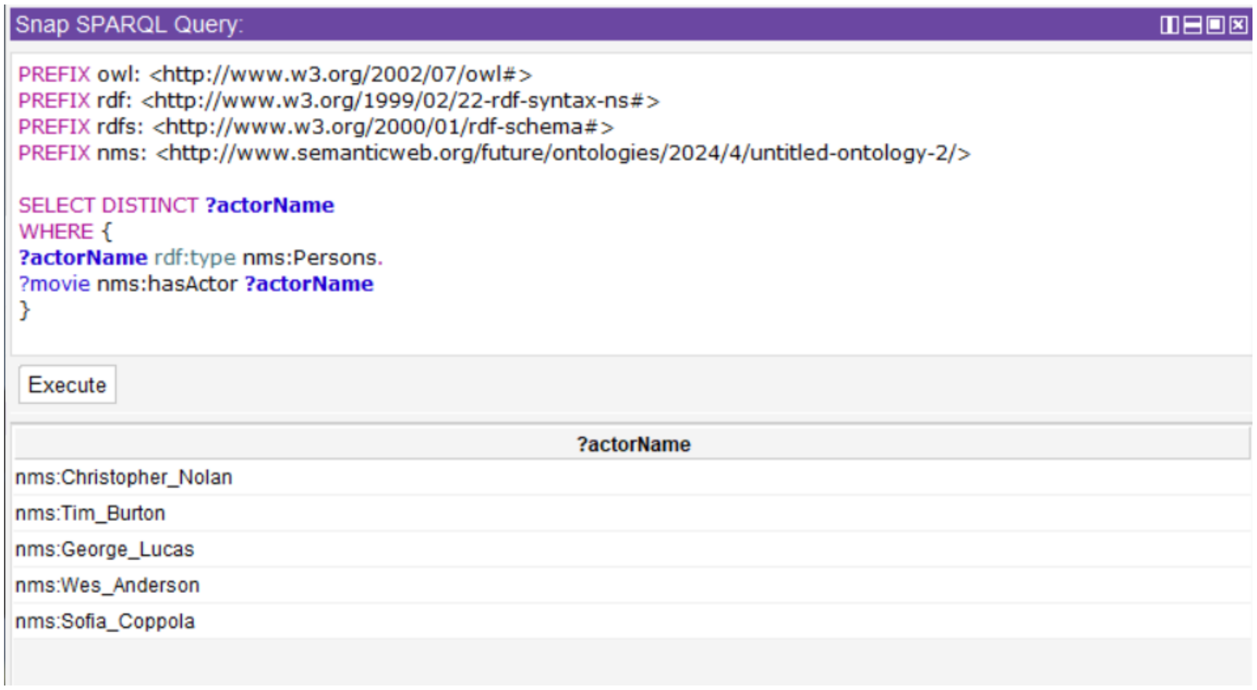
Data property assertions
- hasName "George Lucas"
- hasAge "72"^^xsd:int
- hasNationality "German"

Negative object property assertions

Negative data property assertions

SPARQL Query:

1. List the instances of the class Actor



Snap SPARQL Query:

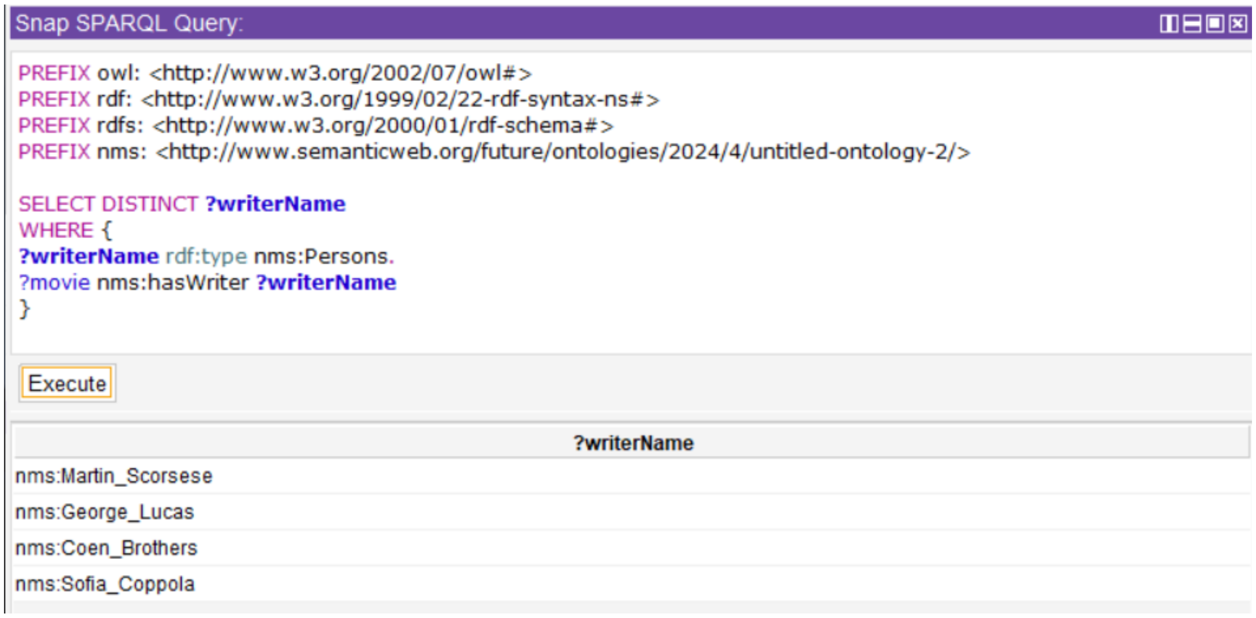
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT ?actorName
WHERE {
?actorName rdf:type nms:Persons.
?movie nms:hasActor ?actorName
}
```

Execute

