

ProgettoModSem2023



Questa repo contiene l'ontologia OntoGames relativa al mondo dei videogiochi.

Fatta in collaborazione da **Andrea Cacioli** e **Samuele Cavagnino**.

Qui un link alla documentazione generata automaticamente con LODE: un tool che descrive le ontologie.

Files

In questa repo si trovano:

- Tre Ontologie (Una RDF inferita, una TTL Aserita, una RDF Aserita)
- Una applicazione web
- File di documentazione

Motivazioni

Come indicato in precedenza questa ontologia è stata pensata per il mondo dei videogiochi e serve a rappresentare sia il concetto di videogioco come prodotto vendibile, sia come ambientazione di eventi immaginari.

Tale ontologia renderà possibile accedere ai dati in maniera semplice per via della sua natura a grafo, per cui informazioni collegate fra loro saranno di facile accesso.

Sitografia

Per popolare l'ontologia con una A-Box reale sono state usate diverse fonti:

- Wikidata
 - Fandom Wiki
 - Steam DB
 - rlstats.net
 - Sito ufficiale Assetto Corsa Competizione
-

Requisiti

La finalità generale dell'ontologia è quella di accesso ai dati da parte di utenti che non necessariamente devono conoscere il dominio. Un utente inesperto, infatti, potrebbe voler trovare delle informazioni su un gioco in vista di un possibile acquisto e per farlo vorrebbe poter cercare tra i generi e tra i personaggi. Inoltre potrebbe essere interessato a giochi di un certo studio a cui potrebbe essere affezionato. Oppure potrebbe anche voler sapere quali sono i giochi in una certa saga che ha iniziato e non sa come proseguire.

Descrizione Dominio

Il dominio videoludico è un dominio vastissimo, per cui permette di avere numerose interpretazioni. Infatti la nostra scelta non è affatto l'unica plausibile, tuttavia ci è sembrata una scelta ragionevole in quanto modella facilmente la doppia natura del videogame:

- Il videogame come prodotto vendibile
- Il videogame come ambientazione di storie

Esistono inoltre giochi che non hanno la mira di raccontare una storia, invece sono pensati per mettere alla prova il talento dei giocatori e per permettere ad essi di sfidarsi in competizioni.

Inoltre la “fanbase” di un videogame potrebbe interessarsi ai così detti spin-off: una apparizione di un personaggio di un videogame anche in altri videogiochi. Ciò è importante che sia rappresentato nella nostra ontologia.

Abbiamo trovato una sola altra ontologia riguardante il dominio dei videogiochi: videogame ontology. Sebbene sia un'ottima risorsa, tale ontologia non si pone lo stesso nostro obiettivo, invece rappresenta principalmente il concetto di evento di gioco: ovvero l'azione di un giocatore che gli fa ottenere un risultato, ad esempio achievements o acquisti di gioco.

Competency Questions

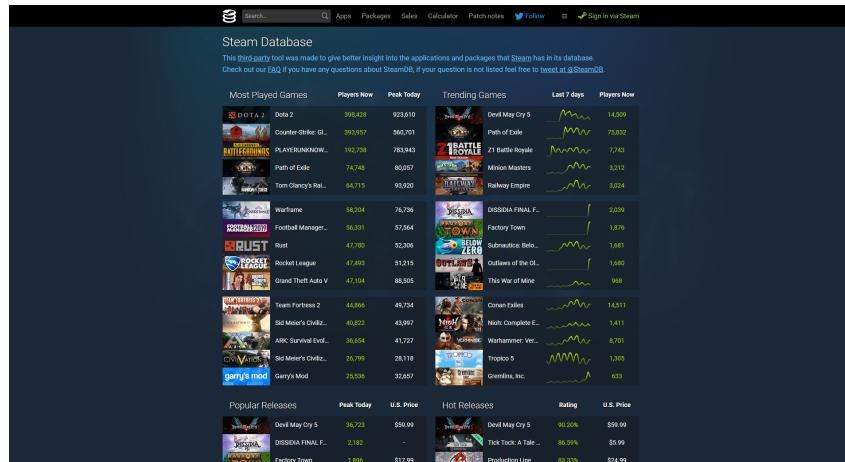
L'ontologia può “rispondere” a numerose domande, qui ne riportiamo alcune:

