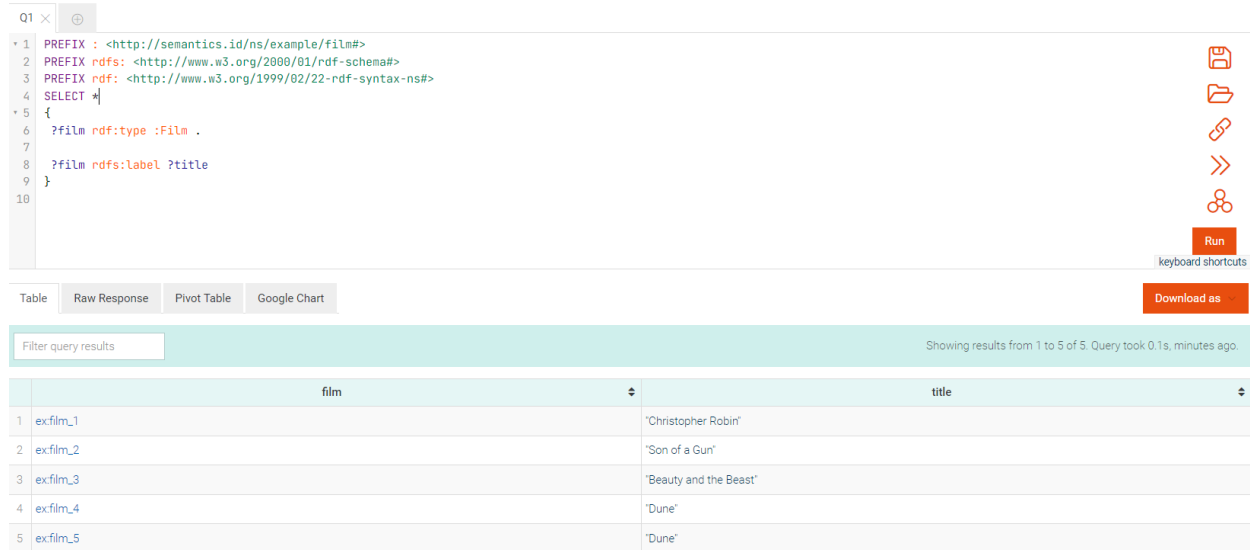


Assignment 2 - Querying Knowledge Graphs with SPARQL

Teodor Chakarov 12141198

Q2:

Get the film titles



The screenshot shows a SPARQL query interface. The query is as follows:

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4 SELECT *
5 {
6   ?film rdf:type :Film .
7
8   ?film rdfs:label ?title
9 }
10
```

Below the query editor, there are tabs for 'Table', 'Raw Response', 'Pivot Table', and 'Google Chart'. The 'Table' tab is selected, showing the following results:

	film	title
1	ex:film_1	'Christopher Robin'
2	ex:film_2	'Son of a Gun'
3	ex:film_3	'Beauty and the Beast'
4	ex:film_4	'Dune'
5	ex:film_5	'Dune'

At the bottom right, there is a 'Download as' button and a status bar indicating 'Showing results from 1 to 5 of 5. Query took 0.1s, minutes ago.'

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT *

{

?film rdf:type :Film .

?film rdfs:label ?title

}

Q4

Are there movies before 1984 called Dunes ?

Q1 × Q4 ×

```
1 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4 ASK {
5   ?x rdfs:label "Dune" .
6   ?x :releaseYear ?releaseYear .
7   FILTER (?releaseYear < "1984"^^xsd:integer)
8 }
```

Run keyboard shortcuts

Query took 0.1s, moments ago.

NO

Q6

Describe the movie Dune made in 1984 year.

Q1 × Q4 × Q6 ×

```
1 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
4 Describe * {
5   ?x rdfs:label "Dune".
6   ?x :releaseYear "1984"^^xsd:integer
7 }
```

Run keyboard shortcuts

Table Raw Response Pivot Table Google Chart Download as Visual

Filter query results Showing results from 1 to 8 of 8. Query took 0.1s, minutes ago.

	subject	predicate	object
1	ex:film_5	rdfs:type	Film
2	ex:film_5	rdfs:type	Artwork
3	ex:film_5	rdfs:type	owl:NamedIndividual
4	ex:film_5	rdfs:label	"Dune"
5	ex:film_5	hasActor	kyle_macLachlan
6	ex:film_5	hasPerformer	kyle_macLachlan
7	ex:film_5	hasGenre	genre_science_fiction
8	ex:film_5	releaseYear	"1984"^^xsd:integer

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

```
Describe * {
    ?x rdfs:label "Dune".

