ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ

Αναφορά Project

Ευάγγελος Καψουλάκης 1047062

Ερώτημα 1

Α. Το γνωστικό πεδίο της οντολογίας είναι οι ταινίες και πιο συγκεκριμένα όλη η παραγωγή μιας ταινίας (ηθοποιοί, παραγωγοί, συγγραφείς, μουσική, σενάριο, συνθέτες κτλπ) καθώς και τα βραβεία που έχει λάβει μια ταινία.

Β.

• Η οντολογία θα καλύψει το γνωστικό επίπεδο των ταινιών. Θα περιέχει ταινίες(σενάριο και μουσική), το προσωπικό που δούλεψε, βραβεία κτλπ.

• Σκοπός της οντολογίας είναι η εύρεση ταινιών από αρκετούς παράγοντες.

• Η οντολογία θα παρέχει πληροφορίες για το τι είδος ταινίας είναι μια (φαντασία, δράση κτλπ), σε τι κοινό απευθύνετε με βάση τη γλώσσα (Αγγλικά κτλπ), που είναι γυρισμένη(τοποθεσία) καθώς και πληροφορίες για όλο το προσωπικό των ταινιών (σκηνοθέτες, ηθοποιοί).

• Με τον μηχανισμό του συμπερασμού θα μπορούσαν να βρεθούν ηθοποιοί, σκηνοθέτες κτλπ με κοινές ταινίες ή το αντίθετο (ταινίες με κοινούς ηθοποιούς κτλπ)

Έμπνευση ήταν το imdb καθώς και ένα project που υλοποίησα πιο παλιά στο μάθημα θεωρία αποφάσεων που ήταν ένα σύστημα movie recommendation βάση δημογραφικών στοιχείων.

Εφαρμογή αναπτύχθηκε για το Project που μπορεί να βρίσκει ο χρήστης ταινίες με βάση πιο εξειδικευμένα στοιχεία όπως πχ με βάση των αγαπημένων του ηθοποιών, παραγωγών. Βάση τι γλώσσες ξέρει ή ακόμα και για ποιες ηλικίες απευθύνετε.

Γ.

Movies: η κλάση όλων.

- **FilmFestival:** Η κλάση περιέχει τα φεστιβάλ απονομής βραβείων ταινιών.
- **Languages:** Η κλάση περιέχει τις γλώσσες που περιέχει μια ταινία.
- **MovieAward:** Το βραβείο που απομένετε από ένα φεστιβάλ.
- **MovieCrew:** Η κλάση που περιέχει όλο το προσωπικό μιας ταινίας.
  - **Organization:** Εταιρίες.
    - **Distributor:** Περιέχει τις εταιρίες παραγωγής ταινιών.
  - **Person:** Individuals (ανθρώπινο δυναμικό)
    - **Actor:** Κλάση ηθοποιών.
    - **Composer:** Κλάση συνθετών μουσικής.
    - **Director:** Κλάση σκηνοθετών.
    - **Writer:** Κλάση σεναριογράφων.
- **MovieGenre:** Κλάση είδος ταινιών.
- **MovieStuff:**
  - **Movie:** Περιέχει τις ταινίες.
  - **MovieAssociatedProduct:**
    - **Screenplay:** Κλάση σεναρίου
      - **AdaptedScreenplay:** Αν σενάριο είναι εμπνευσμένο από αλλού.
      - **OriginalScreenplay:** Αν το σενάριο είναι original
    - **Soundtrack:** Κλάση μουσικής
      - **LicensedSoundtrack:** Κλάση μουσικής είναι «αγορασμένη»
      - **OriginalSoundtrack:** Κλάση αν η μουσική είναι original
- **Place:** Κλάση που περιέχει τοποθεσίες γυρισμάτων.
  - **Country:** Κλάση που περιέχει χώρα παραγωγής.

Δ.

Data Properties

birth_date: Ιδιότητα γενέθλιών **για τη κλάση Person** με range **xsd:date**

budget: Ιδιότητα για κόστος παραγωγής **για τη κλάση Movie** με range **xsd:int**

duration: Ιδιότητα για διάρκεια ταινίας(min) στη **κλάση Movie** με range **xsd:int**

name: Ιδιότητα ονόματος **για τη κλάση Person** με range **xsd:string**

plot_summary: Ιδιότητα περιγραφής ταινίας **για τη κλάση Movie** με range **xsd:string**

rating: Ιδιότητα καταλληλόλητας ταινίας βάση ηλικίας **για τη κλάση Movie** με range **xsd:string**

release_date: Ιδιότητα ημερομηνίας κυκλοφορίας σε μορφή (Y-M-D) **για τη κλάση Movie** με range **xsd:date**

title: Ιδιότητα τίτλου ταινίας **για τη κλάση Movie** με range **xsd:string**

Object Properties

associated_with: Ιδιότητα που συνδέει τη κλάση Movie με τις κλάσεις Screenplay και Soundtrack (υποκλάσεις της κλάσης MovieAssociatedProduct).
Domain: *Movie or MovieAssociatedProduct* **Range:** *Movie or MovieAssociatedProduct*
Ειδική ιδιότητα: *Συμμετρική*.

awarded_at: Ιδιότητα που λέει σε ποιο φεστιβάλ απονέμετε το συγκεκριμένο βραβείο.
Domain: *MovieAward* **Range:** *FilmFestival* **Ειδική ιδιότητα:** *Συναρτησιακή*.
has_award: Αντίθετη ιδιότητα της awarded_at (το φεστιβάλ απονέμει συγκεκριμένο βραβείο)
Domain: *FilmFestival* **Range:** *MovieAward* **Ειδική ιδιότητα:** *Αντίθετα Συναρτησιακή*.
awarded_to: Ιδιότητα που λέει σε *ποιόν ή που* απονέμετε ένα βραβείο.
Domain: *MovieAward* **Range:** *MovieStuff or Person.*
winner_of: Αντίθετη ιδιότητα της awarded_to. (κάποιος ή κάτι έχει ένα βραβείο)
Domain: *MovieStuff or Person.* **Range:** *MovieAward* .
composed_by: Ιδιότητα που λέει η μουσική μιας ταινίας συντέθηκε από έναν συνθέτη.
Domain: *Soundtrack* **Range:** *Composer*.
composer_of: Αντίθετη ιδιότητα της composed_by. (αυτός ο συνθέτης έγραψε αυτή τη μουσική.) **Domain:** *Composer*. **Range:** *Soundtrack.*
directed_by: Ιδιότητα που λέει μια ταινία σκηνοθετήθηκε από έναν σκηνοθέτη.
Domain: *Movie*. **Range:** *Director*.
director_of: Αντίθετη ιδιότητα της directed_by. (αυτός ο σκηνοθέτης σκηνοθέτησε αυτή τη ταινία.) **Domain:** *Director*. **Range:** *Movie*.
distributed_by: Ιδιότητα που λέει μια ταινία διανεμήθηκε από έναν οργανισμό ή άνθρωπο.
Domain: *Movie*. **Range:** *Distributor or Person.*
distributer_of: Αντίθετη ιδιότητα της distributed_by. (αυτός ο οργανισμός ή άνθρωπος έχει διανέμει κάποια ταινία) **Domain:** *Distributor or Person.* **Range:** *Movie*.