- Qual é il titolo dei videogiochi che appartengono ad una saga?
 - Di che saga fa parte un videogioco?
 - Qual é lo studio di un videogioco?
 - Quanti sono i videogiochi di una particolare saga?
 - Qual é la data di fondazione di uno studio?
 - Qual é il protagonista di un certo videogioco?
 - Qual é l'ordine cronologico dei videogiochi di una saga?
 - Tutti i titoli con protagonista femminile.
 - Tutti i titoli con storie d'amore contraccambiate.
-

Documentazione

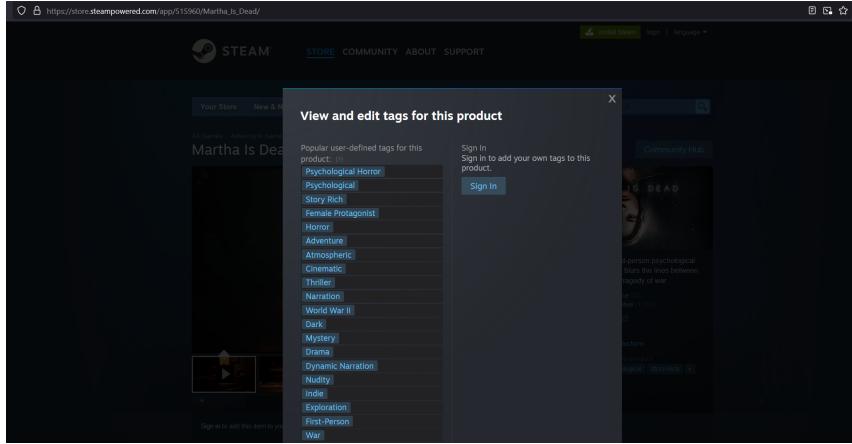
Per questo progetto ci siamo posti il problema di rappresentare il videogioco partendo dalle varie fonti giá presenti sulla rete. In particolare abbiamo utilizzato:

- **SteamDB** Un database in cui é possibile trovare informazioni sul gioco visto come prodotto: le copie vendute, gli utenti online che stanno giocando ad un qualche gioco, il costo di un gioco, il suo indice di gradimento, le piattaforme supportate, i developer e molto altro ancora. Qui sotto si possono vedere delle immagini della pagina principale di questo sito e della pagina del videogioco “Cyberpunk 2077”.