```
?actorName
nms:Christopher_Nolan
nms:Tim_Burton
nms:George_Lucas
nms:Wes_Anderson
nms:Sofia_Coppola
```

2. List the instances of the class writer



Snap SPARQL Query:

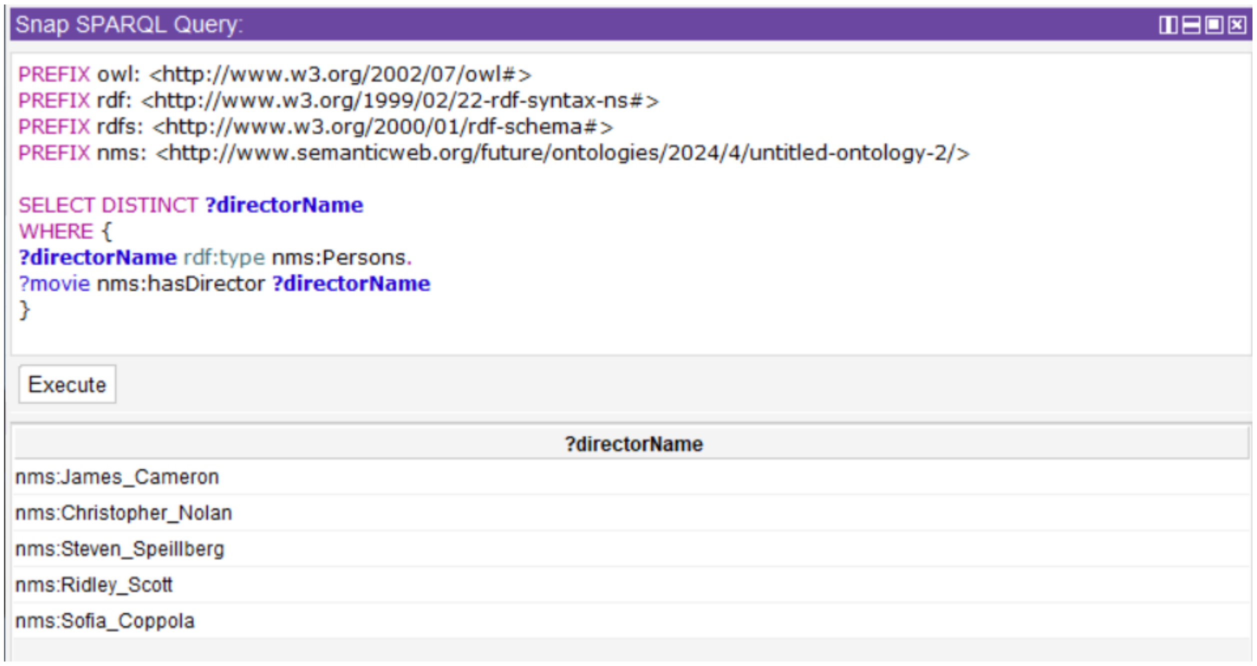
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT ?writerName
WHERE {
?writerName rdf:type nms:Persons.
?movie nms:hasWriter ?writerName
}
```

Execute

```
?writerName
nms:Martin_Scorsese
nms:George_Lucas
nms:Coen_Brothers
nms:Sofia_Coppola
```

3. List the instances of the class director



Snap SPARQL Query:

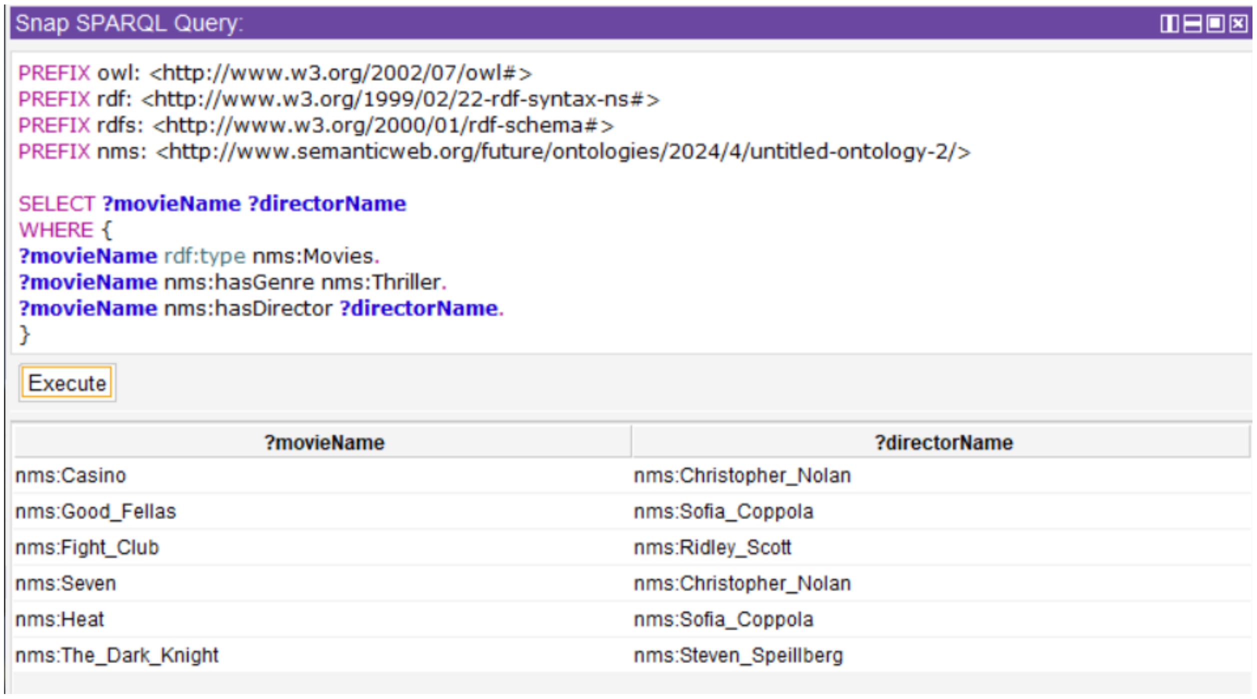
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT ?directorName
WHERE {
?directorName rdf:type nms:Persons.
?movie nms:hasDirector ?directorName
}
```

Execute

?directorName
nms:James_Cameron
nms:Christopher_Nolan
nms:Steven_Speillberg
nms:Ridley_Scott
nms:Sofia_Coppola

4. List the name of all Thriller movies. For each one, display its director.



Snap SPARQL Query:

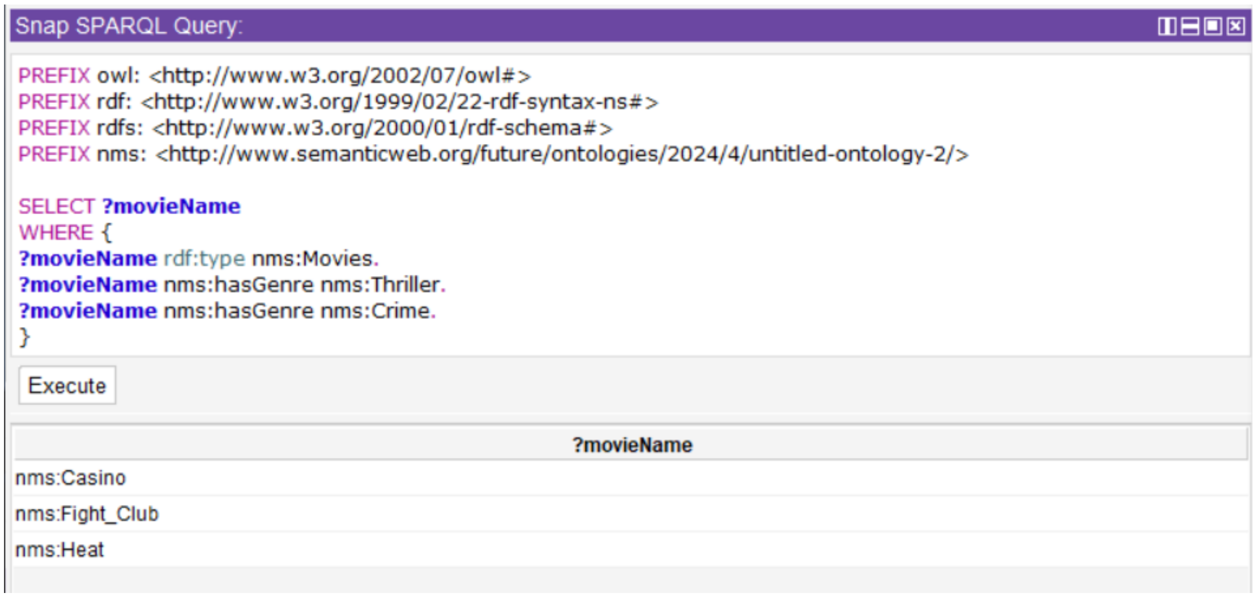
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT ?movieName ?directorName
WHERE {
?movieName rdf:type nms:Movies.
?movieName nms:hasGenre nms:Thriller.
?movieName nms:hasDirector ?directorName.
}
```

Execute

?movieName	?directorName
nms:Casino	nms:Christopher_Nolan
nms:Good_Fellas	nms:Sofia_Coppola
nms:Fight_Club	nms:Ridley_Scott
nms:Seven	nms:Christopher_Nolan
nms:Heat	nms:Sofia_Coppola
nms:The_Dark_Knight	nms:Steven_Speillberg

5. List the name of all Crime Thriller movies.


Snap SPARQL Query:

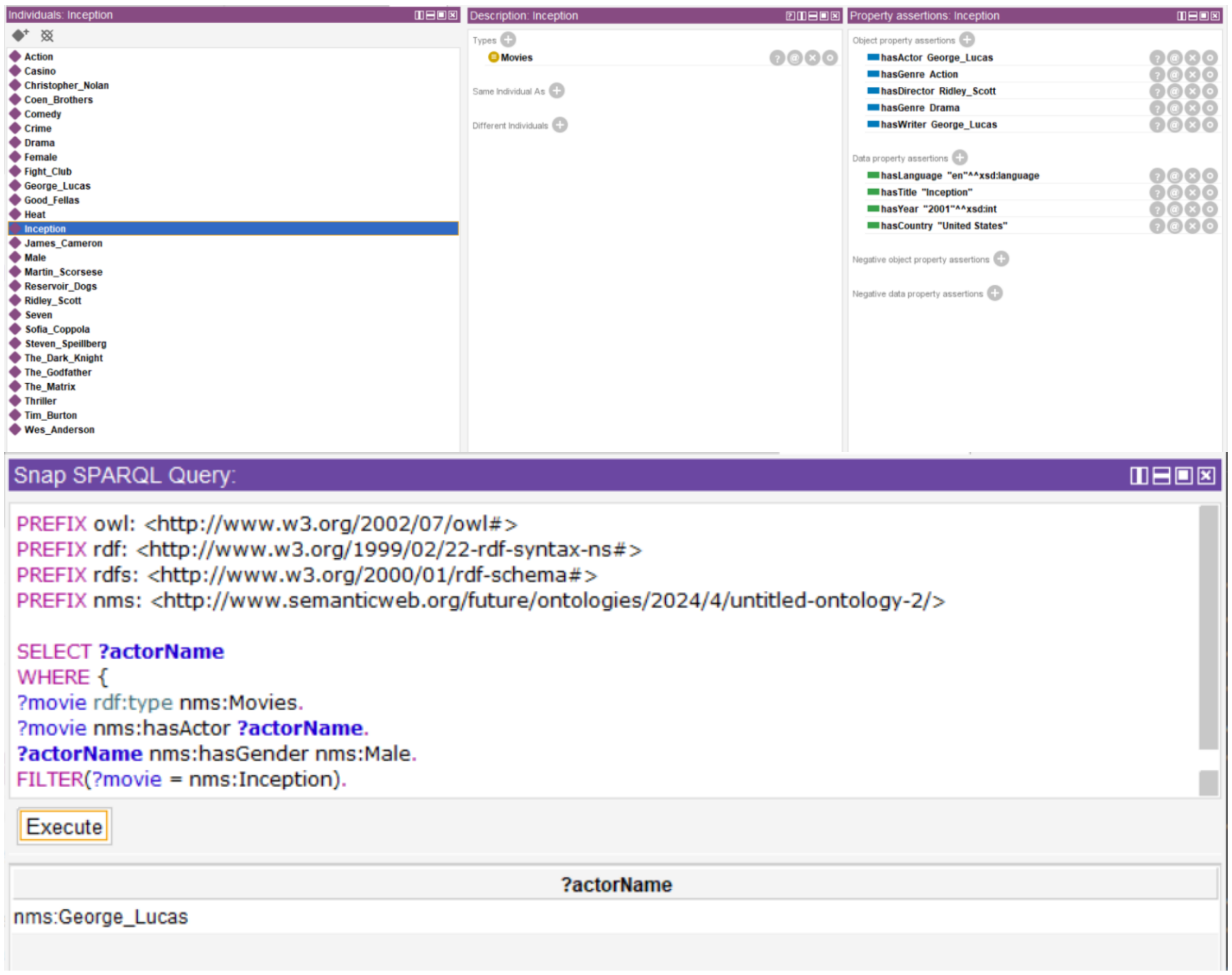
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT ?movieName
WHERE {
?movieName rdf:type nms:Movies.
?movieName nms:hasGenre nms:Thriller.
?movieName nms:hasGenre nms:Crime.
}
```

Execute

?movieName
nms:Casino
nms:Fight_Club
nms:Heat

6. List the male actors in the movie in specific film


Individuals: Inception

Action
Casino
Christopher_Nolan
Coen_Brothers
Comedy
Crime
Drama
Female
Fight_Club
George_Lucas
Good_Fellas
Heat
Inception
James_Cameron
Male
Martin_Scorsese
Reservoir_Dogs
Ridley_Scott
Seven
Sofia_Coppola
Steven_Speillberg
The_Dark_Knight
The_Godfather
The_Matrix
Thriller
Tim_Burton
Wes_Anderson

Description: Inception

Types
Movies
Same Individual As
Different Individuals

Property assertions: Inception

Object property assertions
hasActor George_Lucas
hasGenre Action
hasDirector Ridley_Scott
hasGenre Drama
hasWriter George_Lucas

Data property assertions
hasLanguage "en"^^xsd:language
hasTitle "Inception"
hasYear "2001"^^xsd:int
hasCountry "United States"

Negative object property assertions
Negative data property assertions

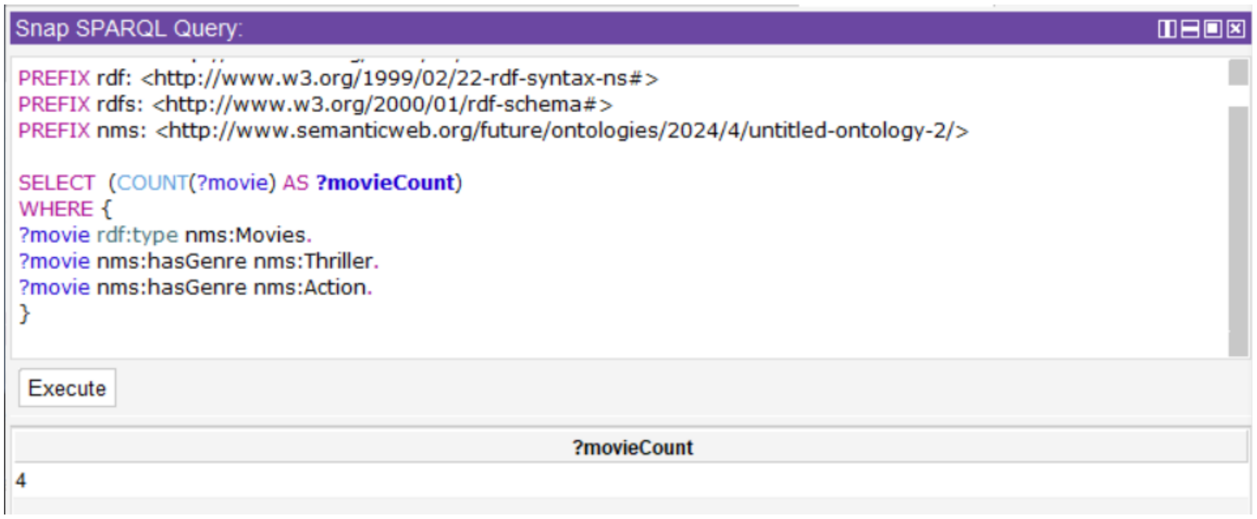
Snap SPARQL Query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT ?actorName
WHERE {
?movie rdf:type nms:Movies.
?movie nms:hasActor ?actorName.
?actorName nms:hasGender nms:Male.
FILTER(?movie = nms:Inception).
```