    ?x :releaseYear "1984"^^xsd:integer
}
```

Q7

Get the Film studio and the writers for each movie.

Q1

Q4

Q6

Q7

Q9

```

1 PREFIX : <http://semantics.id/ns/example/film#>
2 Construct {
3   ?x :hasFilmStudio ?FilmStudio.
4   ?x :hasScriptWriter ?ScriptWriter.
5 } Where {
6   ?x :hasScriptWriter ?ScriptWriter.
7   ?x :hasFilmStudio ?FilmStudio.
8 }
```

Run

keyboard shortcuts

Table

Raw Response

Pivot Table

Google Chart

Download as

Visual

Filter query results

Showing results from 1 to 9 of 9. Query took 0.1s, today at 15:11.

	subject	predicate	object
1	ex:film_1	hasFilmStudio	ex:WaltDisneyPictures
2	ex:film_1	hasScriptWriter	ex:writer_1
3	ex:film_1	hasScriptWriter	ex:writer_2
4	ex:film_2	hasFilmStudio	ex:EntertainmentOne
5	ex:film_2	hasScriptWriter	ex:julius_avery
6	ex:film_2	hasScriptWriter	ex:writer_3
7	ex:film_3	hasFilmStudio	ex:WaltDisneyPictures
8	ex:film_3	hasScriptWriter	ex:writer_4
9	ex:film_3	hasScriptWriter	ex:writer_5

PREFIX : <http://semantics.id/ns/example/film#>

Construct {

 ?x :hasFilmStudio ?FilmStudio.

 ?x :hasScriptWriter ?ScriptWriter.

} Where {

 ?x :hasScriptWriter ?ScriptWriter.

 ?x :hasFilmStudio ?FilmStudio.

}

Q9

Get the script Writer and the film studio he/she work in.

The screenshot shows a SPARQL query editor with the following query:

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 select ?ScriptWriter ?FilmStudio {
3   ?x :hasScriptWriter ?ScriptWriter .
4   ?x :hasFilmStudio ?FilmStudio .
5 }
6
```

Below the query editor is a table with two columns: **ScriptWriter** and **FilmStudio**. The table contains 6 rows of results.

	ScriptWriter	FilmStudio
1	ex:writer_1	ex:WaltDisneyPictures
2	ex:writer_2	ex:WaltDisneyPictures
3	ex:julius_avery	ex:EntertainmentOne
4	ex:writer_3	ex:EntertainmentOne
5	ex:writer_4	ex:WaltDisneyPictures
6	ex:writer_5	ex:WaltDisneyPictures

PREFIX : <http://semantics.id/ns/example/film#>

select ?ScriptWriter ?FilmStudio {

 ?x :hasScriptWriter ?ScriptWriter .

 ?x :hasFilmStudio ?FilmStudio .

}

Q11

Get actors played in moves released between 2014 and 2020

The screenshot shows a SPARQL query editor with the following query:

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
3 Select DISTINCT ?hasActor {
4   ?x :hasActor ?hasActor .
5   ?x :releaseYear ?releaseYear .
6   Filter(?releaseYear > "2014"^^xsd:integer && ?releaseYear < "2020"^^xsd:integer)
7 }
8
```

Below the query editor is a table with one column: **hasActor**. The table contains 4 rows of results.

	hasActor
1	ex:dan_stevens
2	ex:emma_watson
3	ex:ewan_mcgregor
4	ex:hayley_atwell

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

Select DISTINCT ?hasActor {

`?x :hasActor ?hasActor.`

`?x :releaseYear ?releaseYear.`

`Filter(?releaseYear > "2014"^^xsd:integer && ?releaseYear < "2020"^^xsd:integer) }`

Q13

Get the names of the movies released between 2014 and 2020.