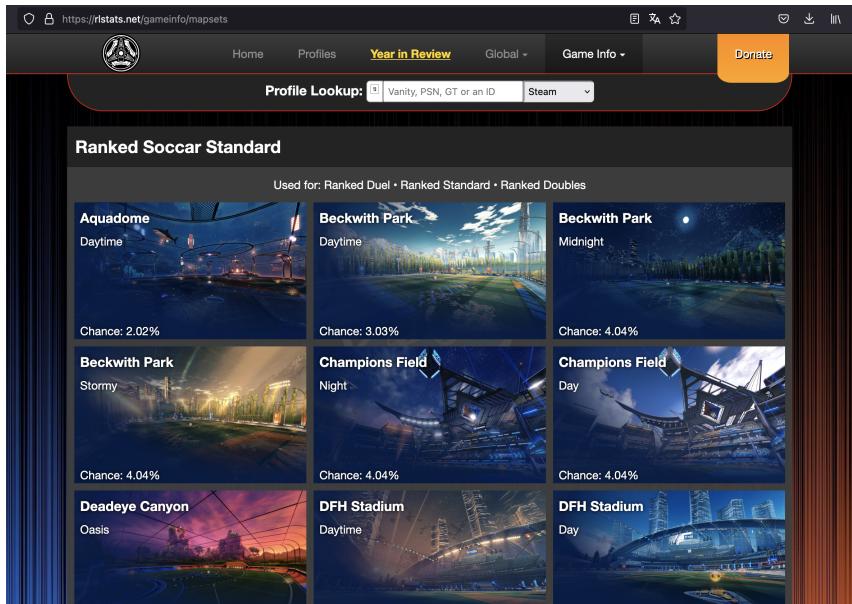


- **Wiki Fandom** Wiki Fandom è un sito che permette a tutti i fan di qualcosa di creare le proprie wiki in cui raccogliere le informazioni relative alla loro passione. In particolare sono presenti numerose wiki su videogiochi, film e libri. Tale sito è una risorsa importante per tutti i giochi con una storia visti come ambientazione di eventi e, appunto, come storie. Qui di seguito abbiamo riportato la pagina del videogioco “Martha is Dead” presa da Wiki Fandom.

- **Steam** Steam è il principale marketplace per la vendita di videogiochi su PC ed in quanto tale ha delle pagine per ogni videogioco. In tali pagine si possono vedere le categorie a cui il videogioco appartiene. Durante la modellazione delle classi e per alcune regole SWRL, questo strumento ci ha permesso di trovare ulteriori categorizzazioni che si possono dare ai videogiochi. Ad esempio RPG o Female Protagonist. Qui di seguito una foto della pagina del videogioco “Martha is Dead” nella sezione categorie.



- **rlStats** rlStats è un sito web in cui si possono trovare informazioni interessanti sul gioco “Rocket League” relative alle mappe ed agli asset di gioco. Lo abbiamo utilizzato per trovare l’elenco delle vetture disponibili sul gioco e per l’elenco di mappe disponibili sul gioco.



- **Sito ufficiale di Assetto Corsa Competizione**

<https://assettocorsa.gg/assetto-corsa-competizione/>

Abbiamo utilizzato il sito ufficiale di Assetto Corsa Competizione per trovare informazioni relative a tale gioco come i nomi delle vetture ed delle mappe di gioco.

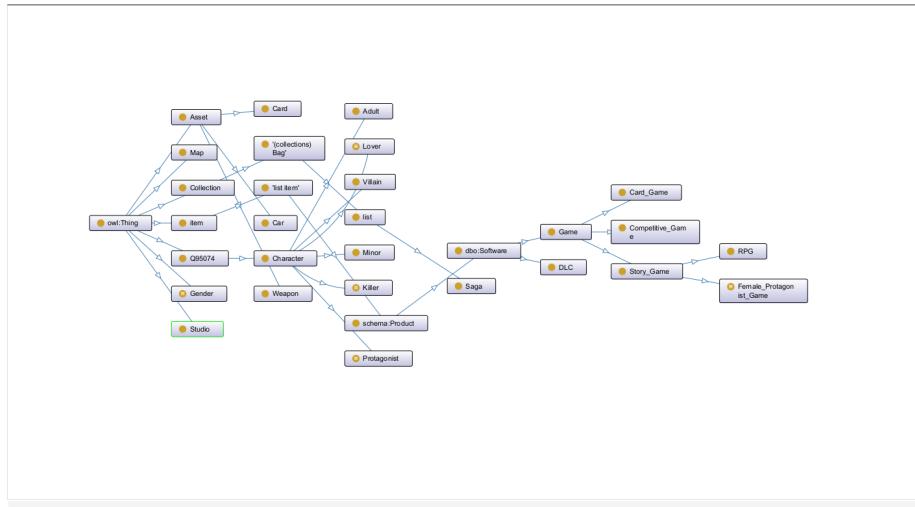
Per essere in linea con le ontologie già esistenti, abbiamo utilizzato:

- dbpedia
 - Concetto di Software
- wikidata
 - Concetto dei personaggi fintizi (immaginari)
- schema
 - Concetto di prodotto come di qualcosa che può essere venduto