featured_in: Ιδιότητα που λέει ένας ηθοποιός μπορεί να παίζει σε κάποιες ταινίες.
Domain: *Actor*. **Range:** *Movie*. **Ειδική ιδιότητα:** *Μεταβατική*.

features_actor: Αντίθετη ιδιότητα της featured_in. (σε αυτή τη ταινία πρωταγωνιστούν αυτοί οι ηθοποιοί ή ηθοποιός) **Domain:** *Movie*. **Range:** *Actor*.

has_filming_location: Ιδιότητα που λέει μια ταινία έχει γυριστεί σε αυτό το σημείο
Domain: *Movie*. **Range:** *Place.*
has_production_country: Υπό-ιδιότητα της has_filming_location. που λέει μια ταινία έχει παραχθεί σε μία χώρα **Domain:** *Movie*. **Range:** *Country*.

has_genre: Ιδιότητα που λέει μια ταινία είναι κάποιου ή κάποιων είδος.
Domain: *Movie*. **Range:** *MovieGenre*. **Ειδική ιδιότητα:** *Μεταβατική.*

has_language: Ιδιότητα που λέει μια ταινία είναι κάποιας γλώσσας.
Domain: *Movie*. **Range:** *Languages*. **Ειδική ιδιότητα:** *Συναρτησιακή*.

written_by: Ιδιότητα που λέει το σενάριο μιας ταινίας γράφτηκε από κάποιον συγγραφέα.
Domain: *Screenplay*. **Range:** *Writer*.

writer_of: Αντίθετη ιδιότητα της written_by. (αυτός ο συγγραφέας έγραψε αυτό το σενάριο.) **Domain:** *Writer*. **Range:** *Screenplay*.

Ε.
Στιγμιότυπα

FilmFestival: Academy_Awards, BAFTA_Awards, Berlin_Film_Festival, Cannes_Film_ Festival, Golden_Globes, Venice_Film_Festival

Languages: English, Greek

MovieAward: Oscar, BAFTA, Golden_Bear, Golden_Globe, Golden_Lion, Palme_d'Or

Distributor: FinosFilm, Miramax_Films, Paramount_Pictures, Walt_Disney_Studios, Warner_Bros.

Actor: Actor1, Actor2, Actor3, Actor4, Actor5, Actor6, Actor7

Composer: Composer1, Composer2, Composer3, Composer4

Director: Director1, Director2, Director3, Director4, Director5

Writer: Writer1, Writer2, Writer3, Writer4

MovieGenre: Action, Adventure, Comedy, Drama, Fantasy, Horror, Mystery, Romance, Western

Movie: Movie1, Movie2, Movie3, Movie4, Movie5, Movie6

AdaptedScreenplay: Movie1_script, Movie2_script, Movie6_script

OriginalScreenplay: Movie3_script, Movie4_script

LicensedSoundtrack: Movie1_OST

OriginalSoundtrack: Movie2_OST, Movie4_OST, Movie6_OST

Place: 1o_lykeio_amarousiou, Wild_west_Texas

Country: Greece, US, UK

Ερώτημα 2

Παρέχετε το owl αρχείο μέσα στο συμπιεσμένο.
Παραθέτω 2 owl αρχεία **test.owl** και **movie_onto.owl**. Το **movie_onto.owl** είναι σε μορφή που παίρνει αφού εισάγετε νέα πληροφορία από τη Java και το **test.owl** το export από το Protégé.

Ερώτημα 3

Actor2 featured_in Movie4:
Movie4 **features_actor** Actor2 → **featured_in** InversOf **features_actor.** Αφού ένας ηθοποιός παίζει σε μια ταινία, τότε μια ταινία παρουσιάζει αυτόν τον ηθοποιό.

Actor7 featured_in Movie6:
Movie6 **features_actor** Actor7 → **featured_in** InversOf **features_actor.** Αφού ένας ηθοποιός παίζει σε μια ταινία, τότε μια ταινία παρουσιάζει αυτόν τον ηθοποιό.

Actor2 winner_of BAFTA:
BAFTA **awarded_to** Actor2 → **awarded_to** InversOf **winner_of.** Αφού ένας ηθοποιός κέρδισε ένα βραβείο, τότε αυτό το βραβείο δόθηκε σε αυτόν τον ηθοποιό.

Composer4 composer_of Movie6_OST:
Movie6_OST **composed_by** Composer4 → **composed_by** InversOf **composer_of.** Αφού ένας συνθέτης σύνθεσε μια μουσική, τότε η μουσική θα έχει αυτόν για συνθέτη.

Writer3 writer_of Movie6_script:
Movie6_script **written_by** Writer3 → **writer_of** InversOf **written_by.** Αφού ένας συγγραφέας έγραψε ένα σενάριο για μια ταινία, τότε το σενάριο έχει αυτόν για συγγραφέα.

Director1 director_of Movie1:
Movie1 **directed_by** Director1 → **directed_by** InversOf **director_of.** Αφού ένας σκηνοθέτης σκηνοθέτησε μια ταινία, τότε μία ταινία έχει αυτόν για σκηνοθέτη.

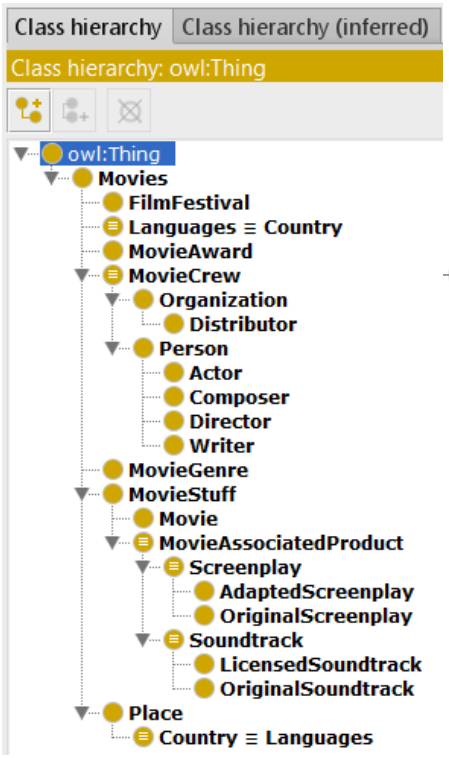
FinosFilm distributer_of Movie5:
Movie5 **distributed_by** FinosFilm → **distributed_by** InversOf **distributer_of.** Αφού ένας διανομέας διανέμει μια ταινία, τότε αυτή η ταινία διανεμήθηκε από αυτόν τον διανομέα.