```
Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × ⓘ
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 Select DISTINCT ?label {
5   ?x :releaseYear ?releaseYear.
6   ?x :hasGenre ?hasGenre .
7   ?x rdfs:label ?label.
8   Filter(?releaseYear > "2014"^^xsd:integer && ?releaseYear < "2020"^^xsd:integer)
9 }
Run
Press Alt+Enter keyboard shortcuts
Download as
```

Table Raw Response Pivot Table Google Chart

Filter query results Showing results from 1 to 2 of 2. Query took 0.1s, moments ago.

	label
1	"Beauty and the Beast"
2	"Christopher Robin"

`PREFIX : <http://semantics.id/ns/example/film#>`

`PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>`

`PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>`

`Select DISTINCT ?label {`

`?x :releaseYear ?releaseYear.`

`?x :hasGenre ?hasGenre .`

`?x rdfs:label ?label.`

`Filter(?releaseYear > "2014"^^xsd:integer && ?releaseYear < "2020"^^xsd:integer)`

`}`

Q14

Get the actors and their birthdates.

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × ⓘ

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 SELECT ?name ?bday {
3   ?x :hasActor ?hasActor .
4   ?hasActor :fullName ?name.
5   ?hasActor :dateOfBirth ?bday.
6 }
7 ORDER BY ASC(?bday)
```

📄 📁 🔗 ⏏️ ⚙️

Run

keyboard shortcuts

Table Raw Response Pivot Table Google Chart

Download as

Filter query results

Showing results from 1 to 8 of 8. Query took 0.1s, moments ago.

	name	bday
1	"Kyle Merritt MacLachlan"	"1959-02-22""xsd:date
2	"Ewan McGregor"	"1971-03-31""xsd:date
3	"Ewan McGregor"	"1971-03-31""xsd:date
4	"Ewan McGregor"	"1971-03-31""xsd:date
5	"Hayley Atwell"	"1982-04-05""xsd:date
6	"Dan Stevens"	"1982-10-10""xsd:date
7	"Emma Watson"	"1990-04-15""xsd:date
8	"Zendaya Maree Stoermer Coleman"	"1996-09-01""xsd:date

PREFIX : <http://semantics.id/ns/example/film#>

Select ?name ?bday {

 ?x :hasActor ?hasActor .

 ?hasActor :fullName ?name.

 ?hasActor :dateOfBirth ?bday.

}






Order by ASC(?bday)

Q16

Get the name of the movie and the writer, orderd by the writers name.

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × ⓘ

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 SELECT ?label ?name {
4   ?x a :Film .
5   ?x rdfs:label ?label .
6   ?x :hasScriptWriter ?writer.
7   ?writer :fullName ?name
8 }
9 Order by DESC(?writer)
10
```



Run

keyboard shortcuts

Table Raw Response Pivot Table Google Chart

Download as

Filter query results Showing results from 1 to 6 of 6. Query took 0.1s, moments ago.

	label	name
1	"Beauty and the Beast"	"Evan Spiliotopoulos"
2	"Beauty and the Beast"	"Stephen Chbosky"
3	"Son of a Gun"	"John Collee"
4	"Christopher Robin"	"Alex Ross"
5	"Christopher Robin"	"Tom McCarthy"
6	"Son of a Gun"	"Julius Avery"

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?label ?name {

 ?x a :Film .

 ?x rdfs:label ?label .

 ?x :hasScriptWriter ?writer.

 ?writer :fullName ?name

}

Order by DESC(?writer)

Q17

Get the title of the movie and the name of the actor, order by name.

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × Q17 ×

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4 select ?title ?name
5 where{
6   ?film a :Film .
7   ?film rdfs:label ?title
8   {?film :hasActor ?crew} union
9   {?film :hasScriptWriter ?crew} union
10  {?film :hasDirector ?crew}
11  ?crew :fullName ?name .
12 }
13 order by ?name
```

Run keyboard shortcuts

Table Raw Response Pivot Table Google Chart

Download as

Filter query results

Showing results from 1 to 18 of 18. Query took 0.1s, moments ago

	title	name
1	"Christopher Robin"	"Alex Ross"
2	"Son of a Gun"	"Alicia Vikander"
3	"Beauty and the Beast"	"Bill Condon"
4	"Beauty and the Beast"	"Dan Stevens"
5	"Beauty and the Beast"	"Emma Watson"
6	"Beauty and the Beast"	"Evan Spiliotopoulos"
7	"Christopher Robin"	"Ewan McGregor"
8	"Son of a Gun"	"Ewan McGregor"
9	"Beauty and the Beast"	"Ewan McGregor"
10	"Christopher Robin"	"Hayley Atwell"
11	"Son of a Gun"	"John Collee"

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?title ?name

where{

 ?film a :Film .