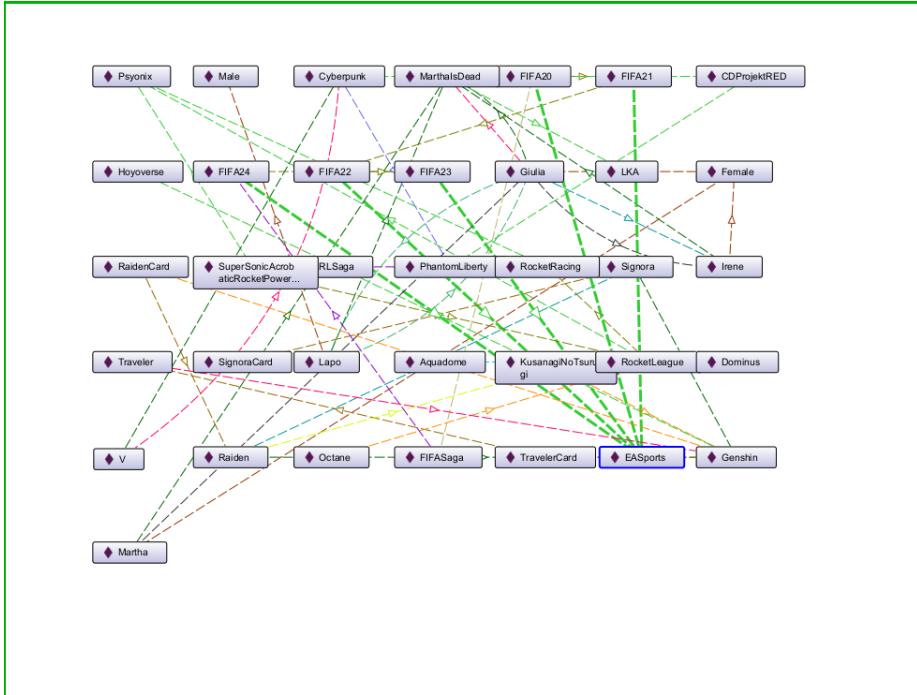
Inoltre abbiamo utilizzato l'**Ontology Design Pattern** lista per modellare i giochi di una saga.

Visualizzazione

Tassonomia:



A-Box:



Esempio Triple da sezione Descrizione Dominio.

Esempio videogioco come prodotto vendibile

soggetto	predicato	oggetto
Rocket League	madeBy	Psyponix
Rocket Racing	madeBy	Psyponix
Octane	assetOf	Rocket League
Aquadome	mapOf	Rocket League

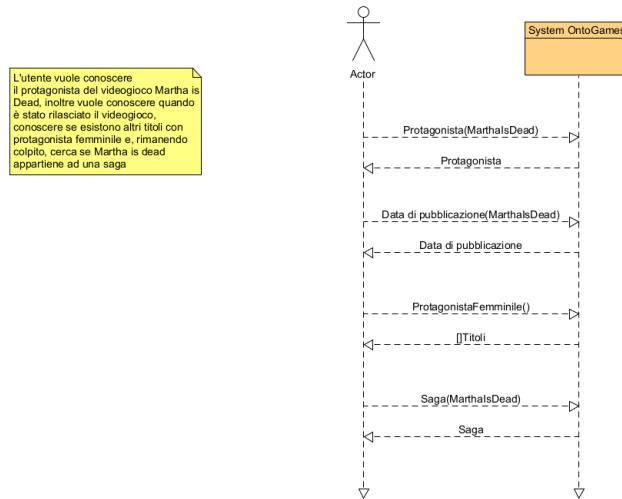
Esempio videogioco come ambientazione di una storia

soggetto	predicato	oggetto
Martha	appearsIn	Martha Is Dead
Giulia	appearsIn	Martha Is Dead
Giulia	protagonistOf	Martha Is Dead
Lapo	loves	Giulia
Giulia	loves	Lapo