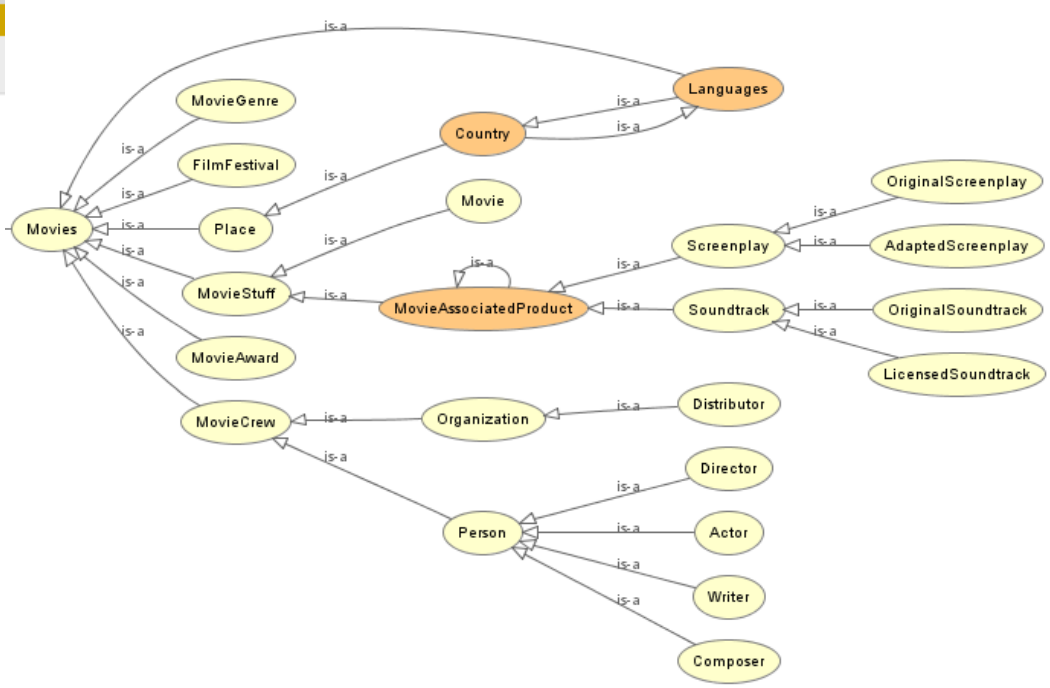
Movie1 associated_with Movie1_OST:
Movie1_OST **associated_with** Movie1 → Symmetric: **associated_with.** Αφού μια ταινία έχει ένα σενάριο, τότε αυτό το σενάριο είναι σενάριο αυτής της ταινίας.

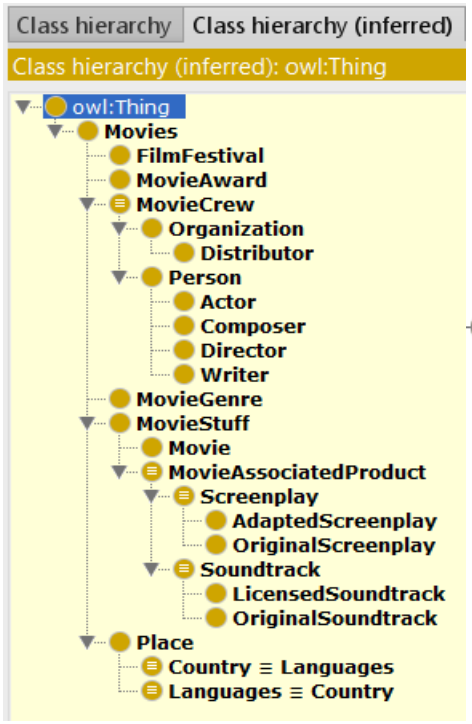
Movie1 associated_with Movie1_script:
Movie1_script **associated_with** Movie1 → Symmetric: **associated_with.** Αφού μια ταινία έχει μια μουσική, τότε αυτή η μουσική παίζεται σε αυτή τη ταινία.

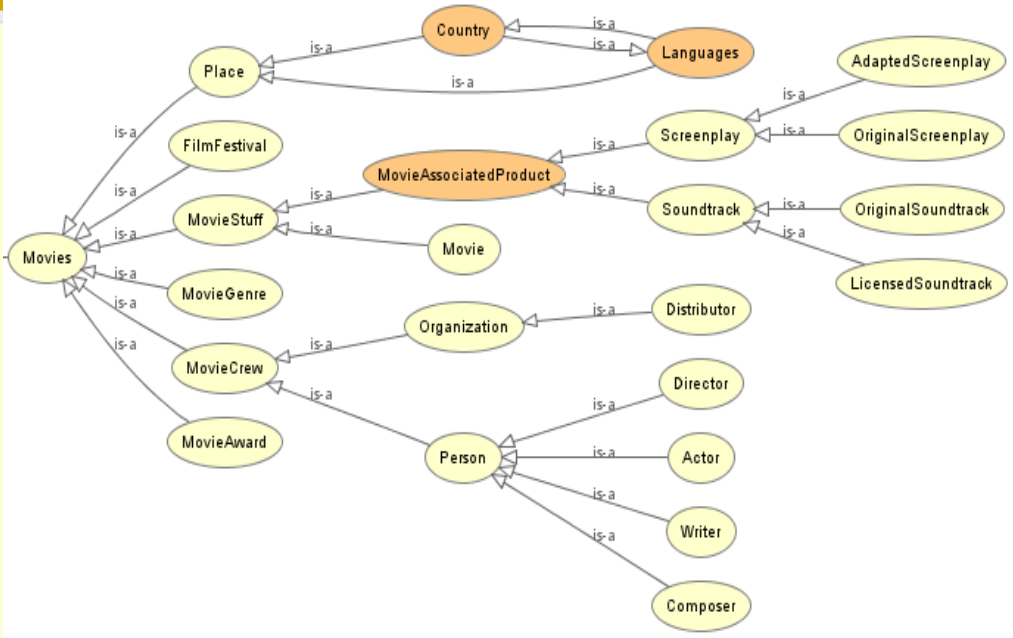
Movie6 has_filming_loction US:
Movie6 **has_production_country** US → **has_production_country** SubPropertyOf **has_filming_loction.** Αφού μια ταινία έχει γυριστεί σε μια τοποθεσία, τότε έχει παραγωγή από αυτή τη χώρα της τοποθεσίας.