?actorName
nms:George_Lucas

7. How many movies have both "Action" and "Thriller" as genres?

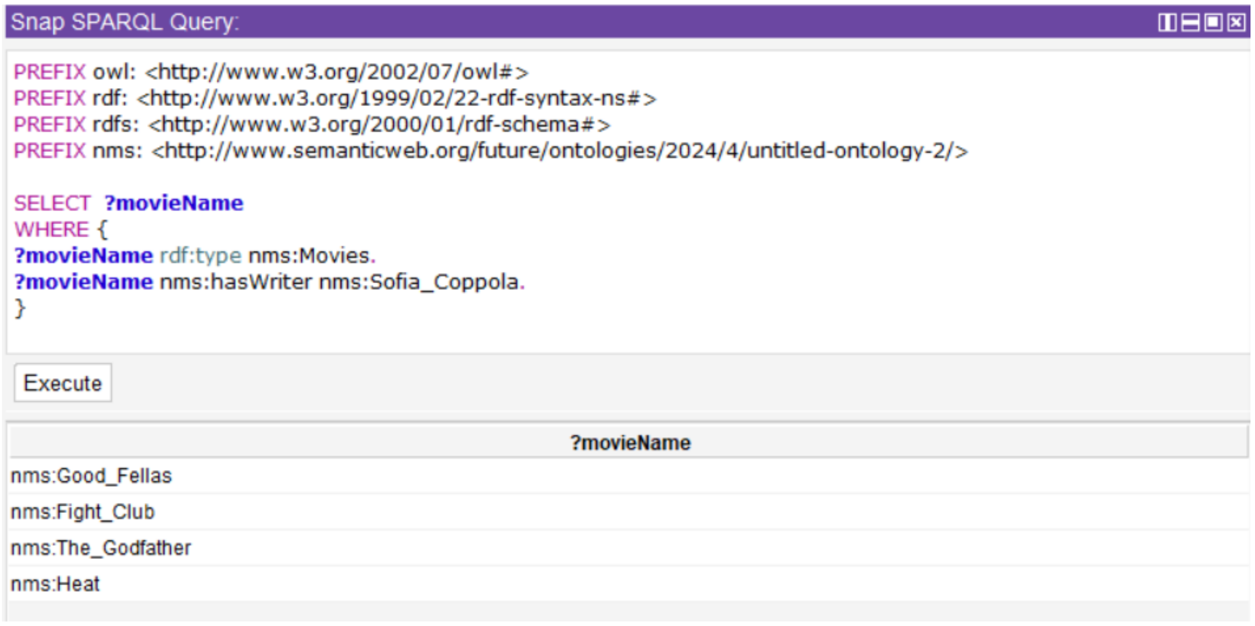

Snap SPARQL Query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT  (COUNT(?movie) AS ?movieCount)
WHERE {
?movie rdf:type nms:Movies.
?movie nms:hasGenre nms:Thriller.
?movie nms:hasGenre nms:Action.
}
```

?movieCount
4

8. List all the movies written by a specific writer.


Snap SPARQL Query:

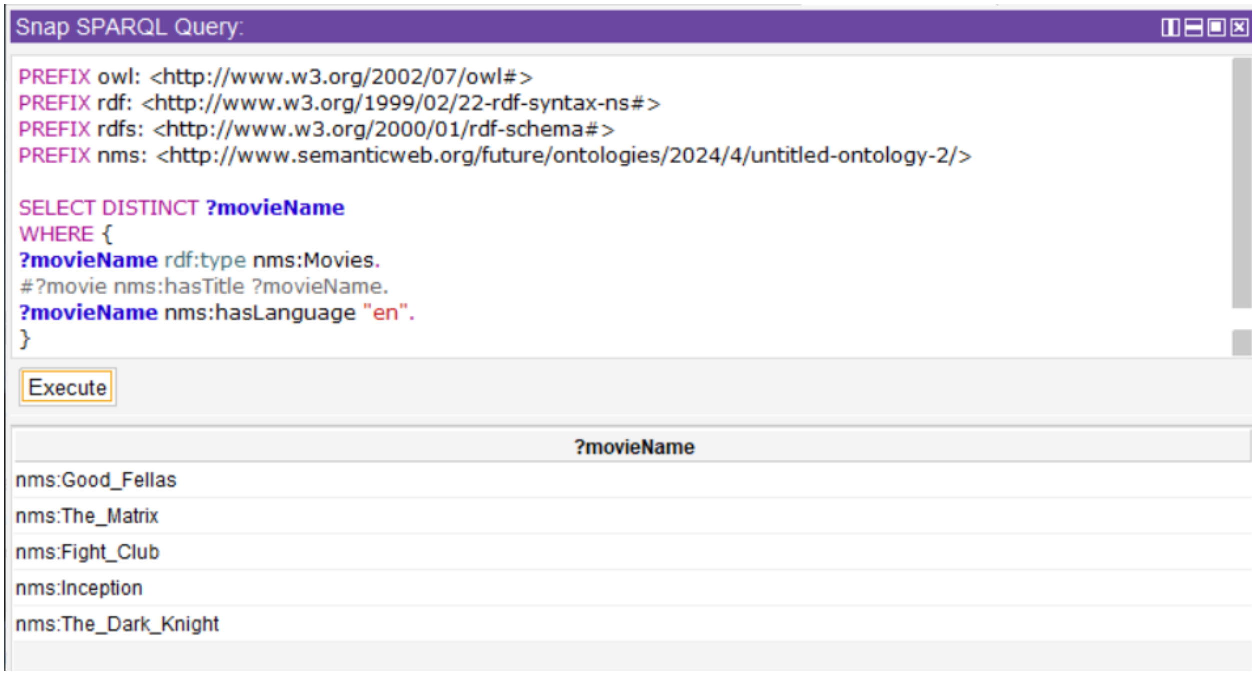
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT  ?movieName
WHERE {
?movieName rdf:type nms:Movies.
?movieName nms:hasWriter nms:Sofia_Coppola.
}
```