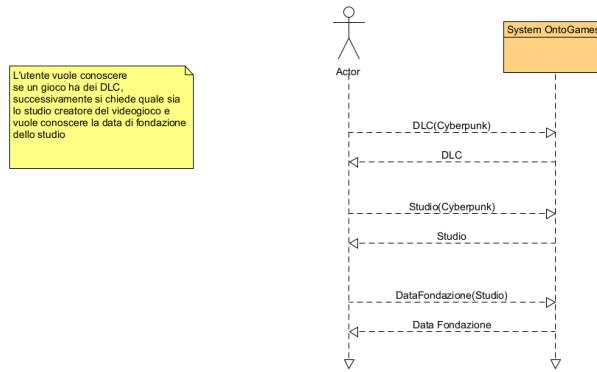
Interazione Utente

Esempi Interazione dell'utente col sistema

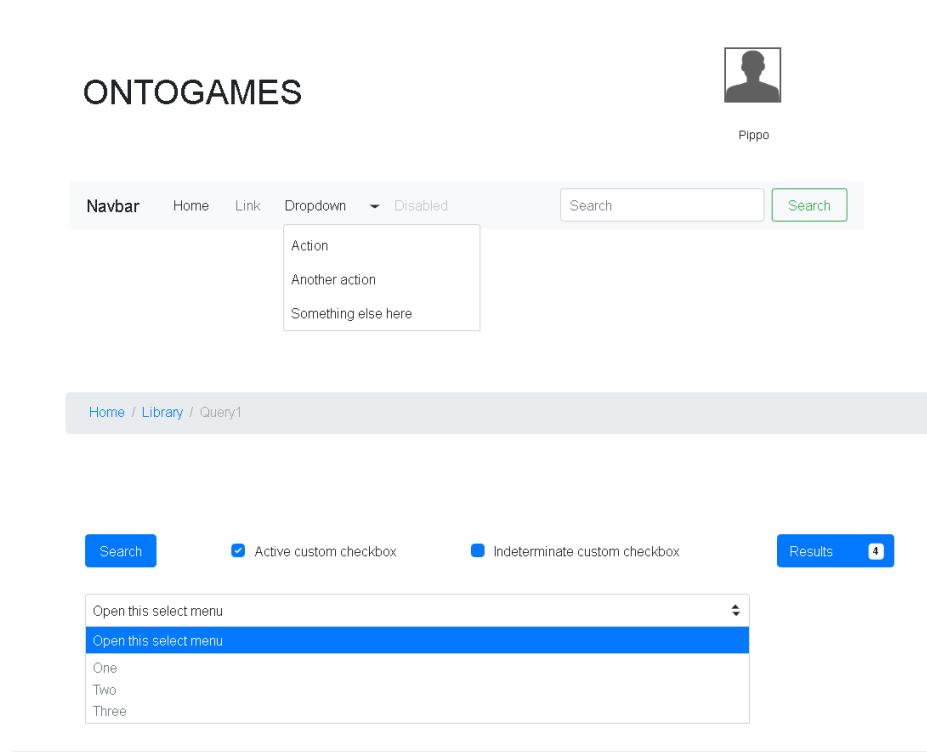
Esempio 1:



Esempio 2:



Mock Up Interfaccia Client

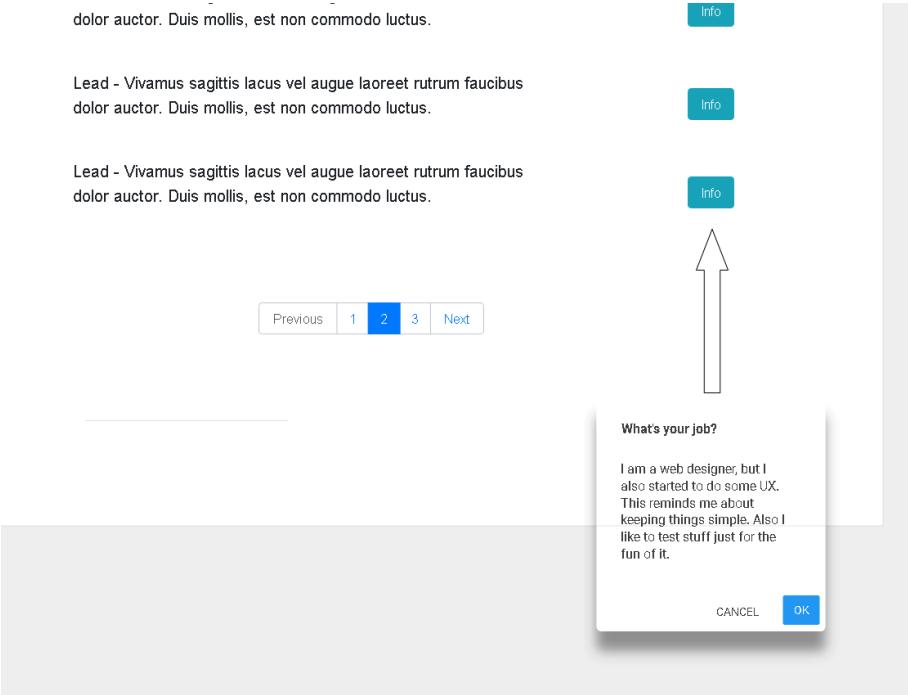


The screenshot displays a user interface component with the following elements:

- A top navigation bar with a "Search" button, two checkboxes labeled "Active custom checkbox" and "Indeterminate custom checkbox", and a "Results" button with a "4" notification.
- A dropdown menu titled "Open this select menu" containing three items: "One", "Two", and "Three".
- A search bar with a placeholder "Search" and a "Cancel" button.
- A section titled "Results" containing four entries, each consisting of a lead text and an "Info" button.

The lead text for each entry is:

- Lead - Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.
- Lead - Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.
- Lead - Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.
- Lead - Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.



SPARQL Queries

È riportata qua sotto una lista delle query che sono state sviluppate.

- 1) Qual è il titolo dei videogiochi che appartengono ad una saga?

SPARQL Query & Update ?

UnnamedQuery 1Query 2Query 3⊕

```
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/OntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
4
5 select ?titolo where {
6   ?game a wd:Game .
7   ?saga a :Saga .
8   ?saga odp:hasItem ?game .
9   ?game :name ?titolo .
10  filter(lang(?titolo) = "en")
11 } limit 100
12
```

TableRaw ResponsePivot TableGoogle Chart

#	
1	"Rocket League"@en
2	"FIFA 2020"@en
3	"FIFA 2021"@en
4	"FIFA 2022"@en
5	"FIFA 2023"@en
6	"FIFA 2024"@en
7	"Rocket League"@"en
8	"Rocket Racing"@en
9	"Supersonic Acrobatic Rocket Powered Battle Cars"@en

Per ogni videogioco che appartiene ad una qualche saga, lo si estrae e si restituisce il nome in inglese.

- 2) Di che saga fa parte un videogioco?

SPARQL Query & Update (i)

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

- Query Tab:** Unnamed, Query 1 (selected), Query 2, Query 3.
- Query Text:**

```
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/0ntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
4 select ?seriesName where {
5   ?g a wd:Game .
6   ?series a :Saga .
7   ?series odp:hasItem ?g .
8   ?series :name ?seriesName .
9   ?g :name ?gName .
10  filter (lang(?seriesName) = "en") .
11  filter (lang(?gName) = "en") .
12  filter (str(?gName) = "FIFA 2020")
13 } limit 100
```
- Result Options:** Table (selected), Raw Response, Pivot Table, Google Chart.
- Filter:** Filter query results.
- Results Table:**

1	"The FIFA Saga"@en

Si parte dal nome di un videogioco e si trova la saga di appartenenza di tale videogioco. In questo esempio è mostrato il gioco FIFA 2020.

- 3) Qual è lo studio di un videogioco?

SPARQL Query & Update ?

Unnamed × Query 1 × Query 2 × Query 3 × ⊕

```
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/0ntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
4
5 select ?studioName where {
6   ?g a wd:Game .
7   ?g :name ?gName .
8   filter (lang(?gName) = "en") .
9   filter (str(?gName) = "Rocket League") .
10  ?g :madeBy ?studio .
11  ?studio :name ?studioName .
12  filter (lang(?studioName) = "it")
13 } limit 100 |
```

Table Raw Response Pivot Table Google Chart

Filter query results

1	"Psyonix"@it

Si parte dal nome di un videogioco e si trova il nome dello studio che lo ha creato. Nell'esempio la lingua di ritorno è quella italiana

- 4) Quanti sono i videogiochi di una particolare saga?