Ερώτημα 4







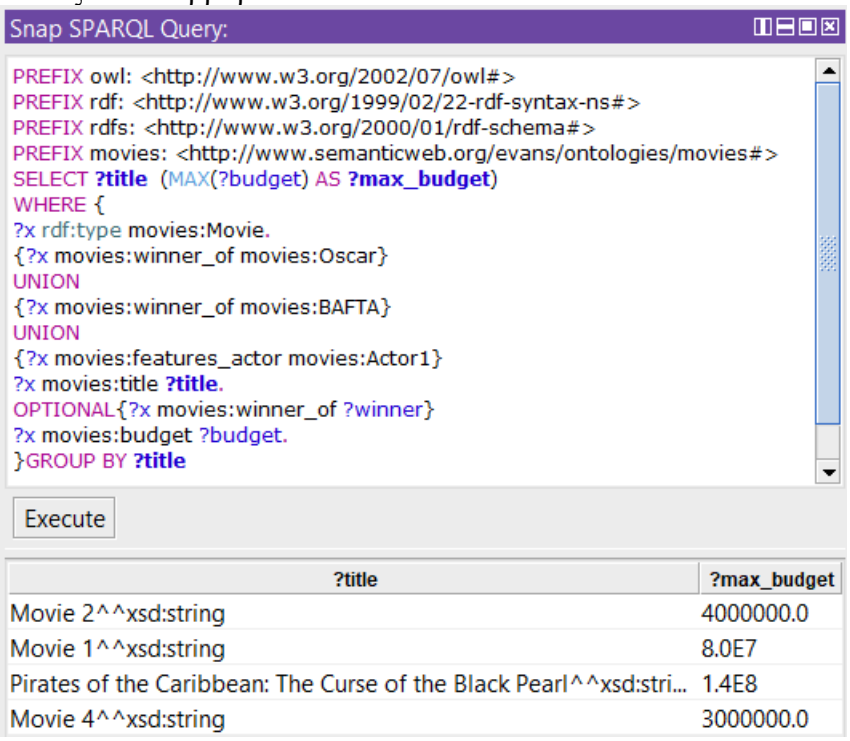


Η διαφορά είναι ότι στο inferred ΄μια γλώσσα σχετίζεται με τη κλάση Place. Το οποίο θεωρητικά είναι σωστό αλλά πρακτικά δεν δουλεύει όπως θα ήθελα στη παρών οντολογία.

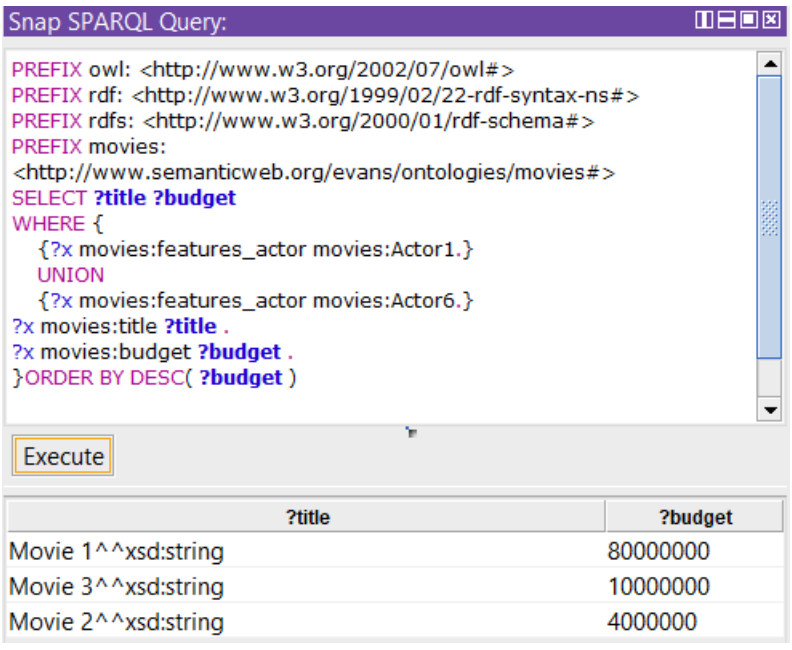
Ερώτημα 5

Επισυνάπτω τα screenshot των SPARQL ερωτημάτων και των αποτελεσμάτων. Έγινε χρήση του Snap SPARQL Query αλλά παρατηρήθηκε ότι τα ερωτήματα τρέχανε μόνο όταν ο Reasoner ήταν ανοιχτός.

1. Εμφάνισε τους τίτλους και το budget των ταινιών που έχουν κερδίσει Oscar ή Bafta βραβεία ή παίζει ο ηθοποιός Actor1. (χρήση MAX, GROUP BY, UNION, OPTIONAL). Παρατηρήθηκε ότι με τη χρήση του MAX οι αριθμοί int του budget αλλάζουν μορφή. Και για κάποιο λόγο ο τύπος των τίτλων ταινιών (xsd:string) εμφανίζεται. Το δοκίμασα και με Literal type αλλά υπήρχε το ίδιο πρόβλημα. Το OPTIONAL χρειάζεται γιατί δεν έχουν όλες οι ταινίες κάποιο βραβείο.



2. Εμφάνισε τους τίτλους και το budget των ταινιών παίζουν οι ηθοποιοί Actor1 και Actor6 κατά φθίνουσα σειρά με βάση τα budget. (Χρήση UNION, ORDER BY DESC). Παρατηρήθηκε ίδιο πρόβλημα με τους τίτλους ταινιών.



3. Εμφάνισε τους τίτλους, τη διάρκεια και τη πλοκή των ταινιών φαντασίας με αύξουσα σειρά κατά τη διάρκεια τους. (Χρήση OPTIONAL, ORDER BY ASC) Το OPTIONAL χρειάζεται γιατί δεν έχουν όλες οι ταινίες περιγραφή. Ξανά το ίδιο πρόβλημα με τους τίτλους και στη πλοκή.

