Errori comuni riscontrati durante le correzioni dei compiti di Gestione dei Dati e della Conoscenza

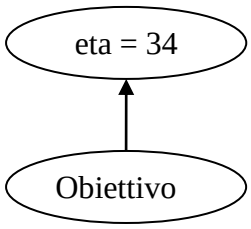
Premessa:

Proprio a causa della loro "forte presenza" nei compiti, gli errori o le imprecisioni più comuni non hanno avuto un peso rilevante sulla valutazione del compito (questo vale sempre per le imprecisioni, dipende invece, caso per caso, per gli errori)

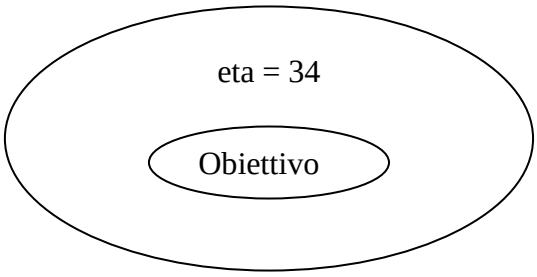
Esercizio 7 tra quelli forniti agli studenti:

```
<owl:Class rdf:ID="Obiettivo">
    <owl:subClassOf>
        <owl:Restriction>
            <owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#int">34</owl:hasValue>
            <owl:onProperty rdf:resource="#eta"/>
        </owl:Restriction>
    </owl:subClassOf>
</owl:Class>
```

La classe Obiettivo è dichiarata subClassOf della classe rappresentata da tutti gli individui che hanno 34 anni.



o, da un punto di vista insiemistico:



Sapendo che una persona ha 34 anni, non posso quindi stabilire (a differenza di come riportato da molti) se essa rientra nella classe obiettivo.

Espressioni description logics

Molte volte non si capisce il criterio con cui sono "assemblate".

Ricordiamo che, tutte le seguenti espressioni sono delle classi:

[≥≤=] *numintero prop*
∃*prop Classe*
∀*prop Classe*
prop ∋ value
¬ *Classe,*
Classe1 ⊓ *Classe2,*
Classe1 ⊔ *Classe2*

Altre costruzioni sono, prima ancora di contenere qualche problema a livello semantico, sintatticamente scorrette.

Verso delle proprietà

Molte volte si confonde il senso di una proprietà. *padre* cosa vuole significare? Ha come range il padre di una persona, o ha come range il figlio di una persona che è quindi intesa come padre della stessa?

Il più delle volte, il nome di una proprietà si deve intendere come *ha_nomeprop*, ovvero: padre punterebbe al padre (oggetto) della persona che stiamo descrivendo (soggetto).

Considerate che, in molti casi, il vero significato può essere compreso studiando il range e il dominio della proprietà stessa o anche di sue sotto o super proprietà. Ad esempio, la proprietà padre, se ha come range uomo e come dominio persona, ha una chiara interpretazione (quella errata verrebbe contraddetta esattamente a livello logico, qualora venisse applicata, perché troverei dei casi nella Assertion Box che contraddirebbero la teoria nella Terminology Box).

SPARQL

Supponiamo di voler recuperare l'insieme delle persone che amano e odiano Roma (inteso come un oggetto rappresentato dalla istanza :Roma).

Perché fare cose come:

```
?a  :ama  :Roma
?b  :odia  :Roma
FILTER (?a == ?b)
```

??

L'*unificazione* delle variabili ci garantisce una dichiarazione molto più semplice:

```
?a  :ama  :Roma
?a  :odia  :Roma
```

OPTIONAL E BOUND

Non abusate poi degli OPTIONAL e BOUND, quando non servono. Il pattern mostrato per implementare la negation-as-failure, serve solo per quella. Allo stesso tempo, la semplice OPTIONAL serve in tutti i casi in cui non si vuol far fallire l'unificazione di un pattern, anche quando un determinato valore possa non essere istanziato.

Dichiarazione di appartenenza ad una classe

Supponiamo che la prop :ama sia definita avere come domain :Person

Volendo definire tutte le persone che amano Mary, non vi è necessità, nella query, di eseguire il check della appartenenza degli innamorati di Mary, ad essere delle persone. Assumendo che i dati siano scritti in modo coerente con il modello del mondo (il vocabolario, o la Terminology Box, se volete), verificare che tali persone, siano per l'appunto delle :Person, è ridondante.

Perciò:

```
?x      a       :Person
?x      :ama    :Mary
```

È ridondante e meno performante

Meglio eseguire semplicemente:

```
?x      :ama    :Mary
```

E dal dominio di :ama, definito prima, siamo sicuri che le sostituzioni delle ?x siano delle :Person