 ?film rdfs:label ?title

 {?film :hasActor ?crew} union

 {?film :hasScriptWriter ?crew} union

 {?film :hasDirector ?crew}


 ?crew :fullName ?name .

}

order by ?name

Q18

Counting the number of movies based in South Korea



```
1 PREFIX dbp: <http://dbpedia.org/property/>
2 PREFIX db: <http://dbpedia.org/>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX movie: <http://dbpedia.org/ontology/>
6 SELECT (count(?title) as ?movieNumber)
7 {
8   ?title rdf:type movie:Film.
9   ?title dbp:country ?a .
10  ?a rdfs:label ?country .
11  FILTER CONTAINS(?country,"South Korea")
12 }
```

Table

Response

Gallery

Chart

Geo

Geo-3D

Geo events

Markup

Network

Pivot

movieNumber

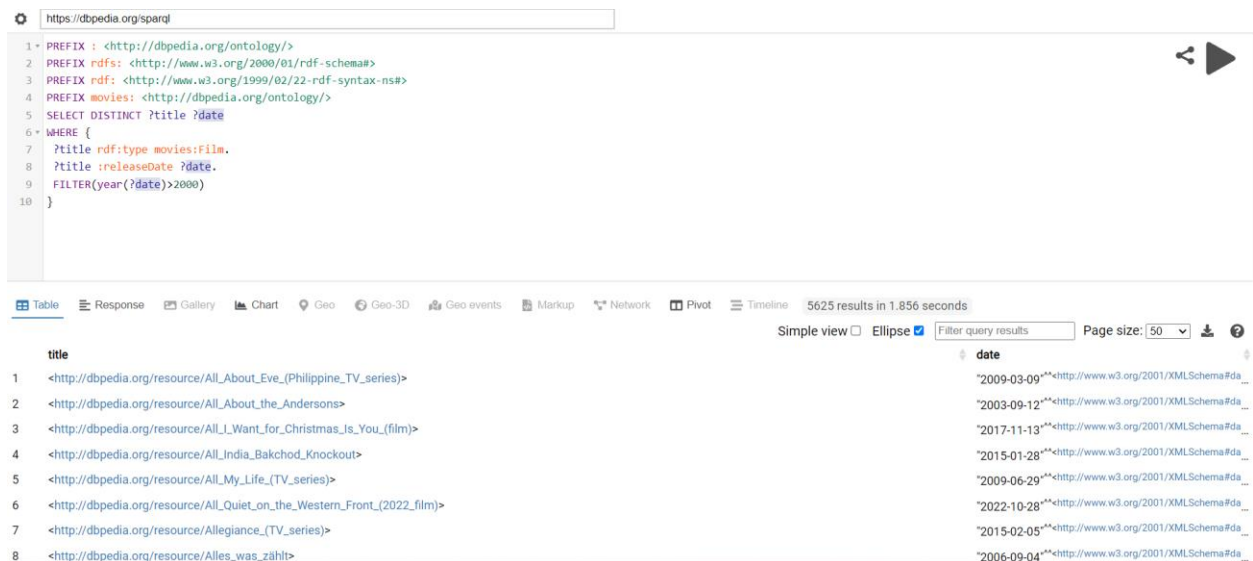
1 "132"^^<http://www.w3.org/2001/XMLSchema#integer>

Showing 1 to 1 of 1 entries

```
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX db: <http://dbpedia.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX movie: <http://dbpedia.org/ontology/>
SELECT (count(?title) as ?movieNumber)
{
  ?title rdf:type movie:Film.
  ?title dbp:country ?a .
  ?a rdfs:label ?country .
  FILTER CONTAINS(?country,"South Korea")
}
```

Q19

Show movies created after 2000 but only unique titles



The screenshot shows a SPARQL query interface. The query is as follows:

```
1 PREFIX : <http://dbpedia.org/ontology/>
2 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4 PREFIX movies: <http://dbpedia.org/ontology/>
5 SELECT DISTINCT ?title ?date
6 WHERE {
7   ?title rdf:type movies:Film.
8   ?title :releaseDate ?date.
9   FILTER(year(?date)>2000)
10 }
```