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 movies: <http://www.semanticweb.org/evans/ontologies/movies#>
SELECT ?title ?duration ?plot
WHERE {
?x movies:has_genre movies:Fantasy ;
movies:title ?title;
movies:duration ?duration.
OPTIONAL{?x movies:plot_summary ?plot}
}ORDER BY ASC( ?duration )
```

Execute

?title	?duration	?plot
Movie 4^^xsd:string	90	A very very scary movie with awards^^...
Movie 1^^xsd:string	120	
Pirates of the Caribbea...	143	Blacksmith Will Turner teams up with e...

4. Εμφάνισε το rate των ταινιών που είναι φαντασίας ή παίζει ο Actor1 και έχουν Rating αυστηρά PG 13.(Χρήση SELECT DISTINCT, UNION, FILTER). Το Distinct χρειάζεται γιατί στη ταινία Movie1 είναι και φαντασίας και παίζει ο Actor1 και εμφανίζεται 2 φορές χωρίς αυτό.

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 movies: <http://www.semanticweb.org/evans/ontologies/movies#>
SELECT DISTINCT  ?x ?rated
WHERE {
{?x movies:has_genre movies:Fantasy}
UNION
{?x movies:features_actor movies:Actor1}
?x movies:rating ?rated
FILTER (?rated = "PG 13")
}
```

Execute

?x	?rated
movies:Movie6	PG 13^^xsd:string
movies:Movie2	PG 13^^xsd:string
movies:Movie1	PG 13^^xsd:string

5. Εμφάνισε τους τίτλους και τη πλοκή των ταινιών που δεν είναι αγγλικές.(Χρήση OPTIONAL, FILTER)

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 movies: <http://www.semanticweb.org/evans/ontologies/movies#>
SELECT DISTINCT  ?title ?plot
WHERE {
?x rdf:type movies:Movie;
movies:title ?title;
movies:has_language ?language
OPTIONAL {?x movies:plot_summary ?plot}
FILTER (?language != movies:English)
}
```

Execute

?title	?plot
Movie 5^...	A funny old movie featuring 2 huge greek cinema stars thanas...

SWRL

Το SWRLTab δεν κατάφερα ποτέ να το τρέξω γιατί όταν έβαζα έναν κανόνα μου χάλαγε όλη την οντολογία. Επίσης η αλήθεια είναι ότι με δυσκόλεψε λίγο.

Οπότε παραθέτω κάποιους κανόνες οι οποίοι δεν έχουν ελεγχθεί και δεν ξέρω κατά πόσο είναι σωστοί.

- Has_filming_location(?Movie5,?1o_lykeio_amarousiou)^has_location ,?1o_lykeio_amarousiou ?Greece) → Has_filming_location (?Movie5, ?Greece)

  Αφού μια ταινία έχει γυριστεί στη τοποθεσία 1ο λύκειο Αμαρουσίου και το 1ο λύκειο Αμαρουσίου είναι στην Ελλάδα η ταινία έχει γυριστεί στην Ελλάδα.

- Has_filming_location(?Movie5,?1o_lykeio_amarousiou)^has_production_cou ntry(?Movie5, ?Greece) → has_language(?Movie5, ?Greek)

  Αφού μια ταινία έχει γυριστεί στη τοποθεσία 1ο λύκειο Αμαρουσίου και η τανία έχει γυριστεί στην Ελλάδα τότε περιέχει την ελληνική γλώσσα

- winner_of(?Actor5, ?Oscar)^features_actor(?Movie5, ?Actor5) → Winner_of(?Movie5, ?Oscar)

  Άμα ένας ηθοποιός έχει πάρει Oscar και παίζει σε μια ταινία συγκεκριμένη τότε και η ταινία βραβεύθηκε με Oscar ηθοποιού.

Ερώτημα 6

• **open-world assumption:** Αν επειδή κάτι δεν έχει δηλωθεί ως αληθές, δεν μπορεί να θεωρηθεί ψευδές.

Για παράδειγμα: Το στιγμιότυπο ***Greek*** ανήκει στη κλάση ***Languages***. Ανήκει και στη κλάση ***Country***; Σύμφωνα με το **open-world assumption** δεν γνωρίζουμε αν ανήκει ή όχι, οπότε μπορεί να η πρόταση να είναι αληθής

• **non-unique-name assumption:** Αντικείμενα με διαφορετικά ονόματα μπορεί να αναφέρονται στην ίδια έννοια.

Για παράδειγμα: Το στιγμιότυπο ***Greek*** και ***Greece*** θεωρητικά σημαίνουν το ίδιο πράγμα. (Αφού μια ταινία έχει γυριστεί στην Ελλάδα περιέχει και την ελληνική γλώσσα)

Ερώτημα 7

Video παρουσίασης λειτουργείας:

https://www.youtube.com/watch?v=KuOnzobrXpc&ab_channel=Asteryx21

Παρατηρήσεις / Προβλήματα

- Δεν κατάφερα να φορτώσω στο NetBeans την οντολογία ως OWL/XML Syntax export από το Protégé όπως αναφέρει η εκφώνηση. Αντ' αυτού την έκανα export ως RDF/XML Syntax και δούλευε κανονικά.
- Όταν κάνω μια νέα εισαγωγή στην οντολογία μέσω Java αλλάζει η μορφή του αρχείου owl. Παραθέτω screenshots από πριν (1ο screenshot) και μετά (2ο screenshot) την εισαγωγή.

```
<!-- http://www.semanticweb.org/evans/ontologies/movies#Movie1 -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/evans/ontologies/movies#Movie1">
    <rdf:type rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Movie"/>
    <movies:directed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Director1"/>
    <movies:distributed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Paramount_Pictures"/>
    <movies:features_actor rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Actor1"/>
    <movies:features_actor rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Actor2"/>
    <movies:has_genre rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Action"/>
    <movies:has_genre rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Fantasy"/>
    <movies:has_language rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#English"/>
    <movies:produced_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Producer1"/>
    <movies:winner_of rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Oscar"/>
    <movies:budget rdf:datatype="http://www.w3.org/2001/XMLSchema#int">80000000</movies:budget>
    <movies:duration rdf:datatype="http://www.w3.org/2001/XMLSchema#int">120</movies:duration>
    <movies:rating rdf:datatype="http://www.w3.org/2001/XMLSchema#string">PG 13</movies:rating>
    <movies:release_date rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2002-01-04</movies:release_date>
    <movies:title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Movie 1</movies:title>
</owl:NamedIndividual>
```