SPARQL Query & Update [?](#)

Unnamed X [+](#)

```

1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/0ntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
4
5 select (COUNT(?titles) as ?count) where {
6     ?series a :Saga .
7     ?series :name ?seriesName .
8     filter (lang(?seriesName) = "en") .
9     #filter (str(?seriesName) = "The FIFA Saga")
10    filter (str(?seriesName) = "The Rocket League Saga") .
11    ?series odp:hasItem ?titles
12 }
13 limit 100

```

Table Raw Response Pivot Table Google Chart

Filter query results

1	"3"	xsd:integer
---	-----	-------------

Si parte dal nome di una saga e si restituisce il COUNT dei titoli che ne fanno parte.

- 5) Qual è la data di fondazione di uno studio?

SPARQL Query & Update [?](#)

Unnamed X [Query 5 X](#) [+](#)

```

1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/0ntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX wdt: <http://www.wikidata.org/prop/direct/>
4 PREFIX wikibase: <http://wikiba.se/ontology#>
5 PREFIX bd: <http://www.bigdata.com/rdf#>
6 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
7
8 select ?studioDate where {
9     ?studio a :Studio .
10    ?studio :name ?studioName .
11    filter (lang(?studioName) = "en") .
12    filter (str(?studioName) = "Psyonix") .
13    SERVICE <https://query.wikidata.org/sparql> {
14        ?studioWD wdt:P31 wd:Q210167 .
15        ?studioWD rdfs:label ?studioWDLLabel .
16        filter (lang(?studioWDLLabel) = "en") .
17        filter( str(?studioName) = str(?studioWDLLabel)) .
18        ?studioWD wdt:P571 ?studioDate
19    }
20 }
21 limit 100

```

SPARQL Query & Update [?](#)

Table Raw Response Pivot Table Google Chart

Filter query results

1	"2000-01-01T00:00:00Z"^^xsd:dateTime
---	--------------------------------------

Si parte dal nome di uno studio e si fa la query federata per trovare uno “studio di wikidata” e se ne estrae la data di fondazione.

- 6) Qual é il protagonista di un certo videogioco?

SPARQL Query & Update [?](#)

Unnamed × Query 6 × [⊕](#)

```
PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/OntoGames_Ontology/>
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>

select ?protagonist where {
  ?game a wd:Game .
  ?game :name ?gameName .
  filter (str(?gameName) = "Martha is Dead") .
  filter (lang(?gameName) = "en") .
  ?protagonist :isProtagonistOf ?game
} limit 100
```

Table Raw Response Pivot Table Google Chart

Filter query results

1	OntoGames_Ontology:Giulia
---	---------------------------

Si parte dal nome di un videogioco e si trova un personaggio che é protagonista del gioco stesso.

- 7) Qual é l'ordine cronologico dei videogiochi di una saga?

SPARQL Query & Update i

```

Query 1 × Query 7 × Query 2 × Query 3 × Query 4 × Query 5 × Query 6 × +
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/OntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX wdt: <http://www.wikidata.org/prop/direct/>
4 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
5 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl#>
6
7 select ?game ?gameWDDate where {
8   ?series a :Saga .
9   ?series :name ?seriesName .
10  filter (lang(?seriesName) = "en") .
11  filter (str(?seriesName) = "The Rocket League Saga") .
12  ?game a wd:Game .
13  ?series odp:hasItem ?game .
14  ?game :name ?gameName .
15  filter (lang(?gameName) = "en") .
16  SERVICE <https://query.wikidata.org/sparql> {
17    #Find the game on wikidata
18    ?gameWD wdt:P31 wd:Q7889 .
19    ?gameWD rdfs:label ?gameWDLLabel .
20    filter (lang(?gameWDLLabel) = "en") .
21    filter( str(?gameWDLLabel) = str(?gameName)) .
22    #Get the game date
23    ?gameWD wdt:P577 ?gameWDDate
24  }
25
26 } order by ?gameWDDate
27 limit 100

```

SPARQL Query & Update i

Table Raw Response Print Table Google Chart

Filter query results

Showing results from 1 to 3 of 3. Query took 1m 7s, 1 minute ago.