Execute

?movieName
nms:Good_Fellas
nms:Fight_Club
nms:The_Godfather
nms:Heat

9. Find movies with a certain language.


Snap SPARQL Query:

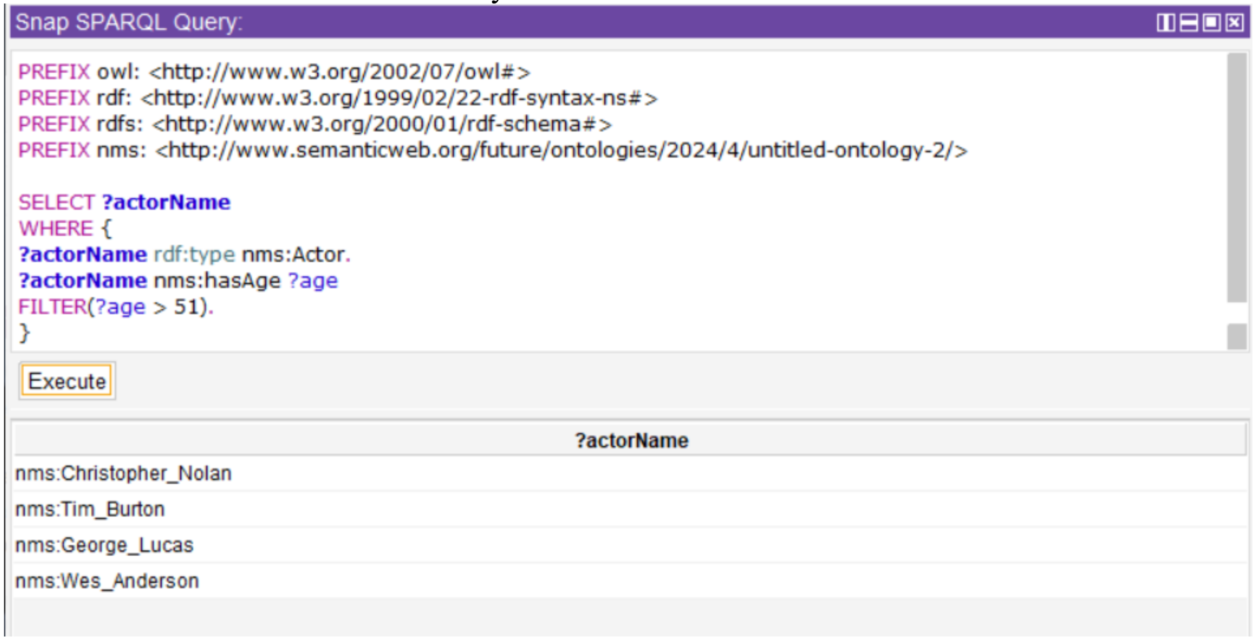
```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT ?movieName
WHERE {
?movieName rdf:type nms:Movies.
#?movie nms:hasTitle ?movieName.
?movieName nms:hasLanguage "en".
}
```

Execute

?movieName
nms:Good_Fellas
nms:The_Matrix
nms:Fight_Club
nms:Inception
nms:The_Dark_Knight

10. List the name of Actors older than 51 years.



```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT ?actorName
WHERE {
?actorName rdf:type nms:Actor.
?actorName nms:hasAge ?age
FILTER(?age > 51).
}
```

?actorName
nms:Christopher_Nolan
nms:Tim_Burton
nms:George_Lucas
nms:Wes_Anderson

EXTRA SPARQL Queries:

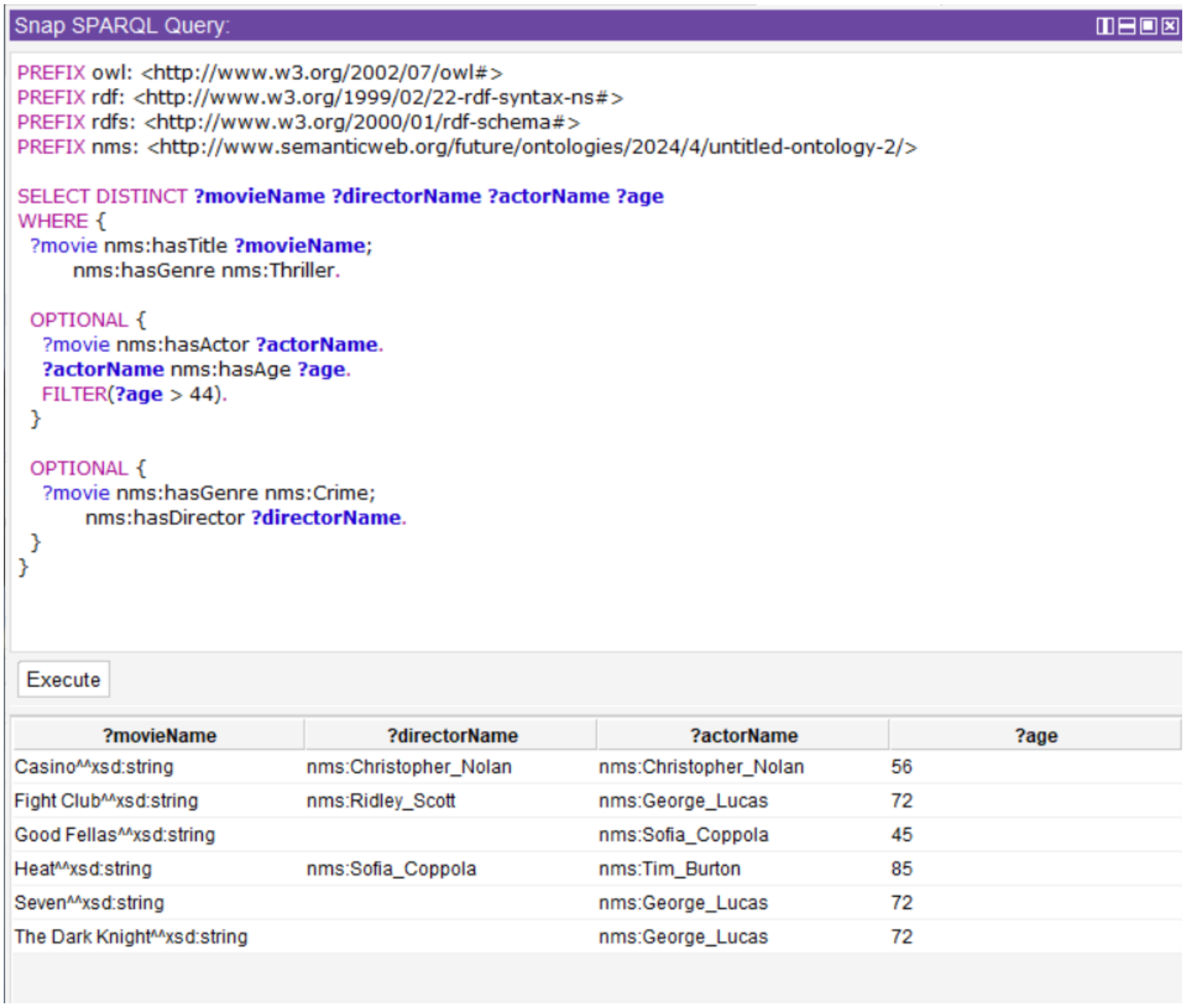
1. Output all the thriller movies, and if one of those thrillers is crime then print also its director. Also, if one of those thrillers have actors that are older than 44, then output the names and ages of those actors

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms:
<http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-
2/>

SELECT DISTINCT ?movieName ?directorName ?actorName ?age
WHERE {
  ?movie nms:hasTitle ?movieName;
         nms:hasGenre nms:Thriller.

  OPTIONAL {
    ?movie nms:hasActor ?actorName.
    ?actorName nms:hasAge ?age.
    FILTER(?age > 44).
  }

  OPTIONAL {
    ?movie nms:hasGenre nms:Crime;
           nms:hasDirector ?directorName.
  }
}
```



Snap SPARQL Query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT ?movieName ?directorName ?actorName ?age
WHERE {
 ?movie nms:hasTitle ?movieName;
      nms:hasGenre nms:Thriller.

 OPTIONAL {
  ?movie nms:hasActor ?actorName.
  ?actorName nms:hasAge ?age.
  FILTER(?age > 44).
 }

 OPTIONAL {
  ?movie nms:hasGenre nms:Crime;
       nms:hasDirector ?directorName.
 }
}
```