```
<rdf:Description rdf:about="http://www.semanticweb.org/evans/ontologies/movies#Movie1">
  <movies:budget rdf:datatype="http://www.w3.org/2001/XMLSchema#int">80000000</movies:budget>
  <movies:rating rdf:datatype="http://www.w3.org/2001/XMLSchema#string">PG 13</movies:rating>
  <movies:release_date rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2002-01-04</movies:release_date>
  <movies:title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Movie 1</movies:title>
  <rdf:type rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Movie"/>
  <movies:directed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Director1"/>
  <movies:duration rdf:datatype="http://www.w3.org/2001/XMLSchema#int">120</movies:duration>
  <movies:features_actor rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Actor2"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
  <movies:has_genre rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Fantasy"/>
  <movies:has_language rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#English"/>
  <movies:features_actor rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Actor1"/>
  <movies:winner_of rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Oscar"/>
  <movies:distributed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Paramount_Pictures"/>
  <movies:has_genre rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Action"/>
</rdf:Description>
```

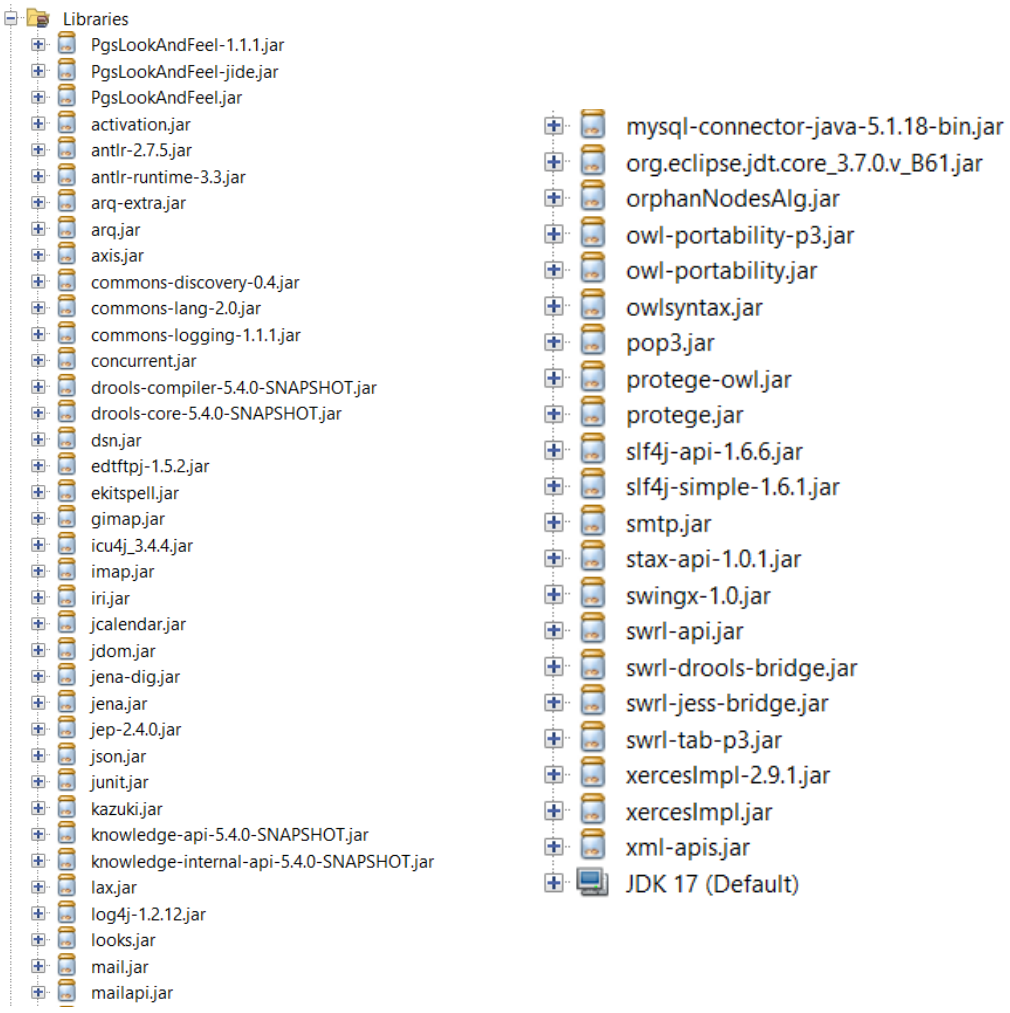
- Όταν φορτώνεται η οντολογία στο NetBeans ο πίνακας που δείχνει πληροφορίες για μια ταινία άμα έχει 2 είδη ταινίας εμφανίζεται 2 φορές όπως στο screenshot παρακάτω.

Pirates of the Caribbean: Th...	PG 13	2003-07-09	Action
Pirates of the Caribbean: Th...	PG 13	2003-07-09	Fantasy
Pirates of the Caribbean: Th...	PG 13	2003-07-09	Adventure

Σχολιασμός

Βιβλιοθήκες:

Ίδιες με την 2η Άσκηση. Βρισκόντουσαν στο αρχείο του JENA APACHE. Παρουσιάζεται ένα πρόβλημα που και που. Δεν ξέρω αν φταίνε οι βιβλιοθήκες ή το NetBeans, αλλά όταν δεν έχω κάνει edit ένα .java κώδικα αρκετή ώρα και πάω να τρέξω το application, το java file που δεν έκανα edit δεν τρέχει. Αλλά άμα πχ σβήσω μια γραμμή ή ένα γράμμα και το ξανατρέξω είναι μια χαρά.



Για τη φόρτωση της οντολογίας και του μοντέλου χρησιμοποιώ αυτά που βρέθηκαν στο βοηθητικό site στην εκφώνηση. Εφαρμόζεται μοντέλο συμπερασμού με το κανόνα "**OWL MEN MICRO RULE INF**".

```
// model load and validation check
Model data = FileManager.get().loadModel("movies_onto.owl");
OntModel infmodel = ModelFactory.createOntologyModel( OWL_MEM_MICRO_RULE_INF, data );
ValidityReport validity = infmodel.validate();
if (validity.isValid()){
    System.out.print("OK");
} else {
    System.out.print("Conflicts");
    for (Iterator i = validity.getReports(); i.hasNext();){
        System.out.print("-" + i.next());
    }
}
```

Όλα τα drop-down menus είναι δυναμικά γεμισμένα. Δηλαδή προσκομίζουν τη πληροφορία τους από την οντολογία δεν είναι στατικά. Για κάθε drop-down menu(Combo box) χρησιμοποιήθηκε η ίδια λογική. SPARQL ερωτήματα.

Όπως αναφαίρετε και στο Ερώτημα 5 για κάποιο λόγο τα SPARQL ερωτήματα που γύρναγαν πληροφορία μαζί με τον τύπο. Δηλαδή πχ εδώ που παίρνει τα Actor names μου γύρναγε Leonardo DiCaprio^^http://www.w3.org/2001/XMLSchema#string.
Για αυτό το λόγο τα αφαιρώ από το string με την εντολή replace.