The results are displayed in a table with two columns: **title** and **date**. The table shows 8 rows of data, each representing a unique movie title and its release date.

title	date
<http://dbpedia.org/resource/All_About_Eve_(Philippine_TV_series)>	"2009-03-09"
<http://dbpedia.org/resource/All_About_the_Andersons>	"2003-09-12"
<http://dbpedia.org/resource/All_I_Want_for_Christmas_Is_You_(film)>	"2017-11-13"
<http://dbpedia.org/resource/All_India_Bakchod_Knockout>	"2015-01-28"
<http://dbpedia.org/resource/All_My_Life_(TV_series)>	"2009-06-29"
<http://dbpedia.org/resource/All_Quiet_on_the_Western_Front_(2022_film)>	"2022-10-28"
<http://dbpedia.org/resource/Alliance_(TV_series)>	"2015-02-05"
<http://dbpedia.org/resource/Alles_was_zählt>	"2006-09-04"

PREFIX : <http://dbpedia.org/ontology/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX movies: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?title ?date

WHERE {

?title rdf:type movies:Film.


?title :releaseDate ?date.

FILTER(year(?date)>2000)


}


Q20


Gives all movies which is being created by Steven Spielsberg and Tom Hanks is not an actor.





```
4 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
6 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
7 SELECT DISTINCT ?title
8 WHERE {
9   ?title rdfs:type :Film.
10  ?title :director ?director.
11  ?director rdfs:label ?directorName
12  FILTER CONTAINS(?directorName,"Steven Spielberg")
13  FILTER NOT EXISTS {
14    ?title dbp:starring ?a.
15    FILTER(REGEX(?a, ".Tom Hanks."))
16  }
17 }
```


 Table


 Response


 Gallery


 Chart


 Geo

 Geo-3D

 Geo events

 Markup

 Network

 P

title

- 1 db:resource/Always_(1989_film)
- 2 db:resource/Amblin'
- 3 db:resource/Amistad_(film)
- 4 db:resource/War_Horse_(film)
- 5 db:resource/War_of_the_Worlds_(2005_film)
- 6 db:resource/West_Side_Story_(2021_film)
- 7 db:resource/Empire_of_the_Sun_(film)
- 8 db:resource/The_Adventures_of_Tintin_(film)
- 9 db:resource/The_BFG_(2016_film)

PREFIX dbp: <http://dbpedia.org/property/>

PREFIX db: <http://dbpedia.org/>

PREFIX : <http://dbpedia.org/ontology/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?title

WHERE {

 ?title rdfs:type :Film.

```

?title :director ?director.

?director rdfs:label ?directorName

FILTER CONTAINS(?directorName,"Steven Spielberg")

FILTER NOT EXISTS {

?title dbp:starring ?a.

FILTER(REGEX(?a, ".Tom Hanks."))

}

}

```

Q21

Gives all movies and the capital from New Zealand

```

PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX db: <http://dbpedia.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX movie: <http://dbpedia.org/ontology/>
SELECT ?title ?capital
{
  ?title rdf:type movie:Film.
  ?title dbp:country ?a .
  ?a dbo:capital ?capital .
  ?a rdfs:label ?country .
  FILTER CONTAINS(?country,"New Zealand")
}

```

22 results in 2.17 seconds

Simple view ☐ Ellipse ☒ Filter query results

le	capital
:resource/Waru_(2017_film)	db:resource/Wellington
:resource/Mana_Waka	db:resource/Wellington
:resource/As_Dreams_Are_Made_On	db:resource/Wellington
:resource/Daffodils_(film)	db:resource/Wellington
:resource/Broken_Barrier	db:resource/Wellington
:resource/Reckless_Behavior_Caught_on_Tape	db:resource/Wellington
:resource/The_Fanimatrix	db:resource/Wellington
:resource/The_Grasscutter	db:resource/Wellington
:resource/The_Rainbow_Warrior_(film)	db:resource/Wellington
:resource/Close_Up_(TV_programme)	db:resource/Wellington
:resource/Saturday_Disney_(New_Zealand_TV_series)	db:resource/Wellington

```

PREFIX dbo: <http://dbpedia.org/ontology/>

PREFIX dbp: <http://dbpedia.org/property/>

PREFIX db: <http://dbpedia.org/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX movie: <http://dbpedia.org/ontology/>

SELECT ?title ?capital{