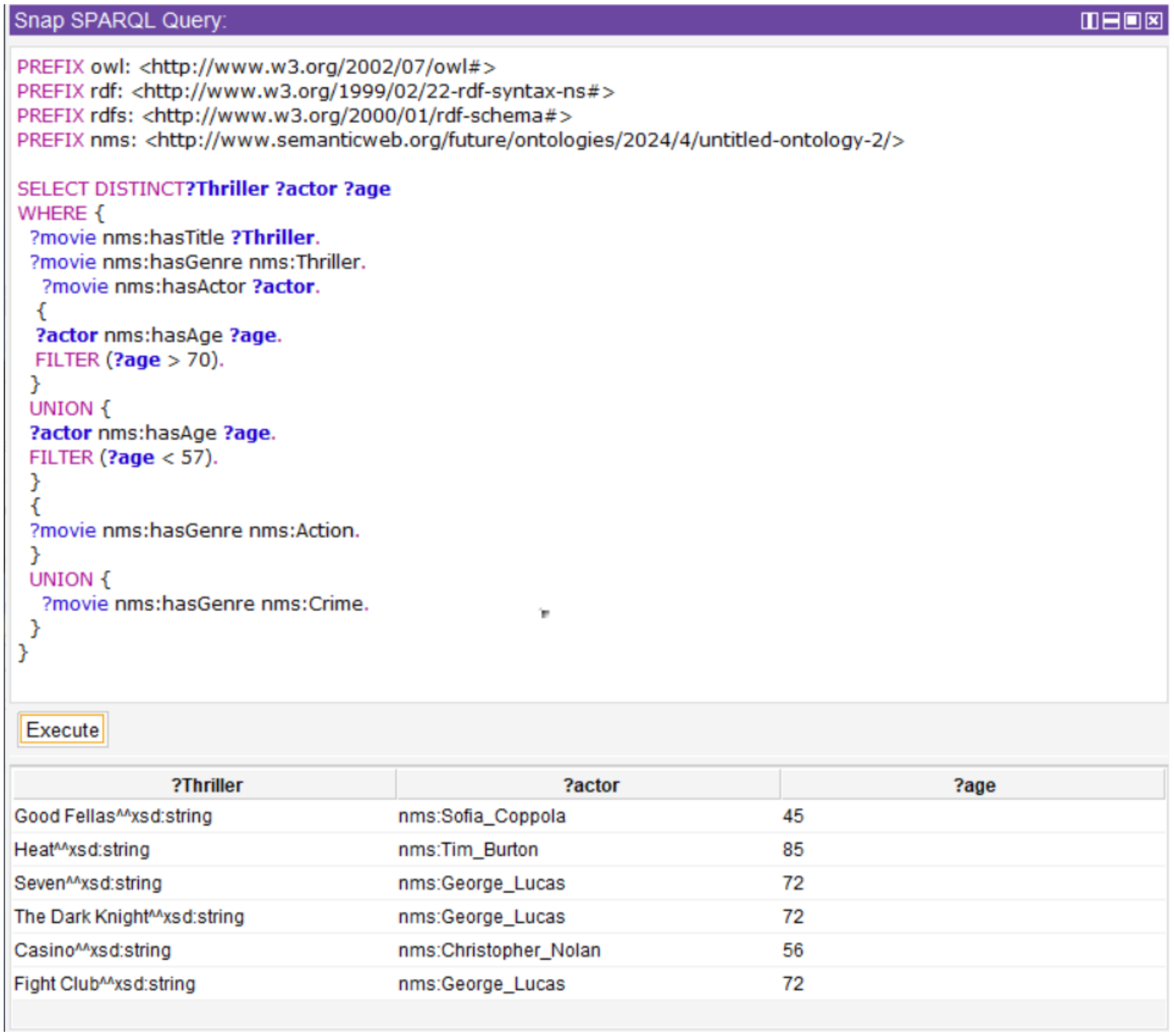
Execute

?movieName	?directorName	?actorName	?age
Casino^^xsd:string	nms:Christopher_Nolan	nms:Christopher_Nolan	56
Fight Club^^xsd:string	nms:Ridley_Scott	nms:George_Lucas	72
Good Fellas^^xsd:string		nms:Sofia_Coppola	45
Heat^^xsd:string	nms:Sofia_Coppola	nms:Tim_Burton	85
Seven^^xsd:string		nms:George_Lucas	72
The Dark Knight^^xsd:string		nms:George_Lucas	72

2. This query outputs all thriller movies that are either also Action movies or Crime movies. Also, outputs the actors' age that are either age<57 or >70.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms:
<http://www.semanticweb.org/future/ontologies/2024/4/untitled-
ontology-2/>

SELECT DISTINCT ?Thriller ?actor ?age
WHERE {
 ?movie nms:hasTitle ?Thriller.
 ?movie nms:hasGenre nms:Thriller.
  ?movie nms:hasActor ?actor.
  {
  ?actor nms:hasAge ?age.
  FILTER (?age > 70).
 }
 UNION {
 ?actor nms:hasAge ?age.
 FILTER (?age < 57).
 }
 {
 ?movie nms:hasGenre nms:Action.
 }
 UNION {
  ?movie nms:hasGenre nms:Crime.
 }
}
```


Snap SPARQL Query:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms: <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

SELECT DISTINCT?Thriller ?actor ?age
WHERE {
 ?movie nms:hasTitle ?Thriller.
 ?movie nms:hasGenre nms:Thriller.
  ?movie nms:hasActor ?actor.
  {
  ?actor nms:hasAge ?age.
  FILTER (?age > 70).
 }
 UNION {
 ?actor nms:hasAge ?age.
 FILTER (?age < 57).
 }
 {
 ?movie nms:hasGenre nms:Action.
 }
 UNION {
  ?movie nms:hasGenre nms:Crime.
 }
}
```

Execute

?Thriller	?actor	?age
Good Fellas^^xsd:string	nms:Sofia_Coppola	45
Heat^^xsd:string	nms:Tim_Burton	85
Seven^^xsd:string	nms:George_Lucas	72
The Dark Knight^^xsd:string	nms:George_Lucas	72
Casino^^xsd:string	nms:Christopher_Nolan	56
Fight Club^^xsd:string	nms:George_Lucas	72

3. This query constructs triples where each movie has Sofia Coppola as one of its actors.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nms:
<http://www.semanticweb.org/future/ontologies/2024/4/untitled-
ontology-2/>
PREFIX nmsP:
<http://www.semanticweb.org/future/ontologies/2024/4/untitled-
ontology-2#>

CONSTRUCT {?movie nms:hasActor ?actor}
WHERE {
  ?movie nms:hasActor ?actor
  FILTER (?actor = 'Sofia_Coppola').
}
```

4. This query checks whether the individual "Riddley_Scott" has an age property with the value of 37. If the provided information matches the data in the ontology, the query will return true. Otherwise, it will return false.

   ```
   PREFIX owl: <http://www.w3.org/2002/07/owl#>
   PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
   PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
   PREFIX nms:
   <http://www.semanticweb.org/future/ontologies/2024/4/untitled-ontology-2/>

   ASK
   WHERE {
     nms:Riddley_Scott nms:hasAge 37.
   }
   ```