```
/////////////// fill drop down menu with actor names//////////////////////////
    String actor_names;
    actor_names = "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/2000/01/22-rdf-schema#>"
            + "PREFIX movies: <http://www.semanticweb.org/evans/ontologies/movies#>"
            + "SELECT ?name "
            + "WHERE {"
            + "?x rdf:type movies:Actor ."
            + "?x movies:name ?name ."
            + "}";
    Query query_actor_names = QueryFactory.create(actor_names) ;
    query_actor_names.serialize(new IndentedWriter(System.out,false)) ;
    QueryExecution qexec = QueryExecutionFactory.create(query_actor_names, infmodel) ;
    ResultSet rs_actor_names = qexec.execSelect() ;
    for ( ; rs_actor_names.hasNext() ; ) {
        QuerySolution sol = rs_actor_names.nextSolution();
        RDFNode name = sol.get("name") ;
        jComboBox1.addItem(name.toString().replace("^^http://www.w3.org/2001/XMLSchema#string",""));
    }
```

Για το γέμισμα του πίνακα και το "φιλτράρισμα" του ανάλογα με ηθοποιούς / τύπου ταινίας(1η οθόνη στο video) είναι λίγο διαφορετικό από το αντίστοιχο της 2ης άσκησης.
Αρχικά βάζω 2 κενές τιμές στα 2 Combo Box που συμβολίζουν το καμία επιλογή οπότε όταν και τα 2 είναι κενά εμφανίζει όλες της ταινίες της οντολογίας.

Για το φιλτράρισμα **by genre** είναι όπως την άσκηση 2 χρησιμοποιώντας την κλασική συνάρτηση φιλτραρίσματος για τα **jTables**. Που εκμεταλλεύεται ουσιαστικά ότι στο drop down μενού έχει ίδιες «τιμές» με κάποια στήλη του πίνακα. Τα **genre** δηλαδή. Ενώ πχ στο by **Actor** επειδή ο πίνακας δεν έχει πουθενά τους ηθοποιούς δεν λειτουργεί.

```
private void filter (String query){
    ////FILTER movie_table WITH PARAMETER DROPDOWN MENU SELECTED VALUE (GENRE)
    DefaultTableModel tblMode_filter_movie = (DefaultTableModel)movie_table.getModel();
    TableRowSorter<DefaultTableModel> tr=new TableRowSorter<DefaultTableModel>(tblMode_filter_movie);
    movie_table.setRowSorter(tr);
    tr.setRowFilter(RowFilter.regexFilter(query));
}
```

Για το Search by actor είναι διαφορετικό. Το φιλτράρισμα του πίνακα γίνεται μέσω SPARQL ερωτήματος ουσιαστικά. Η συνάρτηση filter_actor η οποία ουσιαστικά εκτελεί ένα “δυναμικό” SPARQL ερώτημα με βάση την επιλογή από το drop-down menu. Δηλαδή όπως φαίνεται στη τελευταία γραμμή του ερωτήματος SPARQL με τη χρήση **FILTER** όπου υπάρχει ένας placeholder μεταβλητή *‘%s’* όπου παίρνει την τιμή της μεταβλητής query κάθε φορά που καλείτε η συνάρτηση **filter_actor**. Η μεταβλητή query περιέχει το επιλεγμένο όνομα του ηθοποιού από το drop-down menu και έτσι μέσω του SPARQL ερωτήματος παίρνουμε τις ταινίες που παίζει ο συγκεκριμένος ηθοποιός.

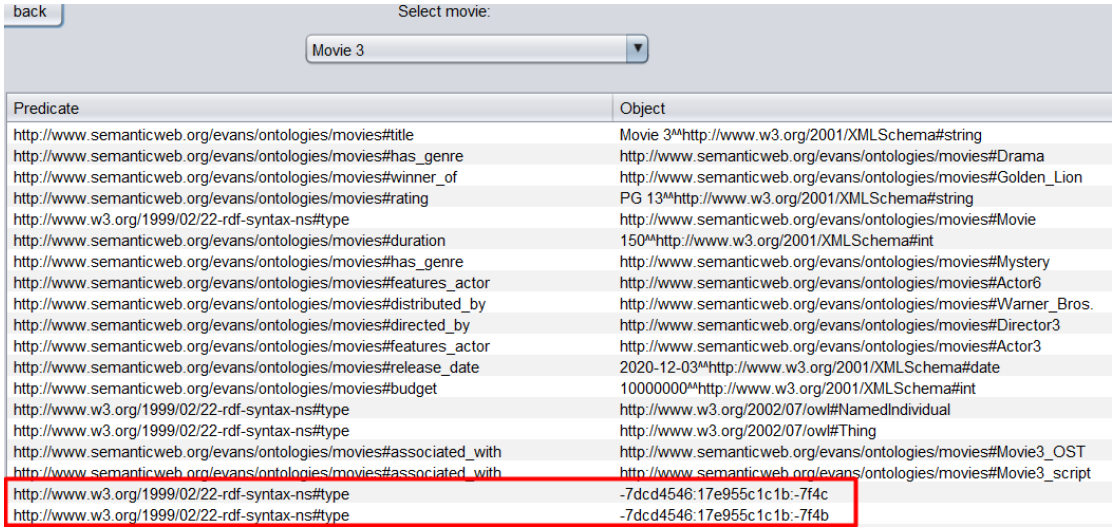
Εδώ θα μπορούσε να φαίνεται και η λειτουργεία **Reasoning** στην οντολογία μου. Με τη χρήση των ιδιοτήτων:
(Ταινία)**features_actor**(Ηθοποιός)
(Ηθοποιός)**featured_in**(Ταινία).
Δηλαδή αντί στο sparql ερώτημα που έχω να έβρισκα της ταινίες με το **features_actor** να πήγαινα πρώτα στους ηθοποιούς και να χρησιμοποιούσα το **featured_in** εκμεταλλεύοντας το γεγονός ότι «όταν μια ταινία έχει έναν ηθοποιό, τότε ο ηθοποιός παίζει στη ταινία»

Αλλά το συνειδητοποίησα στο τέλος που έγραφα την αναφορά και δεν ήθελα ο κώδικας που θα έστελνα να είναι διαφορετικός από το κώδικα στην αναφορά.