game	gameWDDate
1 OntoGames_Ontology_SonicAcrobaticRocketPoweredBattleCars	"2008-10-09T00:00:00Z"^^xsd:dateTime
2 OntoGames_Ontology_RocketLeague	"2015-07-07T00:00:00Z"^^xsd:dateTime
3 OntoGames_Ontology_RocketLeague	"2016-02-17T00:00:00Z"^^xsd:dateTime

Si parte dal nome di una saga, si prendono tutti i suoi giochi e per ognuno di essi si trova la data di pubblicazione in maniera federata da wikidata e la si restituisce insieme al gioco stesso.

- 8) Tutti i titoli con protagonista femminile.

SPARQL Query & Update ⓘ

```
Unnamed × ⓘ
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/OntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl>
4
5 select ?game where {
6   ?game a :Female_Protagonist_Game
7 } limit 100
8
```

Table Raw Response Pivot Table Google Chart

Filter query results

	game
1	OntoGames_Ontology:MarthalIsDead

Si restituiscono tutti i titoli con protagonista femminile.

- 9) Tutti i titoli con storie d'amore contraccambiate.

SPARQL Query & Update ⓘ

```
Unnamed × ⓘ
1 PREFIX : <http://www.semanticweb.org/cava/ontologies/2023/11/OntoGames_Ontology/>
2 PREFIX wd: <http://www.wikidata.org/entity/>
3 PREFIX odp: <http://www.ontologydesignpatterns.org/cp/owl/bag.owl>
4
5 select DISTINCT ?game where {
6   ?game a wd:Game .
7   ?t a :Character .
8   ?s a :Character .
9   ?s :loves ?t .
10  ?t :loves ?s .
11  ?t :appearsIn ?game .
12  ?s :appearsIn ?game
13 } limit 100
14
```

Table Raw Response Pivot Table Google Chart

Filter query results

	game
1	OntoGames_Ontology:MarthalIsDead

Si trovano tutti i personaggi che si amano a vicenda e si restituisce il gioco in cui entrambi compaiono.

Estensioni

Applicazione Client

L'applicazione client è nella cartella 'OntoGames-frontend' e la si può eseguire in locale coi seguenti comandi.

Dipendenze

- NodeJs (npm)
- GraphDb

Far partire in locale

```
npm install  
npm start
```

Il frontend sarà disponibile alla porta :4200 (porta base di Angular) Il frontend per poter funzionare ha bisogno di GraphDb. Questo può essere installato in locale oppure tramite Docker. Il docker file è nella root folder della repo.

Semplicemente, per far partire GraphDb:

```
docker compose up -d
```

e

```
docker compose down
```

per fermarlo.

La GUI di GraphDb sarà disponibile alla porta :7200 (porta base di GraphDb)

Regole SWRL

Abbiamo creato 5 regole SWRL per definire concetti di supporto per l'ontologia:

1) Adult: Un personaggio di età maggiore di 18 anni

```
Character(?c) ^ age(?c, ?cage) ^ swrlb:greaterThanOrEqual(?cage, 18) -> Adult(?c)
```

2) Card Game: Un gioco che permette di fare partite di carte.

```
autogen0:Game(?g) ^ Card(?c) ^ isAssetOf(?c, ?g) -> Card_Game(?g)
```

3) Minor: Un personaggio di età minore di 18 anni

```
Character(?c) ^ age(?c, ?cage) ^ swrlb:lessThan(?cage, 18) -> Minor(?c)
```

4) MutuallyLoves: Relazione tra personaggi che si amano a vicenda.

```
loves(?x, ?z) ^ loves(?z, ?x) -> mutuallyLoves(?x, ?z)
```

- 5) RPG: (Role Playing Game) Un gioco di ruolo in cui ci sono personaggi non umani

```
autogen0:Game(?g) ^ Character(?c) ^ appearsIn(?c, ?g) ^ class(?c, ?cclass) ^  
swrlb:notEqual(?cclass, "Human") -> RPG(?g)
```