```

```

?title rdf:type movie:Film.

?title dbp:country ?a .

?a dbo:capital ?capital .

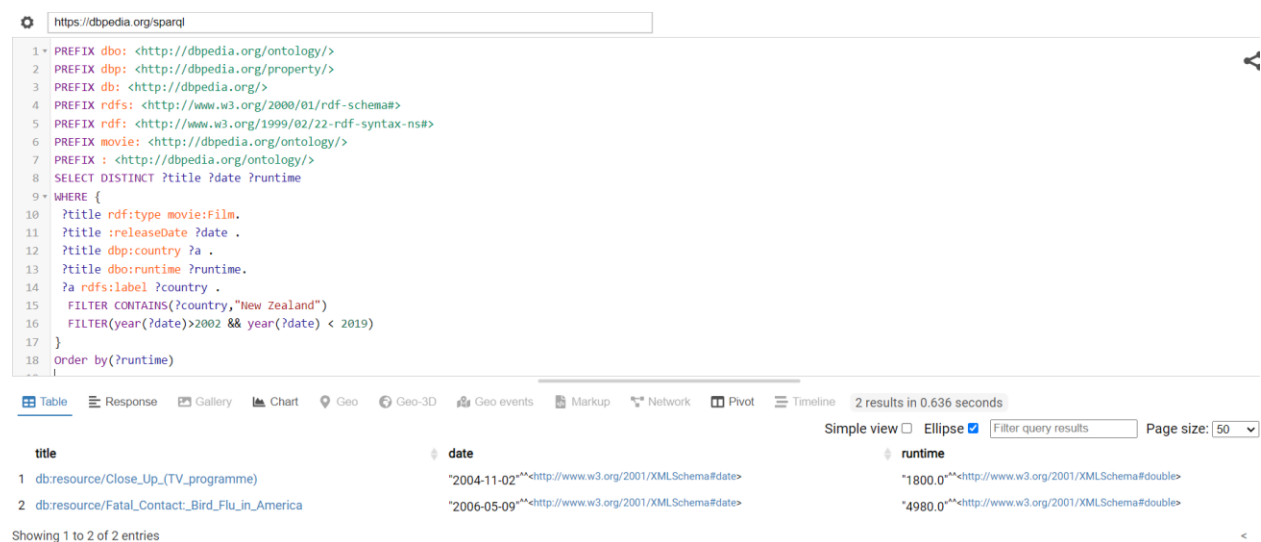
?a rdfs:label ?country .

FILTER CONTAINS(?country,"New Zealand")

```

Q22

Gives all movies, date and runtime which are created in New Zealand and between 2002 and 2019 and ordered by runtime



The screenshot shows a SPARQL query interface with the following query:

```

1 PREFIX dbo: <http://dbpedia.org/ontology/>
2 PREFIX dbp: <http://dbpedia.org/property/>
3 PREFIX db: <http://dbpedia.org/>
4 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
6 PREFIX movie: <http://dbpedia.org/ontology/>
7 PREFIX : <http://dbpedia.org/ontology/>
8 SELECT DISTINCT ?title ?date ?runtime
9 WHERE {
10   ?title rdf:type movie:Film.
11   ?title :releaseDate ?date .
12   ?title dbp:country ?a .
13   ?title dbo:runtime ?runtime.
14   ?a rdfs:label ?country .
15   FILTER CONTAINS(?country,"New Zealand")
16   FILTER(year(?date)>2002 && year(?date) < 2019)
17 }
18 Order by(?runtime)

```

The results table shows 2 results in 0.636 seconds:

title	date	runtime
1 db:resource/Close_Up_(TV_programme)	"2004-11-02" <http://www.w3.org/2001/XMLSchema#date>	"1800.0" <http://www.w3.org/2001/XMLSchema#double>
2 db:resource/Fatal_Contact_Bird_Flu_in_America	"2006-05-09" <http://www.w3.org/2001/XMLSchema#date>	"4980.0" <http://www.w3.org/2001/XMLSchema#double>

Showing 1 to 2 of 2 entries

```

PREFIX dbo: <http://dbpedia.org/ontology/>

PREFIX dbp: <http://dbpedia.org/property/>

PREFIX db: <http://dbpedia.org/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX movie: <http://dbpedia.org/ontology/>

PREFIX : <http://dbpedia.org/ontology/>

SELECT DISTINCT ?title ?date ?runtime

WHERE {

  ?title rdf:type movie:Film.

  ?title :releaseDate ?date .