```
private void filter_actor (String query){
  ////FILTER movie_table WITH PARAMETER DROPDOWN MENU SELECTED VALUE (ACTORS)
  Model data = FileManager.get().loadModel("movies_onto.owl");
  InfModel infmodel = ModelFactory.createRDFSModel(data);
  DefaultTableModel aModel = (DefaultTableModel) movie_table.getModel();
  aModel.setRowCount(0); // first clear existing table contents
  String filter_by_actor = String.format("PREFIX owl: <http://www.w3.org/2002/07/owl#> "
          + "\nPREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> "
          + "\nPREFIX rdfs: <http://www.w3.org/2000/01/22-rdf-schema#> "
          + "\nPREFIX movies: <http://www.semanticweb.org/evans/ontologies/movies#> "
          + "\nSELECT ?title ?rating ?date ?genre "
          + "\nWHERE { "
          + "\n?x rdf:type movies:Movie ; "
          + "\nmovies:title ?title ;"
          + "\nmovies:rating ?rating ;"
          + "\nmovies:release_date ?date ;"
          + "\nmovies:has_genre ?genre ;"
          + "\nmovies:features_actor ?y ."
          +"\n?y rdf:type movies:Actor ;"
          +"\nmovies:name ?name ."
          + "\nFILTER(?name = '%s') \n}" , query);
  Query movie_details_query = QueryFactory.create(filter_by_actor) ;
  movie_details_query.serialize(new IndentedWriter(System.out,true)) ;
  QueryExecution qexec1 = QueryExecutionFactory.create(movie_details_query, infmodel) ;
  ResultSet movie_details_rs= qexec1.execSelect() ;
  for ( ; movie_details_rs.hasNext() ; ) {
      QuerySolution sol1 = movie_details_rs.nextSolution();
      RDFNode mov_title = sol1.get("title") ;
      RDFNode mov_rate = sol1.get("rating") ;
      RDFNode mov_release = sol1.get("date");
      RDFNode mov_genre = sol1.get("genre");
      aModel.addRow(new Object[]{ mov_title.toString().replace("^^http://www.w3.org/2001/XMLSchema#string",""),mov_rate.toString().repl
      movie_table.setModel(aModel);
  }
}
```

Τα δύο φιλτραρίσματα (by actor, by genre) λειτουργούν και συνδυαστικά όπως φαίνεται και στο βίντεο.

Για το κουμπί More_details ήθελα να επιλέγει ο χρήστης μια ταινία και να του εμφανίζει ότι πληροφορία έχει η οντολογία για αυτή. (Όπως φαίνεται και στο βίντεο) αλλά μου εμφάνιζε και κάτι περίεργα πράγματα: όπως φαίνεται στο κόκκινο.
Θεωρώ έχει να κάνει επειδή αλλάζει μορφή το Owl αρχείο μου με την εισαγωγή νέας πληροφορίας.



Predicate	Object
http://www.semanticweb.org/evans/ontologies/movies#title	Movie 3^^http://www.w3.org/2001/XMLSchema#string
http://www.semanticweb.org/evans/ontologies/movies#has_genre	http://www.semanticweb.org/evans/ontologies/movies#Drama
http://www.semanticweb.org/evans/ontologies/movies#winner_of	http://www.semanticweb.org/evans/ontologies/movies#Golden_Lion
http://www.semanticweb.org/evans/ontologies/movies#rating	PG 13^^http://www.w3.org/2001/XMLSchema#string
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.semanticweb.org/evans/ontologies/movies#Movie
http://www.semanticweb.org/evans/ontologies/movies#duration	150^^http://www.w3.org/2001/XMLSchema#int
http://www.semanticweb.org/evans/ontologies/movies#has_genre	http://www.semanticweb.org/evans/ontologies/movies#Mystery
http://www.semanticweb.org/evans/ontologies/movies#features_actor	http://www.semanticweb.org/evans/ontologies/movies#Actor6
http://www.semanticweb.org/evans/ontologies/movies#distributed_by	http://www.semanticweb.org/evans/ontologies/movies#Warner_Bros.
http://www.semanticweb.org/evans/ontologies/movies#directed_by	http://www.semanticweb.org/evans/ontologies/movies#Director3
http://www.semanticweb.org/evans/ontologies/movies#features_actor	http://www.semanticweb.org/evans/ontologies/movies#Actor3
http://www.semanticweb.org/evans/ontologies/movies#release_date	2020-12-03^^http://www.w3.org/2001/XMLSchema#date
http://www.semanticweb.org/evans/ontologies/movies#budget	10000000^^http://www.w3.org/2001/XMLSchema#int
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#NamedIndividual
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.w3.org/2002/07/owl#Thing
http://www.semanticweb.org/evans/ontologies/movies#associated_with	http://www.semanticweb.org/evans/ontologies/movies#Movie3_OST
http://www.semanticweb.org/evans/ontologies/movies#associated_with	http://www.semanticweb.org/evans/ontologies/movies#Movie3_script
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	-7dcd4546:17e955c1c1b:-7f4c
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	-7dcd4546:17e955c1c1b:-7f4b

Για το add new (movies/actor/writer etc) και edit a movie χρησιμοποιήθηκε η ίδια λογική.

```
Individual i = infmodel.createIndividual("http://www.semanticweb.org/evans/ontologies/movies#"+actor_name.replaceAll(" ", "_"), infmodel.createResource("http://www.semanticweb.org/evans/ontologies/movies#Actor"));
i.addRDFType(OWL2.NamedIndividual);
i.addProperty(infmodel.getProperty("http://www.semanticweb.org/evans/ontologies/movies#name"), infmodel.createTypedLiteral(actor_name, XSDDatatype.XSDstring));
i.addProperty(infmodel.getProperty("http://www.semanticweb.org/evans/ontologies/movies#birth_date"), infmodel.createTypedLiteral(actor_birth, XSDDatatype.XSDdateTime));
infmodel.write( new FileOutputStream("movies_onto.owl") ,"");
```

Χρήση των εντολών για δημιουργία ενός individual της JENA.
Παίρνουμε το user_input βάζουμε «_» αντί για καινά για το τίτλο **του Individual** και λέμε ότι είναι τύπος μίας κλάσης που λέγεται Actor. Προσθέτουμε ότι είναι τύπος individual και τέλος προσθέτουμε τα properties ανάλογα το Insert που κάνουμε και τι είδη properties έχει. Τέλος γράφουμε στο αρχείο Owl.
Όταν επεξεργάζομαι ένα είδη υπάρχων Individual αντί για createIndividual χρησιμοποιώ getIndividual και προσθέτω τα ανάλογα Properties.

Εδώ υπάρχει ένα πρόβλημα. Για κάποιο λόγο όταν κάνω edit μια ταινία παίρνει ότι είναι και τύπος ObjectProperty

```
<rdf:Description rdf:about="http://www.semanticweb.org/evans/ontologies/movies#Horror_House">
  <movies:title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Horror House</movies:title>
  <movies:release_date rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2022-12-12</movies:release_date>
  <movies:has_language rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Chinese"/>
  <movies:rating rdf:datatype="http://www.w3.org/2001/XMLSchema#string">R</movies:rating>
  <movies:distributed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Paramount_Pictures"/>
  <rdf:type rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Movie"/>
  <movies:budget rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1500000</movies:budget>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#NamedIndividual"/>
  <movies:directed_by rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Test_Director"/>
  <movies:duration rdf:datatype="http://www.w3.org/2001/XMLSchema#int">65</movies:duration>
  <movies:features_actor rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Test_Actor"/>
  <movies:has_genre rdf:resource="http://www.semanticweb.org/evans/ontologies/movies#Horror"/>
</rdf:Description>
```