```

```

?title dbp:country ?a .

?title dbo:runtime ?runtime.

?a rdfs:label ?country .

FILTER CONTAINS(?country,"New Zealand")

FILTER(year(?date)>2002 && year(?date) < 2019)

}

Order by(?runtime)

```

Q23

With Inference

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × Q17 × Q23 × ⓘ

```

1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX ex: <http://semantics.id/ns/example#>
6
7 SELECT ?p
8 WHERE {?p a :Artwork.}
9

```

Press Alt+Enter to autocomplete

Table Raw Response Pivot Table Google Chart Download as

Filter query results Showing results from 1 to 6 of 6. Query took 0.1s, moments ago.

	p
1	ex:film_1
2	ex:film_2
3	ex:film_3
4	Belle
5	ex:film_4
6	ex:film_5

```

PREFIX : <http://semantics.id/ns/example/film#>

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX ex: <http://semantics.id/ns/example#>

SELECT ?p

WHERE {?p a :Artwork.

}

```

Without Inference

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × Q17 × Q23 × ⓘ

```
1 PREFIX : <http://semantics.id/ns/example/film#>
2 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
5 PREFIX ex: <http://semantics.id/ns/example#>
6
7 SELECT ?p
8 WHERE { ?p a :Artwork.
9 }
```

⌨️ 📄 🗂️ 🔗 ⏏️ ⚙️

Press Alt+Enter keyboard shortcuts Run

Table Raw Response Pivot Table Google Chart Download as

Filter query results No results. Query took 0.1s, moments ago.

	p
No data available in table	

Q24

With Inference

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × Q17 × Q23 × ⓘ

```
1 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 select ?p
4 where{
5 ?p a :Performer
6 }
7
```

⌨️ 📄 🗂️ 🔗 ⏏️ ⚙️

Press Alt+Enter to autocomplete Run

Table Raw Response Pivot Table Google Chart Download as

Filter query results Showing results from 1 to 8 of 8. Query took 0.1s, moments ago.

	p
1	ex:alicia_vikander
2	ex:dan_stevens
3	ex:emma_watson
4	ex:ewan_mcgregor
5	ex:hayley_atwell
6	zendaya
7	kyle_maciachlan
8	alan_menken

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

PREFIX : <http://semantics.id/ns/example/film#>

select ?p

where{

?p a :Performer

}

Without inference

Q1 × Q4 × Q6 × Q7 × Q9 × Q11 × Q13 × Q14 × Q16 × Q17 × Q23 ×

```
1 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 select ?p
4 where{
5   ?p a :Performer
6 }
7
```

Run keyboard shortcuts

Table Raw Response Pivot Table Google Chart Download as

Filter query results Showing results from 1 to 1 of 1. Query took 0.1s, moments ago.

	p
1	alan_menken

Q25

With Inference

SmarticSystems en

```
1 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 select ?p
4 where{
5   ?p a foaf:Person
6 }
7
```

Run keyboard shortcuts

Table Raw Response Pivot Table Google Chart Download as

Filter query results Showing results from 1 to 17 of 17. Query took 0.1s, moments ago.

	p
1	ex:alicia_vikander
2	ex:bill_condon
3	ex:dan_stevens
4	ex:emma_watson
5	ex:ewan_mcgregor
6	ex:hayley_atwell
7	ex:marc_forster
8	ex:writer_1
9	ex:writer_2
10	ex:julius_avery
11	ex:writer_3

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

PREFIX : <http://semantics.id/ns/example/film#>

select ?p

where{

?p a foaf:Person

}

Without inference

1 PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2 PREFIX : <http://semantics.id/ns/example/film#>
3 select ?p
4 where {
5 ?p a foaf:Person
6 }
7

Save icon
Copy icon
Link icon
Share icon
Refresh icon
Run button

keyboard shortcuts

TableRaw ResponsePivot TableGoogle Chart

Download as

Filter query results

No results. Query took 0.1s, moments ago.

P
No data available in table